aws

User Guide

AWS Batch

AWS Batch: User Guide

Table of Contents

Was ist AWS Batch?

Mit AWS Batch können Sie Batch-Verarbeitungs-Workloads in der AWS Cloud ausführen. Batch Computing ist eine gängige Methode für Entwickler, Forscher und Techniker, um auf große Mengen an Rechenressourcen zuzugreifen. AWS Batch entfernt die undifferenzierte, schwerste Arbeit bei der Konfiguration und Verwaltung der erforderlichen Infrastruktur, ähnlich wie herkömmliche Batch-Computing-Software. Mit diesem Service können Sie Ressourcen als Reaktion auf gesendete Aufträge effizient bereitstellen, um Kapazitätsbeschränkungen aufzuheben, IT-Kosten zu reduzieren und Ergebnisse schnell bereitzustellen.

Als vollständig verwalteter Service AWS Batch unterstützt Sie bei der Ausführung von Batch-Computing-Workloads jeder Größenordnung. stellt AWS Batch automatisch Rechenressourcen bereit und optimiert die Workload-Verteilung basierend auf der Menge und Skalierung der Workloads. Mit müssen Sie Batch-Computing-Software AWS Batchnicht installieren oder verwalten, sodass Sie sich auf die Analyse von Ergebnissen und die Lösung von Problemen konzentrieren können.

Themen

Komponenten im AWS Batch

AWS Batch vereinfacht die Ausführung von Batch-Aufträgen über mehrere Availability Zones innerhalb einer Region hinweg. Sie können AWS Batch-Datenverarbeitungsumgebungen in einer neuen oder bestehenden VPC erstellen. Nachdem eine Datenverarbeitungsumgebung erstellt und einer Auftragswarteschlange zugewiesen wurde, können Sie Auftragsdefinitionen definieren, die Docker-Containerabbilder für die Ausführung Ihrer Aufträge festzulegen. Containerabbilder werden in Container-Registrys gespeichert und daraus abgerufen, die sich innerhalb oder außerhalb Ihrer AWS-Infrastruktur befinden können.

Aufträge

Eine Arbeitseinheit (etwa als Shell-Script, eine ausführbare Linux-Datei oder ein Docker-Containerabbild), die Sie an AWS Batch senden. Es hat einen Namen und wird als containerisierte Anwendung auf AWS Fargate oder Amazon EC2-Ressourcen in Ihrer Datenverarbeitungsumgebung

ausgeführt, wobei Parameter verwendet werden, die Sie in einer Auftragsdefinition angeben. Aufträge können auf andere Aufträge nach Name oder ID verweisen und vom erfolgreichen Abschluss anderer Aufträge abhängen. Weitere Informationen finden Sie unter Aufträge.

Auftragsdefinitionen

Eine Auftragsdefinition gibt an, wie Aufträge ausgeführt werden sollen. Sie können sich eine Auftragsdefinition als Vorlage für die Ressourcen in Ihrem Auftrag vorstellen. Sie können Ihrem Auftrag eine IAM-Rolle zur Verfügung stellen, um Zugriff auf andere -AWSRessourcen zu gewähren. Sie geben auch sowohl Speicher- als auch CPU-Anforderungen an. Mit der Auftragsdefinition können zudem Containereigenschaften, Umgebungsvariablen und Bereitstellungspunkte zur persistente Speicherung gesteuert werden. Viele der Spezifikationen in einer Auftragsdefinition können überschrieben werden, indem Sie beim Senden einzelner Aufträge neue Werte angeben. Weitere Informationen finden Sie unter Auftragsdefinitionen.

Auftragswarteschlangen

Wenn Sie einen AWS Batch Auftrag einreichen, senden Sie ihn an eine bestimmte Auftragswarteschlange, in der sich der Auftrag befindet, bis er in einer Datenverarbeitungsumgebung geplant ist. Sie verknüpfen eine oder mehrere Datenverarbeitungsumgebungen mit einer Auftragswarteschlange. Sie können diesen Datenverarbeitungsumgebungen und sogar Auftragswarteschlangen selbst Prioritätswerte zuweisen. Sie können beispielsweise eine Warteschlange mit hoher Priorität haben, an die Sie zeitkritische Aufträge senden, und eine Warteschlange mit niedriger Priorität für Aufträge, die jederzeit ausgeführt werden können, wenn die Rechenressourcen günstiger sind.

Datenverarbeitungsumgebung

Bei einer Datenverarbeitungsumgebung handelt es sich um eine Gruppe verwalteter oder nicht verwalteter Datenverarbeitungsressourcen für die Ausführung von Aufträgen. Mit verwalteten Datenverarbeitungsumgebungen können Sie den gewünschten Datenverarbeitungstyp (Fargate oder EC2) auf mehreren Detailebenen angeben. Sie können Datenverarbeitungsumgebungen einrichten, die einen bestimmten Typ von EC2-Instance verwenden, ein bestimmtes Modell wie `c5.2xlarge` oder `m5.10xlarge`. Oder Sie können nur angeben, dass Sie die neuesten Instance-Typen verwenden möchten. Sie können auch die minimale, gewünschte und maximale Anzahl von vCPUs für die Umgebung angeben, zusammen mit dem Betrag, den Sie für eine Spot-Instance zu zahlen bereit sind, als Prozentsatz des On-Demand-Instance-Preises und eines Zielsatzes von VPC-Subnetzen. AWS Batch startet, verwaltet und beendet Rechentypen effizient nach Bedarf. Sie können

zudem Ihre eigenen Datenverarbeitungsumgebungen verwalten. Daher sind Sie für die Einrichtung und Skalierung der Instances in einem Amazon-ECS-Cluster verantwortlich, den für Sie AWS Batch erstellt. Weitere Informationen finden Sie unter Datenverarbeitungsumgebung.

Erste Schritte

Führen Sie die ersten Schritte mit AWS Batch aus, indem Sie eine Auftragsdefinition, Datenverarbeitungs-Umgebung und eine Auftragswarteschlange in der AWS Batch-Konsole erstellen.

Der AWS Batch Ersteinrichtungsassistent bietet Ihnen die Möglichkeit, eine Datenverarbeitungsumgebung und eine Auftragswarteschlange zu erstellen und einen Hello World-Beispielauftrag zu senden. Wenn Sie bereits über ein Docker-Image verfügen, das Sie in starten möchtenAWS Batch, können Sie eine Auftragsdefinition mit diesem Image erstellen und dieses stattdessen an Ihre Warteschlange senden. Weitere Informationen finden Sie unter Erste Schritte mit AWS Batch.

Dashboard

Auf dem AWS Batch Dashboard können Sie aktuelle Aufträge, Auftragswarteschlangen und Datenverarbeitungsumgebungen überwachen. Standardmäßig werden die folgenden Dashboard-Widgets angezeigt:

- Auftragsübersicht – Weitere Informationen zu AWS Batch Aufträgen finden Sie unter Aufträge.
- Übersicht über die Auftragswarteschlange – Weitere Informationen zu AWS Batch Auftragswarteschlangen finden Sie unter Warteschlangen für Job.
- Übersicht über die Datenverarbeitungsumgebung – Weitere Informationen zu AWS Batch Datenverarbeitungsumgebungen finden Sie unter Datenverarbeitungsumgebung.

Sie können die Widgets anpassen, die auf der Dashboard-Seite angezeigt werden. In den folgenden Abschnitten werden zusätzliche Widgets beschrieben, die Sie installieren können.

Einzelne Auftragswarteschlange

Dieses Widget zeigt detaillierte Informationen zu einer einzelnen Auftragswarteschlange an.

Gehen Sie folgendermaßen vor, um dieses Widget hinzuzufügen.

1. Öffnen Sie die AWS Batch-Konsole.

2. Wählen Sie in der Navigationsleiste die aus, AWS-Region die Sie möchten.
3. Wählen Sie im Navigationsbereich Dashboard (Dashboard).
4. Wählen Sie Widgets hinzufügen aus.
5. Wählen Sie für Einzelne Auftragswarteschlange die Option Widget hinzufügen aus.
6. Wählen Sie für Auftragswarteschlange die gewünschte Auftragswarteschlange aus.
7. Wählen Sie für Auftragsstatus die Auftragsstatus aus, die Sie anzeigen möchten.
8. (Optional) Deaktivieren Sie Verbundene Rechenumgebungen anzeigen, wenn Sie die Eigenschaften für Rechenumgebungen nicht anzeigen möchten.
9. Wählen Sie unter Eigenschaften der Datenverarbeitungsumgebung die gewünschten Eigenschaften aus.
10. Wählen Sie Hinzufügen aus.

CloudWatch Container Insights

Dieses Widget zeigt aggregierte Metriken für AWS Batch Datenverarbeitungsumgebungen und Aufträge an. Weitere Informationen zu Container Insights finden Sie unter CloudWatch Container Insights.

Gehen Sie folgendermaßen vor, um dieses Widget hinzuzufügen.

1. Öffnen Sie die AWS Batch-Konsole.
2. Wählen Sie in der Navigationsleiste die aus, AWS-Region die Sie möchten.
3. Wählen Sie im Navigationsbereich Dashboard (Dashboard).
4. Wählen Sie Widgets hinzufügen aus.
5. Wählen Sie für Container Insights die Option Widget hinzufügen aus.
6. Wählen Sie für Datenverarbeitungsumgebung die gewünschte Datenverarbeitungsumgebung aus.
7. Wählen Sie Hinzufügen aus.

Auftragsprotokolle

Dieses Widget zeigt verschiedene Protokolle Ihrer Aufträge an einem praktischen Ort an. Weitere Informationen zu Auftragsprotokollen finden Sie unter the section called “ Job-Logs”.

Gehen Sie folgendermaßen vor, um dieses Widget hinzuzufügen.

1. Öffnen Sie die AWS Batch-Konsole.
2. Wählen Sie in der Navigationsleiste die aus, AWS-Region die Sie möchten.
3. Wählen Sie im Navigationsbereich Dashboard (Dashboard).
4. Wählen Sie Widgets hinzufügen aus.
5. Wählen Sie für Auftragsprotokolle die Option Widget hinzufügen aus.
6. Geben Sie für Auftrags-ID die Auftrags-ID für den gewünschten Auftrag ein.
7. Wählen Sie Hinzufügen aus.

Einrichtung mit AWS Batch

Wenn Sie sich bereits für Amazon Web Services (AWS) registriert haben und Amazon Elastic Compute Cloud (Amazon EC2) oder Amazon Elastic Container Service (Amazon ECS) verwenden, können Sie dies bald nutzen AWS Batch. Der Einrichtungsprozess für diese Dienste ist ähnlich. Dies liegt daran, dass Amazon ECS-Container-Instances in seinen Rechenumgebungen AWS Batch verwendet werden. Um with verwenden zu können AWS Batch , müssen Sie eine Version von verwenden AWS CLI , die die neuesten AWS Batch Funktionen unterstützt. AWS CLI Wenn Sie in der keine Unterstützung für eine AWS Batch Funktion sehen AWS CLI, führen Sie ein Upgrade auf die neueste Version durch. Weitere Informationen finden Sie unter http://aws.amazon.com/cli/.

> **Note**
>
> Da Komponenten von Amazon EC2 AWS Batch verwendet werden, verwenden Sie für viele dieser Schritte die Amazon EC2 EC2-Konsole.

Erledigen Sie die folgenden Aufgaben, um sich darauf vorzubereiten. AWS Batch Wenn Sie einen dieser Schritte bereits abgeschlossen haben, können Sie direkt mit der Installation von fortfahren AWS CLI.

Themen

Melde dich an für ein AWS-Konto

Wenn Sie noch keine haben AWS-Konto, führen Sie die folgenden Schritte aus, um eine zu erstellen.

Um sich für eine anzumelden AWS-Konto

1. Öffnen Sie https://portal.aws.amazon.com/billing/signup.
2. Folgen Sie den Online-Anweisungen.

   Bei der Anmeldung müssen Sie auch einen Telefonanruf entgegennehmen und einen Verifizierungscode über die Telefontasten eingeben.

   Wenn Sie sich für eine anmelden AWS-Konto, Root-Benutzer des AWS-Kontoswird eine erstellt. Der Root-Benutzer hat Zugriff auf alle AWS-Services und Ressourcen des Kontos. Aus Sicherheitsgründen sollten Sie einem Benutzer Administratorzugriff zuweisen und nur den Root-Benutzer verwenden, um Aufgaben auszuführen, für die Root-Benutzerzugriff erforderlich ist.

AWS sendet Ihnen nach Abschluss des Anmeldevorgangs eine Bestätigungs-E-Mail. Sie können jederzeit Ihre aktuelle Kontoaktivität anzeigen und Ihr Konto verwalten. Rufen Sie dazu https://aws.amazon.com/ auf und klicken Sie auf Mein Konto.

Erstellen Sie einen Benutzer mit Administratorzugriff

Nachdem Sie sich für einen angemeldet haben AWS-Konto, sichern Sie Ihren Root-Benutzer des AWS-Kontos AWS IAM Identity Center, aktivieren und erstellen Sie einen Administratorbenutzer, sodass Sie den Root-Benutzer nicht für alltägliche Aufgaben verwenden.

Sichern Sie Ihre Root-Benutzer des AWS-Kontos

1. Melden Sie sich AWS Management Consoleals Kontoinhaber an, indem Sie Root-Benutzer auswählen und Ihre AWS-Konto E-Mail-Adresse eingeben. Geben Sie auf der nächsten Seite Ihr Passwort ein.

   Hilfe bei der Anmeldung mit dem Root-Benutzer finden Sie unter Anmelden als Root-Benutzer im AWS-Anmeldung Benutzerhandbuch zu.
2. Aktivieren Sie die Multi-Faktor-Authentifizierung (MFA) für den Root-Benutzer.

   Anweisungen finden Sie unter Aktivieren eines virtuellen MFA-Geräts für Ihren AWS-Konto Root-Benutzer (Konsole) im IAM-Benutzerhandbuch.

Erstellen Sie einen Benutzer mit Administratorzugriff

1. Aktivieren Sie das IAM Identity Center.

   Anweisungen finden Sie unter Aktivieren AWS IAM Identity Center im AWS IAM Identity Center Benutzerhandbuch.

2. Gewähren Sie einem Benutzer in IAM Identity Center Administratorzugriff.

   Ein Tutorial zur Verwendung von IAM-Identity-Center-Verzeichnis als Identitätsquelle finden Sie unter Benutzerzugriff mit der Standardeinstellung konfigurieren IAM-Identity-Center-Verzeichnis im AWS IAM Identity Center Benutzerhandbuch.

Melden Sie sich als Benutzer mit Administratorzugriff an

- Um sich mit Ihrem IAM-Identity-Center-Benutzer anzumelden, verwenden Sie die Anmelde-URL, die an Ihre E-Mail-Adresse gesendet wurde, als Sie den IAM-Identity-Center-Benutzer erstellt haben.

  Hilfe bei der Anmeldung mit einem IAM Identity Center-Benutzer finden Sie im AWS-Anmeldung Benutzerhandbuch unter Anmeldung beim AWS Zugriffsportal.

Weisen Sie weiteren Benutzern Zugriff zu

1. Erstellen Sie in IAM Identity Center einen Berechtigungssatz, der der bewährten Methode zur Anwendung von Berechtigungen mit den geringsten Rechten folgt.

   Anweisungen finden Sie im Benutzerhandbuch unter Einen Berechtigungssatz erstellen.AWS IAM Identity Center

2. Weisen Sie Benutzer einer Gruppe zu und weisen Sie der Gruppe dann Single Sign-On-Zugriff zu.

   Anweisungen finden Sie im AWS IAM Identity Center Benutzerhandbuch unter Gruppen hinzufügen.

Erstellen Sie IAM-Rollen für Ihre Rechenumgebungen und Container-Instances

Ihre AWS Batch Rechenumgebungen und Container-Instances benötigen AWS-Konto Anmeldeinformationen, um in Ihrem Namen Aufrufe an andere AWS APIs zu tätigen. Erstellen Sie eine IAM-Rolle, die diese Anmeldeinformationen für Ihre Rechenumgebungen und Container-Instances bereitstellt, und ordnen Sie diese Rolle dann Ihren Rechenumgebungen zu.

> **Note**
>
> Die Rollen AWS Batch Rechenumgebung und Container-Instance werden bei der ersten Ausführung der Konsole automatisch für Sie erstellt. Wenn Sie also beabsichtigen, die AWS Batch Konsole zu verwenden, können Sie mit dem nächsten Abschnitt fortfahren. Wenn Sie AWS CLI stattdessen die verwenden möchten, führen Sie die Verfahren in Verwenden von serviceverknüpften Rollen für AWS Batch und Amazon-ECS-Instance-Rolle vor dem Erstellen Ihrer ersten Rechenumgebung durch.

Erstellen eines Schlüsselpaares

AWS verwendet Public-Key-Kryptografie, um die Anmeldeinformationen für Ihre Instance zu sichern. Eine Linux-Instance, z. B. eine AWS Batch Compute-Environment-Container-Instance, hat kein Passwort, das für den SSH-Zugriff verwendet werden könnte. Sie verwenden ein Schlüsselpaar, um sich sicher an Ihrer Instance anzumelden. Sie geben beim Erstellen Ihrer Datenverarbeitungsumgebung den Namen des Schlüsselpaares an und stellen dann beim Anmelden über SSH den privaten Schlüssel bereit.

Wenn Sie noch kein key pair erstellt haben, können Sie eines mit der Amazon EC2 EC2-Konsole erstellen. Beachten Sie, dass Sie, wenn Sie planen, Instances in mehreren zu starten AWS-Regionen, in jeder Region ein key pair erstellen müssen. Weitere Informationen zu Regionen finden Sie unter Regionen und Availability Zones im Amazon EC2 EC2-Benutzerhandbuch.

So erstellen Sie ein Schlüsselpaar

1. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
2. Wählen Sie in der Navigationsleiste eine AWS-Region für das key pair aus. Sie können unabhängig von Ihrem Standort jede Region auswählen, die Ihnen zur Verfügung steht.

Schlüsselpaare sind jedoch regionsspezifisch. Wenn Sie beispielsweise planen, eine Instance in der Region USA West (Oregon) zu starten, erstellen Sie ein key pair für die Instance in derselben Region.

3. Wählen Sie im Navigationsbereich Key Pairs (Schlüsselpaare) und Create Key Pair (Schlüsselpaar erstellen) aus.
4. Geben Sie im Dialogfeld Create Key Pair (Schlüsselpaar erstellen) im Feld Key pair name (Name des Schlüsselpaars) einen Namen für das neue Schlüsselpaar ein und klicken Sie auf Erstellen. Wählen Sie einen Namen, an den Sie sich erinnern können, z. B. Ihren Benutzernamen, gefolgt von `-key-pair`, plus dem Namen der Region. Beispiel: me-key-pair-uswest2.
5. Die private Schlüsseldatei wird von Ihrem Browser automatisch runtergeladen. Der Basisdateiname ist der Name, den Sie als Namen für Ihr Schlüsselpaar festgelegt haben, und die Dateinamenerweiterung lautet `.pem`. Speichern Sie die Datei mit dem privaten Schlüssel an einem sicheren Ort.

   > **Important**
   >
   > Dies ist die einzige Möglichkeit, die private Schlüsseldatei zu speichern. Sie müssen den Namen Ihres key pair angeben, wenn Sie eine Instance starten, und den entsprechenden privaten Schlüssel jedes Mal, wenn Sie sich mit der Instance verbinden.

6. Wenn Sie einen SSH-Client auf einem Mac- oder Linux-Computer verwenden, um eine Verbindung zu Ihrer Linux-Instance herzustellen, verwenden Sie den folgenden Befehl, um die Berechtigungen für Ihre private Schlüsseldatei festzulegen. Auf diese Weise können nur Sie es lesen.

   ```
   $ chmod 400 your_user_name-key-pair-region_name.pem
   ```

Weitere Informationen finden Sie unter Amazon EC2 EC2-Schlüsselpaare im Amazon EC2 EC2-Benutzerhandbuch.

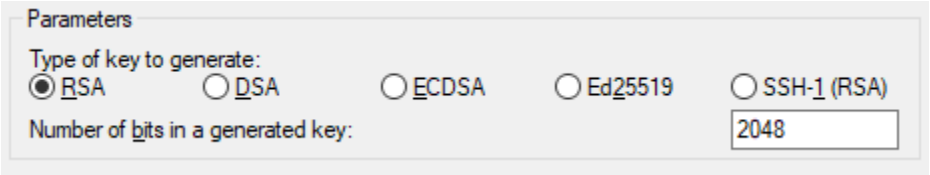
So stellen Sie über Ihr Schlüsselpaar eine Verbindung zu Ihrer Instance her

Wenn Sie von einem Computer mit Mac oder Linux eine Verbindung zu Ihrer Linux-Instance herstellen möchten, geben Sie die `.pem`-Datei für Ihren SSH-Client mit der Option `-i` und dem Pfad zu Ihrem privaten Schlüssel an. Um von einem Computer unter Windows aus eine Verbindung zu Ihrer Linux-Instance herzustellen, verwenden Sie entweder MindTerm oder PuTTY. Wenn Sie PuTTY

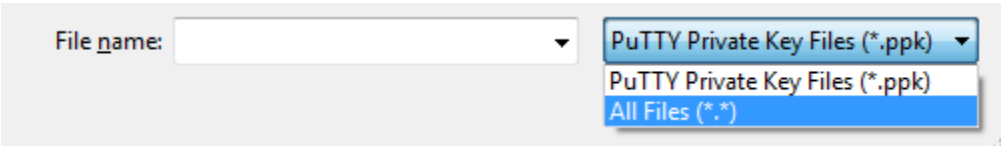
verwenden möchten, installieren Sie es und konvertieren Sie die `.pem` Datei wie folgt in eine `.ppk` Datei.

(Optional) So bereiten Sie die Verbindung zu einer Linux-Instanz von Windows aus mithilfe von PuTTY vor

1. Laden Sie PuTTY unter http://www.chiark.greenend.org.uk/~sgtatham/putty/ herunter und installieren Sie die Anwendung. Vergewissern Sie sich, dass Sie die gesamte Suite installieren.
2. Starten Sie PuTTYgen (wählen Sie beispielsweise im Startmenü Alle Programme, PuTTY und PuTTYgen).
3. Wählen Sie unter Type of key to generate die Option RSA. Wenn Sie eine frühere Version von PuTTYgen verwenden, wählen Sie SSH-2 RSA.



4. Wählen Sie Load (Laden) aus. PuTTYgen zeigt standardmäßig nur Dateien mit der Erweiterung `.ppk` an. Damit Sie die `.pem`-Datei finden, wählen Sie die Option zur Anzeige aller Dateitypen aus.



5. Wählen Sie die Datei mit dem privaten Schlüssel aus, die Sie im vorherigen Schritt erstellt haben, und klicken Sie auf Open. Klicken Sie auf OK, um das Bestätigungsdialogfeld zu verlassen.
6. Wählen Sie Save private key (Privaten Schlüssel speichern) aus. PuTTYgen zeigt einen Warnhinweis zur Speicherung des Schlüssels ohne Passphrase an. Wählen Sie Yes (Ja).
7. Geben Sie für den Schlüssel denselben Namen an wie für das Schlüsselpaar. PuTTY fügt automatisch die Dateierweiterung `.ppk` hinzu.

Erstellen einer VPC

Mit Amazon Virtual Private Cloud (Amazon VPC) können Sie AWS Ressourcen in einem von Ihnen definierten virtuellen Netzwerk starten. Wir empfehlen dringend, dass Sie Ihre Container-Instances in einer VPC starten.

Wenn Sie eine Standard-VPC haben, können Sie diesen Abschnitt auch überspringen und mit der nächsten Aufgabe Eine Sicherheitsgruppe erstellen fortfahren. Informationen darüber, ob Sie über eine Standard-VPC verfügen, finden Sie unter Unterstützte Plattformen in der Amazon EC2 EC2-Konsole im Amazon EC2 EC2-Benutzerhandbuch.

Informationen zum Erstellen einer Amazon VPC finden Sie unter Nur VPC erstellen im Amazon VPC-Benutzerhandbuch. In der folgenden Tabelle können Sie ermitteln, welche Optionen Sie auswählen müssen.

Option	Wert
Zu erstellende Ressourcen	Nur VPC
Name	Geben Sie optional einen Namen für Ihre VPC ein.
IPv4 CIDR block (IPv4-CIDR-Block)	Manuelle IPv4-CIDR-Eingabe Die CIDR-Blockgröße muss eine Größe zwischen /16 und /28. haben.
IPv6-CIDR-Block	Kein IPv6-CIDR-Block
Tenancy	Standard

Weitere Informationen über Amazon VPC finden Sie unter Was ist Amazon VPC? im Amazon VPC-Benutzerhandbuch.

Eine Sicherheitsgruppe erstellen

Sicherheitsgruppen fungieren als Firewall für zugehörige Container-Instances in der Datenverarbeitungsumgebung. Sie kontrollieren den ein- und ausgehenden Datenverkehr auf Container-Instance-Ebene. Eine Sicherheitsgruppe kann nur in der VPC verwendet werden, für die sie erstellt wird.

Sie können einer Sicherheitsgruppe Regeln hinzufügen, die es Ihnen ermöglichen, von Ihrer IP-Adresse über SSH eine Verbindung zu Ihrer Container-Instance herzustellen. Sie können auch

Regeln hinzufügen, die den ein- und ausgehenden HTTP- und HTTPS-Zugriff von überall zulassen. Fügen Sie alle Regeln zum Öffnen von Ports hinzu, die für Ihre Aufgaben benötigt werden.

Beachten Sie, dass Sie, wenn Sie Container-Instances in mehreren Regionen starten möchten, in jeder Region eine Sicherheitsgruppe erstellen müssen. Weitere Informationen finden Sie unter Regionen und Availability Zones im Amazon EC2 EC2-Benutzerhandbuch.

> **Note**
>
> Sie benötigen die öffentliche IP-Adresse Ihres lokalen Computers, die Sie mithilfe eines Service abrufen können. Wir stellen z. B. folgenden Service bereit: http://checkip.amazonaws.com/ oder https://checkip.amazonaws.com/. Wenn Sie einen anderen Service suchen möchten, der Ihnen Ihre IP-Adresse nennt, verwenden Sie den Suchausdruck "wie ist meine IP-Adresse". Wenn Sie eine Verbindung über einen Internetdienstanbieter (ISP) oder hinter einer Firewall ohne statische IP-Adresse herstellen, ermitteln Sie den Bereich der IP-Adressen, die von Client-Computern verwendet werden.

Erstellen einer Sicherheitsgruppe mithilfe der Konsole

1. Öffnen Sie die Amazon VPC-Konsole unter https://console.aws.amazon.com/vpc/.
2. Wählen Sie im Navigationsbereich Sicherheitsgruppen aus.
3. Wählen Sie Create security group (Sicherheitsgruppe erstellen) aus.
4. Geben Sie einen Namen und eine Beschreibung für die Sicherheitsgruppe ein. Sie können den Namen und die Beschreibung einer Regelgruppe nach der Erstellung nicht mehr ändern.
5. Wählen Sie unter VPC die VPC aus.
6. (Optional) Standardmäßig beginnen neue Sicherheitsgruppen nur mit einer Regel für ausgehenden Datenverkehr, die es erlaubt, dass der gesamte Datenverkehr die Ressource verlässt. Sie müssen Regeln hinzufügen, um eingehenden Datenverkehr zuzulassen oder den ausgehenden Datenverkehr einzuschränken.

   AWS Batch Für Container-Instances müssen keine eingehenden Ports geöffnet sein. Möglicherweise möchten Sie jedoch eine SSH-Regel hinzufügen. Auf diese Weise können Sie sich bei der Container-Instance anmelden und die Container in Jobs mit Docker-Befehlen untersuchen. Wenn Sie möchten, dass Ihre Container-Instance einen Job hostet, der einen Webserver ausführt, können Sie auch Regeln für HTTP hinzufügen. Führen Sie die folgenden Schritte aus, um diese optionale Regeln für die Sicherheitsgruppe hinzuzufügen.

Erstellen Sie auf der Registerkarte Eingehend die folgenden Regeln und wählen Sie Erstellen aus:

- Klicken Sie auf Add Rule (Regel hinzufügen). Wählen Sie für Typ die Option HTTP aus. Wählen Sie unter Quelle die Option Anywhere (Beliebig) (`0.0.0.0/0`) aus.
- Klicken Sie auf Add Rule (Regel hinzufügen). Wählen Sie unter Typ die Option SSH aus. Wählen Sie für Source die Option Custom IP und geben Sie die öffentliche IP-Adresse Ihres Computers oder Netzwerks in der CIDR-Notation (Classless Inter-Domain Routing) an. Wenn Ihr Unternehmen Adressen aus einem Bereich zuweist, geben Sie den gesamten Bereich an, z. B. `203.0.113.0/24`. Um eine individuelle IP-Adresse in CIDR-Notation anzugeben, wählen Sie Meine IP aus. Dadurch wird das Routing-Präfix `/32` zur öffentlichen IP-Adresse hinzugefügt.

  > **Note**
  >
  > Aus Sicherheitsgründen empfehlen wir nicht, SSH-Zugriff von allen IP-Adressen (`0.0.0.0/0`) auf Ihre Instance zuzulassen, sondern nur zu Testzwecken und nur für kurze Zeit.

7. Sie können Tags (Markierungen) jetzt erstellen oder zu einem späteren Zeitpunkt hinzufügen. Um eine Markierung hinzuzufügen, wählen Sie Add Tags (Tags (Markierung) hinzufügen) und geben Sie dann den Markierungsschlüssel und -Wert ein.
8. Wählen Sie Sicherheitsgruppe erstellen aus.

Informationen zum Erstellen einer Sicherheitsgruppe über die Befehlszeile finden Sie unter create-security-group ()AWS CLI

Weitere Informationen zu Sicherheitsgruppen finden Sie unter Arbeiten mit Sicherheitsgruppen.

Installieren Sie das AWS CLI

Um das AWS CLI mit zu verwenden AWS Batch, installieren Sie die neueste AWS CLI Version. Informationen zur Installation AWS CLI oder zum Upgrade auf die neueste Version finden Sie unter Installation der AWS Befehlszeilenschnittstelle im AWS Command Line Interface Benutzerhandbuch.

Erste Schritte mit AWS Batch

Sie können den AWS Batch Ersteinrichtungsassistenten verwenden, um schnell mit zu beginnen AWS Batch. Nachdem Sie die Voraussetzungen erfüllt haben, können Sie den Assistenten für die erste Ausführung verwenden, um eine Datenverarbeitungsumgebung, eine Auftragsdefinition und eine Auftragswarteschlange zu erstellen.

Sie können auch einen Beispielauftrag „Hello World“ mit dem Assistenten für die AWS Batch Erstausführung einreichen, um Ihre Konfiguration zu testen. Wenn Sie bereits über ein Docker-Image verfügen, das Sie in starten möchten AWS Batch, können Sie dieses Image verwenden, um eine Auftragsdefinition zu erstellen.

Voraussetzungen

Stellen Sie sicher, dass Sie Folgendes tun, bevor Sie den AWS Batch Ersteinrichtungsassistenten starten:

- Führen Sie die unter beschriebenen Schritte ausEinrichtung mit AWS Batch.
- Stellen Sie sicher, dass Ihr über die erforderlichen Berechtigungen AWS-Konto verfügt.

Erste Schritte – Amazon EC2

Amazon Elastic Compute Cloud (Amazon EC2) – Bietet sichere und skalierbare Rechenkapazität in der AWS Cloud. Amazon EC2 beseitigt die Notwendigkeit, im Voraus in Hardware investieren zu müssen. Daher können Sie Anwendungen schneller entwickeln und bereitstellen.

Mit Amazon EC2 können Sie so viele oder so wenige virtuelle Server starten, wie Sie benötigen, die Sicherheit und das Netzwerk konfigurieren und den Speicher verwalten. Amazon EC2 ermöglicht Ihnen, hoch- und runterzuskalieren, um Anforderungsänderungen oder Nutzungsspitzen zu bewältigen, was Ihren Bedarf an Traffic-Prognosen reduziert.

Erstellen einer Datenverarbeitungsumgebung

Gehen Sie wie folgt vor, um eine Datenverarbeitungsumgebung für eine Amazon EC2-Orchestrierung zu erstellen:

1. Öffnen Sie den AWS Batch Assistenten für die Erstausführung der -Konsole.
2. Wählen Sie für Orchestrierungstyp auswählen die Option Amazon Elastic Compute Cloud (Amazon EC2) aus.
3. Wählen Sie Weiter aus.
4. Geben Sie im Abschnitt Konfiguration der Datenverarbeitungsumgebung für Name einen eindeutigen Namen für Ihre Datenverarbeitungsumgebung an. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
5. Wählen Sie für Instance-Rolle ein vorhandenes Instance-Profil aus, dem die erforderlichen IAM-Berechtigungen zugeordnet sind. Dieses Instance-Profil ermöglicht es den Amazon-ECS-Container-Instances in Ihrer Computing-Umgebung, Aufrufe an die erforderlichen AWS API-Operationen zu tätigen. Weitere Informationen finden Sie unter Amazon-ECS-Instance-Rolle.
6. (Optional) Ein Tag ist eine Bezeichnung, die einer Ressource zugewiesen ist. Um ein Tag oder ein Amazon EC2-Tag hinzuzufügen, erweitern Sie Tags und wählen Sie dann Tag hinzufügen aus. Geben Sie ein Schlüssel-Wert-Paar ein und wählen Sie dann erneut Tag hinzufügen aus.

   > **Important**
   >
   > Wenn Sie Tag hinzufügen auswählen, müssen Sie ein Schlüssel-Wert-Paar eingeben und erneut Tag hinzufügen oder Tag entfernen auswählen.

7. (Optional) Aktivieren Sie im Abschnitt Instance-Konfiguration für Verwenden von Amazon EC2-Spot-Instances die Option Aktivieren mit Spot-Instances.
8. (Nur Spot) Geben Sie für Maximaler On-Demand-Preis in % den maximalen Prozentsatz der On-Demand-Preise ein, den Sie für Spot-Ressourcen zahlen möchten.
9. (Optional) (Nur Spot) Wählen Sie für Spot-Flottenrolle eine vorhandene Amazon EC2 Spot-Flotten-IAM-Rolle aus, die auf Ihre Spot-Datenverarbeitungsumgebung angewendet werden soll. Wenn Sie noch keine vorhandene IAM-Rolle für Amazon EC2 Spot Fleet haben, müssen Sie zuerst eine erstellen. Weitere Informationen finden Sie unter Amazon EC2-Spot-Flottenrolle.

   > **Important**
   >
   > Um Ihre Spot-Instances bei der Erstellung zu markieren, muss Ihre IAM-Rolle der Amazon EC2-Spot-Flotte die neuere von AmazonEC2SpotFleetTaggingRole verwaltete Richtlinie verwenden. Die von AmazonEC2SpotFleetRole verwaltete Richtlinie verfügt nicht über die erforderlichen Berechtigungen zum Markieren von Spot-Instances. Weitere

Informationen finden Sie unter Spot-Instances wurden bei der Erstellung nicht markiert und the section called “Markieren Ihrer -Ressourcen”.

10. Wählen Sie für Minimale vCPUs die Mindestanzahl von EC2-vCPUs aus, die Ihre Datenverarbeitungsumgebung verwaltet, unabhängig vom Bedarf an Auftragswarteschlangen.
11. Wählen Sie für Gewünschte vCPUs die Anzahl der EC2 vCPUs aus, mit denen Ihre Datenverarbeitungsumgebung gestartet wird. Wenn der Bedarf an Auftragswarteschlangen steigt, AWS Batch erhöht die gewünschte Anzahl von vCPUs und fügt EC2-Instances hinzu. Die Anzahl der vCPUs kann bis zur maximalen Anzahl von vCPUs erhöht werden. Wenn der Bedarf sinkt, AWS Batch verringert die gewünschte Anzahl von vCPUs und entfernt Instances. Die Anzahl der verringert sich bis auf die Mindestanzahl von vCPUs.
12. Wählen Sie für Maximale Zahl von vCPUs die Höchstanzahl von EC2-vCPUs, auf die Ihre Datenverarbeitungsumgebung unabhängig von den Anforderungen der Auftragswarteschlangen erweitert werden kann.
13. Wählen Sie für Zulässige Instance-Typen die Amazon EC2-Instance-Typen aus, die gestartet werden können. Sie können Instance-Familien angeben, um einen beliebigen Instance-Typ innerhalb dieser Familien zu starten (z. B. `c5`c5n, oder p3). Sie können auch bestimmte Größen innerhalb einer Familie angeben (z. B. `c5.8xlarge`). Metal-Instance-Typen gehören nicht zu den Instance-Familien. Beispielsweise enthält `c5` nicht `c5.metal`. Sie können auch Instance-`optimal`Typen (aus den Instance-R4Familien M4, und C4) auswählen, die den Anforderungen Ihrer Auftragswarteschlangen entsprechen.

> **Note**
>
> Wenn Sie eine Compute-Umgebung erstellen, müssen die Instance-Typen, die Sie für die Compute-Umgebung auswählen, dieselbe Architektur verwenden. Beispielsweise ist es nicht möglich, x86- und ARM-Instances in derselben Compute-Umgebung zu kombinieren.

> **Note**
>
> AWS Batch skaliert GPUs basierend auf der erforderlichen Menge in Ihren Auftragswarteschlangen. Um die GPU-Planung verwenden zu können, muss die Datenverarbeitungsumgebung Instance-Typen aus der p2-, -p3, p4-, -p5, g3s-, g3-g4oder -g5Familie enthalten.

> **Note**
>
> Derzeit `optimal` verwendet Instance-Typen aus den Instance-R4Familien M4, und C4. In AWS-Regionen , die keine Instance-Typen aus diesen Instance-Familien haben, werden Instance-Typen aus den C5InstanceR5-Familien M5, und verwendet.

14. Erweitern Sie Additional configuration (Zusätzliche Konfiguration).
15. (Optional) Geben Sie für Platzierungsgruppe einen Platzierungsgruppennamen ein, um Ressourcen in der Datenverarbeitungsumgebung zu gruppieren.
16. (Optional) Wählen Sie für EC2-Schlüsselpaar ein öffentliches und ein privates Schlüsselpaar als Sicherheitsanmeldeinformationen aus, wenn Sie eine Verbindung mit der Instance herstellen. Weitere Informationen zu Amazon EC2-Schlüsselpaaren finden Sie unter Amazon EC2-Schlüsselpaare und Linux-Instances.
17. Wählen Sie als Allocation strategy (Zuordnungsstrategie) die Zuordnungsstrategie aus, die bei der Auswahl von Instance-Typen aus der Liste der zulässigen Instance-Typen verwendet werden soll. BEST_ Bol_PROGRESSIVE ist in der Regel die bessere Wahl für EC2-On-Demand-Datenverarbeitungsumgebungen und SPOT_CAPACITY_OPTIMIZED für EC2-Spot-Datenverarbeitungsumgebungen. Weitere Informationen finden Sie unter the section called “Zuweisungsstrategien”.
18. (Optional) Wählen Sie für EC2-Konfiguration die Option EC2-Konfiguration hinzufügen aus. Wählen Sie Image-Typ und Image-ID-Überschreibungswerte aus, um Informationen für bereitzustellen AWS Batch , um Amazon Machine Images (AMIs) für Instances in der Datenverarbeitungsumgebung auszuwählen. Wenn die Image-ID-Überschreibung nicht für jeden Image-Typ angegeben ist, AWS Batch wählt ein aktuelles Amazon-ECS-optimiertes AMI aus. Wenn kein Image-Typ angegeben ist, ist der Standardwert eine Amazon Linux 2 für Nicht-GPU- und Nicht- AWS Graviton-Instance.

> **Important**
>
> Um ein benutzerdefiniertes AMI zu verwenden, wählen Sie den Image-Typ aus und geben Sie dann die benutzerdefinierte AMI-ID in das Feld Überschreibung der Image-ID ein.

Amazon Linux 2

Standard für alle AWS Graviton-basierten Instance-Familien (z. B. C6g, M6gR6g, und T4g) und kann für alle Nicht-GPU-Instance-Typen verwendet werden.

Amazon Linux 2 (GPU)

Standard für alle GPU-Instance-Familien (z. B. P4 und G4) und kann für alle nicht AWS auf Graviton basierenden Instance-Typen verwendet werden.

Amazon Linux

Kann für Nicht-GPU- und Nicht- AWS Graviton-Instance-Familien verwendet werden. Die Standardunterstützung für Amazon Linux AMI ist beendet. Weitere Informationen finden Sie unter Amazon Linux AMI.

> **Note**
>
> Das AMI, das Sie für eine Datenverarbeitungsumgebung auswählen, muss mit der Architektur der Instance-Typen übereinstimmen, die Sie für diese Datenverarbeitungsumgebung verwenden möchten. Wenn Ihre Datenverarbeitungsumgebung beispielsweise A1 Instance-Typen verwendet, muss das von Ihnen gewählte Datenverarbeitungsressourcen-AMI Arm Instances unterstützen. Amazon ECS verkauft sowohl - als auch -x86ArmVersionen des Amazon-ECS-optimierten Amazon Linux 2-AMI. Weitere Informationen finden Sie unter Amazon-ECS-optimiertes Amazon Linux 2-AMI im Entwicklerhandbuch für Amazon Elastic Container Service.

19. (Optional) Wählen Sie für Startvorlage eine vorhandene Amazon EC2-Startvorlage aus, um Ihre Rechenressourcen zu konfigurieren. Die Standardversion der Vorlage wird automatisch ausgefüllt. Weitere Informationen finden Sie unter Support für Startvorlagen.

> **Note**
>
> In einer Startvorlage können Sie ein benutzerdefiniertes AMI angeben, das Sie erstellt haben.

20. (Optional) Geben Sie unter Launch template version (Version der Startvorlage) `$Default`, `$Latest` oder eine bestimmte Versionsnummer ein, die verwendet werden soll.

> **Important**
>
> Nachdem die Datenverarbeitungsumgebung erstellt wurde, wird die verwendete Startvorlagenversion nicht geändert, selbst wenn die - `$Default` oder -`$Latest`Version für die Startvorlage aktualisiert wird. Um eine neue Startvorlagenversion zu verwenden, erstellen Sie zunächst eine neue Datenverarbeitungsumgebung und fügen Sie die neue Datenverarbeitungsumgebung der vorhandenen Auftragswarteschlange hinzu. Entfernen Sie dann die alte Datenverarbeitungsumgebung aus der Auftragswarteschlange und löschen Sie die alte Datenverarbeitungsumgebung.

21. Im Abschnitt Netzwerkkonfiguration:

    a. Wählen Sie für Virtual Private Cloud (VPC) ID eine Amazon VPC aus.

    b. Für Subnetze AWS-Konto werden die Subnetze für Ihr aufgelistet. Wenn Sie einen benutzerdefinierten Satz von Subnetzen erstellen möchten, wählen Sie Subnetze löschen und dann die gewünschten Subnetze aus.

    > **Important**
    >
    > Rechenressourcen müssen über einen VPC-Endpunkt oder mehrere öffentliche IP-Adressen mit dem Amazon-ECS-VPC-Endpunkt kommunizieren. Weitere Informationen finden Sie unter Amazon-ECS-Schnittstellen-VPC-Endpunkte (AWS PrivateLink). Wenn für Ihre Instance kein VPC-Endpunkt oder keine öffentliche IP-Adresse konfiguriert ist, können Sie Network Address Translation (NAT) verwenden. Weitere Informationen zu NAT finden Sie unter NAT-Gateways und Erstellen einer Virtual Private Cloud .

    c. Wählen Sie für Sicherheitsgruppen die Amazon EC2-Sicherheitsgruppen aus, die Sie der Instance zuordnen möchten. Wenn Sie einen benutzerdefinierten Satz von Sicherheitsgruppen erstellen möchten, wählen Sie Sicherheitsgruppen löschen aus. Wählen Sie dann die gewünschten Sicherheitsgruppen aus.

22. Wählen Sie Weiter aus.

Erstellen einer Auftragswarteschlange

Eine Auftragswarteschlange speichert Ihre übermittelten Aufträge, bis der AWS Batch Scheduler den Auftrag auf einer Ressource in Ihrer Datenverarbeitungsumgebung ausführt. Weitere Informationen finden Sie unter Warteschlangen für Job.

Gehen Sie wie folgt vor, um eine Auftragswarteschlange für eine Amazon EC2-Orchestrierung zu erstellen:

1. Geben Sie im Abschnitt Konfiguration der Auftragswarteschlange für Name einen eindeutigen Namen für Ihre Datenverarbeitungsumgebung an. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
2. Geben Sie für Priorität eine Ganzzahl zwischen 0 und 100 für die Auftragswarteschlange ein.

   > **Important**
   >
   > Höheren Ganzzahlwerten wird vom AWS Batch Scheduler eine höhere Priorität zugewiesen.

3. Wählen Sie Weiter aus.

Erstellen Sie eine Auftragsdefinition

AWS Batch -Auftragsdefinitionen geben an, wie Aufträge ausgeführt werden sollen. Obwohl jeder Auftrag auf eine Auftragsdefinition verweisen muss, können viele der in der Auftragsdefinition angegebenen Parameter zur Laufzeit überschrieben werden.

So erstellen Sie die Auftragsdefinition:

1. Im Abschnitt Allgemeine Konfiguration:

   a. Geben Sie im Abschnitt Allgemeine Konfiguration für Name einen eindeutigen Namen für Ihre Datenverarbeitungsumgebung an. Der Name kann bis zu 128 Zeichen lang sein. Der Name kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.

   b. (Optional) Geben Sie unter Ausführungs-Timeout die Zeit (in Sekunden) ein, nach der ein nicht abgeschlossener Auftrag beendet wird.

> **Important**
>
> Das minimale Timeout beträgt 60 Sekunden.

c. (Optional) Ein Tag ist eine Bezeichnung, die einer Ressource zugewiesen ist. Um ein Tag hinzuzufügen, erweitern Sie Tags und wählen Sie dann Tag hinzufügen aus. Geben Sie ein Schlüssel-Wert-Paar ein und wählen Sie dann erneut Tag hinzufügen aus.

> **Important**
>
> Wenn Sie Tag hinzufügen wählen, müssen Sie ein Schlüssel-Wert-Paar eingeben und erneut Tag hinzufügen oder Tag entfernen auswählen.

d. (Optional) Aktivieren Sie die Option Tags propagieren, um Tags an die Amazon Elastic Container Service-Aufgabe zu propagieren.

2. Im Abschnitt Container-Konfiguration:

a. Geben Sie für Image den Namen des Images ein, das zum Starten des Containers verwendet wird. Standardmäßig sind alle Images in der Docker-Hub-Registrierung verfügbar. Sie können auch andere Repositorys im Format repository-url/image:tag angeben. Der Parameter kann bis zu 255 Zeichen lang sein. Der Parameter kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-), Unterstriche (_), Doppelpunkte (:), Punkte (.), Schrägstriche (/) und Zahlenzeichen (#) enthalten. Der Parameter wird `Image` im Abschnitt Erstellen eines Containers der Docker Remote API und dem `IMAGE` Parameter zugeordnetdocker run.

> **Note**
>
> Docker Die Image-Architektur muss mit der Prozessorarchitektur der Rechenressourcen übereinstimmen, für die sie geplant sind. Beispielsweise können Armbasierte Docker Images nur auf -Armbasierten Rechenressourcen ausgeführt werden.

- Bilder in öffentlichen Amazon-ECR-Repositorys verwenden die vollständige - `registry/repository[:tag]` oder `-registry/repository[@digest]`Namenskonvention (z. B. `public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).

- Bilder in Amazon-ECR-Repositorys verwenden die vollständige `registry/repository:tag` Namenskonvention (z. B. *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*).
- Images in offiziellen Repositorys in Docker Hub verwenden einen einzelnen Namen (z. B. `ubuntu` oder `mongo`).
- Images in anderen Repositorys in Docker Hub sind mit einem Organisationsnamen qualifiziert (z. B, `amazon/amazon-ecs-agent`).
- Image in anderen Online-Repositorys sind durch einen Domain-Namen zusätzlich qualifiziert (z. B. `quay.io/assemblyline/ubuntu`).

b. Geben Sie unter Befehl den Befehl an, der an den Container übergeben werden soll. Dieser Parameter ist `Cmd` im Abschnitt Erstellen eines Containers der Docker Remote-API und dem Parameter `COMMAND` von docker run zugeordnet. Weitere Informationen zum Docker-Parameter `CMD` finden Sie unter https://docs.docker.com/engine/reference/builder/#cmd.

> **Note**
>
> Sie können Standardwerte und Platzhalter für die Parameterersetzung in Ihrem Befehl verwenden. Weitere Informationen finden Sie unter Parameter.

c. (Optional) Geben Sie für Ausführungsrolle eine IAM-Rolle an, die den Amazon-ECS-Container-Agenten die Berechtigung erteilt, AWS API-Aufrufe in Ihrem Namen durchzuführen. Diese Funktion verwendet Amazon-ECS-IAM-Rollen für Aufgaben. Weitere Informationen finden Sie unter IAM-Rollen für die Amazon-ECS-Aufgabenausführung im Amazon Elastic Container Service-Entwicklerhandbuch.

d. (Optional) Wählen Sie für die Konfiguration der Auftragsrolle eine IAM-Rolle aus, die über Berechtigungen für die AWS APIs verfügt. Diese Funktion verwendet Amazon-ECS-IAM-Rollen für Aufgaben. Weitere Informationen finden Sie unter IAM-Rollen für Aufgaben im Entwicklerhandbuch zum Amazon Elastic Container Service.

> **Note**
>
> Hier werden nur Rollen angezeigt, die über die Vertrauensstellung der Aufgabenrolle von Amazon Elastic Container Service verfügen. Weitere Informationen zum Erstellen einer IAM-Rolle für Ihre AWS Batch Aufträge finden Sie unter Erstellen

einer IAM-Rolle und -Richtlinie für Ihre Aufgaben im Amazon Elastic Container Service-Entwicklerhandbuch.

e. (Optional) Sie können der Auftragsdefinition Parameter als Schlüssel-Wert-Zuweisungen hinzufügen, um die Standardwerte der Auftragsdefinition zu überschreiben. So fügen Sie einen Parameter hinzu:

   - Wählen Sie für Parameter die Option Parameter hinzufügen aus. Geben Sie ein Schlüssel-Wert-Paar ein und wählen Sie dann erneut Parameter hinzufügen aus.

   > **Important**
   >
   > Wenn Sie Parameter hinzufügen auswählen, müssen Sie mindestens einen Parameter konfigurieren oder Parameter entfernen auswählen.

f. Geben Sie im Abschnitt Umgebungskonfiguration für vCPUs die Anzahl der vCPUs an, die für den Container reserviert werden sollen. Dieser Parameter ordnet zu `CpuShares` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--cpu-shares` für die docker run zu. Jede vCPU entspricht 1.024 CPU-Anteilen.

g. Geben Sie für Speicher das harte Limit (in MiB) des Speichers an, der dem Auftragscontainer präsentiert werden soll. Wenn Ihr Container versucht, den hier angegebenen Speicher zu überschreiten, wird der Container gestoppt. Dieser Parameter ordnet zu `Memory` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--memory` für die docker run zu.

h. Wählen Sie unter Anzahl der GPUs die Anzahl der GPUs aus, die für den Container reserviert werden sollen.

i. (Optional) Wählen Sie unter Konfiguration von Umgebungsvariablen die Option Umgebungsvariablen hinzufügen aus, um Umgebungsvariablen hinzuzufügen, die an den Container übergeben werden sollen. Dieser Parameter ordnet zu `Env` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--env` für die docker run zu.

j. (Optional) Wählen Sie für Secrets die Option Secret hinzufügen aus, um Secrets als Name-Wert-Paare hinzuzufügen. Diese Secrets werden im Container verfügbar gemacht. Weitere Informationen finden Sie unter secretOptions in Jobdefinitionsparameter für ContainerProperties.

k. (Optional) Im Abschnitt Linux-Konfiguration:

i. Für Benutzer geben Sie den Benutzernamen zur Verwendung innerhalb des Containers ein. Dieser Parameter ordnet zu `User` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--user` für die docker run zu.

ii. Um dem Auftragscontainer erhöhte Berechtigungen auf der Host-Instance zu erteilen (ähnlich wie der `root` Benutzer), ziehen Sie den Schieberegler Privilegiert nach rechts. Dieser Parameter ordnet zu `Privileged` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--privileged` für die docker run zu.

iii. Aktivieren Sie Init-Prozess aktivieren, um einen -`init`Prozess innerhalb des Containers auszuführen. Dieser Prozess leitet Signale weiter und nimmt Prozesse auf.

l. (Optional) Im Abschnitt Dateisystemkonfiguration:

i. Aktivieren Sie Schreibgeschütztes Dateisystem aktivieren, um den Schreibzugriff auf das Volume zu entfernen.

ii. Geben Sie für Größe des gemeinsam genutzten Speichers die Größe (in MiB) des `/dev/shm` Volumes ein.

iii. Geben Sie für Maximale Auslagerungsgröße die Gesamtmenge des Auslagerungsspeichers (in MiB) ein, die der Container verwenden kann.

iv. Geben Sie für Auslagerung einen Wert zwischen 0 und 100 ein, um das Auslagerungsverhalten des Containers anzugeben. Wenn Sie keinen Wert angeben und Auslagern aktiviert ist, ist der Wert standardmäßig 60. Weitere Informationen finden Sie unter Auslagerung in Jobdefinitionsparameter für ContainerProperties.

v. (Optional) Erweitern Sie Zusätzliche Konfiguration .

vi. Wählen Sie für Tmpfs die Option tmpfs hinzufügen aus, um einen `tmpfs` Mount hinzuzufügen.

vii. Wählen Sie für Geräte die Option Gerät hinzufügen aus, um ein Gerät hinzuzufügen:

A. Geben Sie für Container path (Container-Pfad) den Pfad in der Container-Instance an, um das der Host-Instance zugeordnete Gerät zugänglich zu machen. Wenn Sie dieses Feld leer lassen, wird der Hostpfad im Container verwendet.

B. Geben Sie für Host path (Host-Pfad) den Pfad eines Geräts in der Host-Instance an.

C. Wählen Sie für Berechtigungen eine oder mehrere Berechtigungen aus, die auf das Gerät angewendet werden sollen. Die verfügbaren Berechtigungen sind READ , WRITE und MVNOD .

viii. (Optional) Wählen Sie für Ulimits-Konfiguration die Option Ulimit hinzufügen aus, um einen `ulimits` Wert für den Container hinzuzufügen. Geben Sie den Namen , das weiche Limit und die Hard-Limit-Werte ein und wählen Sie dann Ulimit hinzufügen aus.

3. Wählen Sie Weiter aus.

Erstellen eines Auftrags

Gehen Sie wie folgt vor, um einen Auftrag zu erstellen:

1. Geben Sie im Abschnitt Auftragskonfiguration für Name einen eindeutigen Namen für den Auftrag an. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
2. Wählen Sie Weiter aus.

Überprüfen und erstellen

Überprüfen Sie auf der Seite Überprüfen und erstellen die Konfigurationsschritte. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Ressourcen erstellen aus.

Erste Schritte – Fargate

AWS Fargate startet und skaliert die Rechenleistung so, dass sie genau den Ressourcenanforderungen entspricht, die Sie für den Container angeben. Mit Fargate müssen Sie keine zu hohe Bereitstellung oder zusätzliche Server zahlen. Weitere Informationen finden Sie unter Fargate .

Erstellen einer Datenverarbeitungsumgebung

Gehen Sie wie folgt vor, um eine Datenverarbeitungsumgebung für eine Fargate-Orchestrierung zu erstellen:

1. Öffnen Sie den AWS Batch Assistenten für die Erstausführung der Konsole.
2. Wählen Sie für Orchestrierungstyp auswählen die Option Fargate aus.
3. Wählen Sie Weiter aus.

4. Geben Sie im Abschnitt Konfiguration der Datenverarbeitungsumgebung für Name einen eindeutigen Namen für Ihre Datenverarbeitungsumgebung an. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
5. (Optional) Ein Tag ist eine Bezeichnung, die einer Ressource zugewiesen ist. Um ein Tag hinzuzufügen, erweitern Sie Tags und wählen Sie dann Tag hinzufügen aus. Geben Sie ein Schlüssel-Wert-Paar ein und wählen Sie dann erneut Tag hinzufügen aus.

   > **Important**
   >
   > Wenn Sie Tag hinzufügen auswählen, müssen Sie ein Schlüssel-Wert-Paar eingeben und erneut Tag hinzufügen oder Tag entfernen auswählen.

6. (Optional) Aktivieren Sie im Abschnitt Instance-Konfiguration für Fargate-Spot-Kapazität verwenden die Option Mit Spot-Instances aktivieren.
7. Geben Sie unter Maximale vCPUs die maximale Anzahl von vCPUs ein, die die Instance verwenden kann.
8. Im Abschnitt Netzwerkkonfiguration:

   a. Wählen Sie für Virtual Private Cloud (VPC) ID eine Amazon VPC aus.

   b. Für Subnetze AWS-Konto werden die Subnetze für Ihr aufgelistet. Wenn Sie einen benutzerdefinierten Satz von Subnetzen erstellen möchten, wählen Sie Subnetze löschen und dann die gewünschten Subnetze aus.

      > **Important**
      >
      > Rechenressourcen müssen über einen VPC-Endpunkt oder mehrere öffentliche IP-Adressen mit dem Amazon-ECS-VPC-Endpunkt kommunizieren. Weitere Informationen finden Sie unter Amazon-ECS-Schnittstellen-VPC-Endpunkte (AWS PrivateLink). Wenn für Ihre Instance kein VPC-Endpunkt oder keine öffentliche IP-Adresse konfiguriert ist, können Sie Network Address Translation (NAT) verwenden. Weitere Informationen zu NAT finden Sie unter NAT-Gateways und Erstellen einer Virtual Private Cloud .

   c. Wählen Sie für Sicherheitsgruppen die Amazon EC2-Sicherheitsgruppen aus, die Sie der Instance zuordnen möchten. Wenn Sie einen benutzerdefinierten Satz von

Sicherheitsgruppen erstellen möchten, wählen Sie Sicherheitsgruppen löschen aus. Wählen Sie dann die gewünschten Sicherheitsgruppen aus.

9. Wählen Sie Weiter aus.

Erstellen einer Auftragswarteschlange

Eine Auftragswarteschlange speichert Ihre übermittelten Aufträge, bis der AWS Batch Scheduler den Auftrag auf einer Ressource in Ihrer Datenverarbeitungsumgebung ausführt. So erstellen Sie eine Auftragswarteschlange:

Gehen Sie wie folgt vor, um eine Auftragswarteschlange für eine Fargate-Orchestrierung zu erstellen:

1. Geben Sie im Abschnitt Konfiguration der Auftragswarteschlange für Name einen eindeutigen Namen für Ihre Datenverarbeitungsumgebung an. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
2. Geben Sie für Priorität eine Ganzzahl zwischen 0 und 100 für die Auftragswarteschlange ein.

   > **Important**
   >
   > Höheren Ganzzahlwerten wird vom AWS Batch Scheduler eine höhere Priorität zugewiesen.

3. Wählen Sie Weiter aus.

Erstellen Sie eine Auftragsdefinition

So erstellen Sie die Auftragsdefinition:

1. Im Abschnitt Allgemeine Konfiguration:

   a. Geben Sie unter Name einen benutzerdefinierten Auftragsdefinitionsnamen ein.

      Geben Sie im Abschnitt Allgemeine Konfiguration für Name einen eindeutigen Namen für Ihre Datenverarbeitungsumgebung an. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.

   b. (Optional) Geben Sie unter Ausführungs-Timeout die Zeit (in Sekunden) ein, nach der ein nicht abgeschlossener Auftrag beendet wird.

> **Important**
>
> Das minimale Timeout beträgt 60 Sekunden.

c. (Optional) Ein Tag ist eine Bezeichnung, die einer Ressource zugewiesen ist. Um ein Tag hinzuzufügen, erweitern Sie Tags und wählen Sie dann Tag hinzufügen aus. Geben Sie ein Schlüssel-Wert-Paar ein und wählen Sie dann erneut Tag hinzufügen aus.

> **Important**
>
> Wenn Sie Tag hinzufügen wählen, müssen Sie ein Schlüssel-Wert-Paar eingeben und erneut Tag hinzufügen oder Tag entfernen auswählen.

d. (Optional) Aktivieren Sie die Option Tags propagieren, um Tags an die Amazon Elastic Container Service-Aufgabe zu propagieren.

2. Im Abschnitt Fargate-Plattformkonfiguration:

a. (Optional) Geben Sie für Fargate-Plattformversion die gewünschte spezifische Laufzeitumgebung ein.

b. Wählen Sie für Laufzeitplattform eine LINUX- oder Windows- aus.

c. (Nur Windows) Wählen Sie für Betriebssystemfamilie ein Betriebssystem aus.

d. Wählen Sie für CPU-Architektur die gewünschte CPU-Architektur aus.

e. (Optional) Aktivieren Sie Öffentliche IP zuweisen, um eine öffentliche IP-Adresse zuzuweisen.

f. Geben Sie für Flüchtiger Speicher die gewünschte Menge an flüchtigem Speicher ein.

> **Note**
>
> Standardmäßig werden 20 GiB flüchtiger Speicher verwendet. Um zusätzlichen flüchtigen Speicher zu verwenden, geben Sie einen Wert zwischen 21 GiB und 100 GiB ein.

g. Wählen Sie für Ausführungsrolle eine Aufgabenausführungsrolle aus, mit der Amazon Elastic Container Service (Amazon ECS)-Agenten in Ihrem Namen AWS Aufrufe tätigen können. Sie können beispielsweise ecsTaskExecutionRolle auswählen.

3. Im Abschnitt Container-Konfiguration:

a. Geben Sie für Image den Namen des Images ein, das zum Starten des Containers verwendet wird. Standardmäßig sind alle Images in der Docker-Hub-Registrierung verfügbar. Sie können auch andere Repositorys im Format repository-url/image:tag angeben. Der Parameter kann bis zu 255 Zeichen lang sein. Sie kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-), Unterstriche (_), Doppelstriche (:), Punkte (.), Schrägstriche (/) und Zahlenzeichen (#) enthalten. Der Parameter wird `Image` im Abschnitt Erstellen eines Containers der Docker Remote API und dem `IMAGE` Parameter zugeordnetdocker run.

> **Note**
>
> Docker Die Image-Architektur muss mit der Prozessorarchitektur der Rechenressourcen übereinstimmen, für die sie geplant sind. Beispielsweise können Armbasierte Docker Images nur auf -Armbasierten Rechenressourcen ausgeführt werden.

- Images in öffentlichen Amazon-ECR-Repositorys verwenden die vollständige - `registry/repository[:tag]` oder `-registry/repository[@digest]`Namenskonvention (z. B. `public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).
- Bilder in Amazon-ECR-Repositorys verwenden die vollständige `registry/repository:tag` Namenskonvention (z. B. *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*).
- Images in offiziellen Repositorys in Docker Hub verwenden einen einzelnen Namen (z. B. `ubuntu` oder `mongo`).
- Images in anderen Repositorys in Docker Hub sind mit einem Organisationsnamen qualifiziert (z. B, `amazon/amazon-ecs-agent`).
- Image in anderen Online-Repositorys sind durch einen Domain-Namen zusätzlich qualifiziert (z. B. `quay.io/assemblyline/ubuntu`).

b. Geben Sie unter Befehl den Befehl an, der an den Container übergeben werden soll. Dieser Parameter ist `Cmd` im Abschnitt Erstellen eines Containers der Docker Remote-API und dem Parameter `COMMAND` von docker run zugeordnet. Weitere Informationen zum Docker-Parameter `CMD` finden Sie unter https://docs.docker.com/engine/reference/builder/#cmd.

> **Note**
>
> Sie können Standardwerte und Platzhalter für die Parameterersetzung in Ihrem Befehl verwenden. Weitere Informationen finden Sie unter Parameter.

> **Tip**
>
> Wählen Sie Info aus, um sich die Bash- und JSON-Codebeispiele anzusehen.

c. (Optional) Sie können der Auftragsdefinition Parameter als Schlüssel-Wert-Zuweisungen hinzufügen, um die Standardeinstellungen der Auftragsdefinition zu überschreiben. So fügen Sie einen Parameter hinzu:

   - Wählen Sie für Parameter die Option Parameter hinzufügen aus. Geben Sie ein Schlüssel-Wert-Paar ein und wählen Sie dann erneut Parameter hinzufügen aus.

> **Important**
>
> Wenn Sie Parameter hinzufügen auswählen, müssen Sie mindestens einen Parameter konfigurieren oder Parameter entfernen auswählen.

d. (Optional) Wählen Sie im Abschnitt Umgebungskonfiguration für die Konfiguration der Auftragsrolle eine IAM-Rolle aus, die die Berechtigung zur Verwendung der AWS APIs bereitstellt.

e. Geben Sie im Abschnitt Umgebungskonfiguration für vCPUs die Anzahl der vCPUs an, die für den Container reserviert werden sollen. Dieser Parameter ordnet zu `CpuShares` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--cpu-shares` für die docker run zu. Jede vCPU entspricht 1.024 CPU-Anteilen.

f. Geben Sie für Speicher das harte Limit (in MiB) des Speichers an, der dem Auftragscontainer präsentiert werden soll. Wenn Ihr Container versucht, den hier angegebenen Speicher zu überschreiten, wird der Container gestoppt. Dieser Parameter ordnet zu `Memory` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--memory` für die docker run zu.

g. (Optional) Wählen Sie für Umgebungsvariablen die Option Umgebungsvariablen hinzufügen aus, um Umgebungsvariablen hinzuzufügen, die an den Container übergeben werden

sollen. Dieser Parameter ordnet zu `Env` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--env` für die docker run zu.

4. Wählen Sie Weiter aus.

Erstellen eines Auftrags

Gehen Sie wie folgt vor, um einen Fargate-Auftrag zu erstellen:

1. Geben Sie im Abschnitt Auftragskonfiguration für Name einen eindeutigen Namen für den Auftrag an. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
2. Wählen Sie Weiter aus.

Überprüfen und erstellen

Überprüfen Sie auf der Seite Überprüfen und erstellen die Konfigurationsschritte. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Ressourcen erstellen aus.

Erste Schritte mit AWS Batch in Amazon EKS

AWS Batch in Amazon EKS ist ein verwalteter Service für die Planung und Skalierung von Batch-Workloads in bestehende Amazon- AWS Batch EKS-Cluster. erstellt, verwaltet oder führt keine Lebenszyklusvorgänge Ihrer Amazon-EKS-Cluster in Ihrem Namen durch. AWS Batch Orchestrierung skaliert Knoten, die von verwaltet werden, hoch und herunter AWS Batch und führt Pods auf diesen Knoten aus.

AWS Batch stößt keine Knoten, Auto-Scaling-Knotengruppen oder Pods auf, die keinen AWS Batch Datenverarbeitungsumgebungen in Ihrem Amazon-EKS-Cluster zugeordnet sind. Damit effektiv arbeiten AWS Batch kann, benötigt seine serviceverknüpfte Rolle RBAC-Berechtigungen (Kubernetesrollenbasierte Zugriffskontrolle) in Ihrem vorhandenen Amazon-EKS-Cluster. Weitere Informationen finden Sie unter Verwenden der RBAC-Autorisierung in der Kubernetes - Dokumentation.

AWS Batch erfordert einen Kubernetes Namespace, in den Pods als AWS Batch Aufträge einteilen kann. Wir empfehlen einen dedizierten Namespace, um die AWS Batch Pods von Ihren anderen Cluster-Workloads zu isolieren.

Nachdem RBAC Zugriff gewährt und ein Namespace eingerichtet wurde, können Sie diesen Amazon-AWS Batch EKS-Cluster mithilfe der CreateComputeEnvironment-API-Operation einer AWS Batch Computing-Umgebung zuordnen. Eine Auftragswarteschlange kann dieser neuen Amazon-EKS-Datenverarbeitungsumgebung zugeordnet werden. AWS Batch Aufträge werden mithilfe der SubmitJob-API-Operation an die Auftragswarteschlange übermittelt. AWS Batch startet dann AWS Batch verwaltete Knoten und platziert Aufträge aus der Auftragswarteschlange als Pods in dem EKSKubernetes-Cluster, der einer - AWS Batch Datenverarbeitungsumgebung zugeordnet ist.

In den folgenden Abschnitten wird beschrieben, wie Sie für AWS Batch auf Amazon EKS einrichten.

Inhalt

Voraussetzungen

Bevor Sie mit diesem Tutorial beginnen, müssen Sie die folgenden Tools und Ressourcen installieren und konfigurieren, die Sie zum Erstellen und Verwalten von AWS Batch und Amazon-EKS-Ressourcen benötigen.

- AWS CLI – Ein Befehlszeilen-Tool für die Arbeit mit AWS -Services, einschließlich Amazon EKS. Dieses Handbuch erfordert, dass Sie Version 2.8.6 oder höher oder 1.26.0 oder höher verwenden. Weitere Informationen finden Sie unter Installieren, Aktualisieren und Deinstallieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch. Nach der Installation der empfehlen wir Ihnen AWS CLI, sie ebenfalls zu konfigurieren. Weitere Informationen finden Sie unter Schnellkonfiguration mit `aws configure` im AWS Command Line Interface -Benutzerhandbuch.
- **`kubectl`** – Ein Befehlszeilentool für die Arbeit mit Kubernetes-Clustern. Dieses Handbuch erfordert, dass Sie Version `1.23` oder höher verwenden. Weitere Informationen finden Sie unter Installieren oder Aktualisieren von `kubectl` im Amazon-EKS-Benutzerhandbuch.
- **`eksctl`** – Ein Befehlszeilen-Tool für die Arbeit mit Amazon-EKS-Clustern, das viele einzelne Aufgaben automatisiert. Dieses Handbuch erfordert, dass Sie Version `0.115.0` oder höher verwenden. Weitere Informationen finden Sie unter Installieren oder Aktualisieren von `eksctl` im Amazon-EKS-Benutzerhandbuch.
- Erforderliche IAM-Berechtigungen – Der von Ihnen verwendete IAM-Sicherheitsprinzipal muss über Berechtigungen zum Arbeiten mit Amazon-EKS-IAM-Rollen und serviceverknüpften Rollen AWS CloudFormation sowie einer VPC und zugehörigen Ressourcen verfügen. Weitere Informationen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon Elastic Kubernetes Service und Verwenden von serviceverknüpften Rollen im IAM-Benutzerhandbuch. Sie müssen alle Schritte in diesem Handbuch als derselbe Benutzer ausführen.
- Erstellen eines Amazon-EKS-Clusters – Weitere Informationen finden Sie unter Erste Schritte mit Amazon EKS – `eksctl` im Amazon-EKS-Benutzerhandbuch.

> **Note**
>
> AWS Batch unterstützt nur Amazon-EKS-Cluster mit API-Server-Endpunkten, die öffentlichen Zugriff haben und für das öffentliche Internet zugänglich sind. Standardmäßig haben Amazon-EKS-Cluster-API-Server-Endpunkte öffentlichen Zugriff. Weitere Informationen finden Sie unter Zugriffskontrolle für Amazon-EKS-Cluster-Endpunkte im Amazon-EKS-Benutzerhandbuch.

> **Note**
>
> AWS Batch bietet keine Orchestrierung für verwaltete Knoten für CoreDNS oder andere Bereitstellungs-Pods. Wenn Sie CoreDNS benötigen, finden Sie weitere Informationen unter Hinzufügen des CoreDNS-Amazon-EKS-Add-ons im Amazon-EKS-

Benutzerhandbuch. Oder verwenden Sie , `eksctl create cluster create` um den Cluster zu erstellen, er enthält standardmäßig CoreDNS.

- Berechtigungen – Benutzer, die den CreateComputeEnvironment -API-Vorgang aufrufen, um eine Datenverarbeitungsumgebung zu erstellen, die Amazon-EKS-Ressourcen verwendet, benötigen Berechtigungen für den `eks:DescribeCluster`-API-Vorgang. Die Verwendung der AWS Management Console zum Erstellen einer Rechenressource mit Amazon-EKS-Ressourcen erfordert Berechtigungen für `eks:DescribeCluster` und `eks:ListClusters`.

Schritt 1: Vorbereiten Ihres Amazon-EKS-Clusters für AWS Batch

Alle Schritte sind erforderlich.

1. Erstellen eines dedizierten Namespace für AWS Batch Aufträge

   Verwenden Sie `kubectl`, um einen neuen Namespace zu erstellen.

   ```
   $ namespace=my-aws-batch-namespace
   $ cat - <<EOF | kubectl create -f -
   {
     "apiVersion": "v1",
     "kind": "Namespace",
     "metadata": {
       "name": "${namespace}",
       "labels": {
         "name": "${namespace}"
       }
     }
   }
   EOF
   ```

   Ausgabe:

   ```
   namespace/my-aws-batch-namespace created
   ```

2. Aktivieren des Zugriffs über die rollenbasierte Zugriffskontrolle (RBAC)

   Verwenden Sie `kubectl`, um eine Kubernetes Rolle für den Cluster AWS Batch zu erstellen, damit Knoten und Pods überwachen und die Rolle binden kann. Sie müssen dies einmal für jeden EKS-Cluster tun.

> **Note**
>
> Weitere Informationen zur Verwendung der RBAC-Autorisierung finden Sie unter Verwenden der RBAC-Autorisierung im Kubernetes -Benutzerhandbuch.

```
$ cat - <<EOF | kubectl apply -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: aws-batch-cluster-role
rules:
  - apiGroups: [""]
    resources: ["namespaces"]
    verbs: ["get"]
  - apiGroups: [""]
    resources: ["nodes"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["configmaps"]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["apps"]
    resources: ["daemonsets", "deployments", "statefulsets", "replicasets"]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["rbac.authorization.k8s.io"]
    resources: ["clusterroles", "clusterrolebindings"]
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: aws-batch-cluster-role-binding
subjects:
- kind: User
  name: aws-batch
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
```

```
  name: aws-batch-cluster-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

Ausgabe:

```
clusterrole.rbac.authorization.k8s.io/aws-batch-cluster-role created
clusterrolebinding.rbac.authorization.k8s.io/aws-batch-cluster-role-binding created
```

Erstellen Sie eine Rolle im Namespace-Bereich Kubernetes für , AWS Batch um Pods zu verwalten und zu Lebenszyklus-Pods zu binden. Sie müssen dies einmal für jeden eindeutigen Namespace tun.

```
$ namespace=my-aws-batch-namespace
$ cat - <<EOF | kubectl apply -f - --namespace "${namespace}"
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: aws-batch-compute-environment-role
  namespace: ${namespace}
rules:
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["create", "get", "list", "watch", "delete", "patch"]
  - apiGroups: [""]
    resources: ["serviceaccounts"]
    verbs: ["get", "list"]
  - apiGroups: ["rbac.authorization.k8s.io"]
    resources: ["roles", "rolebindings"]
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: aws-batch-compute-environment-role-binding
  namespace: ${namespace}
subjects:
- kind: User
  name: aws-batch
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
```

```
  name: aws-batch-compute-environment-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

Ausgabe:

```
role.rbac.authorization.k8s.io/aws-batch-compute-environment-role created
rolebinding.rbac.authorization.k8s.io/aws-batch-compute-environment-role-binding
 created
```

Aktualisieren Sie die Kubernetes `aws-auth` Konfigurationszuordnung, um die vorherigen RBAC-Berechtigungen der AWS Batch serviceverknüpften Rolle zuzuordnen.

```
$ eksctl create iamidentitymapping \
    --cluster my-cluster-name \
    --arn "arn:aws:iam::<your-account>:role/AWSServiceRoleForBatch" \
    --username aws-batch
```

Ausgabe:

```
2022-10-25 20:19:57 [#]  adding identity "arn:aws:iam::<your-account>:role/
AWSServiceRoleForBatch" to auth ConfigMap
```

> **Note**
>
> Der Pfad `aws-service-role/batch.amazonaws.com/` wurde aus dem ARN der serviceverknüpften Rolle entfernt. Dies liegt an einem Problem mit der `aws-auth` Konfigurationszuordnung. Weitere Informationen finden Sie unter Rollen mit Pfaden funktionieren nicht, wenn der Pfad in ihrem ARN enthalten ist im aws-authconfigmap.

Schritt 2: Erstellen einer Amazon-EKS-Datenverarbeitungsumgebung

AWS Batch -Rechenumgebungen definieren Datenverarbeitungsressourcenparameter, um Ihre Batch-Workload-Anforderungen zu erfüllen. In einer verwalteten Datenverarbeitungsumgebung AWS Batch unterstützt Sie bei der Verwaltung der Kapazität und der Instance-Typen der Datenverarbeitungsressourcen (Kubernetes-Knoten) in Ihrem Amazon-EKS-Cluster. Dies basiert auf

der Spezifikation der Rechenressourcen, die Sie beim Erstellen der Rechenumgebung definieren. Sie können EC2-On-Demand-Instances oder EC2-Spot-Instances verwenden.

Nachdem die AWSServiceRoleForBatch serviceverknüpfte Rolle Zugriff auf Ihren Amazon-EKS-Cluster hat, können Sie - AWS Batch Ressourcen erstellen. Erstellen Sie zunächst eine Datenverarbeitungsumgebung, die auf Ihren Amazon-EKS-Cluster verweist.

```
$ cat <<EOF > ./batch-eks-compute-environment.json
{
  "computeEnvironmentName": "My-Eks-CE1",
  "type": "MANAGED",
  "state": "ENABLED",
  "eksConfiguration": {
    "eksClusterArn": "arn:aws:eks:<region>:123456789012:cluster/<cluster-name>",
    "kubernetesNamespace": "my-aws-batch-namespace"
  },
  "computeResources": {
    "type": "EC2",
    "allocationStrategy": "BEST_FIT_PROGRESSIVE",
    "minvCpus": 0,
    "maxvCpus": 128,
    "instanceTypes": [
        "m5"
    ],
    "subnets": [
        "<eks-cluster-subnets-with-access-to-internet-for-image-pull>"
    ],
    "securityGroupIds": [
        "<eks-cluster-sg>"
    ],
    "instanceRole": "<eks-instance-profile>"
  }
}
EOF
$ aws batch create-compute-environment --cli-input-json file://./batch-eks-compute-
environment.json
```

Hinweise

- Der `serviceRole` Parameter sollte nicht angegeben werden, dann wird die AWS Batch serviceverknüpfte Rolle verwendet. AWS Batch auf Amazon EKS unterstützt nur die AWS Batch serviceverknüpfte Rolle.

- Für Amazon-EKS-Rechenumgebungen werden nur `BEST_FIT_PROGRESSIVE-SPOT_CAPACITY_OPTIMIZED`, - und `-SPOT_PRICE_CAPACITY_OPTIMIZED`Zuweisungsstrategien unterstützt.

  > **Note**
  >
  > Wir empfehlen, dass Sie `SPOT_CAPACITY_OPTIMIZED` in den meisten `SPOT_PRICE_CAPACITY_OPTIMIZED` Instances anstelle von verwenden.

- Informationen zur finden Sie unter Erstellen der Amazon-EKS-`instanceRole`Knoten-IAM-Rolle und Aktivieren des IAM-Prinzipalzugriffs auf Ihren Cluster im Amazon-EKS-Benutzerhandbuch. https://docs.aws.amazon.com/eks/latest/userguide/create-node-role.html#create-worker-node-role Wenn Sie Pod-Netzwerke verwenden, finden Sie weitere Informationen unter Konfigurieren des Amazon-VPC-CNI-Plugins für Kubernetes zur Verwendung von IAM-Rollen für Servicekonten im Amazon-EKS-Benutzerhandbuch.
- Eine Möglichkeit, funktionierende Subnetze für den `-subnets`Parameter zu erhalten, besteht darin, die von erstellten öffentlichen Subnetze der von Amazon EKS verwalteten Knotengruppen zu verwenden, `eksctl` wenn ein Amazon-EKS-Cluster erstellt wurde. Verwenden Sie andernfalls Subnetze, die einen Netzwerkpfad haben, der das Abrufen von Images unterstützt.
- Der `securityGroupIds` Parameter kann dieselbe Sicherheitsgruppe wie der Amazon-EKS-Cluster verwenden. Dieser Befehl ruft die Sicherheitsgruppen-ID für den Cluster ab.

```
$ eks describe-cluster \
    --name <cluster-name> \
    --query cluster.resourcesVpcConfig.clusterSecurityGroupId
```

- Die Wartung einer Amazon-EKS-Rechenumgebung ist eine übergreifende Verantwortlichkeit. Weitere Informationen finden Sie unter Gemeinsame Verantwortung der Kubernetes Knoten.

> **Important**
>
> Es ist wichtig zu bestätigen, dass die Datenverarbeitungsumgebung fehlerfrei ist, bevor Sie fortfahren. Dazu kann der DescribeComputeEnvironments API-Vorgang verwendet werden.
>
> ```
> $ aws batch describe-compute-environments --compute-environments My-Eks-CE1
> ```

Vergewissern Sie sich, dass der `status` Parameter nicht ist`INVALID`. Wenn dies der Fall ist, sehen Sie sich den `statusReason` Parameter für die Ursache an. Weitere Informationen finden Sie unter Problembehebung AWS Batch.

Schritt 3: Erstellen einer Auftragswarteschlange und Anfügen der Datenverarbeitungsumgebung

```
$ aws batch describe-compute-environments --compute-environments My-Eks-CE1
```

Aufträge, die an diese neue Auftragswarteschlange gesendet werden, werden als Pods auf AWS Batch verwalteten Knoten ausgeführt, die dem Amazon-EKS-Cluster beigetreten sind, der Ihrer Datenverarbeitungsumgebung zugeordnet ist.

```
$ cat <<EOF > ./batch-eks-job-queue.json
 {
    "jobQueueName": "My-Eks-JQ1",
    "priority": 10,
    "computeEnvironmentOrder": [
      {
        "order": 1,
        "computeEnvironment": "My-Eks-CE1"
      }
    ]
  }
EOF
$ aws batch create-job-queue --cli-input-json file://./batch-eks-job-queue.json
```

Schritt 4: Erstellen einer Auftragsdefinition

```
$ cat <<EOF > ./batch-eks-job-definition.json
{
  "jobDefinitionName": "MyJobOnEks_Sleep",
  "type": "container",
  "eksProperties": {
    "podProperties": {
      "hostNetwork": true,
      "containers": [
```

```
        {
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "sleep",
            "60"
          ],
          "resources": {
            "limits": {
              "cpu": "1",
              "memory": "1024Mi"
            }
          }
        }
      ],
      "metadata": {
        "labels": {
          "environment": "test"
        }
      }
    }
  }
}
EOF
$ aws batch register-job-definition --cli-input-json file://./batch-eks-job-
definition.json
```

Hinweise

- Es werden nur einzelne Container-Aufträge unterstützt.
- Es gibt Überlegungen zu den `memory` Parametern `cpu` und . Weitere Informationen finden Sie unter Überlegungen zu Arbeitsspeicher und vCPU für AWS Batch in Amazon EKS.

Schritt 5: Senden eines Auftrags

```
$ aws batch submit-job --job-queue My-Eks-JQ1 \
    --job-definition MyJobOnEks_Sleep --job-name My-Eks-Job1
$ aws batch describe-jobs --job <jobId-from-submit-response>
```

Hinweise

- Es werden nur einzelne Container-Aufträge unterstützt.

- Stellen Sie sicher, dass Sie mit allen relevanten Überlegungen für die `memory` Parameter `cpu` und vertraut sind. Weitere Informationen finden Sie unter Überlegungen zu Arbeitsspeicher und vCPU für AWS Batch in Amazon EKS.
- Weitere Informationen zum Ausführen von Aufträgen auf Amazon-EKS-Ressourcen finden Sie unter Stellenangebote bei Amazon EKS.

(Optional) Senden eines Auftrags mit Überschreibungen

Dieser Auftrag überschreibt den an den Container übergebenen Befehl.

```
$ cat <<EOF > ./submit-job-override.json
{
  "jobName": "EksWithOverrides",
  "jobQueue": "My-Eks-JQ1",
  "jobDefinition": "MyJobOnEks_Sleep",
  "eksPropertiesOverride": {
    "podProperties": {
      "containers": [
        {
          "command": [
            "/bin/sh"
          ],
          "args": [
            "-c",
            "echo hello world"
          ]
        }
      ]
    }
  }
}
EOF
$ aws batch submit-job --cli-input-json file://./submit-job-override.json
```

Hinweise

- AWS Batch bereinigt die Pods aggressiv, nachdem die Aufträge abgeschlossen sind, um die Last auf zu reduzierenKubernetes. Um die Details eines Auftrags zu überprüfen, muss die Protokollierung konfiguriert sein. Weitere Informationen finden Sie unter Verwenden von - CloudWatch Protokollen zur Überwachung AWS Batch von Amazon-EKS-Aufträgen.

- Aktivieren Sie die Amazon-EKS-Steuerebenenprotokollierung, um einen besseren Einblick in die Details der -Operationen zu erhalten. Weitere Informationen finden Sie unter Amazon-EKS-Steuerebenenprotokollierung im Amazon-EKS-Benutzerhandbuch.
- Daemonsets und kubelets Overhead wirken sich auf verfügbare vCPU- und Speicherressourcen aus, insbesondere auf Skalierung und Auftragsplatzierung. Weitere Informationen finden Sie unter Überlegungen zu Arbeitsspeicher und vCPU für AWS Batch in Amazon EKS.

Erste Schritte mit AWS Batch auf privaten Amazon-EKS-Clustern

AWS Batch ist ein verwalteter Service, der Batch-Workloads in Ihren Amazon Elastic Kubernetes Service (Amazon EKS)-Clustern orchestriert. Dazu gehören Warteschlangen, Abhängigkeitsnachverfolgung, verwaltete Auftragswiederholungen und -prioritäten, Pod-Verwaltung und Knotenskalierung. Diese Funktion verbindet Ihren vorhandenen privaten Amazon-EKS-Cluster mit AWS Batch , um Ihre Aufträge in großem Umfang auszuführen. Sie können `eksctl` (eine Befehlszeilenschnittstelle für Amazon EKS), die - AWS Konsole oder die verwenden, AWS Command Line Interface um einen privaten Amazon-EKS-Cluster mit allen anderen erforderlichen Ressourcen zu erstellen. Unterstützung für private Amazon-EKS-Cluster auf AWS Batch ist allgemein in kommerziellen verfügbar AWS-Regionen , in denen AWS Batch verfügbar ist.

Nur private Amazon-EKS-Cluster haben keinen eingehenden/ausgehenden Internetzugang und verfügen nur über private Subnetze. Amazon-VPC-Endpunkte werden verwendet, um privaten Zugriff auf andere AWS -Services zu ermöglichen. `eksctl` unterstützt das Erstellen vollständig privater Cluster mit einer bereits vorhandenen Amazon VPC und Subnetzen. erstellt `eksctl` auch Amazon-VPC-Endpunkte in der bereitgestellten Amazon VPC und ändert Routing-Tabellen für die bereitgestellten Subnetze.

Jedem Subnetz sollte eine explizite Routing-Tabelle zugeordnet sein, da die Haupt-Routing-Tabelle `eksctl` nicht ändert. Ihr Cluster muss Images aus einer Container-Registry abrufen, die sich in Ihrer Amazon VPC befindet. Außerdem können Sie eine Amazon Elastic Container Registry in Ihrer Amazon VPC erstellen und Container-Images dorthin kopieren, damit Ihre Knoten daraus abrufen können. Weitere Informationen finden Sie unter Kopieren eines Container-Images aus einem Repository in ein anderes Repository. Informationen zu den ersten Schritten mit privaten Amazon-ECR-Repositorys finden Sie unter Private Amazon-ECR-Repositorys.

Sie können optional eine Pull-Through-Cache-Regel mit Amazon ECR erstellen. Sobald eine Pull-Through-Cache-Regel für eine externe öffentliche Registrierung erstellt wurde, können Sie ein Image aus dieser externen öffentlichen Registrierung mit Ihrem privaten Amazon-ECR-

Registrierungs-Uriform-Ressourcenidetifier (URI) abrufen. Dann erstellt Amazon ECR ein Repository und speichert das Image im Cache. Wenn ein zwischengespeichertes Image mit dem URI der privaten Registrierung von Amazon ECR abgerufen wird, überprüft Amazon ECR die Remote-Registrierung, um festzustellen, ob eine neue Version des Images vorhanden ist, und aktualisiert Ihre private Registrierung bis zu einmal alle 24 Stunden.

Inhalt

Voraussetzungen

Bevor Sie mit diesem Tutorial beginnen, müssen Sie die folgenden Tools und Ressourcen installieren und konfigurieren, die Sie zum Erstellen und Verwalten von AWS Batch und Amazon-EKS-Ressourcen benötigen. Sie müssen auch alle erforderlichen Ressourcen erstellen, einschließlich VPC, Subnetze, Routing-Tabellen, VPC-Endpunkte und Amazon-EKS-Cluster. Sie müssen die verwenden AWS CLI.

- AWS CLI – Ein Befehlszeilen-Tool für die Arbeit mit - AWS Services, einschließlich Amazon EKS. Dieses Handbuch erfordert, dass Sie Version 2.8.6 oder höher oder 1.26.0 oder höher verwenden. Weitere Informationen finden Sie unter Installieren, Aktualisieren und Deinstallieren der AWS CLI im AWS Command Line Interface -Benutzerhandbuch.

  Nach der Installation der empfehlen wir AWS CLI Ihnen, sie zu konfigurieren. Weitere Informationen finden Sie unter Schnellkonfiguration mit `aws configure` im AWS Command Line Interface -Benutzerhandbuch.
- **`kubectl`** – Ein Befehlszeilen-Tool für die Arbeit mit Kubernetes Clustern. Dieses Handbuch erfordert, dass Sie Version `1.23` oder höher verwenden. Weitere Informationen finden Sie unter Installieren oder Aktualisieren von `kubectl` im Amazon-EKS-Benutzerhandbuch.

- **`eksctl`** – Ein Befehlszeilen-Tool für die Arbeit mit Amazon-EKS-Clustern, das viele einzelne Aufgaben automatisiert. Dieses Handbuch erfordert, dass Sie Version `0.115.0` oder höher verwenden. Weitere Informationen finden Sie unter Installieren oder Aktualisieren von `eksctl` im Amazon-EKS-Benutzerhandbuch.
- Erforderliche AWS Identity and Access Management (IAM)-Berechtigungen – Der von Ihnen verwendete IAM-Sicherheitsprinzipal muss über Berechtigungen zum Arbeiten mit Amazon-EKS-IAM-Rollen und serviceverknüpften Rollen AWS CloudFormation sowie einer VPC und zugehörigen Ressourcen verfügen. Weitere Informationen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon Elastic Kubernetes Service und Verwenden von serviceverknüpften Rollen im IAM-Benutzerhandbuch. Sie müssen alle Schritte in diesem Handbuch als derselbe Benutzer ausführen.
- Erstellen eines Amazon-EKS-Clusters – Weitere Informationen finden Sie unter Erste Schritte mit Amazon EKS – `eksctl` im Amazon-EKS-Benutzerhandbuch.

  > **Note**
  >
  > AWS Batch bietet keine Orchestrierung für verwaltete Knoten für CoreDNS oder andere Bereitstellungs-Pods. Wenn Sie CoreDNS benötigen, finden Sie weitere Informationen unter Hinzufügen des CoreDNS-Amazon-EKS-Add-ons im Amazon-EKS-Benutzerhandbuch. Oder verwenden Sie , `eksctl create cluster create` um den Cluster zu erstellen, er enthält standardmäßig CoreDNS.

- Berechtigungen – Benutzer, die den CreateComputeEnvironment -API-Vorgang aufrufen, um eine Datenverarbeitungsumgebung zu erstellen, die Amazon-EKS-Ressourcen verwendet, benötigen Berechtigungen für den `eks:DescribeCluster`-API-Vorgang. Die Verwendung der AWS Management Console zum Erstellen einer Rechenressource mit Amazon-EKS-Ressourcen erfordert Berechtigungen für `eks:DescribeCluster` und `eks:ListClusters`.
- Erstellen Sie mithilfe der `eksctl` Beispielkonfigurationsdatei einen privaten EKS-Cluster in der Region us-east-1.

```
kind: ClusterConfig
apiVersion: eksctl.io/v1alpha5
availabilityZones:
    - us-east-1a
    - us-east-1b
    - us-east-1d
managedNodeGroups:
    privateNetworking: true
```

```
privateCluster:
    enabled: true
    skipEndpointCreation: false
```

Erstellen Sie Ihre -Ressourcen mit dem -Befehl: `eksctl create cluster -f clusterConfig.yaml`

- Batchverwaltete Knoten müssen in Subnetzen bereitgestellt werden, die über die von Ihnen benötigten VPC-Schnittstellenendpunkte verfügen. Weitere Informationen finden Sie unter Anforderungen an private Cluster.

Schritt 1: Vorbereiten Ihres EKS-Clusters für AWS Batch

Alle Schritte sind erforderlich.

1. Erstellen eines dedizierten Namespace für AWS Batch Aufträge

   Verwenden Sie `kubectl`, um einen neuen Namespace zu erstellen.

   ```
   $ namespace=my-aws-batch-namespace
   $ cat - <<EOF | kubectl create -f -
   {
     "apiVersion": "v1",
     "kind": "Namespace",
     "metadata": {
       "name": "${namespace}",
       "labels": {
         "name": "${namespace}"
       }
     }
   }
   EOF
   ```

   Ausgabe:

   ```
   namespace/my-aws-batch-namespace created
   ```

2. Aktivieren des Zugriffs über die rollenbasierte Zugriffskontrolle (RBAC)

   Verwenden Sie `kubectl`, um eine Kubernetes Rolle für den Cluster zu erstellen, damit Knoten und Pods AWS Batch überwachen und die Rolle binden kann. Sie müssen dies einmal für jeden Amazon-EKS-Cluster tun.

   > **Note**
   >
   > Weitere Informationen zur Verwendung der RBAC-Autorisierung finden Sie unter Verwenden der RBAC-Autorisierung in der Kubernetes -Dokumentation.

```
$ cat - <<EOF | kubectl apply -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: aws-batch-cluster-role
rules:
  - apiGroups: [""]
    resources: ["namespaces"]
    verbs: ["get"]
  - apiGroups: [""]
    resources: ["nodes"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["configmaps"]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["apps"]
    resources: ["daemonsets", "deployments", "statefulsets", "replicasets"]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["rbac.authorization.k8s.io"]
    resources: ["clusterroles", "clusterrolebindings"]
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: aws-batch-cluster-role-binding
subjects:
```

```
- kind: User
  name: aws-batch
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: aws-batch-cluster-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

Ausgabe:

```
clusterrole.rbac.authorization.k8s.io/aws-batch-cluster-role created
clusterrolebinding.rbac.authorization.k8s.io/aws-batch-cluster-role-binding created
```

Erstellen Sie eine Rolle im Namespace-Bereich Kubernetes für , AWS Batch um Pods zu verwalten und zu Lebenszyklus-Pods zu binden. Sie müssen dies einmal für jeden eindeutigen Namespace tun.

```
$ namespace=my-aws-batch-namespace
$ cat - <<EOF | kubectl apply -f - --namespace "${namespace}"
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: aws-batch-compute-environment-role
  namespace: ${namespace}
rules:
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["create", "get", "list", "watch", "delete", "patch"]
  - apiGroups: [""]
    resources: ["serviceaccounts"]
    verbs: ["get", "list"]
  - apiGroups: ["rbac.authorization.k8s.io"]
    resources: ["roles", "rolebindings"]
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: aws-batch-compute-environment-role-binding
  namespace: ${namespace}
subjects:
```

```
- kind: User
  name: aws-batch
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: aws-batch-compute-environment-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

Ausgabe:

```
role.rbac.authorization.k8s.io/aws-batch-compute-environment-role created
rolebinding.rbac.authorization.k8s.io/aws-batch-compute-environment-role-binding
 created
```

Aktualisieren Sie die Kubernetes `aws-auth` Konfigurationszuordnung, um die vorherigen RBAC-Berechtigungen der AWS Batch serviceverknüpften Rolle zuzuordnen.

```
$ eksctl create iamidentitymapping \
    --cluster my-cluster-name \
    --arn "arn:aws:iam::<your-account>:role/AWSServiceRoleForBatch" \
    --username aws-batch
```

Ausgabe:

```
2022-10-25 20:19:57 [#]  adding identity "arn:aws:iam::<your-account>:role/
AWSServiceRoleForBatch" to auth ConfigMap
```

> **Note**
>
> Der Pfad `aws-service-role/batch.amazonaws.com/` wurde aus dem ARN der serviceverknüpften Rolle entfernt. Dies liegt an einem Problem mit der `aws-auth` Konfigurationszuordnung. Weitere Informationen finden Sie unter Rollen mit Pfaden funktionieren nicht, wenn der Pfad in ihrem ARN enthalten ist im aws-authconfigmap.

Schritt 2: Erstellen einer Amazon-EKS-Rechenumgebung

AWS Batch -Rechenumgebungen definieren Datenverarbeitungsressourcenparameter, um Ihre Batch-Workload-Anforderungen zu erfüllen. In einer verwalteten Datenverarbeitungsumgebung AWS Batch unterstützt Sie bei der Verwaltung der Kapazität und Instance-Typen der Datenverarbeitungsressourcen (Kubernetes-Knoten) in Ihrem Amazon-EKS-Cluster. Dies basiert auf der Spezifikation der Rechenressourcen, die Sie beim Erstellen der Rechenumgebung definieren. Sie können EC2-On-Demand-Instances oder EC2-Spot-Instances verwenden.

Nachdem die AWSServiceRoleForBatch serviceverknüpfte Rolle Zugriff auf Ihren Amazon-EKS-Cluster hat, können Sie - AWS Batch Ressourcen erstellen. Erstellen Sie zunächst eine Datenverarbeitungsumgebung, die auf Ihren Amazon-EKS-Cluster verweist.

```
$ cat <<EOF > ./batch-eks-compute-environment.json
{
  "computeEnvironmentName": "My-Eks-CE1",
  "type": "MANAGED",
  "state": "ENABLED",
  "eksConfiguration": {
    "eksClusterArn": "arn:aws:eks:<region>:123456789012:cluster/<cluster-name>",
    "kubernetesNamespace": "my-aws-batch-namespace"
  },
  "computeResources": {
    "type": "EC2",
    "allocationStrategy": "BEST_FIT_PROGRESSIVE",
    "minvCpus": 0,
    "maxvCpus": 128,
    "instanceTypes": [
        "m5"
    ],
    "subnets": [
        "<eks-cluster-subnets-with-access-to-the-image-for-image-pull>"
    ],
    "securityGroupIds": [
        "<eks-cluster-sg>"
    ],
    "instanceRole": "<eks-instance-profile>"
  }
}
EOF
$ aws batch create-compute-environment --cli-input-json file://./batch-eks-compute-environment.json
```

Hinweise

- Der `serviceRole` Parameter sollte nicht angegeben werden, dann wird die AWS Batch serviceverknüpfte Rolle verwendet. AWS Batch auf Amazon EKS unterstützt nur die AWS Batch serviceverknüpfte Rolle.
- Für Amazon-EKS-Rechenumgebungen werden nur `BEST_FIT_PROGRESSIVE`-`SPOT_CAPACITY_OPTIMIZED`, - und -`SPOT_PRICE_CAPACITY_OPTIMIZED`Zuweisungsstrategien unterstützt.

  > **Note**
  >
  > Wir empfehlen, in den meisten `SPOT_PRICE_CAPACITY_OPTIMIZED` Instances anstelle von `SPOT_CAPACITY_OPTIMIZED`n zu verwenden.

- Informationen zur `instanceRole`finden Sie unter Erstellen der IAM-Rolle des Amazon-EKS-Knotens und Aktivieren des IAM-Prinzipalzugriffs auf Ihren Cluster im Amazon-EKS-Benutzerhandbuch. Wenn Sie Pod-Netzwerke verwenden, finden Sie weitere Informationen unter Konfigurieren des Amazon-VPC-CNI-Plugins für Kubernetes zur Verwendung von IAM-Rollen für Servicekonten im Amazon-EKS-Benutzerhandbuch.
- Eine Möglichkeit, funktionierende Subnetze für den -`subnets`Parameter zu erhalten, besteht darin, die von erstellten öffentlichen Subnetze der von Amazon EKS verwalteten Knotengruppen zu verwenden, `eksctl` wenn ein Amazon-EKS-Cluster erstellt wurde. Verwenden Sie andernfalls Subnetze mit einem Netzwerkpfad, der das Abrufen von Images unterstützt.
- Der `securityGroupIds` Parameter kann dieselbe Sicherheitsgruppe wie der Amazon-EKS-Cluster verwenden. Dieser Befehl ruft die Sicherheitsgruppen-ID für den Cluster ab.

```
$ eks describe-cluster \
    --name <cluster-name> \
    --query cluster.resourcesVpcConfig.clusterSecurityGroupId
```

- Die Wartung einer Amazon-EKS-Rechenumgebung ist eine übergreifende Verantwortlichkeit. Weitere Informationen finden Sie unter Sicherheit in Amazon EKS.

> **Important**
>
> Es ist wichtig zu bestätigen, dass die Datenverarbeitungsumgebung fehlerfrei ist, bevor Sie fortfahren. Dazu kann der DescribeComputeEnvironments API-Vorgang verwendet werden.

```
$ aws batch describe-compute-environments --compute-environments My-Eks-CE1
```

Vergewissern Sie sich, dass der `status` Parameter nicht ist`INVALID`. Wenn dies der Fall ist, sehen Sie sich den `statusReason` Parameter für die Ursache an. Weitere Informationen finden Sie unter Problembehebung AWS Batch.

Schritt 3: Erstellen einer Auftragswarteschlange und Anfügen der Datenverarbeitungsumgebung

```
$ aws batch describe-compute-environments --compute-environments My-Eks-CE1
```

Aufträge, die an diese neue Auftragswarteschlange gesendet werden, werden als Pods auf AWS Batch verwalteten Knoten ausgeführt, die dem Amazon-EKS-Cluster beigetreten sind, der Ihrer Datenverarbeitungsumgebung zugeordnet ist.

```
$ cat <<EOF > ./batch-eks-job-queue.json
 {
    "jobQueueName": "My-Eks-JQ1",
    "priority": 10,
    "computeEnvironmentOrder": [
      {
        "order": 1,
        "computeEnvironment": "My-Eks-CE1"
      }
    ]
  }
EOF
$ aws batch create-job-queue --cli-input-json file://./batch-eks-job-queue.json
```

Schritt 4: Erstellen einer Auftragsdefinition

Anstatt einen Link zum Image in einem öffentlichen ECR-Repository bereitzustellen, geben Sie im Image-Feld der Auftragsdefinition den Link zu dem in unserem privaten ECR-Repository gespeicherten Image an. Sehen Sie sich die folgende Beispielauftragsdefinition an:

```
$ cat <<EOF > ./batch-eks-job-definition.json
{
```

```
  "jobDefinitionName": "MyJobOnEks_Sleep",
  "type": "container",
  "eksProperties": {
    "podProperties": {
      "hostNetwork": true,
      "containers": [
        {
          "image": "account-id.dkr.ecr.region.amazonaws.com/amazonlinux:2",
          "command": [
            "sleep",
            "60"
          ],
          "resources": {
            "limits": {
              "cpu": "1",
              "memory": "1024Mi"
            }
          }
        }
      ],
      "metadata": {
        "labels": {
          "environment": "test"
        }
      }
    }
  }
}
EOF
$ aws batch register-job-definition --cli-input-json file://./batch-eks-job-
definition.json
```

Um kubectl-Befehle auszuführen, benötigen Sie privaten Zugriff auf Ihren Amazon-EKS-Cluster. Dies bedeutet, dass der gesamte Datenverkehr zu Ihrem Cluster-API-Server aus der VPC Ihres Clusters oder einem verbundenen Netzwerk stammen muss.

Schritt 5: Senden eines Auftrags

```
$ aws batch submit-job - -job-queue My-Eks-JQ1 \
    - -job-definition MyJobOnEks_Sleep - -job-name My-Eks-Job1
$ aws batch describe-jobs - -job <jobId-from-submit-response>
```

Hinweise

- Es werden nur einzelne Container-Aufträge unterstützt.
- Stellen Sie sicher, dass Sie mit allen relevanten Überlegungen für die `memory` Parameter `cpu` und vertraut sind. Weitere Informationen finden Sie unter Überlegungen zu Arbeitsspeicher und vCPU für AWS Batch in Amazon EKS.
- Weitere Informationen zum Ausführen von Aufträgen auf Amazon-EKS-Ressourcen finden Sie unter Stellenangebote bei Amazon EKS.

(Optional) Senden eines Auftrags mit Überschreibungen

Dieser Auftrag überschreibt den an den Container übergebenen Befehl.

```
$ cat <<EOF > ./submit-job-override.json
{
  "jobName": "EksWithOverrides",
  "jobQueue": "My-Eks-JQ1",
  "jobDefinition": "MyJobOnEks_Sleep",
  "eksPropertiesOverride": {
    "podProperties": {
      "containers": [
        {
          "command": [
            "/bin/sh"
          ],
          "args": [
            "-c",
            "echo hello world"
          ]
        }
      ]
    }
  }
}
EOF
$ aws batch submit-job - -cli-input-json file://./submit-job-override.json
```

Hinweise

- AWS Batch bereinigt die Pods aggressiv, nachdem die Aufträge abgeschlossen sind, um die Last auf zu reduzierenKubernetes. Um die Details eines Auftrags zu überprüfen, muss die

Protokollierung konfiguriert sein. Weitere Informationen finden Sie unter Verwenden von - CloudWatch Protokollen zur Überwachung AWS Batch von Amazon-EKS-Aufträgen.

- Aktivieren Sie die Amazon-EKS-Steuerebenenprotokollierung, um einen besseren Einblick in die Details der -Operationen zu erhalten. Weitere Informationen finden Sie unter Amazon-EKS-Steuerebenenprotokollierung im Amazon-EKS-Benutzerhandbuch.
- Daemonsets und kubelets Overhead wirken sich auf verfügbare vCPU- und Speicherressourcen aus, insbesondere auf Skalierung und Auftragsplatzierung. Weitere Informationen finden Sie unter Überlegungen zu Arbeitsspeicher und vCPU für AWS Batch in Amazon EKS.

Fehlerbehebung

Wenn Knoten AWS Batch , die von gestartet werden, keinen Zugriff auf das Amazon-ECR-Repository (oder ein anderes Repository) haben, das Ihr Image speichert, können Ihre Aufträge im STARTING-Status verbleiben. Dies liegt daran, dass der Pod das Image nicht herunterladen und Ihren AWS Batch Auftrag ausführen kann. Wenn Sie auf den von gestarteten Pod-Namen klicken, sollten AWS Batch Sie die Fehlermeldung sehen und das Problem bestätigen können. Die Fehlermeldung sollte in etwa wie folgt aussehen:

```
Failed to pull image "public.ecr.aws/amazonlinux/amazonlinux:2": rpc error: code =
Unknown desc = failed to pull and unpack image
"public.ecr.aws/amazonlinux/amazonlinux:2": failed to resolve reference
"public.ecr.aws/amazonlinux/amazonlinux:2": failed to do request: Head
"https://public.ecr.aws/v2/amazonlinux/amazonlinux/manifests/2": dial tcp: i/o timeout
```

Weitere gängige Problembehandlungsszenarien finden Sie unter Problembehandlung. AWS Batch Informationen zur Fehlerbehebung auf der Grundlage des Pod-Status finden Sie unter Wie behebe ich Probleme mit dem Pod-Status in Amazon EKS?.

Aufträge

Jobs sind die Arbeitseinheit, mit der begonnen wurde AWS Batch. Jobs können als containerisierte Anwendungen aufgerufen werden, die auf Amazon ECS-Container-Instances in einem ECS-Cluster ausgeführt werden.

Aufträge in Containern können auf ein Container-Image, einen Befehl und Parameter verweisen. Weitere Informationen finden Sie unter Jobdefinitionsparameter für ContainerProperties.

Sie können eine große Anzahl von unabhängigen, einfachen Aufträgen senden.

Themen

Einen Job einreichen

Nachdem Sie eine Jobdefinition registriert haben, können Sie sie als Job an AWS Batch eine Job-Warteschlange senden. Sie können viele der Parameter, die in der Auftragsdefinition angegeben sind, zur Laufzeit überschreiben.

So senden Sie einen Auftrag

1. Öffnen Sie die AWS Batch Konsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie in der Navigationsleiste die aus, die Sie verwenden AWS-Region möchten.
3. Wählen Sie im Navigationsbereich die Option Jobs (Aufträge) aus.
4. Wählen Sie Neuen Job einreichen aus.
5. Geben Sie unter Name einen eindeutigen Namen für Ihre Jobdefinition ein. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
6. Wählen Sie unter Jobdefinition eine bestehende Jobdefinition für Ihren Job aus. Weitere Informationen finden Sie unter Erstellen einer Auftragsdefinition mit einem Knoten .
7. Wählen Sie für Job-Warteschlange eine bestehende Job-Warteschlange aus. Weitere Informationen finden Sie unter Erstellen einer Auftragswarteschlange.
8. Wählen Sie für Jobabhängigkeiten die Option Jobabhängigkeiten hinzufügen aus.

   - Geben Sie unter Job-ID die Job-ID für alle Abhängigkeiten ein. Wählen Sie dann Jobabhängigkeiten hinzufügen aus. Ein Job kann bis zu 20 Abhängigkeiten haben. Weitere Informationen finden Sie unter Abhängigkeiten von Job.

9. (Nur Array-Aufträge) Geben Sie für Array-Größe eine Array-Größe zwischen 2 und 10.000 an.
10. (Optional) Erweitern Sie Tags und wählen Sie dann Tag hinzufügen aus, um der Ressource Tags hinzuzufügen. Geben Sie einen Schlüssel und einen optionalen Wert ein und wählen Sie dann Tag hinzufügen.
11. Wählen Sie „Nächste Seite“.
12. Gehen Sie im Abschnitt Job-Overrides wie folgt vor:

    a. (Optional) Geben Sie für Planungspriorität einen Wert zwischen 0 und 100 ein. Höhere Werte erhalten eine höhere Priorität.

    b. (Optional) Geben Sie für Auftragsversuche ein, wie oft maximal AWS Batch versucht wird, den Job in einen `RUNNABLE` Status zu versetzen. Sie können eine Zahl zwischen 1 und 10 eingeben. Weitere Informationen finden Sie unter Automatisierte Auftragswiederholungen.

    c. (Optional) Geben Sie für das Ausführungs-Timeout den Timeout-Wert (in Sekunden) ein. Das Ausführungs-Timeout ist die Zeitspanne, bis ein unvollendeter Job beendet wird. Wenn ein Versuch die Timeout-Dauer überschreitet, wird er gestoppt und erhält einen Status.

`FAILED` Weitere Informationen finden Sie unter Timeouts bei der Job. Der Mindestwert beträgt 60 Sekunden.

> **Important**
>
> Verlassen Sie sich nicht darauf, dass Jobs, die auf Fargate-Ressourcen ausgeführt werden, länger als 14 Tage laufen. Nach 14 Tagen sind die Fargate-Ressourcen möglicherweise nicht mehr verfügbar, und der Job wird wahrscheinlich beendet.

d. (Optional) Aktivieren Sie die Option Tags weitergeben, um Tags aus dem Auftrag und der Auftragsdefinition an die Amazon ECS-Aufgabe weiterzugeben.

13. Erweitern Sie Additional configuration (Zusätzliche Konfiguration).
14. (Optional) Wählen Sie unter Bedingungen für die Wiederholungsstrategie die Option Beim Beenden bewerten hinzufügen aus. Geben Sie mindestens einen Parameterwert ein und wählen Sie dann eine Aktion aus. Für jeden Satz von Bedingungen muss Aktion entweder auf „Erneut versuchen“ oder „Beenden“ gesetzt sein. Diese Aktionen bedeuten Folgendes:

    - Wiederholen — AWS Batch Wiederholt die Versuche, bis die von Ihnen angegebene Anzahl von Auftragsversuchen erreicht ist.
    - Beenden — AWS Batch beendet die Wiederholung des Auftrags.

    > **Important**
    >
    > Wenn Sie „Beim Beenden bewerten hinzufügen“ wählen, konfigurieren Sie mindestens einen Parameter und wählen Sie entweder eine Aktion oder „Beim Beenden auswerten entfernen“.

15. Wählen Sie unter Parameter die Option Parameter hinzufügen aus, um Platzhalter für die Parametersetzung hinzuzufügen. Geben Sie dann einen Schlüssel und einen optionalen Wert ein.
16. Gehen Sie im Abschnitt Container-Überschreibungen wie folgt vor:

    a. Geben Sie unter Befehl den Befehl an, der an den Container übergeben werden soll. Bei einfachen Befehlen geben Sie den Befehl wie bei einer Befehlszeile ein. Verwenden Sie für kompliziertere Befehle (z. B. mit Sonderzeichen) die JSON-Syntax.

> **Note**
>
> Dieser Parameter darf keine leere Zeichenfolge enthalten.

b. Geben Sie für vCPUs die Anzahl der vCPUs ein, die für den Container reserviert werden sollen. Dieser Parameter ordnet zu `CpuShares` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--cpu-shares` für die docker run zu. Jede vCPU entspricht 1.024 CPU-Anteilen. Sie müssen mindestens eine vCPU angeben.

c. Geben Sie für Speicher das Speicherlimit ein, das für den Container verfügbar ist. Wenn Ihr Container versucht, den hier angegebenen Speicher zu überschreiten, wird der Container gestoppt. Dieser Parameter ordnet zu `Memory` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--memory` für die docker run zu. Sie müssen mindestens 4 MB Arbeitsspeicher für einen Auftrag festlegen.

> **Note**
>
> Um Ihre Ressourcennutzung zu maximieren, priorisieren Sie den Arbeitsspeicher für Jobs eines bestimmten Instance-Typs. Weitere Informationen finden Sie unter DatenverarbeitungsressourceSpeicherverwaltung.

d. (Optional) Wählen Sie unter Anzahl der GPUs die Anzahl der GPUs aus, die für den Container reserviert werden sollen.

e. (Optional) Wählen Sie für Umgebungsvariablen die Option Umgebungsvariable hinzufügen aus, um Umgebungsvariablen als Name-Wert-Paare hinzuzufügen. Diese Variablen werden an den Container übergeben.

f. Wählen Sie Nächste Seite.

g. Überprüfen Sie für Job review die Konfigurationsschritte. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Jobdefinition erstellen.

Auftragsstatus

Wenn Sie einen AWS Batch Job an eine Auftragswarteschlange weiterleiten, wechselt der Job in den `SUBMITTED` Status. Danach durchläuft er die folgenden Zustände, bis er erfolgreich ist (Abschluss

mit Code 0) oder fehlschlägt (Abschluss mit einem Code ungleich null). AWS Batch -Aufträge können die folgenden Status haben:

`SUBMITTED`

Ein Job, der an die Warteschlange gesendet wurde und noch nicht vom Scheduler bewertet wurde. Der Scheduler überprüft den Auftrag, um zu bestimmen, ob offene Abhängigkeiten von der erfolgreichen Ausführung anderer Aufträge vorliegen. Wenn Abhängigkeiten vorhanden sind, wird der Auftrag in den Status `PENDING` verschoben. Sind keine Abhängigkeiten vorhanden, wechselt der Auftrag in den Status `RUNNABLE`.

`PENDING`

Ein Job, der sich in der Warteschlange befindet und aufgrund einer Abhängigkeit von einem anderen Job oder einer anderen Ressource noch nicht ausgeführt werden kann. Nachdem die Abhängigkeiten erfüllt sind, wird der Auftrag in den Status `RUNNABLE` verschoben.

`RUNNABLE`

Ein Auftrag, der sich in der Warteschlange befindet, keine ausstehenden Abhängigkeiten hat und daher für einen Host geplant werden kann. Jobs in diesem Status werden gestartet, sobald in einer der Rechenumgebungen, die der Warteschlange des Jobs zugeordnet sind, genügend Ressourcen verfügbar sind. Aufträge können jedoch für einen unbegrenzten Zeitraum in diesem Status verbleiben, wenn keine ausreichenden Ressourcen zur Verfügung stehen.

> **Note**
>
> Wenn Ihre Jobs nicht weitergeführt werden`STARTING`, finden Sie Jobs stecken in einem `RUNNABLE` Status fest im Abschnitt zur Fehlerbehebung weitere Informationen.

`STARTING`

Diese Aufträge wurden für einen Host geplant und die relevanten Vorgänge zur Container-Initiierung werden noch ausgeführt. Nachdem das Container-Image durch Pull-Übertragung abgerufen und der Container gestartet wurde, geht der Auftrag in den Status `RUNNING` über.

Die Dauer des Abrufs von Images, die Dauer der Fertigstellung von Amazon EKS InitContainer und die Dauer der Auflösung von Amazon ECS ContainerDependency erfolgen im Status STARTING. Die Zeit, die zum Abrufen eines Images für Ihren Job benötigt wird, entspricht der Zeit, in der sich Ihr Job im Status START befindet.

Wenn es beispielsweise drei Minuten dauert, das Bild für Ihren Job abzurufen, befindet sich Ihr Job drei Minuten lang im Status START. Wenn die Ausführung von InitContainers insgesamt zehn Minuten dauert, wird Ihr Amazon EKS-Job zehn Minuten lang in STARTING sein. Wenn Sie Amazon ECS-ContainerDependencies-Sets in Ihrem Amazon ECS-Job haben, befindet sich der Job in STARTING, bis alle Container-Abhängigkeiten (ihre Laufzeit) aufgelöst sind. STARTING ist in den Timeouts nicht enthalten; die Dauer beginnt bei RUNNING. Weitere Informationen finden Sie unter Jobstatus.

`RUNNING`

Der Job wird als Container-Job auf einer Amazon ECS-Container-Instance in einer Rechenumgebung ausgeführt. Wenn der Container des Auftrags beendet wird, bestimmt der Prozessbeendigungscode, ob der Auftrag erfolgreich war oder nicht. Der Beendigungscode `0` bedeutet eine erfolgreiche Ausführung und ein Beendigungscode ungleich Null bedeutet, dass ein Fehler aufgetreten ist. Wenn für den mit einem fehlgeschlagenen Versuch verbundene Aufträge verbleibende Versuche in der optionalen Wiederholungsstrategiekonfiguration vorhanden sind, wird der Auftrag erneut in den Status `RUNNABLE` verschoben. Weitere Informationen finden Sie unter Automatisierte Auftragswiederholungen.

> **Note**
>
> Protokolle für `RUNNING` Jobs sind unter CloudWatch Logs verfügbar. Die Protokollgruppe ist`/aws/batch/job`, und das Format des Protokollstream-Namens lautet wie folgt:*`first200CharsOfJobDefinitionName`*`/default/`*`ecs_task_id`*. Dieses Format könnte sich in future ändern.
> Nachdem ein Job den `RUNNING` Status erreicht hat, können Sie seinen Log-Stream-Namen mithilfe der DescribeJobsAPI-Operation programmgesteuert abrufen. Weitere Informationen finden Sie unter An CloudWatch Logs gesendete Protokolldaten anzeigen im Amazon CloudWatch Logs-Benutzerhandbuch. Standardmäßig laufen diese Protokolle nie ab. Sie können den Aufbewahrungszeitraum jedoch ändern. Weitere Informationen finden Sie unter Aufbewahrung von Protokolldaten in CloudWatch Protokollen ändern im Amazon CloudWatch Logs-Benutzerhandbuch.

`SUCCEEDED`

Der Auftrag wurde mit dem Beendigungscode `0` erfolgreich abgeschlossen. Der Auftragsstatus für `SUCCEEDED` Jobs wird AWS Batch für mindestens 7 Tage beibehalten.

> **Note**
>
> Protokolle für `SUCCEEDED` Jobs sind unter CloudWatch Logs verfügbar. Die Protokollgruppe ist `/aws/batch/job`, und das Format des Protokollstream-Namens lautet wie folgt: *`first200CharsOfJobDefinitionName`*`/default/`*`ecs_task_id`*. Dieses Format kann sich in future ändern.
> Nachdem ein Job den `RUNNING` Status erreicht hat, können Sie seinen Log-Stream-Namen mithilfe der DescribeJobsAPI-Operation programmgesteuert abrufen. Weitere Informationen finden Sie unter An CloudWatch Logs gesendete Protokolldaten anzeigen im Amazon CloudWatch Logs-Benutzerhandbuch. Standardmäßig laufen diese Protokolle nie ab. Sie können den Aufbewahrungszeitraum jedoch ändern. Weitere Informationen finden Sie unter Aufbewahrung von Protokolldaten in CloudWatch Protokollen ändern im Amazon CloudWatch Logs-Benutzerhandbuch.

`FAILED`

Der Auftrag wurde bei allen verfügbaren Versuchen nicht erfolgreich beendet. Der Auftragsstatus für `FAILED` Jobs wird AWS Batch für mindestens 7 Tage beibehalten.

> **Note**
>
> Protokolle für `FAILED` Jobs sind unter CloudWatch Logs verfügbar. Die Protokollgruppe ist `/aws/batch/job`, und das Format des Protokollstream-Namens lautet wie folgt: *`first200CharsOfJobDefinitionName`*`/default/`*`ecs_task_id`*. Dieses Format kann sich in future ändern.
> Nachdem ein Job den `RUNNING` Status erreicht hat, können Sie seinen Protokollstream mithilfe der DescribeJobsAPI-Operation programmgesteuert abrufen. Weitere Informationen finden Sie unter An CloudWatch Logs gesendete Protokolldaten anzeigen im Amazon CloudWatch Logs-Benutzerhandbuch. Standardmäßig laufen diese Protokolle nie ab. Sie können den Aufbewahrungszeitraum jedoch ändern. Weitere Informationen finden Sie unter Aufbewahrung von Protokolldaten in CloudWatch Protokollen ändern im Amazon CloudWatch Logs-Benutzerhandbuch.

AWS Batch Variablen der Arbeitsumgebung

AWS Batch legt spezifische Umgebungsvariablen in Container-Jobs fest. Diese Umgebungsvariablen ermöglichen eine Introspektion der Container innerhalb von Jobs. Sie können die Werte dieser Variablen in der Logik Ihrer Anwendungen verwenden. Alle Variablen, die AWS Batch gesetzt werden, beginnen mit dem `AWS_BATCH_` Präfix. Dies ist ein geschütztes Umgebungsvariablenpräfix. Sie können dieses Präfix nicht für Ihre eigenen Variablen in Jobdefinitionen oder Überschreibungen verwenden.

Die folgenden Umgebungsvariablen stehen in Auftragscontainern zur Verfügung:

`AWS_BATCH_CE_NAME`

Diese Variable ist auf den Namen der Rechenumgebung gesetzt, in der Ihr Job platziert ist.

`AWS_BATCH_JOB_ARRAY_INDEX`

Diese Variable wird nur in untergeordneten Array-Aufträgen festgelegt. Der Array-Auftrags-Index beginnt bei 0 und alle untergeordneten Aufträge erhalten eine eindeutige Indexnummer. Beispielsweise weist ein Array-Auftrag mit 10 untergeordneten Aufträge die Indexwerte 0–9 auf. Sie können diesen Indexwert dazu verwenden, zu steuern, wie Ihre untergeordneten Array-Auftragselemente differenziert werden. Weitere Informationen finden Sie unter Tutorial: Verwenden des Array-Job-Index zur Steuerung der Jobdifferenzierung.

`AWS_BATCH_JOB_ARRAY_SIZE`

Diese Variable ist auf die Größe des übergeordneten Array-Jobs gesetzt. Die Größe des übergeordneten Array-Jobs wird in dieser Variablen an den untergeordneten Array-Job übergeben.

`AWS_BATCH_JOB_ATTEMPT`

Für diese Variable wird die Auftragsversuchsnummer festgelegt. Der erste Versuch erhält die Nummer 1. Weitere Informationen finden Sie unter Automatisierte Auftragswiederholungen.

`AWS_BATCH_JOB_ID`

Diese Variable ist auf die AWS Batch Job-ID gesetzt.

`AWS_BATCH_JOB_KUBERNETES_NODE_UID`

Diese Variable wird als Kubernetes UID des Knotenobjekts festgelegt, das sich im Kubernetes-Cluster befindet, auf dem der Pod ausgeführt wird. Diese Variable ist nur für Jobs festgelegt,

die auf Amazon EKS-Ressourcen ausgeführt werden. Weitere Informationen finden Sie in der KubernetesDokumentation unter UIDs.

`AWS_BATCH_JOB_MAIN_NODE_INDEX`

Diese Variable wird nur in parallelen Aufträgen mit mehreren Knoten festgelegt. Diese Variable ist auf die Indexnummer des Hauptknotens des Auftrags festgelegt. Ihr Anwendungscode kann den mit dem `AWS_BATCH_JOB_MAIN_NODE_INDEX` `AWS_BATCH_JOB_NODE_INDEX` auf einem einzelnen Knoten vergleichen, um festzustellen, ob es sich um den Hauptknoten handelt.

`AWS_BATCH_JOB_MAIN_NODE_PRIVATE_IPV4_ADDRESS`

Diese Variable wird nur in untergeordneten Knoten für parallel Jobs mit mehreren Knoten gesetzt. Diese Variable ist auf dem Hauptknoten nicht vorhanden, sondern auf die private IPv4-Adresse des Hauptknotens des Jobs gesetzt. Der Anwendungscode Ihres untergeordneten Knotens kann diese Adresse zur Kommunikation mit dem Hauptknoten verwenden.

`AWS_BATCH_JOB_NODE_INDEX`

Diese Variable wird nur in parallelen Aufträgen mit mehreren Knoten festgelegt. Diese Variable ist auf die Indexnummer des Knotens festgelegt. Der Knotenindex beginnt bei 0 und jeder Knoten erhält eine eindeutige Indexnummer. Beispielsweise weist ein paralleler Auftrag mit mehreren Knoten und 10 untergeordneten Aufträgen die Indexwerte 0–9 auf.

`AWS_BATCH_JOB_NUM_NODES`

Diese Variable wird nur in parallelen Aufträgen mit mehreren Knoten festgelegt. Diese Variable ist auf die Anzahl der Knoten gesetzt, die Sie für Ihren Paralleljob mit mehreren Knoten angefordert haben.

`AWS_BATCH_JQ_NAME`

Für diese Variable ist der Name der Auftragswarteschlange festgelegt, an die der Auftrag übermittelt wurde.

Automatisierte Auftragswiederholungen

Sie können auf Ihre Aufträge und Auftragsdefinitionen eine Wiederholungsstrategie anwenden, mit der fehlgeschlagenen Aufträge automatisch wiederholt werden. Zu den möglichen Ausfallszenarien gehören die folgenden:

- Ein Beendigungscode ungleich Null von einem Container-Auftrag

- Ausfall oder Kündigung der Amazon EC2 EC2-Instance
- Interner AWS Servicefehler oder Ausfall

Wenn ein Job an eine Auftragswarteschlange weitergeleitet und in den `RUNNING` Status versetzt wird, der als Versuch gewertet wird. Standardmäßig erhält jeder Auftrag einen Versuch, um entweder in den Auftragsstatus `SUCCEEDED` oder `FAILED` verschoben zu werden. Sowohl die Arbeitsabläufe für die Auftragsdefinition als auch für die Auftragsübermittlung können jedoch verwendet werden, um eine Wiederholungsstrategie mit einem bis zehn Versuchen festzulegen. Wenn Evaluieren angegeben OnExit ist, kann es bis zu 5 Wiederholungsstrategien enthalten. Wenn evaluieren angegeben OnExit ist, aber keine der Wiederholungsstrategien übereinstimmt, wird der Job erneut versucht. Fügen Sie für Jobs, die nicht mit exit übereinstimmen, einen letzten Eintrag hinzu, der aus einem beliebigen Grund beendet wird. Dieses `evaluateOnExit` Objekt hat beispielsweise zwei Einträge mit der Aktion von `RETRY` und einen letzten Eintrag mit der Aktion von`EXIT`.

```
"evaluateOnExit": [
    {
        "action": "RETRY",
        "onReason": "AGENT"
    },
    {
        "action": "RETRY",
        "onStatusReason": "Task failed to start"
    },
    {
        "action": "EXIT",
        "onReason": "*"
    }
]
```

Zur Laufzeit wird die Umgebungsvariable `AWS_BATCH_JOB_ATTEMPT` auf die Nummer des entsprechenden Auftragsversuchs des Containers festgelegt. Der erste Versuch ist nummeriert`1`, und die nachfolgenden Versuche sind in aufsteigender Reihenfolge (z. B. 2, 3, 4).

Nehmen wir beispielsweise an, dass ein Jobversuch aus irgendeinem Grund fehlschlägt und die in der Konfiguration für die Wiederholung angegebene Anzahl von Versuchen größer ist als die `AWS_BATCH_JOB_ATTEMPT` Anzahl. Dann wird der Job wieder in den `RUNNABLE` Status versetzt. Weitere Informationen finden Sie unter Auftragsstatus.

> **Note**
>
> Aufträge, die storniert oder beendet wurden, werden nicht erneut versucht. Außerdem werden Jobs, die aufgrund einer ungültigen Jobdefinition fehlschlagen, nicht erneut versucht.

Weitere Informationen finden Sie unter Strategie erneut versuchenErstellen einer Auftragsdefinition mit einem Knoten , Einen Job einreichen und Fehlercodes für gestoppte Aufgaben.

Abhängigkeiten von Job

Wenn Sie einen AWS Batch Job einreichen, können Sie die Job-IDs angeben, von denen der Job abhängt. Dabei sorgt der AWS Batch -Scheduler dafür, dass Ihr Auftrag erst dann ausgeführt wird, wenn die angegebenen Abhängigkeiten erfolgreich abgeschlossen sind. Nachdem diese erfolgreich waren, wechselt der abhängige Auftrag von `PENDING` zu `RUNNABLE` und dann zu `STARTING` und `RUNNING`. Wenn eine der Auftragsabhängigkeiten fehlschlägt, wechselt der abhängige Auftrag automatisch von `PENDING` zu `FAILED`.

Beispielsweise kann Auftrag A eine Abhängigkeit von bis zu 20 anderen Aufträge haben. Diese müssen erfolgreich sein, bevor er ausgeführt werden kann. Sie können dann weitere von Auftrag A und bis zu 19 weiteren Aufträgen abhängige Aufträge senden.

Bei Array-Aufträgen können Sie eine `SEQUENTIAL`-Typabhängigkeit angeben, ohne eine Auftrags-ID anzugeben, sodass jeder untergeordnete Array-Auftrag sequentiell abgeschlossen wird (beginnend mit Index 0). Sie können auch eine `N_TO_N`-Typabhängigkeit mit einer Auftrags-ID festlegen. So muss jeder untergeordnete Index dieses Auftrags warten, bis der entsprechende untergeordnete Index jeder Abhängigkeit abgeschlossen ist. Weitere Informationen finden Sie unter Array-Jobs.

Informationen zum Einreichen eines AWS Batch Jobs mit Abhängigkeiten finden Sie unterEinen Job einreichen.

Timeouts bei der Job

Sie können ein Timeout für Ihre Aufträge so konfigurieren, dass AWS Batch den Auftrag beendet, wenn die Ausführung länger als den angegebenen Wert dauert. Sie haben beispielsweise einen Auftrag, der zu seiner Ausführung Ihres Wissens nach nur 15 Minuten in Anspruch nehmen sollte. Manchmal bleibt Ihre Anwendung in einer Schleife hängen und wird endlos ausgeführt. Sie können

also eine Zeitbeschränkung von 30 Minuten festlegen, bis der hängen gebliebene Auftrag beendet wird.

> **Important**
>
> Standardmäßig AWS Batch gibt es kein Job-Timeout. Wenn Sie kein Job-Timeout definieren, wird der Job ausgeführt, bis der Container beendet wird.

Sie geben entweder in Ihrer Auftragsdefinition oder bei der Übermittlung des Auftrags einen `attemptDurationSeconds`-Parameter an, der einen Mindestwert von 60 Sekunden haben muss. Wenn diese Anzahl von Sekunden nach dem `startedAt` Zeitstempel des Auftragsversuchs vergangen ist, wird der Job AWS Batch beendet. Auf der Datenverarbeitungsressource empfängt Ihr Auftrags-Container ein `SIGTERM`-Signal, um Ihrer Anwendung die Möglichkeit zu geben, ordnungsgemäß zu schließen. Wenn sich der Container nach 30 Sekunden noch immer in der Ausführung befindet, wird ein `SIGKILL`-Signal gesendet, um den Container zwangsweise herunterzufahren.

Timeout-Beendigungen werden auf einer Best-Effort-Basis abgewickelt. Sie sollten nicht erwarten, dass Ihr Timeout genau dann beendet wird, wenn das Zeitlimit für den Jobversuch überschritten wird (es kann einige Sekunden länger dauern). Wenn Ihre Anwendung eine genaue Timeout-Ausführung benötigt, sollten Sie diese Logik innerhalb der Anwendung implementieren. Wenn Sie eine große Anzahl von Aufträgen haben, bei denen gleichzeitig Zeitüberschreitungen auftreten, werden die Timeout-Beendigungen in einer FIFO(First-in-First-out)-Warteschlange abgearbeitet, wobei Aufträge stapelweise beendet werden.

> **Note**
>
> Es gibt keinen maximalen Timeout-Wert für einen AWS Batch Job.

Wenn ein Job wegen Überschreitung der Timeoutdauer beendet wird, wird er nicht erneut versucht. Wenn ein Auftragsversuch fehlschlägt, kann er wiederholt werden, wenn Wiederholversuche aktiviert sind, und der Zeitüberschreitungs-Countdown wird für den neuen Versuch erneut gestartet.

> **Important**
>
> Jobs, die auf Fargate-Ressourcen ausgeführt werden, können nicht damit rechnen, länger als 14 Tage zu laufen. Wenn die Timeout-Dauer 14 Tage überschreitet, sind die Fargate-Ressourcen möglicherweise nicht mehr verfügbar und der Job wird beendet.

Bei Array-Aufgaben haben untergeordnete Aufträge dieselbe Timeout-Konfiguration wie der übergeordnete Auftrag.

Hinweise zum Einreichen eines AWS Batch Jobs mit einer Timeout-Konfiguration finden Sie unter. Einen Job einreichen

Stellenangebote bei Amazon EKS

Ein Job ist die kleinste Arbeitseinheit in AWS Batch. Ein AWS Batch Job auf Amazon EKS hat eine one-to-one Zuordnung zu einem Kubernetes Pod. Eine AWS Batch Jobdefinition ist eine Vorlage für einen AWS Batch Job. Wenn Sie einen AWS Batch Job einreichen, verweisen Sie auf eine Jobdefinition, geben eine Job-Warteschlange als Ziel an und geben einen Namen für einen Job an. In der Auftragsdefinition eines AWS Batch Jobs auf Amazon EKS definiert der Parameter eksProperties den Parametersatz, den ein Job AWS Batch auf Amazon EKS unterstützt. In einer SubmitJobAnfrage ermöglicht der PropertiesOverrideeks-Parameter das Überschreiben einiger gängiger Parameter. Auf diese Weise können Sie Vorlagen von Jobdefinitionen für mehrere Jobs verwenden. Wenn ein Auftrag an Ihren Amazon EKS-Cluster gesendet wird, AWS Batch wandelt er den Auftrag in einen `podspec` (`Kind: Pod`) um. Der `podspec` verwendet einige zusätzliche AWS Batch Parameter, um sicherzustellen, dass Jobs korrekt skaliert und geplant werden. AWS Batch kombiniert Labels und Taints, um sicherzustellen, dass Jobs nur auf AWS Batch verwalteten Knoten ausgeführt werden und dass andere Pods nicht auf diesen Knoten ausgeführt werden.

> **Important**
>
> - Wenn der `hostNetwork` Parameter in einer Amazon EKS-Auftragsdefinition nicht explizit festgelegt ist, wird für den Pod-Netzwerkmodus AWS Batch standardmäßig der Hostmodus verwendet. Insbesondere werden die folgenden Einstellungen angewendet: `hostNetwork=true` und`dnsPolicy=ClusterFirstWithHostNet`.
> - AWS Batch bereinigt Job-Pods, kurz nachdem ein Pod seinen Job abgeschlossen hat. Um Pod-Anwendungsprotokolle zu sehen, konfigurieren Sie einen Protokollierungsdienst

für Ihren Cluster. Weitere Informationen finden Sie unter Verwenden von - CloudWatch Protokollen zur Überwachung AWS Batch von Amazon-EKS-Aufträgen.

Ordnen Sie einen laufenden Job einem Pod und einem Knoten zu

Für einen laufenden Job wurden die `nodeName` Parameter `podName` und Parameter für den aktuellen Jobversuch festgelegt. `podProperties` Verwenden Sie den DescribeJobsAPI-Vorgang, um diese Parameter anzuzeigen.

Es folgt eine Beispielausgabe.

```
$ aws batch describe-jobs --job 2d044787-c663-4ce6-a6fe-f2baf7e51b04
{
 "jobs": [
  {
   "status": "RUNNING",
   "jobArn": "arn:aws:batch:us-east-1:123456789012:job/2d044787-c663-4ce6-a6fe-
f2baf7e51b04",
   "jobDefinition": "arn:aws:batch:us-east-1:123456789012:job-definition/
MyJobOnEks_SleepWithRequestsOnly:1",
   "jobQueue": "arn:aws:batch:us-east-1:123456789012:job-queue/My-Eks-JQ1",
   "jobId": "2d044787-c663-4ce6-a6fe-f2baf7e51b04",
   "eksProperties": {
    "podProperties": {
     "nodeName": "ip-192-168-55-175.ec2.internal",
     "containers": [
      {
       "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
       "resources": {
        "requests": {
         "cpu": "1",
         "memory": "1024Mi"
        }
       }
      }
     ],
     "podName": "aws-batch.b0aca953-ba8f-3791-83e2-ed13af39428c"
    }
   }
  }
 ]
```

```
}
```

Bei einem Job mit aktivierten Wiederholungsversuchen steht das `podName` `nodeName` Ende jedes abgeschlossenen Versuchs im `eksAttempts` Listenparameter des DescribeJobsAPI-Vorgangs. Das `podName` Ende `nodeName` des aktuell ausgeführten Versuchs befindet sich im `podProperties` Objekt.

Wie ordnet man einen laufenden Pod seinem Job zu

Ein Pod hat Beschriftungen, die das `jobId` `uuid` Ende der Rechenumgebung angeben, zu der er gehört. AWS Batch fügt Umgebungsvariablen ein, sodass die Laufzeit des Jobs auf Jobinformationen verweisen kann. Weitere Informationen finden Sie unter AWS Batch Variablen der Arbeitsumgebung. Sie können diese Informationen anzeigen, indem Sie den folgenden Befehl ausführen. Die Ausgabe sieht wie folgt aus.

```
$ kubectl describe pod aws-batch.14638eb9-d218-372d-ba5c-1c9ab9c7f2a1 -n my-aws-batch-
namespace
Name:         aws-batch.14638eb9-d218-372d-ba5c-1c9ab9c7f2a1
Namespace:    my-aws-batch-namespace
Priority:     0
Node:         ip-192-168-45-88.ec2.internal/192.168.45.88
Start Time:   Wed, 26 Oct 2022 00:30:48 +0000
Labels:       batch.amazonaws.com/compute-environment-uuid=5c19160b-
d450-31c9-8454-86cf5b30548f
              batch.amazonaws.com/job-id=f980f2cf-6309-4c77-a2b2-d83fbba0e9f0
              batch.amazonaws.com/node-uid=a4be5c1d-9881-4524-b967-587789094647
...
Status:       Running
IP:           192.168.45.88
IPs:
  IP:  192.168.45.88
Containers:
  default:
    Image:         public.ecr.aws/amazonlinux/amazonlinux:2
    ...
    Environment:
      AWS_BATCH_JOB_KUBERNETES_NODE_UID:  a4be5c1d-9881-4524-b967-587789094647
      AWS_BATCH_JOB_ID:                   f980f2cf-6309-4c77-a2b2-d83fbba0e9f0
      AWS_BATCH_JQ_NAME:                  My-Eks-JQ1
      AWS_BATCH_JOB_ATTEMPT:              1
      AWS_BATCH_CE_NAME:                  My-Eks-CE1
```

```
...
```

Funktionen, die AWS Batch Amazon EKS-Jobs unterstützen

Dies sind die AWS Batch spezifischen Funktionen, die auch für Kubernetes Jobs gelten, die auf Amazon EKS ausgeführt werden:

- Abhängigkeiten von Job
- Array-Jobs
- Timeouts bei der Job
- Automatisierte Auftragswiederholungen
- Faire Planung der Anteile

Kubernetes**`Secrets`** und **`ServiceAccounts`**

AWS Batch unterstützt Referenzierung Kubernetes `Secrets` und`ServiceAccounts`. Sie können Pods so konfigurieren, dass sie Amazon EKS-IAM-Rollen für Dienstkonten verwenden. Weitere Informationen finden Sie unter Konfiguration von Pods für die Verwendung eines Kubernetes Servicekontos im Amazon EKS-Benutzerhandbuch.

Verwandte Dokumente

- Überlegungen zu Arbeitsspeicher und vCPU für AWS Batch in Amazon EKS
- So erstellen Sie einen GPU-basierten Job auf Amazon EKS-Ressourcen
- Jobs stecken in einem `RUNNABLE` Status fest

Array-Jobs

Ein Array-Auftrag ist ein Auftrag, der gemeinsame Parameter wie Auftragsdefinition, vCPUs und Speicher nutzt. Es wird als Sammlung verwandter, aber separater Basisjobs ausgeführt, die möglicherweise auf mehrere Hosts verteilt sind und gleichzeitig ausgeführt werden. Array-Jobs sind die effizienteste Methode, um extrem parallel Jobs wie Monte-Carlo-Simulationen, parametrische Sweeps oder umfangreiche Rendering-Jobs auszuführen.

AWS Batch Array-Jobs werden genauso wie normale Jobs eingereicht. Sie geben jedoch eine Array-Größe (zwischen 2 und 10.000) an. Diese legt fest, wie viele untergeordnete Aufträge im Array

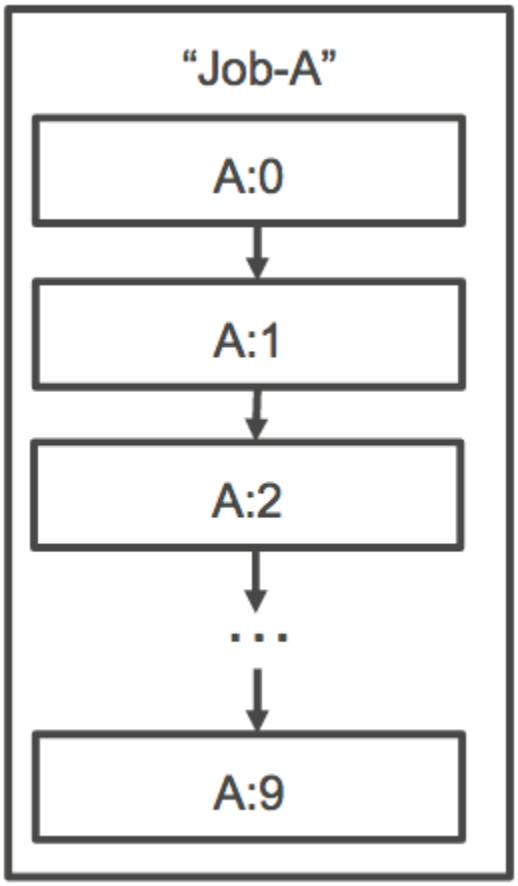
ausgeführt werden sollen. Wenn Sie einen Auftrag mit einer Array-Größe von 1000 senden, wird ein einzelner Auftrag ausgeführt und erzeugt 1000 untergeordnete Aufträge. Der Array-Auftrag ist eine Referenz oder ein Zeiger, um alle untergeordneten Aufträge zu verwalten. Auf diese Weise können Sie große Workloads mit einer einzigen Abfrage einreichen. Das im `attemptDurationSeconds` Parameter angegebene Timeout gilt für jeden untergeordneten Job. Der übergeordnete Array-Job hat kein Timeout.

Wenn Sie einen Array-Job einreichen, erhält der übergeordnete Array-Job eine normale AWS Batch Job-ID. Jeder untergeordnete Job hat dieselbe Basis-ID. Der Array-Index für den untergeordneten Job wird jedoch an das Ende der übergeordneten ID angehängt, z. B. *`example_job_ID`*`:0` für den ersten untergeordneten Job des Arrays.

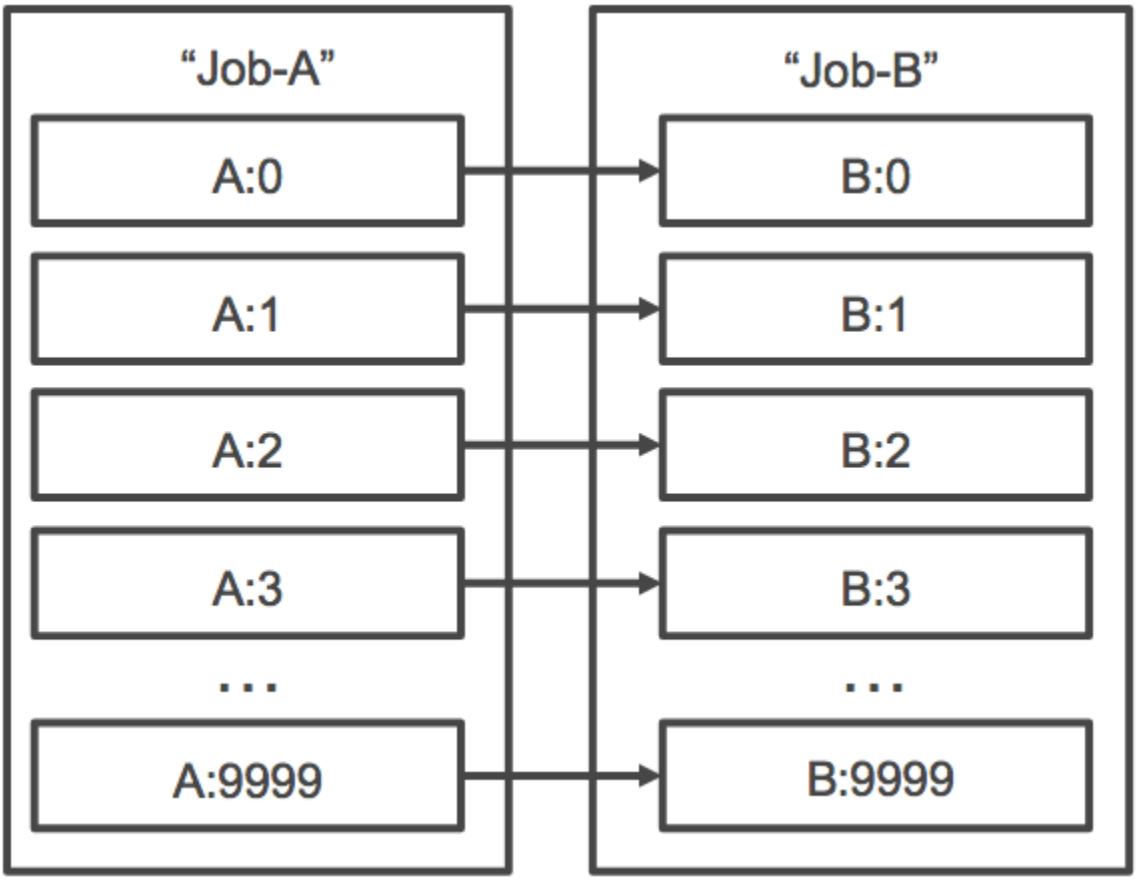
Der übergeordnete Array-Job kann den `SUCCEEDED` Status `SUBMITTEDPENDING`,`FAILED`, oder annehmen. Ein übergeordneter Array-Job wird aktualisiert`PENDING`, wenn ein untergeordneter Job aktualisiert wird`RUNNABLE`. Weitere Hinweise zu Auftragsabhängigkeiten finden Sie unterAbhängigkeiten von Job.

Zur Laufzeit wird die Umgebungsvariable `AWS_BATCH_JOB_ARRAY_INDEX` auf die Nummer des entsprechenden Auftrags-Array-Index des Containers festgelegt. Der erste Array-Auftragsindex ist nummeriert`0`, und die nachfolgenden Versuche erfolgen in aufsteigender Reihenfolge (z. B. 1, 2 und 3). Sie können diesen Indexwert dazu verwenden, zu steuern, wie Ihre untergeordneten Array-Auftragselemente differenziert werden. Weitere Informationen finden Sie unter Tutorial: Verwenden des Array-Job-Index zur Steuerung der Jobdifferenzierung.

Für Array-Auftrags-Abhängigkeiten können Sie einen Typ für eine Abhängigkeit angeben (z. B. `SEQUENTIAL` oder `N_TO_N`). Sie können eine `SEQUENTIAL`-Typabhängigkeit angeben, ohne eine Auftrags-ID anzugeben, sodass jeder untergeordnete Array-Auftrag sequentiell abgeschlossen wird (beginnend mit Index 0). Wenn Sie z. B. einen Array-Auftrag mit einer Array-Größe von 100 senden und eine Abhängigkeit vom Typ `SEQUENTIAL` angeben, werden 100 untergeordnete Aufträge sequentiell erzeugt, wobei der erste untergeordnete Auftrag erfolgreich sein muss, bevor der nächste untergeordnete Auftrag startet. Die folgende Abbildung zeigt Auftrag A, einen Array-Auftrag mit einer Array-Größe von 10. Jeder Auftrag im untergeordneten Index von Auftrag A ist vom vorherigen untergeordneten Auftrag abhängig. Auftrag A:1 kann erst gestartet werden, wenn Auftrag A:0 beendet ist.



Sie können auch eine Abhängigkeit vom `N_TO_N`-Typ mit einer Aufgaben-ID für Array-Aufgaben angeben. So muss jeder untergeordnete Index dieses Auftrags warten, bis der entsprechende untergeordnete Index jeder Abhängigkeit abgeschlossen ist. Die folgende Abbildung zeigt Job A und Job B, zwei Array-Jobs mit einer Array-Größe von jeweils 10.000. Jeder Auftrag im untergeordneten Index von Auftrag B ist abhängig von dem entsprechenden Index in Auftrag A. Auftrag B:1 kann erst gestartet werden, wenn Auftrag A:1 beendet ist.



Wenn Sie einen übergeordneten Array-Auftrag abbrechen oder beenden, werden alle untergeordneten Aufträge abgebrochen oder damit beendet. Sie können einzelne untergeordnete Aufträge abbrechen oder beenden (wodurch sie in den Status `FAILED` versetzt werden), ohne die anderen untergeordneten Aufträge zu beeinflussen. Wenn jedoch ein untergeordneter Array-Auftrag fehlschlägt (eigenständig oder durch manuelles Abbrechen oder Beenden), schlägt auch der übergeordnete Auftrag fehl.

Beispiel für einen Array-Job-Workflow

Ein üblicher Arbeitsablauf für AWS Batch Kunden besteht darin, einen vorausgesetzten Setup-Job auszuführen, eine Reihe von Befehlen für eine große Anzahl von Eingabeaufgaben auszuführen und dann mit einem Job abzuschließen, der Ergebnisse aggregiert und Zusammenfassungsdaten in Amazon S3, DynamoDB, Amazon Redshift oder Aurora schreibt.

Beispielsweise:

- `JobA`: Ein standardmäßiger Nicht-Array-Job, der eine schnelle Auflistung und Metadatenvalidierung von Objekten in einem Amazon S3 S3-Bucket durchführt,`BucketA`. Die SubmitJobJSON-Syntax lautet wie folgt.

```
{
    "jobName": "JobA",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobA-list-and-validate:1"
}
```

- `JobB`: Ein davon abhängiger Array-Job mit 10.000 Kopien`JobA`, der rechenintensive Befehle für jedes Objekt ausführt `BucketA` und Ergebnisse in dieses hochlädt. `BucketB` Die SubmitJobJSON-Syntax lautet wie folgt.

```
{
    "jobName": "JobB",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobB-CPU-Intensive-Processing:1",
    "containerOverrides": {
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "4096"
            },
            {
                "type": "VCPU",
                "value": "32"
            }
        ]
   }
    "arrayProperties": {
        "size": 10000
    },
    "dependsOn": [
        {
            "jobId": "JobA_job_ID"
  }
    ]
}
```

- `JobC`: Ein weiterer Array-Job `JobB` mit 10.000 Kopien, von dem ein `N_TO_N` Abhängigkeitsmodell abhängig ist, das speicherintensive Befehle für jedes Element ausführt`BucketB`, Metadaten

in DynamoDB schreibt und die resultierende Ausgabe dorthin hochlädt. `BucketC` Die SubmitJobJSON-Syntax lautet wie folgt.

```
{
    "jobName": "JobC",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobC-Memory-Intensive-Processing:1",
    "containerOverrides": {
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "32768"
            },
            {
                "type": "VCPU",
                "value": "1"
            }
        ]
   }
    "arrayProperties": {
        "size": 10000
    },
    "dependsOn": [
        {
            "jobId": "JobB_job_ID",
            "type": "N_TO_N"
        }
    ]
}
```

- `JobD`: Ein Array-Job, der 10 Validierungsschritte ausführt, die jeweils DynamoDB abfragen müssen und mit jedem der oben genannten Amazon S3 S3-Buckets interagieren können. Jeder der Schritte in `JobD` führt denselben Befehl aus. Das Verhalten ist jedoch je nach Wert der `AWS_BATCH_JOB_ARRAY_INDEX` Umgebungsvariablen im Container des Jobs unterschiedlich. Diese Validierungsschritte werden sequentiell ausgeführt (z. B. `JobD:0` und dann`JobD:1`). Die SubmitJobJSON-Syntax lautet wie folgt.

```
{
    "jobName": "JobD",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobD-Sequential-Validation:1",
    "containerOverrides": {
```

```
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "32768"
            },
            {
                "type": "VCPU",
                "value": "1"
            }
        ]
   }
    "arrayProperties": {
        "size": 10
    },
    "dependsOn": [
        {
            "jobId": "JobC_job_ID"
        },
        {
            "type": "SEQUENTIAL"
        },

    ]
}
```

- JobE: Ein letzter Job, der kein Array ist und einige einfache Bereinigungsvorgänge durchführt und eine Amazon SNS SNS-Benachrichtigung mit einer Meldung sendet, dass die Pipeline abgeschlossen wurde, und einem Link zur Ausgabe-URL. Die SubmitJobJSON-Syntax lautet wie folgt.

```
{
    "jobName": "JobE",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobE-Cleanup-and-Notification:1",
    "parameters": {
        "SourceBucket": "s3://JobD-Output-Bucket",
        "Recipient": "pipeline-notifications@mycompany.com"
    },
    "dependsOn": [
        {
            "jobId": "JobD_job_ID"
        }
    ]
```

```
}
```

Tutorial: Verwenden des Array-Job-Index zur Steuerung der Jobdifferenzierung

In diesem Tutorial wird beschrieben, wie Sie die `AWS_BATCH_JOB_ARRAY_INDEX` Umgebungsvariable verwenden, um die untergeordneten Jobs zu unterscheiden. Jeder untergeordnete Job ist dieser Variablen zugewiesen. In diesem Beispiel wird die Indexnummer des untergeordneten Jobs verwendet, um eine bestimmte Zeile in einer Datei zu lesen. Anschließend wird der mit dieser Zeilennummer verknüpfte Parameter durch einen Befehl im Container des Jobs ersetzt. Das Ergebnis ist, dass Sie mehrere AWS Batch Jobs haben können, die dasselbe Docker-Image und dieselben Befehlsargumente ausführen. Die Ergebnisse sind jedoch unterschiedlich, da der Array-Jobindex als Modifikator verwendet wird.

In diesem Tutorial erstellen Sie eine Textdatei, die – jeweils in einer eigenen Zeile stehend – alle Farben des Regenbogens umfasst. Anschließend erstellen Sie ein Einstiegsskript für einen Docker-Container, das den Index in einen Wert konvertiert, der für eine Zeilennummer in der Farbdatei verwendet werden kann. Der Index beginnt bei Null, aber die Zeilennummern beginnen bei Eins. Erstellen Sie ein Dockerfile, das die Farb- und Indexdateien in das Container-Image kopiert und `ENTRYPOINT` für das Bild das Entrypoint-Skript festlegt. Die Docker-Datei und die Ressourcen werden in einem Docker-Image erstellt, das an Amazon ECR übertragen wird. Anschließend registrieren Sie eine Jobdefinition, die Ihr neues Container-Image verwendet, reichen einen AWS Batch Array-Job mit dieser Jobdefinition ein und sehen sich die Ergebnisse an.

Voraussetzungen

Für dieses Tutorial müssen die folgenden Voraussetzungen erfüllt sein:

- Eine AWS Batch Rechenumgebung. Weitere Informationen finden Sie unter Eine Rechenumgebung erstellen.
- Eine AWS Batch Jobwarteschlange und eine zugehörige Rechenumgebung. Weitere Informationen finden Sie unter Erstellen einer Auftragswarteschlange.
- Die AWS CLI ist auf Ihrem lokalen System installiert. Weitere Informationen finden Sie unter Installieren der AWS Command Line Interface im AWS Command Line Interface - Benutzerhandbuch.
- Das auf dem lokalen System installierte Docker. Weitere Informationen finden Sie unter Über Docker CE in der Docker-Dokumentation.

Schritt 1: Erstellen Sie ein Container-Image

Sie können die Definition `AWS_BATCH_JOB_ARRAY_INDEX` in einer Jobdefinition im Befehlsparameter verwenden. Wir empfehlen jedoch, stattdessen ein Container-Image zu erstellen, das die Variable in einem Entrypoint-Skript verwendet. In diesem Abschnitt wird beschrieben, wie Sie ein solches Container-Image erstellen.

So erstellen Sie ein Docker-Container-Image:

1. Erstellen Sie ein neues Verzeichnis zur Verwendung als Docker-Image-Workspace und wechseln Sie dort hin.
2. Erstellen Sie eine Datei mit dem Namen `colors.txt` in Ihrem Workspace-Verzeichnis und fügen Sie Folgendes ein.

   ```
   red
   orange
   yellow
   green
   blue
   indigo
   violet
   ```

3. Erstellen Sie eine Datei mit dem Namen `print-color.sh` in Ihrem Workspace-Verzeichnis und fügen Sie Folgendes ein.

   > **Note**
   >
   > Für die `LINE`-Variable ist `AWS_BATCH_JOB_ARRAY_INDEX` + 1 festgelegt, da der Array-Index bei 0 beginnt, die Zeilennummern aber bei 1. Die `COLOR` Variable wird auf die Farbe gesetzt`colors.txt`, die ihrer Zeilennummer zugeordnet ist.

   ```
   #!/bin/sh
   LINE=$((AWS_BATCH_JOB_ARRAY_INDEX + 1))
   COLOR=$(sed -n ${LINE}p /tmp/colors.txt)
   echo My favorite color of the rainbow is $COLOR.
   ```

4. Erstellen Sie eine Datei mit dem Namen `Dockerfile` in Ihrem Workspace-Verzeichnis und fügen Sie den folgenden Inhalt ein. Diese Docker-Datei kopiert die vorherigen Dateien auf Ihren

Container und legt fest, dass das entrypoint-Skript ausgeführt wird, wenn der Container gestartet wird.

```
FROM busybox
COPY print-color.sh /tmp/print-color.sh
COPY colors.txt /tmp/colors.txt
RUN chmod +x /tmp/print-color.sh
ENTRYPOINT /tmp/print-color.sh
```

5. Erstellen Sie das Docker-Image.

```
$ docker build -t print-color .
```

6. Testen Sie den Container mit dem folgenden Skript. Dieses Skript setzt die `AWS_BATCH_JOB_ARRAY_INDEX` Variable lokal auf 0 und erhöht sie dann, um zu simulieren, was ein Array-Job mit sieben untergeordneten Objekten bewirkt.

```
$ AWS_BATCH_JOB_ARRAY_INDEX=0
while [ $AWS_BATCH_JOB_ARRAY_INDEX -le 6 ]
do
    docker run -e AWS_BATCH_JOB_ARRAY_INDEX=$AWS_BATCH_JOB_ARRAY_INDEX print-color
    AWS_BATCH_JOB_ARRAY_INDEX=$((AWS_BATCH_JOB_ARRAY_INDEX + 1))
done
```

Im Folgenden wird die Ausgabe dargestellt.

```
My favorite color of the rainbow is red.
My favorite color of the rainbow is orange.
My favorite color of the rainbow is yellow.
My favorite color of the rainbow is green.
My favorite color of the rainbow is blue.
My favorite color of the rainbow is indigo.
My favorite color of the rainbow is violet.
```

Schritt 2: Pushen Sie Ihr Bild auf Amazon ECR

Nachdem Sie Ihren Docker-Container erstellt und getestet haben, können Sie ihn in ein Image-Repository verschieben. In diesem Beispiel wird Amazon ECR verwendet, Sie können jedoch auch eine andere Registrierung verwenden, z. B. DockerHub

1. Erstellen Sie ein Amazon ECR-Image-Repository, um Ihr Container-Image zu speichern. In diesem Beispiel wird nur der verwendet AWS CLI, Sie können aber auch den AWS Management Console verwenden. Weitere Informationen finden Sie unter Creating a Repository im Amazon Elastic Container Registry-Benutzerhandbuch.

   ```
   $ aws ecr create-repository --repository-name print-color
   ```

2. Kennzeichnen Sie Ihr `print-color` Bild mit Ihrer Amazon ECR-Repository-URI, die im vorherigen Schritt zurückgegeben wurde.

   ```
   $ docker tag print-color aws_account_id.dkr.ecr.region.amazonaws.com/print-color
   ```

3. Melden Sie sich bei Ihrer Amazon ECR-Registrierung an. Weitere Informationen finden Sie unter Registry-Authentifizierung im Benutzerhandbuch zu Amazon-Elastic-Container-Registry.

   ```
   $ aws ecr get-login-password \
       --region region | docker login \
       --username AWS \
       --password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
   ```

4. Übertragen Sie Ihr Bild auf Amazon ECR.

   ```
   $ docker push aws_account_id.dkr.ecr.region.amazonaws.com/print-color
   ```

Schritt 3: Erstellen und registrieren Sie eine Jobdefinition

Da sich Ihr Docker-Image nun in einer Image-Registry befindet, können Sie es in einer AWS Batch Jobdefinition angeben. Anschließend können Sie es später verwenden, um einen Array-Job auszuführen. In diesem Beispiel wird nur der verwendet AWS CLI. Sie können jedoch auch die verwenden AWS Management Console. Weitere Informationen finden Sie unter Erstellen einer Auftragsdefinition mit einem Knoten .

So erstellen Sie eine Auftragsdefinition:

1. Erstellen Sie eine Datei mit dem Namen `print-color-job-def.json` in Ihrem Workspace-Verzeichnis und fügen Sie Folgendes ein. Ersetzen Sie den Image-Repository-URI durch den URI des Bildes.

   ```
   {
   ```

```
  "jobDefinitionName": "print-color",
  "type": "container",
  "containerProperties": {
    "image": "aws_account_id.dkr.ecr.region.amazonaws.com/print-color",
    "resourceRequirements": [
        {
            "type": "MEMORY",
            "value": "250"
        },
        {
            "type": "VCPU",
            "value": "1"
        }
    ]
  }
}
```

2. Registrieren Sie die Jobdefinition bei AWS Batch.

```
$ aws batch register-job-definition --cli-input-json file://print-color-job-def.json
```

Schritt 4: Reichen Sie einen AWS Batch Array-Job ein

Nachdem Sie Ihre Jobdefinition registriert haben, können Sie einen AWS Batch Array-Job einreichen, der Ihr neues Container-Image verwendet.

Um einen AWS Batch Array-Job einzureichen

1. Erstellen Sie eine Datei mit dem Namen `print-color-job.json` in Ihrem Workspace-Verzeichnis und fügen Sie Folgendes ein.

> **Note**
>
> In diesem Beispiel wird die im the section called “Voraussetzungen” Abschnitt erwähnte Job-Warteschlange verwendet.

```
{
  "jobName": "print-color",
```

```
  "jobQueue": "existing-job-queue",
  "arrayProperties": {
    "size": 7
  },
  "jobDefinition": "print-color"
}
```

2. Reichen Sie den Job in Ihre AWS Batch Job-Warteschlange ein. Notieren Sie sich die Job-ID, die in der Ausgabe zurückgegeben wird.

```
$ aws batch submit-job --cli-input-json file://print-color-job.json
```

3. Beschreiben Sie den Status des Auftrags und warten Sie, bis der Auftrag den Status `SUCCEEDED` erhält.

Schritt 5: Zeigen Sie Ihre Array-Jobprotokolle an

Nachdem Ihr Job den `SUCCEEDED` Status erreicht hat, können Sie die CloudWatch Protokolle im Container des Jobs einsehen.

Um die Logs Ihres Jobs in CloudWatch Logs einzusehen

1. Öffnen Sie die AWS Batch Konsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie im linken Navigationsbereich Aufträge aus.
3. Wählen Sie für Auftragswarteschlange eine Warteschlange aus.
4. Wählen Sie im Bereich Status die Option erfolgreich aus.
5. Zum Anzeigen aller untergeordneten Aufträge für den Array-Auftrag wählen Sie die Auftrags-ID aus, die im vorherigen Abschnitt zurückgegeben wurde.
6. Zum Anzeigen der Protokolle aus dem Container des Auftrags wählen Sie einen der untergeordneten Aufträge und dann die Option Protokolle anzeigen aus.

7. Zeigen Sie die Protokolle der anderen untergeordneten Protokolle an. Jeder Auftrag gibt eine andere Farbe des Regenbogens zurück.

parallel Jobs mit mehreren Knoten

Sie können parallel Jobs mit mehreren Knoten verwenden, um einzelne Jobs auszuführen, die sich über mehrere Amazon EC2 EC2-Instances erstrecken. Mit parallel Aufträgen mit AWS Batch mehreren Knoten können Sie umfangreiche Hochleistungsrechnungsanwendungen und das Training verteilter GPU-Modelle ausführen, ohne Amazon EC2 EC2-Ressourcen direkt starten, konfigurieren und verwalten zu müssen. Ein parallel Job mit AWS Batch mehreren Knoten ist mit jedem Framework kompatibel, das IP-basierte Internode-Kommunikation unterstützt. Beispiele hierfür sind Apache MXNet TensorFlow, Caffe2 oder Message Passing Interface (MPI).

Parallele Aufträge mit mehreren Knoten werden als ein einzelner Auftrag übermittelt. Ihre Auftragsdefinition (oder Überschreibungen durch den Auftragsübergabeknoten) legt jedoch die Anzahl der Knoten fest, die für den Auftrag erstellt werden müssen, und welche Knotengruppen erstellt werden müssen. Jeder parallele Auftrag mit mehreren Knoten enthält einen Hauptknoten, der zuerst gestartet wird. Sobald der Hauptknoten aktiv ist, werden die untergeordneten Knoten gestartet. Der Job ist nur abgeschlossen, wenn der Hauptknoten beendet wird. Alle untergeordneten Knoten werden dann gestoppt. Weitere Informationen finden Sie unter Knotengruppen.

parallel Jobknoten mit mehreren Knoten sind Single-Tenants. Das bedeutet, dass auf jeder Amazon EC2 EC2-Instance nur ein einziger Jobcontainer ausgeführt wird.

Der letzte Auftragsstatus (`SUCCEEDED` oder `FAILED`) wird durch den letzten Auftragsstatus des Hauptknotens bestimmt. Um den Status eines parallel Jobs mit mehreren Knoten abzurufen, beschreiben Sie den Job anhand der Job-ID, die beim Absenden des Jobs zurückgegeben wurde. Wenn Sie die Details für untergeordnete Knoten benötigen, beschreiben Sie jeden untergeordneten Knoten einzeln. Sie können Knoten anhand der #*N* Notation adressieren (beginnend mit 0). Um beispielsweise auf die Details des zweiten Knotens eines Jobs zuzugreifen, beschreiben Sie *aws_batch_job_id* #1 mithilfe der API-Operation. AWS Batch DescribeJobs Die Informationen `started`, `stoppedAt`, `statusReason` und `exit` für einen parallelen Auftrag mit mehreren Knoten stammen aus dem Hauptknoten.

Wenn Sie Auftragswiederholungen angeben, führt ein Ausfall des Hauptknotens zu einem weiteren Versuch. Ausfälle von untergeordneten Knoten führen nicht zu weiteren Versuchen. Jeder neue Versuch eines parallelen Auftrags mit mehreren Knoten aktualisiert den entsprechenden Versuch seiner zugehörigen untergeordneten Knoten.

Um parallel Jobs mit mehreren Knoten ausführen zu können AWS Batch, muss Ihr Anwendungscode die Frameworks und Bibliotheken enthalten, die für die verteilte Kommunikation erforderlich sind.

Umgebungsvariablen

Zur Laufzeit werden für jeden Knoten die Standardumgebungsvariablen konfiguriert, die alle AWS Batch Jobs erhalten. Darüber hinaus sind die Knoten mit den folgenden Umgebungsvariablen konfiguriert, die für parallel Jobs mit mehreren Knoten spezifisch sind:

`AWS_BATCH_JOB_MAIN_NODE_INDEX`

Diese Variable ist auf die Indexnummer des Hauptknotens des Auftrags festgelegt. Ihr Anwendungscode kann den mit dem `AWS_BATCH_JOB_MAIN_NODE_INDEX` `AWS_BATCH_JOB_NODE_INDEX` auf einem einzelnen Knoten vergleichen, um festzustellen, ob es sich um den Hauptknoten handelt.

`AWS_BATCH_JOB_MAIN_NODE_PRIVATE_IPV4_ADDRESS`

Diese Variable wird nur in untergeordneten Knoten für parallel Jobs mit mehreren Knoten gesetzt. Diese Variable ist auf dem Hauptknoten nicht vorhanden. Diese Variable ist auf die private IPv4-Adresse des Hauptknotens des Auftrags festgelegt. Der Anwendungscode Ihres untergeordneten Knotens kann diese Adresse zur Kommunikation mit dem Hauptknoten verwenden.

`AWS_BATCH_JOB_NODE_INDEX`

Diese Variable ist auf die Indexnummer des Knotens festgelegt. Der Knotenindex beginnt bei 0 und jeder Knoten erhält eine eindeutige Indexnummer. Beispielsweise weist ein paralleler Auftrag mit mehreren Knoten und 10 untergeordneten Aufträgen die Indexwerte 0–9 auf.

`AWS_BATCH_JOB_NUM_NODES`

Diese Variable ist auf die Anzahl der Knoten festgelegt, die Sie für Ihren parallelen Auftrag mit mehreren Knoten angefordert haben.

Knotengruppen

Eine Knotengruppe ist eine identische Gruppe von Job-Knoten, die alle dieselben Container-Eigenschaften haben. Sie können AWS Batch damit bis zu fünf verschiedene Knotengruppen für jeden Job angeben.

Jede Gruppe kann ihre eigenen Container-Images, Befehle, Umgebungsvariablen und so weiter besitzen. Sie können beispielsweise einen Job einreichen, für den eine einzelne `c5.xlarge` Instanz

für den Hauptknoten und fünf untergeordnete `c5.xlarge` Instanzknoten erforderlich sind. Jede dieser unterschiedlichen Knotengruppen kann unterschiedliche Container-Images oder Befehle angeben, die für jeden Job ausgeführt werden sollen.

Alternativ können alle Knoten in Ihrem Job eine einzelne Knotengruppe verwenden. Darüber hinaus kann Ihr Anwendungscode zwischen Knotenrollen wie dem Hauptknoten und dem untergeordneten Knoten unterscheiden. Dazu wird die `AWS_BATCH_JOB_MAIN_NODE_INDEX` Umgebungsvariable mit ihrem eigenen Wert für verglichen`AWS_BATCH_JOB_NODE_INDEX`. Sie können bis zu 1.000 Knoten in einem einzigen Job haben. Dies ist das Standardlimit für Instances in einem Amazon ECS-Cluster. Sie können eine Erhöhung dieses Limits beantragen.

> **Note**
>
> Derzeit müssen alle Knotengruppen in einem parallelen Auftrag mit mehreren Knoten den gleichen Instance-Typ verwenden.

Lebenszyklus des Job

Wenn Sie einen parallel Job mit mehreren Knoten einreichen, erhält der Job den `SUBMITTED` Status. Anschließend wartet der Job darauf, dass alle Auftragsabhängigkeiten abgeschlossen sind. Der Job wird ebenfalls in den `RUNNABLE` Status versetzt. Schließlich wird die AWS Batch Instance-Kapazität bereitgestellt, die für die Ausführung Ihres Jobs erforderlich ist, und diese Instances werden gestartet.

Jeder parallele Auftrag mit mehreren Knoten enthält einen Hauptknoten. Der Hauptknoten ist eine einzelne Unteraufgabe, die das Ergebnis des eingereichten Auftrags mit mehreren Knoten AWS Batch überwacht und bestimmt. Der Hauptknoten wird zum ersten Mal gestartet und wechselt in den `STARTING`-Status. Der im `attemptDurationSeconds` Parameter angegebene Timeout-Wert gilt für den gesamten Job und nicht für die Knoten.

Wenn der Hauptknoten den `RUNNING` Status erreicht, nachdem der Container des Knotens ausgeführt wurde, werden die untergeordneten Knoten gestartet und nehmen ebenfalls diesen `STARTING` Status an. Die untergeordneten Knoten werden in zufälliger Reihenfolge angezeigt. Es gibt keine Garantien hinsichtlich des Zeitpunkts oder der Reihenfolge, in der die untergeordneten Knoten starten. Um sicherzustellen, dass sich alle Knoten der Jobs im `RUNNING` Status befinden, nachdem der Container des Knotens ausgeführt wurde, kann Ihr Anwendungscode die AWS Batch API abfragen, um Informationen über den Hauptknoten und den untergeordneten Knoten abzurufen. Alternativ kann der Anwendungscode warten, bis alle Knoten online sind, bevor

eine verteilte Verarbeitungsaufgabe gestartet wird. Die private IP-Adresse des Hauptknotens ist als `AWS_BATCH_JOB_MAIN_NODE_PRIVATE_IPV4_ADDRESS`-Umgebungsvariable in jedem untergeordneten Knoten verfügbar. Ihr Anwendungscode kann diese Informationen verwenden, um Daten und die Kommunikation zwischen den einzelnen Aufgaben zu koordinieren.

Wenn einzelne Knoten beendet werden, wechseln sie je nach ihrem Beendigungscode zu `SUCCEEDED` oder `FAILED`. Wenn der Hauptknoten beendet wird, gilt der Auftrag als abgeschlossen und alle seine untergeordneten Knoten werden beendet. Wenn ein untergeordneter Knoten ausfällt, werden AWS Batch keine Aktionen für die anderen Knoten im Job ausgeführt. Wenn Sie nicht möchten, dass Ihr Job mit einer reduzierten Anzahl von Knoten fortgeführt wird, müssen Sie dies in Ihrem Anwendungscode berücksichtigen. Dadurch wird der Job beendet oder abgebrochen.

Überlegungen zur Rechenumgebung

Es gibt mehrere Dinge, die Sie berücksichtigen sollten, wenn Sie die Datenverarbeitungsumgebungen so konfigurieren, dass parallele Aufträge mit mehreren Knoten mit AWS Batch ausgeführt werden.

- parallel Jobs mit mehreren Knoten werden in `UNMANAGED` Computerumgebungen nicht unterstützt.
- Wenn Sie parallel Jobs mit mehreren Knoten an eine Rechenumgebung senden möchten, erstellen Sie eine Cluster-Platzierungsgruppe in einer einzigen Availability Zone und ordnen Sie sie Ihren Rechenressourcen zu. Dadurch bleiben Ihre parallel Jobs mit mehreren Knoten in einer logischen Gruppierung von Instanzen mit hohem Netzwerkflusspotenzial nahe beieinander. Weitere Informationen finden Sie unter Platzierungsgruppen im Amazon-EC2-Benutzerhandbuch.
- parallel Jobs mit mehreren Knoten werden in Computerumgebungen, die Spot-Instances verwenden, nicht unterstützt.
- AWS Batch parallel Jobs mit mehreren Knoten verwenden den Amazon `awsvpc` ECS-Netzwerkmodus, der Ihren parallel Job-Containern mit mehreren Knoten dieselben Netzwerkeigenschaften wie Amazon EC2 EC2-Instances verleiht. Jeder Container eines parallelen Auftrags mit mehreren Knoten erhält seine eigene Elastic Network-Schnittstelle, eine primäre private IP-Adresse und einen internen DNS-Hostnamen. Die Netzwerkschnittstelle wird im gleichen VPC-Subnetz erstellt wie ihre Host-Datenverarbeitungsressource. Alle Sicherheitsgruppen, die auf Ihre Datenverarbeitungsressourcen angewendet werden, werden auch darauf angewendet. Weitere Informationen finden Sie unter Task Networking with the awsvpc Network Mode im Amazon Elastic Container Service Developer Guide.
- Ihrer Datenverarbeitungsumgebung sind möglicherweise nicht mehr als fünf Sicherheitsgruppen zugeordnet.

- Der `awsvpc` Netzwerkmodus bietet keine elastischen Netzwerkschnittstellen für parallel Jobs mit mehreren Knoten und öffentlichen IP-Adressen. Um auf das Internet zuzugreifen, müssen Ihre Datenverarbeitungsressourcen in einem privaten Subnetz gestartet werden, das für die Verwendung eines NAT-Gateways konfiguriert ist. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch. Für die Kommunikation zwischen den Knoten muss die private IP-Adresse oder der DNS-Hostnamen für den Knoten verwendet werden. parallel Jobs mit mehreren Knoten, die auf Rechenressourcen in öffentlichen Subnetzen ausgeführt werden, haben keinen ausgehenden Netzwerkzugriff. Weitere Informationen zum Erstellen einer VPC mit privaten Subnetzen und einem NAT-Gateway finden Sie unter Erstellen einer Virtual Private Cloud .
- Die Elastic Network-Schnittstellen, die erstellt und an Ihre Rechenressourcen angehängt werden, können nicht manuell getrennt oder von Ihrem Konto geändert werden. Dadurch soll verhindert werden, dass versehentlich eine elastic network interface gelöscht wird, die mit einem laufenden Job verknüpft ist. Um die Elastic Network-Schnittstellen für eine Aufgabe freizugeben, beenden Sie den Auftrag.
- Ihre Datenverarbeitungsumgebung muss genügend maximale vCPUs zur Unterstützung Ihres parallelen Auftrags mit mehreren Knoten haben.
- Ihr Amazon EC2 EC2-Instance-Kontingent beinhaltet die Anzahl der Instances, die für die Ausführung Ihres Jobs erforderlich sind. Nehmen wir zum Beispiel an, dass für Ihren Job 30 Instances erforderlich sind, Ihr Konto jedoch nur 20 Instances in einer Region ausführen kann. Dann bleibt Ihr Job im `RUNNABLE` Status hängen.
- Wenn Sie in einem parallel Job mit mehreren Knoten einen Instanztyp für eine Knotengruppe angeben, muss Ihre Rechenumgebung diesen Instanztyp starten.

GPU-Aufträge

GPU-Jobs helfen Ihnen dabei, Jobs auszuführen, die die GPUs einer Instanz verwenden.

Die folgenden GPU-basierten Instance-Typen von Amazon EC2 werden unterstützt. Weitere Informationen finden Sie unter Amazon EC2 G3-Instances, Amazon EC2 G4-Instances, Amazon EC2 G5-Instances, Amazon EC2 P2-Instances, Amazon EC2 P3-Instances, Amazon EC2 P4d-Instances und Amazon EC2 P5-Instances.

Instance-Typ	GPUs	GPU-Arbeitsspeicher	vCPUs	Arbeitsspeicher	Netzwerkbandbreite
g3s.xgroß	1	8 GiB	4	30,5 GiB	10 Gbit/s
g3.4xgroß	1	8 GiB	16	122 GiB	Bis zu 10 Gbit/s
g3.8xgroß	2	16 GiB	32	244 GiB	10 Gbit/s
g3.16xgroß	4	32 GiB	64	488 GiB	25 Gbit/s
g4dn.xgroß	1	16 GiB	4	16 GiB	Bis zu 25 Gbit/s
g4dn.2xgroß	1	16 GiB	8	32 GiB	Bis zu 25 Gbit/s
g4dn.4xgroß	1	16 GiB	16	64 GiB	Bis zu 25 Gbit/s
g4dn.8xgroß	1	16 GiB	32	128 GiB	50 Gbit/s
g4dn.12xgroß	4	64 GiB	48	192 GiB	50 Gbit/s
g4dn.16xgroß	1	16 GiB	64	256 GiB	50 Gbit/s
g5.xgroß	1	24 GiB	4	16 GiB	Bis zu 10 Gbit/s
g5.2xlarge	1	24 GiB	8	32 GiB	Bis zu 10 Gbit/s
g5.4xlarge	1	24 GiB	16	64 GiB	Bis zu 25 Gbit/s
g5.8xlarge	1	24 GiB	32	128 GiB	25 Gbit/s
g5.16xlarge	1	24 GiB	64	256 GiB	25 Gbit/s
g5.12xlarge	4	96 GiB	48	192 GiB	40 GBit/s
g5.24xlarge	4	96 GiB	96	384 GiB	50 Gbit/s
g5.48xlarge	8	192 GiB	192	768 GiB	100 Gbit/s
p2.xgroß	1	12 GiB	4	61 GiB	Hoch
p2.8xgroß	8	96 GiB	32	488 GiB	10 Gbit/s

Instance-Typ	GPUs	GPU-Arbeitsspeicher	vCPUs	Arbeitsspeicher	Netzwerkbandbreite
p2.16xgroß	16	192 GiB	64	732 GiB	20 Gbit/s
p3.2xgroß	1	16 GiB	8	61 GiB	Bis zu 10 Gbit/s
p3.8xgroß	4	64 GiB	32	244 GiB	10 Gbit/s
p3.16xgroß	8	128 GiB	64	488 GiB	25 Gbit/s
p3dn.24xgroß	8	256 GiB	96	768 GiB	100 Gbit/s
p4d.24xgroß	8	320 GiB	96	1152 GiB	4 x 100 Gbit/s
p5.48xlarge	8	640 GiB	192	2 TiB	32 x 100 Gbit/s

> **Note**
>
> Nur Instance-Typen, die eine NVIDIA-GPU unterstützen und eine x86_64-Architektur verwenden, werden für GPU-Jobs in unterstützt. AWS Batch Beispielsweise werden die `G5g`Instanzfamilien `G4ad`und nicht unterstützt.

Der ResourceRequirements-Parameter für die Jobdefinition gibt die Anzahl der GPUs an, die an den Container angeheftet werden sollen. Diese Anzahl von GPUs ist für die Dauer dieses Auftrags für keinen anderen Job verfügbar, der auf dieser Instanz ausgeführt wird. Alle Instance-Typen in einer Rechenumgebung, die GPU-Jobs ausführen, müssen zu den `g5` Instance-Familien `p2` `p3p4`,`p5`,`g3`,`g3s`,`g4`,, oder gehören. Geschieht dies nicht, kann es sein, dass ein GPU-Job im `RUNNABLE` Status hängen bleibt.

Jobs, die die GPUs nicht verwenden, können auf GPU-Instanzen ausgeführt werden. Ihre Ausführung auf GPU-Instanzen kann jedoch mehr kosten als auf ähnlichen Nicht-GPU-Instanzen. Abhängig von der spezifischen vCPU, dem Arbeitsspeicher und der benötigten Zeit können diese Nicht-GPU-Jobs die Ausführung von GPU-Jobs blockieren.

So erstellen Sie einen GPU-basierten Job auf Amazon EKS-Ressourcen

In diesem Abschnitt wird beschrieben, wie Sie einen Amazon EKS-GPU-Workload ausführen AWS Batch.

Inhalt

So erstellen Sie einen GPU-basierten Kubernetes Cluster auf Amazon EKS

Bevor Sie einen GPU-basierten Kubernetes Cluster auf Amazon EKS erstellen, müssen Sie die Schritte unter abgeschlossen haben. Erste Schritte mit AWS Batch in Amazon EKS Beachten Sie außerdem Folgendes:

- AWS Batch unterstützt Instance-Typen mit NVIDIA-GPUs.
- AWS Batch Wählt standardmäßig das Amazon EKS-beschleunigte AMI mit der Kubernetes Version aus, die Ihrer Amazon EKS-Cluster-Steuerebenenversion entspricht.

```
$ cat <<EOF > ./batch-eks-gpu-ce.json
{
  "computeEnvironmentName": "My-Eks-GPU-CE1",
  "type": "MANAGED",
  "state": "ENABLED",
  "eksConfiguration": {
    "eksClusterArn": "arn:aws:eks:<region>:<account>:cluster/<cluster-name>",
    "kubernetesNamespace": "my-aws-batch-namespace"
  },
  "computeResources": {
    "type": "EC2",
    "allocationStrategy": "BEST_FIT_PROGRESSIVE",
    "minvCpus": 0,
    "maxvCpus": 1024,
    "instanceTypes": [
      "p3dn.24xlarge",
```

```
      "p4d.24xlarge"
    ],
    "subnets": [
        "<eks-cluster-subnets-with-access-to-internet-for-image-pull>"
    ],
    "securityGroupIds": [
        "<eks-cluster-sg>"
    ],
    "instanceRole": "<eks-instance-profile>"
  }
}
EOF

$ aws batch create-compute-environment --cli-input-json file://./batch-eks-gpu-ce.json
```

AWS Batch verwaltet das NVIDIA GPU-Geräte-Plugin nicht in Ihrem Namen. Sie müssen dieses Plugin in Ihrem Amazon EKS-Cluster installieren und zulassen, dass es auf die AWS Batch Knoten abzielt. Weitere Informationen finden Sie unter Enabling GPU Support in Kubernetes on GitHub.

Führen Sie die folgenden Befehle aus, um NVIDIA das Geräte-Plugin (`DaemonSet`) so zu konfigurieren, dass es auf die AWS Batch Knoten abzielt.

```
# pull nvidia daemonset spec
$ curl -O https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/v0.12.2/nvidia-device-plugin.yml
# using your favorite editor, add Batch node toleration
# this will allow the DaemonSet to run on Batch nodes
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"

$ kubectl apply -f nvidia-device-plugin.yml
```

Es wird nicht empfohlen, rechenbasierte Workloads (CPU und Arbeitsspeicher) mit GPU-basierten Workloads in derselben Kombination aus Rechenumgebung und Jobwarteschlange zu kombinieren. Das liegt daran, dass Rechenaufträge GPU-Kapazität verbrauchen können.

Führen Sie die folgenden Befehle aus, um Jobwarteschlangen anzuhängen.

```
$ cat <<EOF > ./batch-eks-gpu-jq.json
 {
```

```
    "jobQueueName": "My-Eks-GPU-JQ1",
    "priority": 10,
    "computeEnvironmentOrder": [
      {
        "order": 1,
        "computeEnvironment": "My-Eks-GPU-CE1"
      }
    ]
  }
EOF

$ aws batch create-job-queue --cli-input-json file://./batch-eks-gpu-jq.json
```

So erstellen Sie eine Amazon EKS-GPU-Auftragsdefinition

Derzeit `nvidia.com/gpu` wird nur unterstützt, und der von Ihnen festgelegte Ressourcenwert muss eine ganze Zahl sein. Sie können keine Bruchteile der GPU verwenden. Weitere Informationen finden Sie in der Dokumentation unter GPUs planen. Kubernetes

Führen Sie die folgenden Befehle aus, um eine GPU-Auftragsdefinition für Amazon EKS zu registrieren.

```
$ cat <<EOF > ./batch-eks-gpu-jd.json
{
    "jobDefinitionName": "MyGPUJobOnEks_Smi",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "hostNetwork": true,
            "containers": [
                {
                    "image": "nvcr.io/nvidia/cuda:10.2-runtime-centos7",
                    "command": ["nvidia-smi"],
                    "resources": {
                        "limits": {
                            "cpu": "1",
                            "memory": "1024Mi",
                            "nvidia.com/gpu": "1"
                        }
                    }
                }
            ]
```

```
        }
    }
}
EOF

$ aws batch register-job-definition --cli-input-json file://./batch-eks-gpu-jd.json
```

So führen Sie einen GPU-Job in Ihrem Amazon EKS-Cluster aus

Die GPU-Ressource ist nicht komprimierbar. AWS Batch erstellt eine Pod-Spezifikation für GPU-Jobs, bei der der Wert der Anfrage dem Wert der Grenzwerte entspricht. Dies ist eine Kubernetes Anforderung.

Führen Sie die folgenden Befehle aus, um einen GPU-Job einzureichen.

```
$ aws batch submit-job --job-queue My-Eks-GPU-JQ1 --job-definition MyGPUJobOnEks_Smi --
job-name My-Eks-GPU-Job

# locate information that can help debug or find logs (if using Amazon CloudWatch Logs
 with Fluent Bit)
$ aws batch describe-jobs --job <job-id> | jq '.jobs[].eksProperties.podProperties |
 {podName, nodeName}'
{
  "podName": "aws-batch.f3d697c4-3bb5-3955-aa6c-977fcf1cb0ca",
  "nodeName": "ip-192-168-59-101.ec2.internal"
}
```

Suchen und filtern Sie AWS Batch Jobs

Sie können die Jobs in einer Auftragswarteschlange mithilfe der AWS Batch Konsole auflisten. Wenn sich jedoch viele Jobs in der Auftragswarteschlange befinden, kann es schwierig sein, einen bestimmten Job zu finden.

Mithilfe von Suchen und Filtern können Sie Jobs auflisten, die den von Ihnen angegebenen Suchkriterien entsprechen.

1. Öffnen Sie die AWS Batch -Konsole.
2. Wählen Sie Jobs (Aufträge) aus.
3. Aktivieren Sie Suchen und Filtern.

> **Note**
>
> Wenn Sie mehrere Jobs haben, kann dieser Vorgang einige Minuten dauern.

4. Wählen Sie im Feld Bitte wählen Sie eine Auftragswarteschlange aus die Sie durchsuchen möchten.
5. Wählen Sie im Feld Ressourcen nach Eigenschaft oder Wert filtern eine der aufgelisteten Eigenschaften aus.
6. Wählen Sie den Operator aus, den Sie verwenden möchten. Wählen Sie beispielsweise Status =.

> **Tip**
>
> Um eine andere Eigenschaft oder einen anderen Operator zu verwenden, schließen Sie das aktuelle Kriterium. Wählen Sie dann die gewünschte Eigenschaft und den gewünschten Operator aus.

7. Geben Sie einen Eigenschaftswert ein oder wählen Sie ihn aus. Geben Sie beispielsweise einen Auftragsnamen ganz oder teilweise ein oder wählen Sie Status = AUSFÜHRBAR.
8. Wählen Sie den gewünschten Job in der gefilterten Liste aus.

> **Tip**
>
> Wenn Sie den gewünschten Job nicht sehen, blättern Sie durch die gefilterte Liste.

Job-Logs

Sie können Ihre AWS Batch Jobs so konfigurieren, dass Protokollinformationen an CloudWatch Logs gesendet werden. Auf diese Weise können Sie verschiedene Protokolle Ihrer Jobs bequem an einem zentralen Ort einsehen. Weitere Informationen finden Sie unter CloudWatch Logs verwenden mit AWS Batch.

Sie können auch Job-Logs in der AWS Batch Konsole verwenden, um einen AWS Batch Job zu überwachen oder Fehler zu beheben.

1. Öffnen Sie die AWS Batch -Konsole.

2. Wählen Sie Jobs (Aufträge) aus.
3. Wählen Sie Job Job-Warteschlange die gewünschte Job-Warteschlange aus.

> **Tip**
>
> Befinden sich mehrere Jobs in der Auftragswarteschlange, können Sie Suchen und Filtern aktivieren, um einen Job schneller zu finden. Weitere Informationen finden Sie unter Suchen und filtern Sie AWS Batch Jobs.

4. Wählen Sie unter Status den gewünschten Auftragsstatus aus.
5. Wählen Sie den gewünschten Job aus.
6. Scrollen Sie auf der Detailseite nach unten zu Job Logs.
7. Wählen Sie Protokolle abrufen aus.
8. Geben Sie für Autorisierung erforderlich ein und wählen Sie dann Autorisieren **OK**, um CloudWatch Amazon-Gebühren zu akzeptieren.

> **Note**
>
> Um Ihre Autorisierung für CloudWatch Gebühren zu widerrufen:
>
> 1. Wählen Sie im linken Navigationsbereich Berechtigungen aus.
> 2. Wählen Sie für Job-Logs die Option Bearbeiten aus.
> 3. Deaktivieren Sie das CloudWatch Kontrollkästchen Batch zur Verwendung autorisieren.
> 4. Wählen Sie Änderungen speichern aus.

9. Überprüfen Sie die Protokolldaten für den AWS Batch Job.

> **Tip**
>
> Sie können das Protokoll nach Schlüsselwörtern, Max. Ergebnissen und Sortierung filtern. Sie können auch eines der Standardzeitintervalle wählen oder ein benutzerdefiniertes Intervall erstellen, um die Ergebnisse anzupassen.

Informationen zur Job

Sie können AWS Batch Jobinformationen wie Status, Jobdefinition und Containerinformationen überprüfen.

1. Öffnen Sie die AWS Batch -Konsole.
2. Wählen Sie Jobs (Aufträge) aus.
3. Wählen Sie Job Job-Warteschlange die gewünschte Job-Warteschlange aus.

   > **Tip**
   >
   > Befinden sich mehrere Jobs in der Auftragswarteschlange, können Sie Suchen und Filtern aktivieren, um einen Job schneller zu finden. Weitere Informationen finden Sie unter Suchen und filtern Sie AWS Batch Jobs.

4. Wählen Sie den gewünschten Job aus.

> **Note**
>
> Sie können auch das AWS Command Line Interface (AWS CLI) verwenden, um Details zu einem AWS Batch Job anzuzeigen. Weitere Informationen finden Sie unter describe-jobs in der AWS CLI Befehlsreferenz.

Auftragsdefinitionen

AWS Batch -Auftragsdefinitionen geben an, wie Aufträge ausgeführt werden sollen. Während jeder Auftrag auf eine Auftragsdefinition verweisen muss, können viele der Parameter, die in der Auftragsdefinition angegeben sind, zur Laufzeit überschrieben werden.

Inhalt

Einige der Attribute in einer Auftragsdefinition sind folgende:

- Das Docker-Image, das mit dem Container in Ihrem Auftrag verwendet werden soll
- Wie viele vCPUs und wie viel Arbeitsspeicher mit dem Container verwendet werden sollen
- Der Befehl, den der Container beim Start ausführen soll
- Welche Umgebungsvariablen ggf. an den Container beim Start übergeben werden sollen
- Die jeweiligen Daten-Volumes, die mit dem Container verwendet werden sollen
- Welche (falls vorhanden) IAM-Rolle Ihr Auftrag für AWS Berechtigungen verwenden soll

Eine vollständige Beschreibung der in einer Auftragsdefinition verfügbaren Parameter finden Sie unter Jobdefinitionsparameter für ContainerProperties.

Erstellen einer Auftragsdefinition mit einem Knoten

Bevor Sie Aufträge in AWS Batch ausführen können, müssen Sie eine Auftragsdefinition erstellen. Dieser Prozess variiert geringfügig zwischen parallelen Aufträgen mit einem Knoten und mehreren

Knoten. In diesem Thema wird insbesondere beschrieben, wie Sie eine Auftragsdefinition für einen -AWS BatchAuftrag erstellen, bei dem es sich nicht um einen parallelen Auftrag mit mehreren Knoten handelt.

Sie können eine parallele Auftragsdefinition mit mehreren Knoten auf Amazon Elastic Container Service-Ressourcen erstellen. Weitere Informationen finden Sie unter the section called “Erstellen einer parallelen Auftragsdefinition mit mehreren Knoten”.

Themen

Erstellen einer Auftragsdefinition mit einem Knoten auf Amazon EC2Ressourcen

So erstellen Sie eine neue Auftragsdefinition auf Amazon EC2-Ressourcen:

1. Öffnen Sie die -AWS BatchKonsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie in der Navigationsleiste die zu AWS-Region verwendende aus.
3. Wählen Sie im linken Navigationsbereich Auftragsdefinitionen aus.
4. Wählen Sie Erstellen.
5. Wählen Sie für Orchestrierungstyp Amazon Elastic Compute Cloud (Amazon EC2) aus.
6. Deaktivieren Sie für die EC2-Plattformkonfiguration die Option Parallelverarbeitung mit mehreren Knoten aktivieren.
7. Geben Sie unter Name einen eindeutigen Namen für Ihre Auftragsdefinition ein. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
8. (Optional) Geben Sie für Ausführungs-Timeout den Timeout-Wert (in Sekunden) ein. Das Ausführungs-Timeout ist die Zeitspanne, bis ein nicht abgeschlossener Auftrag beendet wird. Wenn ein Versuch die Timeout-Dauer überschreitet, wird der Versuch gestoppt und wechselt in einen `-FAILED`Status. Weitere Informationen finden Sie unter Timeouts bei der Job. Der Mindestwert beträgt 60 Sekunden.
9. (Optional) Aktivieren Sie die Planungspriorität . Geben Sie einen Wert für die Planungspriorität zwischen 0 und 100 ein. Höhere Werte haben eine höhere Priorität.

10. (Optional) Geben Sie unter Auftragsversuche ein, wie oft AWS Batch versucht, den Auftrag in den RUNNABLE Status zu verschieben. Geben Sie eine Zahl zwischen 1 und 10 ein.
11. (Optional) Wählen Sie für Bedingungen für Wiederholungsstrategie die Option Bewertung beim Beenden hinzufügen aus. Geben Sie mindestens einen Parameterwert ein und wählen Sie dann eine Aktion aus. Für jeden Bedingungssatz muss Aktion entweder auf Wiederholen oder Beenden gesetzt werden. Diese Aktionen bedeuten Folgendes:

    - Wiederholen – versucht AWS Batch es erneut, bis die Anzahl der von Ihnen angegebenen Auftragsversuche erreicht ist.
    - Beenden – AWS Batch stoppt den erneuten Versuch des Auftrags.

    > **Important**
    >
    > Wenn Sie Bewertung beim Beenden hinzufügen auswählen, müssen Sie mindestens einen Parameter konfigurieren und entweder eine Aktion oder beim Beenden die Option Bewertung entfernen auswählen.

12. (Optional) Erweitern Sie Tags und wählen Sie dann Tag hinzufügen aus, um der Ressource Tags hinzuzufügen. Geben Sie einen Schlüssel und einen optionalen Wert ein und wählen Sie dann Tag hinzufügen aus.
13. (Optional) Aktivieren Sie Tags propagieren, um Tags aus dem Auftrag und der Auftragsdefinition an die Amazon-ECS-Aufgabe zu propagieren.
14. Wählen Sie Nächste Seite aus.
15. Im Abschnitt Container-Konfiguration:

    a. Wählen Sie für Image das Docker Image aus, das Sie für Ihren Auftrag verwenden möchten. Images in der Docker Hub-Registrierung sind standardmäßig verfügbar. Sie können auch andere Repositorys mit *`repository-url`*`/`*`image`*`:`*`tag`* angeben. Der Name kann bis zu 225 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-), Unterstriche (_), Doppelpunkte (:), Schrägstriche (/) und Zahlenzeichen (#) enthalten. Dieser Parameter wird `Image` zugeordnet im Abschnitt Create a container (Erstellen eines Containers) im Docker-Remote-API und dem `IMAGE`-Parameter von docker run.

> **Note**
>
> Docker Die Image-Architektur muss mit der Prozessorarchitektur der Rechenressourcen übereinstimmen, für die sie geplant sind. Beispielsweise können Armbasierte Docker Images nur auf -Armbasierten Rechenressourcen ausgeführt werden.

- Images in öffentlichen Amazon-ECR-Repositorys verwenden die vollständige - `registry/repository[:tag]` oder `-registry/repository[@digest]`Namenskonvention (z. B. `public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).
- Bilder in Amazon-ECR-Repositorys verwenden die vollständige `registry/repository[:tag]` Namenskonvention (z. B. *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*).
- Images in offiziellen Repositorys in Docker Hub verwenden einen einzelnen Namen (z. B. `ubuntu` oder `mongo`).
- Images in anderen Repositorys in Docker Hub sind mit einem Organisationsnamen qualifiziert (z. B `amazon/amazon-ecs-agent`).
- Image in anderen Online-Repositorys sind durch einen Domain-Namen zusätzlich qualifiziert (z. B. `quay.io/assemblyline/ubuntu`).

b. Wählen Sie für Befehlssyntax Bash oder JSON aus.

c. Geben Sie unter Befehl den Befehl an, der an den Container übergeben werden soll. Geben Sie für einfachere Befehle den Befehl wie für eine Eingabeaufforderung ein. Überprüfen Sie dann, ob das JSON Ergebnis korrekt ist, und übergeben Sie es an den Docker daemon. Verwenden Sie für kompliziertere Befehle (z. B. mit Sonderzeichen) die JSON-Syntax.

> **Tip**
>
> Wählen Sie Info, um Beispiele anzuzeigen Bash und zu JSONcodieren.

Dieser Parameter ist `Cmd` im Abschnitt Erstellen eines Containers der Docker Remote-API und dem Parameter `COMMAND` von docker run zugeordnet. Weitere Informationen zum

Docker-Parameter `CMD` finden Sie unter https://docs.docker.com/engine/reference/builder/#cmd.

> **Note**
>
> Sie können Standardwerte für die Parameterersetzung und Platzhalter in Ihrem Befehl verwenden. Weitere Informationen finden Sie unter Parameter.

d. (Optional) Geben Sie für Ausführungsrolle eine IAM-Rolle an, die den Amazon-ECS-Container-Agenten die Berechtigung erteilt, AWS API-Aufrufe in Ihrem Namen durchzuführen. Diese Funktion verwendet Amazon-ECS-IAM-Rollen für Aufgaben. Weitere Informationen finden Sie unter IAM-Rollen für die Amazon-ECS-Aufgabenausführung im Amazon Elastic Container Service-Entwicklerhandbuch.

e. Wählen Sie für die Konfiguration der Auftragsrolle eine IAM-Rolle aus, die über Berechtigungen für die AWS APIs verfügt. Diese Funktion verwendet Amazon-ECS-IAM-Rollen für Aufgaben. Weitere Informationen finden Sie unter IAM-Rollen für Aufgaben im Entwicklerhandbuch zum Amazon Elastic Container Service.

> **Note**
>
> Hier werden nur Rollen angezeigt, die über die Vertrauensstellung der Amazon Elastic Container Service-Aufgabenrolle verfügen. Weitere Informationen zum Erstellen einer IAM-Rolle für Ihre AWS Batch Aufträge finden Sie unter Erstellen einer IAM-Rolle und -Richtlinie für Ihre Aufgaben im Amazon Elastic Container Service-Entwicklerhandbuch.

16. Wählen Sie für Parameter die Option Parameter hinzufügen aus, um Platzhalter für die Parameterersetzung als Schlüssel- und optionale Wertpaare hinzuzufügen.
17. Im Abschnitt Umgebungskonfiguration:

    a. Geben Sie für vCPUs die Anzahl der vCPUs ein, die für den Container reserviert werden sollen. Dieser Parameter ordnet zu `CpuShares` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--cpu-shares` für die docker run zu. Jede vCPU entspricht 1.024 CPU-Anteilen. Sie müssen mindestens eine vCPU angeben.

    b. Geben Sie für Speicher das Speicherlimit ein, das dem Container zur Verfügung steht. Wenn Ihr Container versucht, die hier angegebene Speichermenge zu überschreiten, wird der Container gestoppt. Dieser Parameter ordnet zu `Memory` im Bereich Erstellen eines

Containers der Docker Remote API und der Option `--memory` für die docker run zu. Sie müssen mindestens 4 MB Arbeitsspeicher für einen Auftrag festlegen.

> **Note**
>
> Um Ihre Ressourcenauslastung zu maximieren, priorisieren Sie den Speicher für Aufträge eines bestimmten Instance-Typs. Weitere Informationen finden Sie unter DatenverarbeitungsressourceSpeicherverwaltung.

c. Wählen Sie unter Anzahl der GPUs die Anzahl der GPUs aus, die für den Container reserviert werden sollen.

d. (Optional) Wählen Sie für Umgebungsvariablen die Option Umgebungsvariable hinzufügen aus, um Umgebungsvariablen als Name-Wert-Paare hinzuzufügen. Diese Variablen werden an den Container übergeben.

e. (Optional) Wählen Sie für Secrets die Option Secret hinzufügen aus, um Secrets als Name-Wert-Paare hinzuzufügen. Diese Secrets werden im Container verfügbar gemacht. Weitere Informationen finden Sie unter secretOptions in Jobdefinitionsparameter für ContainerProperties.

18. Wählen Sie Nächste Seite aus.

19. Im Abschnitt Linux-Konfiguration:

    a. Für Benutzer geben Sie den Benutzernamen zur Verwendung innerhalb des Containers ein. Dieser Parameter ordnet zu `User` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--user` für die docker run zu.

    b. (Optional) Um dem Auftragscontainer erhöhte Berechtigungen auf der Host-Instance zu erteilen (ähnlich wie der `root` Benutzer), ziehen Sie den Schieberegler Privilegiert nach rechts. Dieser Parameter ordnet zu `Privileged` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--privileged` für die docker run zu.

    c. (Optional) Aktivieren Sie Init-Prozess aktivieren, um einen `-init`Prozess innerhalb des Containers auszuführen. Dieser Prozess leitet Signale weiter und nimmt Prozesse auf.

20. (Optional) Im Abschnitt Dateisystemkonfiguration:

    a. Aktivieren Sie Schreibgeschütztes Dateisystem aktivieren, um den Schreibzugriff auf das Volume zu entfernen.

    b. Geben Sie für Größe des gemeinsam genutzten Speichers die Größe (in MiB) des `/dev/shm` Volumes ein.

c. Geben Sie für Maximale Auslagerungsgröße die Gesamtmenge des Auslagerungsspeichers (in MiB) ein, die der Container verwenden kann.

d. Geben Sie für Auslagerung einen Wert zwischen 0 und 100 ein, um das Auslagerungsverhalten des Containers anzugeben. Wenn Sie keinen Wert angeben und Auslagern aktiviert ist, ist der Wert standardmäßig 60. Weitere Informationen finden Sie unter Auslagerung in Jobdefinitionsparameter für ContainerProperties.

e. (Optional) Erweitern Sie Zusätzliche Konfiguration .

f. (Optional) Wählen Sie für Tmpfs die Option tmpfs hinzufügen aus, um ein `tmpfs` Mount hinzuzufügen.

g. (Optional) Wählen Sie unter Geräte die Option Gerät hinzufügen aus, um ein Gerät hinzuzufügen:

   i. Geben Sie für Container path (Container-Pfad) den Pfad in der Container-Instance an, um das der Host-Instance zugeordnete Gerät zugänglich zu machen. Wenn Sie dieses Feld leer lassen, wird der Host-Pfad im Container verwendet.

   ii. Geben Sie für Host path (Host-Pfad) den Pfad eines Geräts in der Host-Instance an.

   iii. Wählen Sie für Berechtigungen eine oder mehrere Berechtigungen aus, die auf das Gerät angewendet werden sollen. Die verfügbaren Berechtigungen sind READ , WRITE und MVNOD .

h. (Optional) Wählen Sie für Volumes-Konfiguration die Option Volume hinzufügen aus, um eine Liste von Volumes zu erstellen, die an den Container übergeben werden sollen. Geben Sie Name und Quellpfad für das Volume ein und wählen Sie dann Volume hinzufügen aus. Sie können auch EFS aktivieren aktivieren.

i. (Optional) Wählen Sie unter Mountingpunkte die Option Konfiguration für Mountingpunkte hinzufügen aus, um Mountingpunkte für Daten-Volumes hinzuzufügen. Sie müssen das Quell-Volume und den Container-Pfad angeben. Diese Mountingpunkte werden an die Docker daemon auf einer Container-Instance übergeben. Sie können das Volume auch schreibgeschützt machen.

j. (Optional) Wählen Sie für Ulimits-Konfiguration die Option Ulimit hinzufügen aus, um einen `ulimits` Wert für den Container hinzuzufügen. Geben Sie den Namen , das weiche Limit und die Hard-Limit-Werte ein und wählen Sie dann Ulimit hinzufügen aus.

21. (Optional) Im Abschnitt Protokollierungskonfiguration:

a. Wählen Sie für Protokolltreiber den zu verwendenden Protokolltreiber aus. Weitere Informationen zu den verfügbaren Protokolltreibern finden Sie unter logDriver in Jobdefinitionsparameter für ContainerProperties.

> **Note**
>
> Standardmäßig wird der `awslogs`Protokolltreiber verwendet.

b. Wählen Sie für Optionen die Option Option Hinzufügen aus, um eine Option hinzuzufügen. Geben Sie ein Name-Wert-Paar ein und wählen Sie dann Option hinzufügen aus.

c. Wählen Sie für Secrets die Option Secret hinzufügen aus. Geben Sie ein Name-Wert-Paar ein und wählen Sie dann Secret hinzufügen, um ein Secret hinzuzufügen.

> **Tip**
>
> Weitere Informationen finden Sie unter secretOptions in Jobdefinitionsparameter für ContainerProperties.

22. Wählen Sie Nächste Seite aus.
23. Überprüfen Sie unter Überprüfung der Auftragsdefinition die Konfigurationsschritte. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Auftragsdefinition erstellen aus.

Erstellen einer Auftragsdefinition mit einem Knoten für -AWS FargateRessourcen

So erstellen Sie eine neue Auftragsdefinition für AWS Fargate Ressourcen:

1. Öffnen Sie die -AWS BatchKonsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie in der oberen Navigationsleiste die zu AWS-Region verwendende aus.
3. Wählen Sie im linken Navigationsbereich Auftragsdefinitionen aus.
4. Wählen Sie Erstellen.
5. Wählen Sie für Orchestrierungstyp Fargate aus. Weitere Informationen finden Sie unter AWS Batch auf AWS Fargate.

6. Geben Sie unter Name einen eindeutigen Namen für Ihre Auftragsdefinition ein. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
7. (Optional) Geben Sie für Ausführungs-Timeout den Timeout-Wert (in Sekunden) ein. Das Ausführungs-Timeout ist die Zeitspanne, bis ein nicht abgeschlossener Auftrag beendet wird. Wenn ein Versuch die Timeout-Dauer überschreitet, wird der Versuch gestoppt und wechselt in den `FAILED` Status . Weitere Informationen finden Sie unter Timeouts bei der Job. Der Mindestwert beträgt 60 Sekunden.
8. (Optional) Aktivieren Sie die Planungspriorität . Geben Sie einen Wert für die Planungspriorität zwischen 0 und 100 ein. Höhere Werte haben eine höhere Priorität als niedrigere Werte.
9. (Optional) Erweitern Sie Tags und wählen Sie dann Tag hinzufügen aus, um der Ressource Tags hinzuzufügen. Aktivieren Sie Tags propagieren, um Tags aus dem Auftrag und der Auftragsdefinition zu propagieren.
10. Im Abschnitt Fargate-Plattformkonfiguration:
    a. Wählen Sie für Laufzeitplattform die Architektur der Datenverarbeitungsumgebung aus.
    b. Wählen Sie für Betriebssystemfamilie das Betriebssystem für die Datenverarbeitungsumgebung aus.
    c. Wählen Sie für CPU-Architektur die vCPU-Architektur aus.
    d. Geben Sie für Fargate-Plattformversion `LATEST` oder eine bestimmte Laufzeitumgebungsversion ein.
    e. (Optional) Aktivieren Sie Öffentliche IP zuweisen, um einer Fargate-Auftragsnetzwerkschnittstelle eine öffentliche IP-Adresse zuzuweisen. Damit ein Auftrag, der in einem privaten Subnetz ausgeführt wird, ausgehenden Datenverkehr an das Internet sendet, erfordert das private Subnetz, dass ein NAT-Gateway angefügt wird, um Anfragen an das Internet weiterzuleiten. Möglicherweise möchten Sie dies tun, damit Sie Container-Images abrufen können. Weitere Informationen finden Sie unter Vernetzung von Amazon-ECS-Aufgaben im Entwicklerhandbuch zum Amazon Elastic Container Service.
    f. (Optional) Geben Sie für Flüchtiger Speicher die Menge des flüchtigen Speichers ein, die der Aufgabe zugewiesen werden soll. Der flüchtige Speicher muss zwischen 21 GiB und 200 GiB liegen. Standardmäßig werden 20 GiB flüchtiger Speicher zugewiesen, wenn Sie keinen Wert eingeben.

> **Note**
>
> Flüchtiger Speicher erfordert Fargate-Plattformversion 1.4 oder höher.

g. Geben Sie für Ausführungsrolle eine IAM-Rolle an, die dem Amazon-ECS-Container und den Fargate-Agenten die Berechtigung erteilt, AWS API-Aufrufe in Ihrem Namen durchzuführen. Diese Funktion verwendet Amazon-ECS-IAM-Rollen für Aufgabenfunktionen. Weitere Informationen, einschließlich Konfigurationsvoraussetzungen, finden Sie unter IAM-Rollen für die Amazon-ECS-Aufgabenausführung im Amazon Elastic Container Service-Entwicklerhandbuch.

h. Geben Sie unter Auftragsversuche die Häufigkeit ein, mit der AWS Batch versucht, den Auftrag in einen -RUNNABLEStatus zu verschieben. Geben Sie eine Zahl zwischen 1 und 10 ein.

i. Optional) Wählen Sie für Bedingungen für Wiederholungsstrategie die Option Bewertung beim Beenden hinzufügen aus. Geben Sie mindestens einen Parameterwert ein und wählen Sie dann eine Aktion aus. Für jeden Bedingungssatz muss Aktion entweder auf Wiederholen oder Beenden festgelegt werden. Diese Aktionen bedeuten Folgendes:

   - Wiederholen – versucht AWS Batch es erneut, bis die Anzahl der von Ihnen angegebenen Auftragsversuche erreicht ist.
   - Beenden – AWS Batch stoppt den erneuten Versuch des Auftrags.

> **Important**
>
> Wenn Sie Bewertung beim Beenden hinzufügen wählen, müssen Sie mindestens einen Parameter konfigurieren und eine Aktion oder beim Beenden die Option Bewertung entfernen auswählen.

11. Wählen Sie Next page aus.
12. Im Abschnitt Container-Konfiguration:

    a. Wählen Sie für Image das Docker-Image aus, das Sie für Ihren Auftrag verwenden möchten. Images in der Docker-Hub-Registry sind standardmäßig verfügbar. Sie können auch andere Repositorys mit *repository-url*/*image*:*tag* angeben. Der Name kann bis zu 225 Zeichen lang sein. Sie kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-), Unterstriche (_), Doppelstriche (:), Punkte (.), Schrägstriche (/) und Zahlenzeichen

(#) enthalten. Dieser Parameter wird `Image` zugeordnet im Abschnitt Create a container (Erstellen eines Containers) im Docker-Remote-API und dem `IMAGE`-Parameter von docker run.

> **Note**
>
> Docker Die Image-Architektur muss mit der Prozessorarchitektur der Rechenressourcen übereinstimmen, für die sie geplant sind. Beispielsweise können Armbasierte Docker Images nur auf -Armbasierten Rechenressourcen ausgeführt werden.

- Images in öffentlichen Amazon-ECR-Repositorys verwenden die vollständige - `registry/repository[:tag]` oder `-registry/repository[@digest]`Namenskonvention (z. B. `public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).
- Bilder in Amazon-ECR-Repositorys verwenden die vollständige `registry/repository[:tag]` Namenskonvention (z. B. *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*).
- Images in offiziellen Repositorys in Docker Hub verwenden einen einzelnen Namen (z. B. `ubuntu` oder `mongo`).
- Images in anderen Repositorys in Docker Hub sind mit einem Organisationsnamen qualifiziert (z. B `amazon/amazon-ecs-agent`).
- Image in anderen Online-Repositorys sind durch einen Domain-Namen zusätzlich qualifiziert (z. B. `quay.io/assemblyline/ubuntu`).

b. Wählen Sie für Befehlssyntax Bash oder JSON aus.

c. Geben Sie unter Befehl den Befehl an, der an den Container übergeben werden soll. Geben Sie bei einfachen Befehlen den Befehl wie bei einer Eingabeaufforderung ein und überprüfen Sie dann, ob das JSON Ergebnis korrekt ist. Sie wird an den Docker -Daemon übergeben. Verwenden Sie für kompliziertere Befehle (z. B. mit Sonderzeichen) die JSON-Syntax.

> **Tip**
>
> Wählen Sie Info, um Beispiele anzuzeigen Bash und zu JSONcodieren.

Dieser Parameter ist `Cmd` im Abschnitt Erstellen eines Containers der Docker Remote-API und dem Parameter `COMMAND` von docker run zugeordnet. Weitere Informationen zum Docker-Parameter `CMD` finden Sie unter https://docs.docker.com/engine/reference/builder/#cmd.

> **Note**
>
> Sie können Standardwerte für die Parameterersetzung und Platzhalter in Ihrem Befehl verwenden. Weitere Informationen finden Sie unter Parameter.

d. (Optional) Fügen Sie der Auftragsdefinition Parameter als Name-Wert-Zuweisungen hinzu, um die Standardwerte der Auftragsdefinition zu überschreiben. So fügen Sie einen Parameter hinzu:

   - Wählen Sie für Parameter die Option Parameter hinzufügen aus, geben Sie ein Name-Wert-Paar ein und wählen Sie dann Parameter hinzufügen aus.

> **Important**
>
> Wenn Sie Parameter hinzufügen auswählen, müssen Sie entweder mindestens einen Parameter konfigurieren oder Parameter entfernen auswählen

e. Im Abschnitt Umgebungskonfiguration:

   i. Wählen Sie für Auftragsrollenkonfiguration eine IAM-Rolle aus, die über Berechtigungen für die AWS APIs verfügt. Diese Funktion verwendet Amazon-ECS-IAM-Rollen für Aufgabenfunktionen. Weitere Informationen finden Sie unter IAM-Rollen für Aufgaben im Entwicklerhandbuch zum Amazon Elastic Container Service.

> **Note**
>
> Hier werden nur Rollen angezeigt, die über die Vertrauensstellung der Aufgabenrolle von Amazon Elastic Container Service verfügen. Weitere Informationen zum Erstellen einer IAM-Rolle für Ihre AWS Batch Aufträge finden Sie unter Erstellen einer IAM-Rolle und -Richtlinie für Ihre Aufgaben im Amazon Elastic Container Service-Entwicklerhandbuch.

ii. Geben Sie für vCPUs die Anzahl der vCPUs ein, die für den Container reserviert werden sollen. Dieser Parameter ordnet zu `CpuShares` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--cpu-shares` für die docker run zu. Jede vCPU entspricht 1.024 CPU-Anteilen. Sie müssen mindestens eine vCPU angeben.

iii. Geben Sie für Speicher das Speicherlimit ein, das für den Container verfügbar ist. Wenn Ihr Container versucht, den hier angegebenen Speicher zu überschreiten, wird der Container gestoppt. Dieser Parameter ordnet zu `Memory` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--memory` für die docker run zu. Sie müssen mindestens 4 MB Arbeitsspeicher für einen Auftrag festlegen.

Wenn Sie GuardDuty Laufzeit-Überwachung verwenden, entsteht ein geringer Speicheraufwand für den GuardDuty Sicherheitsagenten. Daher muss das Speicherlimit die Größe des GuardDuty Sicherheitsagenten enthalten. Informationen zu den Speicherlimits für GuardDuty Sicherheitsagenten finden Sie unter CPU- und Speicherlimits im GuardDuty -Benutzerhandbuch. Informationen zu den bewährten Methoden finden Sie unter Wie behebe ich Fehler aufgrund von unzureichendem Arbeitsspeicher für meine Fargate-Aufgaben nach der Aktivierung der Laufzeit-Überwachung im Amazon-ECS-Entwicklerhandbuch.

> **Note**
>
> Um Ihre Ressourcenauslastung zu maximieren, sollten Sie den Speicher für Aufträge eines bestimmten Instance-Typs zuvor nutzen. Weitere Informationen finden Sie unter DatenverarbeitungsressourceSpeicherverwaltung.

f. (Optional) Wählen Sie für Umgebungsvariablen die Option Umgebungsvariable hinzufügen aus, um Umgebungsvariablen als Name-Wert-Paare hinzuzufügen. Diese Variablen werden an den Container übergeben.

g. (Optional) Wählen Sie für Secrets die Option Secret hinzufügen aus, um Secrets als Name-Wert-Paare hinzuzufügen. Diese Secrets werden im Container verfügbar gemacht. Weitere Informationen finden Sie unter secretOptions in Jobdefinitionsparameter für ContainerProperties.

h. Wählen Sie Nächste Seite aus.

13. (Optional) Im Abschnitt Linux-Konfiguration:

a. Geben Sie für Benutzer einen Benutzernamen ein, der im Container verwendet werden soll.

b. Aktivieren Sie Init-Prozess aktivieren, um einen Init-Prozess innerhalb des Containers auszuführen. Dieser Prozess leitet Signale weiter und nimmt Prozesse auf.

c. Aktivieren Sie Schreibgeschütztes Dateisystem aktivieren, um den Schreibzugriff auf das Volume zu entfernen.

d. (Optional) Erweitern Sie Zusätzliche Konfiguration .

e. Wählen Sie für Konfiguration von Mountingpunkten die Option Konfiguration von Mountingpunkten hinzufügen aus, um Mountingpunkte für Daten-Volumes hinzuzufügen. Sie müssen das Quell-Volume und den Container-Pfad angeben. Diese Mountingpunkte werden an die Docker daemon auf einer Container-Instance übergeben.

f. Wählen Sie für Volumes-Konfiguration die Option Volume hinzufügen aus, um eine Liste von Volumes zu erstellen, die an den Container übergeben werden sollen. Geben Sie einen Name nund einen Quellpfad für das Volume ein und wählen Sie dann Volume hinzufügen aus.

g. Im Abschnitt Protokollierungskonfiguration:

   i. (Optional) Wählen Sie für Protokolltreiber den zu verwendenden Protokolltreiber aus. Weitere Informationen zu den verfügbaren Protokolltreibern finden Sie unter logDriver in Jobdefinitionsparameter für ContainerProperties.

   > **Note**
   >
   > Standardmäßig wird der `awslogs`Protokolltreiber verwendet.

   ii. (Optional) Wählen Sie für Optionen die Option Option Hinzufügen aus, um eine Option hinzuzufügen. Geben Sie ein Name-Wert-Paar ein und wählen Sie dann Option hinzufügen aus.

   iii. (Optional) Wählen Sie für Secrets die Option Secret hinzufügen aus, um ein Secret hinzuzufügen. Geben Sie dann ein Name-Wert-Paar ein und wählen Sie Secret hinzufügen aus.

   > **Tip**
   >
   > Weitere Informationen finden Sie unter secretOptions in Jobdefinitionsparameter für ContainerProperties.

14. Wählen Sie Nächste Seite aus.

15. Überprüfen Sie für Überprüfung der Auftragsdefinition die Konfigurationsschritte. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Auftragsdefinition erstellen aus.

Erstellen einer Auftragsdefinition mit einem Knoten auf Amazon-EKS-Ressourcen

So erstellen Sie eine neue Auftragsdefinition für Amazon Elastic Kubernetes Service-Ressourcen:

1. Öffnen Sie die -AWS BatchKonsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie in der oberen Navigationsleiste die zu AWS-Region verwendende aus.
3. Wählen Sie im linken Navigationsbereich Auftragsdefinitionen aus.
4. Wählen Sie Erstellen.
5. Wählen Sie für Orchestrierungstyp die Option Elastic Kubernetes Service (EKS) aus.
6. Geben Sie unter Name einen eindeutigen Namen für Ihre Auftragsdefinition ein. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
7. (Optional) Geben Sie für Ausführungs-Timeout den Timeout-Wert (in Sekunden) ein. Das Ausführungs-Timeout ist die Zeitspanne, bis ein nicht abgeschlossener Auftrag beendet wird. Wenn ein Versuch die Timeout-Dauer überschreitet, wird der Versuch gestoppt und wechselt in den `FAILED` Status . Weitere Informationen finden Sie unter Timeouts bei der Job. Der Mindestwert beträgt 60 Sekunden.
8. (Optional) Aktivieren Sie die Planungspriorität . Geben Sie einen Wert für die Planungspriorität zwischen 0 und 100 ein. Höhere Werte haben eine höhere Priorität als niedrigere Werte.
9. (Optional) Erweitern Sie Tags und wählen Sie dann Tag hinzufügen aus, um der Ressource Tags hinzuzufügen.
10. Wählen Sie Nächste Seite aus.
11. Im Abschnitt EKS-podEigenschaften:

    a. Geben Sie für Servicekontoname ein Konto ein, das eine Identität für Prozesse bereitstellt, die in einem ausgeführt werdenpod.

    b. Aktivieren Sie das Host-Netzwerk, um das Kubernetes pod Netzwerkmodell zu verwenden, und öffnen Sie einen Listening-Port für eingehende Verbindungen. Deaktivieren Sie diese Einstellung nur für ausgehende Kommunikation.

c. Wählen Sie für DNS-Richtlinie eine der folgenden Optionen aus:

- Kein Wert (null) – Die pod ignoriert die DNS-Einstellungen aus der Kubernetes Umgebung.
- Standard – Der pod erbt die Konfiguration für die Namensauflösung von dem Knoten, auf dem er ausgeführt wird.

> **Note**
>
> Wenn keine DNS-Richtlinie angegeben ist, ist Standard nicht die Standard-DNS-Richtlinie. Stattdessen ClusterFirst wird verwendet.

- ClusterFirst – Jede DNS-Abfrage, die nicht mit dem konfigurierten Cluster-Domain-Suffix übereinstimmt, wird an den Upstream-Nameserver weitergeleitet, der vom Knoten geerbt wird.
- ClusterFirstWithHostNet – Wird verwendet, wenn das Host-Netzwerk aktiviert ist.

d. (Optional) Wählen Sie für Pod-Labels die Option Pod-Labels hinzufügen aus und geben Sie dann ein Name-Wert-Paar ein.

> **Important**
>
> Das Präfix für eine Pod-Bezeichnung darf nicht `kubernetes.io/`, `k8s.io/`oder enthaltenbatch.amazonaws.com/.

e. Wählen Sie Nächste Seite aus.

f. Im Abschnitt Container-Konfiguration:

i. Geben Sie unter Name einen eindeutigen Namen für den Container ein. Der Name muss mit einem Buchstaben oder einer Zahl beginnen und kann bis zu 63 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen und Bindestriche (-) enthalten.

ii. Wählen Sie für Image das Docker Image aus, das Sie für Ihren Auftrag verwenden möchten. Images in der Docker-Hub-Registry sind standardmäßig verfügbar. Sie können auch andere Repositorys mit *`repository-url/image:tag`* angeben. Er kann bis zu 255 Zeichen lang sein. Sie kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-), Unterstriche (_), Doppelstriche (:), Punkte (.), Schrägstriche (/) und Zahlenzeichen (#) enthalten. Dieser Parameter ist `Image` im Abschnitt Erstellen eines Containers der Docker Remote API und dem `IMAGE` Parameter zugeordnet docker run

> **Note**
>
> Docker Die Image-Architektur muss mit der Prozessorarchitektur der Rechenressourcen übereinstimmen, für die sie geplant sind. Beispielsweise können Armbasierte Docker Images nur auf -Armbasierten Rechenressourcen ausgeführt werden.

- Bilder in öffentlichen Amazon-ECR-Repositorys verwenden die vollständige - `registry/repository[:tag]` oder `-registry/repository[@digest]`Namenskonvention (z. B. `public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).
- Bilder in Amazon-ECR-Repositorys verwenden die vollständige `registry/repository[:tag]` Namenskonvention (z. B. *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*).
- Images in offiziellen Repositorys in Docker Hub verwenden einen einzelnen Namen (z. B. `ubuntu` oder `mongo`).
- Images in anderen Repositorys in Docker Hub sind mit einem Organisationsnamen qualifiziert (z. B `amazon/amazon-ecs-agent`).
- Image in anderen Online-Repositorys sind durch einen Domain-Namen zusätzlich qualifiziert (z. B. `quay.io/assemblyline/ubuntu`).

iii. (Optional) Wählen Sie für Image-Pull-Richtlinie aus, wann Images abgerufen werden.

iv. (Optional) Geben Sie für Befehl einen - oder -JSONBefehl ein, der an den Container übergeben werden soll. Bash

v. (Optional) Geben Sie für Argumente Argumente ein, die an den Container übergeben werden sollen. Wenn kein Argument angegeben wird, wird der Container-Image-Befehl verwendet.

g. (Optional) Sie können der Auftragsdefinition Parameter als Name-Wert-Zuweisungen hinzufügen, um die Standardwerte der Auftragsdefinition zu überschreiben. So fügen Sie einen Parameter hinzu:

- Geben Sie für Parameter ein Name-Wert-Paar ein und wählen Sie dann Parameter hinzufügen aus.

> **Important**
>
> Wenn Sie Parameter hinzufügen auswählen, müssen Sie mindestens einen Parameter konfigurieren oder Parameter entfernen auswählen

h. Im Abschnitt Umgebungskonfiguration:

   i. Geben Sie für vCPUs die Anzahl der vCPUs ein, die für den Container reserviert werden sollen. Dieser Parameter ordnet zu `CpuShares` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--cpu-shares` für die docker run zu. Jede vCPU entspricht 1.024 CPU-Anteilen. Sie müssen mindestens eine vCPU angeben.

   ii. Geben Sie für Speicher das Speicherlimit ein, das dem Container zur Verfügung steht. Wenn Ihr Container versucht, den hier angegebenen Speicher zu überschreiten, wird der Container gestoppt. Dieser Parameter ordnet zu `Memory` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--memory` für die docker run zu. Sie müssen mindestens 4 MB Arbeitsspeicher für einen Auftrag festlegen.

> **Note**
>
> Um Ihre Ressourcenauslastung zu maximieren, priorisieren Sie den Speicher für Aufträge eines bestimmten Instance-Typs. Weitere Informationen finden Sie unter DatenverarbeitungsressourceSpeicherverwaltung.

i. (Optional) Wählen Sie für Umgebungsvariablen die Option Umgebungsvariable hinzufügen aus, um Umgebungsvariablen als Name-Wert-Paare hinzuzufügen. Diese Variablen werden an den Container übergeben.

j. (Optional) Für Volume-Mount:

   i. Wählen Sie Volume-Mount hinzufügen aus.

   ii. Geben Sie einen Namen und dann einen Mount-Pfad in den Container ein, in dem das Volume gemountet ist.

   iii. Wählen Sie Nur lesen, um Schreibberechtigungen für das Volume zu entfernen.

   iv. Wählen Sie Volume-Mount hinzufügen aus.

k. (Optional) Geben Sie unter Als Benutzer ausführen eine Benutzer-ID ein, um den Container-Prozess auszuführen.

> **Note**
>
> Die Benutzer-ID muss im Image vorhanden sein, damit der Container ausgeführt werden kann.

l. (Optional) Geben Sie für Als Gruppe ausführen eine Gruppen-ID ein, um die Laufzeit des Containerprozesses auszuführen.

> **Note**
>
> Die Gruppen-ID muss im Image vorhanden sein, damit der Container ausgeführt werden kann.

m. (Optional) Um dem Container Ihres Auftrags erhöhte Berechtigungen auf der Host-Instance zu erteilen (ähnlich wie der `root` Benutzer), ziehen Sie den Schieberegler Privilegiert nach rechts. Dieser Parameter ordnet zu `Privileged` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--privileged` für die docker run zu.

n. (Optional) Aktivieren Sie Schreibgeschütztes Stammdateisystem, um den Schreibzugriff auf das Stammdateisystem zu entfernen.

o. (Optional) Aktivieren Sie Als Nicht-Root ausführen, um die Container in der pod als Nicht-Root-Benutzer auszuführen.

> **Note**
>
> Wenn Als Nicht-Root ausführen aktiviert ist, kubelet validiert die das Image zur Laufzeit, um sicherzustellen, dass das Image nicht als UID 0 ausgeführt wird.

p. Wählen Sie Nächste Seite aus.

12. Überprüfen Sie für Überprüfung der Auftragsdefinition die Konfigurationsschritte. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Auftragsdefinition erstellen aus.

Erstellen einer parallelen Auftragsdefinition mit mehreren Knoten

Bevor Sie Aufträge in AWS Batch ausführen können, müssen Sie eine Auftragsdefinition erstellen. Dieser Prozess variiert geringfügig zwischen parallelen Aufträgen mit einem Knoten und mehreren

Knoten. In diesem Thema wird insbesondere beschrieben, wie Sie eine Auftragsdefinition für einen parallelen Auftrag mit AWS Batch mehreren Knoten erstellen. Weitere Informationen finden Sie unter parallel Jobs mit mehreren Knoten.

> **Note**
>
> AWS Fargate unterstützt keine parallelen Aufträge mit mehreren Knoten.

Erstellen einer parallelen Auftragsdefinition mit mehreren Knoten auf Amazon EC2Ressourcen

Weitere Informationen zum Erstellen einer Definition für Aufträge mit einzelnen Knoten finden Sie unter Erstellen einer Auftragsdefinition mit einem Knoten .

So erstellen Sie eine parallele Auftragsdefinition mit mehreren Knoten auf Amazon Elastic Compute Cloud-Ressourcen:

1. Öffnen Sie die -AWS BatchKonsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie die zu verwendende AWS-Region in der Navigationsleiste aus.
3. Wählen Sie im Navigationsbereich Auftragsdefinitionen aus.
4. Wählen Sie Erstellen.
5. Wählen Sie für Orchestrierungstyp Amazon Elastic Compute Cloud (Amazon EC2) aus.
6. Aktivieren Sie für Parallel mit mehreren Knoten aktivieren die Option Parallel mit mehreren Knoten.
7. Geben Sie unter Name einen eindeutigen Namen für Ihre Auftragsdefinition ein. Der Name kann bis zu 128 Zeichen lang sein und Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
8. (Optional) Geben Sie unter Ausführungs-Timeout die maximale Anzahl von Sekunden an, die Auftragsversuche ausführen soll. Wenn ein Versuch die Timeout-Dauer überschreitet, wird der Versuch gestoppt und wechselt in einen -`FAILED`Status. Weitere Informationen finden Sie unter Timeouts bei der Job.
9. (Optional) Aktivieren Sie die Planungspriorität . Geben Sie einen Wert für die Planungspriorität zwischen 0 und 100 ein. Höhere Werte haben eine höhere Priorität als niedrigere Werte.
10. (Optional) Geben Sie unter Auftragsversuche ein, wie oft AWS Batch versucht, den Auftrag in den `RUNNABLE` Status zu verschieben. Geben Sie eine Zahl zwischen 1 und 10 ein.

11. (Optional) Wählen Sie für Bedingungen für Wiederholungsstrategie die Option Bewertung beim Beenden hinzufügen aus. Geben Sie mindestens einen Parameterwert ein und wählen Sie dann eine Aktion aus. Für jeden Bedingungssatz muss Aktion entweder auf Wiederholen oder Beenden festgelegt werden. Diese Aktionen bedeuten Folgendes:

    - Wiederholen – versucht AWS Batch es erneut, bis die von Ihnen angegebene Anzahl von Auftragsversuchen erreicht ist.
    - Beenden – AWS Batch stoppt den erneuten Versuch des Auftrags.

    > **Important**
    >
    > Wenn Sie Bewertung beim Beenden hinzufügen auswählen, müssen Sie mindestens einen Parameter konfigurieren und entweder eine Aktion oder beim Beenden die Option Bewertung entfernen auswählen.

12. (Optional) Erweitern Sie Tags und wählen Sie dann Tag hinzufügen aus, um der Ressource Tags hinzuzufügen. Geben Sie einen Schlüssel und einen optionalen Wert ein und wählen Sie dann Tag hinzufügen aus. Sie können auch Tags propagieren aktivieren, um Tags aus dem Auftrag und der Auftragsdefinition an die Amazon-ECS-Aufgabe zu propagieren.
13. Wählen Sie Nächste Seite aus.
14. Geben Sie unter Number of Nodes (Anzahl von Knoten) die Gesamtanzahl von Knoten ein, die für Ihren Auftrag verwendet werden sollen.
15. Unter Main node (Hauptknoten) geben Sie den für den Hauptknoten zu verwendenden Knotenindex ein. Der standardmäßige Hauptknotenindex lautet `0`.
16. Wählen Sie für Instance-Typ einen Instance-Typ aus.

    > **Note**
    >
    > Der ausgewählte Instance-Typ gilt für alle Knoten.

17. Wählen Sie für Parameter die Option Parameter hinzufügen aus, um Platzhalter für die Parameterersetzung als Schlüssel- und optionale Wertpaare hinzuzufügen.
18. Im Abschnitt Knotenbereiche:

    a. Wählen Sie Knotenbereich hinzufügen aus. Dadurch wird ein Knotenbereich erstellt.

b. Legen Sie unter Target nodes (Zielknoten) mit der Notation *range_start*:*range_end* den Bereich für Ihre Knotengruppe fest.

   Sie können bis zu fünf Knotenbereiche für die Knoten erstellen, die Sie für Ihren Auftrag angegeben haben. Knotenbereiche verwenden den Indexwert für einen Knoten und der Knotenindex beginnt bei 0. Stellen Sie sicher, dass der Wert des Bereichsendindex Ihrer endgültigen Knotengruppe um eins kleiner ist als die von Ihnen angegebene Anzahl von Knoten. Angenommen, Sie haben 10 Knoten angegeben und möchten eine einzelne Knotengruppe verwenden. Dann beträgt Ihr Endbereich 9.

c. Wählen Sie für Image das Docker Image aus, das Sie für Ihren Auftrag verwenden möchten. Images in der Docker Hub-Registrierung sind standardmäßig verfügbar. Sie können auch andere Repositorys mit *repository-url*/*image*:*tag* angeben. Der Name kann bis zu 225 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-), Unterstriche (_), Doppelpunkte (:), Schrägstriche (/) und Zahlenzeichen (#) enthalten. Dieser Parameter wird `Image` zugeordnet im Abschnitt Create a container (Erstellen eines Containers) im Docker-Remote-API und dem `IMAGE`-Parameter von docker run.

   > **Note**
   >
   > Docker Die Image-Architektur muss mit der Prozessorarchitektur der Rechenressourcen übereinstimmen, für die sie geplant sind. Beispielsweise können Armbasierte Docker Images nur auf -Armbasierten Rechenressourcen ausgeführt werden.

   - Images in öffentlichen Amazon-ECR-Repositorys verwenden die vollständige - `registry/repository[:tag]` oder `-registry/repository[@digest]`Namenskonvention (z. B. `public.ecr.aws/`*registry_alias*`/`*my-web-app*`:`*latest*).
   - Bilder in Amazon-ECR-Repositorys verwenden die vollständige `registry/repository[:tag]` Namenskonvention. Beispiel: *aws_account_id*`.dkr.ecr.`*region*`.amazonaws.com/`*my-web-app*`:`*latest*
   - Images in offiziellen Repositorys in Docker Hub verwenden einen einzelnen Namen (z. B. `ubuntu` oder `mongo`).
   - Images in anderen Repositorys in Docker Hub sind mit einem Organisationsnamen qualifiziert (z. B `amazon/amazon-ecs-agent`).

- Image in anderen Online-Repositorys sind durch einen Domain-Namen zusätzlich qualifiziert (z. B. `quay.io/assemblyline/ubuntu`).

d. Wählen Sie für Befehlssyntax Bash oder JSON aus.

e. Geben Sie unter Befehl den Befehl an, der an den Container übergeben werden soll. Bei einfachen Befehlen können Sie den Befehl wie bei einer Eingabeaufforderung auf der Registerkarte Leerzeichen als Trennzeichen eingeben. Überprüfen Sie dann, ob das JSON Ergebnis korrekt ist. Das JSON-Ergebnis wird an den übergebenDocker daemon. Zur Eingabe komplizierter Befehle (z. B. mit Sonderzeichen) wechseln Sie zur Registerkarte JSON und geben Sie das Äquivalent des Zeichenfolgen-Arrays hier ein.

   Dieser Parameter ist `Cmd` im Abschnitt Erstellen eines Containers der Docker Remote-API und dem Parameter `COMMAND` von docker run zugeordnet. Weitere Informationen zum Docker-Parameter `CMD` finden Sie unter https://docs.docker.com/engine/reference/builder/#cmd.

   > **Note**
   >
   > Sie können Standardwerte für die Parameterersetzung und Platzhalter in Ihrem Befehl verwenden. Weitere Informationen finden Sie unter Parameter.

f. Geben Sie für vCPUs die Anzahl von vCPUs ein, die für den Container reserviert werden sollen. Dieser Parameter ordnet zu `CpuShares` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--cpu-shares` für die docker run zu. Jede vCPU entspricht 1.024 CPU-Anteilen. Sie müssen mindestens eine vCPU angeben.

g. Geben Sie unter Arbeitsspeicher die Arbeitsspeichergrenze (in MiB) für die Präsentation an den Container des Auftrags an. Wenn Ihr Container versucht, den hier angegebenen Speicher zu überschreiten, wird der Container gestoppt. Dieser Parameter ordnet zu `Memory` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--memory` für die docker run zu. Sie müssen mindestens 4 MB Arbeitsspeicher für einen Auftrag festlegen.

   > **Note**
   >
   > Um Ihre Ressourcenauslastung zu maximieren, können Sie Ihren Aufträgen so viel Speicher wie möglich für einen bestimmten Instance-Typ bereitstellen. Weitere Informationen finden Sie unter DatenverarbeitungsressourceSpeicherverwaltung.

h. (Optional) Geben Sie unter Anzahl der GPUs die Anzahl der GPUs an, die Ihr Auftrag verwendet. Der Auftrag wird auf einem Container mit der angegebenen Anzahl von GPUs ausgeführt, die an diesen Container angeheftet sind.

i. (Optional) Für Auftragsrolle können Sie eine IAM-Rolle angeben, die dem Container in Ihrem Auftrag Berechtigungen zur Verwendung der AWS APIs bereitstellt. Diese Funktion verwendet Amazon-ECS-IAM-Rollen für die Aufgabenfunktionalität. Weitere Informationen, einschließlich Konfigurationsvoraussetzungen, finden Sie unter IAM-Rollen für Aufgaben im Amazon Elastic Container Service-Entwicklerhandbuch.

> **Note**
>
> Für Aufträge, die auf Fargate-Ressourcen ausgeführt werden, ist eine Auftragsrolle erforderlich.

> **Note**
>
> Hier werden nur Rollen angezeigt, die über die Vertrauensstellung der Amazon Elastic Container Service-Aufgabenrolle verfügen. Weitere Informationen zum Erstellen einer IAM-Rolle für Ihre AWS Batch Aufträge finden Sie unter Erstellen einer IAM-Rolle und -Richtlinie für Ihre Aufgaben im Amazon Elastic Container Service-Entwicklerhandbuch.

j. (Optional) Geben Sie für Ausführungsrolle eine IAM-Rolle an, die den Amazon-ECS-Container-Agenten die Berechtigung erteilt, AWS API-Aufrufe in Ihrem Namen durchzuführen. Diese Funktion verwendet Amazon-ECS-IAM-Rollen für die Aufgabenfunktionalität. Weitere Informationen finden Sie unter IAM-Rollen für die Amazon-ECS-Aufgabenausführung im Amazon Elastic Container Service-Entwicklerhandbuch.

19. (Optional) Erweitern Sie Zusätzliche Konfiguration:

a. Wählen Sie für Umgebungsvariablen die Option Umgebungsvariable hinzufügen aus, um Umgebungsvariablen als Name-Wert-Paare hinzuzufügen. Diese Variablen werden an den Container übergeben.

b. Für die Konfiguration der Auftragsrolle können Sie eine IAM-Rolle angeben, die dem Container in Ihrem Auftrag Berechtigungen zur Verwendung der AWS APIs bereitstellt. Diese Funktion verwendet Amazon-ECS-IAM-Rollen für die Aufgabenfunktionalität. Weitere

Informationen, einschließlich Konfigurationsvoraussetzungen, finden Sie unter IAM-Rollen für Aufgaben im Amazon Elastic Container Service-Entwicklerhandbuch.

> **Note**
>
> Für Aufträge, die auf Fargate-Ressourcen ausgeführt werden, ist eine Auftragsrolle erforderlich.

> **Note**
>
> Hier werden nur Rollen angezeigt, die über die Vertrauensstellung der Aufgabenrolle des Amazon Elastic Container Service verfügen. Weitere Informationen zum Erstellen einer IAM-Rolle für Ihre AWS Batch Aufträge finden Sie unter Erstellen einer IAM-Rolle und -Richtlinie für Ihre Aufgaben im Amazon Elastic Container Service-Entwicklerhandbuch.

c. Geben Sie für Ausführungsrolle eine IAM-Rolle an, die den Amazon-ECS-Container-Agenten die Berechtigung erteilt, AWS API-Aufrufe in Ihrem Namen durchzuführen. Diese Funktion verwendet Amazon-ECS-IAM-Rollen für die Aufgabenfunktionalität. Weitere Informationen finden Sie unter IAM-Rollen für die Amazon-ECS-Aufgabenausführung im Amazon Elastic Container Service-Entwicklerhandbuch.

20. Im Abschnitt Sicherheitskonfiguration:

    a. (Optional) Um dem Container Ihres Auftrags erhöhte Berechtigungen auf der Host-Instance zu erteilen (ähnlich wie der `root` Benutzer), aktivieren Sie Privilegierte . Dieser Parameter ordnet zu `Privileged` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--privileged` für die docker run zu.

    b. (Optional) Geben Sie für Benutzer den Benutzernamen ein, der im Container verwendet werden soll. Dieser Parameter ordnet zu `User` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--user` für die docker run zu.

    c. (Optional) Wählen Sie für Secrets die Option Secret hinzufügen aus, um Secrets als Name-Wert-Paare hinzuzufügen. Diese Secrets werden im Container verfügbar gemacht. Weitere Informationen finden Sie unter secretOptions in Jobdefinitionsparameter für ContainerProperties.

21. Im Abschnitt Linux-Konfiguration:

a. Aktivieren Sie Schreibgeschütztes Dateisystem aktivieren, um den Schreibzugriff auf das Volume zu entfernen.

b. (Optional) Aktivieren Sie init Prozess aktivieren, um einen -`init`Prozess innerhalb des Containers auszuführen. Dieser Prozess leitet Signale weiter und nimmt Prozesse auf.

c. Geben Sie für Größe des gemeinsam genutzten Speichers die Größe (in MiB) des `/dev/shm` Volumes ein.

d. Geben Sie für Maximale Auslagerungsgröße die Gesamtmenge des Auslagerungsspeichers (in MiB) ein, die der Container verwenden kann.

e. Geben Sie für Auslagerung einen Wert zwischen 0 und 100 ein, um das Auslagerungsverhalten des Containers anzugeben. Wenn Sie keinen Wert angeben und Auslagern aktiviert ist, ist der Wert standardmäßig 60. Weitere Informationen finden Sie unter Auslagerung in Jobdefinitionsparameter für ContainerProperties.

f. (Optional) Wählen Sie unter Geräte die Option Gerät hinzufügen aus, um ein Gerät hinzuzufügen:

   i. Geben Sie für Container path (Container-Pfad) den Pfad in der Container-Instance an, um das der Host-Instance zugeordnete Gerät zugänglich zu machen. Wenn Sie dieses Feld leer lassen, wird der Host-Pfad im Container verwendet.

   ii. Geben Sie für Host path (Host-Pfad) den Pfad eines Geräts in der Host-Instance an.

   iii. Wählen Sie für Berechtigungen eine oder mehrere Berechtigungen aus, die auf das Gerät angewendet werden sollen. Die verfügbaren Berechtigungen sind READ , WRITE und MVNOD .

22. (Optional) Wählen Sie unter Mountingpunkte die Option Konfiguration für Mountingpunkte hinzufügen aus, um Mountingpunkte für Daten-Volumes hinzuzufügen. Sie müssen das Quell-Volume und den Container-Pfad angeben. Diese Mountingpunkte werden an den Docker Daemon auf einer Container-Instance übergeben. Sie können das Volume auch schreibgeschützt machen.

23. (Optional) Wählen Sie für Ulimits-Konfiguration die Option Ulimit hinzufügen aus, um einen `ulimits` Wert für den Container hinzuzufügen. Geben Sie den Namen , das weiche Limit und die Hard-Limit-Werte ein und wählen Sie dann Ulimit hinzufügen aus.

24. (Optional) Wählen Sie für Volumes-Konfiguration die Option Volume hinzufügen aus, um eine Liste von Volumes zu erstellen, die an den Container übergeben werden sollen. Geben Sie Name und Quellpfad für das Volume ein und wählen Sie dann Volume hinzufügen aus. Sie können auch EFS aktivieren aktivieren.

25. (Optional) Wählen Sie für Tmpfs die Option tmpfs hinzufügen aus, um ein `tmpfs` Mount hinzuzufügen.
26. (Optional) Im Abschnitt Protokollierungskonfiguration:
    a. Wählen Sie für Protokolltreiber den zu verwendenden Protokolltreiber aus. Weitere Informationen zu den verfügbaren Protokolltreibern finden Sie unter logDriver in Jobdefinitionsparameter für ContainerProperties.

    > **Note**
    >
    > Standardmäßig wird der `-awslogs`Protokolltreiber verwendet.

    b. Wählen Sie für Optionen die Option Option Hinzufügen aus, um eine Option hinzuzufügen. Geben Sie ein Name-Wert-Paar ein und wählen Sie dann Option hinzufügen aus.
    c. Wählen Sie für Secrets die Option Secret hinzufügen aus. Geben Sie ein Name-Wert-Paar ein und wählen Sie dann Secret hinzufügen, um ein Secret hinzuzufügen.

    > **Tip**
    >
    > Weitere Informationen finden Sie unter secretOptions in Jobdefinitionsparameter für ContainerProperties.

27. Wählen Sie Nächste Seite aus.
28. Überprüfen Sie für die Überprüfung der Auftragsdefinition die Konfigurationsschritte. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Auftragsdefinition erstellen aus.

Erstellen von Auftragsdefinitionen mit ContainerProperties

Im Folgenden finden Sie eine leere Auftragsdefinitionsvorlage, die einen einzelnen Container enthält. Sie können diese Vorlage verwenden, um Ihre Auftragsdefinition zu erstellen, die dann in einer Datei gespeichert und mit der AWS CLI `--cli-input-json` Option verwendet werden kann. Weitere Informationen zu diesen Parametern finden Sie unter Jobdefinitionsparameter für ContainerProperties.

```
{
    "jobDefinitionName": "",
```

```
    "type": "container",
    "parameters": {
        "KeyName": ""
    },
    "schedulingPriority": 0,
    "containerProperties": {
        "image": "",
        "vcpus": 0,
        "memory": 0,
        "command": [
            ""
        ],
        "jobRoleArn": "",
        "executionRoleArn": "",
        "volumes": [
            {
                "host": {
                    "sourcePath": ""
                },
                "name": "",
                "efsVolumeConfiguration": {
                    "fileSystemId": "",
                    "rootDirectory": "",
                    "transitEncryption": "ENABLED",
                    "transitEncryptionPort": 0,
                    "authorizationConfig": {
                        "accessPointId": "",
                        "iam": "DISABLED"
                    }
                }
            }
        ],
        "environment": [
            {
                "name": "",
                "value": ""
            }
        ],
        "mountPoints": [
            {
                "containerPath": "",
                "readOnly": true,
                "sourceVolume": ""
            }
```

```
        ],
        "readonlyRootFilesystem": true,
        "privileged": true,
        "ulimits": [
            {
                "hardLimit": 0,
                "name": "",
                "softLimit": 0
            }
        ],
        "user": "",
        "instanceType": "",
        "resourceRequirements": [
            {
                "value": "",
                "type": "MEMORY"
            }
        ],
        "linuxParameters": {
            "devices": [
                {
                    "hostPath": "",
                    "containerPath": "",
                    "permissions": [
                        "WRITE"
                    ]
                }
            ],
            "initProcessEnabled": true,
            "sharedMemorySize": 0,
            "tmpfs": [
                {
                    "containerPath": "",
                    "size": 0,
                    "mountOptions": [
                        ""
                    ]
                }
            ],
            "maxSwap": 0,
            "swappiness": 0
        },
        "logConfiguration": {
            "logDriver": "syslog",
```

```
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ]
        },
        "secrets": [
            {
                "name": "",
                "valueFrom": ""
            }
        ],
        "networkConfiguration": {
            "assignPublicIp": "DISABLED"
        },
        "fargatePlatformConfiguration": {
            "platformVersion": ""
        }
    },
    "nodeProperties": {
        "numNodes": 0,
        "mainNode": 0,
        "nodeRangeProperties": [
            {
                "targetNodes": "",
                "container": {
                    "image": "",
                    "vcpus": 0,
                    "memory": 0,
                    "command": [
                        ""
                    ],
                    "jobRoleArn": "",
                    "executionRoleArn": "",
                    "volumes": [
                        {
                            "host": {
                                "sourcePath": ""
                            },
                            "name": "",
```

```
                "efsVolumeConfiguration": {
                    "fileSystemId": "",
                    "rootDirectory": "",
                    "transitEncryption": "DISABLED",
                    "transitEncryptionPort": 0,
                    "authorizationConfig": {
                        "accessPointId": "",
                        "iam": "ENABLED"
                    }
                }
            }
        ],
        "environment": [
            {
                "name": "",
                "value": ""
            }
        ],
        "mountPoints": [
            {
                "containerPath": "",
                "readOnly": true,
                "sourceVolume": ""
            }
        ],
        "readonlyRootFilesystem": true,
        "privileged": true,
        "ulimits": [
            {
                "hardLimit": 0,
                "name": "",
                "softLimit": 0
            }
        ],
        "user": "",
        "instanceType": "",
        "resourceRequirements": [
            {
                "value": "",
                "type": "MEMORY"
            }
        ],
        "linuxParameters": {
            "devices": [
```

```
                    {
                        "hostPath": "",
                        "containerPath": "",
                        "permissions": [
                            "WRITE"
                        ]
                    }
                ],
                "initProcessEnabled": true,
                "sharedMemorySize": 0,
                "tmpfs": [
                    {
                        "containerPath": "",
                        "size": 0,
                        "mountOptions": [
                            ""
                        ]
                    }
                ],
                "maxSwap": 0,
                "swappiness": 0
            },
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "KeyName": ""
                },
                "secretOptions": [
                    {
                        "name": "",
                        "valueFrom": ""
                    }
                ]
            },
            "secrets": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ],
            "networkConfiguration": {
                "assignPublicIp": "DISABLED"
            },
            "fargatePlatformConfiguration": {
```

```
                    "platformVersion": ""
                }
            }
        }
    ]
},
"retryStrategy": {
    "attempts": 0,
    "evaluateOnExit": [
        {
            "onStatusReason": "",
            "onReason": "",
            "onExitCode": "",
            "action": "RETRY"
        }
    ]
},
"propagateTags": true,
"timeout": {
    "attemptDurationSeconds": 0
},
"tags": {
    "KeyName": ""
},
"platformCapabilities": [
    "EC2"
],
"eksProperties": {
    "podProperties": {
        "serviceAccountName": "",
        "hostNetwork": true,
        "dnsPolicy": "",
        "containers": [
            {
                "name": "",
                "image": "",
                "imagePullPolicy": "",
                "command": [
                    ""
                ],
                "args": [
                    ""
                ],
                "env": [
```

```
                    {
                        "name": "",
                        "value": ""
                    }
                ],
                "resources": {
                    "limits": {
                        "KeyName": ""
                    },
                    "requests": {
                        "KeyName": ""
                    }
                },
                "volumeMounts": [
                    {
                        "name": "",
                        "mountPath": "",
                        "readOnly": true
                    }
                ],
                "securityContext": {
                    "runAsUser": 0,
                    "runAsGroup": 0,
                    "privileged": true,
                    "readOnlyRootFilesystem": true,
                    "runAsNonRoot": true
                }
            }
        ],
        "volumes": [
            {
                "name": "",
                "hostPath": {
                    "path": ""
                },
                "emptyDir": {
                    "medium": "",
                    "sizeLimit": ""
                },
                "secret": {
                    "secretName": "",
                    "optional": true
                }
            }
```

```
            ]
        }
    }
}
```

> **Note**
>
> Sie können eine Auftragsdefinitionsvorlage für den Singtle-Container mit dem folgenden AWS CLI Befehl generieren:
>
> ```
> $ aws batch register-job-definition --generate-cli-skeleton
> ```

Jobdefinitionsparameter für ContainerProperties

Jobdefinitionen, die ich verwende, `ContainerProperties`sind in mehrere Teile aufgeteilt:

- der Name der Jobdefinition
- der Typ der Jobdefinition
- Die Standardwerte für den Platzhalter für die Parameterersetzung
- die Container-Eigenschaften für den Job
- die Amazon EKS-Eigenschaften für die Auftragsdefinition, die für Jobs erforderlich sind, die auf Amazon EKS-Ressourcen ausgeführt werden
- die Knoteneigenschaften, die für einen Paralleljob mit mehreren Knoten erforderlich sind
- die Plattformfunktionen, die für Jobs erforderlich sind, die auf Fargate-Ressourcen ausgeführt werden
- die standardmäßigen Details zur Tag-Propagierung der Auftragsdefinition
- die standardmäßige Wiederholungsstrategie für die Auftragsdefinition
- die standardmäßige Planungspriorität für die Auftragsdefinition
- die Standard-Tags für die Jobdefinition
- das Standard-Timeout für die Jobdefinition

Inhalt

- Typ
- Parameter
- Eigenschaften des Containers
- Amazon EKS-Eigenschaften
- Funktionen der Plattform
- Tags weitergeben
- Eigenschaften des Knotens
- Strategie erneut versuchen
- Priorität bei der Planung
- Tags
- Timeout

Name der Jobdefinition

`jobDefinitionName`

Wenn Sie eine Auftragsdefinition registrieren, geben Sie einen Namen ein. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten. Die erste Jobdefinition, die mit diesem Namen registriert ist, erhält die Revision 1. Alle nachfolgenden Auftragsdefinitionen, die mit diesem Namen registriert werden, erhalten eine inkrementelle Versionsnummer.

Typ: Zeichenfolge

Erforderlich: Ja

Typ

`type`

Wenn Sie eine Auftragsdefinition registrieren, geben Sie den Auftragstyp an. Wenn der Job auf Fargate-Ressourcen ausgeführt wird, wird er `multinode` nicht unterstützt. Weitere Informationen zu parallelen Aufträgen mit mehreren Knoten finden Sie unter the section called “Erstellen einer parallelen Auftragsdefinition mit mehreren Knoten”.

Typ: Zeichenfolge

Zulässige Werte: `container` | `multinode`

Erforderlich: Ja

Parameter

`parameters`

Wenn Sie einen Job einreichen, können Sie Parameter angeben, die die Platzhalter ersetzen oder die Standardparameter der Jobdefinition überschreiben. Die Parameter in Anforderungen zum Senden von Aufträgen haben Vorrang vor den Standardeinstellungen in einer Auftragsdefinition. Das bedeutet, dass Sie dieselbe Jobdefinition für mehrere Jobs verwenden können, die dasselbe Format verwenden. Sie können Werte im Befehl auch während der Übermittlung programmgesteuert ändern.

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Erforderlich: Nein

Wenn Sie eine Auftragsdefinition registrieren, können Sie Parameterersetzungsplatzhalter in das Feld `command` der Container-Eigenschaften eines Auftrags verwenden. Die Syntax ist wie folgt.

```
"command": [
    "ffmpeg",
    "-i",
    "Ref::inputfile",
    "-c",
    "Ref::codec",
    "-o",
    "Ref::outputfile"
]
```

Im obigen Beispiel befinden sich die Parameterersetzungsplatzhalter *`Ref::inputfile`*, *`Ref::codec`* und *`Ref::outputfile`* im Befehl. Sie können das `parameters` Objekt in der Jobdefinition verwenden, um Standardwerte für diese Platzhalter festzulegen. Um beispielsweise einen Standard für die *`Ref::codec`*-Platzhalter einzustellen, legen Sie Folgendes in der Auftragsdefinition fest:

```
"parameters" : {"codec" : "mp4"}
```

Wenn diese Auftragsdefinition zur Ausführung übergeben wird, wird das *Ref::codec* Argument im Befehl für den Container durch den Standardwert ersetzt. mp4

Eigenschaften des Containers

Wenn Sie eine Jobdefinition registrieren, geben Sie eine Liste von Container-Eigenschaften an, die bei der Auftragserteilung an den Docker-Daemon auf einer Container-Instance übergeben werden. Die folgenden Container-Eigenschaften sind in einer Auftragsdefinition zulässig. Für Aufträge mit einzelnen Knoten werden diese Container-Eigenschaften auf Auftragsdefinitionsebene festgelegt. Für parallele Aufträge mit mehreren Knoten werden die Container-Eigenschaften für jede Knotengruppe auf der Ebene der Eigenschaften des Knotens festgelegt.

`command`

Der Befehl, der an den Container übergeben wird. Dieser Parameter ist `Cmd` im Abschnitt Erstellen eines Containers der Docker Remote-API und dem Parameter `COMMAND` von docker run zugeordnet. Weitere Informationen zum Docker-Parameter `CMD` finden Sie unter https://docs.docker.com/engine/reference/builder/#cmd.

```
"command": ["string", ...]
```

Typ: Zeichenfolgen-Array

Erforderlich: Nein

`environment`

Die Umgebungsvariablen, die an einen Container übergeben werden. Dieser Parameter ordnet zu `Env` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--env` für die docker run zu.

> **Important**
>
> Wir raten davon ab, Umgebungsvariablen im Klartext für sensible Informationen wie Anmeldeinformationen zu verwenden.

> **Note**
>
> Umgebungsvariablen dürfen nicht mit beginnen. `AWS_BATCH` Diese Namenskonvention ist Variablen vorbehalten, die vom AWS Batch Dienst festgelegt werden.

Typ: Array von Schlüssel-Wert-Paaren

Erforderlich: Nein

`name`

Der Name der Umgebungsvariable.

Typ: Zeichenfolge

Erforderlich: Ja, wenn `environment` verwendet wird.

`value`

Der Wert der Umgebungsvariable.

Typ: Zeichenfolge

Erforderlich: Ja, wenn `environment` verwendet wird.

```
"environment" : [
    { "name" : "envName1", "value" : "envValue1" },
    { "name" : "envName2", "value" : "envValue2" }
]
```

`executionRoleArn`

Wenn Sie eine Jobdefinition registrieren, können Sie eine IAM-Rolle angeben. Die Rolle gewährt dem Amazon ECS-Container-Agenten die Erlaubnis, die API-Aktionen, die in den zugehörigen Richtlinien angegeben sind, in Ihrem Namen aufzurufen. Jobs, die auf Fargate-Ressourcen ausgeführt werden, müssen eine Ausführungsrolle bereitstellen. Weitere Informationen finden Sie unter AWS Batch Ausführung, IAM-Rolle.

Typ: Zeichenfolge

Erforderlich: Nein

`fargatePlatformConfiguration`

Die Plattformkonfiguration für Jobs, die auf Fargate-Ressourcen ausgeführt werden. Aufgaben, die auf EC2-Ressourcen laufen, dürfen diesen Parameter nicht angeben.

Typ: FargatePlatformKonfigurationsobjekt

Erforderlich: Nein

`platformVersion`

Die AWS Fargate-Plattformversion, die für die Jobs verwendet wird, oder `LATEST` um eine aktuelle, zugelassene Version der AWS Fargate-Plattform zu verwenden.

Typ: Zeichenfolge

Standard: `LATEST`

Erforderlich: Nein

`image`

Das Bild, mit dem ein Job gestartet wurde. Diese Zeichenfolge wird direkt an den Docker-Daemon übergeben. Images in der Docker Hub-Registrierung sind standardmäßig verfügbar. Sie können auch andere Repositorys mit *`repository-url`*`/`*`image`*`:`*`tag`* angeben. Bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Ziffern, Bindestriche, Unterstriche, Doppelpunkte, Punkte und Schrägstriche sind zulässig. Dieser Parameter wird `Image` zugeordnet im Abschnitt Create a container (Erstellen eines Containers) im Docker-Remote-API und dem `IMAGE`-Parameter von docker run.

> **Note**
>
> DockerDie Image-Architektur muss mit der Prozessorarchitektur der Rechenressourcen übereinstimmen, für die sie geplant sind. Beispielsweise können Arm basierte Docker Images nur auf Arm basierten Rechenressourcen ausgeführt werden.

- Bilder in öffentlichen Amazon ECR-Repositorys verwenden die vollständigen `registry/repository[:tag]` oder die `registry/repository[@digest]` Namenskonventionen (z. B.`public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).

- Bilder in Amazon ECR-Repositorys verwenden die vollständige `registry/repository:[tag]` Namenskonvention. z. B. *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*.
- Images in offiziellen Repositorys in Docker Hub verwenden einen einzelnen Namen (z. B. `ubuntu` oder `mongo`).
- Images in anderen Repositorys in Docker Hub sind mit einem Organisationsnamen qualifiziert (z. B, `amazon/amazon-ecs-agent`).
- Image in anderen Online-Repositorys sind durch einen Domain-Namen zusätzlich qualifiziert (z. B. `quay.io/assemblyline/ubuntu`).

Typ: Zeichenfolge

Erforderlich: Ja

`instanceType`

Der Instance-Typ zur Verwendung für einen parallelen Auftrag mit mehreren Knoten. Alle Knotengruppen müssen in einem parallelen Auftrag mit mehreren Knoten den gleichen Instance-Typ verwenden. Dieser Parameter ist nicht gültig für Container-Jobs mit einem Knoten oder für Jobs, die auf Fargate-Ressourcen ausgeführt werden.

Typ: Zeichenfolge

Erforderlich: Nein

`jobRoleArn`

Wenn Sie eine Jobdefinition registrieren, können Sie eine IAM-Rolle angeben. Die Rolle stellt dem Auftrags-Container Berechtigungen zum Aufrufen der API-Aktionen zur Verfügung, die in den zugehörigen Richtlinien in Ihrem Namen festgelegt sind. Weitere Informationen finden Sie unter IAM-Rollen für Aufgaben im Entwicklerhandbuch zum Amazon Elastic Container Service.

Typ: Zeichenfolge

Erforderlich: Nein

`linuxParameters`

Linux-spezifische Änderungen, die auf den Container angewendet werden, z. B. Details für Gerätezuordnungen.

```
"linuxParameters": {
```

```
    "devices": [
        {
            "hostPath": "string",
            "containerPath": "string",
            "permissions": [
                "READ", "WRITE", "MKNOD"
            ]
        }
    ],
    "initProcessEnabled": true|false,
    "sharedMemorySize": 0,
    "tmpfs": [
        {
            "containerPath": "string",
            "size": integer,
            "mountOptions": [
                "string"
            ]
        }
    ],
    "maxSwap": integer,
    "swappiness": integer
}
```

Typ: LinuxParameters Objekt

Erforderlich: Nein

`devices`

Liste der im Container zugeordneten Geräte. Dieser Parameter ist `Devices` im Abschnitt Einen Container erstellen der Docker-Remote-API und der Option `--device` für die Docker-Ausführung zugeordnet.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Typ: Array von Device-Objekten

Erforderlich: Nein

`hostPath`

Pfad, in dem sich das in der Host-Container-Instance verfügbare Gerät befindet.

Typ: Zeichenfolge

Erforderlich: Ja

`containerPath`

Pfad, in dem das Gerät im Container verfügbar gemacht wird. Wenn dies nicht angegeben ist, wird das Gerät unter demselben Pfad wie der Hostpfad verfügbar gemacht.

Typ: Zeichenfolge

Erforderlich: Nein

`permissions`

Berechtigungen für das Gerät im Container. Wenn dies nicht angegeben ist, sind die Berechtigungen auf `READWRITE`, und gesetzt`MKNOD`.

Typ: Zeichenfolgen-Array

Erforderlich: Nein

Zulässige Werte: `READ` | `WRITE` | `MKNOD`

`initProcessEnabled`

Wenn der Wert "true" lautet, führen Sie einen `init`-Prozess innerhalb des Containers aus, der Signalen weiterleitet und Prozesse aufnimmt. Dieser Parameter wird der Option `--init` für die Docker-Ausführung zugeordnet. Für diesen Parameter muss Ihre Docker Remote API Version 1.25 oder höher in Ihrer Container-Instance verwenden. Um die Docker Remote API-Version auf Ihrer Container-Instance zu überprüfen, melden Sie sich bei Ihrer Container-Instance an und führen Sie den folgenden Befehl aus: `sudo docker version | grep "Server API version"`

Typ: Boolesch

Erforderlich: Nein

`maxSwap`

Die Gesamtmenge an Swap-Speicher (in MiB), die ein Job verwenden kann. Dieser Parameter ist in die Option `--memory-swap` zur Docker-Ausführung übersetzt, wobei der Wert die

Summe aus dem Container-Speicher und dem Wert `maxSwap` ist. Weitere Informationen finden Sie unter `--memory-swap`-Details in der Docker-Dokumentation.

Wenn als `maxSwap`-Wert `0` angegeben wird, verwendet der Container keine Auslagerung. Zulässige Werte sind `0` oder eine beliebige positive Ganzzahl. Wenn der `maxSwap` Parameter weggelassen wird, verwendet der Container die Swap-Konfiguration für die Container-Instance, auf der er ausgeführt wird. Es muss ein Wert für `maxSwap` festgelegt werden, damit der Parameter `swappiness` verwendet werden kann.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Typ: Ganzzahl

Erforderlich: Nein

`sharedMemorySize`

Der Wert für die Größe des `/dev/shm`-Volumes (in MiB). Dieser Parameter wird der Option `--shm-size` für die Docker-Ausführung zugeordnet.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Typ: Ganzzahl

Erforderlich: Nein

`swappiness`

Auf diese Weise können Sie das Speicherauslagerungsverhalten eines Containers optimieren. Ein `swappiness` Wert von `0` bewirkt, dass ein Austausch nicht stattfindet, es sei denn, dies ist unbedingt erforderlich. Ein `swappiness`-Wert von `100` führt dazu, dass Seiten aggressiv ausgelagert werden. Akzeptierte Werte sind Ganzzahlen zwischen `0` und `100`. Wenn der Parameter `swappiness` nicht angegeben wird, wird der Standardwert `60` verwendet. Wenn

kein Wert für `maxSwap` angegeben ist, wird dieser Parameter ignoriert. Wenn `maxSwap` auf 0 gesetzt ist, verwendet der Container kein Auslagern. Dieser Parameter wird der Option `--memory-swappiness` für die Docker-Ausführung zugeordnet.

Beachten Sie Folgendes, wenn Sie eine Auslagerungskonfiguration pro Container verwenden.

- Der Auslagerungsbereich muss auf der Container-Instance aktiviert und zugewiesen werden, damit die Container verwenden können.

  > **Note**
  >
  > Für die von Amazon ECS optimierten AMIs ist Auslagern nicht standardmäßig aktiviert. Sie müssen die Auslagerung auf der Instance aktivieren, um dieses Feature verwenden zu können. Weitere Informationen finden Sie unter Instance Store Swap Volumes im Amazon EC2 EC2-Benutzerhandbuch oder Wie weise ich mithilfe einer Swap-Datei Speicher zu, der als Auslagerungsspeicher in einer Amazon EC2 EC2-Instance verwendet werden soll? .

- Die Parameter des Auslagerungs werden nur für Aufgabendefinitionen unterstützt, die EC2-Ressourcen verwenden.
- Wenn die Container-Definitionsparameter `maxSwap` und `swappiness` in einer Aufgabendefinition weggelassen werden, hat jeder Container den `swappiness`-Standardwert 60. Die gesamte Swap-Nutzung ist auf das Zweifache der Speicherreservierung des Containers begrenzt.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Typ: Ganzzahl

Erforderlich: Nein

`tmpfs`

Der Container-Pfad, Mount-Optionen und Größe des tmpfs-Mounts.

Typ: Array von Tmpfs-Objekten

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Erforderlich: Nein

`containerPath`

Der absolute Dateipfad in dem Container, in dem das tmpfs-Volume gemountet ist.

Typ: Zeichenfolge

Erforderlich: Ja

`mountOptions`

Die Liste der Mount-Optionen für das tmpfs-Volume.

Gültige Werte: `defaults` "`ro`" | `rw` "|`suid`" | "|`nosuid`" | `dev` "|`nodev`" | `exec` "|`noexec`" | `sync` "|`async`" | "|`dirsync`" | `remount` "|`mand`" | `nomand` "|`atime`" | `noatime` "|`diratime`" | "|`nodiratime`" | `bind` "|`rbind`" | `unbindable` "|`runbindable`" | `private` "|" | `rprivate` "|`shared`" | `rshared` "|`slave`" | `rslave` "|`relatime`" | `norelatime` "|" `strictatime` "|`nostrictatime`" |“ `mode`"|" `uid` "|" `gid` "|" `nr_inodes` "|" | `nr_blocks` "|"`mpol`“

Typ: Zeichenfolgen-Array

Erforderlich: Nein

`size`

Die Größe des tmpfs-Volumes (in MB).

Typ: Ganzzahl

Erforderlich: Ja

`logConfiguration`

Die Protokollkonfigurationsspezifikation für den Job.

Dieser Parameter ist `LogConfig` im Abschnitt Einen Container erstellen der Docker-Remote-API und der Option `--log-driver` für die Docker-Ausführung zugeordnet. Standardmäßig verwenden Container den gleichen Protokolltreiber wie der Docker-Daemon. Der Container kann jedoch einen anderen Logging-Treiber als der Docker-Daemon verwenden, indem ein Protokolltreiber mit diesem Parameter in der Container-Definition angegeben wird. Um einen anderen Logging-Treiber für einen Container zu verwenden, muss das Protokollsystem entweder auf der Container-Instance oder auf einem anderen Log-Server konfiguriert werden, um Remote-Logging-Optionen bereitzustellen. Weitere Informationen zu den Optionen für die verschiedenen unterstützten Protokolltreiber finden Sie unter Configure logging drivers (Protokolltreiber konfigurieren) in der Docker-Dokumentation.

> **Note**
>
> AWS Batch unterstützt derzeit eine Teilmenge der Protokollierungstreiber, die für den Docker-Daemon verfügbar sind (siehe LogConfigurationDatentyp).

Für diesen Parameter muss Ihre Docker Remote API Version 1.18 oder höher in Ihrer Container-Instance verwenden. Um die Docker Remote API-Version auf Ihrer Container-Instance zu überprüfen, melden Sie sich bei Ihrer Container-Instance an und führen Sie den folgenden Befehl aus: `sudo docker version | grep "Server API version"`

```
"logConfiguration": {
    "devices": [
        {
            "logDriver": "string",
            "options": {
                "optionName1" : "optionValue1",
                "optionName2" : "optionValue2"
            }
            "secretOptions": [
              {
                  "name" : "secretOptionName1",
                  "valueFrom" : "secretOptionArn1"
              },
              {
                  "name" : "secretOptionName2",
                  "valueFrom" : "secretOptionArn2"
              }
            ]
```

```
        }
    ]
}
```

Typ: LogConfiguration Objekt

Erforderlich: Nein

`logDriver`

Der Protokolltreiber, der für den Job verwendet werden soll. AWS Batch Aktiviert standardmäßig den `awslogs` Protokolltreiber. Die für diesen Parameter aufgelisteten gültigen Werte sind Protokolltreiber, mit denen der Amazon-ECS-Container-Agent standardmäßig kommunizieren kann.

Dieser Parameter ist `LogConfig` im Abschnitt Einen Container erstellen der Docker-Remote-API und der Option `--log-driver` für die Docker-Ausführung zugeordnet. Standardmäßig verwenden Jobs denselben Protokollierungstreiber, den der Docker-Daemon verwendet. Der Job kann jedoch einen anderen Logging-Treiber als der Docker-Daemon verwenden, indem ein Protokolltreiber mit diesem Parameter in der Jobdefinition angegeben wird. Wenn Sie einen anderen Protokollierungstreiber für einen Job angeben möchten, muss das Protokollsystem auf der Container-Instance in der Rechenumgebung konfiguriert werden. Oder konfigurieren Sie es alternativ auf einem anderen Protokollserver, um Optionen für die Remote-Protokollierung bereitzustellen. Weitere Informationen zu den Optionen für die verschiedenen unterstützten Protokolltreiber finden Sie unter Configure logging drivers (Protokolltreiber konfigurieren) in der Docker-Dokumentation.

> **Note**
>
> AWS Batch unterstützt derzeit eine Teilmenge der Protokollierungstreiber, die für den Docker-Daemon verfügbar sind. Es ist möglich, dass in zukünftigen Releases des Amazon-ECS-Container-Agenten weitere Protokolltreiber zur Verfügung stehen.

Die unterstützten Protokolltreiber sind `awslogs`, `fluentd`, `gelf`, `json-file`, `journald`, `logentries`, `syslog` und `splunk`.

> **Note**
>
> Jobs, die auf Fargate-Ressourcen ausgeführt werden, sind auf die Treiber `awslogs` und die `splunk` Protokolltreiber beschränkt.

Für diesen Parameter muss Ihre Docker Remote API Version 1.18 oder höher in Ihrer Container-Instance verwenden. Um die Docker Remote API-Version auf Ihrer Container-Instance zu überprüfen, melden Sie sich bei Ihrer Container-Instance an und führen Sie den folgenden Befehl aus: `sudo docker version | grep "Server API version"`

> **Note**
>
> Der Amazon ECS-Container-Agent, der auf einer Container-Instance ausgeführt wird, muss die auf dieser Instance verfügbaren Protokollierungstreiber mit der `ECS_AVAILABLE_LOGGING_DRIVERS` Umgebungsvariablen registrieren. Andernfalls können die auf dieser Instance platzierten Container diese Protokollkonfigurationsoptionen nicht verwenden. Weitere Informationen finden Sie unter Amazon ECS Container Agent Configuration im Entwicklerhandbuch zum Amazon Elastic Container Service.

`awslogs`

Gibt den Amazon CloudWatch Logs-Protokollierungstreiber an. Weitere Informationen finden Sie unter Verwenden des awslogs-Protokolltreibers und Amazon CloudWatch Logs-Protokollierungstreiber in der Docker-Dokumentation.

`fluentd`

Gibt den Fluentd-Protokolltreiber an. Weitere Informationen, einschließlich Verwendung und Optionen, finden Sie unter Fluentd-Logging-Treiber in der Docker-Dokumentation.

`gelf`

Gibt den Graylog Extended Format (GELF)-Protokolltreiber an. Weitere Informationen, einschließlich Verwendung und Optionen, finden Sie unter Graylog Extended Format-Logging-Treiber in der Docker-Dokumentation.

`journald`

Gibt den Journald-Protokolltreiber an. Weitere Informationen, einschließlich Verwendung und Optionen, finden Sie unter Journald Logging-Treiber in der Docker-Dokumentation.

`json-file`

Gibt den Treiber für die JSON-Datei an. Weitere Informationen, einschließlich Verwendung und Optionen, finden Sie unter Treiber für die JSON-Dateiprotokollierung in der Docker-Dokumentation.

`splunk`

Gibt den Splunk-Protokolltreiber an. Weitere Informationen, einschließlich Verwendung und Optionen, finden Sie unter Splunk-Logging-Treiber in der Docker-Dokumentation.

`syslog`

Gibt den Syslog-Protokolltreiber an. Weitere Informationen, einschließlich Verwendung und Optionen, finden Sie unter Syslog-Logging-Treiber in der Docker-Dokumentation.

Typ: Zeichenfolge

Erforderlich: Ja

Zulässige Werte: `awslogs` | `fluentd` | `gelf` | `journald` | `json-file` | `splunk` | `syslog`

> **Note**
>
> Wenn Sie einen benutzerdefinierten Treiber haben, der nicht zuvor aufgeführt ist und den Sie mit dem Amazon ECS-Container-Agenten verwenden möchten, können Sie das Amazon ECS-Container-Agent-Projekt, auf dem verfügbar ist, forken GitHub und es so anpassen, dass es mit diesem Treiber funktioniert. Wir möchten Sie bitten, uns eventuelle Änderungswünsche mitzuteilen. Amazon Web Services unterstützt derzeit jedoch keine Anfragen, die modifizierte Kopien dieser Software ausführen.

`options`

Protokollkonfigurationsoptionen, die für den Job an einen Protokolltreiber gesendet werden sollen.

Für diesen Parameter muss Ihre Docker Remote API Version 1.19 oder höher in Ihrer Container-Instance verwenden.

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Erforderlich: Nein

`secretOptions`

Ein Objekt, welches das Secret darstellt, der an die Protokollkonfiguration übergeben werden soll. Weitere Informationen finden Sie unter Angabe sensibler Daten.

Typ: Objekt-Array

Erforderlich: Nein

`name`

Der Name der Protokolltreiberoption, die im Job festgelegt werden soll.

Typ: Zeichenfolge

Erforderlich: Ja

`valueFrom`

Der Amazon-Ressourcenname (ARN) des Geheimnisses, das in der Protokollkonfiguration des Containers offengelegt werden soll. Die unterstützten Werte sind entweder der vollständige ARN des Secrets Manager Manager-Geheimnisses oder der vollständige ARN des Parameters im SSM-Parameterspeicher.

> **Note**
>
> Wenn der Parameter SSM Parameter Store in derselben Datei AWS-Region wie die Aufgabe, die Sie starten, vorhanden ist, können Sie entweder den vollständigen ARN oder den Namen des Parameters verwenden. Wenn der Parameter in einer anderen Region existiert, muss der volle ARN angegeben werden.

Typ: Zeichenfolge

Erforderlich: Ja

`memory`

Dieser Parameter ist veraltet. Verwenden Sie ihn stattdessen. *`resourceRequirements`*

Die Anzahl der MiB an Speicher, die für den Job reserviert sind.

Als Beispiel für die Verwendung`resourceRequirements`, wenn Ihre Jobdefinition eine Syntax enthält, die der folgenden ähnelt.

```
"containerProperties": {
  "memory": 512
}
```

Die entsprechende Syntax, die Sie verwenden, `resourceRequirements` lautet wie folgt.

```
"containerProperties": {
  "resourceRequirements": [
    {
      "type": "MEMORY",
      "value": "512"
    }
  ]
}
```

Typ: Ganzzahl

Erforderlich: Ja

`mountPoints`

Die Mountingpunkte für Daten-Volumes in Ihrem Container. Dieser Parameter ordnet zu `Volumes` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--volume` für die docker run zu.

```
"mountPoints": [
                {
                  "sourceVolume": "string",
                  "containerPath": "string",
                  "readOnly": true|false
                }
              ]
```

Typ: Objekt-Array

Erforderlich: Nein

`sourceVolume`

Der Name des zu mountenden Volumes.

Typ: Zeichenfolge

Erforderlich: Ja, wenn `mountPoints` verwendet wird.

`containerPath`

Der Pfad auf dem Container, in den das Host-Volume gemountet werden soll.

Typ: Zeichenfolge

Erforderlich: Ja, wenn `mountPoints` verwendet wird.

`readOnly`

Wenn dieser Wert `true` lautet, verfügt der Container über schreibgeschützten Zugriff auf das Volume. Lautet der Wert `false`, dann verfügt der Container über Schreibzugriff auf das Volume.

Typ: Boolesch

Erforderlich: Nein

Standard: False

`networkConfiguration`

Die Netzwerkkonfiguration für Jobs, die auf Fargate-Ressourcen ausgeführt werden. Aufgaben, die auf EC2-Ressourcen laufen, dürfen diesen Parameter nicht angeben.

```
"networkConfiguration": {
   "assignPublicIp": "string"
}
```

Typ: Objekt-Array

Erforderlich: Nein

`assignPublicIp`

Gibt an, ob die Aufgabe eine öffentliche IP-Adresse hat. Dies ist erforderlich, wenn der Job ausgehenden Netzwerkzugriff benötigt.

Typ: Zeichenfolge

Zulässige Werte: `ENABLED` | `DISABLED`

Erforderlich: Nein

Standard: `DISABLED`

`privileged`

Wenn dieser Parameter den Wert „true“ aufweist, erhält der Container erhöhte Berechtigungen auf der Host-Container-Instance (ähnlich wie der `root`-Benutzer). Dieser Parameter ordnet zu `Privileged` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--privileged` für die docker run zu. Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden. Geben Sie es nicht an und geben Sie es nicht als falsch an.

```
"privileged": true|false
```

Typ: Boolesch

Erforderlich: Nein

`readonlyRootFilesystem`

Wenn dieser Parameter den Wert „true“ aufweist, erhält der Container Lesezugriff auf das Root-Dateisystem. Dieser Parameter ordnet zu `ReadonlyRootfs` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--read-only` für die docker run zu.

```
"readonlyRootFilesystem": true|false
```

Typ: Boolesch

Erforderlich: Nein

`resourceRequirements`

Die Art und Menge einer Ressource, die einem Container zugewiesen werden soll. Zu den unterstützten Ressourcen gehören `GPU`, `MEMORY` und `VCPU`.

```
"resourceRequirements" : [
  {
    "type": "GPU",
    "value": "number"
  }
]
```

Typ: Objekt-Array

Erforderlich: Nein

`type`

Die Art einer Ressource, die einem Container zugewiesen werden soll. Zu den unterstützten Ressourcen gehören `GPU`, `MEMORY` und `VCPU`.

Typ: Zeichenfolge

Erforderlich: Ja, wenn `resourceRequirements` verwendet wird.

`value`

Die Menge der angegebenen Ressource, die für den Container reserviert werden soll. Die Werte variieren abhängig vom angegebenen `type`.

type="GPU"

Die Anzahl der physischen GPUs für die Reservierung für den Container. Die Anzahl der GPUs, die für alle Container in einem Job reserviert sind, darf die Anzahl der verfügbaren GPUs auf der Rechenressource, auf der der Job gestartet wird, nicht überschreiten.

Typ="SPEICHER"

Die harte Arbeitsspeichergrenze (in MiB), die dem Container bereitgestellt wird. Wenn Ihr Container versucht, das hier angegebene Limit zu überschreiten, wird der Container beendet. Dieser Parameter ist `Memory` im Abschnitt Einen Container erstellen der Docker-Remote-API und der Option `--memory` für die Docker-Ausführung zugeordnet. Sie müssen mindestens 4 MB Arbeitsspeicher für einen Auftrag festlegen. Dies ist erforderlich, kann jedoch an mehreren Stellen für parallele (MNP)-Aufgaben mit mehreren Knoten angegeben werden. Es muss mindestens einmal für jeden Knoten angegeben werden. Dieser Parameter ist `Memory` im Abschnitt Einen Container erstellen der Docker-Remote-API und der Option `--memory` für die Docker-Ausführung zugeordnet.

> **Note**
>
> Wenn Sie versuchen, Ihre Ressourcennutzung zu maximieren, indem Sie Ihren Jobs so viel Speicher wie möglich für einen bestimmten Instance-Typ zur Verfügung stellen, finden Sie weitere Informationen unter. DatenverarbeitungsressourceSpeicherverwaltung

Für Jobs, die auf Fargate-Ressourcen ausgeführt werden, `value` müssen sie einem der unterstützten Werte entsprechen. Außerdem müssen die `VCPU` Werte einer der Werte sein, die für diesen Speicherwert unterstützt werden.

VCPU	MEMORY
0,25 vCPU	512, 1024 und 2048 MiB
0,5 vCPU	1024-4096 MiB in Schritten von 1024 MiB
1 vCPU	2048-8192 MiB in Schritten von 1024 MiB
2 vCPU	4096-16384 MiB in Schritten von 1024 MiB
4 vCPU	8192-30720 MiB in Schritten von 1024 MiB
8 vCPU	16384-61440 MiB in Schritten von 4096 MiB
16 vCPU	32768-122880 MiB in Schritten von 8192 MiB

Typ="VCPU“

Die Anzahl von vCPUs, die für die Aufgabe reserviert sind. Dieser Parameter ist `CpuShares` im Abschnitt Einen Container erstellen der Docker-Remote-API und der Option `--cpu-shares` für die Docker-Ausführung zugeordnet. Jede vCPU entspricht 1.024 CPU-Anteilen. Für Jobs, die auf EC2-Ressourcen ausgeführt werden, müssen Sie mindestens eine vCPU angeben. Dies ist erforderlich, kann aber an mehreren Stellen angegeben werden. Es muss mindestens einmal für jeden Knoten angegeben werden.

Jobs, die auf Fargate-Ressourcen ausgeführt werden, `value` müssen einem der unterstützten Werte entsprechen und die `MEMORY` Werte müssen einem der Werte entsprechen, die für diesen VCPU-Wert unterstützt werden. Die unterstützten Werte sind 0,25, 0,5, 1, 2, 4, 8 und 16.

Der Standardwert für das Kontingent für die Anzahl der vCPU-Ressourcen von Fargate On-Demand beträgt 6 vCPUs. Weitere Informationen zu Fargate-Kontingenten finden Sie unter AWS Fargate-Kontingente in der. Allgemeine Amazon Web Services-Referenz

Typ: Zeichenfolge

Erforderlich: Ja, wenn `resourceRequirements` verwendet wird.

`secrets`

Die Geheimnisse für den Job, die als Umgebungsvariablen verfügbar gemacht werden. Weitere Informationen finden Sie unter Angabe sensibler Daten.

```
"secrets": [
    {
      "name": "secretName1",
      "valueFrom": "secretArn1"
    },
    {
      "name": "secretName2",
      "valueFrom": "secretArn2"
    }
    ...
]
```

Typ: Objekt-Array

Erforderlich: Nein

`name`

Der Name der Umgebungsvariablen, die das Geheimnis enthält.

Typ: Zeichenfolge

Erforderlich: Ja, wenn `secrets` verwendet wird.

`valueFrom`

Das Geheimnis, das dem Container exponiert werden muss. Die unterstützten Werte sind entweder der vollständige Amazon Resource Name (ARN) des Secrets Manager Manager-Geheimnisses oder der vollständige ARN des Parameters im SSM Parameter Store.

> **Note**
>
> Wenn der SSM Parameter Store-Parameter in derselben Datei AWS-Region wie der Job, den Sie starten, vorhanden ist, können Sie entweder den vollständigen ARN oder den Namen des Parameters verwenden. Wenn der Parameter in einer anderen Region existiert, muss der volle ARN angegeben werden.

Typ: Zeichenfolge

Erforderlich: Ja, wenn `secrets` verwendet wird.

`ulimits`

Eine Liste der `ulimits`-Werte, die im Container festgelegt werden sollen. Dieser Parameter ordnet zu `Ulimits` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--ulimit` für die docker run zu.

```
"ulimits": [
  {
    "name": string,
    "softLimit": integer,
    "hardLimit": integer
  }
  ...
]
```

Typ: Objekt-Array

Erforderlich: Nein

`name`

Der `type` des `ulimit`-Werts.

Typ: Zeichenfolge

Erforderlich: Ja, wenn `ulimits` verwendet wird.

`hardLimit`

Das harte Limit für den `ulimit`-Typ.

Typ: Ganzzahl

Erforderlich: Ja, wenn `ulimits` verwendet wird.

`softLimit`

Das weiche Limit für den `ulimit`-Typ.

Typ: Ganzzahl

Erforderlich: Ja, wenn `ulimits` verwendet wird.

`user`

Der Benutzername, der im Container verwendet werden soll. Dieser Parameter ordnet zu `User` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--user` für die docker run zu.

```
"user": "string"
```

Typ: Zeichenfolge

Erforderlich: Nein

`vcpus`

Dieser Parameter ist veraltet. Verwenden Sie ihn stattdessen. *`resourceRequirements`*

Die Anzahl von vCPUs, die für den Container reserviert sind.

Als Beispiel für die Verwendung`resourceRequirements`, wenn Ihre Jobdefinition Zeilen enthält, die der folgenden ähneln:

```
"containerProperties": {
  "vcpus": 2
}
```

Die entsprechende Verwendung von Zeilen `resourceRequirements` lautet wie folgt.

```
"containerProperties": {
  "resourceRequirements": [
    {
      "type": "VCPU",
      "value": "2"
    }
  ]
}
```

Typ: Ganzzahl

Erforderlich: Ja

`volumes`

Wenn Sie eine Auftragsdefinition registrieren, können Sie eine Liste der Volumes angeben, die an den Docker-Daemon in einer Container-Instance übergeben werden sollen. Die folgenden Parameter sind in den Container-Eigenschaften zulässig:

```
"volumes": [
  {
    "name": "string",
    "host": {
      "sourcePath": "string"
    },
    "efsVolumeConfiguration": {
      "authorizationConfig": {
        "accessPointId": "string",
        "iam": "string"
      },
      "fileSystemId": "string",
      "rootDirectory": "string",
      "transitEncryption": "string",
      "transitEncryptionPort": number
    }
  }
]
```

`name`

Der Name des Volumes. Bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Ziffern, Bindestriche und Unterstriche sind zulässig. Auf diesen Namen wird im Parameter `sourceVolume` der Containerdefinition `mountPoints` verwiesen.

Typ: Zeichenfolge

Erforderlich: Nein

`host`

Der Inhalt des Parameters `host` bestimmt, ob Ihr Daten-Volume auf der Host-Container-Instance erhalten bleibt und wo es gespeichert wird. Wenn der Parameter `host` leer ist, weist der Docker-Daemon einen Host-Pfad für Ihr Daten-Volume zu. Es kann jedoch nicht garantiert werden, dass die Daten auch dann bestehen bleiben, wenn der ihnen zugeordnete Container nicht mehr läuft.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Typ: Objekt

Erforderlich: Nein

`sourcePath`

Der Pfad auf der Host-Container-Instance, die dem Container angezeigt wird. Wenn dieser Parameter leer ist, weist der Docker-Daemon einen Host-Pfad für Sie zu.

Wenn der Parameter `host` den Speicherort `sourcePath` enthält, bleibt das Daten-Volume an der angegebenen Position der Host-Container-Instance erhalten, bis Sie es manuell löschen. Wenn der `sourcePath`-Wert auf der Host-Container-Instance nicht vorhanden ist, wird der Docker-Daemon ihn erstellen. Wenn der Speicherort nicht vorhanden ist, wird der Inhalt des Quellpfadordners exportiert.

Typ: Zeichenfolge

Erforderlich: Nein

`efsVolumeConfiguration`

Dieser Parameter wird angegeben, wenn Sie ein Dateisystem von Amazon Elastic File System für die Aufgabenspeicherung verwenden. Weitere Informationen finden Sie unter Amazon EFS-Volumes.

Typ: Objekt

Erforderlich: Nein

`authorizationConfig`

Die Autorisierungskonfigurationsdetails für das Amazon EFS-Dateisystem.

Typ: Zeichenfolge

Erforderlich: Nein

`accessPointId`

Die Amazon EFS-Zugriffspunkt-ID, die verwendet werden soll. Wenn ein Zugriffspunkt angegeben ist, `EFSVolumeConfiguration` muss der in der angegebene Stammverzeichniswert entweder weggelassen oder auf `/` gesetzt werden. Dadurch wird der Pfad durchgesetzt, der auf dem EFS-Zugriffspunkt festgelegt ist. Wenn ein Zugriffspunkt verwendet wird, muss in `EFSVolumeConfiguration` die Transitverschlüsselung aktiviert sein. Weitere Informationen finden Sie unter Arbeiten mit Amazon EFS-Zugriffspunkten im Amazon Elastic File System-Benutzerhandbuch.

Typ: Zeichenfolge

Erforderlich: Nein

`iam`

Legt fest, ob beim Mounten des Amazon EFS-Dateisystems die in einer Auftragsdefinition definierte AWS Batch Job-IAM-Rolle verwendet werden soll. Wenn diese Option aktiviert ist, muss die Transit-Verschlüsselung in `EFSVolumeConfiguration` aktiviert sein. Wenn dieser Parameter nicht angegeben ist, wird der Standardwert `DISABLED` verwendet. Weitere Informationen finden Sie unter Verwenden von Amazon EFS Access Points.

Typ: Zeichenfolge

Zulässige Werte: `ENABLED` | `DISABLED`

Erforderlich: Nein

`fileSystemId`

Die zu verwendende Amazon EFS-Dateisystem-ID.

Typ: Zeichenfolge

Erforderlich: Nein

`rootDirectory`

Das Verzeichnis im Amazon EFS-Dateisystem, das als Stammverzeichnis im Host bereitgestellt werden soll. Wenn dieser Parameter weggelassen wird, wird der Stamm des Amazon-EFS-Volumes verwendet. Wenn Sie dies angeben`/`, hat dies dieselbe Wirkung wie das Weglassen dieses Parameters. Die maximale Länge beträgt 4,096 Zeichen.

> **Important**
>
> Wenn in der ein EFS-Zugriffspunkt angegeben ist`authorizationConfig`, muss der Stammverzeichnisparameter entweder weggelassen oder auf gesetzt werden`/`. Dadurch wird der Pfad durchgesetzt, der auf dem Amazon EFS-Zugriffspunkt festgelegt ist.

Typ: Zeichenfolge

Erforderlich: Nein

`transitEncryption`

Legt fest, ob die Verschlüsselung für Amazon-EFS-Daten während der Übertragung zwischen dem Amazon-ECS-Host und dem Amazon-EFS-Server aktiviert werden soll. Die Transit-Verschlüsselung muss aktiviert sein, wenn die Amazon-EFS-IAM-Autorisierung verwendet wird. Wenn dieser Parameter nicht angegeben ist, wird der Standardwert `DISABLED` verwendet. Weitere Informationen finden Sie unter Verschlüsseln von Daten während der Übertragung im Benutzerhandbuch für Amazon Elastic File System.

Typ: Zeichenfolge

Zulässige Werte: `ENABLED` | `DISABLED`

Erforderlich: Nein

`transitEncryptionPort`

Der zu verwendende Port zum Senden verschlüsselter Daten zwischen dem Amazon-ECS-Host und dem Amazon EFS-Server. Wenn Sie keinen Transit-Verschlüsselungsport angeben, wird die Port-Auswahlstrategie verwendet, die der Amazon EFS-Mount-Helfer verwendet. Dieser Wert muss zwischen 0 und 65.535 liegen. Weitere Informationen finden Sie unter EFS-Mount-Helfer im Benutzerhandbuch für Amazon Elastic File System.

Typ: Ganzzahl

Erforderlich: Nein

Amazon EKS-Eigenschaften

Ein Objekt mit verschiedenen Eigenschaften, die für Amazon-EKS-basierte Aufträge spezifisch sind. Dies darf für Amazon ECS-basierte Jobdefinitionen nicht angegeben werden.

`podProperties`

Die Eigenschaften für die Kubernetes Pod-Ressourcen eines Jobs.

Typ: EksPodEigenschaften-Objekt

Erforderlich: Nein

`containers`

Die Eigenschaften des Containers, der im Amazon-EKS-Pod verwendet wird.

Typ: EksContainer Objekt

Erforderlich: Nein

`args`

Eine Reihe von Argumenten für den Einstiegspunkt. Wenn dies nicht angegeben ist, wird der `CMD` des Container-Images verwendet. Dies entspricht dem `args` Element im Entrypoint-Bereich des Pods in. Kubernetes Verweise auf Umgebungsvariablen werden mithilfe der Umgebung des Containers erweitert.

Wenn die referenzierte Umgebungsvariable nicht existiert, wird die Referenz im Befehl nicht geändert. Wenn der Verweis beispielsweise „`$(NAME1)`“ lautet und die `NAME1`-Umgebungsvariable nicht existiert, bleibt die Befehlszeichenfolge „`$(NAME1)`“ erhalten. `$$` wird ersetzt durch `$` und die resultierende Zeichenfolge wird nicht erweitert. Beispielsweise wird `$$(VAR_NAME)` als `$(VAR_NAME)` übergeben, unabhängig davon, ob die Umgebungsvariable `VAR_NAME` vorhanden ist oder nicht. Weitere Informationen finden Sie unter CMD in der Dockerfile-Referenz und Definieren Sie einen Befehl und Argumente für einen Pod in der Dokumentation. Kubernetes

Typ: Zeichenfolgen-Array

Erforderlich: Nein

`command`

Der Einstiegspunkt für den Container. Dies wird nicht in einer Shell ausgeführt. Wenn dies nicht angegeben ist, wird der `ENTRYPOINT` des Container-Images verwendet. Verweise auf Umgebungsvariablen werden mithilfe der Umgebung des Containers erweitert.

Wenn die referenzierte Umgebungsvariable nicht existiert, wird die Referenz im Befehl nicht geändert. Wenn der Verweis beispielsweise „`$(NAME1)`“ lautet und die `NAME1`-Umgebungsvariable nicht existiert, bleibt die Befehlszeichenfolge „`$(NAME1)`“ erhalten. `$$` wird ersetzt durch `$`, und die resultierende Zeichenfolge wird nicht erweitert. Beispielsweise wird `$$(VAR_NAME)` als `$(VAR_NAME)` übergeben, unabhängig davon, ob die Umgebungsvariable `VAR_NAME` vorhanden ist oder nicht. Der Einstiegspunkt kann nicht aktualisiert werden. Weitere Informationen finden Sie unter ENTRYPOINT in der Dockerfile-Referenz und Definieren Sie einen Befehl und Argumente für einen Container und Entrypoint in der Dokumentation. Kubernetes

Typ: Zeichenfolgen-Array

Erforderlich: Nein

`env`

Die Umgebungsvariablen, die an einen Container übergeben werden.

> **Note**
>
> Umgebungsvariablen können nicht mit `AWS_BATCH` beginnen. Diese Benennungskonvention ist Variablen vorbehalten, die festlegen. AWS Batch

Typ: Array von EksContainerEnvironmentVariable-Objekten

Erforderlich: Nein

`name`

Der Name der Umgebungsvariable.

Typ: Zeichenfolge

Erforderlich: Ja

`value`

Der Wert der Umgebungsvariable.

Typ: Zeichenfolge

Erforderlich: Nein

`image`

Das Docker-Image zum Starten des Containers.

Typ: Zeichenfolge

Erforderlich: Ja

`imagePullPolicy`

Die Image-Pull-Richtlinie für den Container. Unterstützte Werte sind `Always`, `IfNotPresent` und `Never`. Dieser Parameter lautet standardmäßig `IfNotPresent`. Wenn das `:latest`-Tag jedoch angegeben ist, ist es standardmäßig `Always`. Weitere Informationen finden Sie in der KubernetesDokumentation unter Bilder aktualisieren.

Typ: Zeichenfolge

Erforderlich: Nein

`name`

Name des Containers. Wenn der Name nicht angegeben ist, wird der Standardname „`Default`“ verwendet. Jeder Container in einem Pod muss einen eindeutigen Namen haben.

Typ: Zeichenfolge

Erforderlich: Nein

`resources`

Die Art und Menge an Ressourcen, die einem Container zugewiesen werden sollen. Zu den unterstützten Ressourcen gehören `memory`, `cpu` und `nvidia.com/gpu`. Weitere Informationen finden Sie in der KubernetesDokumentation unter Ressourcenverwaltung für Pods und Container.

Typ: EksContainerResourceRequirements Objekt

Erforderlich: Nein

`limits`

Typ und Menge der Ressourcen, die für den Container reserviert werden sollen. Die Werte variieren abhängig vom angegebenen `name`. Ressourcen können entweder mithilfe der `limits`- oder `requests`-Objekte angefordert werden.

memory

Die harte Speichergrenze (in MB) für den Container, unter Verwendung ganzer Zahlen, mit dem Suffix „Mi“. Wenn Ihr Container versucht, das angegebene Limit zu überschreiten, wird der Container beendet. Sie müssen mindestens 4 MB Arbeitsspeicher für einen Auftrag festlegen. `memory` kann in `limits`, `requests` oder in beiden angegeben werden. Wenn `memory` in beiden Orten angegeben ist, muss der Wert, der in `limits` angegeben ist, dem Wert entsprechen, der in `requests` angegeben ist.

> **Note**
>
> Um Ihre Ressourcenauslastung zu maximieren, stellen Sie Ihren Aufträgen so viel Arbeitsspeicher wie möglich für den spezifischen Instance-Typ zur Verfügung, den Sie verwenden. Um zu erfahren wie dies geht, vgl. DatenverarbeitungsressourceSpeicherverwaltung.

cpu

Die Anzahl der für den Container reservierten CPUs. Werte müssen ein gerade Vielfaches von `0.25` sein. `cpu` kann in `limits`, `requests` oder in beiden angegeben werden. Wenn `cpu` in beiden Orten angegeben ist, muss der Wert, der in `limits` angegeben ist, mindestens so groß sein wie der Wert, der in `requests` angegeben ist.

nvidia.com/gpu

Die Anzahl der GPUs, die für den Container reserviert sind. Werte müssen eine ganze Zahl sein. `memory` kann in `limits`, `requests` oder in beiden angegeben werden. Wenn `memory` in beiden Orten angegeben ist, muss der Wert, der in `limits` angegeben ist, dem Wert entsprechen, der in `requests` angegeben ist.

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Längenbeschränkungen des Wertes: Minimale Länge von 1. Maximale Länge beträgt 256 Zeichen.

Erforderlich: Nein

`requests`

Typ und Menge der für den Container anzufordernden Ressourcen. Die Werte variieren abhängig vom angegebenen `name`. Ressourcen können entweder mithilfe der `limits`-Objekte oder der `requests`-Objekte angefordert werden.

memory

Die harte Speichergrenze (in MB) für den Container, unter Verwendung ganzer Zahlen, mit dem Suffix „Mi“. Wenn Ihr Container versucht, das angegebene Limit zu überschreiten, wird der Container beendet. Sie müssen mindestens 4 MB Arbeitsspeicher für einen Auftrag festlegen. `memory` kann in `limits`, `requests` oder in beiden angegeben werden. Wenn `memory` in beiden angegeben ist, muss der Wert, der in `limits` angegeben ist, dem Wert entsprechen, der in `requests` angegeben ist.

> **Note**
>
> Wenn Sie versuchen, Ihre Ressourcennutzung zu maximieren, indem Sie Ihren Jobs so viel Speicher wie möglich für einen bestimmten Instance-Typ zur Verfügung stellen, finden Sie unterDatenverarbeitungsressourceSpeicherverwaltung.

cpu

Die Anzahl von CPUs, die für den Container reserviert sind. Werte müssen ein gerade Vielfaches von `0.25` sein. `cpu` kann in `limits`, `requests` oder in beiden angegeben werden. Wenn `cpu` in beiden angegeben ist, muss der Wert, der in `limits` angegeben ist, mindestens so groß sein wie der Wert, der in `requests` angegeben ist.

nvidia.com/gpu

Die Anzahl der GPUs die für den Container reserviert sind. Werte müssen eine ganze Zahl sein. `nvidia.com/gpu` kann in `limits`, `requests` oder in beiden

angegeben werden. Wenn `nvidia.com/gpu` in beiden angegeben ist, muss der Wert, der in `limits` angegeben ist, dem Wert entsprechen, der in `requests` angegeben ist.

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Längenbeschränkungen des Wertes: Minimale Länge von 1. Maximale Länge beträgt 256 Zeichen.

Erforderlich: Nein

`securityContext`

Der Sicherheitskontext für einen Auftrag. Weitere Informationen finden Sie in der KubernetesDokumentation unter Konfigurieren eines Sicherheitskontextes für einen Pod oder Container.

Typ: EksContainerSecurityContext Objekt

Erforderlich: Nein

`privileged`

Wenn dieser Parameter den Wert `true` aufweist, erhält der Container erhöhte Berechtigungen auf der Host-Container-Instance. Die Ebene der Berechtigungen ähnelt den `root` Benutzerberechtigungen. Der Standardwert ist `false`. Dieser Parameter entspricht der `privileged` Richtlinie in den Sicherheitsrichtlinien für privilegierte Pods in der KubernetesDokumentation.

Typ: Boolesch

Erforderlich: Nein

`readOnlyRootFilesystem`

Wenn dieser Parameter den Wert `true` aufweist, erhält der Container Lesezugriff auf das Root-Dateisystem. Der Standardwert ist `false`. Dieser Parameter ist der `ReadOnlyRootFilesystem` Richtlinie in den Pod-Sicherheitsrichtlinien für Volumes und Dateisysteme in der KubernetesDokumentation zugeordnet.

Typ: Boolesch

Erforderlich: Nein

`runAsGroup`

Wenn dieser Parameter angegeben ist, wird der Container mit der angegebenen Gruppen-ID (`gid`) ausgeführt. Wird dieser Parameter nicht angegeben, wird standardmäßig die Gruppe verwendet, die in den Image-Metadaten angegeben ist. Dieser Parameter ist einer `MustRunAs` Richtlinie in den Pod-Sicherheitsrichtlinien für Benutzer und Gruppen in der KubernetesDokumentation zugeordnet. `RunAsGroup`

Type: Long

Erforderlich: Nein

`runAsNonRoot`

Wenn dieser Parameter angegeben ist, wird der Container als Benutzer mit einem `uid` mit anderen Wert als 0 ausgeführt. Wenn dieser Parameter nicht angegeben ist, wird eine solche Regel durchgesetzt. Dieser Parameter ist einer `MustRunAsNonRoot` Richtlinie in den Pod-Sicherheitsrichtlinien für Benutzer und Gruppen in der KubernetesDokumentation zugeordnet. `RunAsUser`

Type: Long

Erforderlich: Nein

`runAsUser`

Wenn dieser Parameter angegeben ist, wird der Container mit der angegebenen Benutzer-ID (`uid`) ausgeführt. Wird dieser Parameter nicht angegeben, wird standardmäßig der in den Image-Metadaten angegebene Benutzer verwendet. Dieser Parameter ist einer `MustRanAs` Richtlinie in den Pod-Sicherheitsrichtlinien für Benutzer und Gruppen in der KubernetesDokumentation zugeordnet. `RunAsUser`

Type: Long

Erforderlich: Nein

`volumeMounts`

Die Volume-Mounts für einen Container für einen Amazon-EKS-Auftrag. Weitere Informationen zu Volumes und zum Einhängen von Volumes finden Sie in Kubernetes der KubernetesDokumentation unter Volumes.

Typ: Array von EksContainerVolumeMount-Objekten

Erforderlich: Nein

`mountPath`

Der Pfad auf dem Container, in dem das Volume gemountet ist.

Typ: Zeichenfolge

Erforderlich: Nein

`name`

Der Name des zu mountenden Volumes. Dieser muss mit dem Namen eines der Volumes im Pod übereinstimmen.

Typ: Zeichenfolge

Erforderlich: Nein

`readOnly`

Wenn dieser Wert `true` lautet, verfügt der Container über schreibgeschützten Zugriff auf das Volume. Andernfalls kann der Container auf das Volume schreiben. Der Standardwert ist `false`.

Typ: Boolesch

Erforderlich: Nein

`dnsPolicy`

Die DNS-Richtlinie für den Pod. Der Standardwert ist `ClusterFirst`. Wenn der `hostNetwork`-Parameter nicht angegeben ist, ist der Standardwert `ClusterFirstWithHostNet`. `ClusterFirst` gibt an, dass jede DNS-Abfrage, die nicht mit dem konfigurierten Cluster-Domain-Suffix übereinstimmt, an den vom Knoten geerbten Upstream-Nameserver weitergeleitet wird. Wenn `dnsPolicy` im RegisterJobDefinition-API-Vorgang kein Wert für angegeben wurde, wird weder `dnsPolicy` von DescribeJobDefinitionen noch von DescribeJobsAPI-Vorgängen ein Wert für zurückgegeben. Die Pod-Spezifikationseinstellung enthält entweder `ClusterFirst` oder `ClusterFirstWithHostNet`, abhängig vom Wert des `hostNetwork`-Parameters. Weitere Informationen finden Sie in der KubernetesDokumentation zur DNS-Richtlinie von Pod.

Zulässige Werte: `Default` | `ClusterFirst` | `ClusterFirstWithHostNet`

Typ: Zeichenfolge

Erforderlich: Nein

`hostNetwork`

Gibt an, ob der Pod die Netzwerk-IP-Adresse der Hosts verwendet. Der Standardwert ist `true`. Wenn Sie dies auf einstellen, wird das Kubernetes Pod-Netzwerkmodell `false` aktiviert. Die meisten AWS Batch Workloads sind ausschließlich für ausgehende Verbindungen bestimmt und erfordern nicht den Mehraufwand der IP-Zuweisung für jeden Pod für eingehende Verbindungen. Weitere Informationen finden Sie in der Dokumentation unter Host-Namespaces und Pod-Netzwerke. Kubernetes

Typ: Boolesch

Erforderlich: Nein

`serviceAccountName`

Der Name des Servicekontos, das zum Ausführen des Pods verwendet wird. Weitere Informationen finden Sie unter KubernetesDienstkonten und Konfiguration eines Kubernetes Dienstkontos für die Übernahme einer IAM-Rolle im Amazon EKS-Benutzerhandbuch und Servicekonten für Pods konfigurieren in der KubernetesDokumentation.

Typ: Zeichenfolge

Erforderlich: Nein

`volumes`

Gibt die Volumes für eine Auftragsdefinition an, die Amazon-EKS-Ressourcen verwendet.

Typ: Array von EksVolume-Objekten

Erforderlich: Nein

EmptyDir

Gibt die Konfiguration eines Volumes an Kubernetes`emptyDir`. Ein `emptyDir`-Volume wird zuerst erstellt, wenn ein Pod einem Knoten zugewiesen wird. Es existiert, solange der Pod auf diesem Knoten läuft. Das `emptyDir`-Volume ist zunächst leer. Alle Container im Pod können die Dateien im `emptyDir`-Volume lesen und schreiben. Das `emptyDir`-Volume kann jedoch in jedem Container auf den gleichen oder unterschiedlichen Pfaden gemountet werden. Wenn ein Pod aus irgendeinem Grund von einem Knoten entfernt wird, werden die Daten in `emptyDir` dauerhaft gelöscht. Weitere Informationen finden Sie in der Dokumentation unter EmptyDir. Kubernetes

Typ: Dir-Objekt EksEmpty

Erforderlich: Nein

Medium

Das Medium zum Speichern des Volumes. Der Standardwert ist eine leere Zeichenfolge, die den Speicher des Knotens verwendet.

""

(Standard) Verwenden Sie den Festplattenspeicher des Knotens.

„Arbeitsspeicher“

Verwenden Sie das `tmpfs`-Volume, das vom RAM des Knotens unterstützt wird. Der Inhalt des Volumes geht verloren, wenn der Knoten neu gestartet wird, und jeder Speicherplatz auf dem Volume wird auf das Speicherlimit des Containers angerechnet.

Typ: Zeichenfolge

Erforderlich: Nein

Größenbeschränkung

Die maximale Größe des Volumes. Standardmäßig ist keine maximale Größe definiert.

Typ: Zeichenfolge

Längenbeschränkungen: Minimale Länge beträgt 1 Zeichen. Maximale Länge beträgt 256 Zeichen.

Erforderlich: Nein

Host-Pfad

Gibt die Konfiguration eines Kubernetes `hostPath` Volumes an. Ein `hostPath`-Volume mountet eine vorhandene Datei oder ein Verzeichnis aus dem Dateisystem des Hostknotens in Ihren Pod. Weitere Informationen finden Sie in der KubernetesDokumentation unter HostPath.

Typ: EksHostPath-Objekt

Erforderlich: Nein

path

Der Pfad der Datei oder des Verzeichnisses auf dem Host, das in Containern auf dem Pod gemountet werden soll.

Typ: Zeichenfolge

Erforderlich: Nein

Name

Der Name des Volumes. Der Name muss als DNS-Sub-Domain-Name zulässig sein. Weitere Informationen finden Sie in der KubernetesDokumentation unter DNS-Subdomänennamen.

Typ: Zeichenfolge

Erforderlich: Ja

Secret

Gibt die Konfiguration eines Kubernetes `secret` Volumes an. Weitere Informationen finden Sie in der KubernetesDokumentation unter Secret.

Typ: EksSecret Objekt

Erforderlich: Nein

optional

Gibt an, ob das Geheimnis oder die Schlüssel des Geheimnisses definiert werden müssen.

Typ: Boolesch

Erforderlich: Nein

Geheimer Name

Der Name des Secrets. Der Name muss als DNS-Sub-Domain-Name zulässig sein. Weitere Informationen finden Sie in der Dokumentation unter DNS-Subdomänennamen. Kubernetes

Typ: Zeichenfolge

Erforderlich: Ja

Funktionen der Plattform

`platformCapabilities`

Die Plattformfunktionen, die für die Jobdefinition erforderlich sind. Wenn kein Wert angegeben wird, wird standardmäßig der Wert `EC2` verwendet. Für Jobs, die auf Fargate-Ressourcen laufen, `FARGATE` ist angegeben.

> **Note**
>
> Wenn der Job auf Amazon EKS-Ressourcen ausgeführt wird, dürfen Sie keine Angaben machen`platformCapabilities`.

Typ: Zeichenfolge

Zulässige Werte: `EC2` | `FARGATE`

Erforderlich: Nein

Tags weitergeben

`propagateTags`

Gibt an, ob die Tags aus dem Auftrag oder der Auftragsdefinition an die entsprechende Amazon ECS-Aufgabe weitergegeben werden sollen. Wenn kein Wert angegeben wird, werden die Tags nicht weitergegeben. Tags können nur dann an die Aufgaben weitergegeben werden, wenn die Aufgabe erstellt wird. Bei Tags mit demselben Namen haben Auftrags-Tags Vorrang vor Auftragsdefinitions-Tags. Wenn die Gesamtzahl der kombinierten Tags aus dem Job und der Auftragsdefinition über 50 liegt, wird der Job in den `FAILED` Status verschoben.

> **Note**
>
> Wenn der Job auf Amazon EKS-Ressourcen ausgeführt wird, dürfen Sie keine Angaben machen`propagateTags`.

Typ: Boolesch

Erforderlich: Nein

Eigenschaften des Knotens

`nodeProperties`

Wenn Sie eine parallel Auftragsdefinition mit mehreren Knoten registrieren, müssen Sie eine Liste von Knoteneigenschaften angeben. Diese Knoteneigenschaften definieren die Anzahl der Knoten, die in Ihrem Job verwendet werden sollen, den Hauptknotenindex und die verschiedenen zu verwendenden Knotenbereiche. Wenn der Job auf Fargate-Ressourcen ausgeführt wird, können Sie keine Angaben machen. `nodeProperties` Nutzen Sie stattdessen `containerProperties`. Die folgenden Knoteneigenschaften sind in einer Auftragsdefinition zulässig. Weitere Informationen finden Sie unter parallel Jobs mit mehreren Knoten.

> **Note**
>
> Wenn der Job auf Amazon EKS-Ressourcen ausgeführt wird, dürfen Sie keine Angaben machen`nodeProperties`.

Typ: NodeProperties Objekt

Erforderlich: Nein

`mainNode`

Gibt den Knotenindex für den Hauptknoten eines parallelen Auftrags mit mehreren Knoten an. Dieser Knotenindexwert muss kleiner als die Anzahl der Knoten sein.

Typ: Ganzzahl

Erforderlich: Ja

`numNodes`

Die Anzahl der Knoten, die mit einem parallelen Auftrag mit mehreren Knoten verknüpft sind.

Typ: Ganzzahl

Erforderlich: Ja

`nodeRangeProperties`

Eine Liste von Knotenbereichen und deren Eigenschaften, die mit einem parallelen Auftrag mit mehreren Knoten verknüpft sind.

> **Note**
>
> Eine Knotengruppe ist eine identische Gruppe von Job-Knoten, die alle dieselben Container-Eigenschaften haben. Sie können AWS Batch damit bis zu fünf verschiedene Knotengruppen für jeden Job angeben.

Typ: Anordnung von NodeRangeEigenschaftenobjekten

Erforderlich: Ja

`targetNodes`

Der Bereich der Knoten, die Knotenindexwerte verwenden. Ein Bereich von `0:3` zeigt Knoten mit Indexwerten von `0` bis `3` an. Wenn der Startwert für den Bereich weggelassen wird (`:n`), wird 0 verwendet, um den Bereich zu starten. Wenn der Endbereichswert ausgelassen wird (`n:`), wird für den Endbereich der höchstmögliche Knotenindex verwendet wird. Ihre gesamten Knotenbereiche müssen alle Knoten berücksichtigen (`0:n`). Sie können Knotenbereiche verschachteln, zum Beispiel `0:10` und`4:5`. In diesem Fall haben die `4:5` Bereichseigenschaften Vorrang vor den `0:10` Eigenschaften.

Typ: Zeichenfolge

Erforderlich: Nein

`container`

Die Container-Details für den Knotenbereich. Weitere Informationen finden Sie unter Eigenschaften des Containers.

Typ: ContainerProperties Objekt

Erforderlich: Nein

Strategie erneut versuchen

`retryStrategy`

Wenn Sie eine Auftragsdefinition registrieren, können Sie optional eine Wiederholungsstrategie für fehlgeschlagene Aufträge angeben, die mit dieser Auftragsdefinition übermittelt werden. Jede

Wiederholungsstrategie, die während eines SubmitJobVorgangs angegeben wird, hat Vorrang vor der hier definierten Wiederholungsstrategie. Standardmäßig erfolgt ein Ausführungsversuch pro Auftrag. Wenn Sie mehr als einen Versuch angeben, wird der Job erneut versucht, falls er fehlschlägt. Beispiele für einen fehlgeschlagenen Versuch sind, dass der Job einen Exit-Code ungleich Null zurückgibt oder die Container-Instance beendet wird. Weitere Informationen finden Sie unter Automatisierte Auftragswiederholungen.

Typ: RetryStrategy Objekt

Erforderlich: Nein

`attempts`

Gibt die Zahl der Wiederholungsversuche an, um einen Auftrag in den Status `RUNNABLE` zu verschieben. Sie können zwischen einem und zehn Versuche angeben. Wenn `attempts` größer als 1 ist, wird der Auftrag bei einem auftretenden Fehler entsprechend oft wiederholt, bis er in den Status `RUNNABLE` wechselt.

```
"attempts": integer
```

Typ: Ganzzahl

Erforderlich: Nein

`evaluateOnExit`

Array mit bis zu 5 Objekten, die Bedingungen angeben, unter denen der Job wiederholt wird oder fehlschlägt. Wenn dieser Parameter angegeben wird, muss der Parameter `attempts` ebenfalls angegeben werden. Wenn angegeben `evaluateOnExit` ist, aber keiner der Einträge zutrifft, wird der Job erneut versucht.

```
"evaluateOnExit": [
   {
      "action": "string",
      "onExitCode": "string",
      "onReason": "string",
      "onStatusReason": "string"
   }
]
```

Typ: Array von EvaluateOnExit-Objekten

Erforderlich: Nein

`action`

Gibt die Aktion an, die durchgeführt werden soll, wenn alle angegebenen Bedingungen (`onStatusReason`, `onReason` und `onExitCode`) erfüllt sind. Bei den Werten muss die Groß- und Kleinschreibung beachtet werden.

Typ: Zeichenfolge

Erforderlich: Ja

Zulässige Werte: `RETRY` | `EXIT`

`onExitCode`

Enthält ein Glob-Muster, das mit der Dezimaldarstellung von verglichen werden soll`ExitCode`, das für einen Job zurückgegeben wurde. Jedes Muster kann bis zu 512 Zeichen lang sein. Es kann nur Zahlen enthalten. Es darf keine Buchstaben oder Sonderzeichen enthalten. Es kann optional mit einem Sternchen (*) enden, so dass nur der Anfang der Zeichenfolge genau übereinstimmen muss.

Typ: Zeichenfolge

Erforderlich: Nein

`onReason`

Enthält ein Glob-Muster, das mit dem verglichen werden soll`Reason`, das für einen Job zurückgegeben wurde. Jedes Muster kann bis zu 512 Zeichen lang sein. Es kann Buchstaben, Zahlen, Punkte (.), Doppelpunkte (:) und Leerzeichen (Leerzeichen, Tabulatoren) enthalten. Es kann optional mit einem Sternchen (*) enden, so dass nur der Anfang der Zeichenfolge genau übereinstimmen muss.

Typ: Zeichenfolge

Erforderlich: Nein

`onStatusReason`

Enthält ein Glob-Muster, das mit dem verglichen werden soll`StatusReason`, das für einen Job zurückgegeben wurde. Jedes Muster kann bis zu 512 Zeichen lang sein. Es

kann Buchstaben, Zahlen, Punkte (.), Doppelpunkte (:) und Leerzeichen (Leerzeichen, Tabulatoren) enthalten. Es kann optional mit einem Sternchen (*) enden, so dass nur der Anfang der Zeichenfolge genau übereinstimmen muss.

Typ: Zeichenfolge

Erforderlich: Nein

Priorität bei der Planung

`schedulingPriority`

Die Planungspriorität für Jobs, die mit dieser Jobdefinition eingereicht werden. Dies betrifft nur Aufträge in Auftrags-Warteschlangen mit einer Fair-Share-Richtlinie. Aufgaben mit einer höheren Planungspriorität werden vor Aufgaben mit einer niedrigeren Planungspriorität geplant.

Der unterstützte Mindestwert ist 0 und der unterstützte Höchstwert ist 9999.

Typ: Ganzzahl

Erforderlich: Nein

Tags

`tags`

Schlüssel-Wert-Paar-Tags, die der Jobdefinition zugeordnet werden sollen. Weitere Informationen finden Sie unter Markieren Ihrer AWS Batch-Ressourcen.

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Erforderlich: Nein

Timeout

`timeout`

Sie können eine Zeitüberschreitungsdauer für Ihre Jobs konfigurieren, sodass der Job AWS Batch beendet wird, wenn ein Job länger als diesen Zeitraum läuft. Weitere Informationen finden Sie

unter Timeouts bei der Job. Wenn ein Job aufgrund eines Timeouts beendet wird, wird er nicht erneut versucht. Jede Timeout-Konfiguration, die während eines SubmitJobVorgangs angegeben wird, hat Vorrang vor der hier definierten Timeout-Konfiguration. Weitere Informationen finden Sie unter Timeouts bei der Job.

Typ: JobTimeout Objekt

Erforderlich: Nein

`attemptDurationSeconds`

Die Zeitdauer in Sekunden (gemessen anhand des `startedAt` Zeitstempels des Auftragsversuchs) nach AWS Batch Beendigung nicht abgeschlossener Jobs. Der Mindestwert für die Zeitüberschreitung beträgt 60 Sekunden.

Bei Array-Aufträgen gilt das Timeout für die untergeordneten Aufträge, nicht für den übergeordneten Array-Auftrag.

Bei parallel Aufträgen mit mehreren Knoten (MNP) gilt das Timeout für den gesamten Auftrag, nicht für die einzelnen Knoten.

Typ: Ganzzahl

Erforderlich: Nein

Jobdefinitionen erstellen mit EcsProperties

Mithilfe `EcsProperties`von AWS Batch Jobdefinitionen können Sie Hardware, Sensoren, 3D-Umgebungen und andere Simulationen in separaten Containern modellieren. Sie können diese Funktion verwenden, um Ihre Workload-Komponenten logisch zu organisieren und sie von der Hauptanwendung zu trennen. Diese Funktion kann mit Amazon Elastic Container Service (AWS Batch Amazon ECS), Amazon Elastic Kubernetes Service (Amazon EKS) und verwendet werden. AWS Fargate

`ContainerProperties`im Vergleich zu Jobdefinitionen **`EcsProperties`**

Sie können wählen, ob Sie `EcsProperties`Jobdefinitionen verwenden möchten, je nach Anwendungsfall. `ContainerProperties` Auf hoher Ebene `EcsProperties` ähnelt das Ausführen von AWS Batch Jobs mit dem Ausführen von Jobs mit einem`ContainerProperties`.

Die alte Auftragsdefinitionsstruktur, die verwendet wird`ContainerProperties`, wird weiterhin unterstützt. Wenn Sie derzeit Workflows haben, die diese Struktur verwenden, können Sie sie weiterhin ausführen.

Der Hauptunterschied besteht darin, dass der Auftragsdefinition ein neues Objekt hinzugefügt wurde, um die Basisdefinitionen `EcsProperties` zu berücksichtigen.

Eine Jobdefinition, die `ContainerProperties` auf Amazon ECS und Fargate verwendet wird, hat beispielsweise die folgende Struktur:

```
{
   "containerProperties": {
     ...
     "image": "my_ecr_image1",
     ...
  },
...
}
```

Eine Jobdefinition, die `EcsProperties` auf Amazon ECS und Fargate verwendet wird, hat die folgende Struktur:

```
{
  "ecsProperties": {
    "taskProperties": [{
      "containers": [
        {
          ...
          "image": "my_ecr_image1",
          ...
        },
        {
          ...
          "image": "my_ecr_image2",
          ...
        },
```

Allgemeine Änderungen an den APIs AWS Batch

Im Folgenden werden einige der wichtigsten Unterschiede bei der Verwendung der `EcsProperties` und der `EcsProperties` API-Datentypen näher beschrieben:

- Viele der Parameter, die darin verwendet werden, befinden `ContainerProperties` sich in`TaskContainerProperties`. Einige Beispiele umfassen`command`,`image`,`privileged`,`secrets`, und`users`. Sie sind alle darin zu finden TaskContainerProperties.
- Einige der `TaskContainerProperties` Parameter haben in der alten Struktur keine funktionalen Entsprechungen. Einige Beispiele umfassen,,`dependsOn`, `essential` `nameipcMode`, und. `pidMode` Weitere Informationen finden Sie unter EcsTaskDetailsund TaskContainerProperties.

  Außerdem gibt es in der Struktur für einige `ContainerProperties` Parameter keine Entsprechungen oder Anwendungsmöglichkeiten. `EcsProperties` In `container` wurde durch ersetzt `taskProperties`, `containers` sodass das neue Objekt bis zu zehn Elemente aufnehmen kann. Weitere Informationen finden Sie unter:ContainerProperties und:containersRegisterJobDefinition. EcsTaskProperties
- `taskRoleArn`ist funktionell `jobRoleArn` äquivalent zu. Weitere Informationen finden Sie unter EcsTaskProperties: taskRoleArn und ContainerProperties: jobRoleArn.
- Sie können einen (1) bis zehn (10) Container in die `EcsProperties` Struktur aufnehmen. Weitere Informationen finden Sie unter:ContainerEcsTaskProperties.
- Die Objekte `taskProperties` und instanceTypes sind Arrays, akzeptieren aber derzeit nur ein Element. Zum Beispiel :TaskProperties EcsProperties und:InstanceTypes. NodeRangeProperty

Jobdefinitionen mit mehreren Containern für Amazon ECS

Um der Multi-Container-Struktur für Amazon ECS Rechnung zu tragen, sind einige API-Datentypen unterschiedlich. Zum Beispiel

- `ecsProperties`ist dieselbe Ebene wie `containerProperties` in der Einzelcontainer-Definition. Weitere Informationen finden Sie EcsPropertiesim AWS Batch API-Referenzhandbuch.
- `taskProperties`enthält die für die Amazon ECS-Aufgabe definierten Eigenschaften. Weitere Informationen finden Sie EcsPropertiesim AWS Batch API-Referenzhandbuch.
- `containers`enthält ähnliche Informationen wie `containerProperties` in der Einzelcontainer-Definition. Der Hauptunterschied besteht darin, dass Sie bis zu zehn Container definieren `containers` können. Weitere Informationen finden Sie unter ecs:Containers TaskProperties im AWS Batch API-Referenzhandbuch.
- `essential`Der Parameter gibt an, wie sich der Container auf den Job auswirkt. Alle wichtigen Container müssen erfolgreich abgeschlossen werden (beenden als 0), damit der Job fortgesetzt

werden kann. Wenn ein Container, der als essentail markiert ist, ausfällt (mit einem Wert ungleich 0 beendet wird), schlägt der Job fehl.

Der Standardwert ist `true` und mindestens ein Container muss als markiert sein. `essential` Weitere Informationen finden Sie unter `essential` in der AWS Batch -API-Referenz.

- Mit dem `dependsOn`Parameter können Sie eine Liste von Container-Abhängigkeiten definieren. Weitere Informationen finden Sie unter `dependsOn` in der AWS Batch -API-Referenz.

> **Note**
>
> Die Komplexität der `dependsOn` Liste und die zugehörige Container-Laufzeit können sich auf die Startzeit Ihres Jobs auswirken. Wenn die Ausführung der Abhängigkeiten lange dauert, bleibt der Job so lange in einem `STARTING` Zustand, bis sie abgeschlossen sind.

Weitere Informationen zur UND-Struktur finden Sie unter RegisterJobDefinitionAnforderungssyntax für ecsProperties. `ecsProperties`

Jobdefinitionen mit mehreren Containern für Amazon EKS

Um der Multi-Container-Struktur für Amazon EKS Rechnung zu tragen, sind einige API-Datentypen unterschiedlich. Zum Beispiel

- `name`ist ein eindeutiger Bezeichner für den Container. Dieses Objekt ist nicht für einen einzelnen Container erforderlich, ist jedoch erforderlich, wenn mehrere Container in einem Pod definiert werden. Wenn es `name` nicht für einzelne Container definiert ist, wird der Standardname`default`, angewendet.
- `initContainers`sind innerhalb des eksPodPropertiesDatentyps definiert. Sie werden vor Anwendungscontainern ausgeführt, laufen immer bis zum Abschluss und müssen erfolgreich abgeschlossen werden, bevor der nächste Container gestartet wird.

  Diese Container sind beim Amazon EKS Connector-Agenten registriert und speichern die Registrierungsinformationen im Backend-Datenspeicher von Amazon Elastic Kubernetes Service. Das `initContainers` Objekt kann bis zu zehn (10) Elemente aufnehmen. Weitere Informationen finden Sie in der KubernetesDokumentation unter Init Containers.

> **Note**
>
> Das `initContainers` Objekt kann sich auf die Startzeit Ihres Jobs auswirken. Wenn die Ausführung lange `initContainers` dauert, bleibt der Job so lange erhalten, `STARTING` bis sie abgeschlossen sind.

- `shareProcessNamespace`gibt an, ob sich die Container im Pod denselben Prozess-Namespace teilen können. Die Standardwerte sind. `false` Diese Einstellung `true` soll es Containern ermöglichen, Prozesse in anderen Containern, die sich im selben Pod befinden, zu sehen und zu signalisieren.
- Jeder Container ist wichtig. Alle Container müssen erfolgreich abgeschlossen werden (beenden als 0), damit der Job erfolgreich ist. Wenn ein Container ausfällt (mit einem anderen Wert als 0 beendet wird), schlägt der Job fehl.

Weitere Informationen zur UND-Struktur finden Sie unter RegisterJobDefinitionAnforderungssyntax für eksProperties. `eksProperties`

AWS Batch -Auftragsszenarien mit EcsProperties

Um zu veranschaulichen, wie AWS Batch Auftragsdefinitionen, die verwenden, basierend auf Ihren Anforderungen strukturiert werden `EcsProperties` können, werden in diesem Thema die folgenden `RegisterJobDefinition` Nutzlasten vorgestellt. Sie können diese Beispiele in eine Datei kopieren, sie an Ihre Bedürfnisse anpassen und dann mit der AWS Command Line Interface (AWS CLI) aufrufen`RegisterJobDefinition`.

AWS Batch -Auftrag für Amazon Elastic Container Service in Amazon Elastic Compute Cloud

```
{
    "jobDefinitionName": "multicontainer-ecs-ec2",
    "type": "container",
    "ecsProperties": {
        "taskProperties": [
          {
            "containers": [
              {
                "name": "c1",
                "essential": false,
```

```
                "command": [
                  "echo",
                  "hello world"
                ],
                "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                "resourceRequirements": [
                  {
                    "type": "VCPU",
                    "value": "2"
                  },
                  {
                    "type": "MEMORY",
                    "value": "4096"
                  }
                ]
              },
              {
                "name": "c2",
                "essential": true,
                "command": [
                  "echo",
                  "hello world"
                ],
                "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                "resourceRequirements": [
                  {
                    "type": "VCPU",
                    "value": "6"
                  },
                  {
                    "type": "MEMORY",
                    "value": "12288"
                  }
                ]
              }
            ]
          }
        ]
  }
}
```

AWS Batch -Auftrag für Amazon ECS auf AWS Fargate

```
{
    "jobDefinitionName": "multicontainer-ecs-fargate",
    "type": "container",
    "platformCapabilities": [
        "FARGATE"
    ],
    "ecsProperties": {
        "taskProperties": [
          {
            "containers": [
              {
                "name": "c1",
                "command": [
                  "echo",
                  "hello world"
                ],
                "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                "resourceRequirements": [
                  {
                    "type": "VCPU",
                    "value": "2"
                  },
                  {
                    "type": "MEMORY",
                    "value": "4096"
                  }
                ]
              },
              {
                "name": "c2",
                "essential": true,
                "command": [
                  "echo",
                  "hello world"
                ],
                "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                "resourceRequirements": [
                  {
                    "type": "VCPU",
                    "value": "6"
                  },
```

```
                {
                  "type": "MEMORY",
                  "value": "12288"
                }
              ]
            }
          ],
          "executionRoleArn": "arn:aws:iam::1112223333:role/ecsTaskExecutionRole"
        }
      ]
  }
}
```

AWS Batch -Auftrag für Amazon Elastic Kubernetes Service

```
{
  "jobDefinitionName": "multicontainer-eks",
  "type": "container",
  "eksProperties": {
    "podProperties": {
      "shareProcessNamespace": true,
      "initContainers": [
        {
          "name": "init-container",
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "echo"
          ],
          "args": [
            "hello world"
          ],
          "resources": {
            "requests": {
              "cpu": "1",
              "memory": "512Mi"
            }
          }
        },
        {
          "name": "init-container-2",
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "echo",
```

```
            "my second init container"
          ],
          "resources": {
            "requests": {
              "cpu": "1",
              "memory": "512Mi"
            }
          }
        }
      ],
      "containers": [
        {
          "name": "c1",
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "echo world"
         ],
          "resources": {
            "requests": {
              "cpu": "1",
              "memory": "512Mi"
            }
          }
        },
        {
          "name": "sleep-container",
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "sleep",
            "20"
          ],
          "resources": {
            "requests": {
              "cpu": "1",
              "memory": "512Mi"
            }
          }
        }
      ]
    }
  }
}
```

Paralleler AWS Batch Auftrag mit mehreren Knoten (MNP) mit mehreren Containern pro Knoten

```
{
  "jobDefinitionName": "multicontainer-mnp",
  "type": "multinode",
  "nodeProperties": {
    "numNodes": 6,
    "mainNode": 0,
    "nodeRangeProperties": [
      {
        "targetNodes": "0:5",
        "ecsProperties": {
          "taskProperties": [
            {
              "containers": [
                {
                  "name": "range05-c1",
                  "command": [
                    "echo",
                    "hello world"
                  ],
                  "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                  "resourceRequirements": [
                    {
                      "type": "VCPU",
                      "value": "2"
                    },
                    {
                      "type": "MEMORY",
                      "value": "4096"
                    }
                  ]
                },
                {
                  "name": "range05-c2",
                  "command": [
                    "echo",
                    "hello world"
                  ],
                  "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                  "resourceRequirements": [
                    {
```

```
                    "type": "VCPU",
                    "value": "2"
                  },
                  {
                    "type": "MEMORY",
                    "value": "4096"
                  }
                ]
              }
            ]
          }
        ]
      }
    }
  ]
}
}
```

Verwenden des awslogs-Protokolltreibers

Standardmäßig AWS Batch ermöglicht dem `awslogs` Protokolltreiber das Senden von Protokollinformationen an - CloudWatch Protokolle. Sie können diese Funktion verwenden, um verschiedene Protokolle aus Ihren Containern bequem an einem Ort anzuzeigen und zu verhindern, dass Ihre Containerprotokolle Speicherplatz auf Ihren Container-Instances belegen. Dieses Thema hilft Ihnen bei der Konfiguration des `awslogs` Protokolltreibers in Ihren Auftragsdefinitionen.

> **Note**
>
> In der AWS Batch Konsole können Sie den `awslogs` Protokolltreiber im Abschnitt Protokollierungskonfiguration konfigurieren, wenn Sie eine Auftragsdefinition erstellen.

> **Note**
>
> Die Art der Informationen, die von den Containern in Ihrem Auftrag protokolliert werden, hängt hauptsächlich von deren `ENTRYPOINT` Befehl ab. Standardmäßig zeigen die erfassten Protokolle die Befehlsausgabe an, die Sie normalerweise in einem interaktiven Terminal sehen, wenn Sie den Container lokal ausgeführt haben. Dabei handelt es sich um die `STDERR` E/A-Streams `STDOUT` und . Der `awslogs` Protokolltreiber übergibt diese Protokolle einfach von Docker an CloudWatch Logs. Weitere Informationen dazu, wie Docker-Protokolle

verarbeitet werden, einschließlich alternativer Möglichkeiten zum Erfassen unterschiedlicher Dateidaten oder Streams, finden Sie unter Anzeigen von Protokollen für einen Container oder Service in der Docker-Dokumentation.

Informationen zum Senden von Systemprotokollen von Ihren Container-Instances an - CloudWatch Protokolle finden Sie unter CloudWatch Logs verwenden mit AWS Batch. Weitere Informationen zu - CloudWatch Protokollen finden Sie unter Überwachen von -Protokolldateien und CloudWatch - Protokollkontingenten im Amazon- CloudWatch Logs-Benutzerhandbuch.

Verfügbare awslogs-Protokolltreiberoptionen

Der `awslogs` Protokolltreiber unterstützt die folgenden Optionen in AWS Batch Auftragsdefinitionen. Weitere Informationen finden Sie unter CloudWatch Protokolltreiber in der Docker-Dokumentation.

`awslogs-region`

Erforderlich: Nein

Geben Sie die Region an, in die der `awslogs` Protokolltreiber Ihre Docker-Protokolle senden soll. Standardmäßig ist die verwendete Region dieselbe wie die für den Auftrag. Sie können alle Ihre Protokolle von Aufträgen in verschiedenen Regionen an eine einzelne Region in - CloudWatch Protokollen senden. Auf diese Weise können sie alle von einem Ort aus sichtbar sein. Alternativ können Sie sie nach Region trennen, um einen detaillierteren Ansatz zu erhalten. Wenn Sie diese Option wählen, stellen Sie jedoch sicher, dass die angegebenen Protokollgruppen in der von Ihnen angegebenen Region vorhanden sind.

`awslogs-group`

Erforderlich: Optional

Mit der `awslogs-group` Option können Sie die Protokollgruppe angeben, an die der `awslogs` Protokolltreiber seine Protokollstreams sendet. Wenn dies nicht angegeben ist, `aws/batch/job` wird verwendet.

`awslogs-stream-prefix`

Erforderlich: Optional

Mit der `awslogs-stream-prefix` Option können Sie einen Protokollstream mit dem angegebenen Präfix und der Amazon-ECS-Aufgaben-ID des AWS Batch Auftrags verknüpfen, zu

dem der Container gehört. Wenn Sie einen Präfix mit dieser Option angeben, weist der Protokoll-Stream das folgende Format auf:

```
prefix-name/default/ecs-task-id
```

`awslogs-datetime-format`

Erforderlich: Nein

Diese Option definiert einen mehrzeiliges Startmuster im `strftime`-Format von Python. Eine Protokollnachricht besteht aus einer Zeile, die dem Muster entspricht, und allen folgenden Zeilen, die nicht dem Muster entsprechen. Damit ist die zugeordnete Zeile das Trennzeichen zwischen Protokollnachrichten.

Ein Beispiel für einen Anwendungsfall für die Nutzung dieses Formats ist die Ausgabenanalyse, z. B. einen Stack-Dump, der andernfalls möglicherweise mehrere Einträge protokolliert. Mit dem richtigen Muster kann es in nur einem Eintrag erfasst werden.

Weitere Informationen finden Sie unter awslogs-datetime-format.

Diese Option hat immer Vorrang, wenn sowohl `awslogs-datetime-format` als auch `awslogs-multiline-pattern` konfiguriert sind.

> **Note**
>
> Bei der mehrzeiligen Protokollierung wird eine regelmäßige Ausdrucksanalyse und die Übereinstimmung aller Protokollmeldungen ausgeführt. Dies kann negative Auswirkungen auf die Leistung der Protokollierung haben.

`awslogs-multiline-pattern`

Erforderlich: Nein

Diese Option definiert ein mehrzeiliges Startmuster mit einem regulären Ausdruck. Eine Protokollnachricht besteht aus einer Zeile, die dem Muster entspricht, und allen folgenden Zeilen, die nicht dem Muster entsprechen. Daher ist die übereinstimmende Zeile das Trennzeichen zwischen Protokollnachrichten.

Weitere Informationen finden Sie unter awslogs-multiline-pattern in der Docker-Dokumentation.

Diese Option wird ignoriert, wenn `awslogs-datetime-format` ebenfalls konfiguriert ist.

> **Note**
>
> Bei der mehrzeiligen Protokollierung wird eine regelmäßige Ausdrucksanalyse und die Übereinstimmung aller Protokollmeldungen ausgeführt. Dies kann negative Auswirkungen auf die Leistung der Protokollierung haben.

`awslogs-create-group`

Erforderlich: Nein

Geben Sie an, ob die Protokollgruppe automatisch erstellt werden soll. Wenn diese Option nicht angegeben ist, gilt standardmäßig `false`.

> **Warning**
>
> Diese Option wird nicht empfohlen. Wir empfehlen Ihnen, die Protokollgruppe im Voraus mit der - CloudWatch Protokoll-CreateLogGroupAPI-Aktion zu erstellen, da jeder Auftrag versucht, die Protokollgruppe zu erstellen, wodurch die Wahrscheinlichkeit erhöht wird, dass der Auftrag fehlschlägt.

> **Note**
>
> Die IAM-Richtlinie für Ihre Ausführungsrolle muss die `-logs:CreateLogGroup`Berechtigung enthalten, bevor Sie versuchen, zu verwenden`awslogs-create-group`.

Angeben einer Protokollkonfiguration in Ihrer Auftragsdefinition

Standardmäßig AWS Batch aktiviert den `awslogs` Protokolltreiber. In diesem Abschnitt wird beschrieben, wie Sie die `awslogs` Protokollkonfiguration für einen Auftrag anpassen. Weitere Informationen finden Sie unter Erstellen einer Auftragsdefinition mit einem Knoten .

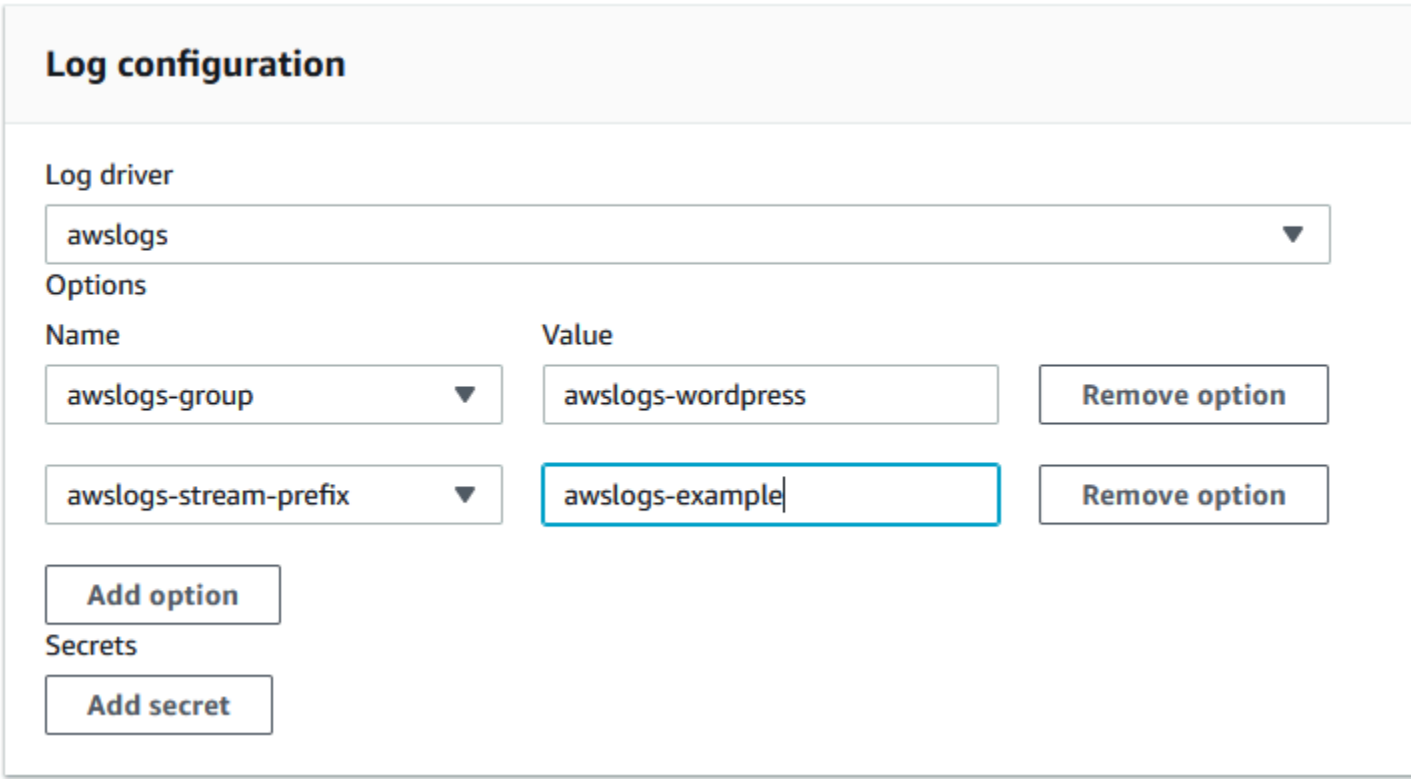
In den folgenden JSON-Snippets für die Protokollkonfiguration ist für jeden Auftrag ein `-logConfiguration`Objekt angegeben. Eine bezieht sich auf einen WordPress Auftrag, der

Protokolle an eine Protokollgruppe namens sendet`awslogs-wordpress`, und eine andere auf einen MySQL-Container, der Protokolle an eine Protokollgruppe namens sendet`awslogs-mysql`. Beide Container verwenden den Protokoll-Stream-Präfix `awslogs-example`.

```
"logConfiguration": {
    "logDriver": "awslogs",
    "options": {
        "awslogs-group": "awslogs-wordpress",
        "awslogs-stream-prefix": "awslogs-example"
    }
}
```

```
"logConfiguration": {
    "logDriver": "awslogs",
    "options": {
        "awslogs-group": "awslogs-mysql",
        "awslogs-stream-prefix": "awslogs-example"
    }
}
```

In der AWS Batch Konsole wird die Protokollkonfiguration für die `wordpress` Auftragsdefinition wie in der folgenden Abbildung dargestellt angegeben.



Nachdem Sie eine Aufgabendefinition mit dem `awslogs` Protokolltreiber in einer Protokollkonfiguration der Auftragsdefinition registriert haben, können Sie einen Auftrag mit dieser Auftragsdefinition senden, um mit dem Senden von Protokollen an CloudWatch Logs zu beginnen. Weitere Informationen finden Sie unter Einen Job einreichen.

Angabe sensibler Daten

Mit können Sie sensible Daten in Ihre Aufträge einfügenAWS Batch, indem Sie Ihre sensiblen Daten entweder in AWS Secrets Manager Secrets oder AWS Systems Manager Parameter Store-Parametern speichern und dann in Ihrer Auftragsdefinition auf sie verweisen.

Geheimnisse können einem Auftrag auf folgende Weise offengelegt werden:

- Um sensible Daten als Umgebungsvariablen in Ihre Container einzufügen, verwenden Sie den `-secrets`Auftragsdefinitionsparameter.
- Um auf vertrauliche Informationen in der Protokollkonfiguration eines Auftrags zu verweisen, verwenden Sie den `secretOptions` Auftragsdefinitionsparameter.

Themen

- Angeben vertraulicher Daten mithilfe von Secrets Manager
- Angeben sensibler Daten mithilfe des Systems Manager-Parameter Stores

Angeben vertraulicher Daten mithilfe von Secrets Manager

Mit können Sie sensible Daten in Ihre Aufträge einfügen AWS Batch, indem Sie Ihre sensiblen Daten in AWS Secrets Manager Secrets speichern und dann in Ihrer Auftragsdefinition auf sie verweisen. Sensible Daten, die in Secrets-Manager-Geheimnissen gespeichert sind, können einem Auftrag als Umgebungsvariablen oder als Teil der Protokollkonfiguration zur Verfügung gestellt werden.

Wenn Sie ein Secret als Umgebungsvariable einfügen, können Sie einen JSON-Schlüssel oder eine Version eines einzufügenden Secrets angeben. Dieser Prozess hilft Ihnen, die sensiblen Daten zu kontrollieren, die Ihrem Auftrag ausgesetzt sind. Weitere Informationen zur geheimen Versionsverwaltung finden Sie unter Schlüsselbegriffe und -konzepte für AWS Secrets Manager im AWS Secrets Manager -Benutzerhandbuch.

Überlegungen zum Angeben sensibler Daten mit Secrets Manager

Folgendes sollte berücksichtigt werden, wenn Secrets Manager verwendet wird, um sensible Daten für Aufträge anzugeben.

- Um ein Secret mit einem bestimmten JSON-Schlüssel oder einer Version eines Secrets einzufügen, muss auf der Container-Instance in Ihrer Computing-Umgebung Version 1.37.0 oder höher des Amazon-ECS-Container-Agenten installiert sein. Wir empfehlen jedoch, die neueste Container-Agent-Version zu verwenden. Informationen zum Überprüfen Ihrer Agentenversion und zum Aktualisieren auf die neueste Version finden Sie unter Aktualisieren des Amazon-ECS-Container-Agenten im Entwicklerhandbuch für Amazon Elastic Container Service.

  Um den vollständigen Inhalt eines Secrets als Umgebungsvariable einzufügen oder ein Secret in eine Protokollkonfiguration einzufügen, muss Ihre Container-Instance Version 1.23.0 oder höher des Container-Agenten haben.
- Nur Secrets, die Textdaten speichern, bei denen es sich um Secrets handelt, die mit dem CreateSecret-`SecretString`Parameter der API erstellt wurden, werden unterstützt. Secrets, die Binärdaten speichern, bei denen es sich um Secrets handelt, die mit dem CreateSecret-`SecretBinary`Parameter der API erstellt wurden, werden nicht unterstützt.
- Wenn Sie eine Auftragsdefinition verwenden, die auf Secrets-Manager-Secrets verweist, um sensible Daten für Ihre Aufträge abzurufen, müssen Sie bei Verwendung von Schnittstellen-VPC-Endpunkten die Schnittstellen-VPC-Endpunkte für Secrets Manager erstellen. Weitere Informationen finden Sie unter Verwenden von Secrets Manager mit VPC-Endpunkten im AWS Secrets Manager -Benutzerhandbuch.
- Sensible Daten werden in Ihren Auftrag eingefügt, wenn der Auftrag zum ersten Mal gestartet wird. Wenn das Secret anschließend aktualisiert oder rotiert wird, erhält der Auftrag nicht automatisch den aktualisierten Wert. Sie müssen einen neuen Auftrag starten, um den Service zu zwingen, einen neuen Auftrag mit dem aktualisierten Secret-Wert zu starten.

Erforderliche IAM-Berechtigungen für AWS Batch Secrets

Um dieses Feature verwenden zu können, müssen Sie über die Ausführungsrolle verfügen und in Ihrer Auftragsdefinition darauf verweisen. Dies ermöglicht dem Container-Agent das Abrufen der erforderlichen Secrets Manager-Ressourcen. Weitere Informationen finden Sie unter AWS Batch Ausführung, IAM-Rolle.

Um Zugriff auf die von Ihnen erstellten Secrets Manager-Secrets zu gewähren, fügen Sie der Ausführungsrolle manuell die folgenden Berechtigungen als Inline-Richtlinie hinzu. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Richtlinien im IAM-Benutzerhandbuch.

- `secretsmanager:GetSecretValue`: Erforderlich, wenn Sie auf ein Secrets Manager-Secret verweisen.
- `kms:Decrypt`: Nur erforderlich, wenn Ihr Secret einen benutzerdefinierten KMS-Schlüssel verwendet und nicht den Standardschlüssel. Der ARN für Ihren benutzerdefinierten Schlüssel sollte als Ressource hinzugefügt werden.

Das folgende Beispiel einer Inline-Richtlinie fügt die erforderlichen Berechtigungen hinzu:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:<secret_name>",
        "arn:aws:kms:<region>:<aws_account_id>:key/<key_id>"
      ]
    }
  ]
}
```

Injizieren sensibler Daten als Umgebungsvariable

In Ihrer Auftragsdefinition können Sie die folgenden Elemente angeben:

- Das `secrets` Objekt, das den Namen der Umgebungsvariablen enthält, die im Auftrag festgelegt werden soll
- Der Amazon-Ressourcenname (ARN) des Secrets Manager-Secrets
- Zusätzliche Parameter, die die sensiblen Daten enthalten, die dem Auftrag präsentiert werden sollen

Das folgende Beispiel zeigt die vollständige Syntax, die für das Secrets Manager-Secret angegeben werden muss.

```
arn:aws:secretsmanager:region:aws_account_id:secret:secret-name:json-key:version-stage:version-id
```

Im folgenden Abschnitt werden die zusätzlichen Parameter beschrieben. Diese Parameter sind optional. Wenn Sie sie jedoch nicht verwenden, müssen Sie die Doppelpunkte einschließen, `:` um die Standardwerte zu verwenden. Beispiele finden Sie unten für weiteren Kontext.

`json-key`

Gibt den Namen des Schlüssels in einem Schlüssel-Wert-Paar mit dem Wert an, den Sie als Umgebungsvariablenwert festlegen möchten. Nur Werte im JSON-Format werden unterstützt. Wenn Sie keinen JSON-Schlüssel angeben, wird der vollständige Inhalt des Secrets verwendet.

`version-stage`

Gibt die Phasenbeschriftung der Version eines Secrets an, die Sie verwenden möchten. Wenn eine Versionsphasenbeschriftung angegeben ist, können Sie keine Versions-ID angeben. Wenn keine Versionsphase angegeben wird, besteht das Standardverhalten darin, das Secret Schlüssel mit der `AWSCURRENT`-Phasenbeschriftung abzurufen.

Phasenbeschriftungen werden verwendet, um verschiedene Versionen eines Secrets zu verfolgen, wenn sie aktualisiert oder rotiert werden. Jede Version eines Secrets hat eine oder mehrere Phasenbeschriftungen und eine ID. Weitere Informationen finden Sie unter Wichtige Begriffe und Konzepte für AWS Secrets Manager im AWS Secrets Manager -Benutzerhandbuch.

`version-id`

Gibt die eindeutige ID der Version des Secrets an, die Sie verwenden möchten. Wenn eine Versions-ID angegeben wird, können Sie keine Versionsphasenbeschriftung angeben. Wenn keine Versions-ID angegeben wird, besteht das Standardverhalten darin, den geheimen Schlüssel mit der `AWSCURRENT`-Phasenbeschriftung abzurufen.

Versions-IDs werden verwendet, um verschiedene Versionen eines Secrets zu verfolgen, wenn sie aktualisiert oder rotiert werden. Jede Version eines Secrets hat eine ID. Weitere Informationen finden Sie unter Wichtige Begriffe und Konzepte für AWS Secrets Manager im AWS Secrets Manager -Benutzerhandbuch.

Beispiel-Containerdefinitionen

Die folgenden Beispiele zeigen, wie Sie auf Secrets Manager-Secrets in Ihren Containerdefinitionen verweisen können.

Example Verweisen auf ein vollständiges Secret

Im Folgenden finden Sie einen Ausschnitt einer Aufgabendefinition mit dem Format beim Verweisen auf den vollständigen Text eines Secrets Manager-Secret.

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
    }]
  }]
}
```

Example Verweisen auf einen bestimmten Schlüssel innerhalb eines Secrets

Im Folgenden wird eine Beispielausgabe eines get-secret-value Befehls gezeigt, der den Inhalt eines Secrets zusammen mit der Versions-Staging-Bezeichnung und der damit verknüpften Versions-ID anzeigt.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "VersionId": "871d9eca-18aa-46a9-8785-981dd39ab30c",
    "SecretString": "{\"username1\":\"password1\",\"username2\":\"password2\",
\"username3\":\"password3\"}",
    "VersionStages": [
        "AWSCURRENT"
    ],
    "CreatedDate": 1581968848.921
}
```

Verweisen Sie auf einen bestimmten Schlüssel aus der vorherigen Ausgabe in einer Containerdefinition, indem Sie den Schlüsselnamen am Ende des ARN angeben.

```
{
```

```
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::"
    }]
  }]
}
```

Example Verweisen auf eine bestimmte Secret-Version

Im Folgenden wird eine Beispielausgabe eines describe-secret -Befehls gezeigt, der den unverschlüsselten Inhalt eines Secrets zusammen mit den Metadaten für alle Versionen des Secrets anzeigt.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "Description": "Example of a secret containing application authorization data.",
    "RotationEnabled": false,
    "LastChangedDate": 1581968848.926,
    "LastAccessedDate": 1581897600.0,
    "Tags": [],
    "VersionIdsToStages": {
        "871d9eca-18aa-46a9-8785-981dd39ab30c": [
            "AWSCURRENT"
        ],
        "9d4cb84b-ad69-40c0-a0ab-cead36b967e8": [
            "AWSPREVIOUS"
        ]
    }
}
```

Verweisen Sie auf eine bestimmte Versionsphasenbeschriftung aus der vorherigen Ausgabe in einer Containerdefinition, indem Sie den Schlüsselnamen am Ende des ARN angeben.

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf::AWSPREVIOUS:"
```

```
    }]
  }]
}
```

Verweisen Sie auf eine bestimmte Versions-ID der vorherigen Ausgabe in einer Containerdefinition, indem Sie den Schlüsselnamen am Ende des ARN angeben.

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf::9d4cb84b-ad69-40c0-a0ab-cead36b967e8"
    }]
  }]
}
```

Example Verweisen auf einen bestimmten Schlüssel und eine Versionsphasenbeschriftung eines Secrets

Im Folgenden wird gezeigt, wie Sie sowohl auf einen bestimmten Schlüssel innerhalb eines Secrets als auch auf eine bestimmte Versionsphasenbeschriftung verweisen.

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf:username1:AWSPREVIOUS:"
    }]
  }]
}
```

Verwenden Sie die folgende Syntax, um einen bestimmten Schlüssel und eine Versions-ID anzugeben.

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf:username1::9d4cb84b-ad69-40c0-a0ab-cead36b967e8"
```

```
    }]
  }]
}
```

Injizieren sensibler Daten in einer Protokollkonfiguration

Wenn Sie in Ihrer Auftragsdefinition angeben, können `logConfiguration` Sie `secretOptions` mit dem Namen der im Container festzulegenden Protokolltreiberoption und dem vollständigen ARN des Secrets-Manager-Secrets angeben, das die sensiblen Daten enthält, die dem Container präsentiert werden sollen.

Im Folgenden finden Sie einen Ausschnitt einer Auftragsdefinition, der das Format zeigt, wenn auf ein Secrets-Manager-Secret verwiesen wird.

```
{
  "containerProperties": [{
    "logConfiguration": [{
      "logDriver": "splunk",
      "options": {
        "splunk-url": "https://cloud.splunk.com:8080"
      },
      "secretOptions": [{
        "name": "splunk-token",
        "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
      }]
    }]
  }]
}
```

Erstellen eines - AWS Secrets Manager Secrets

Sie können die Secrets Manager-Konsole verwenden, um ein Secret für Ihre sensiblen Daten zu erstellen. Weitere Informationen finden Sie unter Erstellen eines Basic Secrets im AWS Secrets Manager -Benutzerhandbuch.

So erstellen Sie ein Basis-Secret

Verwenden Sie Secrets Manager zum Erstellen eines Secrets für Ihre sensiblen Daten.

1. Öffnen Sie die Secrets-Manager-Konsole unterhttps://console.aws.amazon.com/secretsmanager/.

2. Wählen Sie Store a new secret (Ein neues Secret speichern).
3. Wählen Sie für Select secret type (Secret-Typ auswählen) die Option Other type of secrets (Anderer Secret-Typ) aus.
4. Geben Sie die Daten Ihres benutzerdefinierten Secrets in Form von Paaren aus Key (Schlüssel) und Value (Wert) an. Sie können z. B. einen Schlüssel von `UserName` festlegen und dann den entsprechenden Benutzernamen als Wert angeben. Fügen Sie einen zweiten Schlüssel mit dem Namen `Password` und dem Passworttext als Wert hinzu. Sie können auch Einträge für einen Datenbanknamen, eine Serveradresse oder einen TCP-Port hinzufügen. Sie können so viele Paare wie nötig zum Speichern der von Ihnen benötigten Informationen hinzufügen.

   Alternativ können Sie die Registerkarte Plaintext (Klartext) wählen und den Secret-Wert auf beliebige Weise eingeben.
5. Wählen Sie den AWS KMS Verschlüsselungsschlüssel aus, mit dem Sie den geschützten Text im Secret verschlüsseln möchten. Wenn Sie keinen Schlüssel wählen, prüft Secrets Manager, ob es einen Standardschlüssel für das Konto gibt. Fall dies der Fall ist, wird er verwendet. Wenn kein Standardschlüssel vorhanden ist, erstellt Secrets Manager automatisch einen Schlüssel. Sie können auch Add new key (Neuen Schlüssel hinzufügen) auswählen, um einen benutzerdefinierten KMS-Schlüssel speziell für dieses Secret zu erstellen. Um Ihren eigenen KMS-Schlüssel zu erstellen, müssen Sie in Ihrem Konto zum Erstellen von KMS-Schlüssel berechtigt sein.
6. Wählen Sie Weiter aus.
7. Geben Sie für Secret name (Secret-Name) einen optionalen Pfad und Namen ein, z. B. **`production/MyAwesomeAppSecret`** oder **`development/TestSecret`**, und wählen Sie Next (Weiter) aus. Optional können Sie eine Beschreibung hinzufügen. Mit dieser können Sie sich später besser an den Zweck dieses Secrets erinnern.

   Der Secret-Name darf nur ASCII-Zeichen, Ziffern oder eines der folgenden Zeichen enthalten: /_ **+=**.@-
8. (Optional) Jetzt können Sie die Rotation für Ihr Secret konfigurieren. Belassen Sie die Option für dieses Verfahren auf Disable automatic rotation (Automatisches Rotieren deaktivieren) und klicken Sie auf Next (Weiter).

   Informationen zum Konfigurieren der Rotation für neue oder vorhandene Secrets finden Sie unter Rotieren Ihrer AWS Secrets Manager Secrets.
9. Überprüfen Sie die Einstellungen, und wählen Sie dann Store secret (Secret speichern) aus, um Ihre gesamte Eingabe als neues Secret im Secrets Manager zu speichern.

Angeben sensibler Daten mithilfe des Systems Manager-Parameter Stores

Mit können Sie sensible Daten in Ihre Container einfügen AWS Batch, indem Sie Ihre sensiblen Daten in AWS Systems Manager Parameter Store-Parametern speichern und dann in Ihrer Containerdefinition auf sie verweisen.

Themen

Überlegungen zum Angeben sensibler Daten mithilfe des Systems Manager-Parameter Stores

Bei der Angabe sensibler Daten für Container mit Systems Manager-Parameter Store-Parametern sollte Folgendes berücksichtigt werden.

- Diese Funktion erfordert, dass Ihre Container-Instance Version 1.23.0 oder höher des Container-Agenten hat. Wir empfehlen jedoch, die neueste Container-Agent-Version zu verwenden. Informationen zum Überprüfen Ihrer Agentenversion und zum Aktualisieren auf die neueste Version finden Sie unter Aktualisieren des Amazon-ECS-Container-Agenten im Entwicklerhandbuch für Amazon Elastic Container Service.
- Sensible Daten werden in den Container für Ihren Auftrag eingefügt, wenn der Container zum ersten Mal gestartet wird. Wenn der geheime oder der Parameter Store-Parameter aktualisiert oder rotiert wird, erhält der Container nicht automatisch den aktualisierten Wert. Sie müssen einen neuen Auftrag starten, um den Start eines neuen Auftrags mit aktualisierten Secrets zu erzwingen.

Erforderliche IAM-Berechtigungen für AWS Batch Secrets

Um dieses Feature verwenden zu können, müssen Sie über die Ausführungsrolle verfügen und in Ihrer Auftragsdefinition darauf verweisen. Auf diese Weise kann der Amazon-ECS-Container-Agent die erforderlichen AWS Systems Manager Ressourcen abrufen. Weitere Informationen finden Sie unter AWS Batch Ausführung, IAM-Rolle.

Um Zugriff auf die von Ihnen erstellten AWS Systems Manager Parameter Store-Parameter zu gewähren, fügen Sie der Ausführungsrolle manuell die folgenden Berechtigungen als Inline-Richtlinie hinzu. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Richtlinien im IAM-Benutzerhandbuch.

- `ssm:GetParameters`: Erforderlich, wenn in einer Aufgabendefinition auf einen Parameter Store-Parameter von Systems Manager verwiesen wird.
- `secretsmanager:GetSecretValue`: Erforderlich, wenn Sie direkt auf ein Secrets Manager-Secret verweisen oder wenn der Parameter Systems Manager Parameterspeicher in einer Aufgabendefinition auf ein Secrets Manager-Secret verweist.
- `kms:Decrypt` – Nur erforderlich, wenn Ihr Geheimnis einen benutzerdefinierten KMS-Schlüssel verwendet und nicht den Standardschlüssel. Der ARN für Ihren benutzerdefinierten Schlüssel sollte als Ressource hinzugefügt werden.

Das folgende Beispiel einer Inline-Richtlinie fügt die erforderlichen Berechtigungen hinzu:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetParameters",
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:ssm:<region>:<aws_account_id>:parameter/<parameter_name>",
        "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:<secret_name>",
        "arn:aws:kms:<region>:<aws_account_id>:key/<key_id>"
      ]
    }
  ]
}
```

Injizieren sensibler Daten als Umgebungsvariable

Geben Sie in Ihrer Containerdefinition `secrets` mit dem Namen der im Container zu setzenden Umgebungsvariablen und dem Namen oder dem ARN des Systems Manager-Parameter Store-Parameters an, der die sensiblen Daten enthält, die dem Container präsentiert werden sollen.

Im Folgenden finden Sie einen Ausschnitt einer Aufgabendefinition, die das Format zeigt, wenn auf einen Systems Manager Parameter Store-Parameter verwiesen wird. Wenn sich der Systems Manager Parameter Store-Parameter in derselben Region wie die Aufgabe befindet, die Sie starten, können Sie entweder den vollständigen ARN oder den Namen des Parameters verwenden. Wenn der Parameter in einer anderen Region existiert, muss der volle ARN angegeben werden.

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }]
  }]
}
```

Injizieren sensibler Daten in einer Protokollkonfiguration

Beim Angeben von `logConfiguration` können Sie `secretOptions` in Ihrer Containerdefinition mit dem Namen der im Container festzulegenden Protokolltreiberoption und dem vollständigen ARN des Systems Manager-Parameter Store-Parameters angeben, in denen die sensiblen Daten enthalten sind, die dem Container präsentiert werden sollen.

> **Important**
>
> Wenn sich der Systems Manager Parameter Store-Parameter in derselben Region wie die Aufgabe befindet, die Sie starten, können Sie entweder den vollständigen ARN oder den Namen des Parameters verwenden. Wenn der Parameter in einer anderen Region existiert, muss der volle ARN angegeben werden.

Im Folgenden finden Sie einen Ausschnitt einer Aufgabendefinition, die das Format zeigt, wenn auf einen Systems Manager Parameter Store-Parameter verwiesen wird.

```
{
```

```
  "containerProperties": [{
    "logConfiguration": [{
      "logDriver": "fluentd",
      "options": {
        "tag": "fluentd demo"
      },
      "secretOptions": [{
        "name": "fluentd-address",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
      }]
    }]
  }]
}
```

Erstellen eines AWS Systems Manager Parameter Store-Parameters

Sie können die AWS Systems Manager Konsole verwenden, um einen Systems Manager Parameter Store-Parameter für Ihre sensiblen Daten zu erstellen. Weitere Informationen finden Sie unter Walkthrough: Erstellen und Verwenden eines Parameter in einem Befehl (Konsole) im AWS Systems Manager -Benutzerhandbuch.

So erstellen Sie einen Parameter Store-Parameter

1. Öffnen Sie die - AWS Systems Manager Konsole unter https://console.aws.amazon.com/systems-manager/.
2. Wählen Sie im Navigationsbereich Parameter Store (Parameterspeicher) Create parameter (Parameter erstellen).
3. Geben Sie unter Name eine Hierarchie und einen Parameternamen ein. Geben Sie beispielsweise `test/database_password` ein.
4. Geben Sie unter Description (Beschreibung) eine optionale Beschreibung ein.
5. Wählen Sie für Typ die Option Zeichenfolge , StringListoder ausSecureString.

> **Note**
>
> - Wenn Sie auswählenSecureString, wird das Feld KMS-Schlüssel-ID angezeigt. Wenn Sie keine KMS Schlüssel-ID, keinen KMS Schlüssel-ARN, keinen Aliasnamen oder Alias-ARN angeben, verwendet das System `alias/aws/ssm`. Das ist der Standard-KMS-Schlüssel für Systems Manager. Wenn Sie diesen Schlüssel nicht verwenden möchten, können Sie einen benutzerdefinierten Schlüssel auswählen. Weitere

Informationen finden Sie unter Verwenden von sicheren String-Parametern im AWS Systems Manager -Benutzerhandbuch.

- Wenn Sie einen sicheren Stringparameter in der Konsole mit dem Parameter `key-id` entweder mit einem benutzerdefinierten KMS-Schlüssel-Aliasnamen oder einem Alias-ARN erstellen, müssen Sie das Präfix `alias/` vor dem Alias angeben. Nachfolgend ein ARN-Beispiel:

  ```
  arn:aws:kms:us-east-2:123456789012:alias/MyAliasName
  ```

  Im Folgenden finden Sie ein Beispiel für einen Aliasnamen:

  ```
  alias/MyAliasName
  ```

6. Geben Sie für Value (Wert) einen Wert ein. Beispiel: `MyFirstParameter` Wenn Sie ausgewählt habenSecureString, wird der Wert genau so maskiert, wie Sie ihn eingegeben haben.
7. Wählen Sie Create parameter (Parameter erstellen) aus.

Private Registrierungsauthentifizierung für -Aufträge

Mit der privaten Registrierungsauthentifizierung für Aufträge mit AWS Secrets Manager können Sie Ihre Anmeldeinformationen sicher speichern und dann in Ihrer Auftragsdefinition auf sie verweisen. Auf diese Weise können Sie auf Container-Images verweisen, die in privaten Registrierungen außerhalb von vorhanden sind AWS und in Ihren Auftragsdefinitionen eine Authentifizierung erfordern. Diese Funktion wird von Aufträgen unterstützt, die auf Amazon EC2-Instances und Fargate gehostet werden.

Important

Wenn Ihre Auftragsdefinition auf ein Image verweist, das in Amazon ECR gespeichert ist, gilt dieses Thema nicht. Weitere Informationen finden Sie unter Amazon ECR-Images mit Amazon ECS im Amazon Elastic Container-Registry-Benutzerhandbuch.

Für Aufträge, die auf Amazon EC2-Instances gehostet werden, erfordert diese Funktion Version `1.19.0` oder höher des Container-Agenten. Wir empfehlen jedoch, die neueste Container-Agent-Version zu verwenden. Informationen zum Überprüfen Ihrer Agentenversion und zum Aktualisieren

auf die neueste Version finden Sie unter Aktualisieren des Amazon-ECS-Container-Agenten im Entwicklerhandbuch für Amazon Elastic Container Service.

Für Aufträge, die auf Fargate gehostet werden, erfordert diese Funktion die Plattformversion `1.2.0` oder höher. Weitere Informationen finden Sie unter AWS Fargate-Linux-Plattformversionen im Amazon Elastic Container Service-Entwicklerhandbuch.

Geben Sie in Ihrer Containerdefinition das Objekt `repositoryCredentials` mit den Details des von Ihnen erstellten Geheimnisses an. Das Secret, auf das Sie verweisen, kann von einem anderen AWS-Region oder einem anderen Konto stammen als der Auftrag, der es verwendet.

> **Note**
>
> Wenn Sie die AWS Batch -API AWS CLIoder das AWS -SDK verwenden und sich das Secret in derselben AWS-Region wie der Auftrag befindet, den Sie starten, können Sie entweder den vollständigen ARN oder den Namen des Secrets verwenden. Wenn das Geheimnis in einem anderen Konto vorhanden ist, muss der vollständige ARN des Geheimnisses angegeben werden. Bei Verwendung der muss AWS Management Consoleimmer der vollständige ARN des Secrets angegeben werden.

Im Folgenden finden Sie einen Ausschnitt einer Auftragsdefinition, der die erforderlichen Parameter anzeigt:

```
"containerProperties": [
  {
    "image": "private-repo/private-image",
    "repositoryCredentials": {
      "credentialsParameter":
 "arn:aws:secretsmanager:region:123456789012:secret:secret_name"
    }
  }
]
```

Erforderliche IAM-Berechtigungen für die private Registrierungsauthentifizierung

Die Ausführungsrolle ist erforderlich, um diese Funktion verwenden zu können. Auf diese Weise kann der Container-Agent das Container-Image abrufen. Weitere Informationen finden Sie unter AWS Batch Ausführung, IAM-Rolle.

Um Zugriff auf die von Ihnen erstellten Secrets zu gewähren, fügen Sie der Ausführungsrolle die folgenden Berechtigungen als Inline-Richtlinie hinzu. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Richtlinien.

- `secretsmanager:GetSecretValue`
- `kms:Decrypt`: Nur erforderlich, wenn Ihr Schlüssel einen benutzerdefinierten KMS-Schlüssel verwendet und nicht den Standard-Schlüssel. Der Amazon-Ressourcenname (ARN) für Ihren benutzerdefinierten Schlüssel muss als Ressource hinzugefügt werden.

Das folgende Beispiel einer Inline-Richtlinie fügt die Berechtigungen hinzu:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "arn:aws:secretsmanager:region:123456789012:secret:secret_name",
        "arn:aws:kms:region:123456789012:key/key_id"
      ]
    }
  ]
}
```

Verwenden einer privaten Registrierungsauthentifizierung

So erstellen Sie ein Basis-Secret

Verwenden Sie AWS Secrets Manager , um ein Secret für Ihre privaten Registrierungsanmeldeinformationen zu erstellen.

1. Öffnen Sie die - AWS Secrets Manager Konsole unter https://console.aws.amazon.com/secretsmanager/.
2. Wählen Sie Store a new secret (Ein neues Secret speichern).
3. Wählen Sie für Select secret type (Secret-Typ auswählen) die Option Other type of secrets (Anderer Secret-Typ) aus.
4. Wählen Sie Plaintext (Klartext) aus und geben Sie Ihre privaten Registrierungsanmeldeinformationen in folgendem Format ein:

   ```
   {
     "username" : "privateRegistryUsername",
     "password" : "privateRegistryPassword"
   }
   ```

5. Wählen Sie Weiter aus.
6. Geben Sie für Secret name (Secret-Name) einen optionalen Pfad und Namen ein, z. B. **`production/MyAwesomeAppSecret`** oder **`development/TestSecret`**, und wählen Sie Next (Weiter) aus. Optional können Sie eine Beschreibung hinzufügen. Mit dieser können Sie sich später besser an den Zweck dieses Secrets erinnern.

   Der Secret-Name darf nur ASCII-Zeichen, Ziffern oder eines der folgenden Zeichen enthalten: `/_+=.@-`.
7. (Optional) Jetzt können Sie die Rotation für Ihr Secret konfigurieren. Belassen Sie die Option für dieses Verfahren auf Disable automatic rotation (Automatisches Rotieren deaktivieren) und klicken Sie auf Next (Weiter).

   Anweisungen zum Konfigurieren der Rotation für neue oder vorhandene Secrets finden Sie unter Rotieren Ihrer AWS Secrets Manager Secrets.
8. Überprüfen Sie die Einstellungen, und wählen Sie dann Store secret (Secret speichern) aus, um Ihre gesamte Eingabe als neues Secret im Secrets Manager zu speichern.

Registrieren Sie eine Auftragsdefinition und aktivieren Sie unter Private Registrierung die private Registrierungsauthentifizierung. Geben Sie dann unter Secrets-Manager-ARN oder -Name den Amazon-Ressourcennamen (ARN) des Secrets ein. Weitere Informationen finden Sie unter Erforderliche IAM-Berechtigungen für die private Registrierungsauthentifizierung.

Amazon EFS-Volumes

Amazon Elastic File System (Amazon EFS) bietet einfachen, skalierbaren Dateispeicher für die Verwendung mit Ihren AWS Batch Aufträgen. Mit Amazon EFS ist die Speicherkapazität elastisch. Es wird automatisch skaliert, wenn Sie Dateien hinzufügen und entfernen. Ihre Anwendungen verfügen jederzeit über den jeweils erforderlichen Speicherplatz.

Sie können Amazon-EFS-Dateisysteme mit verwendenAWS Batch, um Dateisystemdaten aus Ihrer gesamten Flotte von Container-Instances zu exportieren. Auf diese Weise haben Ihre Aufträge Zugriff auf denselben persistenten Speicher. Sie müssen Ihr Container-Instance-AMI jedoch konfigurieren, um das Amazon EFS-Dateisystem zu mounten, bevor der Docker-Daemon startet. Außerdem müssen Ihre Auftragsdefinitionen auf Volume-Mounts auf der Container-Instance verweisen, um das Dateisystem verwenden zu können. Die folgenden Abschnitte helfen Ihnen bei den ersten Schritten mit Amazon EFS mit AWS Batch.

Überlegungen zu Amazon EFS Volumes

Bei der Verwendung von Amazon EFS-Volumes sollte Folgendes berücksichtigt werden:

- Bei Aufträgen, die EC2-Ressourcen verwenden, wurde die Unterstützung des Amazon-EFS-Dateisystems als öffentliche Vorschau mit Amazon-ECS-optimierter AMI-Version `20191212` mit Container-Agent-Version 1.35.0 hinzugefügt. Die Amazon-EFS-Dateisystemunterstützung hat jedoch mit der Amazon-ECS-optimierten AMI-Version `20200319` mit Container-Agent-Version 1.38.0, die den Amazon-EFS-Zugriffspunkt und die IAM-Autorisierungsfunktionen enthielt, die allgemeine Verfügbarkeit erreicht. Wir empfehlen Ihnen, Amazon-ECS-optimierte AMI-Version `20200319` oder höher zu verwenden, um diese Funktionen zu nutzen. Weitere Informationen finden Sie unter Amazon-ECS-optimierte AMI-Versionen im Amazon Elastic Container Service-Entwicklerhandbuch.

  > **Note**
  >
  > Wenn Sie Ihr eigenes AMI erstellen, müssen Sie Container-Agent 1.38.0 oder höher, `ecs-init` Version 1.38.0-1 oder höher verwenden und die folgenden Befehle auf Ihrer Amazon

EC2-Instance ausführen. Dies ist alles, um das Amazon-ECS-Volume-Plugin zu aktivieren. Die Befehle hängen davon ab, ob Sie Amazon Linux 2 oder Amazon Linux als Basis-Image verwenden.

Amazon Linux 2

```
$ yum install amazon-efs-utils
systemctl enable --now amazon-ecs-volume-plugin
```

Amazon Linux

```
$ yum install amazon-efs-utils
sudo shutdown -r now
```

- Bei Aufträgen, die Fargate-Ressourcen verwenden, wurde die Amazon-EFS-Dateisystemunterstützung bei Verwendung der Plattformversion 1.4.0 oder höher hinzugefügt. Weitere Informationen finden Sie unter AWS Fargate-Plattformversionen im Amazon Elastic Container Service-Entwicklerhandbuch.
- Wenn Sie Amazon-EFS-Volumes in Aufträgen mithilfe von Fargate-Ressourcen angeben, erstellt Fargate einen Supervisor-Container, der für die Verwaltung des Amazon-EFS-Volumes verantwortlich ist. Der Supervisor-Container verwendet einen kleinen Teil des Arbeitsspeichers des Auftrags. Der Supervisor-Container ist beim Abfragen des Endpunkts der Aufgabenmetadaten Version 4 sichtbar. Weitere Informationen finden Sie unter Aufgabenmetadaten-Endpunkt Version 4 im Amazon Elastic Container Service-Benutzerhandbuch für AWS Fargate.

Verwenden von Amazon EFS Access Points

Amazon-EFS-Zugriffspunkte sind anwendungsspezifische Einstiegspunkte in ein EFS-Dateisystem, mit denen Sie den Anwendungszugriff auf freigegebene Datensätze verwalten können. Weitere Informationen zu Amazon EFS Access Points und zur Steuerung des Zugriffs auf diese finden Sie unter Arbeiten mit Amazon EFS Access Points im Amazon Elastic File System-Benutzerhandbuch.

Zugriffspunkte können eine Benutzeridentität, einschließlich der POSIX-Gruppen des Benutzers, für alle Dateisystemanforderungen erzwingen, die über den Zugriffspunkt erfolgen. Zugriffspunkte können auch ein anderes Stammverzeichnis für das Dateisystem erzwingen, so dass Clients nur auf Daten im angegebenen Verzeichnis oder in seinen Unterverzeichnissen zugreifen können.

> **Note**
>
> Beim Erstellen eines EFS-Zugriffspunkts geben Sie einen Pfad im Dateisystem an, der als Stammverzeichnis dienen soll. Wenn Sie in Ihrer AWS Batch Auftragsdefinition auf das EFS-Dateisystem mit einer Zugriffspunkt-ID verweisen, muss das Stammverzeichnis entweder weggelassen oder auf gesetzt werden. `/` Dies erzwingt den Pfad, der auf dem EFS-Zugriffspunkt festgelegt ist.

Sie können eine -AWS BatchAuftrags-IAM-Rolle verwenden, um zu erzwingen, dass bestimmte Anwendungen einen bestimmten Zugriffspunkt verwenden. Durch die Kombination von IAM-Richtlinien mit Zugriffspunkten können Sie ganz einfach einen sicheren Zugriff auf bestimmte Datasets für Ihre Anwendungen bereitstellen. Diese Funktion verwendet Amazon-ECS-IAM-Rollen für Aufgabenfunktionen. Weitere Informationen finden Sie unter IAM-Rollen für Aufgaben im Entwicklerhandbuch zum Amazon Elastic Container Service.

Angeben eines Amazon-EFS-Dateisystems in Ihrer Auftragsdefinition

Um Amazon-EFS-Dateisystem-Volumes für Ihre Container zu verwenden, müssen Sie die Volume- und Mountingpunktkonfigurationen in Ihrer Auftragsdefinition angeben. Das folgende JSON-Snippet der Auftragsdefinition zeigt die Syntax für die `mountPoints` Objekte `volumes` und für einen Container:

```
{
    "containerProperties": [
        {
            "image": "amazonlinux:2",
            "command": [
                "ls",
                "-la",
                "/mount/efs"
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEfsVolume",
                    "containerPath": "/mount/efs",
                    "readOnly": true
                }
            ],
            "volumes": [
```

```
                {
                    "name": "myEfsVolume",
                    "efsVolumeConfiguration": {
                        "fileSystemId": "fs-12345678",
                        "rootDirectory": "/path/to/my/data",
                        "transitEncryption": "ENABLED",
                        "transitEncryptionPort": integer,
                        "authorizationConfig": {
                            "accessPointId": "fsap-1234567890abcdef1",
                            "iam": "ENABLED"
                        }
                    }
                }
            ]
        }
    ]
}
```

`efsVolumeConfiguration`

Typ: Objekt

Erforderlich: Nein

Dieser Parameter wird nur bei der Verwendung von Amazon EFS-Volumes angegeben.

`fileSystemId`

Typ: Zeichenfolge

Erforderlich: Ja

Die zu verwendende Amazon EFS-Dateisystem-ID.

`rootDirectory`

Typ: Zeichenfolge

Erforderlich: Nein

Das Verzeichnis im Amazon EFS-Dateisystem, das als Stammverzeichnis im Host bereitgestellt werden soll. Wenn dieser Parameter weggelassen wird, wird der Stamm des Amazon-EFS-Volumes verwendet. Die Angabe von `/` hat denselben Effekt wie das Weglassen dieses Parameters. Er kann bis zu 4.096 Zeichen lang sein.

> **Important**
>
> Wenn ein EFS-Zugriffspunkt in der angegeben ist`authorizationConfig`, muss der Stammverzeichnisparameter entweder weggelassen oder auf festgelegt werden`/`. Dadurch wird der Pfad erzwungen, der auf dem EFS-Zugriffspunkt festgelegt ist.

`transitEncryption`

Typ: Zeichenfolge

Zulässige Werte: `ENABLED` | `DISABLED`

Erforderlich: Nein

Legt fest, ob die Verschlüsselung für Amazon-EFS-Daten aktiviert werden soll, die zwischen dem AWS Batch Host und dem Amazon-EFS-Server übertragen werden. Die Transit-Verschlüsselung muss aktiviert sein, wenn die Amazon-EFS-IAM-Autorisierung verwendet wird. Wenn dieser Parameter nicht angegeben ist, wird der Standardwert `DISABLED` verwendet. Weitere Informationen finden Sie unter Verschlüsseln von Daten während der Übertragung im Benutzerhandbuch für Amazon Elastic File System.

`transitEncryptionPort`

Typ: Ganzzahl

Erforderlich: Nein

Der Port, der beim Senden verschlüsselter Daten zwischen dem AWS Batch Host und dem Amazon-EFS-Server verwendet werden soll. Wenn Sie keinen Transit-Verschlüsselungsport angeben, wird die Port-Auswahlstrategie verwendet, die der Amazon EFS-Mount-Helfer verwendet. Dieser Wert muss zwischen 0 und 65.535 liegen. Weitere Informationen finden Sie unter EFS-Mount-Helfer im Benutzerhandbuch für Amazon Elastic File System.

`authorizationConfig`

Typ: Objekt

Erforderlich: Nein

Die Autorisierungskonfigurationsdetails für das Amazon EFS-Dateisystem.

`accessPointId`

Typ: Zeichenfolge

Erforderlich: Nein

Die zu verwendende Zugriffspunkt-ID. Wenn ein Zugriffspunkt angegeben ist, `efsVolumeConfiguration` muss der Wert des Stammverzeichnisses im entweder weggelassen oder auf festgelegt werden`/`. Dadurch wird der Pfad erzwungen, der auf dem EFS-Zugriffspunkt festgelegt ist. Wenn ein Zugriffspunkt verwendet wird, muss in `EFSVolumeConfiguration` die Transitverschlüsselung aktiviert sein. Weitere Informationen finden Sie unter Arbeiten mit Amazon EFS-Zugriffspunkten im Amazon Elastic File System-Benutzerhandbuch.

`iam`

Typ: Zeichenfolge

Zulässige Werte: `ENABLED` | `DISABLED`

Erforderlich: Nein

Bestimmt, ob die AWS Batch Auftrags-IAM-Rolle verwendet werden soll, die in einer Auftragsdefinition definiert ist, wenn das Amazon-EFS-Dateisystem gemountet wird. Wenn diese Option aktiviert ist, muss die Transit-Verschlüsselung in `EFSVolumeConfiguration` aktiviert sein. Wenn dieser Parameter nicht angegeben ist, wird der Standardwert `DISABLED` verwendet. Weitere Informationen zur Ausführung von IAM-Rollen finden Sie unter AWS Batch Ausführung, IAM-Rolle.

Beispiel für Auftragsdefinitionen

Die folgenden Beispiele für Auftragsdefinitionen veranschaulichen, wie Sie allgemeine Muster wie Umgebungsvariablen, Parameterersetzung und Volume-Mounts verwenden.

Verwenden von Umgebungsvariablen

In der folgenden Beispielauftragsdefinition werden Umgebungsvariablen verwendet, um einen Dateityp und eine Amazon S3-URL anzugeben. Dieses Beispiel stammt aus dem Datenverarbeitungs-Blogpost zum Anlegen eines einfachen "Fetch & Run"-AWS Batch-Jobs. Das im Blogbeitrag beschriebene `fetch_and_run.sh`Skript verwendet diese Umgebungsvariablen, um das `myjob.sh`Skript von S3 herunterzuladen und seinen Dateityp zu deklarieren.

Obwohl die Befehls- und Umgebungsvariablen in diesem Beispiel fest in die Auftragsdefinition codiert sind, können Sie Überschreibungen von Befehls- und Umgebungsvariablen angeben, um die Auftragsdefinition noch umfassender zu gestalten.

```
{
    "jobDefinitionName": "fetch_and_run",
    "type": "container",
    "containerProperties": {
        "image": "123456789012.dkr.ecr.us-east-1.amazonaws.com/fetch_and_run",
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "2000"
            },
            {
                "type": "VCPU",
                "value": "2"
            }
        ],
        "command": [
            "myjob.sh",
            "60"
        ],
        "jobRoleArn": "arn:aws:iam::123456789012:role/AWSBatchS3ReadOnly",
        "environment": [
            {
                "name": "BATCH_FILE_S3_URL",
                "value": "s3://my-batch-scripts/myjob.sh"
            },
            {
                "name": "BATCH_FILE_TYPE",
                "value": "script"
            }
        ],
        "user": "nobody"
    }
}
```

Verwenden der Parameterersetzung

Im folgenden Auftragsdefinitionsbeispiel wird gezeigt, wie Sie die Parameterersetzung zulassen und Standardwerte festlegen.

Die `Ref::`-Deklarationen im Abschnitt `command` werden zum Festlegen von Platzhaltern für die Parameterersetzung verwendet. Wenn Sie einen Auftrag mit dieser Auftragsdefinition senden, geben Sie die Parameterüberschreibungen für diese Werte an, z. B. `inputfile` und `outputfile`. Im folgenden `parameters` Abschnitt wird ein Standardwert für festgelegt`codec`, Sie können diesen Parameter jedoch nach Bedarf überschreiben.

Weitere Informationen finden Sie unter Parameter.

```
{
    "jobDefinitionName": "ffmpeg_parameters",
    "type": "container",
    "parameters": {"codec": "mp4"},
    "containerProperties": {
        "image": "my_repo/ffmpeg",
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "2000"
            },
            {
                "type": "VCPU",
                "value": "2"
            }
        ],
        "command": [
            "ffmpeg",
            "-i",
            "Ref::inputfile",
            "-c",
            "Ref::codec",
            "-o",
            "Ref::outputfile"
        ],
        "jobRoleArn": "arn:aws:iam::123456789012:role/ECSTask-S3FullAccess",
        "user": "nobody"
    }
}
```

Testen der GPU-Funktionalität

Mit dem folgende Auftragsdefinitionsbeispiel wird getestet, ob die GPU-Workload-AMI in Verwenden eines GPU-Workload-AMI ordnungsgemäß konfiguriert ist. In dieser Beispielauftragsdefinition wird das TensorFlow Deep MNIST-Classifier-Beispiel von ausgeführt GitHub.

```
{
    "containerProperties": {
        "image": "tensorflow/tensorflow:1.8.0-devel-gpu",
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "32000"
            },
            {
                "type": "VCPU",
                "value": "8"
            }
        ],
        "command": [
            "sh",
            "-c",
            "cd /tensorflow/tensorflow/examples/tutorials/mnist; python mnist_deep.py"
        ]
    },
    "type": "container",
    "jobDefinitionName": "tensorflow_mnist_deep"
}
```

Sie können eine Datei mit dem vorherigen JSON-Text erstellen `tensorflow_mnist_deep.json` und dann eine -AWS BatchAuftragsdefinition mit dem folgenden Befehl registrieren:

```
aws batch register-job-definition --cli-input-json file://tensorflow_mnist_deep.json
```

Paralleler Auftrag mit mehreren Knoten

Die folgende beispielhafte Aufgabendefinition veranschaulicht eine parallele Aufgabe mit mehreren Knoten. Weitere Informationen finden Sie unter Erstellen eines eng gekoppelten Workflows für die synchrone Dynamics mit parallelen Aufträgen mit mehreren Knoten in AWS Batch im AWS -Compute-Blog.

```
{
  "jobDefinitionName": "gromacs-jobdef",
  "jobDefinitionArn": "arn:aws:batch:us-east-2:123456789012:job-definition/gromacs-
jobdef:1",
  "revision": 6,
  "status": "ACTIVE",
  "type": "multinode",
  "parameters": {},
  "nodeProperties": {
    "numNodes": 2,
    "mainNode": 0,
    "nodeRangeProperties": [
      {
        "targetNodes": "0:1",
        "container": {
          "image": "123456789012.dkr.ecr.us-east-2.amazonaws.com/gromacs_mpi:latest",
          "resourceRequirements": [
              {
                  "type": "MEMORY",
                  "value": "24000"
              },
              {
                  "type": "VCPU",
                  "value": "8"
              }
          ],
          "command": [],
          "jobRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
          "ulimits": [],
          "instanceType": "p3.2xlarge"
        }
      }
    ]
  }
}
```

Warteschlangen für Job

Jobs werden an eine Auftragswarteschlange weitergeleitet, in der sie gespeichert werden, bis sie für die Ausführung in einer Rechenumgebung geplant werden können. Ein AWS Konto kann mehrere Jobwarteschlangen haben. Sie können beispielsweise eine Warteschlange erstellen, die Amazon EC2 On-Demand-Instances für Jobs mit hoher Priorität verwendet, und eine weitere Warteschlange, die Amazon EC2 Spot-Instances für Jobs mit niedriger Priorität verwendet. Job-Warteschlangen haben eine Priorität, die vom Scheduler verwendet wird, um zu bestimmen, welche Jobs in welcher Warteschlange zuerst zur Ausführung ausgewertet werden sollen.

Themen

Erstellen einer Auftragswarteschlange

Bevor Sie Aufträge in AWS Batch senden können, müssen Sie eine Auftragswarteschlange erstellen. Wenn Sie eine Auftragswarteschlange erstellen, ordnen Sie der Warteschlange eine oder mehrere Datenverarbeitungsumgebungen zu und weisen eine Reihenfolge der Präferenz zu.

Sie legen auch die Priorität für die Auftragswarteschlange fest, die die Reihenfolge bestimmt, in der der AWS Batch-Scheduler Aufträge platziert. Das bedeutet, dass, wenn eine Datenverarbeitungsumgebung mehr als einer Auftragswarteschlange zugeordnet ist, der Auftragswarteschlange mit einer höheren Priorität Vorrang gegeben wird.

Erstellen einer Fargate-Auftragswarteschlange

So erstellen Sie eine Fargate-Auftragswarteschlange

1. Öffnen Sie die -AWS BatchKonsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie die zu verwendende AWS-Region in der Navigationsleiste aus.
3. Wählen Sie im Navigationsbereich Auftragswarteschlangen aus.
4. Wählen Sie Erstellen.
5. Wählen Sie für Orchestrierungstyp Fargate aus.

6. Geben Sie unter Name einen eindeutigen Namen für Ihre Auftragswarteschlange ein. Der Name kann bis zu 128 Zeichen lang sein und Groß- und Kleinbuchstaben, Zahlen und Unterstriche (_) enthalten.
7. Geben Sie für Priorität einen Ganzzahlwert für die Priorität der Auftragswarteschlange ein. Auftragswarteschlangen mit höherer Priorität werden vor Auftragswarteschlangen mit niedrigerer Priorität ausgeführt, die derselben Datenverarbeitungsumgebung zugeordnet sind. Die Priorität wird in absteigender Reihenfolge ermittelt. So erhält beispielsweise eine Auftragswarteschlange mit einem Prioritätswert von 10 Vorrang vor einer Auftragswarteschlange mit dem Wert 1.
8. (Optional) Wählen Sie unter Amazon-Ressourcenname (ARN) der Planungsrichtlinie eine vorhandene Planungsrichtlinie aus.
9. Wählen Sie für Verbundene Datenverarbeitungsumgebungen eine oder mehrere Datenverarbeitungsumgebungen aus der Liste aus, die der Auftragswarteschlange zugeordnet werden sollen. Wählen Sie Datenverarbeitungsumgebungen in der Reihenfolge aus, in der die Warteschlange versuchen soll, die Platzierung der Auftragswarteschlange zu versuchen. Der Auftrags-Scheduler verwendet die Reihenfolge, in der Sie Datenverarbeitungsumgebungen auswählen, um zu bestimmen, welche Datenverarbeitungsumgebung einen bestimmten Auftrag startet. Bevor Sie sie einer Auftragswarteschlange zuordnen können, müssen sich die Datenverarbeitungsumgebungen im `VALID` Status befinden. Sie können bis zu drei Datenverarbeitungsumgebungen mit einer Auftragswarteschlange verknüpfen.

   > **Note**
   >
   > Alle Datenverarbeitungsumgebungen, die einer Auftragswarteschlange zugeordnet sind, müssen dasselbe Bereitstellungsmodell verwenden. unterstützt AWS Batch nicht das Mischen von Bereitstellungsmodellen in einer einzigen Auftragswarteschlange.

10. Wählen Sie für Reihenfolge der Datenverarbeitungsumgebung die Aufwärts- und Abwärtspfeile aus, um die gewünschte Reihenfolge zu konfigurieren.
11. Wählen Sie Auftragswarteschlange erstellen aus, um den Vorgang abzuschließen und Ihre Auftragswarteschlange zu erstellen.

Erstellen einer Amazon EC2-Auftragswarteschlange

So erstellen Sie eine Amazon EC2-Auftragswarteschlange

1. Öffnen Sie die -AWS BatchKonsole unter https://console.aws.amazon.com/batch/.

2. Wählen Sie die zu verwendende AWS-Region in der Navigationsleiste aus.
3. Wählen Sie im Navigationsbereich Auftragswarteschlangen aus.
4. Wählen Sie Erstellen.
5. Wählen Sie für Orchestrierungstyp Amazon Elastic Compute Cloud (Amazon EC2) aus.
6. Geben Sie unter Name einen eindeutigen Namen für Ihre Auftragswarteschlange ein. Der Name kann bis zu 128 Zeichen lang sein und Groß- und Kleinbuchstaben, Zahlen und Unterstriche (_) enthalten.
7. Geben Sie für Priorität einen Ganzzahlwert für die Priorität der Auftragswarteschlange ein. Auftragswarteschlangen mit höherer Priorität werden vor Auftragswarteschlangen mit niedrigerer Priorität ausgeführt, die derselben Datenverarbeitungsumgebung zugeordnet sind. Die Priorität wird in absteigender Reihenfolge ermittelt. So erhält beispielsweise eine Auftragswarteschlange mit einem Prioritätswert von 10 Vorrang vor einer Auftragswarteschlange mit dem Wert 1.
8. (Optional) Wählen Sie unter Amazon-Ressourcenname (ARN) der Planungsrichtlinie eine vorhandene Planungsrichtlinie aus.
9. Wählen Sie für Verbundene Datenverarbeitungsumgebungen eine oder mehrere Datenverarbeitungsumgebungen aus der Liste aus, die der Auftragswarteschlange zugeordnet werden sollen. Wählen Sie Datenverarbeitungsumgebungen in der Reihenfolge aus, in der die Warteschlange versuchen soll, die Platzierung der Auftragswarteschlange zu versuchen. Der Auftrags-Scheduler verwendet die Reihenfolge, in der Sie Datenverarbeitungsumgebungen auswählen, um zu bestimmen, welche Datenverarbeitungsumgebung einen bestimmten Auftrag startet. Bevor Sie sie einer Auftragswarteschlange zuordnen können, müssen sich die Datenverarbeitungsumgebungen im `VALID` Status befinden. Sie können bis zu drei Datenverarbeitungsumgebungen mit einer Auftragswarteschlange verknüpfen. Wenn Sie noch keine Computing-Umgebung haben, wählen Sie Computing-Umgebung erstellen aus.

   > **Note**
   >
   > Alle Datenverarbeitungsumgebungen, die einer Auftragswarteschlange zugeordnet sind, müssen dasselbe Bereitstellungsmodell verwenden. unterstützt AWS Batch nicht das Mischen von Bereitstellungsmodellen in einer einzigen Auftragswarteschlange.

10. Wählen Sie für Reihenfolge der Datenverarbeitungsumgebung die Aufwärts- und Abwärtspfeile aus, um die gewünschte Reihenfolge zu konfigurieren.
11. Wählen Sie Auftragswarteschlange erstellen, um den Vorgang abzuschließen und Ihre Auftragswarteschlange zu erstellen.

Erstellen einer Amazon-EKS-Auftragswarteschlange

So erstellen Sie eine Amazon-EKS-Auftragswarteschlange

1. Öffnen Sie die -AWS BatchKonsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie die zu verwendende AWS-Region in der Navigationsleiste aus.
3. Wählen Sie im Navigationsbereich Auftragswarteschlangen aus.
4. Wählen Sie Erstellen.
5. Wählen Sie für Orchestrierungstyp die Option Amazon Elastic Kubernetes Service (Amazon EKS) aus.
6. Geben Sie unter Name einen eindeutigen Namen für Ihre Auftragswarteschlange ein. Der Name kann bis zu 128 Zeichen lang sein und Groß- und Kleinbuchstaben, Zahlen und Unterstriche (_) enthalten.
7. Geben Sie für Priorität einen Ganzzahlwert für die Priorität der Auftragswarteschlange ein. Auftragswarteschlangen mit höherer Priorität werden vor Auftragswarteschlangen mit niedrigerer Priorität ausgeführt, die derselben Datenverarbeitungsumgebung zugeordnet sind. Die Priorität wird in absteigender Reihenfolge ermittelt. So erhält beispielsweise eine Auftragswarteschlange mit einem Prioritätswert von 10 Vorrang vor einer Auftragswarteschlange mit dem Wert 1.
8. (Optional) Wählen Sie unter Amazon-Ressourcenname (ARN) der Planungsrichtlinie eine vorhandene Planungsrichtlinie aus.
9. Wählen Sie für Verbundene Datenverarbeitungsumgebungen eine oder mehrere Datenverarbeitungsumgebungen aus der Liste aus, die der Auftragswarteschlange zugeordnet werden sollen. Wählen Sie Datenverarbeitungsumgebungen in der Reihenfolge aus, in der die Warteschlange versuchen soll, die Platzierung der Auftragswarteschlange zu versuchen. Der Auftrags-Scheduler verwendet die Reihenfolge, in der Sie Datenverarbeitungsumgebungen auswählen, um zu bestimmen, welche Datenverarbeitungsumgebung einen bestimmten Auftrag startet. Bevor Sie sie einer Auftragswarteschlange zuordnen können, müssen sich die Datenverarbeitungsumgebungen im `VALID` Status befinden. Sie können bis zu drei Datenverarbeitungsumgebungen mit einer Auftragswarteschlange verknüpfen.

> **Note**
>
> Alle Datenverarbeitungsumgebungen, die einer Auftragswarteschlange zugeordnet sind, müssen dasselbe Bereitstellungsmodell verwenden. unterstützt AWS Batch nicht das Mischen von Bereitstellungsmodellen in einer einzigen Auftragswarteschlange.

> **Note**
>
> Alle Computing-Umgebungen, die mit einer Auftragswarteschlange verknüpft sind, müssen dieselbe Architektur verwenden. AWS Batch unterstützt nicht das Vermischen von Architekturtypen der Computing-Umgebung in einer einzigen Auftragswarteschlange.

10. Wählen Sie für Reihenfolge der Datenverarbeitungsumgebung die Aufwärts- und Abwärtspfeile aus, um die gewünschte Reihenfolge zu konfigurieren.
11. Wählen Sie Auftragswarteschlange erstellen aus, um den Vorgang abzuschließen und Ihre Auftragswarteschlange zu erstellen.

Auftragswarteschlangenvorlage

Im Folgenden finden Sie eine leere Auftragswarteschlangenvorlage. Sie können diese Vorlage verwenden, um Ihre Auftragswarteschlange zu erstellen. Anschließend können Sie diese Auftragswarteschlange in einer Datei speichern und mit der AWS CLI `--cli-input-json` Option verwenden. Weitere Informationen zu diesen Parametern finden Sie unter CreateJobQueue in der APIAWS Batch -Referenz zu .

```
{
   "computeEnvironmentOrder": [
      {
         "computeEnvironment": "",
         "order": 0
      }
   ],
   "jobQueueName": "",
   "jobStateTimeLimitActions": [
      {
         "state": "RUNNABLE",
         "action": "CANCEL",
   "maxTimeSeconds": 0,
         "reason": ""

      }
   ],
   "priority": 0,
   "schedulingPolicyArn": "",
```

```
   "state": "ENABLED",
   "tags": {
      "KeyName": ""
   }
}
```

> **Note**
>
> Sie können die vorangehende Auftragswarteschlangenvorlage mit dem folgenden AWS CLI Befehl generieren.
>
> ```
> $ aws batch create-job-queue --generate-cli-skeleton
> ```

Auftragswarteschlangenparameter

Auftragswarteschlangen sind in vier grundlegende Komponenten unterteilt: Name, Status, Priorität und Reihenfolge der Datenverarbeitungsumgebung. In diesem Abschnitt werden diese Komponenten beschrieben, die zugeordnet sind.

Themen

Name der Auftragswarteschlange

`jobQueueName`

Der Name für Ihre Auftragswarteschlange. Bis zu 128 Buchstaben (Groß- und Kleinbuchstaben), Ziffern und Unterstriche sind zulässig.

Typ: Zeichenfolge

Erforderlich: Ja

Aktionen für das Zeitlimit für den Status der Auftragswarteschlange

`jobStateTimeLimitActions`

Der Satz von Aktionen, die für Aufträge AWS Batch ausführt, die im angegebenen Zustand länger als angegeben an der Spitze der Auftragswarteschlange verbleiben. AWS Batch führt jede Aktion aus, nachdem abgelaufen `maxTimeSeconds` ist. (Hinweis: Der Mindestwert für `maxTimeSeconds` ist 600 (10 Minuten) und sein Höchstwert ist 86.400 (24 Stunden).

Typ: Array von `JobStateTimeLimitActions`-Objekten

Erforderlich: Nein

Priorität

`priority`

Die Priorität der Auftragswarteschlange. Auftragswarteschlangen mit höherer Priorität (oder einen höheren Ganzzahlwert für den Parameter `priority`) werden zuerst bewertet, wenn sie der gleichen Datenverarbeitungsumgebung zugewiesen sind. Die Priorität wird in absteigender Reihenfolge ermittelt. Eine Auftragswarteschlange mit einem Prioritätswert von 10 erhält z. B. Vorrang vor einer Auftragswarteschlange mit dem Wert 1. Alle Datenverarbeitungsumgebungen müssen entweder Amazon EC2 (`EC2` oder `SPOT`) oder Fargate (`FARGATE` oder) sein`FARGATE_SPOT`. Amazon EC2- und Fargate-Rechenumgebungen können nicht gemischt werden.

Typ: Ganzzahl

Erforderlich: Ja

Planungsrichtlinie

`schedulingPolicyArn`

Der Amazon-Ressourcenname (ARN) der Planungsrichtlinie für die Auftragswarteschlange. Auftragswarteschlangen, die keine Planungsrichtlinie haben, werden in einem First-In-First-Out-

Modell (FIFO) geplant. Nachdem eine Auftragswarteschlange über eine Planungsrichtlinie verfügt, kann sie ersetzt, aber nicht entfernt werden. Eine Auftragswarteschlange ohne Planungsrichtlinie wird als FIFO-Auftragswarteschlange geplant und kann keine Planungsrichtlinie hinzugefügt werden. Auftragswarteschlangen mit einer Planungsrichtlinie können maximal 500 aktive Fair-Share-Kennungen haben. Wenn das Limit erreicht ist, schlagen alle Aufträge, die eine neue Fair-Share-Kennung hinzufügen, fehl.

Typ: Zeichenfolge

Erforderlich: Nein

Status

`state`

Der Status der Auftragswarteschlange. Wenn der Status der Auftragswarteschlange `ENABLED` (Standardwert) lautet, kann sie Aufträge akzeptieren. Wenn der Status der Auftragswarteschlange `DISABLED` lautet, können der Warteschlange keine neuen Aufträge hinzugefügt werden. Aufträge, die sich bereits in der Warteschlange befinden, können aber beendet werden.

Typ: Zeichenfolge

Zulässige Werte: `ENABLED` | `DISABLED`

Erforderlich: Nein

Reihenfolge der Datenverarbeitungsumgebung

`computeEnvironmentOrder`

Die Gruppe von Datenverarbeitungsumgebungen, die einer Auftragswarteschlange zugeordnet sind, und ihre Reihenfolge untereinander. Der Job Scheduler verwendet diesen Parameter, um zu ermitteln, welche Datenverarbeitungsumgebung einen bestimmten Auftrag ausführen soll. Datenverarbeitungsumgebungen müssen sich im Status `VALID` befinden, bevor Sie sie mit einer Auftragswarteschlange verknüpfen können. Sie können bis zu drei Datenverarbeitungsumgebungen mit einer Auftragswarteschlange verknüpfen. Alle Datenverarbeitungsumgebungen müssen entweder Amazon EC2 (`EC2` oder `SPOT`) oder Fargate (`FARGATE` oder) sein`FARGATE_SPOT`. Amazon EC2- und Fargate-Rechenumgebungen können nicht gemischt werden.

> **Note**
>
> Alle Computing-Umgebungen, die einer Auftragswarteschlange zugeordnet sind, müssen dieselbe Architektur verwenden. unterstützt AWS Batch keine Mischung von Architekturtypen der Computing-Umgebung in einer einzigen Auftragswarteschlange.

Typ: Array von ComputeEnvironmentOrder-Objekten

Erforderlich: Ja

`computeEnvironment`

Der Amazon-Ressourcenname (ARN) der Datenverarbeitungsumgebung.

Typ: Zeichenfolge

Erforderlich: Ja

`order`

Die Reihenfolge der Datenverarbeitungsumgebung. Datenverarbeitungsumgebungen werden in aufsteigender Reihenfolge ausprobiert. Wenn beispielsweise zwei Datenverarbeitungsumgebungen einer Auftragswarteschlange zugeordnet sind, wird zuerst versucht, die Datenverarbeitungsumgebung mit einem niedrigeren Ganzzahlwert `order` für die Auftragsplatzierung zu verwenden.

Tags

`tags`

Schlüssel-Wert-Paar-Tags, die der Auftragswarteschlange zugeordnet werden sollen. Weitere Informationen finden Sie unter Markieren Ihrer AWS Batch-Ressourcen.

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Erforderlich: Nein

Status der Auftragswarteschlange anzeigen

Nachdem Sie eine Auftragswarteschlange erstellt und die Jobs eingereicht haben, ist es wichtig, deren Fortschritt überwachen zu können. Sie können die Seite mit den Auftragsdetails verwenden, um Ihre Job-Warteschlange zu überprüfen, zu verwalten und zu überwachen.

Informationen zur Job-Warteschlange anzeigen

Wählen Sie in der AWS Batch Konsole im Navigationsbereich Auftragswarteschlangen und wählen Sie die gewünschte Auftragswarteschleife aus, um deren Details anzuzeigen. Auf dieser Seite können Sie Ihre Auftragswarteschlange überprüfen und verwalten und zusätzliche Informationen zu den Vorgängen der Warteschlange einsehen, z. B. den Job-Warteschlangen-Snapshot, die Auftragsstatusgrenzen, die Reihenfolge der Umgebung, Tags und den JSON-Code der Job-Warteschlange.

Details zur Auftragswarteschlange

Dieser Abschnitt bietet eine Übersicht und Wartungsoptionen für die Auftragswarteschlange. Es ist wichtig zu beachten, dass Sie den Amazon-Ressourcennamen (ARN) in diesem Abschnitt finden.

Um diese Informationen über zu finden AWS Command Line Interface, verwenden Sie den `DescribeJobQueues`Vorgang zusammen mit dem Namen der Auftragswarteschlange oder dem entsprechenden ARN.

Snapshot der Job-Warteschlange

Dieser Abschnitt enthält eine statische Liste der ersten 100 `RUNNABLE` Jobs, die sich in der Warteschlange befinden. Sie können das Suchfeld verwenden, um die Liste einzugrenzen, indem Sie in einer beliebigen Spalte des Ergebnisabschnitts nach Informationen suchen. Die Jobs im Bereich mit den Snapshot-Ergebnissen sind entsprechend der Ausführungsstrategie der Job-Warteschlange sortiert. Bei first-in-first-out (FIFO-) Jobwarteschlangen basiert die Reihenfolge der Jobs auf der Übermittlungszeit. Bei Jobwarteschlangen (AWS Batch Fair Share Scheduling, FSS) basiert die Reihenfolge der Jobs auf der Priorität der Jobs und der gemeinsamen Nutzung.

Da es sich bei den Ergebnissen um eine Momentaufnahme der Auftragswarteschlange handelt, wird die Ergebnisliste nicht automatisch aktualisiert. Um die Liste zu aktualisieren, wählen Sie oben im Abschnitt die Option Aktualisieren aus. Wählen Sie den Hyperlink mit dem Jobnamen, um zu den Jobdetails zu navigieren und den Status des Jobs und andere zugehörige Informationen einzusehen.

Um diese Informationen über zu finden AWS CLI, verwenden Sie den `GetJobQueueSnapshot`Vorgang zusammen mit dem Namen der Auftragswarteschlange oder dem entsprechenden ARN.

Grenzwerte für den Beschäftigungsstatus

Verwenden Sie diese Registerkarte, um die Konfigurationsinformationen darüber zu überprüfen, wie lange ein Job in einem bestimmten `RUNNABLE` Status verbleiben kann, bevor er storniert wird.

Um diese Informationen über zu finden AWS CLI, verwenden Sie den `DescribeJobQueues`Vorgang zusammen mit dem Namen der Auftragswarteschlange oder dem entsprechenden ARN.

Reihenfolge der Umgebung

Wenn Ihre Job-Warteschlange in mehreren Umgebungen läuft, finden Sie auf dieser Registerkarte deren Reihenfolge und einen Überblick.

Um diese Informationen über zu finden AWS CLI, verwenden Sie den `DescribeJobQueues`Vorgang zusammen mit dem Namen der Auftragswarteschlange oder dem entsprechenden ARN.

Tags

Verwenden Sie diese Registerkarte, um die Tags zu überprüfen und zu verwalten, die dieser Auftragswarteschlange zugeordnet sind.

JSON

Verwenden Sie diese Registerkarte, um den JSON-Code zu kopieren, der dieser Job-Warteschlange zugeordnet ist. Sie können den JSON dann für AWS CloudFormation Vorlagen und AWS CLI Skripte wiederverwenden.

Planung von Aufträgen

Der AWS Batch Scheduler bewertet, wann, wo und wie Jobs ausgeführt werden, die in eine Job-Warteschlange gestellt werden. Wenn Sie beim Erstellen einer Job-Warteschlange keine Scheduling-Richtlinie angeben, verwendet der AWS Batch Job-Scheduler standardmäßig eine FIFO-Strategie (First-In, First Out). Eine FIFO-Strategie kann dazu führen, dass wichtige Jobs hinter Jobs „hängen bleiben“, die zuvor eingereicht wurden. Indem Sie eine andere Planungsrichtlinie angeben, können Sie Rechenressourcen Ihren spezifischen Anforderungen entsprechend zuweisen.

> **Note**
>
> Wenn Sie die spezifische Reihenfolge planen möchten, in der Jobs ausgeführt werden, verwenden Sie den `dependsOn` Parameter in, SubmitJobum die Abhängigkeiten für jeden Job anzugeben.

Wenn Sie eine Planungsrichtlinie erstellen und diese an eine Job-Warteschlange anhängen, ist Fair Share Scheduling aktiviert. Wenn die Auftragswarteschlange über eine Planungsrichtlinie verfügt, bestimmt die Planungsrichtlinie die Reihenfolge, in der Jobs ausgeführt werden. Weitere Informationen finden Sie unter Planungsrichtlinien.

Kennungen teilen

Sie können gemeinsame Kennungen verwenden, um Jobs zu taggen und zwischen Benutzern und Workloads zu unterscheiden. Der AWS Batch Scheduler verfolgt die Nutzung für jeden Fair-Share-Identifier mithilfe der *`(T * weightFactor)`* Formel, wobei die vCPU im Zeitverlauf angegeben *T*ist. Der Scheduler wählt anhand der Share-ID Jobs mit der geringsten Auslastung aus. Sie können einen Fair-Share-Identifier verwenden, ohne ihn zu überschreiben.

> **Note**
>
> Share-IDs sind innerhalb einer Auftragswarteschlange eindeutig und werden nicht in allen Jobwarteschlangen zusammengefasst.

Sie können die Planungspriorität festlegen, um die Reihenfolge zu konfigurieren, in der Jobs anhand einer gemeinsamen ID ausgeführt werden. Jobs mit einer höheren Planungspriorität

werden zuerst geplant. Wenn Sie keine Planungsrichtlinie angeben, werden alle Jobs, die an die Auftragswarteschlange gesendet werden, in der FIFO-Reihenfolge geplant. Wenn Sie einen Job einreichen, können Sie keine gemeinsame Kennung oder Planungspriorität angeben.

> **Note**
>
> Angehängte Rechenressourcen werden allen Share-IDs gleichmäßig zugewiesen, sofern sie nicht ausdrücklich überschrieben werden.

Faire Planung der Anteile

Fair Share Scheduling bietet eine Reihe von Steuerelementen, mit denen Sie Jobs besser planen können.

> **Note**
>
> Weitere Informationen zur Planung von Richtlinienparametern finden Sie unterPlanungsrichtlinienparameter.

- Share Decay-Sekunden — Der Zeitraum (in Sekunden), den der AWS Batch Scheduler verwendet, um für jeden Fair-Share-Identifier einen Fair-Share-Prozentsatz zu berechnen. Ein Wert von Null gibt an, dass nur die aktuelle Nutzung gemessen wird. Eine längere Abklingzeit verleiht der Zeit mehr Gewicht.

  > **Note**
  >
  > Der Zeitraum für den Zerfall wird wie folgt berechnet: *`shareDecaySeconds + OrderMinutes`*wo *`OrderMinutes`* ist die Zeit in der Reihenfolge in Minuten.

- Compute-Reservierung — Verhindert, dass Jobs in einem einzigen Share-Identifier alle Ressourcen verbrauchen, die an die Auftragswarteschlange angehängt sind. Bei der reservierten Quote ActiveFairShares handelt es *`computeReservation/100)^ActiveFairShares`* sich um die Anzahl der aktiven Fair-Share-Identifikatoren.

> **Note**
>
> Wenn ein Share Identifier Jobs in einem`SUBMITTED`,`PENDING`, `RUNNABLE`STARTING, oder `RUNNING` Bundesstaat hat, wird er als aktive Aktienidentifikation betrachtet. Nach Ablauf der Gültigkeitsdauer gilt eine Share-ID als inaktiv.

- Gewichtungsfaktor — Der Gewichtungsfaktor für die Aktienkennung. Der Standardwert lautet 1. Ein niedrigerer Wert ermöglicht die Ausführung von Aufträgen aus dem Share Identifier oder verleiht dem Share Identifier zusätzliche Laufzeit. Jobs, die eine gemeinsame Kennung mit einem Gewichtungsfaktor von 0,125 (1/8) verwenden, werden beispielsweise achtmal so viele Rechenressourcen zugewiesen wie Jobs, die eine gemeinsame ID mit einem Gewichtungsfaktor von 1 verwenden.

  > **Note**
  >
  > Sie müssen dieses Attribut nur definieren, wenn Sie den Standard-Gewichtungsfaktor 1 aktualisieren müssen.

Wenn die Job-Queue aktiv ist und Jobs verarbeitet, können Sie im Job-Queue-Snapshot eine Liste der ersten 100 `RUNNABLE` Jobs überprüfen. Weitere Informationen finden Sie unter Status der Auftragswarteschlange anzeigen.

Datenverarbeitungsumgebung

Auftragswarteschlangen werden einer oder mehreren Datenverarbeitungsumgebungen zugeordnet. Rechenumgebungen enthalten die Amazon ECS-Container-Instances, die zur Ausführung von containerisierten Batch-Jobs verwendet werden. Eine bestimmte Rechenumgebung kann auch einer oder mehreren Auftragswarteschlangen zugeordnet werden. Innerhalb einer Job-Warteschlange haben die zugehörigen Rechenumgebungen jeweils eine Reihenfolge, anhand derer der Scheduler bestimmt, wo Jobs, die zur Ausführung bereit sind, ausgeführt werden sollen. Wenn die erste Rechenumgebung den Status hat `VALID` und über verfügbare Ressourcen verfügt, wird der Job für eine Container-Instance innerhalb dieser Rechenumgebung geplant. Wenn die erste Rechenumgebung den Status hat `INVALID` oder keine geeignete Rechenressource bereitstellen kann, versucht der Scheduler, den Job in der nächsten Rechenumgebung auszuführen.

Themen

Verwaltete Computerumgebungen

Sie können eine verwaltete Rechenumgebung verwenden, um die Kapazität und die Instanztypen der Rechenressourcen innerhalb der Umgebung zu AWS Batch verwalten. Dies basiert auf den Spezifikationen für Rechenressourcen, die Sie bei der Erstellung der Rechenumgebung definieren.

Sie können wählen, ob Sie Amazon EC2 On-Demand-Instances und Amazon EC2-Spot-Instances verwenden möchten. Oder Sie können alternativ die Kapazität von Fargate und Fargate Spot in Ihrer verwalteten Computerumgebung verwenden. Wenn Sie Spot-Instances verwenden, können Sie optional einen Höchstpreis festlegen. Auf diese Weise werden Spot-Instances nur gestartet, wenn der Spot-Instance-Preis unter einem bestimmten Prozentsatz des On-Demand-Preises liegt.

> **Important**
>
> Fargate Spot-Instances werden auf Windows containers on AWS Fargate nicht unterstützt. Eine Job-Warteschlange wird blockiert, wenn ein FargateWindows Job an eine Job-Warteschlange weitergeleitet wird, die nur Fargate Spot-Computerumgebungen verwendet.

Verwaltete Rechenumgebungen starten Amazon EC2 EC2-Instances in der VPC und den Subnetzen, die Sie angeben, und registrieren sie dann bei einem Amazon ECS-Cluster. Die Amazon EC2 EC2-Instances benötigen externen Netzwerkzugriff, um mit dem Amazon ECS-Serviceendpunkt zu kommunizieren. Einige Subnetze stellen Amazon EC2 EC2-Instances keine öffentlichen IP-Adressen zur Verfügung. Wenn Ihre Amazon EC2 EC2-Instances keine öffentliche IP-Adresse haben, müssen sie Network Address Translation (NAT) verwenden, um diesen Zugriff zu erhalten. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch. Weitere Informationen zum Erstellen einer VPC finden Sie unterErstellen einer Virtual Private Cloud .

Standardmäßig verwenden AWS Batch verwaltete Computerumgebungen eine aktuelle, genehmigte Version des für Amazon ECS optimierten AMI für Rechenressourcen. Möglicherweise möchten Sie jedoch aus verschiedenen Gründen Ihr eigenes AMI erstellen, das Sie für Ihre verwalteten Computerumgebungen verwenden können. Weitere Informationen finden Sie unter Ressourcen-AMIs berechnen.

> **Note**
>
> AWS Batch aktualisiert die AMIs in einer Computerumgebung nicht automatisch, nachdem sie erstellt wurden. Beispielsweise werden die AMIs in Ihrer Rechenumgebung nicht aktualisiert, wenn eine neuere Version des für Amazon ECS optimierten AMI veröffentlicht wird. Sie sind für die Verwaltung des Gastbetriebssystems verantwortlich. Dazu gehören alle Updates und Sicherheitspatches. Sie sind auch für jede zusätzliche Anwendungssoftware oder Hilfsprogramme verantwortlich, die Sie auf den Rechenressourcen installieren. Es gibt zwei Möglichkeiten, ein neues AMI für Ihre AWS Batch Jobs zu verwenden. Die ursprüngliche Methode besteht darin, die folgenden Schritte auszuführen:

1. Erstellen Sie eine neue Datenverarbeitungsumgebung mit dem neuen AMI.
2. Fügen Sie die Datenverarbeitungsumgebung einer vorhandenen Auftragswarteschlange hinzu.
3. Entfernen Sie die alte Datenverarbeitungsumgebung aus Ihrer Auftragswarteschlange.
4. Löschen Sie die alte Datenverarbeitungsumgebung.

Im April 2022 AWS Batch wurde erweiterte Unterstützung für die Aktualisierung von Computerumgebungen hinzugefügt. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen. Beachten Sie die folgenden Regeln, um die erweiterte Aktualisierung von Computing-Umgebungen zum Aktualisieren von AMIs zu verwenden:

- Legen Sie entweder den Parameter service role (`serviceRole`) nicht fest oder legen Sie ihn auf die AWSServiceRoleForBatchdienstverknüpfte Rolle fest.
- Setzen Sie den Parameter Allocation Strategy (`allocationStrategy`) auf`BEST_FIT_PROGRESSIVE`, `SPOT_CAPACITY_OPTIMIZED` oder`SPOT_PRICE_CAPACITY_OPTIMIZED`.
- Stellen Sie den Parameter Update auf die neueste Image-Version (`updateToLatestImageVersion`) auf ein`true`.
- Geben Sie keine AMI-ID in `imageId`, `imageIdOverride`(in `ec2Configuration`) oder in der Startvorlage (`launchTemplate`) an. AWS Batch Wählt in diesem Fall das neueste Amazon ECS-optimierte AMI aus, das AWS Batch zum Zeitpunkt der Initiierung des Infrastruktur-Updates unterstützt wird. Alternativ können Sie die AMI-ID in den `imageIdOverride` Parametern `imageId` oder oder die durch die `LaunchTemplate` Eigenschaften identifizierte Startvorlage angeben. Wenn Sie eine dieser Eigenschaften ändern, wird ein Infrastruktur-Update gestartet. Wenn die AMI-ID in der Startvorlage angegeben ist, kann sie nicht durch die Angabe einer AMI-ID in den `imageIdOverride` Parametern `imageId` oder ersetzt werden. Sie kann nur durch Angabe einer anderen Startvorlage ersetzt werden. Oder, wenn die Version der Startvorlage auf `$Default` oder festgelegt ist`$Latest`, indem Sie entweder eine neue Standardversion für die Startvorlage festlegen (falls vorhanden`$Default`) oder indem Sie der Startvorlage eine neue Version hinzufügen (falls vorhanden`$Latest`).

Wenn diese Regeln befolgt werden, führt jedes Update, das ein Infrastruktur-Update startet, dazu, dass die AMI-ID erneut ausgewählt wird. Wenn die `version`Einstellung in der

Startvorlage (`launchTemplate`) auf `$Latest` oder gesetzt ist`$Default`, wird die neueste Version oder Standardversion der Startvorlage zum Zeitpunkt des Infrastruktur-Updates ausgewertet, auch wenn sie nicht aktualisiert `launchTemplate`wurde.

Überlegungen bei der Erstellung parallel Jobs mit mehreren Knoten

AWS Batch empfiehlt, dedizierte Rechenumgebungen für die Ausführung von Multi-Node-Parallel-Jobs (MNP) und Nicht-MNP-Jobs zu erstellen. Dies liegt an der Art und Weise, wie Rechenkapazität in Ihrer verwalteten Computerumgebung geschaffen wird. Wenn Sie beim Erstellen einer neuen verwalteten Rechenumgebung einen `minvCpu` Wert größer als Null angeben, AWS Batch wird ein Instanzpool erstellt, der nur für Nicht-MNP-Jobs verwendet werden kann. Wenn ein parallel Auftrag mit mehreren Knoten eingereicht wird, wird neue Instanzkapazität für die Ausführung der parallel Jobs mit mehreren Knoten AWS Batch erstellt. In Fällen, in denen sowohl Einzelknoten- als auch Paralleljobs mit mehreren Knoten in derselben Rechenumgebung ausgeführt werden, in der entweder ein `minvCpus` oder `maxvCpus` festgelegt ist, AWS Batch wird, falls die erforderlichen Rechenressourcen nicht verfügbar sind, bis die aktuellen Jobs abgeschlossen sind, bevor die Rechenressourcen erstellt werden, die für die Ausführung der neuen Jobs erforderlich sind.

Nicht verwaltete Computerumgebungen

In einer nicht verwalteten Datenverarbeitungsumgebung verwalten Sie Ihre eigenen Datenverarbeitungsressourcen. Sie müssen überprüfen, ob das AMI, das Sie für Ihre Rechenressourcen verwenden, der AMI-Spezifikation für Amazon ECS-Container-Instances entspricht. Weitere Informationen finden Sie unter AMI-Spezifikation für Rechenressourcen und Erstellen eines Compute-Ressourcen-AMI.

> **Note**
>
> AWS Fargate-Ressourcen werden in nicht verwalteten Computerumgebungen nicht unterstützt.

Nachdem Sie Ihre nicht verwaltete Rechenumgebung erstellt haben, verwenden Sie den DescribeComputeEnvironmentsAPI-Vorgang, um die Details der Rechenumgebung anzuzeigen. Suchen Sie den Amazon ECS-Cluster, der mit der Umgebung verknüpft ist, und starten Sie dann Ihre Container-Instances manuell in diesem Amazon ECS-Cluster.

Der folgende AWS CLI Befehl stellt auch den Amazon ECS-Cluster-ARN bereit.

```
$ aws batch describe-compute-environments \
    --compute-environments unmanagedCE \
    --query "computeEnvironments[].ecsClusterArn"
```

Weitere Informationen finden Sie unter Starten einer Amazon ECS-Container-Instance im Amazon Elastic Container Service-Entwicklerhandbuch. Wenn Sie Ihre Rechenressourcen starten, geben Sie den Amazon ECS-Cluster-ARN an, den die Ressourcen mit den folgenden Amazon EC2 EC2-Benutzerdaten registrieren. *ecsClusterArn*Ersetzen Sie durch den Cluster-ARN, den Sie mit dem vorherigen Befehl erhalten haben.

```
#!/bin/bash
echo "ECS_CLUSTER=ecsClusterArn" >> /etc/ecs/ecs.config
```

Ressourcen-AMIs berechnen

Standardmäßig verwenden AWS Batch verwaltete Computerumgebungen eine aktuelle, genehmigte Version des für Amazon ECS optimierten AMI für Rechenressourcen. Möglicherweise möchten Sie jedoch Ihr eigenes AMI erstellen, das Sie für Ihre verwalteten und nicht verwalteten Computerumgebungen verwenden können. Wenn Sie eines der folgenden Dinge benötigen, empfehlen wir Ihnen, Ihr eigenes AMI zu erstellen:

- Erhöhung der Speichergröße Ihrer AMI-Root- oder Datenvolumes
- Hinzufügen von Instance-Speichervolumes für unterstützte Amazon EC2 EC2-Instance-Typen
- Anpassen des Amazon ECS-Container-Agenten
- Anpassen von Docker
- Konfiguration eines GPU-Workload-AMI, um Containern den Zugriff auf GPU-Hardware auf unterstützten Amazon EC2 EC2-Instance-Typen zu ermöglichen

> **Note**
>
> Nachdem eine Rechenumgebung erstellt wurde, werden die AMIs in der Rechenumgebung AWS Batch nicht aktualisiert. AWS Batch aktualisiert auch nicht die AMIs in Ihrer Rechenumgebung, wenn eine neuere Version des für Amazon ECS optimierten AMI verfügbar ist. Sie sind für die Verwaltung des Gastbetriebssystems verantwortlich.

Dazu gehören alle Updates und Sicherheitspatches. Sie sind auch für jede zusätzliche Anwendungssoftware oder Hilfsprogramme verantwortlich, die Sie auf den Rechenressourcen installieren. Gehen Sie wie folgt vor, um ein neues AMI für Ihre AWS Batch Jobs zu verwenden:

1. Erstellen Sie eine neue Datenverarbeitungsumgebung mit dem neuen AMI.
2. Fügen Sie die Datenverarbeitungsumgebung einer vorhandenen Auftragswarteschlange hinzu.
3. Entfernen Sie die alte Datenverarbeitungsumgebung aus Ihrer Auftragswarteschlange.
4. Löschen Sie die alte Datenverarbeitungsumgebung.

Im April 2022 AWS Batch wurde erweiterte Unterstützung für die Aktualisierung von Computerumgebungen hinzugefügt. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen. Beachten Sie die folgenden Regeln, um die erweiterte Aktualisierung von Computing-Umgebungen zum Aktualisieren von AMIs zu verwenden:

- Legen Sie entweder den Parameter service role (`serviceRole`) nicht fest oder legen Sie ihn auf die AWSServiceRoleForBatchdienstverknüpfte Rolle fest.
- Setzen Sie den Parameter Allocation Strategy (`allocationStrategy`) auf `BEST_FIT_PROGRESSIVESPOT_CAPACITY_OPTIMIZED`, oder`SPOT_PRICE_CAPACITY_OPTIMIZED`.
- Stellen Sie den Parameter Update auf die neueste Image-Version (`updateToLatestImageVersion`) auf ein`true`.
- Geben Sie keine AMI-ID in `imageId`, `imageIdOverride`(in `ec2Configuration`) oder in der Startvorlage (`launchTemplate`) an. Wenn Sie keine AMI-ID angeben, AWS Batch wählt das neueste Amazon ECS-optimierte AMI aus, das zum Zeitpunkt der Initiierung des Infrastruktur-Updates AWS Batch unterstützt wird. Alternativ können Sie die AMI-ID in den `imageIdOverride` Parametern `imageId` oder angeben. Sie können auch die Startvorlage angeben, die anhand der `LaunchTemplate` Eigenschaften identifiziert wird. Wenn Sie eine dieser Eigenschaften ändern, wird ein Infrastruktur-Update gestartet. Wenn die AMI-ID in der Startvorlage angegeben ist, kann die AMI-ID nicht durch die Angabe einer AMI-ID in den `imageIdOverride` Parametern `imageId` oder ersetzt werden. Die AMI-ID kann nur durch Angabe einer anderen Startvorlage ersetzt werden. Wenn die Version der Startvorlage auf `$Default` oder gesetzt ist`$Latest`, kann die AMI-ID ersetzt werden, indem entweder eine neue Standardversion für die Startvorlage festgelegt wird (if`$Default`) oder indem der Startvorlage eine neue Version hinzugefügt wird (if`$Latest`).

Wenn diese Regeln befolgt werden, führt jedes Update, das ein Infrastruktur-Update startet, dazu, dass die AMI-ID erneut ausgewählt wird. Wenn die `version`Einstellung in der Startvorlage (`launchTemplate`) auf `$Latest` oder gesetzt ist`$Default`, wird die neueste Version oder Standardversion der Startvorlage zum Zeitpunkt des Infrastruktur-Updates ausgewertet, auch wenn sie `launchTemplate`nicht aktualisiert wurde.

Themen

AMI-Spezifikation für Rechenressourcen

Die grundlegende AMI-Spezifikation für AWS Batch Rechenressourcen besteht aus den folgenden Komponenten:

Erforderlich

- Eine moderne Linux-Distribution, auf der mindestens Version 3.10 des Linux-Kernels auf einem AMI vom Typ HVM-Virtualisierung ausgeführt wird. Windows-Container werden nicht unterstützt.

  Important

  parallel Jobs mit mehreren Knoten können nur auf Rechenressourcen ausgeführt werden, die auf einer Amazon Linux-Instance mit dem installierten `ecs-init` Paket gestartet wurden. Wir empfehlen, dass Sie bei der Erstellung Ihrer Datenverarbeitungsumgebung das standardmäßige, für Amazon ECS optimierte AMI verwenden. Sie können dies tun, indem Sie kein benutzerdefiniertes AMI angeben. Weitere Informationen finden Sie unter parallel Jobs mit mehreren Knoten.

- Der Amazon ECS-Containeragent. Wir empfehlen, dass Sie die neueste Version verwenden. Weitere Informationen finden Sie unter Installation des Amazon ECS Container Agent im Amazon Elastic Container Service Developer Guide.

- Der `awslogs` Protokolltreiber muss als verfügbarer Protokolltreiber mit der `ECS_AVAILABLE_LOGGING_DRIVERS` Umgebungsvariablen angegeben werden, wenn der Amazon ECS-Container-Agent gestartet wird. Weitere Informationen finden Sie unter Amazon ECS Container Agent Configuration im Entwicklerhandbuch zum Amazon Elastic Container Service.
- Ein Docker-Daemon, auf dem mindestens Version 1.9 ausgeführt wird, und alle Abhängigkeiten von der Docker-Laufzeit. Weitere Informationen finden Sie unter Check runtime dependencies (Prüfen der Laufzeitabhängigkeiten) in der Docker-Dokumentation.

> **Note**
>
> Wir empfehlen die Docker-Version, die mit der entsprechenden Amazon ECS-Agentenversion, die Sie verwenden, geliefert wird und mit dieser getestet wurde. Amazon ECS stellt ein Changelog für die Linux-Variante des Amazon ECS-optimierten AMI on bereit. GitHub Weitere Informationen finden Sie unter Änderungsprotokoll.

Empfohlen

- Ein Initialisierungs- und Nanny-Prozess zum Ausführen und Überwachen des Amazon ECS-Agenten. Das für Amazon ECS optimierte AMI verwendet den `ecs-init` Upstart-Prozess, und andere Betriebssysteme verwenden `systemd` ihn möglicherweise. Weitere Informationen und Beispiele finden Sie unter Beispielskripte zur Konfiguration von Benutzerdaten für Container-Instances im Amazon Elastic Container Service Developer Guide. Weitere Informationen zu `ecs-init` finden Sie im `ecs-init`Projekt unter GitHub. Für verwaltete Rechenumgebungen muss der Amazon ECS-Agent mindestens beim Booten gestartet werden. Wenn der Amazon ECS-Agent nicht auf Ihrer Rechenressource läuft, kann er keine Jobs von annehmen AWS Batch.

Das für Amazon ECS optimierte AMI ist mit diesen Anforderungen und Empfehlungen vorkonfiguriert. Wir empfehlen, dass Sie das für Amazon ECS optimierte AMI oder ein Amazon Linux AMI mit dem `ecs-init` Paket verwenden, das für Ihre Rechenressourcen installiert ist. Wählen Sie ein anderes AMI, wenn Ihre Anwendung ein bestimmtes Betriebssystem oder eine Docker-Version benötigt, die in diesen AMIs noch nicht verfügbar ist. Weitere Informationen finden Sie unter Amazon ECS-Optimized AMI im Amazon Elastic Container Service Developer Guide.

Erstellen eines Compute-Ressourcen-AMI

Sie können Ihr eigenes benutzerdefiniertes Compute-Ressourcen-AMI erstellen, das Sie für Ihre verwalteten und nicht verwalteten Computerumgebungen verwenden können. Anweisungen hierzu finden Sie im AMI-Spezifikation für Rechenressourcen. Nachdem Sie ein benutzerdefiniertes AMI erstellt haben, können Sie eine Rechenumgebung erstellen, die dieses AMI verwendet, dem Sie eine Auftragswarteschlange zuordnen können. Beginnen Sie zuletzt damit, Jobs an diese Warteschlange zu senden.

So erstellen Sie ein benutzerdefiniertes Compute-Ressourcen-AMI

1. Wählen Sie ein Basis-AMI aus, von dem aus Sie beginnen möchten. Das Basis-AMI muss HVM-Virtualisierung verwenden. Das Basis-AMI kann kein Windows-AMI sein.

   > **Note**
   >
   > Das AMI, das Sie für eine Rechenumgebung auswählen, muss der Architektur der Instance-Typen entsprechen, die Sie für diese Rechenumgebung verwenden möchten. Wenn Ihre Rechenumgebung beispielsweise A1 Instanztypen verwendet, muss das von Ihnen gewählte Compute-Ressourcen-AMI Arm Instances unterstützen. Amazon ECS verkauft x86 sowohl Arm Versionen als auch Versionen des für Amazon ECS optimierten Amazon Linux 2-AMI. Weitere Informationen finden Sie unter Amazon ECS-optimiertes Amazon Linux 2-AMI im Amazon Elastic Container Service Developer Guide.

   Das für Amazon ECS optimierte Amazon Linux 2-AMI ist das Standard-AMI für Rechenressourcen in verwalteten Computerumgebungen. Das für Amazon ECS optimierte Amazon Linux 2-AMI ist vorkonfiguriert und wurde AWS Batch von AWS Technikern getestet. Es handelt sich um ein minimales AMI, mit dem Sie beginnen können und mit dem Sie Ihre Rechenressourcen AWS schnell zum Laufen bringen können. Weitere Informationen finden Sie unter Amazon ECS Optimized AMI im Amazon Elastic Container Service Developer Guide.

   Alternativ können Sie eine andere Amazon Linux 2-Variante wählen und das `ecs-init` Paket mit den folgenden Befehlen installieren. Weitere Informationen finden Sie unter Installation des Amazon ECS-Container-Agenten auf einer Amazon Linux 2 EC2-Instance im Amazon Elastic Container Service Developer Guide:

   ```
   $ sudo amazon-linux-extras disable docker
   ```

```
$ sudo amazon-linux-extras install ecs-init
```

Wenn Sie beispielsweise GPU-Workloads auf Ihren AWS Batch Rechenressourcen ausführen möchten, können Sie mit dem Amazon Linux Deep Learning AMI beginnen. Konfigurieren Sie dann das AMI für die Ausführung von AWS Batch Jobs. Weitere Informationen finden Sie unter Verwenden eines GPU-Workload-AMI.

> **Important**
>
> Sie können ein Basis-AMI wählen, das das `ecs-init` Paket nicht unterstützt. In diesem Fall müssen Sie jedoch eine Methode konfigurieren, mit der der Amazon ECS-Agent beim Booten gestartet und weiter ausgeführt werden kann. Sie können sich auch mehrere Beispielskripts zur Konfiguration von Benutzerdaten ansehen, die `systemd` zum Starten und Überwachen des Amazon ECS-Container-Agenten verwendet werden. Weitere Informationen finden Sie unter Beispiel für Konfigurationsskripte für Container-Instance-Benutzerdaten im Amazon Elastic Container Service Developer Guide.

2. Starten Sie eine Instance von Ihrem ausgewählten Basis-AMI aus mit den entsprechenden Speicheroptionen für Ihr AMI. Sie können die Größe und Anzahl der angehängten Amazon EBS-Volumes oder Instance-Speicher-Volumes konfigurieren, sofern der von Ihnen gewählte Instance-Typ diese unterstützt. Weitere Informationen finden Sie unter Launching an Instance und Amazon EC2 Instance Store im Amazon EC2 EC2-Benutzerhandbuch.
3. Connect zu Ihrer Instance mit her SSH und führen Sie alle erforderlichen Konfigurationsaufgaben durch. Dies kann einen oder alle der folgenden Schritte beinhalten:

   - Installation des Amazon ECS-Container-Agenten. Weitere Informationen finden Sie unter Installation des Amazon ECS Container Agent im Amazon Elastic Container Service Developer Guide.
   - Konfigurieren Sie ein Skript zum Formatieren von Instance-Speicher-Volumes.
   - Hinzufügen von Instance-Speicher-Volume- oder Amazon EFS-Dateisystemen zur `/etc/fstab` Datei, sodass sie beim Booten gemountet werden.
   - Konfiguration von Docker-Optionen, z. B. das Aktivieren des Debuggens oder das Anpassen der Basis-Image-Größe.
   - Installieren Sie Pakete oder kopieren Sie Dateien.

Weitere Informationen finden Sie unter Herstellen einer Verbindung zu Ihrer Linux-Instance mithilfe von SSH im Amazon EC2 EC2-Benutzerhandbuch.

4. Wenn Sie den Amazon ECS-Container-Agenten auf Ihrer Instance gestartet haben, müssen Sie ihn beenden und alle persistenten Daten-Checkpoint-Dateien entfernen, bevor Sie Ihr AMI erstellen. Andernfalls, wenn Sie dies nicht tun, startet der Agent nicht auf Instances, die von Ihrem AMI aus gestartet werden.

   a. Halten Sie den Amazon-ECS-Container-Agent an.

      - Amazon-ECS-optimiertes Amazon Linux 2-AMI:

        ```
        sudo systemctl stop ecs
        ```

      - Amazon-ECS-optimiertes Amazon Linux AMI:

        ```
        sudo stop ecs
        ```

   b. Entfernen Sie die Checkpoint-Dateien für persistente Daten. Standardmäßig befinden sich diese Dateien im `/var/lib/ecs/data/` Verzeichnis. Verwenden Sie den folgenden Befehl, um diese Dateien zu entfernen, falls vorhanden.

      ```
      sudo rm -rf /var/lib/ecs/data/*
      ```

5. Erstellen Sie ein neues AMI aus Ihrer laufenden Instance. Weitere Informationen finden Sie unter Creating an Amazon EBS Backed Linux AMI im Amazon EC2 EC2-Benutzerhandbuch.

Um Ihr neues AMI zu verwenden mit AWS Batch

1. Nachdem das neue AMI erstellt wurde, erstellen Sie eine Rechenumgebung mit dem neuen AMI. Um das zu tun, wählen Sie den Image-Typ und geben Sie die benutzerdefinierte AMI-ID in die Image-ID ein Überschreiben Sie das Feld, wenn Sie die AWS Batch Rechenumgebung erstellen. Für weitere Informationen siehethe section called “Um eine verwaltete Rechenumgebung mithilfe von EC2-Ressourcen zu erstellen”.

> **Note**
>
> Das AMI, das Sie für eine Rechenumgebung auswählen, muss der Architektur der Instance-Typen entsprechen, die Sie für diese Rechenumgebung verwenden möchten. Wenn Ihre Rechenumgebung beispielsweise A1 Instanztypen verwendet, muss das von Ihnen gewählte Compute-Ressourcen-AMI Arm Instances unterstützen. Amazon ECS verkauft x86 sowohl Arm Versionen als auch Versionen des für Amazon ECS optimierten Amazon Linux 2-AMI. Weitere Informationen finden Sie unter Amazon ECS-optimiertes Amazon Linux 2-AMI im Amazon Elastic Container Service Developer Guide.

2. Erstellen Sie eine Auftragswarteschlange und verknüpfen Sie Ihre neue Datenverarbeitungsumgebung. Weitere Informationen finden Sie unter Erstellen einer Auftragswarteschlange.

> **Note**
>
> Alle Rechenumgebungen, die mit einer Job-Warteschlange verknüpft sind, müssen dieselbe Architektur haben. AWS Batch unterstützt nicht das Mischen von Architekturtypen für Rechenumgebungen in einer einzigen Jobwarteschlange.

3. (Optional) Übermitteln Sie einen Beispielauftrag an die neue Auftragswarteschlange. Weitere Informationen finden Sie unter Beispiel für Auftragsdefinitionen, Erstellen einer Auftragsdefinition mit einem Knoten und Einen Job einreichen.

Verwenden eines GPU-Workload-AMI

Um GPU-Workloads auf Ihren AWS Batch-Datenverarbeitungsressourcen ausführen zu können, müssen Sie ein AMI mit GPU-Unterstützung verwenden. Weitere Informationen finden Sie unter Arbeiten mit GPUs auf Amazon ECS und Amazon-ECS-optimierten AMIs im Amazon Elastic Container Service-Entwicklerhandbuch.

Wenn die Datenverarbeitungsumgebung in verwalteten Datenverarbeitungsumgebungen Instance-`g5`Typen oder Instance-Familien `p3p2`, `g3s`, `p5`, `p4g3`, `g4`, oder angibt, AWS Batch verwendet ein für Amazon ECS GPU optimiertes AMI.

In nicht verwalteten Datenverarbeitungsumgebungen wird ein Amazon-ECS-GPU-optimiertes AMI empfohlen. Sie können die GetParametersByPath Operationen AWS Command Line Interface oder

AWS Systems Manager Parameter Store GetParameter, und verwendenGetParameters, um die Metadaten für die empfohlenen Amazon-ECS-GPU-optimierten AMIs abzurufen.

> **Note**
>
> Die p5 Instance-Familie wird nur auf Versionen unterstützt, die gleich oder höher als `20230912` des Amazon-ECS-GPU-optimierten AMI sind, und sie sind mit den g2 Instance-Typen p2 und nicht kompatibel. Wenn Sie p5 Instances verwenden müssen, stellen Sie sicher, dass Ihre Datenverarbeitungsumgebung keine - p2 oder -g2Instances enthält und das neueste Standard-Batch-AMI verwendet. Das Erstellen einer neuen Computing-Umgebung verwendet das neueste AMI. Wenn Sie Ihre Computing-Umgebung jedoch aktualisieren, um einzuschließenp5, können Sie sicherstellen, dass Sie das neueste AMI verwenden, indem Sie `true` in `ComputeResource` Eigenschaften `updateToLatestImageVersion` auf setzen. Weitere Informationen zur AMI-Kompatibilität mit GPU-Instances finden Sie unter Arbeiten mit GPUs auf Amazon ECS im Amazon Elastic Container Service-Entwicklerhandbuch.

Die folgenden Beispiele demonstrieren die Verwendung des GetParameter-Befehls.

AWS CLI

```
$ aws ssm get-parameter --name /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended \
                        --region us-east-2 --output json
```

Die Ausgabe enthält die AMI-Informationen im -`Value`Parameter.

```
{
    "Parameter": {
        "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended",
        "LastModifiedDate": 1555434128.664,
        "Value": "{\"schema_version\":1,\"image_name\":\"amzn2-ami-ecs-gpu-
hvm-2.0.20190402-x86_64-ebs\",\"image_id\":\"ami-083c800fe4211192f\",\"os\":\"Amazon
 Linux 2\",\"ecs_runtime_version\":\"Docker version 18.06.1-ce\",\"ecs_agent_version
\":\"1.27.0\"}",
        "Version": 9,
        "Type": "String",
        "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended"
```

```
    }
}
```

Python

```
from __future__ import print_function

import json
import boto3

ssm = boto3.client('ssm', 'us-east-2')

response = ssm.get_parameter(Name='/aws/service/ecs/optimized-ami/amazon-linux-2/
gpu/recommended')
jsonVal = json.loads(response['Parameter']['Value'])
print("image_id   = " + jsonVal['image_id'])
print("image_name = " + jsonVal['image_name'])
```

Die Ausgabe umfasst nur die AMI-ID und den AMI-Namen:

```
image_id   = ami-083c800fe4211192f
image_name = amzn2-ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs
```

Die folgenden Beispiele veranschaulichen die Verwendung von GetParameters.

AWS CLI

```
$ aws ssm get-parameters --names  /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended/image_name \
                                  /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended/image_id \
                          --region us-east-2 --output json
```

Die Ausgabe enthält die vollständigen Metadaten für die einzelnen Parameter:

```
{
    "InvalidParameters": [],
    "Parameters": [
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_id",
```

```
            "LastModifiedDate": 1555434128.749,
            "Value": "ami-083c800fe4211192f",
            "Version": 9,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/image_id"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_name",
            "LastModifiedDate": 1555434128.712,
            "Value": "amzn2-ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs",
            "Version": 9,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/image_name"
        }
    ]
}
```

Python

```
from __future__ import print_function

import boto3

ssm = boto3.client('ssm', 'us-east-2')

response = ssm.get_parameters(
            Names=['/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_name',
                   '/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_id'])
for parameter in response['Parameters']:
    print(parameter['Name'] + " = " + parameter['Value'])
```

Die Ausgabe enthält die AMI-ID und den AMI-Namen unter Verwendung des vollständigen Pfads für die Namen.

```
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/image_id =
 ami-083c800fe4211192f
```

```
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/image_name = amzn2-
ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs
```

Die folgenden Beispiele demonstrieren die Verwendung des GetParametersByPath-Befehls.

AWS CLI

```
$ aws ssm get-parameters-by-path --path /aws/service/ecs/optimized-ami/amazon-
linux-2/gpu/recommended \
                                --region us-east-2 --output json
```

Die Ausgabe enthält die vollständigen Metadaten für alle Parameter unter dem angegebenen Pfad.

```
{
    "Parameters": [
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
ecs_agent_version",
            "LastModifiedDate": 1555434128.801,
            "Value": "1.27.0",
            "Version": 8,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/ecs_agent_version"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
ecs_runtime_version",
            "LastModifiedDate": 1548368308.213,
            "Value": "Docker version 18.06.1-ce",
            "Version": 1,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/ecs_runtime_version"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_id",
            "LastModifiedDate": 1555434128.749,
            "Value": "ami-083c800fe4211192f",
            "Version": 9,
```

```
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/image_id"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_name",
            "LastModifiedDate": 1555434128.712,
            "Value": "amzn2-ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs",
            "Version": 9,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/image_name"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
os",
            "LastModifiedDate": 1548368308.143,
            "Value": "Amazon Linux 2",
            "Version": 1,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/os"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
schema_version",
            "LastModifiedDate": 1548368307.914,
            "Value": "1",
            "Version": 1,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/schema_version"
        }
    ]
}
```

Python

```
from __future__ import print_function

import boto3
```

```
ssm = boto3.client('ssm', 'us-east-2')

response = ssm.get_parameters_by_path(Path='/aws/service/ecs/optimized-ami/amazon-
linux-2/gpu/recommended')
for parameter in response['Parameters']:
    print(parameter['Name'] + " = " + parameter['Value'])
```

Die Ausgabe enthält die Werte aller Parameternamen im angegebenen Pfad, wobei der vollständige Pfad für die Namen verwendet wird.

```
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/ecs_agent_version =
 1.27.0
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/ecs_runtime_version =
 Docker version 18.06.1-ce
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/image_id =
 ami-083c800fe4211192f
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/image_name = amzn2-
ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/os = Amazon Linux 2
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/schema_version = 1
```

Weitere Informationen finden Sie unter Abrufen von Amazon-ECS-optimierten AMI-Metadaten im Amazon Elastic Container Service-Entwicklerhandbuch.

Veraltete Amazon Linux-Version

Das Amazon Linux AMI (auch Amazon Linux 1 genannt) hat am 31. Dezember 2023 sein Lebensende erreicht. AWS Batch hat den Support für Amazon Linux AMI eingestellt, da es ab dem 1. Januar 2024 keine Sicherheitsupdates oder Bugfixes erhalten wird. Weitere Informationen zu Amazon Linux end-of-life finden Sie in den häufig gestellten Fragen zu AL.

Wir empfehlen, bestehende Amazon Linux-basierte Rechenumgebungen auf Amazon Linux 2023 zu aktualisieren, um unvorhergesehene Workload-Unterbrechungen zu vermeiden und weiterhin Sicherheits- und andere Updates zu erhalten.

Ihre Rechenumgebungen, die das Amazon Linux AMI verwenden, funktionieren möglicherweise auch nach dem 31. Dezember 2023 end-of-life weiter. Diese Computerumgebungen erhalten jedoch keine neuen Softwareupdates, Sicherheitspatches oder Bugfixes mehr von AWS. Es liegt in Ihrer Verantwortung, diese Rechenumgebungen anschließend auf dem Amazon Linux AMI

aufrechtzuerhalten end-of-life. Wir empfehlen, AWS Batch Rechenumgebungen auf Amazon Linux 2023 oder Amazon Linux 2 zu migrieren, um optimale Leistung und Sicherheit zu gewährleisten.

Hilfe bei der Migration AWS Batch vom Amazon Linux AMI zu Amazon Linux 2023 oder Amazon Linux 2 finden Sie unter Computerumgebungen aktualisieren — AWS Batch.

Support für Startvorlagen

AWS Batch unterstützt die Verwendung von Amazon EC2 EC2-Startvorlagen in Ihren EC2-Rechenumgebungen. Mit Startvorlagen können Sie die Standardkonfiguration Ihrer AWS Batch Rechenressourcen ändern, ohne benutzerdefinierte AMIs erstellen zu müssen.

> **Note**
>
> Startvorlagen werden auf AWS Fargate-Ressourcen nicht unterstützt.

Sie müssen eine Startvorlage erstellen, bevor Sie sie mit einer Datenverarbeitungsumgebung verknüpfen können. Sie können eine Startvorlage in der Amazon EC2 EC2-Konsole erstellen. Sie können auch das AWS CLI oder ein AWS SDK verwenden. Die folgende JSON-Datei stellt beispielsweise eine Startvorlage dar, die die Größe des Docker-Datenvolumes für das AWS Batch Standard-Compute-Ressourcen-AMI ändert und es auch so festlegt, dass es verschlüsselt wird.

```
{
    "LaunchTemplateName": "increase-container-volume-encrypt",
    "LaunchTemplateData": {
        "BlockDeviceMappings": [
            {
                "DeviceName": "/dev/xvda",
                "Ebs": {
                    "Encrypted": true,
                    "VolumeSize": 100,
                    "VolumeType": "gp2"
                }
            }
        ]
    }
}
```

Sie können die vorherige Startvorlage erstellen, indem Sie die JSON-Datei in einer Datei speichern, die aufgerufen wird, `lt-data.json` und den folgenden AWS CLI Befehl ausführen.

```
aws ec2 --region <region> create-launch-template --cli-input-json file://lt-data.json
```

Weitere Informationen zu Startvorlagen finden Sie unter Launching an Instance from a Launch Template im Amazon EC2 EC2-Benutzerhandbuch.

Wenn Sie eine Startvorlage verwenden, um Ihre Datenverarbeitungsumgebung zu erstellen, können Sie die folgenden vorhandenen Parameter der Datenverarbeitungsumgebung in Ihre Startvorlage verschieben:

> **Note**
>
> Nehmen wir an, dass einer dieser Parameter (mit Ausnahme der Amazon EC2-Tags) sowohl in der Startvorlage als auch in der Konfiguration der Rechenumgebung angegeben ist. Dann haben die Parameter der Rechenumgebung Vorrang. Amazon EC2-Tags werden zwischen der Startvorlage und der Konfiguration der Rechenumgebung zusammengeführt. Wenn der Schlüssel des Tags kollidiert, hat der Wert in der Konfiguration der Rechenumgebung Vorrang.

- Amazon EC2 EC2-Schlüsselpaar
- Amazon EC2 EC2-AMI-ID
- Sicherheitsgruppen-IDs
- Amazon EC2-Tags

Die folgenden Parameter für Startvorlagen werden von AWS Batch ignoriert:

- Instance-Typ (Legen Sie beim Erstellen Ihrer Datenverarbeitungsumgebung die gewünschten Instance-Typen fest.)
- Instance-Rolle (Legen Sie beim Erstellen Ihrer Datenverarbeitungsumgebung die gewünschte Instance-Rolle fest.)
- Netzwerkschnittstellen-Subnetze (Legen Sie beim Erstellen Ihrer Datenverarbeitungsumgebung die gewünschten Subnetze fest.)
- Optionen für den Instance-Markt (AWS Batch muss die Spot-Instance-Konfiguration steuern)

- Deaktivieren Sie die API-Kündigung (AWS Batch muss den Instance-Lebenszyklus kontrollieren)

AWS Batch aktualisiert die Startvorlage nur bei Infrastruktur-Updates mit einer neuen Version der Startvorlage. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen.

Amazon EC2 EC2-Benutzerdaten in Startvorlagen

Sie können Amazon EC2 EC2-Benutzerdaten in Ihrer Startvorlage angeben, die von cloud-init ausgeführt wird, wenn Ihre Instances gestartet werden. Mit Ihren Benutzerdaten können gängige Konfigurationsszenarien durchgeführt werden, einschließlich, aber nicht beschränkt auf die folgenden:

- Einschließen von Benutzern oder Gruppen
- Installieren von Paketen
- Erstellen von Partitionen und Dateisystemen

Amazon EC2 EC2-Benutzerdaten in Startvorlagen müssen im mehrteiligen MIME-Archivformat vorliegen. Das liegt daran, dass Ihre Benutzerdaten mit anderen AWS Batch Benutzerdaten zusammengeführt werden, die für die Konfiguration Ihrer Rechenressourcen erforderlich sind. Sie können mehrere Benutzerdatenblöcke in einer einzelnen mehrteiligen MIME-Datei kombinieren. Beispielsweise möchten Sie vielleicht einen Cloud-Boothook, der den Docker-Daemon konfiguriert, mit einem Benutzerdaten-Shell-Skript kombinieren, das Konfigurationsinformationen für den Amazon ECS-Container-Agenten schreibt.

Wenn Sie den AWS::CloudFormation::InitTyp verwenden AWS CloudFormation, kann er zusammen mit dem Hilfsskript cfn-init verwendet werden, um allgemeine Konfigurationsszenarien durchzuführen.

Eine mehrteilige MIME-Datei umfasst folgende Komponenten:

- Deklaration von Inhaltstyp und Teilgrenze: `Content-Type: multipart/mixed; boundary="==BOUNDARY=="`
- Deklaration der MIME-Version: `MIME-Version: 1.0`
- Ein oder mehrere Benutzerdatenblöcke, die die folgenden Komponenten enthalten:
  - Die Öffnungsgrenze, die den Beginn eines Benutzerdatenblocks signalisiert:`--==BOUNDARY==`. Sie müssen die Zeile vor dieser Grenze leer lassen.

  - Die Inhaltstypdeklaration für den Block: `Content-Type: text/cloud-config; charset="us-ascii"`. Weitere Informationen zu Inhaltstypen finden Sie in der Cloud-Init-Dokumentation. Sie müssen die Zeile nach der Inhaltstyp-Deklaration leer lassen.
  - Der Inhalt der Benutzerdaten, z. B. eine Liste von Shell-Befehlen oder `cloud-init` -Direktiven.
- Die schließende Grenze, die das Ende der mehrteiligen MIME-Datei signalisiert:`--==BOUNDARY==--`. Sie müssen die Zeile vor der schließenden Grenze leer lassen.

Im Folgenden finden Sie mehrteilige MIME-Beispieldateien, mit denen Sie Ihre eigenen erstellen können.

> **Note**
>
> Wenn Sie Benutzerdaten zu einer Startvorlage in der Amazon EC2 EC2-Konsole hinzufügen, können Sie sie als Klartext einfügen. Oder Sie können es aus einer Datei hochladen. Wenn Sie das AWS CLI oder ein AWS SDK verwenden, müssen Sie zuerst die Benutzerdaten `base64` codieren und diese Zeichenfolge als Wert des `UserData` Parameters angeben, wenn Sie CreateLaunchTemplate aufrufen, wie in dieser JSON-Datei gezeigt.
>
> ```
> {
>     "LaunchTemplateName": "base64-user-data",
>     "LaunchTemplateData": {
>         "UserData":
>  "ewogICAgIkxhdW5jaFRlbXBsYXRlTmFtZSI6ICJpbmNyZWFzZS1jb250YWluZXItdm9sdW..."
>     }
> }
> ```

Beispiele

- Beispiel: Mounten Sie ein vorhandenes Amazon EFS-Dateisystem
- Beispiel: Standardkonfiguration des Amazon ECS-Container-Agenten überschreiben
- Beispiel: Mounten Sie ein vorhandenes Amazon FSx for Lustre-Dateisystem

Beispiel: Mounten Sie ein vorhandenes Amazon EFS-Dateisystem

Example

Diese mehrteilige MIME-Beispieldatei konfiguriert die Rechenressource für die Installation des `amazon-efs-utils` Pakets und das Mounten eines vorhandenen Amazon EFS-Dateisystems unter. `/mnt/efs`

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
- amazon-efs-utils

runcmd:
- file_system_id_01=fs-abcdef123
- efs_directory=/mnt/efs

- mkdir -p ${efs_directory}
- echo "${file_system_id_01}:/ ${efs_directory} efs tls,_netdev" >> /etc/fstab
- mount -a -t efs defaults

--==MYBOUNDARY==--
```

Beispiel: Standardkonfiguration des Amazon ECS-Container-Agenten überschreiben

Example

Diese mehrteilige MIME-Beispieldatei überschreibt die standardmäßigen Cleanup-Einstellungen des Docker-Images für eine Datenverarbeitungsressource.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
echo ECS_IMAGE_CLEANUP_INTERVAL=60m >> /etc/ecs/ecs.config
```

```
echo ECS_IMAGE_MINIMUM_CLEANUP_AGE=60m >> /etc/ecs/ecs.config

--==MYBOUNDARY==--
```

Beispiel: Mounten Sie ein vorhandenes Amazon FSx for Lustre-Dateisystem

Example

Diese mehrteilige MIME-Beispieldatei konfiguriert die Rechenressource so, dass sie das `lustre2.10` Paket aus der Extras-Bibliothek installiert und ein vorhandenes FSx for Lustre-Dateisystem unter `/scratch` und dem Mount-Namen einhängt. `fsx` Dieses Beispiel bezieht sich auf Amazon Linux 2. Installationsanweisungen für andere Linux-Distributionen finden Sie unter Installation des Lustre-Clients im Amazon FSx for Lustre-Benutzerhandbuch. Weitere Informationen finden Sie unter Automatisches Mounten Ihres Amazon FSx-Dateisystems im Amazon FSx for Lustre-Benutzerhandbuch.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

runcmd:
- file_system_id_01=fs-0abcdef1234567890
- region=us-east-2
- fsx_directory=/scratch
- amazon-linux-extras install -y lustre2.10
- mkdir -p ${fsx_directory}
- mount -t lustre ${file_system_id_01}.fsx.${region}.amazonaws.com@tcp:fsx
 ${fsx_directory}

--==MYBOUNDARY==--
```

Bei den Mitgliedern volumes und mountPoints der Containereigenschaften müssen die Einhängepunkte dem Container zugeordnet sein.

```
{
    "volumes": [
        {
            "host": {
                "sourcePath": "/scratch"
```

```
            },
            "name": "Scratch"
        }
    ],
    "mountPoints": [
        {
            "containerPath": "/scratch",
            "sourceVolume": "Scratch"
        }
    ],
}
```

Eine Rechenumgebung erstellen

Bevor Sie Jobs ausführen können AWS Batch, müssen Sie eine Rechenumgebung erstellen. Sie können eine verwaltete Rechenumgebung erstellen, in der die Amazon EC2 EC2-Instances oder AWS Fargate-Ressourcen innerhalb der Umgebung auf der Grundlage Ihrer Spezifikationen AWS Batch verwaltet werden. Alternativ können Sie auch eine nicht verwaltete Rechenumgebung erstellen, in der Sie die Amazon EC2 EC2-Instance-Konfiguration innerhalb der Umgebung verwalten.

Important

Fargate Spot-Instances werden in den folgenden Szenarien nicht unterstützt:

- Auf Amazon Linux-Containern mit ARM64-Architektur.
- Windows containers on AWS Fargate

In diesen Szenarien wird eine Job-Warteschlange blockiert, wenn ein Job an eine Job-Warteschlange weitergeleitet wird, die nur Fargate Spot-Computerumgebungen verwendet.

Inhalt

Um eine verwaltete Rechenumgebung mit AWS Fargate-Ressourcen zu erstellen

1. Öffnen Sie die AWS Batch Konsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie in der Navigationsleiste die aus, die Sie verwenden AWS-Region möchten.
3. Wählen Sie im Navigationsbereich Datenverarbeitungs-Umgebungen aus.
4. Wählen Sie Erstellen.
5. Konfigurieren Sie die Rechenumgebung.

   > **Note**
   >
   > Rechenumgebungen für Windows containers on AWS Fargate Jobs müssen mindestens eine vCPU haben.

   a. Wählen Sie für die Konfiguration der Rechenumgebung Fargate aus.
   b. Geben Sie unter Name einen eindeutigen Namen für Ihre Rechenumgebung an. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
   c. Wählen Sie unter Servicerolle die dienstbezogene Rolle aus, mit der der AWS Batch Dienst die erforderlichen AWS API-Operationen in Ihrem Namen aufrufen kann. Wählen Sie zum Beispiel aus AWSServiceRoleForBatch. Weitere Informationen finden Sie unter Dienstbezogene Rollenberechtigungen für AWS Batch.
   d. (Optional) Erweitern Sie Tags. Um einen Tag hinzuzufügen, wählen Sie Add tag (Tag hinzufügen). Geben Sie dann einen Schlüsselnamen und optional einen Wert ein. Wählen Sie Add tag.
   e. Wählen Sie „Nächste Seite“.
6. Gehen Sie im Abschnitt Instanzkonfiguration wie folgt vor:

   a. (Optional) Aktivieren Sie Fargate Spot, um Fargate Spot-Kapazität verwenden zu können. Informationen zu Fargate Spot finden Sie unter Amazon EC2 Spot verwenden und Fargate_SPOT.
   b. Wählen Sie unter Maximum vCPUs die maximale Anzahl von vCPUs aus, auf die Ihre Rechenumgebung skaliert werden kann, unabhängig von der Nachfrage in der Job-Warteschlange.

c. Wählen Sie „Nächste Seite“.

7. Konfigurieren Sie das Netzwerk.

> **Important**
>
> Datenverarbeitungsressourcen benötigen einen externen Netzwerkzugriff, um mit dem Amazon-ECS-Service-Endpunkt zu kommunizieren. Dies kann über einen Schnittstellen-VPC-Endpunkt oder über Ihre Datenverarbeitungsressourcen mit öffentlichen IP-Adressen sein.
> Weitere Informationen zu Interface-VPC-Endpunkten finden Sie unter Amazon-ECS-Schnittstellen-VPC-Endpunkte (AWS PrivateLink) im Amazon Elastic Container Service-Entwicklerhandbuch.
> Wenn Sie keinen Schnittstellen-VPC-Endpunkt konfiguriert haben und Ihre Datenverarbeitungsressourcen keine öffentlichen IP-Adressen haben, müssen Sie Netzwerkadressübersetzung (Network Address Translation, NAT) verwenden, um diesen Zugriff zur Verfügung zu stellen. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch. Weitere Informationen finden Sie unter the section called “Erstellen einer VPC”.

a. Wählen Sie für Virtual Private Cloud (VPC) ID eine VPC aus, in der Sie Ihre Instances starten möchten.

b. Wählen Sie für Subnetze die zu verwendenden Subnetze aus. Standardmäßig sind alle Subnetze innerhalb der ausgewählten VPC verfügbar.

> **Note**
>
> AWS Batch on Fargate unterstützt derzeit keine Local Zones. Weitere Informationen finden Sie unter Amazon ECS-Cluster in Local Zones, Wavelength Zones und AWS Outposts im Amazon Elastic Container Service Developer Guide.

c. Wählen Sie für Sicherheitsgruppen eine Sicherheitsgruppe aus, um sie Ihren Instances anzufügen. Standardmäßig ist die Standardsicherheitsgruppe für Ihre VPC ausgewählt.

d. Wählen Sie „Nächste Seite“.

8. Überprüfen Sie die Konfigurationsschritte zur Überprüfung. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Create Compute Environment aus.

Um eine verwaltete Rechenumgebung mithilfe von EC2-Ressourcen zu erstellen

1. Öffnen Sie die AWS Batch Konsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie in der Navigationsleiste die aus, die Sie verwenden AWS-Region möchten.
3. Wählen Sie im Navigationsbereich Datenverarbeitungs-Umgebungen aus.
4. Wählen Sie Erstellen.
5. Konfigurieren Sie die Umgebung.

    a. Wählen Sie für die Konfiguration der Rechenumgebung Amazon Elastic Compute Cloud (Amazon EC2).

    b. Wählen Sie als Orchestrierungstyp die Option Verwaltet aus.

    c. Geben Sie unter Name einen eindeutigen Namen für Ihre Rechenumgebung an. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.

    d. (Optional) Wählen Sie für die Servicerolle die serviceverknüpfte Rolle aus, mit der der AWS Batch Dienst in Ihrem Namen die erforderlichen AWS API-Operationen aufrufen kann. Wählen Sie zum Beispiel aus AWSServiceRoleForBatch. Weitere Informationen finden Sie unter Dienstbezogene Rollenberechtigungen für AWS Batch.

    e. Wählen Sie im Feld Instance-Role (Instance-Rolle) ein neues Instance-Profil aus oder verwenden Sie ein bestehendes Instance-Profil, bei dem die erforderlichen IAM-Berechtigungen angefügt sind. Dieses Instance-Profil ermöglicht es den Amazon ECS-Container-Instances, die für Ihre Rechenumgebung erstellt wurden, die erforderlichen AWS API-Operationen in Ihrem Namen aufzurufen. Weitere Informationen finden Sie unter Amazon-ECS-Instance-Rolle. Wenn Sie ein neues Instance-Profil erstellen, wird die erforderliche Rolle (`ecsInstanceRole`) für Sie erstellt.

    f. (Optional) Erweitern Sie Tags.

    g. (Optional) Wählen Sie für EC2-Tags Tag hinzufügen aus, um Ressourcen, die in der Rechenumgebung gestartet werden, ein Tag hinzuzufügen. Geben Sie dann einen Schlüsselnamen und optional einen Wert ein. Wählen Sie Add tag.

h. (Optional) Wählen Sie für Tags die Option Tag hinzufügen aus. Geben Sie dann einen Schlüsselnamen und optional einen Wert ein. Wählen Sie Add tag.

   Weitere Informationen finden Sie unter Markieren Ihrer AWS Batch-Ressourcen.

i. Wählen Sie „Nächste Seite“.

6. Gehen Sie im Abschnitt Instanzkonfiguration wie folgt vor:

   a. (Optional) Aktivieren Sie unter Verwendung von Spot-Instances die Option Spot. Weitere Informationen finden Sie unter Spot-Instances.

   b. (Nur Spot) Wählen Sie für den On-Demand-Preis von maximal% den maximalen Prozentsatz aus, den ein Spot-Instance-Preis im Vergleich zum On-Demand-Preis für diesen Instance-Typ vor dem Start der Instances betragen kann. Wenn Ihr Höchstpreis beispielsweise 20% beträgt, muss der Spot-Preis weniger als 20% des aktuellen On-Demand-Preises für diese EC2-Instance betragen. Sie zahlen immer nur den niedrigsten (Markt-) Preis und niemals mehr als Ihren maximalen Prozentsatz. Wenn Sie dieses Feld leer lassen, ist der Standardwert 100 % des On-Demand-Preises.

   c. (Nur Spot) Wählen Sie für die Spot-Flottenrolle eine bestehende Amazon EC2 Spot Fleet IAM-Rolle aus, die Sie auf Ihre Spot-Computerumgebung anwenden möchten. Wenn Sie noch keine bestehende Amazon EC2 Spot Fleet IAM-Rolle haben, müssen Sie zuerst eine erstellen. Weitere Informationen finden Sie unter Amazon EC2-Spot-Flottenrolle.

      > **Important**
      >
      > Um Ihre Spot-Instances bei der Erstellung zu taggen, muss Ihre Amazon EC2 Spot Fleet IAM-Rolle die neuere verwaltete SpotFleetTaggingRoleAmazonEC2-Richtlinie verwenden. Die von Amazon EC2 SpotFleet Role verwaltete Richtlinie verfügt nicht über die erforderlichen Berechtigungen, um Spot-Instances zu taggen. Weitere Informationen finden Sie unter Spot-Instances wurden bei der Erstellung nicht markiert und the section called “Markieren Ihrer -Ressourcen”.

   d. Wählen Sie für Mindestanzahl an vCPUs die Mindestanzahl an vCPUs aus, die Ihre Rechenumgebung unabhängig von der Nachfrage in der Jobwarteschlange verwaltet.

   e. Wählen Sie für Desired vCPUs die Anzahl der vCPUs aus, mit denen Ihre Computerumgebung gestartet wird. Wenn die Anforderungen Ihrer Auftragswarteschlange steigen, kann AWS Batch die gewünschte Anzahl von vCPUs in Ihrer Datenverarbeitungsumgebung erhöhen und EC2 Instances hinzufügen, und zwar bis zu

der maximalen Anzahl von vCPUs. Mit abnehmenden Anforderungen kann AWS Batch die gewünschte Anzahl von vCPUs in Ihrer Datenverarbeitungsumgebung verringern und Instances entfernen, bis die Mindestanzahl von vCPUs erreicht ist.

f. Wählen Sie unter Maximum vCPUs die maximale Anzahl von vCPUs aus, auf die Ihre Rechenumgebung skaliert werden kann, unabhängig von der Nachfrage in der Job-Warteschlange.

g. Wählen Sie unter Zulässige Instance-Typen die Amazon EC2 EC2-Instance-Typen aus, die gestartet werden können. Sie können Instance-Familien angeben, um jeden beliebigen Instance-Typ innerhalb dieser Familien zu starten (z. B.`c5`,`c5n`, oder`p3`). Sie können auch bestimmte Größen innerhalb einer Familie angeben (z. B.`c5.8xlarge`). Instanztypen aus Metall gehören nicht zu den Instanzfamilien. Beinhaltet zum Beispiel `c5` nicht`c5.metal`. Sie können auch Instance-Typen (aus den Instance-Familien C4M4, und R4) auswählen, die den Anforderungen Ihrer Job-Warteschlangen entsprechen. `optimal`

> **Note**
>
> Wenn Sie eine Compute-Umgebung erstellen, müssen die Instance-Typen, die Sie für die Compute-Umgebung auswählen, dieselbe Architektur verwenden. Beispielsweise ist es nicht möglich, x86- und ARM-Instances in derselben Compute-Umgebung zu kombinieren.

> **Note**
>
> AWS Batch skaliert GPUs auf der Grundlage der erforderlichen Anzahl in Ihren Job-Warteschlangen. Um die GPU-Planung verwenden zu können, muss die Rechenumgebung Instanztypen aus den `p3` Familien`p2`,`p4`,`p5`,`g3`, `g3sg4`, oder `g5` enthalten.

> **Note**
>
> `optimal`Verwendet derzeit Instanztypen aus den R4 Instanzfamilien C4M4, und. Falls AWS-Regionen es keine Instanztypen aus diesen Instanzfamilien gibt, werden Instanztypen aus den R5 Instanzfamilien C5M5, und verwendet.

h. Erweitern Sie Additional configuration (Zusätzliche Konfiguration).

i. (Optional) Geben Sie für Placement-Gruppe einen Namen für die Placement-Gruppe ein, um Ressourcen in der Rechenumgebung zu gruppieren.

j. (Optional) Wählen Sie für das EC2-Schlüsselpaar ein öffentliches und ein privates key pair als Sicherheitsanmeldedaten aus, wenn Sie eine Verbindung mit der Instance herstellen. Weitere Informationen zu Amazon EC2 EC2-Schlüsselpaaren finden Sie unter Amazon EC2 EC2-Schlüsselpaare und Linux-Instances.

k. Wählen Sie als Allocation strategy (Zuordnungsstrategie) die Zuordnungsstrategie aus, die bei der Auswahl von Instance-Typen aus der Liste der zulässigen Instance-Typen verwendet werden soll. BEST_FIT_PROGRESSIVE ist normalerweise die bessere Wahl für EC2-On-Demand-Rechenumgebungen, SPOT_CAPACITY_OPTIMIZED und SPOT_PRICE_CAPACITY_OPTIMIZED für EC2-Spot-Rechenumgebungen. Weitere Informationen finden Sie unter the section called “Zuweisungsstrategien”.

l. (Optional) Wählen Sie für die EC2-Konfiguration die Werte Image-Typ und Image-ID aus, um Informationen zur AWS Batch Auswahl von Amazon Machine Images (AMIs) für Instances in der Rechenumgebung bereitzustellen. Wenn die Image-ID-Override nicht für jeden Image-Typ angegeben ist, AWS Batch wird ein aktuelles Amazon ECS-optimiertes AMI ausgewählt. Wenn kein Image-Typ angegeben ist, ist der Standard Amazon Linux 2 für Instances ohne GPU oder AWS Graviton.

> **Important**
>
> Um ein benutzerdefiniertes AMI zu verwenden, wählen Sie den Image-Typ aus und geben Sie dann die benutzerdefinierte AMI-ID in das Feld Image-ID Override ein.

Amazon Linux 2

Standard für alle AWS Graviton-basierten Instance-Familien (z. B., `C6g` `M6gR6g`, und`T4g`) und kann für alle Instance-Typen ohne GPU verwendet werden.

Amazon Linux 2 (GPU)

Standard für alle GPU-Instanzfamilien (zum Beispiel `P4` und`G4`) und kann für alle Instance-Typen verwendet werden, die nicht auf AWS Graviton basieren.

Amazon Linux

Kann für Instanzfamilien ohne GPU oder AWS Graviton verwendet werden. Die Standardunterstützung für Amazon Linux AMI wurde eingestellt. Weitere Informationen finden Sie unter Amazon Linux AMI.

> **Note**
>
> Das AMI, das Sie für eine Rechenumgebung auswählen, muss der Architektur der Instance-Typen entsprechen, die Sie für diese Rechenumgebung verwenden möchten. Wenn Ihre Rechenumgebung beispielsweise A1 Instanztypen verwendet, muss das von Ihnen gewählte Compute-Ressourcen-AMI Arm Instances unterstützen. Amazon ECS verkauft x86 sowohl Arm Versionen als auch Versionen des für Amazon ECS optimierten Amazon Linux 2-AMI. Weitere Informationen finden Sie unter Amazon ECS-optimiertes Amazon Linux 2-AMI im Amazon Elastic Container Service Developer Guide.

m. (Optional) Wählen Sie unter Startvorlage eine bestehende Amazon EC2 EC2-Startvorlage aus, um Ihre Rechenressourcen zu konfigurieren. Die Standardversion der Vorlage wird automatisch aufgefüllt. Weitere Informationen finden Sie unter Support für Startvorlagen.

> **Note**
>
> In einer Startvorlage können Sie ein benutzerdefiniertes AMI angeben, das Sie erstellt haben.

n. (Optional) Geben Sie unter Launch template version (Version der Startvorlage) `$Default`, `$Latest` oder eine bestimmte Versionsnummer ein, die verwendet werden soll.

> **Important**
>
> Wenn der Versionsparameter der Startvorlage `$Default` oder ist`$Latest`, wird die Standard- oder neueste Version der angegebenen Startvorlage während eines Infrastruktur-Updates bewertet. Wenn standardmäßig eine andere AMI-ID ausgewählt ist oder die neueste Version der Startvorlage ausgewählt ist, wird diese AMI-ID im Update verwendet. Weitere Informationen finden Sie unter the section called “Aktualisieren der AMI-ID”.

o. Wählen Sie „Nächste Seite“.

7. Gehen Sie im Abschnitt Netzwerkkonfiguration wie folgt vor:

> **Important**
>
> Datenverarbeitungsressourcen benötigen einen externen Netzwerkzugriff, um mit dem Amazon-ECS-Service-Endpunkt zu kommunizieren. Dies kann über einen Schnittstellen-VPC-Endpunkt oder über Ihre Datenverarbeitungsressourcen mit öffentlichen IP-Adressen sein.
> Weitere Informationen zu Interface-VPC-Endpunkten finden Sie unter Amazon-ECS-Schnittstellen-VPC-Endpunkte (AWS PrivateLink) im Amazon Elastic Container Service-Entwicklerhandbuch.
> Wenn Sie keinen Schnittstellen-VPC-Endpunkt konfiguriert haben und Ihre Datenverarbeitungsressourcen keine öffentlichen IP-Adressen haben, müssen Sie Netzwerkadressübersetzung (Network Address Translation, NAT) verwenden, um diesen Zugriff zur Verfügung zu stellen. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch. Weitere Informationen finden Sie unter the section called “Erstellen einer VPC”.

a. Wählen Sie für Virtual Private Cloud (VPC) ID eine VPC aus, auf der Ihre Instances gestartet werden sollen.

b. Wählen Sie für Subnetze die zu verwendenden Subnetze aus. Standardmäßig sind alle Subnetze innerhalb der ausgewählten VPC verfügbar.

> **Note**
>
> AWS Batch auf Amazon EC2 unterstützt Local Zones. Weitere Informationen finden Sie unter Local Zones im Amazon EC2 EC2-Benutzerhandbuch und Amazon ECS-Cluster in Local Zones, Wavelength Zones und AWS Outposts im Amazon Elastic Container Service Developer Guide.

c. (Optional) Wählen Sie für Sicherheitsgruppen eine Sicherheitsgruppe aus, die Sie Ihren Instances zuordnen möchten. Standardmäßig ist die Standardsicherheitsgruppe für Ihre VPC ausgewählt.

8. Wählen Sie „Nächste Seite“.

9. Überprüfen Sie die Konfigurationsschritte zur Überprüfung. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Create Compute Environment aus.

Um eine nicht verwaltete Rechenumgebung mithilfe von EC2-Ressourcen zu erstellen

1. Öffnen Sie die AWS Batch Konsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie in der Navigationsleiste die aus, die Sie verwenden AWS-Region möchten.
3. Wählen Sie auf der Seite Compute Environments die Option Create aus.
4. Konfigurieren Sie die Umgebung.
   a. Wählen Sie für die Konfiguration der Rechenumgebung Amazon Elastic Compute Cloud (Amazon EC2).
   b. Wählen Sie als Orchestrierungstyp die Option Unmanaged aus.
5. Geben Sie unter Name einen eindeutigen Namen für Ihre Rechenumgebung an. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
6. (Optional) Wählen Sie für die Rolle Service eine Rolle aus, mit der der AWS Batch Service die erforderlichen AWS API-Operationen in Ihrem Namen aufrufen kann. Wählen Sie zum Beispiel aus AWSBatchServiceRole. Weitere Informationen finden Sie unter the section called “Verwenden von serviceverknüpften Rollen”..
7. Wählen Sie unter Maximum vCPUs die maximale Anzahl von vCPUs aus, auf die Ihre Rechenumgebung skaliert werden kann, unabhängig von der Nachfrage in der Job-Warteschlange.
8. (Optional) Erweitern Sie Tags. Um einen Tag hinzuzufügen, wählen Sie Add tag (Tag hinzufügen). Geben Sie dann einen Schlüsselnamen und optional einen Wert ein. Wählen Sie Add tag. Weitere Informationen finden Sie unter Markieren Ihrer AWS Batch-Ressourcen.
9. Wählen Sie „Nächste Seite“.
10. Überprüfen Sie die Konfigurationsschritte zur Überprüfung. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Create Compute Environment aus.

Um eine verwaltete Rechenumgebung mit Amazon EKS-Ressourcen zu erstellen

1. Öffnen Sie die AWS Batch Konsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie in der Navigationsleiste die aus, die Sie verwenden AWS-Region möchten.
3. Wählen Sie im Navigationsbereich Datenverarbeitungs-Umgebungen aus.
4. Wählen Sie Erstellen.
5. Wählen Sie für die Konfiguration der Rechenumgebung Amazon Elastic Kubernetes Service (Amazon EKS).
6. Geben Sie unter Name einen eindeutigen Namen für Ihre Rechenumgebung an. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.
7. Wählen Sie unter Instanzrolle ein vorhandenes Instanzprofil aus, dem die erforderlichen IAM-Berechtigungen zugeordnet sind.

   > **Note**
   >
   > Um eine Rechenumgebung in der AWS Batch Konsole zu erstellen, wählen Sie ein Instanzprofil mit den `eks:DescribeCluster` Berechtigungen `eks:ListClusters` und aus.

8. Wählen Sie für EKS-Cluster einen vorhandenen Amazon EKS-Cluster aus.
9. Geben Sie für Namespace einen Kubernetes Namespace ein, um Ihre AWS Batch Prozesse im Cluster zu gruppieren.
10. (Optional) Erweitern Sie Tags. Wählen Sie Tag hinzufügen und geben Sie dann ein Schlüssel-Wert-Paar ein.
11. Wählen Sie „Nächste Seite“.
12. (Optional) Aktivieren Sie für EC2-Spot-Instances verwenden die Option Verwendung von Spot-Instances aktivieren, um Amazon EC2-Spot-Instances zu verwenden.
13. (Nur Spot) Wählen Sie für den On-Demand-Preis von maximal% den maximalen Prozentsatz aus, den ein Spot-Instance-Preis im Vergleich zum On-Demand-Preis für diesen Instance-Typ vor dem Start der Instances betragen kann. Wenn Ihr Höchstpreis beispielsweise 20% beträgt, muss der Spot-Preis weniger als 20% des aktuellen On-Demand-Preises für diese EC2-Instance betragen. Sie zahlen immer nur den niedrigsten (Markt-) Preis und niemals mehr als Ihren

maximalen Prozentsatz. Wenn Sie dieses Feld leer lassen, ist der Standardwert 100 % des On-Demand-Preises.

14. (Nur Spot) Wählen Sie für die Spot-Flottenrolle die Amazon EC2 Spot-Flotten-IAM-Rolle für die SPOT Rechenumgebung aus.

> **Important**
>
> Diese Rolle ist erforderlich, wenn die Zuweisungsstrategie auf festgelegt `BEST_FIT` oder nicht angegeben ist.

15. (Optional) Wählen Sie für Mindestanzahl an vCPUs die Mindestanzahl an vCPUs aus, die Ihre Rechenumgebung verwaltet, unabhängig von der Nachfrage in der Jobwarteschlange.
16. (Optional) Wählen Sie für Maximum vCPUs die maximale Anzahl von vCPUs aus, auf die Ihre Rechenumgebung skaliert werden kann, unabhängig von der Nachfrage in der Job-Warteschlange.
17. Wählen Sie unter Zulässige Instance-Typen die Amazon EC2 EC2-Instance-Typen aus, die gestartet werden können. Sie können Instance-Familien angeben, um jeden beliebigen Instance-Typ innerhalb dieser Familien zu starten (z. B.`c5`,`c5n`, oder`p3`). Sie können auch bestimmte Größen innerhalb einer Familie angeben (z. B.`c5.8xlarge`). Instanztypen aus Metall gehören nicht zu den Instanzfamilien. Beinhaltet zum Beispiel `c5` nicht`c5.metal`. Sie können auch Instance-Typen (aus den Instance-Familien C4M4, und`R4`) auswählen, da Sie diese benötigen, die den Anforderungen Ihrer Job-Warteschlangen entsprechen. `optimal`

> **Note**
>
> Wenn Sie eine Compute-Umgebung erstellen, müssen die Instance-Typen, die Sie für die Compute-Umgebung auswählen, dieselbe Architektur verwenden. Beispielsweise ist es nicht möglich, x86- und ARM-Instances in derselben Compute-Umgebung zu kombinieren.

> **Note**
>
> AWS Batch skaliert GPUs auf der Grundlage der erforderlichen Anzahl in Ihren Job-Warteschlangen. Um die GPU-Planung verwenden zu können, muss die

Rechenumgebung Instance-Typen aus den p3 Familienp2,p4,p5,g3, g3sg4, oder g5 enthalten.

> **Note**
>
> Derzeit nutzt optimal Instance-Typen aus den Instance-Familien C4, M4 und R4. Wenn AWS-Regionen es keine Instance-Typen aus diesen Instance-Familien gibt, werden Instance-Typen aus den R5 Instance-Familien C5M5, und verwendet.

18. (Optional) Erweitern Sie Zusätzliche Konfiguration.

    a. (Optional) Geben Sie unter Platzierungsgruppe einen Namen für die Platzierungsgruppe ein, um Ressourcen in der Rechenumgebung zu gruppieren.

    b. Wählen Sie als Zuweisungsstrategie BEST_FIT_PROGRESSIVE aus.

    c. (Optional) Wählen Sie für die Konfiguration von Amazon Machine Images (AMIs) die Option Amazon Machine Images (AMIS) -Konfiguration hinzufügen aus. Wählen Sie dann einen Image-Typ, geben Sie eine Image-ID, Override und KubernetesVersion ein.

> **Important**
>
> Um ein benutzerdefiniertes AMI zu verwenden, wählen Sie den Image-Typ aus und geben Sie dann die benutzerdefinierte AMI-ID in das Feld Image-ID Override ein.

> **Note**
>
> Wenn die Image-ID-Override nicht für jeden Image-Typ angegeben ist, AWS Batch wird ein aktuelles Amazon ECS-optimiertes AMI ausgewählt. Wenn kein Image-Typ angegeben ist, ist der Standard Amazon Linux 2 für Instances ohne GPU oder AWS Graviton.
>
> Amazon Linux 2
>
> Standard für alle AWS Graviton-basierten Instance-Familien (zum Beispiel, C6g M6gR6g, undT4g) und kann für alle Instance-Typen ohne GPU verwendet werden.

Amazon Linux 2 (GPU)

Standard für alle GPU-Instanzfamilien (zum Beispiel P4 und G4) und kann für alle Instance-Typen verwendet werden, die nicht auf AWS Graviton basieren.

d. (Optional) Wählen Sie als Startvorlage eine vorhandene Startvorlage aus.

e. (Optional) Geben Sie für Version der Startvorlage **`$Default`**, **`$Latest`**, oder eine Versionsnummer ein.

19. Wählen Sie „Nächste Seite“.
20. Wählen Sie für Virtual Private Cloud (VPC) ID eine VPC aus, auf der die Instances gestartet werden sollen.
21. Wählen Sie für Subnetze die zu verwendenden Subnetze aus. Standardmäßig sind alle Subnetze innerhalb der ausgewählten VPC verfügbar.

> **Note**
>
> AWS Batch auf Amazon EKS unterstützt Local Zones. Weitere Informationen finden Sie unter Amazon EKS and AWS Local Zones im Amazon EKS-Benutzerhandbuch.

22. (Optional) Wählen Sie für Sicherheitsgruppen eine Sicherheitsgruppe aus, die Sie Ihren Instances zuordnen möchten. Standardmäßig ist die Standardsicherheitsgruppe für Ihre VPC ausgewählt.
23. Wählen Sie Nächste Seite.
24. Überprüfen Sie die Konfigurationsschritte zur Überprüfung. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Create Compute Environment aus.

Computing-Umgebungsvorlage

Das folgende Beispiel zeigt eine leere Computing-Umgebungsvorlage. Sie können diese Vorlage verwenden, um Ihre Datenverarbeitungsumgebung zu erstellen, die dann in einer Datei gespeichert und mit der AWS CLI `--cli-input-json`-Option verwendet werden kann. Weitere Informationen zu diesen Parametern finden Sie unter CreateComputeEnvironment in der APIAWS Batch-Referenz zu .

```
{
```

```
    "computeEnvironmentName": "",
    "type": "UNMANAGED",
    "state": "DISABLED",
    "unmanagedvCpus": 0,
    "computeResources": {
        "type": "EC2",
        "allocationStrategy": "BEST_FIT_PROGRESSIVE",
        "minvCpus": 0,
        "maxvCpus": 0,
        "desiredvCpus": 0,
        "instanceTypes": [
            ""
        ],
        "imageId": "",
        "subnets": [
            ""
        ],
        "securityGroupIds": [
            ""
        ],
        "ec2KeyPair": "",
        "instanceRole": "",
        "tags": {
            "KeyName": ""
        },
        "placementGroup": "",
        "bidPercentage": 0,
        "spotIamFleetRole": "",
        "launchTemplate": {
            "launchTemplateId": "",
            "launchTemplateName": "",
            "version": ""
        },
        "ec2Configuration": [
            {
                "imageType": "",
                "imageIdOverride": "",
                "imageKubernetesVersion": ""
            }
        ]
    },
    "serviceRole": "",
    "tags": {
        "KeyName": ""
```

```
    },
    "eksConfiguration": {
        "eksClusterArn": "",
        "kubernetesNamespace": ""
    }
}
```

> **Note**
>
> Sie können die vorherige Vorlage für die Datenverarbeitungsumgebung mit dem folgenden AWS CLI Befehl generieren.
>
> ```
> $ aws batch create-compute-environment --generate-cli-skeleton
> ```

Umgebungsparameter berechnen

Rechenumgebungen sind in mehrere grundlegende Komponenten aufgeteilt: Name, Typ und Status der Rechenumgebung, die Rechenressourcendefinition (falls es sich um eine verwaltete Computerumgebung handelt), die Amazon EKS-Konfiguration (wenn sie Amazon EKS-Ressourcen verwendet), die Servicerolle, die verwendet werden soll, um IAM-Berechtigungen zu erteilen AWS Batch, und die Tags für die Rechenumgebung.

Themen

Name der Rechenumgebung

`computeEnvironmentName`

Der Name für Ihre Datenverarbeitungsumgebung. Der Name kann bis zu 128 Zeichen lang sein. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.

Typ: Zeichenfolge

Erforderlich: Ja

Typ

`type`

Der Typ der Datenverarbeitungsumgebung. Wählen Sie`MANAGED`, ob Sie die AWS Batch von Ihnen definierten EC2- oder Fargate-Rechenressourcen verwalten lassen möchten. Weitere Informationen finden Sie unter Ressourcen berechnen. `UNMANAGED`Entscheiden Sie sich dafür, Ihre eigenen EC2-Rechenressourcen zu verwalten.

Typ: Zeichenfolge

Zulässige Werte: `MANAGED` | `UNMANAGED`

Erforderlich: Ja

Status

`state`

Der Status der Datenverarbeitungsumgebung.

Wenn der Status lautet`ENABLED`, versucht der AWS Batch Scheduler, Jobs innerhalb der Umgebung zu platzieren. Diese Jobs stammen aus einer zugehörigen Auftragswarteschlange auf den Rechenressourcen. Wenn die Rechenumgebung verwaltet wird, werden die Instanzen je nach Bedarf in der Auftragswarteschlange automatisch nach oben oder unten skaliert.

Wenn der Status so ist`DISABLED`, versucht der AWS Batch Scheduler nicht, Jobs innerhalb der Umgebung zu platzieren. Jobs, die sich im `RUNNING` Status `STARTING` oder befinden,

werden weiterhin normal bearbeitet. Verwaltete Computerumgebungen, die sich im `DISABLED` Bundesstaat befinden, lassen sich nicht skalieren.

> **Note**
>
> Für Computerumgebungen in einem `DISABLED` Bundesstaat können weiterhin Abrechnungsgebühren anfallen. Um zusätzliche Gebühren zu vermeiden, schalten Sie die Rechenumgebung aus und löschen Sie sie anschließend. Weitere Informationen finden Sie DeleteComputeEnvironmentin der AWS Batch API-Referenz und unter Vermeidung unerwarteter Gebühren im AWS Billing Benutzerhandbuch.

Wenn sich eine Instance im Leerlauf befindet, wird die Instance auf den `minvCpus` Wert herunterskaliert. Die Instanzgröße ändert sich jedoch nicht. Stellen Sie sich zum Beispiel eine `c5.8xlarge` Instanz mit einem `minvCpus` Wert von `4` und einem `desiredvCpus` Wert von vor`36`. Diese Instanz lässt sich nicht auf eine `c5.large` Instanz herunterskalieren.

Typ: Zeichenfolge

Zulässige Werte: `ENABLED` | `DISABLED`

Erforderlich: Nein

Ressourcen berechnen

`computeResources`

Details der Datenverarbeitungsressourcen, die von der Datenverarbeitungsumgebung verwaltet werden. Weitere Informationen finden Sie unter Datenverarbeitungsumgebung.

Typ: ComputeResource Objekt

Erforderlich: Dieser Parameter ist für verwaltete Rechenumgebungen erforderlich

`type`

Der Typ der Datenverarbeitungsumgebung. Sie können wählen, ob Sie EC2 On-Demand-Instances (`EC2`) und EC2-Spot-Instances (`SPOT`) oder Fargate-Kapazität (`FARGATE`) und Fargate-Spot-Kapazität (`FARGATE_SPOT`) in Ihrer verwalteten Rechenumgebung verwenden möchten. Wenn Sie `SPOT` auswählen, müssen Sie auch eine Amazon EC2-Spot-Flottenrolle

mit dem Parameter `spotIamFleetRole` angeben. Weitere Informationen finden Sie unter Amazon EC2-Spot-Flottenrolle.

Zulässige Werte: `EC2` | `SPOT` | `FARGATE` | `FARGATE_SPOT`

Erforderlich: Ja

`allocationStrategy`

Die Zuweisungsstrategie, die für die Rechenressource verwendet werden soll, wenn nicht genügend Instances des am besten geeigneten EC2-Instance-Typs zugewiesen werden können. Dies kann an der Verfügbarkeit des Instance-Typs in den AWS-Region oder an den Amazon EC2-Servicelimits liegen. Weitere Informationen finden Sie unter Zuweisungsstrategien.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

`BEST_FIT` (Standard)

AWS Batch wählt einen Instance-Typ aus, der den Anforderungen der Jobs am besten entspricht, wobei der Instance-Typ mit den niedrigsten Kosten bevorzugt wird. Wenn zusätzliche Instances des ausgewählten Instance-Typs nicht verfügbar sind, wird AWS Batch gewartet, bis die zusätzlichen Instances verfügbar sind. Wenn nicht genügend Instances verfügbar sind oder wenn Sie die Amazon EC2-Servicelimits überschreiten, werden zusätzliche Jobs erst ausgeführt, nachdem die aktuell ausgeführten Jobs abgeschlossen sind. Diese Zuweisungsstrategie hält die Kosten niedriger, kann aber die Skalierung einschränken. Wenn Sie Spot-Flotten mit verwendenBEST_FIT, muss die IAM-Rolle Spot-Flotte angegeben werden. Rechenressourcen, die eine `BEST_FIT` Zuweisungsstrategie verwenden, unterstützen keine Infrastrukturaktualisierungen und können einige Parameter nicht aktualisieren. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen.

> **Note**
>
> `BEST_FIT`wird für Rechenumgebungen, die Amazon EKS-Ressourcen verwenden, nicht unterstützt.

`BEST_FIT_PROGRESSIVE`

Verwenden Sie zusätzliche Instance-Typen, die groß genug sind, um die Anforderungen der Jobs in der Warteschlange zu erfüllen. Bevorzugen Sie Instance-Typen mit niedrigeren Kosten für jede vCPU-Einheit. Wenn zusätzliche Instances der zuvor ausgewählten Instanztypen nicht verfügbar sind, AWS Batch wählt neue Instanztypen aus.

`SPOT_CAPACITY_OPTIMIZED`

(Nur für Spot-Instance-Rechenressourcen verfügbar) Verwenden Sie zusätzliche Instance-Typen, die groß genug sind, um die Anforderungen der Jobs in der Warteschlange zu erfüllen. Bevorzugen Sie Instance-Typen, bei denen die Wahrscheinlichkeit einer Unterbrechung geringer ist.

`SPOT_PRICE_CAPACITY_OPTIMIZED`

(Nur für Spot-Instance-Rechenressourcen verfügbar) Bei der preis- und kapazitätsoptimierten Zuweisungsstrategie werden sowohl der Preis als auch die Kapazität berücksichtigt, um die Spot-Instance-Pools auszuwählen, bei denen die Wahrscheinlichkeit einer Unterbrechung am geringsten ist und die den niedrigstmöglichen Preis haben.

> **Note**
>
> Wir empfehlen, dass Sie `SPOT_PRICE_CAPACITY_OPTIMIZED` eher als `SPOT_CAPACITY_OPTIMIZED` in den meisten Instances verwenden.

Bei `SPOT_PRICE_CAPACITY_OPTIMIZED` Strategien `BEST_FIT_PROGRESSIVESPOT_CAPACITY_OPTIMIZED`, bei denen On-Demand-Instances oder Spot-Instances verwendet werden, und bei der `BEST_FIT` Strategie, die Spot-Instances verwendet, müssen AWS Batch möglicherweise höhere Anforderungen erfüllt werden, `maxvCpus` um Ihre Kapazitätsanforderungen zu erfüllen. Überschreitet `maxvCpus` in diesem Fall AWS Batch nie mehr als eine einzelne Instance.

Zulässige Werte: `BEST_FIT` | `BEST_FIT_PROGRESSIVE` | `SPOT_CAPACITY_OPTIMIZED` | `SPOT_PRICE_CAPACITY_OPTIMIZED`

Erforderlich: Nein

`minvCpus`

Die Mindestanzahl von vCPUs, die eine Umgebung verwaltet, auch wenn es sich um eine Rechenumgebung handelt. `DISABLED`

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Typ: Ganzzahl

Erforderlich: Nein

`maxvCpus`

Die maximale Anzahl von vCPUs, die die AWS Batch Rechenumgebung unterstützen kann.

> **Note**
>
> Bei `BEST_FIT_PROGRESSIVESPOT_CAPACITY_OPTIMIZED`, und `SPOT_PRICE_CAPACITY_OPTIMIZED` Zuweisungsstrategien mit On-Demand-Instances oder Spot-Instances und bei der `BEST_FIT` Strategie, bei der Spot-Instances verwendet werden, müssen AWS Batch möglicherweise höhere Werte erreicht werden, `maxvCpus` um Ihre Kapazitätsanforderungen zu erfüllen. In diesem Fall `maxvCpus` wird AWS Batch niemals mehr als eine einzige Instance überschritten. AWS Batch Verwendet beispielsweise nicht mehr als eine einzelne Instanz aus den in Ihrer Rechenumgebung angegebenen Instanzen.

Typ: Ganzzahl

Erforderlich: Nein

`desiredvCpus`

Die gewünschte Anzahl von vCPUs in der Rechenumgebung. AWS Batch ändert diesen Wert je nach Nachfrage in der Job-Warteschlange zwischen dem Minimal- und dem Maximalwert.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Typ: Ganzzahl

Erforderlich: Nein

`instanceTypes`

Die Instance-Typen, die gestartet werden können. Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden. Geben Sie es nicht an. Sie können Instance-Familien angeben, um jeden Instance-Typ innerhalb dieser Familien zu starten (z. B.`c5`,`c5n`, oder`p3`). Sie können auch bestimmte Größen innerhalb einer Familie angeben (z. B.`c5.8xlarge`). Beachten Sie, dass Metallexemplartypen nicht zu den Exemplarfamilien gehören (z. B. `c5` nicht `c5.metal` enthalten) Außerdem können Sie `optimal` wählen, um Instance-Typen (aus den C-, M- und R-Instance-Familien) auszuwählen, die den Anforderungen der Auftragswarteschlangen entsprechen.

> **Note**
>
> Wenn Sie eine Compute-Umgebung erstellen, müssen die Instance-Typen, die Sie für die Compute-Umgebung auswählen, dieselbe Architektur verwenden. Beispielsweise ist es nicht möglich, x86- und ARM-Instances in derselben Compute-Umgebung zu kombinieren.

> **Note**
>
> Derzeit nutzt `optimal` Instance-Typen aus den Instance-Familien C4, M4 und R4. Falls AWS-Regionen es keine Instance-Typen aus diesen Instance-Familien gibt, werden Instance-Typen aus den Instance-Familien C5, M5 und R5 verwendet.

Typ: Zeichenfolgen-Array

Erforderlich: Ja

`imageId`

Dieser Parameter ist veraltet.

Die Amazon Machine Image (AMI)-ID, die für in der Datenverarbeitungsumgebung gestartete Instances verwendet wird. Dieser Parameter wird vom `imageIdOverride`-Mitglied der `Ec2Configuration`-Struktur überschrieben.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

> **Note**
>
> Das AMI, das Sie für eine Rechenumgebung auswählen, muss der Architektur der Instance-Typen entsprechen, die Sie für diese Rechenumgebung verwenden möchten. Wenn Ihre Rechenumgebung beispielsweise A1 Instanztypen verwendet, muss das von Ihnen gewählte Compute-Ressourcen-AMI Arm Instances unterstützen. Amazon ECS verkauft x86 sowohl Arm Versionen als auch Versionen des für Amazon ECS optimierten Amazon Linux 2-AMI. Weitere Informationen finden Sie unter Amazon ECS-optimiertes Amazon Linux 2-AMI im Amazon Elastic Container Service Developer Guide.

Typ: Zeichenfolge

Erforderlich: Nein

`subnets`

Die VPC-Subnetze, in denen die Datenverarbeitungsressourcen gestartet werden. Diese Subnetze müssen sich in derselben VPC befinden. Fargate-Rechenressourcen können maximal 16 Subnetze enthalten. Weitere Informationen finden Sie unter VPCs und Subnetze im Amazon-VPC-Benutzerhandbuch.

> **Note**
>
> AWS Batch auf Amazon EC2 und AWS Batch auf Amazon EKS unterstützen Local Zones. Weitere Informationen finden Sie unter Local Zones im Amazon EC2 EC2-Benutzerhandbuch, Amazon EKS and AWS Local Zones im Amazon EKS-Benutzerhandbuch und Amazon ECS-Cluster in Local Zones, Wavelength Zones und AWS Outposts im Amazon Elastic Container Service Developer Guide.
> AWS Batch on Fargate unterstützt derzeit keine Local Zones.

Wenn Sie beim Aktualisieren von Rechenumgebungen eine leere Liste von VPC-Subnetzen angeben, unterscheidet sich das resultierende Verhalten zwischen Fargate- und EC2-Rechenressourcen. Für Fargate-Rechenressourcen wird die Bereitstellung einer leeren Liste so behandelt, als ob dieser Parameter nicht angegeben wäre und keine Änderung vorgenommen würde. Bei EC2-Rechenressourcen werden durch das Bereitstellen einer leeren Liste die VPC-Subnetze aus der Rechenressource entfernt. Wenn Sie die VPC-Subnetze ändern, ist ein Infrastruktur-Update der Rechenumgebung erforderlich. Dies ist sowohl bei Fargate- als auch bei EC2-Rechenressourcen der Fall. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen.

Typ: Zeichenfolgen-Array

Erforderlich: Ja

`securityGroupIds`

Die Amazon EC2-Sicherheitsgruppen, die den in der Datenverarbeitungsumgebung gestarteten Instances zugeordnet sind. Eine oder mehrere Sicherheitsgruppen müssen angegeben werden, entweder in `securityGroupIds` oder unter Verwendung einer Startvorlage, auf die in `launchTemplate` verwiesen wird. Dieser Parameter ist für Jobs erforderlich, die auf Fargate-Ressourcen ausgeführt werden und mindestens eine Sicherheitsgruppe enthalten müssen. (Fargate unterstützt keine Startvorlagen.) Wenn Sicherheitsgruppen sowohl mit `securityGroupIds` als auch `launchTemplate` angegeben werden, werden die Werte in `securityGroupIds` verwendet.

Wenn Sie beim Aktualisieren von Rechenumgebungen eine leere Liste von Sicherheitsgruppen angeben, unterscheidet sich das resultierende Verhalten zwischen Fargate- und EC2-Rechenressourcen. Für Fargate-Rechenressourcen wird die Bereitstellung einer leeren Liste so behandelt, als ob dieser Parameter nicht angegeben wäre und

keine Änderung vorgenommen würde. Bei EC2-Rechenressourcen werden durch das Bereitstellen einer leeren Liste die Sicherheitsgruppen aus der Rechenressource entfernt. Wenn Sie die Sicherheitsgruppen ändern, ist ein Infrastruktur-Update der Rechenumgebung erforderlich. Dies ist sowohl bei Fargate- als auch bei EC2-Rechenressourcen der Fall. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen.

Typ: Zeichenfolgen-Array

Erforderlich: Ja

`ec2KeyPair`

Das EC2-Schlüsselpaar, das für Instances verwendet wird, die in der Rechenumgebung gestartet wurden. Sie können dieses Schlüsselpaar für die Anmeldung bei Ihren Instances mit SSH verwenden. Wenn Sie bei der Aktualisierung einer Rechenumgebung das EC2-Schlüsselpaar ändern, ist ein Infrastruktur-Update der Rechenumgebung erforderlich. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Typ: Zeichenfolge

Erforderlich: Nein

`instanceRole`

Das Amazon ECS-Instance-Profil, das an Amazon EC2 EC2-Instances in einer Rechenumgebung angehängt werden soll. Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden. Geben Sie es nicht an. Sie können den kurzen Namen oder den vollständigen Amazon Resource Namen (ARN) eines Instance-Profil angeben. Zum Beispiel `ecsInstanceRole` oder `arn:aws:iam::`*`aws_account_id`*`:instance-profile/ecsInstanceRole`. Weitere Informationen finden Sie unter Amazon-ECS-Instance-Rolle.

Wenn Sie bei der Aktualisierung einer Rechenumgebung diese Einstellung ändern, ist ein Infrastruktur-Update der Rechenumgebung erforderlich. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen.

Typ: Zeichenfolge

Erforderlich: Nein

`tags`

Schlüssel-Wert-Paar-Tags, die auf EC2-Instances angewendet werden, die in der Rechenumgebung gestartet werden. Sie können z. B. `"Name": "AWS Batch Instance - C4OnDemand"` als Tag angeben, sodass jede Instance in Ihrer Datenverarbeitungsumgebung diesen Namen erhält. Dies ist hilfreich, um Ihre AWS Batch Instances in der Amazon EC2 EC2-Konsole zu erkennen. Diese Tags werden bei der Verwendung des AWS BatchListTagsForResourceAPI-Vorgangs nicht angezeigt.

Wenn Sie eine Rechenumgebung aktualisieren und die EC2-Tags ändern, ist ein Infrastruktur-Update der Rechenumgebung erforderlich. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Erforderlich: Nein

`placementGroup`

Die Amazon EC2-Platzierungsgruppe, die Ihren Datenverarbeitungsressourcen zugeordnet werden soll. Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden. Geben Sie es nicht an. Wenn Sie beabsichtigen, parallel Jobs mit mehreren Knoten an Ihre Rechenumgebung zu senden, sollten Sie erwägen, eine Cluster-Platzierungsgruppe zu erstellen und sie Ihren Rechenressourcen zuzuordnen. Auf diese Weise bleibt Ihr paralleler Auftrag mit mehreren Knoten in einer logischen Gruppierung von Instances innerhalb einer einzelnen Availability Zone mit hohem Netzwerk-Flow-Potenzial. Weitere Informationen finden Sie unter Platzierungsgruppen im Amazon EC2-Benutzerhandbuch für Linux-Instances.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Typ: Zeichenfolge

Erforderlich: Nein

`bidPercentage`

Der maximale Prozentsatz, den ein EC2-Spot-Instance-Preis im Vergleich zum On-Demand-Preis für diesen Instance-Typ vor dem Start der Instances erreichen kann. Wenn Ihr maximaler Prozentsatz beispielsweise 20% beträgt, muss der Spot-Preis weniger als 20% des aktuellen On-Demand-Preises für diese EC2-Instance betragen. Sie zahlen immer nur den niedrigsten (Markt-) Preis und niemals mehr als Ihren maximalen Prozentsatz. Wenn Sie dieses Feld leer lassen, ist der Standardwert 100 % des On-Demand-Preises. Für die meisten Anwendungsfälle empfehlen wir, dieses Feld leer zu lassen.

Wenn Sie bei der Aktualisierung einer Rechenumgebung den Gebotsprozentsatz ändern, ist ein Infrastruktur-Update der Rechenumgebung erforderlich. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Erforderlich: Nein

`spotIamFleetRole`

Der Amazon-Ressourcenname (ARN) der Amazon EC2-Spot-Flotten-IAM-Rolle für eine `SPOT`-Datenverarbeitungsumgebung. Diese Rolle ist erforderlich, wenn die Zuordnungsstrategie auf `BEST_FIT` festgelegt ist oder wenn die Zuordnungsstrategie nicht angegeben ist. Weitere Informationen finden Sie unter Amazon EC2-Spot-Flottenrolle.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

> **Important**
>
> Um Ihre Spot-Instances bei der Erstellung zu taggen, muss die hier angegebene Spot-Flotte-IAM-Rolle die neuere verwaltete AmazonEC2-Richtlinie SpotFleet TaggingRole verwenden. Die zuvor empfohlene Richtlinie zur Verwaltung von SpotFleetAmazonEC2-Rollen verfügt nicht über die erforderlichen Berechtigungen, um Spot-Instances zu taggen. Weitere Informationen finden Sie unter Spot-Instances wurden bei der Erstellung nicht markiert.

Typ: Zeichenfolge

Erforderlich: Dieser Parameter ist für SPOT-Datenverarbeitungsumgebungen erforderlich.

`launchTemplate`

Eine optionale Startvorlage, die zu Ihren Datenverarbeitungsressourcen zugeordnet werden soll. Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden. Geben Sie es nicht an. Alle anderen Rechenressourcenparameter, die Sie in einem CreateComputeEnvironmentoder UpdateComputeEnvironmentAPI-Vorgang angeben, haben Vorrang vor denselben Parametern in der Startvorlage. Um eine Startvorlage zu starten, müssen Sie entweder die Startvorlagen-ID oder den Namen der Startvorlage in der Anfrage eingeben, aber nicht beides. Weitere Informationen finden Sie unter Support für Startvorlagen.

Wenn Sie eine Rechenumgebung aktualisieren, um die benutzerdefinierte Startvorlage zu entfernen und die standardmäßige Startvorlage zu verwenden, setzen Sie das `launchTemplateId` oder `launchTemplateName` -Element der Startvorlagenspezifikation auf eine leere Zeichenfolge. Durch das Entfernen der Startvorlage aus einer Rechenumgebung wird das in der Startvorlage angegebene AMI nicht entfernt, sofern es das verwendete war. Um das aus einer Startvorlage ausgewählte AMI zu aktualisieren, muss der `updateToLatestImageVersion` Parameter auf gesetzt sein`true`. Wenn Sie bei der Aktualisierung einer Rechenumgebung die Startvorlage ändern, ist ein Infrastruktur-Update

der Rechenumgebung erforderlich. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen.

Typ: LaunchTemplateSpecification

object

Erforderlich: Nein

`launchTemplateId`

Die ID der Startvorlage.

Typ: Zeichenfolge

Erforderlich: Nein

`launchTemplateName`

Der Name der Startvorlage.

Typ: Zeichenfolge

Erforderlich: Nein

`version`

Die Versionsnummer der Startvorlage: `$Latest` oder `$Default`.

Wenn der Wert `$Latest` lautet, wird die neueste Version der Einführungsvorlage verwendet. Wenn der Wert `$Default` lautet, wird die Standardversion der Einführungsvorlage verwendet. Bei einem Infrastruktur-Update wird die Version der Startvorlage AWS Batch neu bewertet, sofern entweder `$Latest` oder für die Rechenumgebung angegeben `$Default` wurde, und es wird möglicherweise eine andere Version der Startvorlage verwendet. Dies gilt auch dann, wenn die Startvorlage im Update nicht angegeben wurde.

Standard: `$Default`.

Typ: Zeichenfolge

Erforderlich: Nein

`ec2Configuration`

Stellt Informationen bereit, die zur Auswahl von Amazon Machine Images (AMIs) für Instances in der EC2-Rechenumgebung verwendet werden. Wenn `Ec2Configuration`

nicht angegeben, ist die Standardeinstellung Amazon Linux 2 (ECS_AL2). Vor dem 31. März 2021 war dieser Standard Amazon Linux (ECS_AL1) für Nicht-GPU- und AWS Nicht-Graviton-Instances.

Wenn Sie bei der Aktualisierung einer Rechenumgebung diesen Parameter ändern, ist ein Infrastruktur-Update der Rechenumgebung erforderlich. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen.

> **Note**
>
> Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden.

Typ: Array von Ec2Configuration-Objekten

Erforderlich: Nein

`imageIdOverride`

Die AMI-ID, die für Instances verwendet wird, die in der Rechenumgebung gestartet wurden und dem Image-Typ entspricht. Diese Einstellung überschreibt `imageId`, die im Objekt `computeResource` eingestellt ist.

Typ: Zeichenfolge

Erforderlich: Nein

`imageKubernetesVersion`

Die Kubernetes Version für die Rechenumgebung. Wenn Sie keinen Wert angeben, wird die neueste Version, die AWS Batch unterstützt, verwendet.

Typ: Zeichenfolge

Längenbeschränkungen: Minimale Länge beträgt 1 Zeichen. Maximale Länge beträgt 256 Zeichen.

Erforderlich: Nein

`imageType`

Der Image-Typ, der mit dem Instance-Typ übereinstimmen muss, um ein AMI auszuwählen. Die unterstützten Werte unterscheiden sich für ECS- und EKS-Ressourcen.

ECS

Wenn der Parameter `imageIdOverride` nicht angegeben wird, wird ein aktuelles Amazon-ECS-optimiertes Linux–2-AMI (`ECS_AL2`) verwendet. Wenn in einem Update ein neuer Image-Typ angegeben wird, aber `imageId` weder ein noch ein `imageIdOverride` Parameter angegeben ist, wird das neueste Amazon ECS-optimierte AMI für diesen Image-Typ verwendet, das von unterstützt AWS Batch wird.

ECS_AL2

Amazon Linux2: Standard für alle Nicht-GPU-Instance-Familien.

ECS_AL2_NVIDIA

Amazon Linux 2 (GPU): Standard für alle GPU-Instance-Familien (zum Beispiel `P4` und`G4`) und kann für alle Instance-Typen verwendet werden, die nicht AWS auf Graviton basieren.

ECS_AL1

Amazon Linux. Amazon Linux hat den end-of-life Standardsupport erreicht. Weitere Informationen finden Sie unter Amazon Linux AMI.

EKS

Wenn der Parameter `imageIdOverride` nicht angegeben wird, wird ein aktuelles Amazon-EKS-optimiertes Linux-AMI (`EKS_AL2`) verwendet. Wenn in einem Update ein neuer Image-Typ angegeben wird, aber `imageId` weder ein noch ein `imageIdOverride` Parameter angegeben ist, wird das neueste Amazon EKS-optimierte AMI für diesen Image-Typ verwendet, das AWS Batch unterstützt wird.

EKS_AL2

Amazon Linux2: Standard für alle Nicht-GPU-Instance-Familien.

EKS_AL2_NVIDIA

Amazon Linux 2 (beschleunigt): Standard für alle GPU-Instance-Familien (z. B. `P4` und`G4`) und kann für alle Instance-Typen verwendet werden, die nicht auf AWS Graviton basieren.

Typ: Zeichenfolge

Längenbeschränkungen: Minimale Länge beträgt 1 Zeichen. Maximale Länge beträgt 256 Zeichen.

Erforderlich: Ja

Amazon EKS-Konfiguration

Konfiguration für den Amazon EKS-Cluster, der die AWS Batch Rechenumgebung unterstützt. Der Cluster muss existieren, bevor die Rechenumgebung erstellt werden kann.

`eksClusterArn`

Der Amazon-Ressourcenname (ARN) für den Amazon-EKS-Cluster. Ein Beispiel ist `arn:aws:eks:us-east-1:123456789012:cluster/ClusterForBatch`.

Typ: Zeichenfolge

Erforderlich: Ja

`kubernetesNamespace`

Der Namespace des Amazon EKS-Clusters. AWS Batch verwaltet Pods in diesem Namespace. Der Wert darf nicht leer oder null sein. Er muss weniger als 64 Zeichen lang sein, darf nicht auf `default` gesetzt werden, darf nicht mit „`kube-`“ beginnen und muss mit diesem regulären Ausdruck übereinstimmen: `^[a-z0-9]([-a-z0-9]*[a-z0-9])?$`. Weitere Informationen finden Sie unter Namespaces in der Kubernetes-Dokumentation.

Typ: Zeichenfolge

Erforderlich: Ja

Typ: Objekt EksConfiguration

Erforderlich: Nein

Servicerolle

`serviceRole`

Der vollständige Amazon-Ressourcenname (ARN) der IAM-Rolle, mit der Sie AWS Batch in Ihrem Namen Anrufe an andere AWS Dienste tätigen können. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für AWS Batch. Wir empfehlen, dass Sie die

Servicerolle nicht angeben. Auf diese Weise wird die AWSServiceRoleForBatchdienstbezogene Rolle AWS Batch verwendet.

> **Important**
>
> Wenn Ihr Konto die AWS Batch dienstverknüpfte Rolle (AWSServiceRoleForBatch) bereits erstellt hat, wird diese Rolle standardmäßig für Ihre Rechenumgebung verwendet, sofern Sie hier keine Rolle angeben. Wenn die AWS Batch dienstverknüpfte Rolle in Ihrem Konto nicht existiert und hier keine Rolle angegeben ist, versucht der Dienst, die AWS Batch dienstverknüpfte Rolle in Ihrem Konto zu erstellen. Weitere Informationen zur serviceverknüpften AWSServiceRoleForBatch-Rolle finden Sie unter Dienstbezogene Rollenberechtigungen für AWS Batch.
> Wenn die Datenverarbeitungsumgebung mithilfe der AWSServiceRoleForBatchserviceverknüpften Rolle erstellt wurde, kann sie nicht geändert werden, sodass sie eine reguläre IAM-Rolle verwendet. Ebenso kann die Rechenumgebung, wenn sie mit einer regulären IAM-Rolle erstellt wurde, nicht geändert werden, sodass sie die AWSServiceRoleForBatchserviceverknüpfte Rolle verwendet.
> Um die Parameter für die Rechenumgebung zu aktualisieren, für deren Änderung ein Infrastruktur-Update erforderlich ist, muss die AWSServiceRoleForBatchserviceverknüpfte Rolle verwendet werden. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen.

Wenn Ihre angegebene Rolle einen anderen Pfad als hat/, stellen Sie sicher, dass Sie entweder den vollständigen Rollen-ARN (empfohlen) angeben oder dem Rollennamen den Pfad voranstellen.

> **Note**
>
> Je nachdem, wie Sie Ihre AWS Batch Servicerolle erstellt haben, kann ihr Amazon-Ressourcenname (ARN) das `service-role` Pfadpräfix enthalten. Wenn Sie nur den Namen der Servicerolle angeben, AWS Batch wird davon ausgegangen, dass Ihr ARN das `service-role` Pfadpräfix nicht verwendet. Aus diesem Grund empfehlen wir, dass Sie den vollständigen ARN Ihrer Service-Rolle beim Erstellen von Datenverarbeitungsumgebungen angeben.

Typ: Zeichenfolge

Erforderlich: Nein

Tags

`tags`

Schlüssel-Wert-Paar-Tags, die der Rechenumgebung zugeordnet werden sollen. Weitere Informationen finden Sie unter Markieren Ihrer AWS Batch-Ressourcen.

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Erforderlich: Nein

EC2-Konfigurationen

AWS Batch verwendet Amazon ECS-optimierte AMIs für EC2- und EC2-Spot-Rechenumgebungen. Die Standardeinstellung ist Amazon Linux 2 (`ECS_AL2`). Vor dem 31. März 2021 war dieser Standard Amazon Linux (`ECS_AL1`) für Nicht-GPU- und AWS Nicht-Graviton-Instances.

> **Note**
>
> AWS Batch unterstützt auch Amazon Linux 2023.

Das Amazon Linux AMI (auch Amazon Linux 1 genannt) hat am 31. Dezember 2023 sein Lebensende erreicht. AWS Batch hat den Support für Amazon Linux AMI eingestellt, da es ab dem 1. Januar 2024 keine Sicherheitsupdates oder Bugfixes erhalten wird. Weitere Informationen zu Amazon Linux end-of-life finden Sie in den häufig gestellten Fragen zu AL.

Wir empfehlen, bestehende Amazon Linux-basierte Rechenumgebungen auf Amazon Linux 2023 zu aktualisieren, um unvorhergesehene Workload-Unterbrechungen zu vermeiden und weiterhin Sicherheits- und andere Updates zu erhalten.

Ihre Rechenumgebungen, die das Amazon Linux AMI verwenden, funktionieren möglicherweise auch nach dem 31. Dezember 2023 end-of-life weiter. Diese Computerumgebungen erhalten jedoch keine neuen Softwareupdates, Sicherheitspatches oder Bugfixes mehr von AWS. Es liegt in Ihrer Verantwortung, diese Rechenumgebungen anschließend auf dem Amazon Linux AMI aufrechtzuerhalten end-of-life. Wir empfehlen, AWS Batch Rechenumgebungen auf Amazon Linux 2023 oder Amazon Linux 2 zu migrieren, um optimale Leistung und Sicherheit zu gewährleisten.

Hilfe bei der Migration AWS Batch vom Amazon Linux AMI zu Amazon Linux 2023 oder Amazon Linux 2 finden Sie unter Aktualisieren von Rechenumgebungen — AWS Batch

Zuweisungsstrategien

Wenn eine verwaltete Rechenumgebung erstellt wird, AWS Batch wählt die `instanceTypes` angegebenen Instanztypen aus, die den Anforderungen der Jobs am besten entsprechen. Die Zuweisungsstrategie definiert das Verhalten, wenn zusätzliche Kapazität AWS Batch benötigt wird. Dieser Parameter ist nicht auf Aufträge anwendbar, die auf Fargate-Ressourcen ausgeführt werden. Geben Sie diesen Parameter nicht an.

`BEST_FIT` (Standard)

AWS Batch wählt einen Instance-Typ aus, der den Anforderungen der Jobs am besten entspricht, wobei der Instance-Typ mit den niedrigsten Kosten bevorzugt wird. Wenn zusätzliche Instanzen des ausgewählten Instanztyps nicht verfügbar sind, AWS Batch wartet es, bis die zusätzlichen Instanzen verfügbar sind. Wenn nicht genügend Instances verfügbar sind oder wenn der Benutzer die Amazon EC2-Servicekontingente erreicht, werden zusätzliche Jobs erst ausgeführt, wenn die aktuell ausgeführten Jobs abgeschlossen sind. Diese Zuweisungsstrategie hält die Kosten niedriger, kann aber die Skalierung einschränken. Wenn Sie Spot-Flotten mit verwendenBEST_FIT, muss die IAM-Rolle Spot-Flotte angegeben werden. `BEST_FIT`wird bei der Aktualisierung von Rechenumgebungen nicht unterstützt. Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen.

> **Note**
>
> AWS Batch verwaltet AWS Ressourcen in Ihrem Konto. Computerumgebungen mit der BEST_FIT-Zuweisungsstrategie verwendeten ursprünglich standardmäßig Startkonfigurationen. Die Verwendung von Startkonfigurationen mit neuen AWS Konten wird jedoch im Laufe der Zeit eingeschränkt. Daher werden neu erstellte BEST_FIT-Rechenumgebungen ab Ende April 2024 standardmäßig Vorlagen starten. Falls Ihre Servicerolle nicht über die erforderlichen Berechtigungen zur Verwaltung von Startvorlagen verfügt, AWS Batch können Sie weiterhin Startkonfigurationen verwenden. Bestehende Computerumgebungen werden weiterhin Startkonfigurationen verwenden.

`BEST_FIT_PROGRESSIVE`

AWS Batch wählt zusätzliche Instanztypen aus, die groß genug sind, um die Anforderungen der Jobs in der Warteschlange zu erfüllen. Instanztypen mit niedrigeren Kosten für jede vCPU-Einheit werden bevorzugt. Wenn zusätzliche Instanzen der zuvor ausgewählten Instanztypen nicht verfügbar sind, AWS Batch wählt neue Instanztypen aus.

`SPOT_CAPACITY_OPTIMIZED`

AWS Batch wählt einen oder mehrere Instance-Typen aus, die groß genug sind, um die Anforderungen der Jobs in der Warteschlange zu erfüllen. Instanztypen, bei denen die Wahrscheinlichkeit einer Unterbrechung geringer ist, werden bevorzugt. Diese Zuweisungsstrategie ist nur für Spot-Instance-Datenverarbeitungsressourcen verfügbar.

`SPOT_PRICE_CAPACITY_OPTIMIZED`

Die preis- und kapazitätsoptimierte Zuweisungsstrategie betrachtet sowohl den Preis als auch die Kapazität, um die Spot-Instance-Pools auszuwählen, die am unwahrscheinlichsten unterbrochen werden und den niedrigstmöglichen Preis haben. Diese Zuweisungsstrategie ist nur für Spot-Instance-Datenverarbeitungsressourcen verfügbar.

> **Note**
>
> Wir empfehlen, dass Sie `SPOT_PRICE_CAPACITY_OPTIMIZED` eher als `SPOT_CAPACITY_OPTIMIZED` in den meisten Fällen verwenden.

Die `BEST_FIT` Strategien `BEST_FIT_PROGRESSIVE` und verwenden On-Demand-Instances oder Spot-Instances, `SPOT_CAPACITY_OPTIMIZED` und die `SPOT_PRICE_CAPACITY_OPTIMIZED` Strategien und verwenden Spot-Instances. AWS Batch Möglicherweise müssen diese jedoch überschritten werden, `maxvCpus` um Ihre Kapazitätsanforderungen zu erfüllen. Überschreitet `maxvCpus` in diesem Fall AWS Batch nie mehr als eine Instanz.

Aktualisieren von Datenverarbeitungsumgebungen

Nachdem Sie eine Datenverarbeitungsumgebung erstellt haben, die EC2-Ressourcen verwendet, können Sie viele der Einstellungen der Datenverarbeitungsumgebung direkt aktualisieren. Das Ändern einiger Einstellungen erfordert jedoch, dass die Instances in der Datenverarbeitungsumgebung AWS Batch ersetzt.

Für Datenverarbeitungsumgebungen, die Fargate-Ressourcen verwenden, können Sie Folgendes aktualisieren.

- `securityGroupIds`
- `subnets`
- `desiredvCpus`
- `maxvCpus`
- `minvCpus`

AWS Batch verfügt über zwei Aktualisierungsmechanismen. Die erste ist eine Skalierungsaktualisierung, bei der Instances der Datenverarbeitungsumgebung hinzugefügt oder daraus entfernt werden. Die zweite ist eine Infrastrukturaktualisierung, bei der die Instances in der Datenverarbeitungsumgebung ersetzt werden. Ein Infrastruktur-Update dauert viel länger als ein Skalierungs-Update.

Wenn Sie Datenverarbeitungsumgebungen mit aktualisierenAWS Batch, führt das Ändern nur dieser Einstellungen zu einer Skalierungsaktualisierung: gewünschte vCPUs (`desiredvCpus`), maximale vCPUs (`maxvCpus`), minimale vCPUs (`minvCpus`), Servicerolle (`serviceRole`) und Status (`state`).

> **Note**
>
> Wenn Sie die `desiredvCpus` Einstellung aktualisieren, muss der Wert zwischen den `maxvCpus` Werten `minvCpus` und liegen.
> Darüber hinaus muss der aktualisierte `desiredvCpus` Wert größer oder gleich dem aktuellen `desiredvCpus` Wert sein. Weitere Informationen finden Sie unter the section called “Fehlermeldung beim Aktualisieren der `desiredvCpus` Einstellung”.

Wenn eine der folgenden Einstellungen in einer UpdateComputeEnvironment API-Aktion geändert wird, AWS Batch initiiert eine Infrastrukturaktualisierung. Ein Infrastruktur-Update erfordert, dass die Servicerolle auf AWSServiceRoleForBatch (Standard) gesetzt ist und dass die Zuweisungsstrategie `BEST_FIT_PROGRESSIVE`SPOT_CAPACITY_OPTIMIZED, oder istSPOT_PRICE_CAPACITY_OPTIMIZED. `BEST_FIT` wird nicht unterstützt. Mit Ausnahme der Servicerolle können alle Einstellungen, die für ein Skalierungs-Update geändert werden können, auch für ein Infrastruktur-Update geändert werden.

> **Note**
>
> Wir empfehlen, in den meisten `SPOT_PRICE_CAPACITY_OPTIMIZED` Instances anstelle von `SPOT_CAPACITY_OPTIMIZED`n zu verwenden.

Während einer Infrastrukturaktualisierung ändert sich der Status der Datenverarbeitungsumgebung in `UPDATING`. Neue Instances werden mit den aktualisierten Einstellungen gestartet. Neue Aufträge werden für die neuen Instances geplant. Aufträge, die derzeit ausgeführt werden, werden gemäß der Infrastruktur-Aktualisierungsrichtlinie versendet. Weitere Informationen finden Sie unter UpdateComputeEnvironment und UpdatePolicy in der AWS Batch-API-Referenz.

Berücksichtigen Sie im `UpdatePolicy` Datentyp die folgenden Szenarien:

> **Note**
>
> In diesen Szenarien gilt Folgendes. Wenn eine Instance beendet wird, werden laufende Aufträge gestoppt. Standardmäßig werden diese Aufträge nicht erneut versucht. Um einen dieser Aufträge nach dem Beenden einer Instance erneut zu versuchen, konfigurieren Sie eine Strategie für den erneuten Versuch von Aufträgen. Weitere Informationen finden Sie unter the section called “Automatisierte Auftragswiederholungen” im AWS Batch-Benutzerhandbuch.

- Wenn die `terminateJobsOnUpdate` Einstellung auf festgelegt ist`true`, werden laufende Aufträge während einer Infrastrukturaktualisierung beendet. Die `jobExecutionTimeoutMinutes` Einstellung wird ignoriert.
- Wenn die `terminateJobsOnUpdate` Einstellung auf festgelegt ist`false`, können Aufträge nach dem Infrastruktur-Update noch länger ausgeführt werden. Diese zusätzliche Zeit wird in der -`jobExecutionTimeoutMinutes`Einstellung konfiguriert. Standardmäßig ist die `jobExecutionTimeoutMinutes` Einstellung 30 Minuten.

Sobald Kapazität in der Datenverarbeitungsumgebung verfügbar ist, werden neue Instances mit den aktualisierten Einstellungen gestartet und Aufträge werden auf den neuen Instances gestartet. Da alle Aufträge auf Instances mit den alten Einstellungen abgeschlossen werden, werden die alten Instances beendet. Welche Kapazität verfügbar wird, bedeutet, dass die gewünschte Anzahl von

vCPUs um mindestens so viele vCPUs unter der maximalen Anzahl von vCPUs liegt, wie für den kleinsten Instance-Typ erforderlich.

Infrastruktur-Updates

Ein Infrastruktur-Update ist erforderlich, um einige Einstellungen für eine Datenverarbeitungsumgebung zu ändern. Wenn eine der folgenden Einstellungen geändert wird, wird ein Infrastruktur-Update gestartet:

> **Important**
>
> Die Datenverarbeitungsumgebung muss die AWSServiceRoleForBatch serviceverknüpfte Rolle verwenden, um Änderungen vorzunehmen, die eine Infrastrukturaktualisierung erfordern.
> Wenn die Datenverarbeitungsumgebung eine serviceverknüpfte Rolle verwendet, kann sie nicht geändert werden, um eine reguläre IAM-Rolle zu verwenden. Ebenso kann die Datenverarbeitungsumgebung nicht geändert werden, um eine serviceverknüpfte Rolle zu verwenden, wenn sie über eine reguläre IAM-Rolle verfügt. Daher können Sie Infrastrukturaktualisierungen nur für Datenverarbeitungsumgebungen durchführen, die mithilfe einer serviceverknüpften Rolle erstellt wurden.

- Zuweisungsstrategie (`allocationStrategy`, muss entweder `BEST_FIT_PROGRESSIVE`, `SPOT_CAPACITY_OPTIMIZED`oder sein`SPOT_PRICE_CAPACITY_OPTIMIZED`. Wenn die ursprüngliche Zuweisungsstrategie lautet`BEST_FIT`, werden Infrastruktur-Updates nicht unterstützt.)

  > **Note**
  >
  > Wir empfehlen, in den meisten `SPOT_PRICE_CAPACITY_OPTIMIZED` Instances anstelle von `SPOT_CAPACITY_OPTIMIZED`n zu verwenden.

- Bid-Prozentsatz (`bidPercentage`)
- EC2-Konfiguration (`ec2Configuration`)
- Schlüsselpaar (`ec2KeyPair`)
- Image-ID (`imageId`)
- Instance-Rolle (`instanceRole`)

- Instance-Typen (`instanceTypes`)
- Startvorlage (`launchTemplate`)
- Platzierungsgruppe (`placementGroup`)
- Sicherheitsgruppen (`securityGroupIds`)
- VPC-Subnetze (`subnets`)
- EC2-Tags (`tags`)
- Datenverarbeitungsumgebungstyp (`type`, kann einer von `EC2` oder sein`SPOT`)
- Ob auf das neueste AMI aktualisiert werden soll, das von AWS Batch während eines Infrastruktur-Updates unterstützt wird `updateToLatestImageVersion`

Aktualisieren der AMI-ID

Während einer Infrastrukturaktualisierung kann sich die AMI-ID der Datenverarbeitungsumgebung ändern, je nachdem, ob AMIs in einer dieser drei Einstellungen angegeben sind. AMIs werden in der `imageId` (in `computeResources`), `imageIdOverride` (in `ec2Configuration`) oder der in angegebenen Startvorlage angegeben`launchTemplate`. Angenommen, in einer dieser Einstellungen sind keine AMI-IDs angegeben und die `updateToLatestImageVersion` Einstellung ist `true`. Anschließend wird das neueste Amazon-ECS-optimierte AMI, das von unterstützt AWS Batch wird, für jedes Infrastruktur-Update verwendet.

Wenn in mindestens einer dieser Einstellungen eine AMI-ID angegeben ist, hängt das Update davon ab, welche Einstellung die AMI-ID bereitgestellt hat, die vor dem Update verwendet wurde. Wenn Sie eine Datenverarbeitungsumgebung erstellen, ist die Priorität für die Auswahl einer AMI-ID zuerst die Startvorlage, dann die `imageId` Einstellung und schließlich die `imageIdOverride` Einstellung. Wenn jedoch die verwendete AMI-ID aus der Startvorlage stammt, wird die AMI-ID durch das Aktualisieren der `imageIdOverride` Einstellungen `imageId` oder nicht aktualisiert. Die einzige Möglichkeit, eine aus der Startvorlage ausgewählte AMI-ID zu aktualisieren, besteht darin, die Startvorlage zu aktualisieren. Wenn der Versionsparameter der Startvorlage `$Default` oder ist`$Latest`, wird die Standardversion oder die neueste Version der angegebenen Startvorlage ausgewertet. Wenn standardmäßig eine andere AMI-ID oder die neueste Version der Startvorlage ausgewählt ist, wird diese AMI-ID bei der Aktualisierung verwendet.

Wenn die Startvorlage nicht zur Auswahl der AMI-ID verwendet wurde, wird die AMI-ID verwendet, die in den `imageIdOverride` Parametern `imageId` oder angegeben ist. Wenn beide angegeben sind, wird die im `imageIdOverride` Parameter angegebene AMI-ID verwendet.

Angenommen, die Datenverarbeitungsumgebung verwendet eine AMI-IDimageId0verride, die durch die launchTemplate Parameter imageId, oder angegeben wird, und Sie möchten das neueste Amazon-ECS-optimierte AMI verwenden, das von unterstützt wirdAWS Batch. Anschließend muss das Update die Einstellungen entfernen, die AMI-IDs bereitgestellt haben. Für erfordert dies imageIddie Angabe einer leeren Zeichenfolge für diesen Parameter. Für erfordert dies imageId0verridedie Angabe einer leeren Zeichenfolge für den -ec2ConfigurationParameter.

Wenn die AMI-ID aus der Startvorlage stammt, können Sie zum neuesten Amazon-ECS-optimierten AMI wechseln, das AWS Batch von auf eine der folgenden Arten unterstützt wird:

- Entfernen Sie die Startvorlage, indem Sie eine leere Zeichenfolge für den launchTemplateName Parameter launchTemplateId oder angeben. Dadurch wird die gesamte Startvorlage entfernt und nicht nur die AMI-ID.
- Wenn die aktualisierte Version der Startvorlage keine AMI-ID angibt, muss der updateToLatestImageVersion Parameter auf gesetzt werdentrue.

Amazon-EKS-Rechenumgebungen

Erste Schritte mit AWS Batch in Amazon EKS bietet eine kurze Anleitung zum Erstellen von EKS-Rechenumgebungen. In diesem Abschnitt finden Sie weitere Details zu Amazon-EKS-Rechenumgebungen.

Themen

- Standard-AMI-Auswahl
- Unterstützte Kubernetes-Versionen
- Aktualisieren der Kubernetes Version der Datenverarbeitungsumgebung
- Gemeinsame Verantwortung der Kubernetes Knoten
- Ausführen eines DaemonSet auf AWS Batch verwalteten Knoten
- Anpassen mit Startvorlagen

Standard-AMI-Auswahl

Wenn Sie eine Amazon-EKS-Rechenumgebung erstellen, müssen Sie kein Amazon Machine Image (AMI) AWS Batch angeben. wählt ein Amazon-EKS-optimiertes AMI basierend auf der Kubernetes Version und den Instance-Typen aus, die in Ihrer CreateComputeEnvironment Anforderung

angegeben sind. Im Allgemeinen empfehlen wir, die Standard-AMI-Auswahl zu verwenden. Weitere Informationen zu Amazon-EKS-optimierten AMIs finden Sie unterAmazon-EKS-optimierte Amazon-Linux-AMIs im Amazon-EKS-Benutzerhandbuch.

Führen Sie den folgenden Befehl aus, um zu sehen, welcher AMI-Typ für Ihre Amazon-EKS-Datenverarbeitungsumgebung ausgewählt AWS Batch ist. Das folgende Beispiel ist ein Nicht-GPU-Instance-Typ.

```
# compute CE example: indicates Batch has chosen the AL2 x86 or ARM EKS 1.29 AMI,
 depending on instance types
$ aws batch describe-compute-environments --compute-environments My-Eks-CE1 \
    | jq '.computeEnvironments[].computeResources.ec2Configuration'
[
  {
    "imageType": "EKS_AL2",
    "imageKubernetesVersion": "1.29"
  }
]
```

Das folgende Beispiel ist ein GPU-Instance-Typ.

```
# GPU CE example: indicates Batch has choosen the AL2 x86 EKS Accelerated 1.29 AMI
$ aws batch describe-compute-environments --compute-environments My-Eks-GPU-CE \
    | jq '.computeEnvironments[].computeResources.ec2Configuration'
[
  {
    "imageType": "EKS_AL2_NVIDIA",
    "imageKubernetesVersion": "1.29"
  }
]
```

Unterstützte Kubernetes-Versionen

AWS Batch auf Amazon EKS unterstützt derzeit die folgenden Kubernetes Versionen:

- 1.29
- 1.28
- 1.27
- 1.26
- 1.25

- 1.24
- 1.23

Möglicherweise wird eine Fehlermeldung angezeigt, die der folgenden ähnelt, wenn Sie die -CreateComputeEnvironmentAPI-Operation oder die -UpdateComputeEnvironmentAPI-Operation zum Erstellen oder Aktualisieren einer Computing-Umgebung verwenden. Dieses Problem tritt auf, wenn Sie eine nicht unterstützte Kubernetes Version in angebenEC2Configuration.

```
At least one imageKubernetesVersion in EC2Configuration is not supported.
```

Um dieses Problem zu beheben, löschen Sie die Datenverarbeitungsumgebung und erstellen Sie sie dann mit einer unterstützten Kubernetes Version neu.

Sie können ein Nebenversions-Upgrade auf Ihrem Amazon-EKS-Cluster durchführen. Sie können den Cluster beispielsweise von 1.xx auf aktualisieren, 1.yy auch wenn die Nebenversion nicht unterstützt wird.

Der Status der Datenverarbeitungsumgebung kann sich jedoch INVALID nach einer Hauptversionsaktualisierung in ändern. Wenn Sie beispielsweise ein Hauptversions-Upgrade von 1.xx auf durchführen2.yy. Wenn die Hauptversion von nicht unterstützt wird AWS Batch, wird eine Fehlermeldung ähnlich der folgenden angezeigt.

```
reason=CLIENT_ERROR - ... EKS Cluster version [2.yy] is unsupported
```

Aktualisieren der Kubernetes Version der Datenverarbeitungsumgebung

Mit können Sie die Kubernetes Version einer Datenverarbeitungsumgebung aktualisieren AWS Batch, um Amazon-EKS-Cluster-Upgrades zu unterstützen. Die Kubernetes Version einer Datenverarbeitungsumgebung ist die Amazon-EKS-AMI-Version für die Kubernetes Knoten, die zur Ausführung von Aufträgen AWS Batch startet. Sie können ein Kubernetes Versionsupgrade auf ihren Amazon-EKS-Knoten durchführen, bevor oder nachdem Sie die Version der Steuerebene des Amazon-EKS-Clusters aktualisiert haben. Wir empfehlen, die Knoten nach dem Upgrade der Steuerebene zu aktualisieren. Weitere Informationen finden Sie unter Aktualisieren einer Amazon-EKS-KubernetesClusterversion im Amazon-EKS-Benutzerhandbuch.

Verwenden Sie die API-Operation , um die Kubernetes Version einer ComputingUpdateComputeEnvironment-Umgebung zu aktualisieren.

```
$ aws batch update-compute-environment \
    --compute-environment <compute-environment-name> \
    --compute-resources \
      'ec2Configuration=[{imageType=EKS_AL2,imageKubernetesVersion=1.23}]'
```

Gemeinsame Verantwortung der Kubernetes Knoten

Die Wartung der Datenverarbeitungsumgebungen ist eine übergreifende Verantwortlichkeit.

- Ändern oder entfernen Sie keine AWS Batch Knoten, Labels, Taints, Namespaces, Startvorlagen oder Auto-Scaling-Gruppen. Fügen Sie keine Taints zu AWS Batch verwalteten Knoten hinzu. Wenn Sie eine dieser Änderungen vornehmen, kann Ihre Datenverarbeitungsumgebung nicht unterstützt werden und es treten Fehler auf, einschließlich inaktiver Instances.
- Richten Sie Ihre Pods nicht auf AWS Batch verwaltete Knoten aus. Wenn Sie Ihre Pods auf die verwalteten Knoten ausrichten, treten eine fehlerhafte Skalierung und hängen gebliebene Auftragswarteschlangen auf. Führen Sie Workloads aus, die nicht AWS Batch auf selbstverwalteten Knoten oder verwalteten Knotengruppen verwenden. Weitere Informationen finden Sie unter Verwaltete Knotengruppen im Amazon-EKS-Benutzerhandbuch.
- Sie können auf eine abzielenDaemonSet, die auf AWS Batch verwalteten Knoten ausgeführt wird. Weitere Informationen finden Sie unter Ausführen eines DaemonSet auf AWS Batch verwalteten Knoten.

AWS Batch aktualisiert die AMIs der Datenverarbeitungsumgebung nicht automatisch. Es liegt in Ihrer Verantwortung, sie zu aktualisieren. Führen Sie den folgenden Befehl aus, um Ihre AMIs auf die neueste AMI-Version zu aktualisieren.

```
$ aws batch update-compute-environment \
    --compute-environment <compute-environment-name> \
    --compute-resources 'updateToLatestImageVersion=true'
```

AWS Batch aktualisiert die Kubernetes Version nicht automatisch. Führen Sie den folgenden Befehl aus, um die Kubernetes Version Ihrer Computerumgebung auf *1.23* zu aktualisieren.

```
$ aws batch update-compute-environment \
    --compute-environment <compute-environment-name> \
    --compute-resources \
      'ec2Configuration=[{imageType=EKS_AL2,imageKubernetesVersion=1.23}]'
```

Beim Aktualisieren auf ein neueres AMI oder die Kubernetes Version können Sie angeben, ob Aufträge beendet werden sollen, wenn sie aktualisiert werden (`terminateJobsOnUpdate`), und wie lange gewartet werden soll, bevor eine Instance ersetzt wird, wenn laufende Aufträge nicht abgeschlossen werden (`jobExecutionTimeoutMinutes`.) Weitere Informationen finden Sie unter Aktualisieren von Datenverarbeitungsumgebungen und in der Infrastruktur-Aktualisierungsrichtlinie (UpdatePolicy), die im UpdateComputeEnvironment-API-Vorgang festgelegt wurde.

Ausführen eines DaemonSet auf AWS Batch verwalteten Knoten

AWS Batch legt Taints auf AWS Batch verwalteten Kubernetes Knoten fest. Sie können mit den folgenden auf eine abzielen, DaemonSet die auf AWS Batch verwalteten Knoten ausgeführt wird`tolerations`.

```
tolerations:
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"
```

Eine andere Möglichkeit, dies zu tun, ist die folgende `tolerations`.

```
tolerations:
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"
  effect: "NoSchedule"
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"
  effect: "NoExecute"
```

Anpassen mit Startvorlagen

AWS Batch in Amazon EKS unterstützt Startvorlagen. Es gibt Einschränkungen hinsichtlich der Aktionen, die Ihre Startvorlage ausführen kann.

> **Important**
>
> AWS Batch führt aus`/etc/eks/bootstrap.sh`. Führen Sie nicht `/etc/eks/bootstrap.sh` in Ihrer Startvorlage oder Ihren cloud-init user-data Skripten aus. Sie können neben dem `--kubelet-extra-args` Parameter weitere Parameter zu bootstrap.sh hinzufügen. Legen Sie dazu die `AWS_BATCH_KUBELET_EXTRA_ARGS` Variable in der `/etc/aws-batch/batch.config` Datei fest. Einzelheiten finden Sie im folgenden Beispiel.

> **Note**
>
> Wenn die Startvorlage geändert wird, nachdem aufgerufen CreateComputeEnvironment wurde, UpdateComputeEnvironment muss aufgerufen werden, um die Version der Startvorlage als Ersatz auszuwerten.

Themen

- Hinzufügen kubelet zusätzlicher Argumente
- Konfigurieren der Container-Laufzeit
- Mounten eines Amazon-EFS-Volumes
- IPv6-Support

Hinzufügen **`kubelet`** zusätzlicher Argumente

AWS Batch unterstützt das Hinzufügen zusätzlicher Argumente zum `kubelet` Befehl . Eine Liste der unterstützten Parameter finden Sie unter kubelet in der Kubernetes -Dokumentation. Im folgenden Beispiel *`--node-labels mylabel=helloworld`* wird der `kubelet` Befehlszeile hinzugefügt.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
mkdir -p /etc/aws-batch

echo AWS_BATCH_KUBELET_EXTRA_ARGS=\"--node-labels mylabel=helloworld\" >> /etc/aws-
batch/batch.config

--==MYBOUNDARY==--
```

Konfigurieren der Container-Laufzeit

Sie können die AWS Batch `CONTAINER_RUNTIME` Umgebungsvariable verwenden, um die Container-Laufzeit auf einem verwalteten Knoten zu konfigurieren. Im folgenden Beispiel wird die

Container-Laufzeit auf festgelegt`containerd`, wenn `bootstrap.sh` ausgeführt wird. Weitere Informationen finden Sie unter `containerd` in der Kubernetes -Dokumentation.

> **Note**
>
> Die `CONTAINER_RUNTIME` Umgebungsvariable entspricht der `--container-runtime` Option von `bootstrap.sh`. Weitere Informationen finden Sie unter Options in der Kubernetes -Dokumentation.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
mkdir -p /etc/aws-batch

echo CONTAINER_RUNTIME=containerd >> /etc/aws-batch/batch.config

--==MYBOUNDARY==--
```

Mounten eines Amazon-EFS-Volumes

Sie können Startvorlagen verwenden, um Volumes auf dem Knoten zu mounten. Im folgenden Beispiel werden die `runcmd` Einstellungen `cloud-config` `packages` und verwendet. Weitere Informationen finden Sie unter Beispiele für Cloud-Konfigurationen in der cloud-init -Dokumentation.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
- amazon-efs-utils

runcmd:
- file_system_id_01=fs-abcdef123
- efs_directory=/mnt/efs
```

```
- mkdir -p ${efs_directory}
- echo "${file_system_id_01}:/ ${efs_directory} efs _netdev,noresvport,tls,iam 0 0"
 >> /etc/fstab
- mount -t efs -o tls ${file_system_id_01}:/ ${efs_directory}

--==MYBOUNDARY==--
```

Um dieses Volume im Auftrag zu verwenden, muss es im Parameter eksProperties zu hinzugefügt werdenRegisterJobDefinition. Das folgende Beispiel ist ein großer Teil der Auftragsdefinition.

```
{
    "jobDefinitionName": "MyJobOnEks_EFS",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": ["ls", "-la", "/efs"],
                    "resources": {
                        "limits": {
                            "cpu": "1",
                            "memory": "1024Mi"
                        }
                    },
                    "volumeMounts": [
                        {
                            "name": "efs-volume",
                            "mountPath": "/efs"
                        }
                    ]
                }
            ],
            "volumes": [
                {
                    "name": "efs-volume",
                    "hostPath": {
                        "path": "/mnt/efs"
                    }
                }
            ]
        }
```

```
    }
}
```

Im Knoten wird das Amazon-EFS-Volume im `/mnt/efs` Verzeichnis gemountet. Im Container für den Amazon-EKS-Auftrag wird das Volume im `/efs` Verzeichnis gemountet.

IPv6-Support

AWS Batch unterstützt Amazon-EKS-Cluster mit IPv6-Adressen. Für die AWS Batch Unterstützung sind keine Anpassungen erforderlich. Bevor Sie beginnen, empfehlen wir Ihnen jedoch, die Überlegungen und Bedingungen zu lesen, die unter Zuweisen von IPv6-Adressen zu Pods und Services im Amazon-EKS-Benutzerhandbuch beschrieben sind.

DatenverarbeitungsressourceSpeicherverwaltung

Wenn der Amazon-ECS-Container-Agent eine Rechenressource in einer Rechenumgebung registriert, muss der Agent bestimmen, wie viel Speicher die Rechenressource zur Reservierung für Ihre Aufträge zur Verfügung hat. Aufgrund des Plattformspeicher-Overheads und des vom Systemkernel belegten Speichers unterscheidet sich diese Zahl von der installierten Speichermenge für Amazon EC2-Instances. Eine `m4.large`-Instance beispielsweise besitzt 8 GiB installierten Speicher. Dies wird jedoch nicht immer auf genau 8 192 MiB Arbeitsspeicher übersetzt, der für Aufträge verfügbar ist, wenn die Rechenressource registriert wird.

Angenommen, Sie geben 8 192 MiB für den Auftrag an und keine Ihrer Rechenressourcen verfügt über 8 192 MiB oder mehr Arbeitsspeicher, um diese Anforderung zu erfüllen. Anschließend kann der Auftrag nicht in Ihrer Datenverarbeitungsumgebung platziert werden. Wenn Sie eine verwaltete Datenverarbeitungsumgebung verwenden, AWS Batch muss einen größeren Instance-Typ starten, um die Anforderung zu erfüllen.

Das standardmäßige AWS Batch-Rechenressourcen-AMI reserviert außerdem 32 MiB Arbeitsspeicher für den Amazon-ECS-Container-Agenten und andere wichtige Systemprozesse. Dieser Speicher ist nicht für die Auftragszuweisung verfügbar. Weitere Informationen finden Sie unter Reservieren von Systemspeicher.

Der Amazon-ECS-Container-Agent verwendet die Docker-Funktion `ReadMemInfo()` für die Abfrage des gesamten für das Betriebssystem verfügbaren Arbeitsspeichers. Linux bietet Befehlszeilen-Dienstprogramme zur Bestimmung des gesamten Arbeitsspeichers.

Example – Bestimmung des Gesamtspeichers für Linux

Der free Befehl gibt den gesamten Arbeitsspeicher zurück, der vom Betriebssystem erkannt wird.

```
$ free -b
```

Im Folgenden finden Sie eine Beispielausgabe für eine `m4.large` Instance, auf der das Amazon-ECS-optimierte Amazon Linux AMI ausgeführt wird.

```
             total       used       free     shared    buffers     cached
Mem:    8373026816  348180480 8024846336      90112   25534464  205418496
-/+ buffers/cache:  117227520 8255799296
```

Diese Instance hat insgesamt 8373026816 Byte Arbeitsspeicher. Das bedeutet, dass 7985 MiB für Aufgaben verfügbar sind.

Reservieren von Systemspeicher

Wenn Sie den gesamten Speicher auf einer Rechenressource mit Ihren Aufträgen belegen, konkurrieren Ihre Aufträge möglicherweise mit kritischen Systemprozessen um den Speicher und verursachen möglicherweise einen Systemausfall. Der Amazon-ECS-Container-Agent stellt eine Konfigurationsvariable namens bereit`ECS_RESERVED_MEMORY`. Sie können diese Konfigurationsvariable verwenden, um eine bestimmte Anzahl von MiB Arbeitsspeicher aus dem Pool zu entfernen, der Ihren Aufträgen zugewiesen ist. Damit wird dieser Arbeitsspeicher für wichtige Systemprozesse reserviert.

Das standardmäßige AWS Batch-Rechenressourcen-AMI reserviert 32 MiB Arbeitsspeicher für den Amazon-ECS-Container-Agenten und andere wichtige Systemprozesse.

Anzeigen von -DatenverarbeitungsressourceMemory (Arbeitsspeicher)

Sie können in der Amazon-ECS-Konsole oder mit der DescribeContainerInstances-API-Operation anzeigen, wie viel Speicher eine Rechenressource bei registriert. Wenn Sie versuchen, Ihre Ressourcenauslastung zu maximieren, indem Sie Ihren Aufträgen so viel Speicher wie möglich für einen bestimmten Instance-Typ bereitstellen, können Sie den für diese Rechenressource verfügbaren Speicher beobachten und dann Ihren Aufträgen diesen Speicherplatz zuweisen.

So zeigen Sie den Arbeitsspeicher für Rechenressourcen an

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.

2. Wählen Sie Cluster und dann den Cluster aus, der Ihre Rechenressourcen hostet, die Sie anzeigen möchten.

   Der Cluster-Name für Ihre Computing-Umgebung beginnt mit dem Namen Ihrer Computing-Umgebung.
3. Wählen Sie Infrastruktur aus.
4. Wählen Sie unter Container-Instances die Container-Instance aus.
5. Im Abschnitt Ressourcen und Netzwerke wird der registrierte und verfügbare Speicher für die Rechenressource angezeigt.

   Der registrierte Speicherwert ist die Rechenressource, die beim ersten Start bei Amazon ECS registriert wurde, und der Wert Verfügbarer Speicher ist die Ressource, die noch nicht Aufträgen zugewiesen wurde.

Überlegungen zu Arbeitsspeicher und vCPU für AWS Batch in Amazon EKS

In AWS Batch auf Amazon EKS können Sie die Ressourcen angeben, die einem Container zur Verfügung gestellt werden. Sie können beispielsweise - `requests` oder -`limits`Werte für vCPU und Arbeitsspeicherressourcen angeben.

Die folgenden Einschränkungen gelten für die Angabe von vCPU-Ressourcen:

- Es muss mindestens eine vCPU `requests` oder ein `limits` Wert angegeben werden.
- Eine vCPU-Einheit entspricht einem physischen oder virtuellen Kern.
- Der vCPU-Wert muss in ganzen Zahlen oder in Schritten von 0,25 eingegeben werden.
- Der kleinste gültige vCPU-Wert ist 0,25.
- Wenn beide angegeben sind, muss der `requests` Wert kleiner oder gleich dem `limits` Wert sein. Auf diese Weise können Sie sowohl weiche als auch harte vCPU-Konfigurationen konfigurieren.
- vCPU-Werte können nicht im milliCPU angegeben werden. Beispielsweise `100m` ist kein gültiger Wert.
- AWS Batch verwendet den `requests` Wert für Skalierungsentscheidungen. Wenn kein `requests` Wert angegeben wird, wird der `limits` Wert in den `requests` Wert kopiert.

Im Folgenden sind Einschränkungen für die Angabe von Speicherressourcen aufgeführt:

- Es muss mindestens ein Speicher `requests` oder `limits` Wert angegeben werden.
- Speicherwerte müssen sich in mebibytes (MiBs) befinden.
- Wenn beide angegeben sind, muss der `requests` Wert gleich dem `limits` Wert sein.
- AWS Batch verwendet den `requests` Wert für Skalierungsentscheidungen. Wenn kein `requests` Wert angegeben ist, wird der `limits` Wert in den `requests` Wert kopiert.

Im Folgenden sind Einschränkungen für die Angabe von GPU-Ressourcen aufgeführt:

- Wenn beide angegeben sind, muss der `requests` Wert gleich dem `limits` Wert sein.
- AWS Batch verwendet den `requests` Wert für Skalierungsentscheidungen. Wenn kein `requests` Wert angegeben wird, wird der `limits` Wert in den `requests` Wert kopiert.

Beispiel für Auftragsdefinitionen

Im Folgenden AWS Batch in der Amazon-EKS-Auftragsdefinition werden weiche vCPU-Freigaben konfiguriert. Auf diese Weise kann AWS Batch auf Amazon EKS die gesamte vCPU-Kapazität für den Instance-Typ nutzen. Wenn jedoch andere Aufträge ausgeführt werden, wird dem Auftrag ein Maximum von 2 vCPUs zugewiesen. Der Arbeitsspeicher ist auf 2 GB begrenzt.

```
{
    "jobDefinitionName": "MyJobOnEks_Sleep",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": ["sleep", "60"],
                    "resources": {
                        "requests": {
                            "cpu": "2",
                            "memory": "2048Mi"
                        }
                    }
                }
            ]
        }
    }
}
```

Die folgende AWS Batch in der Amazon-EKS-Auftragsdefinition hat den `request` Wert 1 und weist dem Auftrag maximal 4 vCPUs zu.

```
{
    "jobDefinitionName": "MyJobOnEks_Sleep",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": ["sleep", "60"],
                    "resources": {
                        "requests": {
                            "cpu": "1"
                        },
                        "limits": {
                            "cpu": "4",
                            "memory": "2048Mi"
                        }
                    }
                }
            ]
        }
    }
}
```

Im Folgenden wird AWS Batch in der Amazon-EKS-Auftragsdefinition ein vCPU-`limits`Wert von 1 und ein `limits` Speicherwert von 1 GB festgelegt.

```
{
    "jobDefinitionName": "MyJobOnEks_Sleep",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": ["sleep", "60"],
                    "resources": {
                        "limits": {
                            "cpu": "1",
                            "memory": "1024Mi"
```

```
                        }
                    }
                }
            ]
        }
    }
}
```

Wenn einen AWS Batch in Amazon-EKS-Auftrag in einen Amazon-EKS-Pod AWS Batch übersetzt, AWS Batch kopiert den`limits` Wert in den `requests` Wert . Dies ist der Fall, wenn kein `requests` Wert angegeben ist. Wenn Sie die vorherige Beispielauftragsdefinition einreichen, `spec` lautet der Pod wie folgt.

```
apiVersion: v1
kind: Pod
...
spec:
  ...
  containers:
    - command:
        - sleep
        - 60
      image: public.ecr.aws/amazonlinux/amazonlinux:2
      resources:
        limits:
          cpu: 1
          memory: 1024Mi
        requests:
          cpu: 1
          memory: 1024Mi
      ...
```

Knoten-CPU- und Speicherreservierungen

AWS Batch stützt sich auf die Standardlogik der `bootstrap.sh` Datei für vCPU- und Speicherreservierungen. Weitere Informationen zur -`bootstrap.sh`Datei finden Sie unter bootstrap.sh. Beachten Sie bei der Größe Ihrer vCPU und Speicherressourcen die folgenden Beispiele.

> **Note**
>
> Wenn keine Instances ausgeführt werden, können sich vCPU- und Speicherreservierungen zunächst auf die AWS Batch Skalierungslogik und die Entscheidungsfindung auswirken. Nachdem die Instances ausgeführt wurden, AWS Batch passt die anfänglichen Zuweisungen an.

Beispiel für eine Knoten-CPU-Reservierung

Der CPU-Reservierungswert wird in Millisekunden berechnet, wobei die Gesamtzahl der vCPUs verwendet wird, die für die Instance verfügbar sind.

vCPU-Nummer	Reservierter Prozentsatz
1	6 %
2	1 %
3-4	0.5%
4 und höher	0,25 %

Bei Verwendung der vorhergehenden Werte gilt Folgendes:

- Der CPU-Reservierungswert für eine `c5.large` Instance mit 2 vCPUs beträgt 70 m. Dies wird wie folgt berechnet: (1*60) + (1*10) = 70 m .
- Der CPU-Reservierungswert für eine `c5.24xlarge` Instance mit 96 vCPUs beträgt 310 m. Dies wird wie folgt berechnet: (1*60) + (1*10) + (2*5) + (92*2.5) = 310 m.

In diesem Beispiel sind 1930 (berechnete 2000–70) Millicore vCPU-Einheiten verfügbar, um Aufträge auf einer `c5.large` Instance auszuführen. Angenommen, Ihr Auftrag erfordert 2 (2*1000 m) vCPU-Einheiten, der Auftrag passt nicht auf eine einzelne `c5.large` Instance. Ein Auftrag, für den `1.75` vCPU-Einheiten angepasst werden müssen.

Beispiel für eine Knotenspeicherreservierung

Der Speicherreservierungswert wird wie folgt in Mebibyte berechnet:

- Die Instance-Kapazität in Mebibyte. Eine 8-GB-Instance beträgt beispielsweise 7.748 MiB.
- Der `kubeReserved` Wert. Der `kubeReserved` Wert ist die Speichermenge, die für System-Daemons reserviert werden soll. Der `kubeReserved` Wert wird wie folgt berechnet: ((11 * maximale Anzahl von Pods, die vom Instance-Typ unterstützt werden) + 255). Informationen zur maximalen Anzahl von Pods, die von einem Instance-Typ unterstützt werden, finden Sie unter eni-max-pods.txt
- Der `HardEvictionLimit` Wert. Wenn der verfügbare Speicher unter den `-HardEvictionLimit`Wert fällt, versucht die Instance, Pods zu bereinigen.

Die Formel zur Berechnung des zuweisbaren Speichers lautet wie folgt: (*`instance_capacity_in_MiB`*) - (11 * (*`Maximum_number_of_pods`*) - 255 - (-*`HardEvictionLimitWert.`*)).

Eine Instance `c5.large` unterstützt bis zu 29 Pods. Bei einer 8-GB`c5.large`-Instance mit einem `HardEvictionLimit` Wert von 100 MiB beträgt der zuweisbare Speicher 7074 MiB. Dies wird wie folgt berechnet: (7748 - (11 * 29) -255 -100) = 7074 MiB . In diesem Beispiel passt ein 8.192-MiBAuftrag nicht zu dieser Instance, obwohl es sich um eine 8 gibibyte (GiB)-Instance handelt.

DaemonSets

Beachten Sie DaemonSetsbei der Verwendung von Folgendes:

- Wenn kein AWS Batch auf Amazon-EKS-Instances ausgeführt wird, DaemonSets kann sich zunächst auf die AWS Batch Skalierungslogik und die Entscheidungsfindung auswirken. weist AWS Batch zunächst 0,5 vCPU-Einheiten und 500 MiB für erwartete zuDaemonSets. Nachdem die Instances ausgeführt wurden, AWS Batch passt die anfänglichen Zuweisungen an.
- Wenn ein vCPU- oder Speicherlimits DaemonSet definiert, verfügen AWS Batch Amazon-EKS-Aufträge über weniger Ressourcen. Wir empfehlen Ihnen, die Anzahl der DaemonSets, die AWS Batch Aufträgen zugewiesen sind, so gering wie möglich zu halten.

Planungsrichtlinien

Sie können Planungsrichtlinien verwenden, um zu konfigurieren, wie Rechenressourcen in einer Auftragswarteschlange Benutzern oder Workloads zugewiesen werden. Mithilfe von Planungsrichtlinien können Sie Workloads oder Benutzern unterschiedliche Fair-Share-Kennungen zuweisen. AWS Batch weist jeder Fair-Share-Kennung einen Prozentsatz der gesamten Ressourcen zu, die während eines bestimmten Zeitraums verfügbar sind.

Der Fair-Share-Prozentsatz wird anhand der `shareDistribution` Werte `shareDecaySeconds` und berechnet. Sie können der Fair-Share-Analyse Zeit hinzufügen, indem Sie der Richtlinie eine Freigabe-Zerfallszeit zuweisen. Durch das Hinzufügen von Zeit wird die Zeit stärker gewichtet und weniger die definierte Gewichtung. Sie können Rechenressourcen für Fair-Share-Kennungen reservieren, die nicht aktiv sind, indem Sie eine Rechenreservierung angeben. Weitere Informationen finden Sie unter Planungsrichtlinienparameter.

Themen

Erstellen einer Planungsrichtlinie

Bevor Sie eine Auftragswarteschlange mit einer Planungsrichtlinie erstellen können, müssen Sie eine Planungsrichtlinie erstellen. Wenn Sie eine Planungsrichtlinie erstellen, verknüpfen Sie eine oder mehrere Fair-Share-Kennungen oder Fair-Share-Kennungspräfixe mit Gewichtungen für die Warteschlange und weisen der Richtlinie optional einen Zerfallszeitraum und eine Rechenreservierung zu.

So erstellen Sie eine Planungsrichtlinie

1. Öffnen Sie die -AWS BatchKonsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
3. Wählen Sie im Navigationsbereich Planungsrichtlinien, Erstellen aus.
4. Geben Sie unter Name einen eindeutigen Namen für Ihre Planungsrichtlinie ein. Bis zu 128 Buchstaben (Groß- und Kleinbuchstaben), Ziffern, Bindestriche und Unterstriche sind zulässig.

5. (Optional) Geben Sie für Zerfallssekunden teilen einen Ganzzahlwert für die Zerfallszeit der Freigaberichtlinie der Planungsrichtlinie ein. Bei einer längeren Freigabezeit wird bei der Planung von Aufträgen die Nutzung von Rechenressourcen über einen längeren Zeitraum berücksichtigt. Dies kann es Aufträgen, die eine Fair-Share-Kennung verwenden, ermöglichen, vorübergehend mehr Rechenressourcen zu verwenden, als die Gewichtung für diese Fair-Share-Kennung zulassen würde, wenn diese Fair-Share-Kennung in letzter Zeit nicht über Rechenressourcen verfügt.
6. (Optional) Geben Sie für Datenverarbeitungsreservierung einen Ganzzahlwert für die Datenverarbeitungsreservierung der Planungsrichtlinie ein. Die Rechenreservierung enthält einige vCPUs als Reserve, die für Fair-Share-Kennungen verwendet werden können, die derzeit nicht aktiv sind.

   Das reservierte Verhältnis ist `(`*`computeReservation`*`/100)^`*`ActiveFairShares`*, wobei ActiveFairShares die Anzahl der aktiven Kennungen ist.

   Ein `computeReservation` Wert von 50 gibt beispielsweise an, dass 50 % der maximal verfügbaren VCPU reservieren AWS Batch soll, wenn es nur eine Fair-Share-Kennung gibt, 25 %, wenn es zwei Fair-Share-Kennungen gibt, und 12,5 %, wenn es drei Fair-Share-Kennungen gibt. Ein `computeReservation` Wert von 25 gibt an, dass 25 % der maximal verfügbaren VCPU reservieren AWS Batch soll, wenn es nur eine Fair-Share-Kennung gibt, 6,25 %, wenn es zwei Fair-Share-Kennungen gibt, und 1,56 %, wenn es drei Fair-Share-Kennungen gibt.
7. Im Abschnitt Attribute teilen können Sie die Kennung für Fair Share und die Gewichtung für jede Kennung für Fair Share angeben, die der Planungsrichtlinie zugeordnet werden soll.

   a. Wählen Sie Freigabe-ID hinzufügen aus.

   b. Geben Sie für Freigabekennung die Fair-Share-Kennung an. Wenn die Zeichenfolge mit „*“ endet, wird dies zu einem Fair-Share-Kennungspräfix, das verwendet wird, um Fair-Share-Kennungen für Aufträge abzugleichen. Alle Fair-Share-Kennungen und Fair-Share-Kennungspräfixe in einer Planungsrichtlinie müssen eindeutig sein und dürfen sich nicht überschneiden. Beispielsweise können Sie in derselben Planungsrichtlinie keine Fair-Share-Kennungen mit dem Präfix „UserA *“ und der Fair-Share-Kennung „UserA1“ haben.

   c. Geben Sie für Gewichtungsfaktor die relative Gewichtung für die Fair-Share-Kennung an. Der Standardwert lautet 1.0. Ein niedrigerer Wert hat eine höhere Priorität für Rechenressourcen. Wenn ein Fair-Share-Kennungspräfix verwendet wird, teilen sich Aufträge mit Fair-Share-Kennungen, die mit dem Präfix beginnen, den Gewichtungsfaktor. Dadurch wird der Gewichtungsfaktor für diese Aufträge effektiv erhöht, wodurch ihre

individuelle Priorität verringert und derselbe Gewichtungsfaktor für das Fair-Share-ID-Präfix beibehalten wird.

8. (Optional) Im Abschnitt Tags können Sie den Schlüssel und den Wert für jedes Tag angeben, das der Planungsrichtlinie zugeordnet werden soll. Weitere Informationen finden Sie unter Markieren Ihrer AWS Batch-Ressourcen.
9. Wählen Sie Senden, um den Vorgang abzuschließen und Ihre Planungsrichtlinie zu erstellen.

Vorlage für Planungsrichtlinien

Im Folgenden wird eine leere Vorlage für die Planungsrichtlinie gezeigt. Sie können diese Vorlage verwenden, um Ihre Planungsrichtlinie zu erstellen, die dann in einer Datei gespeichert und mit der AWS CLI `--cli-input-json` Option verwendet werden kann. Weitere Informationen zu diesen Parametern finden Sie unter CreateSchedulingPolicy in der APIAWS Batch-Referenz zu .

```
{
    "name": "",
    "fairsharePolicy": {
        "shareDecaySeconds": 0,
        "computeReservation": 0,
        "shareDistribution": [
            {
                "shareIdentifier": "",
                "weightFactor": 0.0
            }
        ]
    },
    "tags": {
        "KeyName": ""
    }
}
```

> **Note**
>
> Sie können die vorangehende Auftragswarteschlangenvorlage mit dem folgenden AWS CLI Befehl generieren.
>
> ```
> $ aws batch create-scheduling-policy --generate-cli-skeleton
> ```

Planungsrichtlinienparameter

Planungsrichtlinien sind in drei grundlegende Komponenten unterteilt: den Namen, die Fair-Share-Richtlinie und Tags der Planungsrichtlinie.

Themen

Name der Planungsrichtlinie

`name`

Der Name für Ihre Planungsrichtlinie. Bis zu 128 Buchstaben (Groß- und Kleinbuchstaben), Ziffern, Bindestriche und Unterstriche sind zulässig.

Typ: Zeichenfolge

Erforderlich: Ja

Fair-Share-Richtlinie

`fairsharePolicy`

Die Fair-Share-Richtlinie für eine Planungs-Richtlinie.

```
"fairsharePolicy": {
   "computeReservation": number,
   "shareDecaySeconds": number,
   "shareDistribution": [
      {
         "shareIdentifier": "string",
         "weightFactor": number
      }
   ]
}
```

Typ: Objekt

Erforderlich: Nein

`computeReservation`

Ein Wert, der verwendet wird, um einen Teil der verfügbaren maximalen VCPU für Fair-Share-Kennungen zu reservieren, die noch nicht verwendet wurden.

Das reservierte Verhältnis ist `(`*computeReservation*`/100)^`*ActiveFairShares*, wobei ActiveFairShares die Anzahl der aktiven Kennungen ist.

Ein `computeReservation` Wert von 50 gibt beispielsweise an, dass 50 % der maximal verfügbaren VCPU reservieren AWS Batch soll, wenn nur eine aktive Fair-Share-Kennung vorhanden ist, 25 %, wenn es zwei aktive Fair-Share-Kennungen gibt, und 12,5 %, wenn es drei aktive Fair-Share-Kennungen gibt. Ein `computeReservation` Wert von 25 gibt an, dass 25 % der maximal verfügbaren VCPU reservieren AWS Batch soll, wenn nur eine aktive Fair-Share-Kennung vorhanden ist, 6,25 %, wenn es zwei aktive Fair-Share-Kennungen gibt, und 1,56 %, wenn es drei aktive Fair-Share-Kennungen gibt.

Typ: Ganzzahl

Gültiger Bereich: Mindestwert 0. Maximaler Wert von 99.

Erforderlich: Nein

`shareDecaySeconds`

Der Zeitraum, der zur Berechnung eines Fair-Share-Prozentsatzes für jede verwendete Fair-Share-Kennung verwendet werden soll. Der Wert Null (0) zeigt an, dass nur die aktuelle Nutzung gemessen werden sollte. Der Zerfall ermöglicht es, dass kürzlich ausgeführte Aufträge mehr Gewicht haben als Aufträge, die früher ausgeführt wurden.

Typ: Ganzzahl

Gültiger Bereich: Mindestwert 0. Maximaler Wert von 604 800 (1 Woche).

Erforderlich: Nein

`shareDistribution`

Array von Objekten, die die Gewichtungen für die Fair-Share-Kennungen für die Fair-Share-Richtlinie enthalten. Fair-Share-Kennungen, die nicht enthalten sind, haben eine Standardgewichtung von `1.0`.

```
"shareDistribution": [
   {
      "shareIdentifier": "string",
      "weightFactor": number
   }
]
```

Typ: Array

Erforderlich: Nein

`shareIdentifier`

Eine Fair-Share-Kennung oder ein Fair-Share-Kennungspräfix. Wenn die Zeichenfolge mit „*“ endet, gibt diese Zeichenfolge ein Fair-Share-Kennungspräfix für Fair-Share-Kennungen an, die mit diesem Präfix beginnen. Wenn der Wert beispielsweise lautet `UserA*` und 1 `weightFactor` ist und es zwei Fair-Share-Kennungen gibt, die mit beginnen`UserA`, hat jede dieser Fair-Share-Kennungen eine Gewichtung von 2; wenn es fünf solche Fair-Share-Kennungen gibt, hat jede Gewichtung von 5.

Die Liste der Fair-Share-Kennungen und Fair-Share-Kennungspräfixe in einer Fair-Share-Richtlinie darf sich nicht überschneiden. Beispielsweise können Sie in `UserA-1` derselben Fair-Share-Richtlinie kein Fair-Share-Kennungspräfix von `UserA*` und eine Fair-Share-Kennung von haben.

Typ: Zeichenfolge

Erforderlich: Ja

`weightFactor`

Der Gewichtungsfaktor für die Fair-Share-Kennung. Der Standardwert lautet 1.0.
Ein niedrigerer Wert hat eine höhere Priorität für Rechenressourcen. Beispielsweise erhalten Aufträge, die eine Share-Kennung mit einem Gewichtungsfaktor von 0,125 (1/8) verwenden, 8-mal so viele Rechenressourcen wie Aufträge, die eine Share-Kennung mit einem Gewichtungsfaktor von 1 verwenden.

Der kleinste unterstützte Wert ist 0,0001 und der größte unterstützte Wert ist 999.9999.

Typ: Float

Erforderlich: Nein

Tags

`tags`

Schlüssel-Wert-Paar-Tags, die der Planungsrichtlinie zugeordnet werden sollen. Weitere Informationen finden Sie unter Markieren Ihrer AWS Batch-Ressourcen.

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Erforderlich: Nein

Orchestrieren von AWS Batch Aufträgen mit Step-Functions-Zustandsautomaten in der AWS Batch Konsole

Sie können die AWS Batch Konsole verwenden, um Details zu Ihren Step-Functions-Zustandsautomaten und den Funktionen anzuzeigen, die sie verwenden.

Sections

Anzeigen von Details für Zustandsautomaten

Die AWS Batch Konsole zeigt eine Liste Ihrer Zustandsautomaten im aktuellen anAWS-Region, die mindestens einen Workflow-Schritt enthalten, der einen AWS Batch Auftrag übermittelt.

Wählen Sie einen Zustandsautomaten aus, um eine grafische Darstellung des Workflows anzuzeigen. Blau hervorgehobene Schritte stellen AWS Batch Aufträge dar. Verwenden Sie die Diagrammsteuerelemente, um das Diagramm zu vergrößern, zu verkleinern und zu zentrieren.

> **Note**
>
> Wenn ein AWS Batch Auftrag dynamisch mit in der Definition des Zustandsautomaten referenziert JsonPath wird, können die Funktionsdetails nicht in der AWS Batch Konsole angezeigt werden. Stattdessen wird der Auftragsname als dynamische Referenz aufgeführt, und die entsprechenden Schritte im Diagramm sind ausgegraut.

So lassen Sie sich Details zu den Zustandsautomaten anzeigen

1. Öffnen Sie die Seite Workflow-Orchestrierung der -AWS BatchKonsole, die von Step Functions unterstützt wird. https://console.aws.amazon.com/batch/home#stepfunctions
2. Wählen Sie einen Zustandsautomaten aus.
   <result>

   In der AWS Batch-Konsole wird die Seite Details geöffnet.

</result>

Weitere Informationen finden Sie unter Step Functions im AWS Step Functions-Entwicklerhandbuch.

Bearbeiten eines Zustandsautomaten

Wenn Sie einen Zustandsautomaten bearbeiten möchten, AWS Batch öffnet die Seite Definition bearbeiten der Step-Functions-Konsole.

So bearbeiten Sie einen Zustandsautomaten

1. Öffnen Sie die Seite Workflow-Orchestrierung der -AWS BatchKonsole, die von Step Functions unterstützt wird. https://console.aws.amazon.com/batch/home#stepfunctions
2. Wählen Sie einen Zustandsautomaten aus.
3. Wählen Sie Bearbeiten aus.

   In der Step-Functions-Konsole wird die Seite Definition bearbeiten geöffnet.
4. Bearbeiten Sie den Zustandsautomaten und wählen Sie Speichern.

Weitere Informationen zum Bearbeiten von Zustandsautomaten finden Sie unter Step-Functions-Zustandsautomaten-Sprache im AWS Step Functions-Entwicklerhandbuch.

Ausführen eines Zustandsautomaten

Wenn Sie einen Zustandsautomaten ausführen möchten, AWS Batch öffnet die Seite Neue Ausführung der Step-Functions-Konsole.

So führen Sie einen Zustandsautomaten aus

1. Öffnen Sie die Seite Workflow-Orchestrierung der -AWS BatchKonsole, die von Step Functions unterstützt wird. https://console.aws.amazon.com/batch/home#stepfunctions
2. Wählen Sie einen Zustandsautomaten aus.
3. Wählen Sie Execute (Ausführen).

   In der Step-Functions-Konsole wird die Seite Neue Ausführung geöffnet.
4. (Optional) Bearbeiten Sie den Zustandsautomaten und wählen Sie Start execution (Ausführung starten) aus.

Weitere Informationen zum Ausführen von Zustandsautomaten finden Sie unter Ausführungskonzepte für Step-Functions-Zustandsautomaten im AWS Step Functions-Entwicklerhandbuch.

AWS Batch auf AWS Fargate

AWS Fargate ist eine Technologie, die Sie mit verwenden können, AWS Batch um Container auszuführen, ohne Server oder Cluster von Amazon EC2 verwalten zu müssen. Mit AWS Fargate müssen Sie keine Cluster virtueller Maschinen mehr bereitstellen, konfigurieren oder skalieren, um Container auszuführen. Auf diese Weise müssen keine Servertypen mehr ausgewählt werden, es muss nicht entschieden werden, wann die Cluster skaliert werden oder das Cluster-Packing optimiert werden.

Wenn Sie Ihre Aufträge mit Fargate-Ressourcen ausführen, packen Sie Ihre Anwendung in Container, geben die CPU- und Speicheranforderungen an, definieren Netzwerk- und IAM-Richtlinien und starten die Anwendung. Jeder Fargate-Auftrag hat seine eigene Isolationsgrenze und verwendet nicht den zugrunde liegenden Kernel, die CPU-Ressourcen, die Speicherressourcen oder die Elastic-Network-Schnittstelle für einen anderen Auftrag.

Inhalt

Wann Fargate verwendet werden sollte

Wir empfehlen in den meisten Szenarien die Verwendung von Fargate. Fargate startet und skaliert die Rechenleistung so, dass sie genau den Ressourcenanforderungen entspricht, die Sie für den Container angeben. Mit Fargate müssen Sie keine Überbereitstellung vornehmen oder für zusätzliche Server bezahlen. Sie müssen sich auch keine Gedanken über die Besonderheiten von infrastrukturbezogenen Parametern wie Instance-Typ machen. Wenn die Datenverarbeitungsumgebung hochskaliert werden muss, können Aufträge, die auf Fargate-Ressourcen ausgeführt werden, schneller gestartet werden. In der Regel dauert es einige Minuten, bis eine neue Amazon EC2 gestartet wird. Aufträge, die auf Fargate ausgeführt werden, können jedoch in etwa 30 Sekunden bereitgestellt werden. Die genaue erforderliche Zeit hängt von mehreren Faktoren ab, einschließlich der Größe des Container-Images und der Anzahl der Aufträge.

Wir empfehlen jedoch, Amazon EC2 zu verwenden, wenn Ihre Aufträge eine der folgenden Anforderungen erfüllen:

- Mehr als 16 vCPUs
- Mehr als 120 Gibibyte (GiB) Arbeitsspeicher
- Eine GPU
- Ein benutzerdefiniertes Amazon Machine Image (AMI)
- Einer der linuxParameters-Parameter

Wenn Sie eine große Anzahl von Aufträgen haben, empfehlen wir Ihnen, die Amazon EC2Infrastruktur zu verwenden. Wenn beispielsweise die Anzahl der gleichzeitig ausgeführten Aufträge die Fargate-Drosselungsgrenzen überschreitet. Dies liegt daran, dass Aufträge mit EC2 mit einer höheren Rate an EC2-Ressourcen als an Fargate-Ressourcen gesendet werden können. Darüber hinaus können mehr Aufträge gleichzeitig ausgeführt werden, wenn Sie EC2 verwenden. Weitere Informationen finden Sie unter AWS Fargate-Servicekontingente im Amazon Elastic Container Service-Entwicklerhandbuch.

Auftragsdefinitionen auf Fargate

AWS Batch -Aufträge auf Fargate unterstützen nicht alle verfügbaren Auftragsdefinitionsparameter. Einige Parameter werden überhaupt nicht unterstützt, andere verhalten sich bei Fargate-Aufträgen anders.

In der folgenden Liste werden die Parameter der Auftragsdefinition beschrieben, die in Fargate-Aufträgen nicht gültig oder anderweitig eingeschränkt sind.

`platformCapabilities`

Muss als angegeben werden`FARGATE`.

```
"platformCapabilities": [ "FARGATE" ]
```

`type`

Muss als angegeben werden`container`.

```
"type": "container"
```

Parameter in `containerProperties`

`executionRoleArn`

Muss für Aufgaben angegeben werden, die auf Fargate-Ressourcen ausgeführt werden. Weitere Informationen finden Sie unter IAM-Rollen für Aufgaben im Entwicklerhandbuch zum Amazon Elastic Container Service.

```
"executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole"
```

`fargatePlatformConfiguration`

(Optional, nur für Fargate-Auftragsdefinitionen). Gibt die Fargate-Plattformversion oder `LATEST` für eine aktuelle Plattformversion an. Mögliche Werte für `platformVersion` sind `1.3.0`1.4.0, und `LATEST` (Standard).

```
"fargatePlatformConfiguration": { "platformVersion": "1.4.0" }
```

`instanceType`, `ulimits`

Gilt nicht für Aufgaben, die auf Fargate-Ressourcen ausgeführt werden.

`memory`, `vcpus`

Diese Einstellungen müssen in angegeben werden. `resourceRequirements`

`privileged`

Geben Sie entweder diesen Parameter nicht an oder geben Sie an`false`.

```
"privileged": false
```

`resourceRequirements`

Sowohl Speicher- als auch vCPU-Anforderungen müssen mit unterstützten Werten angegeben werden. GPU-Ressourcen werden für Aufgaben, die auf Fargate-Ressourcen ausgeführt werden, nicht unterstützt.

Wenn Sie GuardDuty Laufzeit-Überwachung verwenden, entsteht ein geringer Speicheraufwand für den GuardDuty Sicherheitsagenten. Daher muss das Speicherlimit die Größe des GuardDuty Sicherheitsagenten enthalten. Informationen zu den Speicherlimits für GuardDuty Sicherheitsagenten finden Sie unter CPU- und Speicherlimits im GuardDuty -Benutzerhandbuch.

Informationen zu den bewährten Methoden finden Sie unter Wie behebe ich Fehler aufgrund von unzureichendem Arbeitsspeicher für meine Fargate-Aufgaben nach der Aktivierung der Laufzeit-Überwachung im Amazon-ECS-Entwicklerhandbuch.

```
"resourceRequirements": [
  {"type": "MEMORY", "value": "512"},
  {"type": "VCPU",   "value": "0.25"}
]
```

Parameter in `linuxParameters`

`devices`, `maxSwap`, `sharedMemorySize`, `swappiness`, `tmpfs`

Gilt nicht für Aufgaben, die auf Fargate-Ressourcen ausgeführt werden.

Parameter in `logConfiguration`

`logDriver`

Es werden ausschließlich `awslogs` und `splunk` unterstützt. Weitere Informationen finden Sie unter Verwenden des awslogs-Protokolltreibers.

Mitglieder in `networkConfiguration`

`assignPublicIp`

Wenn dem privaten Subnetz kein NAT-Gateway zugeordnet ist, um Datenverkehr an das Internet zu senden, `assignPublicIp` muss „“ sein`ENABLED`. Weitere Informationen finden Sie unter AWS Batch Ausführung, IAM-Rolle.

Auftragswarteschlangen auf Fargate

AWS Batch -Auftragswarteschlangen auf Fargate bleiben im Wesentlichen unverändert. Die einzige Einschränkung besteht darin, dass die in aufgeführten Datenverarbeitungsumgebungen alle Fargate-Datenverarbeitungsumgebungen (`FARGATE` oder) sein `computeEnvironmentOrder` müssen`FARGATE_SPOT`. EC2- und Fargate-Rechenumgebungen können nicht gemischt werden.

Datenverarbeitungsumgebungen auf Fargate

AWS Batch -Rechenumgebungen auf Fargate unterstützen nicht alle verfügbaren Parameter der Computing-Umgebung. Einige Parameter werden überhaupt nicht unterstützt. Andere haben spezifische Anforderungen für Fargate.

In der folgenden Liste werden Parameter der Datenverarbeitungsumgebung beschrieben, die in Fargate-Aufträgen nicht gültig oder anderweitig eingeschränkt sind.

`type`

Dieser Parameter muss sein`MANAGED`.

```
"type": "MANAGED"
```

Parameter im `computeResources` Objekt

`allocationStrategy`, `bidPercentage`, `desiredvCpus`, `imageId`, `instanceTypes`, `ec2Configuration`, `ec2KeyPair`, `instanceRole`, `launchTemplate`, `minvCpus`, `placementGroup`, `spotIamFleetRole`

Diese gelten nicht für Fargate-Rechenumgebungen und können nicht bereitgestellt werden.

`subnets`

Wenn den in diesem Parameter aufgeführten Subnetzen keine NAT-Gateways zugeordnet sind, muss der `assignPublicIp` Parameter in der Auftragsdefinition auf gesetzt werden`ENABLED`.

`tags`

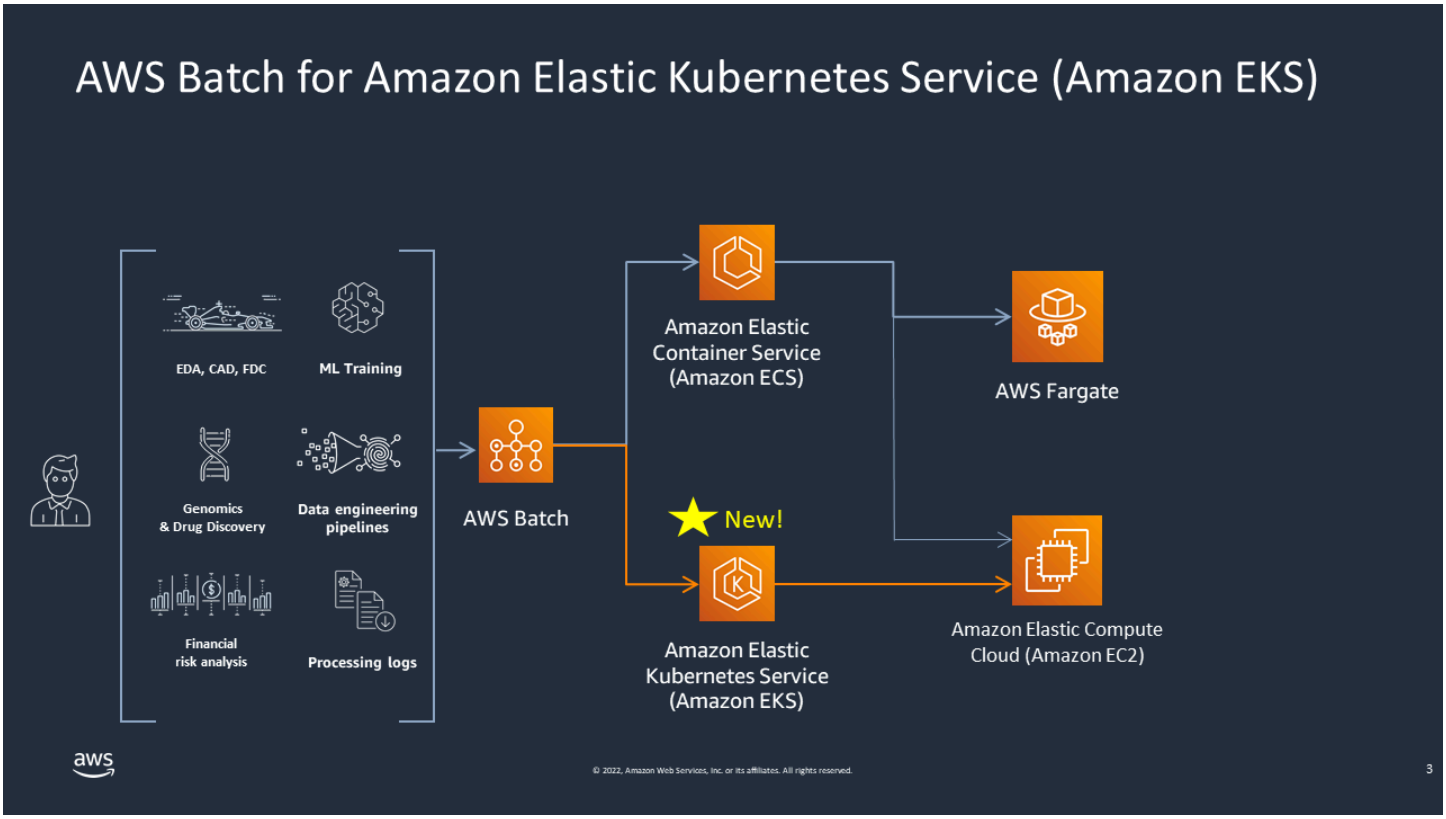
Dies gilt nicht für Fargate-Rechenumgebungen und kann nicht bereitgestellt werden. Um Tags für Fargate-Datenverarbeitungsumgebungen anzugeben, verwenden Sie den `tags` Parameter , der sich nicht im -`computeResources`Objekt befindet.

`type`

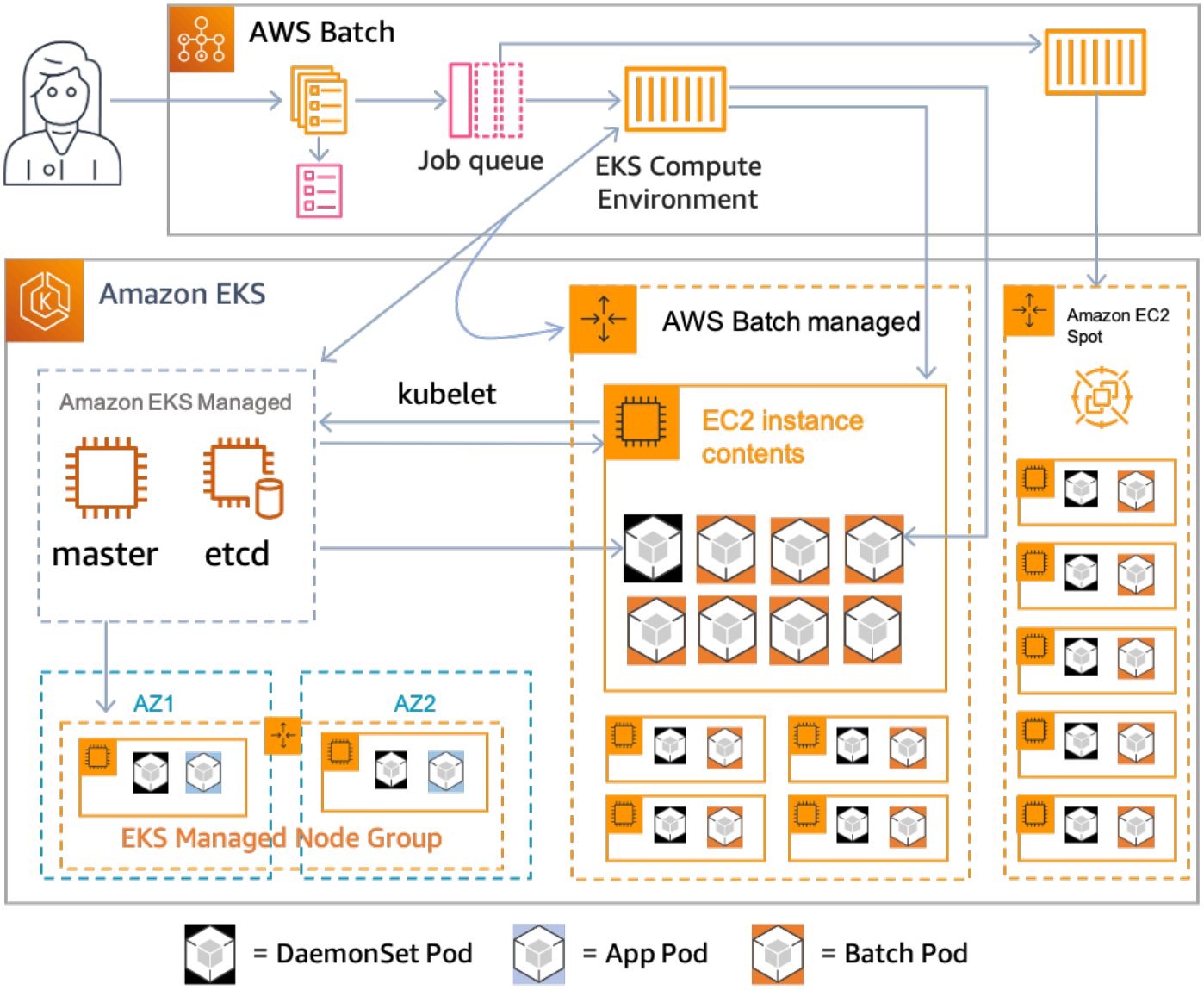
Muss entweder `FARGATE` oder `FARGATE_SPOT` sein.

```
"type": "FARGATE_SPOT"
```

AWS Batch auf Amazon EKS



AWS Batch vereinfacht Ihre Batch-Workloads auf Amazon EKS-Clustern durch die Bereitstellung verwalteter Batch-Funktionen. Dazu gehören Warteschlangen, Abhängigkeitsverfolgung, verwaltete Auftragswiederholungen und Prioritäten, Pod-Verwaltung und Knotenskalierung. AWS Batch kann mehrere Availability Zones und mehrere Amazon EC2 EC2-Instance-Typen und -Größen verarbeiten. AWS Batch integriert mehrere der Best Practices von Amazon EC2 Spot, um Ihre Workloads fehlertolerant auszuführen und so weniger Unterbrechungen zu ermöglichen. Sie können AWS Batch damit problemlos eine Handvoll Jobs über Nacht oder Millionen von geschäftskritischen Jobs ausführen.



AWS Batch ist ein verwalteter Service, der Batch-Workloads in Ihren Kubernetes Clustern orchestriert, die von Amazon Elastic Kubernetes Service (Amazon EKS) verwaltet werden. AWS Batch führt diese Orchestrierung außerhalb Ihrer Cluster mithilfe eines „Overlay“ -Modells durch. Da es AWS Batch sich um einen verwalteten Dienst handelt, müssen keine Kubernetes Komponenten (z. B. Operatoren oder benutzerdefinierte Ressourcen) in Ihrem Cluster installiert oder verwaltet werden. AWS Batch Ihr Cluster muss lediglich mit rollenbasierten Zugriffskontrollen (RBAC) konfiguriert sein, die die Kommunikation mit dem AWS Batch API-Server ermöglichen. Kubernetes AWS Batch ruft Kubernetes APIs auf, um Pods und Knoten zu erstellen, zu überwachen und zu löschenKubernetes.

AWS Batch verfügt über eine integrierte Skalierungslogik zur Skalierung von Kubernetes Knoten auf der Grundlage der Auslastung der Auftragswarteschlangen mit Optimierungen im Hinblick auf die Zuweisung von Jobkapazitäten. Wenn die Auftragswarteschlange leer ist, werden die Knoten

auf die von Ihnen festgelegte Mindestkapazität AWS Batch herunterskaliert, die standardmäßig Null ist. AWS Batch verwaltet den gesamten Lebenszyklus dieser Knoten und schmückt die Knoten mit Beschriftungen und Markierungen. Auf diese Weise werden andere Kubernetes Workloads nicht auf die Knoten übertragen, die von verwaltet werden. AWS Batch Ausgenommen hiervon sind Knoten`DaemonSets`, die auf AWS Batch Knoten abzielen können, um Überwachungs- und andere Funktionen bereitzustellen, die für die ordnungsgemäße Ausführung der Jobs erforderlich sind. Außerdem werden AWS Batch keine Jobs, insbesondere Pods, auf Knoten in Ihrem Cluster ausgeführt, die nicht verwaltet werden. Auf diese Weise können Sie separate Skalierungslogik und Dienste für andere Anwendungen im Cluster verwenden.

Um Jobs an zu senden AWS Batch, interagieren Sie direkt mit der AWS Batch API. AWS Batch übersetzt Jobs in `podspecs` und erstellt dann die Anfragen zum Platzieren von Pods auf Knoten, die von AWS Batch in Ihrem Amazon EKS-Cluster verwaltet werden. Sie können Tools verwenden, `kubectl` um beispielsweise laufende Pods und Knoten anzuzeigen. Wenn ein Pod seine Ausführung abgeschlossen hat, AWS Batch löscht er den Pod, den er erstellt hat, um die Kubernetes Systemlast zu verringern.

Sie können beginnen, indem Sie einen gültigen Amazon EKS-Cluster mit verbinden AWS Batch. Hängen Sie dann eine AWS Batch Auftragswarteschlange an und registrieren Sie eine Amazon EKS-Auftragsdefinition mit `podspec` entsprechenden Attributen. Senden Sie zuletzt Jobs mithilfe des SubmitJobAPI-Vorgangs, der auf die Auftragsdefinition verweist. Weitere Informationen finden Sie unter Erste Schritte mit AWS Batch in Amazon EKS.

Elastic Fabric Adapter

Ein Elastic Fabric Adapter (EFA) ist ein Netzwerkgerät zur Beschleunigung von High Performance Computing(HPC)-Anwendungen. AWS Batch unterstützt Anwendungen, die EFA verwenden, wenn die folgenden Bedingungen erfüllt sind.

- Eine Liste der Instance-Typen, die EFAs unterstützen, finden Sie unter Unterstützte Instance-Typen im Amazon EC2 EC2-Benutzerhandbuch.

  > **Tip**
  >
  > Um eine Liste der Instance-Typen anzuzeigen, die EFAs in unterstützen AWS-Region, führen Sie den folgenden Befehl aus. Vergleichen Sie dann die zurückgegebene Liste mit der Liste der verfügbaren Instance-Typen in der AWS Batch Konsole.
  >
  > ```
  > $ aws ec2 describe-instance-types  --region us-east-1  --filters Name=network-info.efa-supported,Values=true  --query "InstanceTypes[*].[InstanceType]"  --output text | sort
  > ```

- Eine Liste der Betriebssysteme, die EFA unterstützen, finden Sie unter Unterstützte Betriebssysteme.
- Der EFA-Treiber im AMI ist geladen.
- Die Sicherheitsgruppe für den EFA muss allen eingehenden und ausgehenden Datenverkehr von und zu der Sicherheitsgruppe selbst zulassen.
- Alle Instances, die eine EFA verwenden, müssen sich in derselben Cluster-Platzierungsgruppe befinden.
- Die Auftragsdefinition muss ein `devices`-Mitglied enthalten, dessen `hostPath` auf `/dev/infiniband/uverbs0` festgelegt ist, damit das EFA-Gerät an den Container übergeben wird. Wenn `containerPath` angegeben, muss es auch auf gesetzt werden. `/dev/infiniband/uverbs0` Wenn `permissions` festgelegt ist, muss es auf `READ` | `WRITE` | `MKNOD` festgelegt werden.

  Die Standorte der LinuxParametersMitglieder unterscheiden sich bei parallel Jobs mit mehreren Knoten und Container-Jobs mit einem Knoten. Die folgenden Beispiele zeigen die Unterschiede, es fehlen jedoch erforderliche Werte.

Example Beispiel für einen parallelen Auftrag mit mehreren Knoten

```
{
  "jobDefinitionName": "EFA-MNP-JobDef",
  "type": "multinode",
  "nodeProperties": {
    ...
    "nodeRangeProperties": [
      {
        ...
        "container": {
          ...
          "linuxParameters": {
            "devices": [
              {
                "hostPath": "/dev/infiniband/uverbs0",
                "containerPath": "/dev/infiniband/uverbs0",
                "permissions": [
                    "READ", "WRITE", "MKNOD"
                ]
              },
            ],
          },
        },
      },
    ],
  },
}
```

Example Beispiel für einen Container-Auftrag mit Einzelknoten

```
{
  "jobDefinitionName": "EFA-Container-JobDef",
  "type": "container",
  ...
  "containerProperties": {
    ...
    "linuxParameters": {
      "devices": [
        {
          "hostPath": "/dev/infiniband/uverbs0",
        },
```

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

Weitere Informationen zu EFA finden Sie unter Elastic Fabric Adapter im Amazon EC2 EC2-Benutzerhandbuch.

AWS Batch IAM-Richtlinien, -Rollen und -Berechtigungen

Benutzer besitzen standardmäßig keine Berechtigungen zum Erstellen oder Ändern von AWS Batch Ressourcen oder zum Ausführen von Aufgaben mit der AWS Batch API, der AWS BatchKonsole oder der AWS CLI. Damit Benutzer diese Aktionen ausführen können, erstellen Sie IAM-Richtlinien, die Benutzern die Berechtigung für die spezifischen Ressourcen und API-Operationen erteilen. Fügen Sie dann die Richtlinien den Benutzern oder Gruppen hinzu, die diese Berechtigungen benötigen.

Wenn Sie eine Richtlinie an einen Benutzer oder eine Benutzergruppe anfügen, erlaubt oder verweigert die Richtlinie entweder die Berechtigungen zum Ausführen bestimmter Aufgaben für bestimmte Ressourcen. Weitere Informationen finden Sie unter Berechtigungen und Richtlinien im IAM-Benutzerhandbuch. Weitere Informationen zum Verwalten und Erstellen von benutzerdefinierten IAM-Richtlinien finden Sie unter Verwalten von IAM-Richtlinien.

AWS Batch ruft andere AWS-Services in Ihrem Namen auf. Daher AWS Batch muss sich mit Ihren -Anmeldeinformationen authentifizieren. Insbesondere AWS Batchauthentifiziert sich durch Erstellen einer IAM-Rolle und -Richtlinie, die diese Berechtigungen bereitstellt. Anschließend wird die Rolle Ihren Datenverarbeitungsumgebungen zugeordnet, wenn Sie sie erstellen. Weitere Informationen finden Sie unter Amazon-ECS-Instance-Rolle, IAM-Rollen, Verwenden von serviceverknüpften Rollen und Erstellen einer Rolle zum Delegieren von Berechtigungen an einen -AWSService im IAM-Benutzerhandbuch.

Erste Schritte

Eine IAM-Richtlinie muss Berechtigungen zur Verwendung einer oder mehrerer AWS Batch Aktionen erteilen oder verweigern.

Themen

Richtlinienstruktur

In den folgenden Themen wird die Struktur einer IAM-Richtlinie erläutert.

Themen

- Richtliniensyntax
- Aktionen für AWS Batch
- Amazon-Ressourcennamen (ARNs) für AWS Batch
- Prüfen, ob Benutzer über die erforderlichen Berechtigungen verfügen

Richtliniensyntax

Eine IAM-Richtlinie ist ein JSON-Dokument, das eine oder mehrere Anweisungen enthält. Jede Anweisung ist folgendermaßen strukturiert.

```
{
  "Statement":[{
    "Effect":"effect",
    "Action":"action",
    "Resource":"arn",
    "Condition":{
      "condition":{
    "key":"value"
    }
      }
    }
  ]
}
```

Eine Anweisung kann aus verschiedenen Elementen bestehen:

- Effect: Der effect-Wert kann `Allow` oder `Deny` lauten. Standardmäßig haben Benutzer keine Berechtigung zur Verwendung von Ressourcen und API-Aktionen. Daher werden alle Anfragen abgelehnt. Dieser Standardwert kann durch eine explizite Zugriffserlaubnis überschrieben werden. Eine explizite Zugriffsverweigerung überschreibt jedwede Zugriffserlaubnis.
- Aktion : Die Aktion ist die spezifische API-Aktion, für die Sie die Berechtigung erteilen oder verweigern. Anweisungen zur Angabe der Aktion finden Sie unter Aktionen für AWS Batch.

- Resource: Die von einer Aktion betroffene Ressource. Bei einigen AWS Batch API-Aktionen können Sie bestimmte Ressourcen in Ihre Richtlinie aufnehmen, die von der Aktion erstellt oder geändert werden können. Um eine Ressource in der Anweisung anzugeben, verwenden Sie deren Amazon-Ressourcennamen (ARN). Weitere Informationen finden Sie unter Unterstützte Berechtigungen auf Ressourcenebene für AWS Batch-API-Aktionen und Amazon-Ressourcennamen (ARNs) für AWS Batch. Wenn die AWS Batch API-Operation derzeit keine Berechtigungen auf Ressourcenebene unterstützt, fügen Sie einen Platzhalter (*) ein, um anzugeben, dass alle Ressourcen von der Aktion betroffen sein können.
- Condition: Bedingungen sind optional. Mit ihrer Hilfe können Sie bestimmen, wann Ihre Richtlinie wirksam ist.

Weitere Informationen zu Beispiel-IAM-Richtlinienanweisungen für AWS Batchfinden Sie unter Erstellen von AWS Batch IAM-Richtlinien.

Aktionen für AWS Batch

In einer IAM-Richtlinienanweisung können Sie jede API-Aktion von jedem Service, der IAM unterstützt, angeben. AWS BatchVerwenden Sie für das folgende Präfix mit dem Namen der API-Aktion: `batch:` (z. B. `batch:SubmitJob` und `batch:CreateComputeEnvironment`).

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie jede Aktion durch ein Komma.

```
"Action": ["batch:action1", "batch:action2"]
```

Sie können auch mehrere Aktionen angeben, indem Sie einen Platzhalter (*) einschließen. Sie können beispielsweise alle Aktionen mit einem Namen angeben, der mit dem Wort „Beschreiben“ beginnt.

```
"Action": "batch:Describe*"
```

Um alle AWS Batch API-Aktionen anzugeben, fügen Sie einen Platzhalter (*) ein.

```
"Action": "batch:*"
```

Eine Liste der AWS Batch Aktionen finden Sie unter Aktionen in der APIAWS Batch-Referenz zu .

Amazon-Ressourcennamen (ARNs) für AWS Batch

Jede IAM-Richtlinienanweisung gilt für die Ressourcen, die Sie mithilfe ihrer Amazon-Ressourcennamen (ARNs) angeben.

Ein Amazon-Ressourcenname (ARN) hat die folgende allgemeine Syntax:

```
arn:aws:[service]:[region]:[account]:resourceType/resourcePath
```

Service nicht zulässig

Der Service (z. B. `batch`)

Region

Die AWS-Region für die Ressource (z. B. `us-east-2`).

Konto

Die AWS-Konto-ID ohne Bindestriche (z. B. `123456789012`)

RessourcenTyp

Der Typ der Ressource (z. B. `compute-environment`)

resourcePath

Ein Pfad zur Identifizierung der Ressource. Sie können einen Platzhalter (*) in Ihren Pfaden verwenden.

AWS Batch API-Operationen unterstützen derzeit Berechtigungen auf Ressourcenebene für mehrere API-Operationen. Weitere Informationen finden Sie unter Unterstützte Berechtigungen auf Ressourcenebene für AWS Batch-API-Aktionen. Um alle Ressourcen anzugeben oder wenn eine bestimmte API-Aktion keine ARNs unterstützt, fügen Sie einen Platzhalter (*) in das -`Resource`Element ein.

```
"Resource": "*"
```

Prüfen, ob Benutzer über die erforderlichen Berechtigungen verfügen

Bevor Sie eine IAM-Richtlinie in Produktion nehmen, stellen Sie sicher, dass sie Benutzern die Berechtigungen zur Verwendung der spezifischen API-Aktionen und -Ressourcen gewährt, die sie benötigen.

Erstellen Sie dazu zunächst zu Testzwecken einen Benutzer und fügen Sie die IAM-Richtlinie an den Testbenutzer an. Anschließend initiieren Sie mit dem Testbenutzer eine Anforderung. Anforderungen erfolgen über die Konsole oder die AWS CLI.

> **Note**
>
> Sie können Ihre Richtlinien auch mit dem IAM-Richtliniensimulator testen. Weitere Informationen zum Richtliniensimulator finden Sie unter Arbeiten mit dem IAM-Richtliniensimulator im IAM-Benutzerhandbuch.

Falls die Richtlinie dem Benutzer nicht die erwarteten Berechtigungen erteilt oder zu viele Berechtigungen gewährt, können Sie die Richtlinie entsprechend anpassen. Testen Sie so lange, bis Sie die gewünschten Ergebnisse erhalten.

> **Important**
>
> Es kann einige Minuten dauern, bis Richtlinienänderungen wirksam werden. Daher empfehlen wir Ihnen, mindestens fünf Minuten zu warten, bevor Sie Ihre Richtlinienaktualisierungen testen.

Bei einer fehlgeschlagenen Autorisierungsprüfung gibt die Anforderung eine codierte Nachricht mit Diagnoseinformationen zurück. Sie können die Nachricht mit der Aktion `DecodeAuthorizationMessage` decodieren. Weitere Informationen finden Sie unter DecodeAuthorizationMessage in der AWS Security Token Service API-Referenz zu und decode-authorization-message in der AWS CLI -Befehlsreferenz.

Unterstützte Berechtigungen auf Ressourcenebene für AWS Batch-API-Aktionen

Der Begriff Berechtigungen auf Ressourcenebene bezieht sich auf die Möglichkeit, die Ressourcen anzugeben, für die Benutzer Aktionen ausführen dürfen. AWS Batch unterstützt teilweise Berechtigungen auf Ressourcenebene. Bei einigen AWS Batch Aktionen können Sie steuern, wann Benutzer diese Aktionen verwenden dürfen, basierend auf Bedingungen, die erfüllt sein müssen. Sie können auch anhand der spezifischen Ressourcen steuern, die Benutzer verwenden dürfen. Zum

Beispiel können Sie Benutzern die Berechtigungen dazu erteilen, Aufträge zu übergeben, jedoch nur an eine bestimmte Auftragswarteschlange und nur mit einer bestimmten Auftragsdefinition.

In der folgenden Liste werden die AWS Batch API-Aktionen beschrieben, die derzeit Berechtigungen auf Ressourcenebene unterstützen. Die Liste beschreibt auch die unterstützten Ressourcen, Ressourcen-ARNs und Bedingungsschlüssel für jede Aktion.

> **Important**
>
> Wenn eine AWS Batch API-Aktion nicht in dieser Liste aufgeführt ist, unterstützt sie keine Berechtigungen auf Ressourcenebene. Wenn eine AWS Batch API-Aktion keine Berechtigungen auf Ressourcenebene unterstützt, können Sie Benutzern die Berechtigung zur Verwendung der Aktion erteilen. Sie müssen jedoch einen Platzhalter (*) für das Ressourcenelement Ihrer Richtlinienanweisung einfügen.

Aktionen

CancelJob, CreateComputeEnvironment, CreateJobQueue, CreateSchedulingPolicy, DeleteComputeEnvironment, DeleteJobQueue, DeleteSchedulingPolicy, DeregisterJobDefinition, ListTagsForResource, RegisterJobDefinition, SubmitJob, TagResource, TerminateJob, UntagResource, UpdateComputeEnvironment, UpdateSchedulingPolicy, UpdateJobQueue

CancelJob

Bricht einen Auftrag in einer AWS Batch-Warteschlange ab.

Ressource

Job

arn:aws:batch:*region* :*account* :job/*jobId*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

CreateComputeEnvironment

Erstellt eine -AWS BatchDatenverarbeitungsumgebung.

Ressource

Datenverarbeitungsumgebung

arn:aws:batch:*region*:*account*:compute-environment/*compute-environment-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Bedingungsschlüssel

`aws:RequestTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die in der Anforderung übergeben werden.

`aws:TagKeys` (Zeichenfolge)

Filtert Aktionen basierend auf den Tag-Schlüsseln, die in der Anforderung übergeben werden.

CreateJobQueue

Erstellt eine -AWS BatchAuftragswarteschlange.

Ressource

Datenverarbeitungsumgebung

arn:aws:batch:*region*:*account*:compute-environment/*compute-environment-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Auftragswarteschlange

arn:aws:batch:*region*:*account*:job-queue/*queue-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Planungsrichtlinie

arn:aws:batch:*region*:*account*:scheduling-policy/*scheduling-policy-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Bedingungsschlüssel

`aws:RequestTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die in der Anforderung übergeben werden.

`aws:TagKeys` (Zeichenfolge)

Filtert Aktionen basierend auf den Tag-Schlüsseln, die in der Anforderung übergeben werden.

DeleteComputeEnvironment

Löscht eine -AWS BatchRechenumgebung.

Ressource

Datenverarbeitungsumgebung

arn:aws:batch:*region* :*account* :compute-environment/*compute-environment-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

CreateSchedulingPolicy

Erstellt eine -AWS BatchPlanungsrichtlinie.

Ressource

Planungsrichtlinie

arn:aws:batch:*region* :*account* :scheduling-policy/*scheduling-policy-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Bedingungsschlüssel

`aws:RequestTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die in der Anforderung übergeben werden.

`aws:TagKeys` (Zeichenfolge)

Filtert Aktionen basierend auf den Tag-Schlüsseln, die in der Anforderung übergeben werden.

DeleteJobQueue

Löscht die angegebene Auftragswarteschlange Durch das Löschen der Auftragswarteschlange werden schließlich alle Aufträge in der Warteschlange gelöscht. Aufträge werden mit einer Geschwindigkeit von etwa 16 Aufträgen pro Sekunde gelöscht.

Ressource

Auftragswarteschlange

arn:aws:batch:*region* :*account* :job-queue/*queue-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

DeleteSchedulingPolicy

Löscht die angegebene Planungsrichtlinie.

Ressource

Planungsrichtlinie

arn:aws:batch:*region* :*account* :scheduling-policy/*scheduling-policy-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

DeregisterJobDefinition

Hebt die Registrierung einer -AWS BatchAuftragsdefinition auf.

Ressource

Auftragsdefinition

arn:aws:batch:*region* :*account* :job-definition/*definition-name* :*revision*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

ListTagsForResource

Listet die Tags für die angegebene Ressource auf.

Ressource

Datenverarbeitungsumgebung

arn:aws:batch:*region*:*account*:compute-environment/*compute-environment-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Job

arn:aws:batch:*region*:*account*:job/*jobId*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Auftragsdefinition

arn:aws:batch:*region*:*account*:job-definition/*definition-name*:*revision*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Auftragswarteschlange

arn:aws:batch:*region*:*account*:job-queue/*queue-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Planungsrichtlinie

arn:aws:batch:*region* :*account* :scheduling-policy/*scheduling-policy-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

RegisterJobDefinition

Registriert eine -AWS BatchDefinition.

Ressource

Auftragsdefinition

arn:aws:batch:*region* :*account* :job-definition/*definition-name*

Bedingungsschlüssel

`batch:AWSLogsCreateGroup` (Boolean)

Wenn dieser Parameter wahr ist, `awslogs-group` wird für die Protokolle erstellt.

`batch:AWSLogsGroup` (Zeichenfolge)

Die `awslogs` Gruppe, in der sich die Protokolle befinden.

`batch:AWSLogsRegion` (Zeichenfolge)

Die -Region, an die die Protokolle gesendet werden.

`batch:AWSLogsStreamPrefix` (Zeichenfolge)

Das Präfix des `awslogs` Protokollstreams.

`batch:Image` (Zeichenfolge)

Das Docker-Image, das zum Starten eines Auftrags verwendet wird.

`batch:LogDriver` (Zeichenfolge)

Der für den Auftrag verwendete Protokolltreiber.

`batch:Privileged` (Boolean)

Wenn dieser Parameter „true“ ist, erhält der Container für den Auftrag erhöhte Berechtigungen auf der Host-Container-Instance.

`batch:User` (Zeichenfolge)

Der Benutzername oder die numerische UID, die im Container für den Auftrag verwendet werden soll.

`aws:RequestTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die in der Anforderung übergeben werden.

`aws:TagKeys` (Zeichenfolge)

Filtert Aktionen basierend auf den Tag-Schlüsseln, die in der Anforderung übergeben werden.

SubmitJob

Sendet einen -AWS BatchAuftrag aus einer Auftragsdefinition.

Ressource

Job

arn:aws:batch:*region* :*account* :job/*jobId*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Auftragsdefinition

arn:aws:batch:*region* :*account* :job-definition/*definition-name* [:*revision*]

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

> **Note**
>
> Dieser Schlüssel kann nur verwendet werden, wenn der Amazon-Ressourcenname (ARN) der Auftragsdefinition das Format hat. `arn:aws:batch:`*region*`:`*account_number*`:job-definition/`*definition-name*`:`*revision*`.`

Auftragswarteschlange

arn:aws:batch:*region* :*account* :job-queue/*queue-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

TagResource

Markiert die angegebene Ressource.

Ressource

Datenverarbeitungsumgebung

arn:aws:batch:*region* :*account* :compute-environment/*compute-environment-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Job

arn:aws:batch:*region* :*account* :job/*jobId*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Auftragsdefinition

arn:aws:batch:*region* :*account* :job-definition/*definition-name* :*revision*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Auftragswarteschlange

arn:aws:batch:*region* :*account* :job-queue/*queue-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Planungsrichtlinie

arn:aws:batch:*region* :*account* :scheduling-policy/*scheduling-policy-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Bedingungsschlüssel

`aws:RequestTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die in der Anforderung übergeben werden.

`aws:TagKeys` (Zeichenfolge)

Filtert Aktionen basierend auf den Tag-Schlüsseln, die in der Anforderung übergeben werden.

TerminateJob

Beendet einen Auftrag in einer -AWS BatchAuftragswarteschlange.

Ressource

Job

arn:aws:batch:*region* :*account* :job/*jobId*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

UntagResource

Entfernt die Markierung der angegebenen Ressource.

Ressource

Datenverarbeitungsumgebung

arn:aws:batch:*region* :*account* :compute-environment/*compute-environment-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Job

arn:aws:batch:*region* :*account* :job/*jobId*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Auftragsdefinition

arn:aws:batch:*region* :*account* :job-definition/*definition-name* :*revision*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Auftragswarteschlange

arn:aws:batch:*region* :*account* :job-queue/*queue-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Planungsrichtlinie

arn:aws:batch:*region* :*account* :scheduling-policy/*scheduling-policy-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Bedingungsschlüssel

`aws:TagKeys` (Zeichenfolge)

Filtert Aktionen basierend auf den Tag-Schlüsseln, die in der Anforderung übergeben werden.

UpdateComputeEnvironment

Aktualisiert eine AWS Batch-Rechenumgebung.

Ressource

Datenverarbeitungsumgebung

arn:aws:batch:*region* :*account* :compute-environment/*compute-environment-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

UpdateJobQueue

Aktualisiert eine Auftragswarteschlange

Ressource

Auftragswarteschlange

arn:aws:batch:*region* :*account* :job-queue/*queue-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Planungsrichtlinie

arn:aws:batch:*region* :*account* :scheduling-policy/*scheduling-policy-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

UpdateSchedulingPolicy

Aktualisiert eine Planungsrichtlinie.

Ressource

Planungsrichtlinie

arn:aws:batch:*region* :*account* :scheduling-policy/*scheduling-policy-name*

Bedingungsschlüssel

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

Bedingungsschlüssel für AWS Batch API-Aktionen

AWS Batch definiert die folgenden Bedingungsschlüssel, die im `Condition` Element einer IAM-Richtlinie verwendet werden. Sie können diese Schlüssel verwenden, um die Bedingungen zu verfeinern, für die die Richtlinienanweisung gilt. Informationen zum Anzeigen der globalen Bedingungsschlüssel, die für alle Services verfügbar sind, finden Sie unter Verfügbare globale Bedingungsschlüssel im IAM-Benutzerhandbuch.

`batch:AWSLogsCreateGroup` (Boolean)

Wenn dieser Parameter wahr ist, `awslogs-group` wird für die Protokolle erstellt.

`batch:AWSLogsGroup` (Zeichenfolge)

Die `awslogs` Gruppe, in der sich die Protokolle befinden.

`batch:AWSLogsRegion` (Zeichenfolge)

Die AWS-Region, an die die Protokolle gesendet werden.

`batch:AWSLogsStreamPrefix` (Zeichenfolge)

Das Präfix des `awslogs` Protokollstreams.

`batch:Image` (Zeichenfolge)

Das Docker-Image, das zum Starten eines Auftrags verwendet wird.

`batch:LogDriver` (Zeichenfolge)

Der für den Auftrag verwendete Protokolltreiber.

`batch:Privileged` (Boolean)

Wenn dieser Parameter „true“ ist, erhält der Container für den Auftrag erhöhte Berechtigungen auf der Host-Container-Instance (ähnlich wie der Root-Benutzer).

`aws:ResourceTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die der Ressource zugeordnet sind.

`aws:RequestTag/${TagKey}` (Zeichenfolge)

Filtert Aktionen basierend auf den Tags, die in der Anforderung übergeben werden.

`batch:ShareIdentifier` (Zeichenfolge)

Filtert Aktionen basierend auf dem an gesendeten `shareIdentifier` ParameterSubmitJob.

`aws:TagKeys` (Zeichenfolge)

Filtert Aktionen basierend auf den Tag-Schlüsseln, die in der Anforderung übergeben werden.

`batch:User` (Zeichenfolge)

Der Benutzername oder die numerische Benutzer-ID (uid), die im Container für den Auftrag verwendet werden soll.

Beispielrichtlinien

Die folgenden Beispiele zeigen Richtlinienanweisungen, mit denen Sie die Berechtigungen von Benutzern für steuern könnenAWS Batch.

Beispiele

- Schreibgeschützter Zugriff
- Beschränken auf POSIX-Benutzer, Docker-Image, Berechtigungsstufe und Rolle bei der Auftragsübermittlung
- Bei der Auftragsübermittlung auf das Präfix der Auftragsdefinition beschränken
- Auf die Auftragswarteschlange beschränken
- Verweigerungsaktion, wenn Bedingung alle Schlüssel mit Zeichenfolgen übereinstimmen
- Verweigerungsaktion, wenn Bedingungsschlüssel mit Zeichenfolgen übereinstimmen
- Verwenden des batch:ShareIdentifier Bedingungsschlüssels

Schreibgeschützter Zugriff

Die folgende Richtlinie gewährt Benutzern Berechtigungen zur Verwendung aller AWS Batch API-Aktionen mit einem Namen, der mit `Describe` und beginnt`List`.

Sofern ihnen keine andere Anweisung die Berechtigung dazu erteilt, haben Benutzer keine Berechtigung, Aktionen für die Ressourcen auszuführen. Standardmäßig wird ihnen die Berechtigung zur Verwendung von API-Aktionen verweigert.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "batch:Describe*",
                "batch:List*"
            ],
            "Resource": "*"
        }
    ]
}
```

Beschränken auf POSIX-Benutzer, Docker-Image, Berechtigungsstufe und Rolle bei der Auftragsübermittlung

Die folgende Richtlinie ermöglicht es einem POSIX-Benutzer, seinen eigenen Satz eingeschränkter Auftragsdefinitionen zu verwalten.

Verwenden Sie die erste und zweite Anweisung, um jeden Auftragsdefinitionsnamen, dessen Name das Präfix *JobDefA_* hat, zu registrieren und abzumelden.

Die erste Anweisung verwendet auch bedingte Kontextschlüssel zum Beschränken des POSIX-Benutzers, des privilegierten Status und der Container-Image-Werte innerhalb der `containerProperties` einer Auftragsdefinition. Weitere Informationen finden Sie unter RegisterJobDefinition in der AWS Batch-API-Referenz. In diesem Beispiel können Auftragsdefinitionen nur registriert werden, wenn der POSIX-Benutzer auf festgelegt ist`nobody`. Das privilegierte Flag ist auf gesetzt`false`. Zuletzt ist das Image `myImage` in einem Amazon-ECR-Repository auf gesetzt.

> **Important**
>
> Docker löst den `user` Parameter innerhalb des Container-Images zu diesem Benutzer `uid` auf. In den meisten Fällen befindet sich dies in der `/etc/passwd` Datei im Container-Image. Diese Namensauflösung kann vermieden werden, indem sowohl in der Auftragsdefinition als auch in allen zugehörigen IAM-Richtlinien direkte `uid` Werte verwendet werden. Sowohl die AWS Batch API-Operationen als auch die bedingten `batch:User` IAM-Schlüssel unterstützen numerische Werte.

Verwenden Sie die dritte Anweisung, um auf eine bestimmte Rolle auf eine Auftragsdefinition zu beschränken.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "batch:RegisterJobDefinition"
            ],
            "Resource": [
                "arn:aws:batch:<aws_region>:<aws_account_id>:job-definition/JobDefA_*"
            ],
            "Condition": {
                "StringEquals": {
                    "batch:User": [
                        "nobody"
                    ],
                    "batch:Image": [
                        "<aws_account_id>.dkr.ecr.<aws_region>.amazonaws.com/myImage"
                    ]
                },
                "Bool": {
                    "batch:Privileged": "false"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "batch:DeregisterJobDefinition"
            ],
            "Resource": [
                "arn:aws:batch:<aws_region>:<aws_account_id>:job-definition/JobDefA_*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
```

```
                "arn:aws:iam::<aws_account_id>:role/MyBatchJobRole"
            ]
        }
    ]
}
```

Bei der Auftragsübermittlung auf das Präfix der Auftragsdefinition beschränken

Verwenden Sie die folgende Richtlinie, um Aufträge an jede Auftragswarteschlange mit einem beliebigen Auftragsdefinitionsnamen zu senden, der mit *JobDefA* beginnt.

> **Important**
>
> Beim Festlegen des Zugriffs für das Absenden von Aufträgen auf Ressourcenebene müssen Sie die Ressourcentypen der Auftragswarteschlange und der Auftragsdefinition angeben.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:SubmitJob"
      ],
      "Resource": [
        "arn:aws:batch:<aws_region>:<aws_account_id>:job-definition/JobDefA_*",
        "arn:aws:batch:<aws_region>:<aws_account_id>:job-queue/*"
      ]
    }
  ]
}
```

Auf die Auftragswarteschlange beschränken

Verwenden Sie die folgende Richtlinie, um Aufträge an eine bestimmte Auftragswarteschlange mit dem Namen queue1 mit einem beliebigen Auftragsdefinitionsnamen zu senden.

> **Important**
>
> Beim Festlegen des Zugriffs für das Absenden von Aufträgen auf Ressourcenebene müssen Sie die Ressourcentypen der Auftragswarteschlange und der Auftragsdefinition angeben.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:SubmitJob"
      ],
      "Resource": [
        "arn:aws:batch:<aws_region>:<aws_account_id>:job-definition/*",
        "arn:aws:batch:<aws_region>:<aws_account_id>:job-queue/queue1"
      ]
    }
  ]
}
```

Verweigerungsaktion, wenn Bedingung alle Schlüssel mit Zeichenfolgen übereinstimmen

Die folgende Richtlinie verweigert den Zugriff auf den RegisterJobDefinition -API-Vorgang, wenn sowohl der -Bedingungsschlüssel `batch:Image` (Container-Image-ID) „*`string1`“ als auch der -Bedingungsschlüssel `batch:LogDriver` (Container-Protokolltreiber) „*`string2`“ ist. AWS Batch wertet Bedingungsschlüssel für jeden Container aus. Wenn sich ein Auftrag über mehrere Container erstreckt, z. B. einen parallelen Auftrag mit mehreren Knoten, ist es möglich, dass die Container unterschiedliche Konfigurationen haben. Wenn mehrere Bedingungsschlüssel in einer Anweisung ausgewertet werden, werden sie mithilfe von AND Logik kombiniert. Wenn also einer der mehreren Bedingungsschlüssel nicht mit einem Container übereinstimmt, wird der `Deny` Effekt für diesen Container nicht angewendet. Stattdessen kann ein anderer Container im selben Auftrag abgelehnt werden.

Eine Liste der Bedingungsschlüssel für AWS Batchfinden Sie unter Bedingungsschlüssel für AWS Batch in der Service-Autorisierungs-Referenz. Mit Ausnahme von können alle `batch` `batch:ShareIdentifier`Bedingungsschlüssel auf diese Weise verwendet werden. Der

`batch:ShareIdentifier` Bedingungsschlüssel ist für einen Auftrag definiert, nicht für eine Auftragsdefinition.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:RegisterJobDefinition"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Deny",
      "Action": "batch:RegisterJobDefinition",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "batch:Image": "string1",
          "batch:LogDriver": "string2"
        }
      }
    }
  ]
}
```

Verweigerungsaktion, wenn Bedingungsschlüssel mit Zeichenfolgen übereinstimmen

Die folgende Richtlinie verweigert den Zugriff auf den RegisterJobDefinition -API-Vorgang, wenn entweder der -Bedingungsschlüssel `batch:Image` (Container-Image-ID) „*string1*“ oder der -Bedingungsschlüssel `batch:LogDriver` (Container-Protokolltreiber) „*string2*“ ist. Wenn sich ein Auftrag über mehrere Container erstreckt, z. B. einen parallelen Auftrag mit mehreren Knoten, ist es möglich, dass die Container unterschiedliche Konfigurationen haben. Wenn mehrere Bedingungsschlüssel in einer Anweisung ausgewertet werden, werden sie mithilfe von AND Logik kombiniert. Wenn also einer der verschiedenen Bedingungsschlüssel nicht mit einem Container

übereinstimmt, wird der `Deny` Effekt für diesen Container nicht angewendet. Stattdessen kann ein anderer Container im selben Auftrag abgelehnt werden.

Eine Liste der Bedingungsschlüssel für AWS Batchfinden Sie unter Bedingungsschlüssel für AWS Batch in der Service-Autorisierungs-Referenz. Mit Ausnahme von können alle `batch` `batch:ShareIdentifier`Bedingungsschlüssel auf diese Weise verwendet werden. (Der `batch:ShareIdentifier` Bedingungsschlüssel ist für einen Auftrag definiert, nicht für eine Auftragsdefinition.)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:RegisterJobDefinition"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Deny",
      "Action": [
        "batch:RegisterJobDefinition"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "batch:Image": [
            "string1"
          ]
        }
      }
    },
    {
      "Effect": "Deny",
      "Action": [
        "batch:RegisterJobDefinition"
      ],
      "Resource": [
```

```
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "batch:LogDriver": [
            "string2"
          ]
        }
      }
    }
  ]
}
```

Verwenden des **batch:ShareIdentifier** Bedingungsschlüssels

Verwenden Sie die folgende Richtlinie, um Aufträge, die die `jobDefA` Auftragsdefinition verwenden, an die `jobqueue1` Auftragswarteschlange mit der `lowCpu` Freigabe-ID zu senden.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:SubmitJob"
      ],
      "Resource": [
        "arn:aws::batch:<aws_region>:<aws_account_id>:job-definition/JobDefA",
        "arn:aws::batch:<aws_region>:<aws_account_id>:job-queue/jobqueue1"
      ],
      "Condition": {
        "StringEquals": {
          "batch:ShareIdentifier": [
            "lowCpu"
          ]
        }
      }
    }
  ]
}
```

AWS Batch-verwaltete Richtlinie

AWS Batch bietet eine verwaltete Richtlinie, die Sie Benutzern anfügen können und die die Berechtigung zur Verwendung von AWS Batch Ressourcen und API-Operationen bietet. Sie können diese Richtlinie direkt anwenden oder als Ausgangspunkt zur Erstellung eigener Richtlinien verwenden. Weitere Informationen zu den in diesen Richtlinien erwähnten API-Operationen finden Sie unter Aktionen in der APIAWS Batch-Referenz zu .

AWSBatchFullAccess

Diese Richtlinie gewährt den vollständigen Administratorzugriff auf AWS Batch.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "batch:*",
        "cloudwatch:GetMetricStatistics",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeVpcs",
        "ec2:DescribeImages",
        "ec2:DescribeLaunchTemplates",
        "ec2:DescribeLaunchTemplateVersions",
        "ecs:DescribeClusters",
        "ecs:Describe*",
        "ecs:List*",
        "eks:DescribeCluster",
        "eks:ListClusters",
        "logs:Describe*",
        "logs:Get*",
        "logs:TestMetricFilter",
        "logs:FilterLogEvents",
        "iam:ListInstanceProfiles",
        "iam:ListRoles"
      ],
      "Resource":"*"
    },
    {
```

```
      "Effect":"Allow",
      "Action":[
        "iam:PassRole"
      ],
      "Resource":[
        "arn:aws:iam::*:role/AWSBatchServiceRole",
        "arn:aws:iam::*:role/service-role/AWSBatchServiceRole",
        "arn:aws:iam::*:role/ecsInstanceRole",
        "arn:aws:iam::*:instance-profile/ecsInstanceRole",
        "arn:aws:iam::*:role/iaws-ec2-spot-fleet-role",
        "arn:aws:iam::*:role/aws-ec2-spot-fleet-role",
        "arn:aws:iam::*:role/AWSBatchJobRole*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "iam:CreateServiceLinkedRole"
      ],
      "Resource":"arn:aws:iam::*:role/*Batch*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "batch.amazonaws.com"
          }
      }
    }
  ]
}
```

Erstellen von AWS Batch IAM-Richtlinien

Sie können bestimmte IAM-Richtlinien erstellen, um die Aufrufe und Ressourcen einzuschränken, auf die Benutzer in Ihrem Konto Zugriff haben. Anschließend können Sie diese Richtlinien an Benutzer anfügen.

Wenn Sie eine Richtlinie an einen Benutzer oder eine Benutzergruppe anfügen, gewährt oder verweigert die Richtlinie den Benutzern die Berechtigung für bestimmte Aufgaben auf bestimmten Ressourcen. Weitere Informationen finden Sie unter Berechtigungen und Richtlinien im IAM-Benutzerhandbuch. Anweisungen zum Verwalten und Erstellen benutzerdefinierter IAM-Richtlinien finden Sie unter Verwalten von IAM-Richtlinien.

Amazon-ECS-Instance-Rolle

AWS Batch -Rechenumgebungen werden mit Amazon-ECS-Container-Instances gefüllt. Sie führen den Amazon-ECS-Container-Agent lokal aus. Der Amazon-ECS-Container-Agent ruft verschiedene AWS API-Operationen in Ihrem Namen auf. Daher benötigen Container-Instances, die den Agenten ausführen, eine IAM-Richtlinie und -Rolle, damit diese Services erkennen können, dass der Agent Ihnen gehört. Sie müssen eine IAM-Rolle und ein Instance-Profil für die Container-Instances erstellen, die beim Start verwendet werden sollen. Andernfalls können Sie keine Datenverarbeitungsumgebung erstellen und Container-Instances darin starten. Diese Anforderung gilt für Container-Instances, die mit oder ohne das Amazon-ECS-optimierte AMI von Amazon gestartet wurden. Weitere Informationen finden Sie unter IAM-Rolle für Amazon-ECS-Container-Instances im Entwicklerhandbuch für Amazon Elastic Container Service .

Die Amazon-ECS-Instance-Rolle und das Instance-Profil werden in der ersten Ausführung der Konsole automatisch für Sie erstellt. Sie können jedoch diese Schritte ausführen, um zu überprüfen, ob Ihr Konto bereits über die Amazon-ECS-Instance-Rolle und das Instance-Profil verfügt. In den folgenden Schritten wird auch beschrieben, wie Sie die verwaltete IAM-Richtlinie anfügen.

So prüfen Sie **`ecsInstanceRole`** in der IAM-Konsole

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Rollen aus.
3. Suchen Sie in der Liste der Rollen nach `ecsInstanceRole`. Wenn die Rolle nicht vorhanden ist, führen Sie die folgenden Schritte aus, um die Rolle zu erstellen.
    a. Wählen Sie Create Role (Rolle erstellen) aus.
    b. Wählen Sie für Vertrauenswürdige Entität die Option AWS-Service aus.
    c. Wählen Sie für Häufige Anwendungsfälle die Option EC2 aus.
    d. Wählen Sie Weiter aus.
    e. Suchen Sie unter Berechtigungsrichtlinien nach AmazonEC2ContainerServiceforEC2Role .
    f. Aktivieren Sie das Kontrollkästchen neben AmazonEC2ContainerServiceforEC2Role und wählen Sie dann Weiter aus.
    g. Geben Sie unter Role Name (Rollenname) den Namen `ecsInstanceRole` ein und wählen Sie Create role (Rolle erstellen) aus.

> **Note**
>
> Wenn Sie die verwenden, AWS Management Console um eine Rolle für Amazon EC2 zu erstellen, erstellt die Konsole ein Instance-Profil mit demselben Namen wie die Rolle.

Alternativ können Sie die verwenden, AWS CLI um die `ecsInstanceRole` IAM-Rolle zu erstellen. Im folgenden Beispiel wird eine IAM-Rolle mit einer Vertrauensrichtlinie und einer von AWS verwalteten Richtlinie erstellt.

IAM-Rolle und Instance-Profil erstellen (AWS CLI)

1. Erstellen Sie die folgende Vertrauensrichtlinie und speichern Sie sie in einer Textdatei mit dem Namen `ecsInstanceRole-role-trust-policy.json`.

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

2. Verwenden Sie den Befehl create-role, um die `ecsInstanceRole` Rolle zu erstellen. Geben Sie den Speicherort der Vertrauensrichtliniendatei im `assume-role-policy-document` Parameter an.

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

   Nachfolgend finden Sie eine Beispielantwort.

```
{
    "Role": {
```

```
         "Path": "/",
         "RoleName: "ecsInstanceRole",
         "RoleId": "AROAT46P5RDIY4EXAMPLE",
         "Arn": "arn:aws:iam::123456789012:role/ecsInstanceRole".
        "CreateDate": "2022-12-12T23:46:37.247Z",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service: "ec2.amazonaws.com"
                    }
                    "Action": "sts:AssumeRole",
                 }
              ]
          }
        }
```

3. Verwenden Sie den create-instance-profile Befehl , um ein Instance-Profil mit dem Namen zu erstellen`ecsInstanceRole`.

> **Note**
>
> Sie müssen Rollen und Instance-Profile als separate Aktionen in der AWS CLI und API erstellenAWS.

```
$ aws iam create-instance-profile --instance-profile-name ecsInstanceRole
```

Nachfolgend finden Sie eine Beispielantwort.

```
{
    "InstanceProfile": {
        "Path": "/",
        "InstanceProfileName": "ecsInstanceRole",
        "InstanceProfileId": "AIPAT46P5RDITREXAMPLE",
        "Arn": "arn:aws:iam::123456789012:instance-profile/ecsInstanceRole",
        "CreateDate": "2022-06-30T23:53:34.093Z",
        "Roles": [],    }
}
```

4. Verwenden Sie den Befehl add-role-to-instance-profile, um die `ecsInstanceRole` Rolle zum `ecsInstanceRole` Instance-Profil hinzuzufügen.

```
aws iam add-role-to-instance-profile \
    --role-name ecsInstanceRole --instance-profile-name ecsInstanceRole
```

5. Verwenden Sie den attach-role-policy Befehl , um die `AmazonEC2ContainerServiceforEC2Role` AWS verwaltete Richtlinie an die `ecsInstanceRole` Rolle anzuhängen.

```
$ aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role \
    --role-name ecsInstanceRole
```

Amazon EC2-Spot-Flottenrolle

Wenn Sie eine verwaltete Datenverarbeitungsumgebung erstellen, die Amazon EC2 Spot Fleet Instances verwendet, müssen Sie die `AmazonEC2SpotFleetTaggingRole` Richtlinie erstellen. Diese Richtlinie gewährt der Spot-Flotte die Berechtigung, Instances in Ihrem Namen zu starten, zu markieren und zu beenden. Legen Sie die Rolle in Ihrer Spot-Flottenanforderung fest. Sie müssen auch über die AWSServiceRoleForEC2SpotFleet serviceverknüpften Rollen AWSServiceRoleForEC2Spot und für Amazon EC2 Spot und Spot Fleet verfügen. Verwenden Sie die folgende Anweisung, um alle diese Rollen zu erstellen. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen und Erstellen einer Rolle zum Delegieren von Berechtigungen an einen -AWSService im IAM-Benutzerhandbuch.

Themen

- Erstellen von Amazon EC2-Spot-Flottenrollen in der AWS Management Console
- Erstellen von Amazon EC2-Spot-Flottenrollen mit der AWS CLI

Erstellen von Amazon EC2-Spot-Flottenrollen in der AWS Management Console

So erstellen Sie die serviceverknüpfte **`AmazonEC2SpotFleetTaggingRole`** IAM-Rolle für Amazon EC2 Spot Fleet

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie für Zugriffsverwaltung die Option Rollen ,
3. Wählen Sie für Rollen die Option Rolle erstellen aus.
4. Wählen Sie unter Vertrauenswürdige Entität für Vertrauenswürdigen Entitätstyp auswählen die Option ausAWS-Service.
5. Wählen Sie für Anwendungsfälle für andere AWS-Services EC2 und dann EC2 – Spot-Flotten-Markierung aus.
6. Wählen Sie Weiter aus.
7. Überprüfen Sie unter Berechtigungsrichtlinien für den Richtliniennamen `AmazonEC2SpotFleetTaggingRole`.
8. Wählen Sie Weiter aus.
9. Für Name, überprüfen und erstellen Sie :
    a. Geben Sie unter Rollenname einen Namen ein, um die Rolle zu identifizieren.
    b. Geben Sie unter Beschreibung eine kurze Erklärung für die Richtlinie ein.
    c. (Optional) Wählen Sie für Schritt 1: Vertrauenswürdige Entitäten auswählen die Option Bearbeiten aus, um den Code zu ändern.
    d. (Optional) Wählen Sie für Schritt 2: Hinzufügen von Berechtigungen die Option Bearbeiten aus, um den Code zu ändern.
    e. (Optional) Wählen Sie für Tags hinzufügen die Option Tag hinzufügen aus, um der Ressource Tags hinzuzufügen.
    f. Wählen Sie Rolle erstellen aus.

> **Note**
>
> In der Vergangenheit gab es zwei -verwaltete Richtlinien für die Amazon EC2-Spot-Flottenrolle.

- AmazonEC2SpotFleetRole: Dies ist die ursprüngliche verwaltete Richtlinie für die Spot-Flotten-Rolle. Wir empfehlen jedoch nicht mehr, es mit zu verwendenAWS Batch. Diese Richtlinie unterstützt kein Spot-Instance-Tagging in Datenverarbeitungsumgebungen, was für die Verwendung der `AWSServiceRoleForBatch` serviceverknüpften Rolle erforderlich ist. Wenn Sie zuvor eine Spot-Flotten-Rolle mit dieser Richtlinie erstellt haben, wenden Sie die neue empfohlene Richtlinie auf diese Rolle an. Weitere Informationen finden Sie unter Spot-Instances wurden bei der Erstellung nicht markiert.
- AmazonEC2SpotFleetTaggingRole: Diese Rolle bietet alle erforderlichen Berechtigungen zum Markieren von Amazon EC2-Spot-Instances. Verwenden Sie diese Rolle, um das Spot-Instance-Tagging in AWS Batch-Datenverarbeitungsumgebungen zuzulassen.

Erstellen von Amazon EC2-Spot-Flottenrollen mit der AWS CLI

So erstellen Sie die AmazonEC2SpotFleetTaggingRole-IAM-Rolle für Ihre Spot-Flotten-Rechenumgebungen

1. Führen Sie den folgenden Befehl mit der ausAWS CLI.

```
$ aws iam create-role --role-name AmazonEC2SpotFleetTaggingRole \
    --assume-role-policy-document '{
    "Version":"2012-10-17",
    "Statement":[
       {
         "Sid":"",
         "Effect":"Allow",
         "Principal": {
           "Service":"spotfleet.amazonaws.com"
          },
          "Action":"sts:AssumeRole"
        }
      ]
    }'
```

2. Um die von AmazonEC2SpotFleetTaggingRole verwaltete IAM-Richtlinie an Ihre AmazonEC2SpotFleetTaggingRole-Rolle anzuhängen, führen Sie den folgenden Befehl mit der ausAWS CLI.

```
$ aws iam attach-role-policy \
```

```
  --policy-arn \
    arn:aws:iam::aws:policy/service-role/AmazonEC2SpotFleetTaggingRole \
  --role-name \
    AmazonEC2SpotFleetTaggingRole
```

So erstellen Sie die serviceverknüpfte **`AWSServiceRoleForEC2Spot`** IAM-Rolle für Amazon EC2 Spot

> **Note**
>
> Wenn die serviceverknüpfte `AWSServiceRoleForEC2Spot` IAM-Rolle bereits vorhanden ist, wird eine Fehlermeldung ähnlich der folgenden angezeigt.
>
> ```
> An error occurred (InvalidInput) when calling the CreateServiceLinkedRole
>  operation:
> Service role name AWSServiceRoleForEC2Spot has been taken in this account,
>  please try a different suffix.
> ```

- Führen Sie den folgenden Befehl mit der ausAWS CLI.

```
$ aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
```

So erstellen Sie die serviceverknüpfte **`AWSServiceRoleForEC2SpotFleet`** IAM-Rolle für Amazon EC2 Spot Fleet

> **Note**
>
> Wenn die serviceverknüpfte `AWSServiceRoleForEC2SpotFleet` IAM-Rolle bereits vorhanden ist, wird eine Fehlermeldung ähnlich der folgenden angezeigt.
>
> ```
> An error occurred (InvalidInput) when calling the CreateServiceLinkedRole
>  operation:
> Service role name AWSServiceRoleForEC2SpotFleet has been taken in this account,
>  please try a different suffix.
> ```

- Führen Sie den folgenden Befehl mit der ausAWS CLI.

```
$ aws iam create-service-linked-role --aws-service-name spotfleet.amazonaws.com
```

EventBridge IAM-Rolle

Amazon EventBridge stellt einen Stream von Systemereignissen in nahezu Echtzeit bereit, der Änderungen an -AWSRessourcen beschreibt. -AWS BatchAufträge sind als EventBridge Ziele verfügbar. Mit einfachen, schnell eingerichteten Regeln können Sie Ereignisse zuordnen und in Reaktion darauf AWS Batch-Aufträge ausführen. Bevor Sie AWS Batch Aufträge mit EventBridge Regeln und Zielen einreichen können, EventBridge müssen Sie über die Berechtigungen verfügen, AWS Batch Aufträge in Ihrem Namen auszuführen.

> **Note**
>
> Wenn Sie in der EventBridge Konsole eine Regel erstellen, die eine AWS Batch-Warteschlange als Ziel angibt, können Sie diese Rolle erstellen. Ein Beispiel-Walkthrough finden Sie unte AWS Batch Aufträge als EventBridge Ziele. Sie können die EventBridge Rolle manuell mithilfe der IAM-Konsole erstellen. Anweisungen finden Sie unter Erstellen einer Rolle mit benutzerdefinierten Vertrauensrichtlinien (Konsole) im IAM-Benutzerhandbuch.

Die Vertrauensstellung für Ihre EventBridge IAM-Rolle muss dem `events.amazonaws.com` Service-Prinzipal die Möglichkeit geben, die Rolle zu übernehmen.

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

Stellen Sie sicher, dass die Richtlinie, die Ihrer EventBridge IAM-Rolle zugeordnet ist, `batch:SubmitJob` Berechtigungen für Ihre Ressourcen zulässt. Im folgenden Beispiel AWS Batch stellt die -`AWSBatchServiceEventTargetRole`verwaltete Richtlinie bereit, um diese Berechtigungen bereitzustellen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:SubmitJob"
       ],
      "Resource": "*"
    }
  ]
}
```

AWS Batch Ereignisstream für Amazon EventBridge

Sie können den AWS Batch Ereignisstream für Amazon verwenden EventBridge , um nahezu in Echtzeit Benachrichtigungen über den aktuellen Status von Aufträgen in Ihren Auftragswarteschlangen zu erhalten.

Sie können verwenden EventBridge , um weitere Einblicke in Ihren AWS Batch Service zu erhalten. Genauer gesagt können Sie damit den Fortschritt von Aufträgen überprüfen, AWS Batch benutzerdefinierte Workflows erstellen, Nutzungsberichte oder Metriken erstellen oder eigene Dashboards erstellen. Mit AWS Batch und benötigen EventBridgeSie keinen Planungs- und Überwachungscode, der kontinuierlich nach Änderungen des Auftragsstatus AWS Batch abfragt. Stattdessen können Sie Änderungen des AWS Batch Auftragsstatus asynchron mit einer Vielzahl von Amazon- EventBridge Zielen verarbeiten. Dazu gehören AWS Lambda, Amazon Simple Queue Service, Amazon Simple Notification Service oder Amazon Kinesis Data Streams.

Ereignisse aus dem AWS Batch Ereignisstream werden garantiert, dass sie mindestens einmal zugestellt werden. Falls doppelte Ereignisse übermittelt werden, enthält das Ereignis ausreichend Informationen, um Duplikate zu erkennen. Auf diese Weise können Sie den Zeitstempel des Ereignisses und den Auftragsstatus vergleichen.

AWS Batch -Aufträge sind als EventBridge Ziele verfügbar. Mithilfe einfacher Regeln können Sie Ereignisse abgleichen und als Reaktion darauf AWS Batch Aufträge senden. Weitere Informationen finden Sie unter Was ist EventBridge? im Amazon- EventBridge Benutzerhandbuch. Sie können auch verwenden, EventBridge um automatisierte Aktionen zu planen, die sich zu bestimmten Zeiten selbst auslösen, indem Sie - cron oder -Ratenausdrücke verwenden. Weitere Informationen finden Sie unter Erstellen einer Amazon- EventBridge Regel, die nach einem Zeitplan ausgeführt wird im Amazon- EventBridge Benutzerhandbuch. Ein Beispiel-Walkthrough finden Sie unte AWS Batch Aufträge als EventBridge Ziele. Informationen zur Verwendung des EventBridge Schedulers finden Sie unter Einrichten von Amazon EventBridge Scheduler im Amazon EventBridge -Benutzerhandbuch.

Themen

AWS Batch Events

AWS Batch sendet Ereignisse zur Änderung des Auftragsstatus an EventBridge. AWS Batch verfolgt den Status Ihrer Aufträge. Wenn sich der Status eines zuvor übermittelten Auftrags ändert, wird ein Ereignis aufgerufen. Zum Beispiel, wenn ein Auftrag mit dem `RUNNING` Status in den `FAILED` Status wechselt. Diese Ereignisse sind als Auftragsstatus-Änderungsereignisse klassifiziert.

> **Note**
>
> AWS Batch kann in Zukunft weitere Ereignistypen, Quellen und Details hinzufügen. Wenn Sie Ereignis-JSON-Daten programmgesteuert deserialisieren, stellen Sie sicher, dass Ihre Anwendung bereit ist, unbekannte Eigenschaften zu verarbeiten. Dies soll Probleme vermeiden, wenn und wenn diese zusätzlichen Eigenschaften hinzugefügt werden.

Auftragsstatus-Änderungsereignisse

Jedes Mal, wenn ein vorhandener (zuvor übermittelter) Auftrag den Status ändert, wird ein Ereignis erstellt. Weitere Informationen zu AWS Batch Auftragsstatus finden Sie unter Auftragsstatus.

> **Note**
>
> Ereignisse werden für die erste Auftragsübermittlung nicht erstellt.

Example Auftragsstatus-Änderungsereignis

Ereignisse zur Änderung des Auftragsstatus werden im folgenden Format bereitgestellt. Der `detail` Abschnitt ähnelt dem JobDetail Objekt, das von einer DescribeJobs API-Operation in der AWS Batch API-Referenz zurückgegeben wird. Weitere Informationen zu EventBridge Parametern finden Sie unter Ereignisse und Ereignismuster im Amazon- EventBridge Benutzerhandbuch.

```
{
    "version": "0",
    "id": "c8f9c4b5-76e5-d76a-f980-7011e206042b",
    "detail-type": "Batch Job State Change",
    "source": "aws.batch",
    "account": "123456789012",
    "time": "2022-01-11T23:36:40Z",
    "region": "us-east-1",
```

```
    "resources": [
        "arn:aws:batch:us-east-1:123456789012:job/4c7599ae-0a82-49aa-ba5a-4727fcce14a8"
    ],
    "detail": {
        "jobArn": "arn:aws:batch:us-east-1:123456789012:job/4c7599ae-0a82-49aa-
ba5a-4727fcce14a8",
        "jobName": "event-test",
        "jobId": "4c7599ae-0a82-49aa-ba5a-4727fcce14a8",
        "jobQueue": "arn:aws:batch:us-east-1:123456789012:job-queue/
PexjEHappyPathCanary2JobQueue",
        "status": "RUNNABLE",
        "attempts": [],
        "createdAt": 1641944200058,
        "retryStrategy": {
            "attempts": 2,
            "evaluateOnExit": []
        },
        "dependsOn": [],
        "jobDefinition": "arn:aws:batch:us-east-1:123456789012:job-definition/first-
run-job-definition:1",
        "parameters": {},
        "container": {
            "image": "137112412989.dkr.ecr.us-east-1.amazonaws.com/amazonlinux:latest",
            "command": [
                "sleep",
                "600"
            ],
            "volumes": [],
            "environment": [],
            "mountPoints": [],
            "ulimits": [],
            "networkInterfaces": [],
            "resourceRequirements": [
                {
                    "value": "2",
                    "type": "VCPU"
                }, {
                    "value": "256",
                    "type": "MEMORY"
                }
            ],
            "secrets": []
        },
        "tags": {
```

```
            "resourceArn": "arn:aws:batch:us-
east-1:123456789012:job/4c7599ae-0a82-49aa-ba5a-4727fcce14a8"
        },
        "propagateTags": false,
        "platformCapabilities": []
    }
}
```

Ereignisse in der Auftragswarteschlange blockiert

Jedes Mal, wenn einen Auftrag im `RUNNABLE` Status AWS Batch erkennt und somit eine Warteschlange blockiert, wird ein Ereignis in Amazon CloudWatch Events erstellt. Weitere Informationen zu unterstützten Ursachen für blockierte Warteschlangen finden Sie unter Beispielmeldungen für blockierte Auftragswarteschlangen . Derselbe Grund ist auch im `statusReason` Feld in der `DescribeJobs` API-Aktion verfügbar.

Example Auftragsstatus-Änderungsereignis

Ereignisse zur Änderung des Auftragsstatus werden im folgenden Format bereitgestellt. Der `detail` Abschnitt ähnelt dem JobDetail Objekt, das von einer DescribeJobs API-Operation in der AWS Batch API-Referenz zurückgegeben wird. Weitere Informationen zu EventBridge Parametern finden Sie unter Ereignisse und Ereignismuster im Amazon- EventBridge Benutzerhandbuch.

```
{
    "version": "0",
    "id": "c8f9c4b5-76e5-d76a-f980-7011e206042b",
    "detail-type": "Batch Job Queue Blocked",
    "source": "aws.batch",
    "account": "123456789012",
    "time": "2022-01-11T23:36:40Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:batch:us-east-1:123456789012:job/4c7599ae-0a82-49aa-
ba5a-4727fcce14a8",
        "arn:aws:batch:us-east-1:123456789012:job-queue/PexjEHappyPathCanary2JobQueue"
    ],
    "detail": {
        "jobArn": "arn:aws:batch:us-east-1:123456789012:job/4c7599ae-0a82-49aa-
ba5a-4727fcce14a8",
        "jobName": "event-test",
        "jobId": "4c7599ae-0a82-49aa-ba5a-4727fcce14a8",
```

```
        "jobQueue": "arn:aws:batch:us-east-1:123456789012:job-queue/
PexjEHappyPathCanary2JobQueue",
        "status": "RUNNABLE",
        "statusReason": "blocked-reason"
        "attempts": [],
        "createdAt": 1641944200058,
        "retryStrategy": {
            "attempts": 2,
            "evaluateOnExit": []
        },
        "dependsOn": [],
        "jobDefinition": "arn:aws:batch:us-east-1:123456789012:job-definition/first-
run-job-definition:1",
        "parameters": {},
        "container": {
            "image": "137112412989.dkr.ecr.us-east-1.amazonaws.com/amazonlinux:latest",
            "command": [
                "sleep",
                "600"
            ],
            "volumes": [],
            "environment": [],
            "mountPoints": [],
            "ulimits": [],
            "networkInterfaces": [],
            "resourceRequirements": [
                {
                    "value": "2",
                    "type": "VCPU"
                }, {
                    "value": "256",
                    "type": "MEMORY"
                }
            ],
            "secrets": []
        },
        "tags": {
            "resourceArn": "arn:aws:batch:us-
east-1:123456789012:job/4c7599ae-0a82-49aa-ba5a-4727fcce14a8"
        },
        "propagateTags": false,
        "platformCapabilities": []
    }
```

```
}
```

Verwenden von AWS Benutzerbenachrichtigungen mit AWS Batch

Sie können AWS Benutzerbenachrichtigungen verwenden, um Übermittlungskanäle einzurichten, um über AWS Batch Ereignisse benachrichtigt zu werden. Sie erhalten eine Benachrichtigung, wenn ein Ereignis einer von Ihnen angegebenen Regel entspricht. Sie können Benachrichtigungen für Ereignisse über mehrere Kanäle erhalten, einschließlich E-Mail-, AWS Chatbot-Chat- oder AWS Console Mobile Application-Push-Benachrichtigungen. Benachrichtigungen werden auch im Console Notifications Center angezeigt. Die Funktion für Benutzerbenachrichtigungen unterstützt eine Aggregation, wodurch die Anzahl der Benachrichtigungen, die Sie bei bestimmten Ereignissen erhalten, reduziert werden kann.

So konfigurieren Sie Benutzerbenachrichtigungen in AWS Batch:

1. Öffnen Sie die AWS Batch -Konsole.
2. Wählen Sie Dashboard.
3. Wählen Sie Benachrichtigungen konfigurieren aus.
4. Wählen Sie unter AWS Benutzerbenachrichtigungen die Option Benachrichtigungskonfiguration erstellen aus.

Weitere Informationen zum Konfigurieren und Anzeigen von Benutzerbenachrichtigungen finden Sie unter Erste Schritte mit AWS Benutzerbenachrichtigungen.

AWS Batch Aufträge als EventBridge Ziele

Amazon EventBridge stellt einen Stream von Systemereignissen in nahezu Echtzeit bereit, der Änderungen an Amazon Web Services-Ressourcen beschreibt. In der Regel sind AWS Batch auf Amazon Elastic Container Service-, Amazon Elastic Kubernetes Service- und AWS Fargate-Aufträgen als EventBridge Ziele verfügbar. Mithilfe einfacher Regeln können Sie Ereignisse abgleichen und als Reaktion darauf AWS Batch Aufträge senden. Weitere Informationen finden Sie unter Was ist EventBridge? im Amazon- EventBridge Benutzerhandbuch.

Sie können auch verwenden, EventBridge um automatisierte Aktionen zu planen, die zu bestimmten Zeiten mithilfe von - cron oder -Ratenausdrücken aufgerufen werden. Weitere Informationen finden Sie unter Erstellen einer Amazon- EventBridge Regel, die nach einem Zeitplan ausgeführt wird im Amazon- EventBridge Benutzerhandbuch.

Informationen zum Erstellen einer Regel, die ausgeführt wird, wenn ein Ereignis einem Ereignismuster entspricht, finden Sie unter Erstellen von Amazon- EventBridge Regeln, die auf Ereignisse reagieren im Amazon- EventBridge Benutzerhandbuch.

Zu den häufigsten Anwendungsfällen für AWS Batch Aufträge als EventBridge Ziel gehören die folgenden Anwendungsfälle:

- Ein geplanter Auftrag erfolgt in regelmäßigen Zeitintervallen. Beispielsweise tritt ein cron Auftrag nur in Zeiten mit geringer Auslastung auf, wenn Amazon EC2 Spot Instances kostengünstiger sind.
- Ein - AWS Batch Auftrag wird als Reaktion auf einen API-Vorgang ausgeführt, der in protokolliert ist CloudTrail. Beispielsweise wird ein Auftrag übermittelt, wenn ein Objekt in einen angegebenen Amazon S3-Bucket hochgeladen wird. Jedes Mal, wenn dies geschieht, übergibt der EventBridge Eingabe-Transformator den Bucket und den Schlüsselnamen des Objekts an AWS Batch Parameter.

> **Note**
>
> In diesem Szenario müssen sich alle zugehörigen AWS Ressourcen in derselben Region befinden. Dazu gehören Ressourcen wie der Amazon S3-Bucket, EventBridge die Regel und die CloudTrail Protokolle.

Bevor Sie AWS Batch Aufträge mit EventBridge Regeln und Zielen einreichen können, benötigt der EventBridge Service mehrere Berechtigungen zum Ausführen von AWS Batch Aufträgen. Wenn Sie in der EventBridge Konsole eine Regel erstellen, die einen - AWS Batch Auftrag als Ziel angibt, können Sie diese Rolle auch erstellen. Weitere Informationen über die erforderlichen Service-Prinzipal- und IAM;-Berechtigungen für dies Rolle finden Sie unter EventBridge IAM-Rolle.

Erstellen eines geplanten AWS Batch Auftrags

Im folgenden Verfahren wird beschrieben, wie Sie einen geplanten AWS Batch Auftrag und die erforderliche EventBridge IAM-Rolle erstellen.

So erstellen Sie einen geplanten AWS Batch Auftrag mit EventBridge

> **Note**
>
> Dieses Verfahren funktioniert für alle AWS Batch auf Amazon-ECS-, Amazon-EKS- und AWS Fargate-Aufträgen.

1. Öffnen Sie die Amazon- EventBridge Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie in der Navigationsleiste die zu AWS-Region verwendende aus.
3. Wählen Sie im Navigationsbereich Rules aus.
4. Wählen Sie Regel erstellen aus.
5. Geben Sie unter Name einen eindeutigen Namen für Ihre Datenverarbeitungsumgebung an. Der Name kann bis zu 64 Zeichen umfassen. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.

   > **Note**
   >
   > Eine Regel darf nicht denselben Namen wie eine andere Regel in derselben Region und auf demselben Event Bus haben.

6. (Optional) Geben Sie unter Beschreibung eine Beschreibung für die Regel ein.
7. Wählen Sie für Event Bus den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie Standard aus. Wenn ein AWS-Service in Ihrem Konto ein Ereignis ausgibt, wird es immer an den Standard-Event-Bus Ihres Kontos weitergeleitet.
8. (Optional) Deaktivieren Sie die Regel für den ausgewählten Bus, wenn Sie die Regel nicht sofort ausführen möchten.
9. Wählen Sie unter Rule type (Regeltyp) die Option Schedule (Zeitplan) aus.
10. Wählen Sie Weiter, um die Regel zu erstellen, oder Weiter aus.
11. Gehen Sie bei Schedule pattern (Zeitplanmuster) wie folgt vor:

    - Wählen Sie Ein detaillierter Zeitplan, der zu einer bestimmten Zeit ausgeführt wird, z. B. um 8:00 Uhr aus. PST jeden ersten Montag im Monat und geben Sie dann einen Cron-Ausdruck ein. Weitere Informationen finden Sie unter Cron-Ausdrücke im Amazon- EventBridge Benutzerhandbuch.

- Wählen Sie Einen Zeitplan aus, der regelmäßig ausgeführt wird, z. B. alle 10 Minuten, und geben Sie dann einen Ratenausdruck ein.

12. Wählen Sie Weiter aus.
13. Für Target types (Zieltypen), wählen Sie AWS-Service aus.
14. Wählen Sie für Ziel auswählen die Option Batch-Auftragswarteschlange aus. Konfigurieren Sie dann Folgendes:

    - Job queue (Auftragswarteschlange): Geben Sie den Amazon-Ressourcennamen (ARN) der Auftragswarteschlange ein, in der Ihr Auftrag eingeplant werden soll.
    - Auftragsdefinition: Geben Sie den Namen und die Revision oder den vollständigen ARN der Auftragsdefinition ein, die für Ihren Auftrag verwendet werden sollen.
    - Job name (Auftragsname): Geben Sie einen Namen für Ihren Auftrag ein.
    - Array-Größe: (Optional) Geben Sie eine Array-Größe für Ihren Auftrag ein, um mehrere Kopien davon auszuführen. Weitere Informationen finden Sie unter Array-Jobs.
    - Auftragsversuche: (Optional) Geben Sie an, wie oft die Ausführung Ihres Auftrags versucht werden soll, bevor festgelegt wird, dass er fehlgeschlagen ist. Weitere Informationen finden Sie unter Automatisierte Auftragswiederholungen.
15. Bei Batch-Auftragswarteschlangen-Zieltypen EventBridge benötigt die Berechtigung zum Senden von Ereignissen an das Ziel. EventBridge kann die IAM-Rolle erstellen, die für die Ausführung Ihrer Regel erforderlich ist. Führen Sie eine der folgenden Aktionen aus:

    - Um automatisch eine IAM-Rolle zu erstellen, wählen Sie Create a new role for this specific resource (Eine neue Rolle für diese spezifische Ressource erstellen).
    - Um eine bereits erstellte IAM-Rolle zu verwenden, wählen Sie Vorhandene Rolle verwenden aus.
16. (Optional) Erweitern Sie Additional settings (Zusätzliche Einstellungen).

    a. Wählen Sie unter Zieleingabe konfigurieren aus, wie der Text aus einem Ereignis verarbeitet wird, bevor er an das Ziel übergeben wird.

    b. Geben Sie unter Höchstalter des Ereignisses das Zeitintervall für die Aufbewahrungsdauer unverarbeiteter Ereignisse an.

    c. Geben Sie unter Wiederholungsversuche die Häufigkeit ein, mit der ein Ereignis erneut versucht wird.

    d. Wählen Sie für Warteschlange für unzustellbare Nachrichten eine Option aus, um festzulegen, wie unverarbeitete Ereignisse behandelt werden. Geben Sie bei Bedarf die

Amazon SQS-Warteschlange an, die als Warteschlange für unzustellbare Nachrichten verwendet werden soll.

17. (Optional) Wählen Sie Add another target (Weiteres Ziel hinzufügen) aus, um ein weiteres Ziel für diese Regel hinzuzufügen.
18. Wählen Sie Weiter aus.
19. (Optional) Wählen Sie für Tags die Option Neues Tag hinzufügen aus, um eine Ressourcenbezeichnung für die Regel hinzuzufügen. Weitere Informationen finden Sie unter Amazon- EventBridge Tags.
20. Wählen Sie Weiter aus.
21. Überprüfen Sie unter Überprüfen und erstellen die Konfigurationsschritte. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Regel erstellen.

Weitere Informationen zum Erstellen von Regeln finden Sie unter Erstellen einer Amazon-EventBridge Regel, die nach einem Zeitplan ausgeführt wird im Amazon- EventBridge Benutzerhandbuch.

Erstellen einer Regel mit einem Ereignismuster

Im folgenden Verfahren wird beschrieben, wie Sie eine Regel mit einem Ereignismuster erstellen.

So erstellen Sie eine Regel, die das Ereignis an ein Ziel sendet, wenn das Ereignis einem definierten Muster entspricht

> **Note**
>
> Dieses Verfahren funktioniert für alle AWS Batch auf Amazon-ECS-, Amazon-EKS- und AWS Fargate-Aufträgen.

1. Öffnen Sie die Amazon- EventBridge Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie in der Navigationsleiste die zu AWS-Region verwendende aus.
3. Wählen Sie im Navigationsbereich Rules aus.
4. Wählen Sie Regel erstellen aus.

5. Geben Sie unter Name einen eindeutigen Namen für Ihre Datenverarbeitungsumgebung an. Der Name kann bis zu 64 Zeichen umfassen. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.

   > **Note**
   >
   > Eine Regel darf nicht denselben Namen wie eine andere Regel in derselben Region und auf demselben Event Bus haben.

6. (Optional) Geben Sie unter Beschreibung eine Beschreibung für die Regel ein.
7. Wählen Sie für Event Bus den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie Standard aus. Wenn ein AWS-Service in Ihrem Konto ein Ereignis ausgibt, wird es immer an den Standard-Event-Bus Ihres Kontos weitergeleitet.
8. (Optional) Deaktivieren Sie die Regel für den ausgewählten Bus, wenn Sie die Regel nicht sofort ausführen möchten.
9. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
10. Wählen Sie Weiter aus.
11. Wählen Sie für Ereignisquelle die Option AWS Ereignis- oder EventBridge Partnerereignisse aus.
12. (Optional) Für Beispielereignis :
    a. Wählen Sie für Beispielereignistyp die Option AWS Ereignisse aus.
    b. Wählen Sie für Beispielereignisse die Option Statusänderung des Batch-Auftrags aus.
13. Wählen Sie für Erstellungsmethode die Option Musterformular verwenden aus.
14. Für Ereignismuster :
    a. Wählen Sie für Ereignisquelle die Option AWS-Services aus.
    b. AWS-ServiceWählen Sie für Batch aus.
    c. Wählen Sie für Ereignistyp die Option Statusänderung des Batch-Auftrags aus.
15. Wählen Sie Weiter aus.
16. Für Target types (Zieltypen), wählen Sie AWS-Service aus.
17. Wählen Sie unter Ziel auswählen einen Zieltyp aus. Wählen Sie beispielsweise Batch-Auftragswarteschlange aus. Geben Sie dann Folgendes an:

- Job queue (Auftragswarteschlange): Geben Sie den Amazon-Ressourcennamen (ARN) der Auftragswarteschlange ein, in der Ihr Auftrag eingeplant werden soll.
- Auftragsdefinition: Geben Sie den Namen und die Revision oder den vollständigen ARN der Auftragsdefinition ein, die für Ihren Auftrag verwendet werden sollen.
- Job name (Auftragsname): Geben Sie einen Namen für Ihren Auftrag ein.
- Array-Größe: (Optional) Geben Sie eine Array-Größe für Ihren Auftrag ein, um mehrere Kopien davon auszuführen. Weitere Informationen finden Sie unter Array-Jobs.
- Auftragsversuche: (Optional) Geben Sie an, wie oft die Ausführung Ihres Auftrags versucht werden soll, bevor festgelegt wird, dass er fehlgeschlagen ist. Weitere Informationen finden Sie unter Automatisierte Auftragswiederholungen.

18. Bei Batch-Auftragswarteschlangen-Zieltypen EventBridge benötigt die Berechtigung zum Senden von Ereignissen an das Ziel. EventBridge kann die IAM-Rolle erstellen, die für die Ausführung Ihrer Regel erforderlich ist. Führen Sie eine der folgenden Aktionen aus:

    - Um automatisch eine IAM-Rolle zu erstellen, wählen Sie Create a new role for this specific resource (Eine neue Rolle für diese spezifische Ressource erstellen).
    - Wenn Sie eine zuvor erstellte IAM-Rolle verwenden möchten, wählen Sie Use existing role (Vorhandene Rolle verwenden).

19. (Optional) Erweitern Sie Additional settings (Zusätzliche Einstellungen).

    a. Wählen Sie unter Zieleingabe konfigurieren aus, wie Text aus einem Ereignis verarbeitet wird.

    b. Geben Sie unter Höchstalter des Ereignisses das Zeitintervall für die Aufbewahrungsdauer unverarbeiteter Ereignisse an.

    c. Geben Sie für Wiederholungsversuche die Häufigkeit ein, mit der ein Ereignis erneut versucht wird.

    d. Wählen Sie für Warteschlange für unzustellbare Nachrichten eine Option aus, um festzulegen, wie unverarbeitete Ereignisse behandelt werden. Geben Sie bei Bedarf die Amazon SQS-Warteschlange an, die als Warteschlange für unzustellbare Nachrichten verwendet werden soll.

20. (Optional) Wählen Sie Weiteres Ziel hinzufügen, um ein zusätzliches Ziel hinzuzufügen.
21. Wählen Sie Weiter aus.

22. (Optional) Wählen Sie für Tags die Option Neues Tag hinzufügen aus, um eine Ressourcenbezeichnung hinzuzufügen. Weitere Informationen finden Sie unter Amazon-EventBridge Tags im Amazon- EventBridge Benutzerhandbuch.
23. Wählen Sie Weiter aus.
24. Überprüfen Sie unter Überprüfen und erstellen die Konfigurationsschritte. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie fertig sind, wählen Sie Regel erstellen aus.

    Weitere Informationen zum Erstellen von Regeln finden Sie unter Erstellen von Amazon-EventBridge Regeln, die auf Ereignisse reagieren im Amazon- EventBridge Benutzerhandbuch.

Übergeben von Ereignisinformationen an ein AWS Batch Ziel nach einem Zeitplan mithilfe des EventBridge Eingabe-Transformators

Sie können den EventBridge Eingabe-Transformator verwenden, um Ereignisinformationen an AWS Batch in einer Auftragsübermittlung zu übergeben. Dies kann besonders nützlich sein, wenn Sie Aufträge aufgrund anderer AWS Ereignisinformationen aufrufen. Ein Beispiel ist ein Objekt-Upload in einen Amazon S3-Bucket. Sie können auch eine Auftragsdefinition mit Parameterersetzungswerten im -Befehl des Containers verwenden. Der EventBridge Eingabe-Transformator kann die Parameterwerte basierend auf den Ereignisdaten bereitstellen.

Anschließend erstellen Sie ein AWS Batch Ereignisziel, das Informationen aus dem Ereignis analysiert, das es startet, und es in ein `parameters` Objekt umwandelt. Wenn der Auftrag ausgeführt wird, werden die Parameter aus dem Auslöserereignis an den Befehl des Auftragscontainers übergeben.

> **Note**
>
> In diesem Szenario müssen sich alle AWS Ressourcen (wie Amazon S3-Buckets, EventBridge -Regeln und CloudTrail -Protokolle) in derselben Region befinden.

So erstellen Sie ein AWS Batch Ziel, das den Eingabe-Transformator verwendet

1. Öffnen Sie die Amazon- EventBridge Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie in der Navigationsleiste die zu AWS-Region verwendende aus.
3. Wählen Sie im Navigationsbereich Rules aus.

4. Wählen Sie Regel erstellen aus.
5. Geben Sie unter Name einen eindeutigen Namen für Ihre Datenverarbeitungsumgebung an. Der Name kann bis zu 64 Zeichen umfassen. Er kann Groß- und Kleinbuchstaben, Zahlen, Bindestriche (-) und Unterstriche (_) enthalten.

   > **Note**
   >
   > Eine Regel darf nicht denselben Namen wie eine andere Regel im selben AWS-Region und im selben Event Bus haben.

6. (Optional) Geben Sie unter Beschreibung eine Beschreibung für die Regel ein.
7. Wählen Sie für Event Bus den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie Standard aus. Wenn ein AWS-Service in Ihrem Konto ein Ereignis ausgibt, wird es immer an den Standard-Event-Bus Ihres Kontos weitergeleitet.
8. (Optional) Deaktivieren Sie die Regel für den ausgewählten Bus, wenn Sie die Regel nicht sofort ausführen möchten.
9. Wählen Sie unter Rule type (Regeltyp) die Option Schedule (Zeitplan) aus.
10. Wählen Sie Weiter, um die Regel zu erstellen, oder Weiter aus.
11. Gehen Sie bei Schedule pattern (Zeitplanmuster) wie folgt vor:

    - Wählen Sie Ein detaillierter Zeitplan, der zu einer bestimmten Zeit ausgeführt wird, z. B. um 8:00 Uhr aus. PST jeden ersten Montag im Monat und geben Sie dann einen Cron-Ausdruck ein. Weitere Informationen finden Sie unter Cron-Ausdrücke im Amazon- EventBridge Benutzerhandbuch.
    - Wählen Sie Ein Zeitplan, der regelmäßig ausgeführt wird, z. B. alle 10 Minuten., und geben Sie dann einen Ratenausdruck ein.

12. Wählen Sie Weiter aus.
13. Für Target types (Zieltypen), wählen Sie AWS-Service aus.
14. Wählen Sie für Ziel auswählen die Option Batch-Auftragswarteschlange aus. Konfigurieren Sie dann Folgendes:

    - Job queue (Auftragswarteschlange): Geben Sie den Amazon-Ressourcennamen (ARN) der Auftragswarteschlange ein, in der Ihr Auftrag eingeplant werden soll.

- Auftragsdefinition: Geben Sie den Namen und die Revision oder den vollständigen ARN der Auftragsdefinition ein, die für Ihren Auftrag verwendet werden sollen.
- Job name (Auftragsname): Geben Sie einen Namen für Ihren Auftrag ein.
- Array-Größe: (Optional) Geben Sie eine Array-Größe für Ihren Auftrag ein, um mehrere Kopien davon auszuführen. Weitere Informationen finden Sie unter Array-Jobs.
- Auftragsversuche: (Optional) Geben Sie an, wie oft die Ausführung Ihres Auftrags versucht werden soll, bevor festgelegt wird, dass er fehlgeschlagen ist. Weitere Informationen finden Sie unter Automatisierte Auftragswiederholungen.

15. Bei Batch-Auftragswarteschlangen-Zieltypen EventBridge benötigt die Berechtigung zum Senden von Ereignissen an das Ziel. EventBridge kann die IAM-Rolle erstellen, die für die Ausführung Ihrer Regel erforderlich ist. Führen Sie eine der folgenden Aktionen aus:

    - Um automatisch eine IAM-Rolle zu erstellen, wählen Sie Create a new role for this specific resource (Eine neue Rolle für diese spezifische Ressource erstellen).
    - Um eine bereits erstellte IAM-Rolle zu verwenden, wählen Sie Vorhandene Rolle verwenden aus.

16. (Optional) Erweitern Sie Additional settings (Zusätzliche Einstellungen).
17. Wählen Sie im Abschnitt Additional settings (Zusätzliche Einstellungen) unter Configure target input (Zieleingabe konfigurieren) die Option Input Transformer (Eingabetransformator).
18. Wählen Sie Configure input transformer (Eingabetransformator konfigurieren).
19. (Optional) Für Beispielereignis :

    a. Wählen Sie für Beispielereignistyp die Option AWS Ereignisse aus.

    b. Wählen Sie für Beispielereignisse die Option Statusänderung des Batch-Auftrags aus.

20. Geben Sie im Abschnitt Target input transformer (Zieleingabetransformator) für Input path (Eingabepfad) die Werte an, die aus dem auslösenden Ereignis analysiert werden sollen. Verwenden Sie beispielsweise das folgende JSON-Format, um das Ereignis „Batch-Auftragsstatusänderung“ zu analysieren.

```
{
    "instance": "$.detail.jobId",
    "state": "$.detail.status"
}
```

21. Geben Sie für Vorlage Folgendes ein.

```
{
    "instance": <jobId> ,
    "status": <status>
}
```

22. Wählen Sie Bestätigen aus.
23. Geben Sie unter Höchstalter des Ereignisses das Zeitintervall für die Aufbewahrungsdauer unverarbeiteter Ereignisse an.
24. Geben Sie unter Wiederholungsversuche die Häufigkeit ein, mit der ein Ereignis erneut versucht wird.
25. Wählen Sie für Warteschlange für unzustellbare Nachrichten eine Option aus, um festzulegen, wie unverarbeitete Ereignisse behandelt werden. Geben Sie bei Bedarf die Amazon SQS-Warteschlange an, die als Warteschlange für unzustellbare Nachrichten verwendet werden soll.
26. (Optional) Wählen Sie Weiteres Ziel hinzufügen, um ein zusätzliches Ziel hinzuzufügen.
27. Wählen Sie Weiter aus.
28. (Optional) Wählen Sie für Tags die Option Neues Tag hinzufügen aus, um eine Ressourcenbezeichnung hinzuzufügen. Weitere Informationen finden Sie unter Amazon-EventBridge Tags im Amazon- EventBridge Benutzerhandbuch.
29. Wählen Sie Weiter aus.
30. Überprüfen Sie unter Überprüfen und erstellen die Konfigurationsschritte. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Nachdem Sie fertig sind, wählen Sie Regel erstellen aus.

Tutorial: Auf warten AWS Batch EventBridge

In diesem Tutorial richten Sie eine einfache AWS Lambda Funktion ein, die AWS Batch Auftragsereignisse überwacht und sie in einen CloudWatch Logs-Protokollstream schreibt.

Voraussetzungen

Dieses Tutorial unterstellt eine funktionierende Datenverarbeitungsumgebung samt Auftragswarteschlange, die bereit ist, Aufträge aufzunehmen. Wenn Sie nicht über eine laufende Datenverarbeitungsumgebung und Auftragswarteschlange verfügen, aus der Ereignisse erfasst werden sollen, führen Sie die Schritte unter aus, Erste Schritte mit AWS Batch um eine zu erstellen.

Am Ende dieses Tutorials können Sie optional einen Auftrag an diese Auftragswarteschlange senden, um zu testen, ob Sie Ihre Lambda-Funktion korrekt konfiguriert haben.

Schritt 1: Erstellen der Lambda-Funktion

In diesem Verfahren erstellen Sie eine einfache Lambda-Funktion, die als Ziel für AWS Batch Ereignis-Stream-Nachrichten dient.

So erstellen Sie eine Lambda-Zielfunktion

1. Öffnen Sie die AWS Lambda-Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie Funktion erstellen und Von Grund auf neu erstellen aus.
3. Geben Sie für Function name (Funktionsname) batch-event-stream-handler ein.
4. Wählen Sie für Runtime (Laufzeit) die Option Python 3.8 aus.
5. Wählen Sie Funktion erstellen.
6. Bearbeiten Sie im Abschnitt Codequelle den Beispielcode so, dass er dem folgenden Beispiel entspricht:

   ```
   import json


   def lambda_handler(event, _context):
       # _context is not used
       del _context
       if event["source"] != "aws.batch":
           raise ValueError("Function only supports input from events with a source
    type of: aws.batch")

       print(json.dumps(event))
   ```

   Dies ist eine einfache Python-3.8-Funktion, die die von gesendeten Ereignisse drucktAWS Batch. Wenn alles korrekt konfiguriert ist, werden am Ende dieses Tutorials die Ereignisdetails im CloudWatch Protokollstream angezeigt, der dieser Lambda-Funktion zugeordnet ist.
7. Wählen Sie Bereitstellen.

Schritt 2: Registrieren von Ereignisregeln

In diesem Abschnitt erstellen Sie eine EventBridge Ereignisregel, die Auftragsereignisse erfasst, die von Ihren -AWS BatchRessourcen stammen. Diese Regel erfasst alle Ereignisse aus dem Konto, AWS Batch in dem sie definiert ist. Die Auftragsnachrichten selbst enthalten Informationen über die Ereignisquelle, einschließlich der Auftragswarteschlange, an die sie übermittelt wurde. Sie können diese Informationen verwenden, um Ereignisse programmgesteuert zu filtern und zu sortieren.

> **Note**
>
> Wenn Sie die verwenden, AWS Management Console um eine Ereignisregel zu erstellen, fügt die Konsole automatisch die IAM-Berechtigungen für hinzu EventBridge , um Ihre Lambda-Funktion aufzurufen. Wenn Sie jedoch eine Ereignisregel mit der erstellenAWS CLI, müssen Sie Berechtigungen explizit erteilen. Weitere Informationen finden Sie unter Ereignisse und Ereignismuster im Amazon- EventBridge Benutzerhandbuch.

So erstellen Sie Ihre EventBridge Regel

1. Öffnen Sie die Amazon- EventBridge Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen aus.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein.

   Eine Regel darf nicht denselben Namen wie eine andere Regel in derselben Region und auf demselben Event Bus haben.
5. Wählen Sie als Event bus (Event Bus) den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie AWS-Standard-Event-Bus aus. Wenn ein AWS-Service in Ihrem Konto ein Ereignis ausgibt, wird dieses stets an den standardmäßigen Event Bus Ihres Kontos weitergeleitet.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Weiter aus.
8. Wählen Sie für Event source (Ereignisquelle) Other (Andere) aus.
9. Wählen Sie für Ereignismuster die Option Benutzerdefinierte Muster (JSON-Editor) aus.

10. Fügen Sie das folgende Ereignismuster in das Textfeld ein.

```
{
  "source": [
    "aws.batch"
  ]
}
```

    Diese Regel gilt für alle Ihre AWS Batch Gruppen und für jedes AWS Batch Ereignis. Alternativ können Sie eine spezifischere Regel erstellen, damit bestimmte Ergebnisse gefiltert werden.

11. Wählen Sie Weiter aus.
12. Bei Target types (Zieltypen) wählen Sie AWS-Service aus.
13. Wählen Sie unter Ziel auswählen die Option Lambda-Funktion und dann Ihre Lambda-Funktion aus.
14. (Optional) Gehen Sie unter Additional settings (Weitere Einstellungen) wie folgt vor:

    a. Geben Sie für Maximum age of event (Maximales Alter des Ereignisses) einen Wert zwischen einer Minute (00:01) und 24 Stunden (24:00) ein.

    b. Geben Sie für Wiederholungsversuche eine Zahl zwischen 0 und 185 ein.

    c. Wählen Sie für Warteschlange für unzustellbare Nachrichten aus, ob eine standardmäßige Amazon SQS-Warteschlange als Warteschlange für unzustellbare Nachrichten verwendet werden soll. EventBridge sendet Ereignisse, die dieser Regel entsprechen, an die Warteschlange für unzustellbare Nachrichten, wenn sie nicht erfolgreich an das Ziel übermittelt wurden. Führen Sie eine der folgenden Aktionen aus:

        - Klicken Sie auf Keine, um keine Warteschlange für unzustellbare Nachrichten zu verwenden.
        - Klicken Sie auf Select an Amazon SQS queue in the current AWS account to use as the dead-letter queue (Wählen Sie eine Amazon SQS Warteschlange im aktuellen -Konto als Warteschlange für unzustellbare Nachrichten) und wählen Sie nun die Warteschlange aus der Dropdown-Liste aus.
        - Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange in einem anderen AWS-Konto als Warteschlange für unzustellbare Nachrichten und geben Sie dann den ARN der Warteschlange ein, die verwendet werden soll. Sie müssen eine ressourcenbasierte Richtlinie an die Warteschlange anfügen, die die EventBridge Berechtigung zum Senden von Nachrichten an diese gewährt. Weitere Informationen finden Sie unter

Erteilen von Berechtigungen für unzustellbare Nachrichten im Amazon- EventBridge Benutzerhandbuch.

15. Wählen Sie Weiter aus.
16. (Optional) Geben Sie ein oder mehrere Tags für die Regel ein. Weitere Informationen finden Sie unter Amazon- EventBridge Tags im Amazon- EventBridge Benutzerhandbuch.
17. Wählen Sie Weiter aus.
18. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 3: Testen der Konfiguration

Sie können Ihre EventBridge Konfiguration jetzt testen, indem Sie einen Auftrag an Ihre Auftragswarteschlange senden. Wenn alles richtig konfiguriert ist, wird Ihre Lambda-Funktion ausgelöst und schreibt die Ereignisdaten in einen CloudWatch Logs-Protokollstream für die Funktion.

So testen Sie die Konfiguration

1. Öffnen Sie die -AWS BatchKonsole unter https://console.aws.amazon.com/batch/.
2. Senden Sie einen neuen AWS Batch-Auftrag. Weitere Informationen finden Sie unter Einen Job einreichen.
3. Öffnen Sie die - CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
4. Wählen Sie im Navigationsbereich Logs und die Protokollgruppe für Ihre Lambda-Funktion aus (z. B. /aws/lambda/*my-function*).
5. Wählen Sie einen Protokollstream aus, um die Ereignisdaten anzuzeigen.

Tutorial: Senden von Amazon Simple Notification Service-Warnungen für fehlgeschlagene Auftragsereignisse

In diesem Tutorial konfigurieren Sie eine EventBridge Ereignisregel, die nur Auftragsereignisse erfasst, bei denen der Auftrag in einen `-FAILED`Status verschoben wurde. Am Ende dieses Tutorials können Sie optional auch einen Auftrag an diese Auftragswarteschlange senden. Dadurch wird getestet, ob Sie Ihre Amazon SNS-Warnungen korrekt konfiguriert haben.

Voraussetzungen

Dieses Tutorial unterstellt eine funktionierende Datenverarbeitungsumgebung samt Auftragswarteschlange, die bereit ist, Aufträge aufzunehmen. Wenn Sie nicht über eine laufende Datenverarbeitungsumgebung und Auftragswarteschlange verfügen, aus der Ereignisse erfasst werden sollen, führen Sie die Schritte unter aus, Erste Schritte mit AWS Batch um eine zu erstellen.

Schritt 1: Erstellen und Abonnieren eines Amazon SNS-Themas

Mit diesem Tutorial konfigurieren Sie ein Amazon SNS-Thema, das als Ereignisziel für Ihre neue Ereignisregel dient.

Erstellen eines Amazon SNS-Themas

1. Öffnen Sie die Amazon SNS-Konsole unter https://console.aws.amazon.com/sns/v3/home.
2. Wählen Sie Themen, Thema erstellen aus.
3. Wählen Sie unter Type (Typ) die Option Standard aus.
4. Geben Sie für Name **`JobFailedAlert`** einund wählen Sie Thema erstellen aus.
5. Wählen Sie auf dem JobFailedAlert Bildschirm Abonnement erstellen aus.
6. Wählen Sie unter Protocol (Protokoll) die Option Email (E-Mail) aus.
7. Geben Sie für Endpunkt eine E-Mail-Adresse ein, auf die Sie aktuell Zugriff haben, und wählen Sie Abonnement erstellen aus.
8. Überprüfen Sie Ihr E-Mail-Konto und warten Sie auf eine E-Mail-Nachricht zur Bestätigung Ihres Abonnements. Wenn Sie sie erhalten, wählen Sie Confirm Abonnement aus.

Schritt 2: Registrieren von Ereignisregeln

Registrieren Sie nun eine Ereignisregel, mit der nur Ereignisse zu fehlgeschlagenen Aufträgen erfasst werden.

So registrieren Sie Ihre EventBridge Regel

1. Öffnen Sie die Amazon- EventBridge Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen aus.

4. Geben Sie einen Namen und eine Beschreibung für die Regel ein.

   Eine Regel darf nicht denselben Namen wie eine andere Regel in derselben Region und auf demselben Event Bus haben.

5. Wählen Sie als Event bus (Event Bus) den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie AWS -Standard-Event-Bus aus. Wenn ein - AWS Service in Ihrem Konto ein Ereignis ausgibt, wird es immer an den Standard-Event-Bus Ihres Kontos weitergeleitet.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Weiter aus.
8. Wählen Sie für Event source (Ereignisquelle) Other (Andere) aus.
9. Wählen Sie für Ereignismuster die Option Benutzerdefinierte Muster (JSON-Editor) aus.
10. Fügen Sie das folgende Ereignismuster in das Textfeld ein.

```
{
  "detail-type": [
    "Batch Job State Change"
  ],
  "source": [
    "aws.batch"
  ],
  "detail": {
    "status": [
      "FAILED"
    ]
  }
}
```

    Dieser Code definiert eine EventBridge Regel, die jedem Ereignis entspricht, bei dem der Auftragsstatus lautet`FAILED`. Weitere Informationen zu Ereignismustern finden Sie unter Ereignisse und Ereignismuster im Amazon- EventBridge Benutzerhandbuch.

11. Wählen Sie Weiter aus.
12. Bei Target types (Zieltypen) wählen Sie AWS -Service aus.
13. Wählen Sie für Ziel auswählen ein SNS-Thema und für Thema ausJobFailedAlert.
14. (Optional) Gehen Sie unter Additional settings (Weitere Einstellungen) wie folgt vor:

a. Geben Sie für Maximum age of event (Maximales Alter des Ereignisses) einen Wert zwischen einer Minute (00:01) und 24 Stunden (24:00) ein.

b. Geben Sie für Wiederholungsversuche eine Zahl zwischen 0 und 185 ein.

c. Wählen Sie für Warteschlange für unzustellbare Nachrichten aus, ob eine standardmäßige Amazon SQS-Warteschlange als Warteschlange für unzustellbare Nachrichten verwendet werden soll. EventBridge sendet Ereignisse, die dieser Regel entsprechen, an die Warteschlange für unzustellbare Nachrichten, wenn sie nicht erfolgreich an das Ziel übermittelt wurden. Führen Sie eine der folgenden Aktionen aus:

   - Klicken Sie auf Keine, um keine Warteschlange für unzustellbare Nachrichten zu verwenden.
   - Wählen Sie Amazon SQS-Warteschlange im aktuellen AWS Konto auswählen, um sie als Warteschlange für unzustellbare Nachrichten zu verwenden, und wählen Sie dann die zu verwendende Warteschlange aus der Dropdownliste aus.
   - Wählen Sie Amazon SQS-Warteschlange in einem anderen AWS Konto als Warteschlange für unzustellbare Nachrichten auswählen und geben Sie dann den ARN der zu verwendenden Warteschlange ein. Sie müssen eine ressourcenbasierte Richtlinie an die Warteschlange anfügen, die die EventBridge Berechtigung zum Senden von Nachrichten an diese gewährt. Weitere Informationen finden Sie unter Erteilen von Berechtigungen für unzustellbare Nachrichten im Amazon- EventBridge Benutzerhandbuch.

15. Wählen Sie Weiter aus.
16. (Optional) Geben Sie ein oder mehrere Tags für die Regel ein. Weitere Informationen finden Sie unter Amazon- EventBridge Tags im Amazon- EventBridge Benutzerhandbuch.
17. Wählen Sie Weiter aus.
18. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 3: Testen Ihrer Regel

Übermitteln Sie zum Testen der Regel einen Auftrag, der kurz nach dem Start mit einem Beendigungscode ungleich Null beendet wird. Wenn Ihre Ereignisregel korrekt konfiguriert ist, sollten Sie innerhalb weniger Minuten eine E-Mail-Nachricht mit dem Ereignistext erhalten.

So testen Sie eine Regel

1. Öffnen Sie die - AWS Batch Konsole unter https://console.aws.amazon.com/batch/.
2. Senden Sie einen neuen AWS Batch Auftrag. Weitere Informationen finden Sie unter Einen Job einreichen. Ersetzen Sie den Befehl des Auftrags durch diesen Befehl, damit der Container mit einem Beendigungscode von 1 beendet wird.

   ```
   /bin/sh, -c, 'exit 1'
   ```

3. Überprüfen Sie Ihre E-Mail, um zu bestätigen, dass Sie eine E-Mail-Benachrichtigung über die Benachrichtigung über fehlgeschlagene Aufträge erhalten haben.

Alternative Regel: Batch-Auftragswarteschlange blockiert

Wiederholen Sie die Schritte in diesem Tutorial mit den folgenden Änderungen, um eine Ereignisregel zu erstellen, die auf Blockierte Batch-Auftragswarteschlange überwacht:

1. Verwenden Sie in Schritt 1 *BlockedJobQueue* als Themennamen.
2. Verwenden Sie in Schritt 2 das folgende Muster im JSON-Editor:

```
{
   "detail-type": [
     "Batch Job Queue Blocked"
   ],
   "source": [
     "aws.batch"
   ]
}
```

CloudWatch Logs verwenden mit AWS Batch

Sie können Ihre AWS Batch Jobs auf EC2-Ressourcen so konfigurieren, dass detaillierte Protokollinformationen und Metriken an CloudWatch Logs gesendet werden. Auf diese Weise können Sie verschiedene Protokolle Ihrer Jobs an einem zentralen Ort einsehen. Weitere Informationen zu CloudWatch Logs finden Sie unter Was ist Amazon CloudWatch Logs? im CloudWatch Amazon-Benutzerhandbuch.

> **Note**
>
> Standardmäßig ist CloudWatch Logs für AWS Fargate-Container aktiviert.

Um die CloudWatch Log-Protokollierung zu aktivieren und anzupassen, überprüfen Sie die folgenden einmaligen Konfigurationsaufgaben:

- Für AWS Batch Rechenumgebungen, die auf EC2-Ressourcen basieren, fügen Sie der Rolle eine IAM-Richtlinie hinzu. `ecsInstanceRole` Weitere Informationen finden Sie unter the section called “Fügen Sie eine CloudWatch Logs-IAM-Richtlinie hinzu”.
- Erstellen Sie eine Amazon EC2 EC2-Startvorlage, die eine detaillierte CloudWatch Überwachung beinhaltet, und geben Sie die Vorlage dann an, wenn Sie Ihre AWS Batch Datenverarbeitungsumgebung erstellen. Sie können den CloudWatch Agenten auch auf einem vorhandenen Image installieren und das Image dann im Assistenten für die AWS Batch erste Ausführung angeben.
- (Optional) Konfigurieren Sie den awslogs-Treiber. Sie können Parameter hinzufügen, die das Standardverhalten sowohl auf EC2- als auch auf Fargate-Ressourcen ändern. Weitere Informationen finden Sie unter the section called “Verwenden des awslogs-Protokolltreibers”.

Fügen Sie eine CloudWatch Logs-IAM-Richtlinie hinzu

Bevor Ihre Jobs Protokolldaten und detaillierte Metriken an CloudWatch Logs senden können, müssen Sie eine IAM-Richtlinie erstellen, die die CloudWatch Logs-APIs verwendet. Nachdem Sie die IAM-Richtlinie erstellt haben, fügen Sie sie der `ecsInstanceRole` Rolle hinzu.

> **Note**
>
> Wenn die `ECS-CloudWatchLogs` Richtlinie nicht an die `ecsInstanceRole` Rolle angehängt ist, können grundlegende Metriken trotzdem an CloudWatch Logs gesendet werden. Die Basiskennzahlen enthalten jedoch keine Protokolldaten oder detaillierte Messwerte wie den freien Festplattenspeicher.

AWS Batch Rechenumgebungen verwenden Amazon EC2 EC2-Ressourcen. Wenn Sie mit dem Assistenten für die AWS Batch erste Ausführung eine Datenverarbeitungsumgebung erstellen, AWS Batch erstellt er die `ecsInstanceRole` Rolle und konfiguriert die Umgebung damit.

Wenn Sie den Assistenten für die erste Ausführung nicht verwenden, können Sie die `ecsInstanceRole` Rolle angeben, wenn Sie eine Rechenumgebung in der AWS Command Line Interface OR-API erstellen. AWS Batch Weitere Informationen finden Sie in der AWS CLI Befehlsreferenz oder AWS Batch API-Referenz.

So erstellen Sie die **`ECS-CloudWatchLogs`**-IAM-Richtlinie

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Policies aus.
3. Wählen Sie Richtlinie erstellen aus.
4. Wählen Sie JSON und geben Sie dann die folgende Richtlinie ein:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": [
                "arn:aws:logs:*:*:*"
            ]
        }
    ]
```

```
}
```

5. Wählen Sie Weiter: Markierungen.
6. (Optional) Wählen Sie unter Tags hinzufügen die Option Tag hinzufügen aus, um der Richtlinie ein Tag hinzuzufügen.
7. Wählen Sie Weiter: Prüfen aus.
8. Geben Sie auf der Seite „Richtlinie überprüfen“ als Namen eine Beschreibung ein**`ECS-CloudWatchLogs`**, und geben Sie dann optional eine Beschreibung ein.
9. Wählen Sie Richtlinie erstellen aus.

So fügen Sie die Richtlinie **`ECS-CloudWatchLogs`** an die **`ecsInstanceRole`** an

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Rollen aus.
3. Wählen Sie `ecsInstanceRole`. Wenn die Rolle nicht existiert, folgen Sie den Verfahren unter Amazon-ECS-Instance-Rolle So erstellen Sie die Rolle.
4. Wählen Sie „Berechtigungen hinzufügen“ und anschließend „Richtlinien anhängen“.
5. Wählen Sie die CloudWatchECS-Logs-Richtlinie und anschließend die Option Richtlinie anhängen aus.

Installieren und konfigurieren Sie den CloudWatch Agenten

Sie können eine Amazon EC2 EC2-Startvorlage erstellen, die die CloudWatch Überwachung beinhaltet. Weitere Informationen finden Sie unter Starten einer Instance über eine Startvorlage und Erweiterte Details im Amazon EC2 EC2-Benutzerhandbuch.

Sie können den CloudWatch Agenten auch auf einem vorhandenen Amazon EC2 EC2-AMI installieren und dann das Image im Assistenten für die AWS Batch erste Ausführung angeben. Weitere Informationen finden Sie unter Installation des CloudWatch Agenten und Erste Schritte mit. AWS Batch

> **Note**
>
> Startvorlagen werden auf AWS Fargate Ressourcen nicht unterstützt.

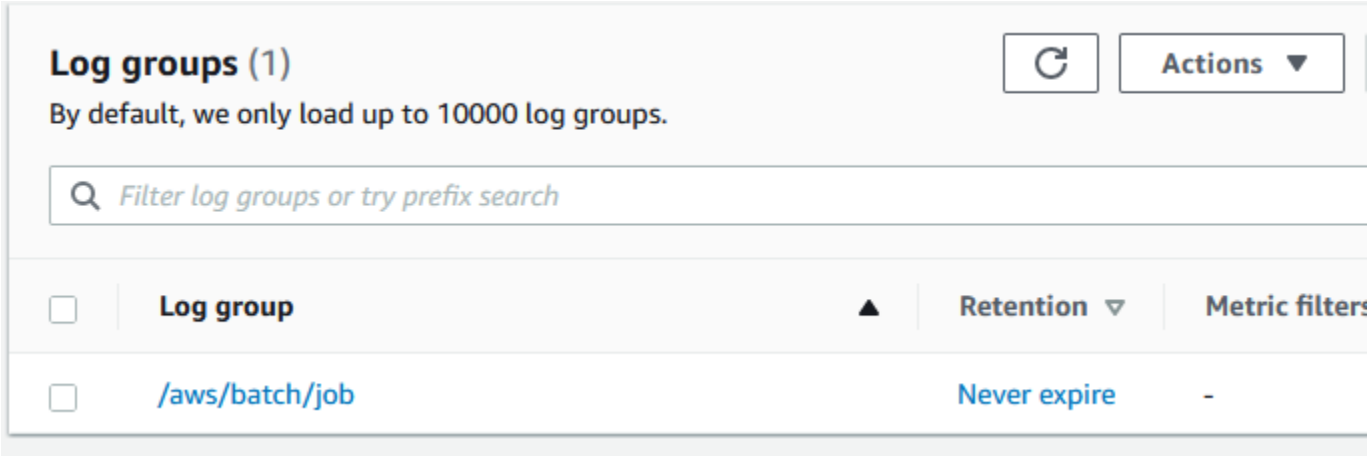
CloudWatch Protokolle anzeigen

Sie können CloudWatch Log-Logs in der anzeigen und durchsuchen AWS Management Console.

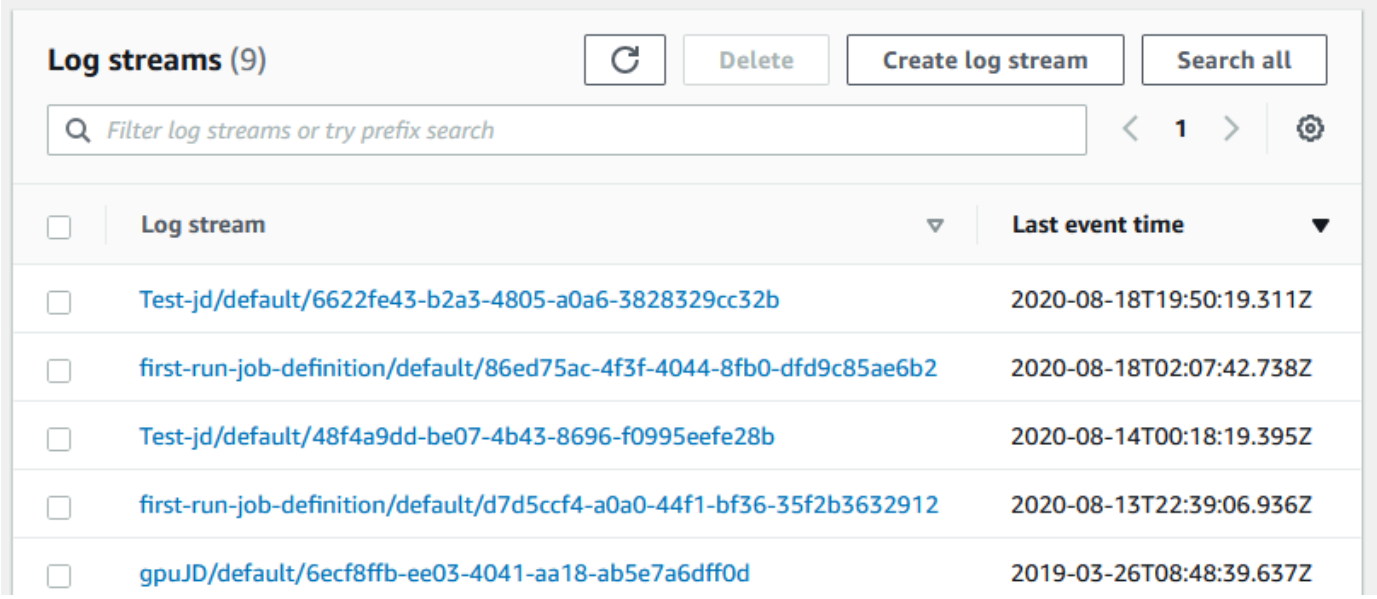
> **Note**
>
> Es kann einige Minuten dauern, bis Daten in CloudWatch Logs angezeigt werden.

Um Ihre CloudWatch Logs-Daten einzusehen

1. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich Logs und anschließend Log groups aus.

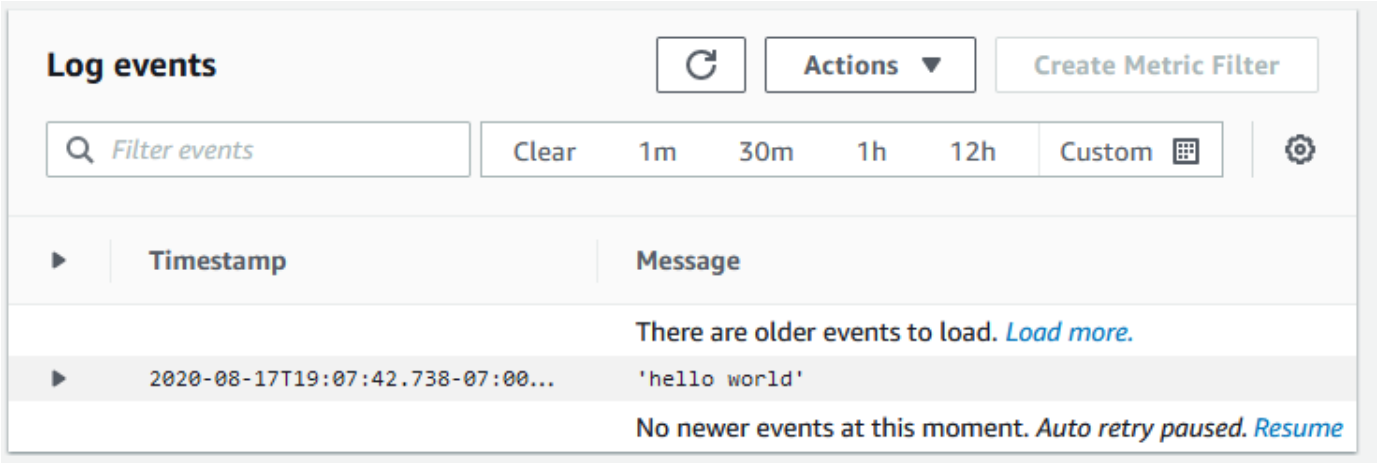


3. Wählen Sie eine Protokollgruppe aus, die angezeigt werden soll.



4. Wählen Sie einen Protokollstream aus, der angezeigt werden soll. Standardmäßig werden die Streams durch die ersten 200 Zeichen des Jobnamens und der Amazon ECS-Aufgaben-ID identifiziert.

> **Tip**
>
> Um Log-Stream-Daten herunterzuladen, wählen Sie Aktionen.



Verwenden von - CloudWatch Protokollen zur Überwachung AWS Batch von Amazon-EKS-Aufträgen

Sie können Amazon CloudWatch Logs verwenden, um alle Ihre Protokolldateien an einem Ort zu überwachen, zu speichern und anzuzeigen. Mit - CloudWatch Protokollen können Sie Protokolldaten aus mehreren Quellen suchen, filtern und analysieren.

Sie können ein AWS -Fluent Bitfor-Image herunterladen, das ein Plugin zur Überwachung AWS Batch von Amazon-EKS-Aufträgen in CloudWatch Logs enthält. Fluent Bit ist ein Open-Source-Protokollprozessor und eine -Weiterleitung, die sowohl Docker als auch Kubernetes kompatibel sind. Wir empfehlen, dass Sie Fluent Bit als Protokollrouter verwenden, da dieser weniger ressourcenintensiv ist als Fluentd. Weitere Informationen finden Sie unter Verwenden von AWS für Fluent-Bit-Image.

Voraussetzungen

Fügen Sie die `CloudWatchAgentServerPolicy` Richtlinie an die AWS Identity and Access Management Richtlinie Ihrer Worker-Knoten an. Weitere Informationen finden Sie unter Überprüfen der Voraussetzungen.

Installieren von AWS für Fluent Bit

Anweisungen zur Installation von AWS für Fluent Bit und zum Erstellen der CloudWatch Gruppen finden Sie unter Einrichten Fluent Bit oder Schnellstart mit dem CloudWatch Agenten und Fluent Bit.

> **Tip**
>
> Denken Sie daran, dass 0,5 CPU und 100 MB Arbeitsspeicher auf AWS Batch Knoten Fluent Bit verwendet. Dies reduziert die verfügbare Gesamtkapazität für AWS Batch Aufträge. Beachten Sie dies, wenn Sie Ihre Aufträge dimensionieren.

Fluent Bit für AWS Batch Knoten aktivieren

Um sicherzustellen, dass die Fluent Bit Protokollierung auf AWS Batch verwalteten Knoten DaemonSet ausgeführt wird, ändern Sie die Fluent Bit DaemonSet Toleranzen:

```
tolerations:
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"
```

AWS Batch CloudWatch Container Insights

CloudWatch Container Insights sammelt, aggregiert und fasst Metriken und Protokolle aus Ihren AWS Batch Datenverarbeitungsumgebungen und -aufträgen zusammen. Zu den Metriken gehören CPU, Arbeitsspeicher, Festplatte und Netzwerkauslastung. Sie können diese Metriken zu CloudWatch Dashboards hinzufügen.

Betriebliche Daten werden als Performance-Protokollereignisse gesammelt. Dabei handelt es sich um Einträge mit einem strukturierten JSON-Schema, das die Aufnahme und Speicherung von Daten hoher Kardinalität in in großem Umfang ermöglicht. Aus diesen Daten CloudWatch erstellt aggregierte Metriken auf höherer Ebene auf Datenverarbeitungsumgebungs- und Auftragsebene als CloudWatch Metriken. Weitere Informationen finden Sie unter Strukturierte Container-Insights-Protokolle für Amazon ECS im Amazon- CloudWatch Benutzerhandbuch.

> **Important**
>
> CloudWatch Container Insights werden von als benutzerdefinierte Metriken berechnet CloudWatch. Weitere Informationen finden Sie unter Amazon CloudWatch Events – Preise

Aktivieren von Container Insights

Sie können Container Insights für AWS Batch Datenverarbeitungsumgebungen aktivieren.

1. Öffnen Sie die AWS Batch-Konsole.
2. Wählen Sie Datenverarbeitungsumgebungen aus.
3. Wählen Sie die gewünschte Datenverarbeitungsumgebung aus.
4. Aktivieren Sie für Container Insights Container Insights für die Datenverarbeitung -Umgebung.

> **Tip**
>
> Sie können ein Standardintervall auswählen, um die Metriken zu aggregieren oder ein benutzerdefiniertes zu erstellen
> Intervall.

Standardmäßig werden die folgenden Metriken angezeigt. Eine vollständige Liste der Amazon-ECS-Container-Insights-Metriken finden Sie unter Amazon-ECS-Container-Insights-Metriken im Amazon-CloudWatch Benutzerhandbuch.

- **`JobCount`** – Die Anzahl der Aufträge, die in der Datenverarbeitungsumgebung ausgeführt werden.
- **`ContainerInstanceCount`** – Die Anzahl der Amazon Elastic Compute Cloud-Instances, die den Amazon ECS-Agenten ausführen und in der Datenverarbeitungsumgebung registriert sind.
- **`MemoryReserved`** – Der Speicher, der von Datenverarbeitungsumgebungsaufträgen reserviert ist. Diese Metrik wird nur für die Aufträge erfasst, die in ihrer Auftragsdefinition eine definierte Speicherreservierung haben.
- **`MemoryUtilized`** – Der Speicher, der von Datenverarbeitungsumgebungsaufträgen verwendet wird. Diese Metrik wird nur für Aufträge erfasst, die eine definierte Speicherreservierung in ihrer Auftragsdefinition haben.
- **`CpuReserved`** – Die CPU-Einheiten, die von Datenverarbeitungsumgebungsaufträgen reserviert sind. Diese Metrik wird nur für Aufträge erfasst, die in ihrer Auftragsdefinition über eine definierte CPU-Reservierung verfügen.
- **`CpuUtilized`** – Die von Aufträgen in der Datenverarbeitungsumgebung verwendeten CPU-Einheiten. Diese Metrik wird nur für Aufträge erfasst, die in ihrer Auftragsdefinition über eine definierte CPU-Reservierung verfügen.
- **`NetworkRxBytes`** – Die Anzahl der empfangenen Bytes. Diese Metrik ist nur für Container in Aufträgen verfügbar, die die Netzwerkmodi `awsvpc` oder Bridge verwenden.
- **`NetworkTxBytes`** – Die Anzahl der übertragenen Bytes. Diese Metrik ist nur für Container in Aufträgen verfügbar, die die Netzwerkmodi `awsvpc` oder Bridge verwenden.
- **`StorageReadBytes`** – Die Anzahl der Bytes, die aus dem Speicher gelesen werden.
- **`StorageWriteBytes`** – Die Anzahl der Bytes, die in den Speicher geschrieben werden.

Protokollieren von AWS Batch-API-Aufrufen mit AWS CloudTrail

AWS Batch ist in integriert, einem ServiceAWS CloudTrail, der die Aktionen eines Benutzers, einer Rolle oder eines -AWSServices in aufzeichnetAWS Batch. CloudTrail erfasst alle API-Aufrufe für AWS Batch als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der AWS Batch-Konsole und Code-Aufrufe der AWS Batch-API-Operationen. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen an einen Amazon S3-Bucket aktivieren, einschließlich Ereignissen für AWS Batch. Wenn Sie keinen Trail konfigurieren, können Sie trotzdem die neuesten Ereignisse in der CloudTrail Konsole unter Ereignisverlauf anzeigen. Anhand der von CloudTrailgesammelten Informationen können Sie die an gestellte AnfrageAWS Batch, die IP-Adresse, von der die Anfrage gestellt wurde, den Initiator der Anfrage, den Zeitpunkt der Anfrage und zusätzliche Details bestimmen.

Weitere Informationen zu CloudTrailfinden Sie im AWS CloudTrail -Benutzerhandbuch.

AWS Batch-Informationen in CloudTrail

CloudTrail wird beim Erstellen des AWS Kontos in Ihrem Konto aktiviert. Wenn eine Aktivität in auftrittAWS Batch, wird diese Aktivität in einem - CloudTrail Ereignis zusammen mit anderen -AWSServiceereignissen im Ereignisverlauf aufgezeichnet. Sie können die neusten Ereignisse in Ihr AWS-Konto herunterladen und dort suchen und anzeigen. Weitere Informationen finden Sie unter Anzeigen von Ereignissen mit dem CloudTrail Ereignisverlauf.

Zur kontinuierlichen Aufzeichnung von Ereignissen in Ihrem AWS-Konto, einschließlich Ereignissen für AWS Batch, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Bereitstellung von Protokolldateien an einen Amazon S3-Bucket. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser für alle AWS-Regionen. Der Trail protokolliert Ereignisse aus allen Regionen in der AWS-Partition und stellt die Protokolldateien in dem von Ihnen angegebenen Amazon-S3-Bucket bereit. Darüber hinaus können Sie andere -AWSServices konfigurieren, um die in den CloudTrail Protokollen erfassten Ereignisdaten weiter zu analysieren und entsprechend zu agieren. Weitere Informationen finden Sie hier:

- Übersicht zum Erstellen eines Trails
- CloudTrail Unterstützte Services und Integrationen
- Konfigurieren von Amazon SNS-Benachrichtigungen für CloudTrail

- Empfangen von CloudTrail Protokolldateien aus mehreren Regionen und Empfangen von CloudTrail Protokolldateien aus mehreren Konten

Alle -AWS BatchAktionen werden von protokolliert CloudTrail und sind in der https://docs.aws.amazon.com/batch/latest/APIReference/. dokumentiert. Zum Beispiel werden durch Aufrufe der `SubmitJob`-, `ListJobs`- und `DescribeJobs`-Abschnitte Einträge in den CloudTrail -Protokolldateien generiert.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Gibt an, ob die Anforderung mit Root- oder IAM-Benutzer-Anmeldeinformationen ausgeführt wurde.
- 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

Weitere Informationen finden Sie unter CloudTrail userIdentity-Element.

Grundlagen von AWS Batch-Protokolldateieinträgen

Ein Trail ist eine Konfiguration, die die Bereitstellung von Ereignissen als Protokolldateien an einen von Ihnen angegebenen Amazon S3-Bucket ermöglicht. CloudTrail Protokolldateien enthalten einen oder mehrere Protokolleinträge. Ein Ereignis ist eine einzelne Anforderung aus einer beliebigen Quelle und enthält Informationen zur angeforderten Aktion, zu Datum und Uhrzeit der Aktion, zu den Anforderungsparametern usw. CloudTrail-Protokolldateien stellen kein geordnetes Stack-Trace der öffentlichen API-Aufrufe dar. Daher werden sie nicht in einer bestimmten Reihenfolge angezeigt.

Das folgende Beispiel zeigt einen - CloudTrail Protokolleintrag, der die `CreateComputeEnvironment` Aktion demonstriert.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE:admin",
    "arn": "arn:aws:sts::012345678910:assumed-role/Admin/admin",
    "accountId": "012345678910",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
```

```
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2017-12-20T00:48:46Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::012345678910:role/Admin",
        "accountId": "012345678910",
        "userName": "Admin"
      }
    }
  },
  "eventTime": "2017-12-20T00:48:46Z",
  "eventSource": "batch.amazonaws.com",
  "eventName": "CreateComputeEnvironment",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.1",
  "userAgent": "aws-cli/1.11.167 Python/2.7.10 Darwin/16.7.0 botocore/1.7.25",
  "requestParameters": {
    "computeResources": {
      "subnets": [
        "subnet-5eda8e04"
      ],
      "tags": {
        "testBatchTags": "CLI testing CE"
      },
      "desiredvCpus": 0,
      "minvCpus": 0,
      "instanceTypes": [
        "optimal"
      ],
      "securityGroupIds": [
        "sg-aba9e8db"
      ],
      "instanceRole": "ecsInstanceRole",
      "maxvCpus": 128,
      "type": "EC2"
    },
    "state": "ENABLED",
    "type": "MANAGED",
    "computeEnvironmentName": "Test"
  },
```

```
  "responseElements": {
    "computeEnvironmentName": "Test",
    "computeEnvironmentArn": "arn:aws:batch:us-east-1:012345678910:compute-environment/
Test"
  },
  "requestID": "890b8639-e51f-11e7-b038-EXAMPLE",
  "eventID": "874f89fa-70fc-4798-bc00-EXAMPLE",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "012345678910"
}
```

Erstellen einer Virtual Private Cloud

Datenverarbeitungsressourcen in Ihren Datenverarbeitungsumgebungen benötigen externen Netzwerkzugriff, um mit AWS Batch und Amazon-ECS-Service-Endpunkten kommunizieren zu können. Möglicherweise haben Sie jedoch Aufträge, die Sie in privaten Subnetzen ausführen möchten. Um die Flexibilität zu haben, Aufträge entweder in einem öffentlichen oder privaten Subnetz auszuführen, erstellen Sie eine VPC, die sowohl öffentliche als auch private Subnetze hat.

Sie können Amazon Virtual Private Cloud (Amazon VPC) verwenden, um AWS Ressourcen in einem von Ihnen definierten virtuellen Netzwerk zu starten. Dieses Thema enthält einen Link zum Amazon-VPC-Assistenten und eine Liste der auszuwählenden Optionen.

Erstellen einer VPC

Informationen zum Erstellen einer Amazon VPC finden Sie unter Erstellen einer VPC nur im Amazon-VPC-Benutzerhandbuch und verwenden Sie die folgende Tabelle, um zu bestimmen, welche Optionen ausgewählt werden sollen.

Option	Wert
Zu erstellende Ressourcen	Nur VPC
Name	Geben Sie optional einen Namen für Ihre VPC ein.
IPv4 CIDR block (IPv4-CIDR-Block)	Manuelle IPv4-CIDR-Eingabe Die CIDR-Blockgröße muss eine Größe zwischen /16 und /28. haben.
IPv6-CIDR-Block	Kein IPv6-CIDR-Block
Tenancy	Standard

Weitere Informationen über Amazon VPC finden Sie unter Was ist Amazon VPC? im Amazon VPC-Benutzerhandbuch.

Nächste Schritte

Nachdem Sie Ihre VPC erstellt haben, sollten Sie die folgenden nächsten Schritte in Betracht ziehen:

- Erstellen von Sicherheitsgruppen für Ihre öffentlichen und privaten Ressourcen, wenn diese eingehenden Netzwerkzugriff benötigen. Weitere Informationen finden Sie unter Arbeiten mit Sicherheitsgruppen im Amazon-VPC-Benutzerhandbuch.
- Erstellen Sie eine verwaltete AWS Batch-Umgebung, die Datenverarbeitungsressourcen in Ihrer neuen VPC startet. Weitere Informationen finden Sie unter Eine Rechenumgebung erstellen. Wenn Sie den Assistenten zur Erstellung der Datenverarbeitungsumgebung in der AWS Batch Konsole verwenden, können Sie die soeben erstellte VPC sowie die öffentlichen oder privaten Subnetze angeben, in denen Sie Ihre Instances starten möchten.
- Erstellen Sie eine -AWS BatchAuftragswarteschlange, die Ihrer neuen Computing-Umgebung zugeordnet ist. Weitere Informationen finden Sie unter Erstellen einer Auftragswarteschlange.
- Erstellen einer Auftragsdefinition für die Ausführung Ihrer Aufträge. Weitere Informationen finden Sie unter Erstellen einer Auftragsdefinition mit einem Knoten .
- Übermitteln eines Auftrags mit Ihrer Auftragsdefinition in die neue Auftragswarteschlange. Dieser Auftrag landet in der Datenverarbeitungsumgebung, die Sie mit Ihrer neuen VPC und Ihren Subnetzen erstellt haben. Weitere Informationen finden Sie unter Einen Job einreichen.

Sicherheit in AWS Batch

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von Rechenzentren und Netzwerkarchitekturen, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame AWS Verantwortung von Ihnen und Ihnen. Das Modell der übergreifenden Verantwortlichkeit 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 . Weitere Informationen zu den Compliance-Programmen, die für gelten AWS Batch, finden Sie unter AWS Services im Umfang nach Compliance-Programmen AWS .
- Sicherheit in der Cloud: Ihr Verantwortungsumfang wird durch den AWS -Service bestimmt, den Sie verwenden. 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 gemeinsamen Verantwortung bei der Verwendung anwenden können AWS Batch. In den folgenden Themen erfahren Sie, wie Sie die Konfiguration vornehmen AWS Batch , um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere AWS Dienste nutzen können, die Sie bei der Überwachung und Sicherung Ihrer AWS Batch Ressourcen unterstützen.

Themen

- Identity and Access Management für AWS Batch
- Zugriff AWS Batch über einen Schnittstellenendpunkt
- Konformitätsprüfung für AWS Batch
- Sicherheit der Infrastruktur in AWS Batch

Identity and Access Management für AWS Batch

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf Ressourcen sicher zu AWS kontrollieren. IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um Ressourcen zu verwenden. AWS Batch IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

Zielgruppe

Die Art und Weise, wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von der Arbeit ab, in der Sie tätig sind. AWS Batch

Dienstbenutzer — Wenn Sie den AWS Batch Dienst für Ihre Arbeit verwenden, stellt Ihnen Ihr Administrator die erforderlichen Anmeldeinformationen und Berechtigungen zur Verfügung. Wenn Sie für Ihre Arbeit mehr AWS Batch Funktionen verwenden, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Fuktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anzufordern müssen. Unter Fehlerbehebung bei AWS Batch Identität und Zugriff finden Sie nützliche Informationen für den Fall, dass Sie keinen Zugriff auf eine Feature in AWS Batch haben.

Serviceadministrator — Wenn Sie in Ihrem Unternehmen für die AWS Batch Ressourcen verantwortlich sind, haben Sie wahrscheinlich vollen Zugriff auf AWS Batch. Es ist Ihre Aufgabe, zu bestimmen, auf welche AWS Batch Funktionen und Ressourcen Ihre Servicebenutzer zugreifen sollen. Sie müssen dann Anträge an Ihren IAM-Administrator stellen, um die Berechtigungen Ihrer Servicenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die Grundkonzepte von

IAM nachzuvollziehen. Weitere Informationen darüber, wie Ihr Unternehmen IAM nutzen kann AWS Batch, finden Sie unterWie AWS Batch funktioniert mit IAM.

IAM-Administrator: Wenn Sie als IAM-Administrator fungieren, sollten Sie Einzelheiten dazu kennen, wie Sie Richtlinien zur Verwaltung des Zugriffs auf AWS Batch verfassen können. Beispiele für AWS Batch identitätsbasierte Richtlinien, die Sie in IAM verwenden können, finden Sie unter. Beispiele für identitätsbasierte Richtlinien für AWS Batch

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen als IAM-Benutzer authentifiziert (angemeldet AWS) sein oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich AWS als föderierte Identität anmelden, indem Sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt wurden. AWS IAM Identity Center (IAM Identity Center) -Benutzer, die Single Sign-On-Authentifizierung Ihres Unternehmens und Ihre Google- oder Facebook-Anmeldeinformationen sind Beispiele für föderierte Identitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie über den Verbund darauf zugreifen AWS , übernehmen Sie indirekt eine Rolle.

Je nachdem, welcher Benutzertyp Sie sind, können Sie sich beim AWS Management Console oder beim AWS Zugangsportal anmelden. Weitere Informationen zur Anmeldung finden Sie AWS unter So melden Sie sich bei Ihrem an AWS-Konto im AWS-Anmeldung Benutzerhandbuch.

Wenn Sie AWS programmgesteuert darauf zugreifen, AWS stellt es ein Software Development Kit (SDK) und eine Befehlszeilenschnittstelle (CLI) bereit, um Ihre Anfragen mithilfe Ihrer Anmeldeinformationen kryptografisch zu signieren. Wenn Sie keine AWS Tools verwenden, müssen Sie Anfragen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen Methode, um Anfragen selbst zu signieren, finden Sie im IAM-Benutzerhandbuch unter AWS API-Anfragen signieren.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen angeben. AWS Empfiehlt beispielsweise, die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere Informationen finden Sie unter Multi-Faktor-Authentifizierung im AWS IAM Identity Center - Benutzerhandbuch und Verwenden der Multi-Faktor-Authentifizierung (MFA) in AWS im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-Benutzer bezeichnet. Sie können darauf zugreifen, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen und verwenden Sie diese, um die Aufgaben auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Aufgaben, die Root-Benutzer-Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

Verbundidentität

Als bewährte Methode sollten menschliche Benutzer, einschließlich Benutzer, die Administratorzugriff benötigen, für den Zugriff AWS-Services mithilfe temporärer Anmeldeinformationen den Verbund mit einem Identitätsanbieter verwenden.

Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensbenutzerverzeichnis, einem Web-Identitätsanbieter AWS Directory Service, dem Identity Center-Verzeichnis oder einem beliebigen Benutzer, der mithilfe AWS-Services von Anmeldeinformationen zugreift, die über eine Identitätsquelle bereitgestellt wurden. Wenn föderierte Identitäten darauf zugreifen AWS-Konten, übernehmen sie Rollen, und die Rollen stellen temporäre Anmeldeinformationen bereit.

Für die zentrale Zugriffsverwaltung empfehlen wir Ihnen, AWS IAM Identity Center zu verwenden. Sie können Benutzer und Gruppen in IAM Identity Center erstellen, oder Sie können eine Verbindung zu einer Gruppe von Benutzern und Gruppen in Ihrer eigenen Identitätsquelle herstellen und diese synchronisieren, um sie in all Ihren AWS-Konten Anwendungen zu verwenden. Informationen zu IAM Identity Center finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center - Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität innerhalb Ihres Unternehmens AWS-Konto , die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. Wenn möglich, empfehlen wir, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bei speziellen Anwendungsfällen, die langfristige Anmeldeinformationen mit IAM-Benutzern erfordern, empfehlen wir jedoch, die Zugriffsschlüssel zu rotieren. Weitere Informationen finden Sie unter Regelmäßiges

Rotieren von Zugriffsschlüsseln für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

Eine IAM-Gruppe ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen vereinfachen die Verwaltung von Berechtigungen, wenn es zahlreiche Benutzer gibt. Sie könnten beispielsweise einer Gruppe mit dem Namen IAMAdmins Berechtigungen zum Verwalten von IAM-Ressourcen erteilen.

Benutzer unterscheiden sich von Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter Erstellen eines IAM-Benutzers (anstatt einer Rolle) im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität innerhalb Ihres Unternehmens AWS-Konto , die über bestimmte Berechtigungen verfügt. Sie ist einem IAM-Benutzer vergleichbar, ist aber nicht mit einer bestimmten Person verknüpft. Sie können vorübergehend eine IAM-Rolle in der übernehmen, AWS Management Console indem Sie die Rollen wechseln. Sie können eine Rolle übernehmen, indem Sie eine AWS CLI oder AWS API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter Verwenden von IAM-Rollen im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- Verbundbenutzerzugriff – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter Erstellen von Rollen für externe Identitätsanbieter im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter Berechtigungssätze im AWS IAM Identity Center -Benutzerhandbuch.
- Temporäre IAM-Benutzerberechtigungen – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.

- Kontoübergreifender Zugriff – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Bei einigen können Sie AWS-Services jedoch eine Richtlinie direkt an eine Ressource anhängen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.
- Serviceübergreifender Zugriff — Einige AWS-Services verwenden Funktionen in anderen AWS-Services. Wenn Sie beispielsweise einen Aufruf in einem Service tätigen, führt dieser Service häufig Anwendungen in Amazon-EC2 aus oder speichert Objekte in Amazon-S3. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.
  - Forward Access Sessions (FAS) — Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.
  - Servicerolle – Eine Servicerolle ist eine IAM-Rolle, die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.
  - Dienstbezogene Rolle — Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen erscheinen in Ihrem Dienst AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.
- Anwendungen, die auf Amazon EC2 ausgeführt werden — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2-Instance ausgeführt werden und API-Anfragen stellen AWS CLI . AWS Das ist eher zu empfehlen, als Zugriffsschlüssel innerhalb der EC2-Instance zu speichern. Um einer EC2-Instance eine AWS

Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein Instance-Profil, das an die Instance angehängt ist. Ein Instance-Profil enthält die Rolle und ermöglicht, dass Programme, die in der EC2-Instance ausgeführt werden, temporäre Anmeldeinformationen erhalten. Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon-EC2-Instances ausgeführt werden im IAM-Benutzerhandbuch.

Informationen dazu, wann Sie IAM-Rollen oder IAM-Benutzer verwenden sollten, finden Sie unter Erstellen einer IAM-Rolle (anstatt eines Benutzers) im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder Ressource zugeordnet ist, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anfrage stellt. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu Struktur und Inhalten von JSON-Richtliniendokumenten finden Sie unter Übersicht über JSON-Richtlinien im IAM-Benutzerhandbuch.

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das bedeutet, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die `iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Rolleninformationen von der AWS Management Console AWS CLI, der oder der AWS API abrufen.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern,

welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Inline-Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem System zuordnen können AWS-Konto. Zu den verwalteten Richtlinien gehören AWS verwaltete Richtlinien und vom Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie und einer eingebundenen Richtlinie wählen, finden Sie unter Auswahl zwischen verwalteten und eingebundenen Richtlinien im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Zugriffssteuerungslisten (ACLs)

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3 und Amazon VPC sind Beispiele für Services, die ACLs unterstützen. AWS WAF Weitere Informationen“ zu ACLs finden Sie unter Zugriffskontrollliste (ACL) – Übersicht (Access Control List) im Amazon-Simple-Storage-Service-Entwicklerhandbuch.

Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger verbreitete Richtlinientypen. Diese Richtlinientypen können die maximalen Berechtigungen festlegen, die Ihnen von den häufiger verwendeten Richtlinientypen erteilt werden können.

- Berechtigungsgrenzen – Eine Berechtigungsgrenze ist ein erweitertes Feature, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die daraus resultierenden Berechtigungen sind der Schnittpunkt der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im IAM-Benutzerhandbuch.
- Service Control Policies (SCPs) — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in festlegen. AWS Organizations AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer Objekte AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie innerhalb einer Organisation alle Features aktivieren, können Sie Service-Kontrollrichtlinien (SCPs) auf alle oder einzelne Ihrer Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Entitäten. Root-Benutzer des AWS-Kontos Weitere Informationen zu Organizations und SCPs finden Sie unter Funktionsweise von SCPs im AWS Organizations -Benutzerhandbuch.
- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Informationen darüber, wie AWS bestimmt wird,

ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie im IAM-Benutzerhandbuch unter Bewertungslogik für Richtlinien.

Wie AWS Batch funktioniert mit IAM

Bevor Sie IAM zur Verwaltung des Zugriffs auf verwenden, sollten Sie sich darüber informieren AWS Batch, mit welchen IAM-Funktionen Sie arbeiten können. AWS Batch

IAM-Funktionen, die Sie mit verwenden können AWS Batch

IAM-Feature	AWS Batch Unterstützung
Identitätsbasierte Richtlinien	Ja
Ressourcenbasierte Richtlinien	Nein
Richtlinienaktionen	Ja
Richtlinienressourcen	Ja
Bedingungsschlüssel für die Richtlinie	Ja
ACLs	Nein
ABAC (Tags in Richtlinien)	Ja
Temporäre Anmeldeinformationen	Ja
Hauptberechtigungen	Ja
Servicerollen	Ja
Service-verknüpfte Rollen	Ja

Einen allgemeinen Überblick darüber, wie AWS Batch und andere AWS Dienste mit den meisten IAM-Funktionen funktionieren, finden Sie im IAM-Benutzerhandbuch unter AWS Dienste, die mit IAM funktionieren.

Identitätsbasierte Richtlinien für AWS Batch

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 Erstellen von IAM-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. Sie können den Prinzipal nicht in einer identitätsbasierten Richtlinie angeben, da er für den Benutzer oder die Rolle gilt, dem er zugeordnet ist. 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 AWS Batch

Beispiele für AWS Batch identitätsbasierte Richtlinien finden Sie unter. Beispiele für identitätsbasierte Richtlinien für AWS Batch

Politische Maßnahmen für AWS Batch

Unterstützt Richtlinienaktionen	Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Richtlinienaktionen haben normalerweise denselben Namen wie der zugehörige AWS API-Vorgang. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keinen passenden API-Vorgang gibt. Es gibt auch einige Operationen, die mehrere Aktionen in einer Richtlinie erfordern. Diese zusätzlichen Aktionen werden als abhängige Aktionen bezeichnet.

Schließen Sie Aktionen in eine Richtlinie ein, um Berechtigungen zur Durchführung der zugeordneten Operation zu erteilen.

Eine Liste der AWS Batch Aktionen finden Sie unter Definierte Aktionen von AWS Batch in der Serviceautorisierungsreferenz.

Bei Richtlinienaktionen wird vor der Aktion das folgende Präfix AWS Batch verwendet:

```
batch
```

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Kommata:

```
"Action": [
      "batch:action1",
      "batch:action2"
         ]
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `Describe` beginnen, einschließlich der folgenden Aktion:

```
"Action": "batch:Describe*"
```

Beispiele für AWS Batch identitätsbasierte Richtlinien finden Sie unter. Beispiele für identitätsbasierte Richtlinien für AWS Batch

Politische Ressourcen für AWS Batch

Unterstützt Richtlinienressourcen	Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das bedeutet die Festlegung, welcher Prinzipal Aktionen für welche Ressourcen unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Anweisungen müssen entweder ein – `Resource`oder ein `NotResource`-Element enthalten.

Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Sie können dies für Aktionen tun, die einen bestimmten Ressourcentyp unterstützen, der als Berechtigungen auf Ressourcenebene bezeichnet wird.

Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, z. B. Auflistungsoperationen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Eine Liste der AWS Batch Ressourcentypen und ihrer ARNs finden Sie unter Resources Defined by AWS Batch in der Service Authorization Reference. Informationen zu den Aktionen, mit denen Sie den ARN einzelner Ressourcen angeben können, finden Sie unter Von AWS Batch definierte Aktionen.

Beispiele für AWS Batch identitätsbasierte Richtlinien finden Sie unter. Beispiele für identitätsbasierte Richtlinien für AWS Batch

Richtlinien-Bedingungsschlüssel für AWS Batch

Unterstützt servicespezifische Richtlini enbedingungsschlüssel	Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Condition` (oder `Condition` block) ermöglicht Ihnen die Angabe der Bedingungen, unter denen eine Anweisung wirksam ist. Das Element `Condition` ist optional. Sie können bedingte Ausdrücke erstellen, die Bedingungsoperatoren verwenden, z. B. ist gleich oder kleiner als, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt.

Wenn Sie mehrere `Condition`-Elemente in einer Anweisung oder mehrere Schlüssel in einem einzelnen `Condition`-Element angeben, wertet AWS diese mittels einer logischen AND-Operation aus. Wenn Sie mehrere Werte für einen einzelnen Bedingungsschlüssel angeben, AWS wertet die

Bedingung mithilfe einer logischen OR Operation aus. Alle Bedingungen müssen erfüllt werden, bevor die Berechtigungen der Anweisung gewährt werden.

Sie können auch Platzhaltervariablen verwenden, wenn Sie Bedingungen angeben. Beispielsweise können Sie einem IAM-Benutzer die Berechtigung für den Zugriff auf eine Ressource nur dann gewähren, wenn sie mit dessen IAM-Benutzernamen gekennzeichnet ist. Weitere Informationen finden Sie unter IAM-Richtlinienelemente: Variablen und Tags im IAM-Benutzerhandbuch.

AWS unterstützt globale Bedingungsschlüssel und dienstspezifische Bedingungsschlüssel. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter Kontextschlüssel für AWS globale Bedingungen im IAM-Benutzerhandbuch.

Eine Liste der AWS Batch Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für AWS Batch in der Service Authorization Reference. Informationen zu den Aktionen und Ressourcen, mit denen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter Definierte Aktionen von AWS Batch.

Beispiele für AWS Batch identitätsbasierte Richtlinien finden Sie unter. Beispiele für identitätsbasierte Richtlinien für AWS Batch

Attributbasierte Zugriffskontrolle (ABAC) mit AWS Batch

Unterstützt ABAC (Tags in Richtlinien)	Ja

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen definiert werden. In werden AWS 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.

Wenn ein Service alle drei Bedingungsschlüssel für jeden Ressourcentyp unterstützt, lautet der Wert für den Service Ja. Wenn ein Service alle drei Bedingungsschlüssel für nur einige Ressourcentypen unterstützt, lautet der Wert Teilweise.

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.

Verwenden temporärer Anmeldeinformationen mit AWS Batch

Unterstützt temporäre Anmeldeinformationen	Ja

Einige funktionieren AWS-Services nicht, wenn Sie sich mit temporären Anmeldeinformationen anmelden. Weitere Informationen, einschließlich Informationen, die mit temporären Anmeldeinformationen AWS-Services funktionieren AWS-Services , finden Sie im IAM-Benutzerhandbuch unter Diese Option funktioniert mit IAM.

Sie verwenden temporäre Anmeldeinformationen, wenn Sie sich mit einer anderen AWS Management Console Methode als einem Benutzernamen und einem Passwort anmelden. Wenn Sie beispielsweise AWS über den Single Sign-On-Link (SSO) Ihres Unternehmens darauf zugreifen, werden bei diesem Vorgang automatisch temporäre Anmeldeinformationen erstellt. Sie erstellen auch automatisch temporäre Anmeldeinformationen, wenn Sie sich als Benutzer bei der Konsole anmelden und dann die Rollen wechseln. Weitere Informationen zum Wechseln von Rollen finden Sie unter Wechseln zu einer Rolle (Konsole) im IAM-Benutzerhandbuch.

Mithilfe der AWS API AWS CLI oder können Sie temporäre Anmeldeinformationen manuell erstellen. Sie können diese temporären Anmeldeinformationen dann für den Zugriff verwenden AWS. AWS empfiehlt, temporäre Anmeldeinformationen dynamisch zu generieren, anstatt langfristige Zugriffsschlüssel zu verwenden. Weitere Informationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen in IAM.

Serviceübergreifende Prinzipal-Berechtigungen für AWS Batch

Unterstützt Forward Access Sessions (FAS)	Ja

Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion

in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, kombiniert mit der Anforderung, Anfragen an nachgelagerte Dienste AWS-Service zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.

Servicerollen für AWS Batch

Unterstützt Servicerollen	Ja

Eine Servicerolle ist eine IAM-Rolle, die ein Service annimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

> **Warning**
>
> Durch das Ändern der Berechtigungen für eine Servicerolle kann die AWS Batch Funktionalität beeinträchtigt werden. Bearbeiten Sie Servicerollen nur, wenn AWS Batch eine Anleitung dazu gibt.

Dienstbezogene Rollen für AWS Batch

Unterstützt serviceverknüpfte Rollen	Ja

Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service 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.

Details zum Erstellen oder Verwalten von serviceverknüpften Rollen finden Sie unter AWS -Services, die mit IAM funktionieren. Suchen Sie in der Tabelle nach einem Service mit einem `Yes` in der Spalte Service-linked role (Serviceverknüpfte Rolle). Wählen Sie den Link Yes (Ja) aus, um die Dokumentation für die serviceverknüpfte Rolle für diesen Service anzuzeigen.

AWS Batch Ausführung, IAM-Rolle

Die Ausführungsrolle erteilt dem Amazon ECS-Container und den AWS Fargate Agenten die Erlaubnis, AWS API-Aufrufe in Ihrem Namen durchzuführen.

> **Note**
>
> Die Ausführungsrolle wird vom Amazon ECS-Container-Agenten Version 1.16.0 und höher unterstützt.

Die Ausführungs-IAM-Rolle ist je nach den Anforderungen Ihrer Aufgabe erforderlich. Sie können mehrere Ausführungsrollen für unterschiedliche Zwecke und Dienste haben, die mit Ihrem Konto verknüpft sind.

> **Note**
>
> Informationen zur Amazon ECS-Instance-Rolle finden Sie unterAmazon-ECS-Instance-Rolle. Informationen zu Servicerollen finden Sie unterWie AWS Batch funktioniert mit IAM.

Amazon ECS stellt die `AmazonECSTaskExecutionRolePolicy` verwaltete Richtlinie bereit. Diese Richtlinie enthält die erforderlichen Berechtigungen für die oben beschriebenen allgemeinen Anwendungsfälle. Für die unten beschriebenen speziellen Anwendungsfälle kann es erforderlich sein, Ihrer Ausführungsrolle Inline-Richtlinien hinzuzufügen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecr:GetAuthorizationToken",
        "ecr:BatchCheckLayerAvailability",
        "ecr:GetDownloadUrlForLayer",
        "ecr:BatchGetImage",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "*"
```

```
    }
  ]
}
```

Mithilfe des folgenden Verfahrens können Sie überprüfen, ob Ihr Konto bereits über die Ausführungsrolle verfügt, und bei Bedarf die verwaltete IAM-Richtlinie anhängen.

So prüfen Sie **`ecsTaskExecutionRole`** in der IAM-Konsole

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Rollen aus.
3. Suchen Sie in der Liste der Rollen nach `ecsTaskExecutionRole`. Wenn Sie die Rolle nicht finden können, finden Sie weitere Informationen unterDie IAM-Ausführungsrolle wird erstellt. Wenn Sie die Rolle gefunden haben, wählen Sie die Rolle aus, um die angehängten Richtlinien einzusehen.
4. Stellen Sie auf der Registerkarte Berechtigungen sicher, dass die von AmazonECS TaskExecution RolePolicy verwaltete Richtlinie mit der Rolle verknüpft ist. Wenn die Richtlinie angehängt ist, ist Ihre Ausführungsrolle ordnungsgemäß konfiguriert. Andernfalls führen Sie die folgenden Teilschritte aus, um die Richtlinie zuzuweisen.

   a. Wählen Sie „Berechtigungen hinzufügen“ und anschließend „Richtlinien anhängen“.

   b. Suchen Sie nach AmazonECS TaskExecution RolePolicy.

   c. Markieren Sie das Kästchen links neben der TaskExecutionRolePolicyAmazonECS-Richtlinie und wählen Sie Richtlinien anhängen aus.
5. Wählen Sie Trust Relationships (Vertrauensbeziehungen) aus.
6. Überprüfen Sie, dass die Vertrauensstellung die folgende Richtlinie enthält. Wenn das Vertrauensverhältnis mit der folgenden Richtlinie übereinstimmt, ist die Rolle korrekt konfiguriert. Wenn die Vertrauensstellung nicht übereinstimmt, wählen Sie Vertrauensrichtlinie bearbeiten, geben Sie Folgendes ein und wählen Sie Richtlinie aktualisieren aus.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
```

```
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Die IAM-Ausführungsrolle wird erstellt

Wenn Ihr Konto noch nicht über eine Ausführungsrolle verfügt, gehen Sie wie folgt vor, um die Rolle zu erstellen.

Um die **`ecsTaskExecutionRole`** IAM-Rolle zu erstellen

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Roles (Rolle) aus.
3. Wählen Sie Rolle erstellen aus.
4. Wählen Sie unter Vertrauenswürdiger Entitätstyp die Option AWS-Service.
5. Wählen Sie für Service oder Anwendungsfall EC2 aus. Wählen Sie dann erneut EC2.
6. Wählen Sie Weiter aus.
7. Suchen Sie für Berechtigungsrichtlinien nach AmazonECS TaskExecution RolePolicy.
8. Aktivieren Sie das Kontrollkästchen links neben der TaskExecutionRolePolicyAmazonECS-Richtlinie und klicken Sie dann auf Weiter.
9. Geben Sie als Rollenname den Text Rolle erstellen ein **`ecsTaskExecutionRole`** und wählen Sie dann Create role aus.

Beispiele für identitätsbasierte Richtlinien für AWS Batch

Benutzer und Rollen haben standardmäßig nicht die Berechtigung, AWS Batch -Ressourcen zu erstellen oder zu ändern. Sie können auch keine Aufgaben mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI) oder AWS API ausführen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Einzelheiten zu Aktionen und Ressourcentypen, die von definiert wurden AWS Batch, einschließlich des Formats der ARNs für jeden der Ressourcentypen, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Batch in der Service Authorization Reference.

Themen

- Bewährte Methoden für Richtlinien
- Verwenden der Konsole AWS Batch
- Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand AWS Batch Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder sie löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um Ihren Benutzern und Workloads zunächst Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter AWS -verwaltete Richtlinien oder AWS -verwaltete Richtlinien für Auftrags-Funktionen im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und

Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. AWS CloudFormation B. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.

- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung zum IAM Access Analyzer im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Verwenden der Konsole AWS Batch

Um auf die AWS Batch Konsole zugreifen zu können, benötigen Sie ein Mindestmaß an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, Details zu den AWS Batch Ressourcen in Ihrem aufzulisten und anzuzeigen AWS-Konto. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (Benutzer oder Rollen) mit dieser Richtlinie.

Sie müssen Benutzern, die nur die API AWS CLI oder die AWS API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die der API-Operation entsprechen, die die Benutzer ausführen möchten.

Um sicherzustellen, dass Benutzer und Rollen die AWS Batch Konsole weiterhin verwenden können, fügen Sie den Entitäten auch die AWS Batch `ConsoleAccess` oder die `ReadOnly` AWS verwaltete

Richtlinie hinzu. Weitere Informationen finden Sie unter Hinzufügen von Berechtigungen zu einem Benutzer im IAM-Benutzerhandbuch.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der AWS CLI AWS OR-API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Serviceübergreifende Confused-Deputy-Prävention

Das Confused-Deputy-Problem ist ein Sicherheitsproblem, bei dem eine juristische Stelle, die nicht über die Berechtigung zum Ausführen einer Aktion verfügt, eine privilegiertere juristische Stelle zwingen kann, die Aktion auszuführen. In kann AWS ein dienstübergreifender Identitätswechsel zu einem Problem mit dem verwirrten Stellvertreter führen. Ein dienstübergreifender Identitätswechsel kann auftreten, wenn ein Dienst (der Anruf-Dienst) einen anderen Dienst anruft (den aufgerufenen Dienst). Der aufrufende Service kann manipuliert werden, um seine Berechtigungen zu verwenden, um Aktionen auf die Ressourcen eines anderen Kunden auszuführen, für die er sonst keine Zugriffsberechtigung haben sollte. Um dies zu verhindern, bietet AWS Tools, mit denen Sie Ihre Daten für alle Services mit Serviceprinzipalen schützen können, die Zugriff auf Ressourcen in Ihrem Konto erhalten haben.

Wir empfehlen, die Kontextschlüssel `aws:SourceArn`und die `aws:SourceAccount`globalen Bedingungsschlüssel in Ressourcenrichtlinien zu verwenden, um die Berechtigungen einzuschränken, die der AWS Batch Ressource einen anderen Dienst gewähren. Wenn der `aws:SourceArn`-Wert die Konto-ID nicht enthält, z. B. einen Amazon-S3-Bucket-ARN, müssen Sie beide globale Bedingungskontextschlüssel verwenden, um Berechtigungen einzuschränken. Wenn Sie beide globale Bedingungskontextschlüssel verwenden und der `aws:SourceArn`-Wert die Konto-ID enthält, müssen der `aws:SourceAccount`-Wert und das Konto im `aws:SourceArn`-Wert dieselbe Konto-ID verwenden, wenn sie in der gleichen Richtlinienanweisung verwendet wird. Verwenden Sie `aws:SourceArn`, wenn Sie nur eine Ressource mit dem betriebsübergreifenden Zugriff verknüpfen möchten. Verwenden Sie `aws:SourceAccount`, wenn Sie zulassen möchten, dass Ressourcen in diesem Konto mit der betriebsübergreifenden Verwendung verknüpft werden.

Der Wert von `aws:SourceArn` muss die Ressource sein, die AWS Batch gespeichert wird.

Der effektivste Weg, um sich vor dem Confused-Deputy-Problem zu schützen, ist die Verwendung des globalen Bedingungskontext-Schlüssels `aws:SourceArn` mit dem vollständigen ARN der Ressource. Wenn Sie den vollständigen ARN der Ressource nicht kennen oder wenn Sie mehrere Ressourcen angeben, verwenden Sie den globalen Kontextbedingungsschlüssel `aws:SourceArn` mit Platzhalterzeichen (*) für die unbekannten Teile des ARN. z. B. `arn:aws:`*`servicename`*`:*:`*`123456789012`*`:*`.

Die folgenden Beispiele zeigen, wie Sie die Kontextschlüssel `aws:SourceArn` und die `aws:SourceAccount` globale Bedingung verwenden können, AWS Batch um das Problem des verwirrten Stellvertreters zu vermeiden.

Beispiel 1: Rolle für den Zugriff auf nur eine Rechenumgebung

Die folgende Rolle kann nur für den Zugriff auf eine Rechenumgebung verwendet werden. Der Jobname muss als angegeben werden*, da die Job-Warteschlange mehreren Rechenumgebungen zugeordnet werden kann.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "batch.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:batch:us-east-1:123456789012:compute-environment/testCE",
            "arn:aws:batch:us-east-1:123456789012:job/*"
          ]
        }
      }
    }
  ]
}
```

Beispiel 2: Rolle für den Zugriff auf mehrere Rechenumgebungen

Die folgende Rolle kann für den Zugriff auf mehrere Rechenumgebungen verwendet werden. Der Jobname muss als angegeben werden*, da die Job-Warteschlange mehreren Rechenumgebungen zugeordnet werden kann.

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

```
        "Service": "batch.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:batch:us-east-1:123456789012:compute-environment/*",
            "arn:aws:batch:us-east-1:123456789012:job/*"
          ]
        }
      }
    }
  ]
}
```

Fehlerbehebung bei AWS Batch Identität und Zugriff

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit AWS Batch und IAM auftreten können.

Themen

- Ich bin nicht autorisiert, eine Aktion in AWS Batch auszuführen.
- Ich bin nicht berechtigt, iam auszuführen: PassRole
- Ich möchte Personen außerhalb meines AWS Kontos den Zugriff auf meine AWS Batch Ressourcen ermöglichen

Ich bin nicht autorisiert, eine Aktion in AWS Batch auszuführen.

Wenn Ihnen AWS Management Console mitgeteilt wird, dass Sie nicht berechtigt sind, eine Aktion durchzuführen, müssen Sie sich an Ihren Administrator wenden, um Unterstützung zu erhalten. Ihr Administrator ist die Person, die Ihnen Ihren Benutzernamen und Ihr Passwort bereitgestellt hat.

Der folgende Beispielfehler tritt auf, wenn der `mateojackson`-Benutzer versucht, die Konsole zum Anzeigen von Details zu einer fiktiven *`my-example-widget`*-Ressource zu verwenden, jedoch nicht über `batch:`*`GetWidget`*-Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 batch:GetWidget on resource: my-example-widget
```

In diesem Fall bittet Mateo seinen Administrator um die Aktualisierung seiner Richtlinien, um unter Verwendung der Aktion *my-example-widget* auf die Ressource `batch:GetWidget` zugreifen zu können. Weitere Informationen zum Erteilen von Berechtigungen zur Weitergabe einer Rolle finden Sie unter Erteilen von Benutzerberechtigungen zur Übergabe einer Rolle an einen AWS Dienst.

Ich bin nicht berechtigt, iam auszuführen: PassRole

Wenn Sie die Fehlermeldung erhalten, dass Sie nicht zum Durchführen der `iam:PassRole`-Aktion autorisiert sind, müssen Ihre Richtlinien aktualisiert werden, um eine Rolle an AWS Batchübergeben zu können.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Dienst zu übergeben, anstatt eine neue Servicerolle oder eine dienstverknüpfte Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in AWS Batch auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich möchte Personen außerhalb meines AWS Kontos den Zugriff auf meine AWS Batch Ressourcen ermöglichen

Sie können eine Rolle erstellen, die Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation für den Zugriff auf Ihre Ressourcen verwenden können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Im Fall von Diensten, die ressourcenbasierte Richtlinien

oder Zugriffskontrolllisten (Access Control Lists, ACLs) verwenden, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob diese Funktionen AWS Batch unterstützt werden, finden Sie unterWie AWS Batch funktioniert mit IAM.
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.

Verwenden von serviceverknüpften Rollen für AWS Batch

AWS Batch verwendet AWS Identity and Access Management (IAM) serviceverknüpfte Rollen. Eine serviceverknüpfte Rolle ist ein einzigartiger Typ von IAM-Rolle, mit der direkt verknüpft ist. AWS Batch Mit Diensten verknüpfte Rollen sind vordefiniert AWS Batch und enthalten alle Berechtigungen, die der Dienst benötigt, um andere AWS Dienste in Ihrem Namen aufzurufen.

Eine dienstbezogene Rolle AWS Batch erleichtert die Einrichtung, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. AWS Batch definiert die Berechtigungen ihrer dienstbezogenen Rollen und AWS Batch kann, sofern nicht anders definiert, nur ihre Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

> **Note**
>
> Führen Sie einen der folgenden Schritte aus, um eine Servicerolle für eine AWS Batch Rechenumgebung anzugeben.

- Verwenden Sie eine leere Zeichenfolge für die Servicerolle. Auf diese Weise können AWS Batch Sie die Servicerolle erstellen.
- Geben Sie die Servicerolle im folgenden Format an:`arn:aws:iam::account_number:role/aws-service-role/batch.amazonaws.com/AWSServiceRoleForBatch`.

Weitere Informationen finden Sie the section called “Falscher Rollenname oder ARN” im AWS Batch Benutzerhandbuch.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dies schützt Ihre AWS Batch -Ressourcen, da Sie nicht versehentlich die Berechtigung für den Zugriff auf die Ressourcen entfernen können.

Informationen zu anderen Services, die serviceverknüpfte Rollen unterstützen, finden Sie unter AWS -Services, die mit IAM funktionieren. Suchen Sie nach den Services, für die Ja in der Spalte Serviceverknüpfte Rolle angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Dienstbezogene Rollenberechtigungen für AWS Batch

AWS Batch verwendet die benannte dienstverknüpfte Rolle. AWSServiceRoleForBatch Die AWSServiceRoleForBatchRolle ermöglicht es AWS Batch , AWS Ressourcen in Ihrem Namen zu erstellen und zu verwalten.

Die AWSServiceRoleForBatchdienstbezogene Rolle vertraut darauf, dass der `batch.amazonaws.com` Dienstprinzipal die Rolle übernimmt.

Die genannte IAM-Richtlinie BatchServiceRolePolicyermöglicht AWS Batch die Ausführung der folgenden Aktionen für bestimmte Ressourcen:

- `autoscaling`— Ermöglicht das AWS Batch Erstellen und Verwalten von Amazon EC2 Auto Scaling Scaling-Ressourcen. AWS Batch erstellt und verwaltet Amazon EC2 Auto Scaling Scaling-Gruppen für die meisten Computerumgebungen.
- `ec2`— Ermöglicht AWS Batch die Kontrolle des Lebenszyklus von Amazon EC2 EC2-Instances sowie die Erstellung und Verwaltung von Startvorlagen und Tags. AWS Batch erstellt und verwaltet EC2 Spot Fleet-Anfragen für einige EC2-Spot-Computerumgebungen.

- `ecs`- Ermöglicht AWS Batch die Erstellung und Verwaltung von Amazon ECS-Clustern, Aufgabendefinitionen und Aufgaben für die Auftragsausführung.
- `eks`- Ermöglicht AWS Batch die Beschreibung der Amazon EKS-Cluster-Ressource für Validierungen.
- `iam`- Ermöglicht AWS Batch die Validierung und Weitergabe von Rollen, die vom Eigentümer bereitgestellt wurden, an Amazon EC2, Amazon EC2 Auto Scaling und Amazon ECS.
- `logs`— Ermöglicht AWS Batch das Erstellen und Verwalten von Protokollgruppen und Protokollströmen für AWS Batch Jobs.

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter serviceverknüpfte Rollenberechtigungen im IAM-Benutzerhandbuch.

Erstellen einer serviceverknüpften Rolle für AWS Batch

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie sich CreateComputeEnvironment in der AWS Management Console AWS CLI, der oder der AWS API befinden und keinen Wert für den `serviceRole` Parameter angeben, AWS Batch wird die dienstbezogene Rolle für Sie erstellt.

> **Important**
>
> Diese serviceverknüpfte Rolle kann in Ihrem Konto erscheinen, wenn Sie eine Aktion in einem anderen Service abgeschlossen haben, der die von dieser Rolle unterstützten Features verwendet. Außerdem, wenn Sie den AWS Batch Dienst vor dem 10. März 2021 genutzt haben, als er begann, dienstbezogene Rollen zu unterstützen, dann AWS Batch haben Sie die AWSServiceRoleForBatch Rolle in Ihrem Konto erstellt. Weitere Informationen finden Sie unter Eine neue Rolle ist in meinem IAM-Konto erschienen.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie CreateComputeEnvironment, AWS Batch erstellt die dienstbezogene Rolle erneut für Sie.

Bearbeitung einer serviceverknüpften Rolle für AWS Batch

Mit AWS Batch können Sie die AWSServiceRoleForBatch dienstverknüpfte Rolle nicht bearbeiten. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach

der Erstellung einer serviceverknüpften Rolle nicht bearbeitet werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Um einer IAM-Entität zu ermöglichen, die Beschreibung der serviceverknüpften Rolle zu bearbeiten AWSServiceRoleForBatch

Fügen Sie der Berechtigungsrichtlinie die folgende Anweisung hinzu. Dadurch kann die IAM-Entität die Beschreibung einer serviceverknüpften Rolle bearbeiten.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:UpdateRoleDescription"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/batch.amazonaws.com/
AWSServiceRoleForBatch",
    "Condition": {"StringLike": {"iam:AWSServiceName": "batch.amazonaws.com"}}
}
```

Löschen einer serviceverknüpften Rolle für AWS Batch

Wir empfehlen, dass Sie diese Rolle löschen, wenn Sie eine Funktion oder einen Dienst nicht mehr verwenden müssen, für den eine dienstbezogene Rolle erforderlich ist. Auf diese Weise verfügen Sie nicht über eine ungenutzte Entität, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch die Ressourcen für Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

Um einer IAM-Entität zu ermöglichen, die mit dem AWSServiceRoleForBatch Service verknüpfte Rolle zu löschen

Fügen Sie der Berechtigungsrichtlinie die folgende Anweisung hinzu. Dadurch kann die IAM-Entität eine dienstverknüpfte Rolle löschen.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
```

```
    "Resource": "arn:aws:iam::*:role/aws-service-role/batch.amazonaws.com/
AWSServiceRoleForBatch",
    "Condition": {"StringLike": {"iam:AWSServiceName": "batch.amazonaws.com"}}
}
```

Bereinigen einer serviceverknüpften Rolle

Bevor Sie IAM verwenden können, um eine dienstverknüpfte Rolle zu löschen, müssen Sie zunächst bestätigen, dass die Rolle keine aktiven Sitzungen hat, und alle AWS Batch Computerumgebungen, die die Rolle verwenden, in allen AWS Regionen in einer einzigen Partition löschen.

So überprüfen Sie, ob die serviceverknüpfte Rolle über eine aktive Sitzung verfügt

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Rollen und dann den AWSServiceRoleForBatch Namen aus (nicht das Kontrollkästchen).
3. Wählen Sie auf der Seite Summary Access Advisor und überprüfen Sie die letzten Aktivitäten auf die serviceverknüpfte Rolle.

> **Note**
>
> Wenn Sie nicht wissen, ob die AWSServiceRoleForBatch Rolle verwendet AWS Batch wird, können Sie versuchen, die Rolle zu löschen. Wenn der Dienst die Rolle verwendet, kann die Rolle nicht gelöscht werden. Sie können die Regionen anzeigen, in denen die Rolle verwendet wird. Wenn die Rolle verwendet wird, müssen Sie warten, bis die Sitzung beendet wird, bevor Sie die Rolle löschen können. Die Sitzung für eine serviceverknüpfte Rolle können Sie nicht widerrufen.

Um AWS Batch Ressourcen zu entfernen, die von der AWSServiceRoleForBatch serviceverknüpften Rolle verwendet werden

Sie müssen alle AWS Batch Rechenumgebungen, die die AWSServiceRoleForBatch Rolle verwenden, in allen AWS Regionen löschen, bevor Sie die AWSServiceRoleForBatch Rolle löschen können.

1. Öffnen Sie die AWS Batch Konsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
3. Wählen Sie im Navigationsbereich Datenverarbeitungs-Umgebungen aus.

4. Wählen Sie die Rechenumgebung aus.
5. Wählen Sie Disable (deaktivieren) aus. Warten Sie, bis der Status auf DEAKTIVIERT wechselt.
6. Wählen Sie die Rechenumgebung aus.
7. Wählen Sie Löschen aus. Bestätigen Sie, dass Sie die Rechenumgebung löschen möchten, indem Sie Rechenumgebung löschen wählen.
8. Wiederholen Sie die Schritte 1—7 für alle Rechenumgebungen, die die serviceverknüpfte Rolle verwenden, in allen Regionen.

Löschen einer serviceverknüpften Rolle in IAM (Konsole)

Sie können die IAM-Konsole für das Löschen einer serviceverknüpften Rolle verwenden.

So löschen Sie eine serviceverknüpfte Rolle (Konsole)

1. Melden Sie sich bei der IAM-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich der IAM Console Roles (Rollen) aus. Aktivieren Sie dann das Kontrollkästchen neben AWSServiceRoleForBatch, nicht den Namen oder die Zeile selbst.
3. Wählen Sie Delete role (Rolle löschen) aus.
4. Überprüfen Sie im Bestätigungsdialogfeld die letzten Service-Zugriffsdaten, die zeigen, wann jede der ausgewählten Rollen zuletzt auf einen AWS-Service zugegriffen hat. Auf diese Weise können Sie leichter bestätigen, ob die Rolle derzeit aktiv ist. Wenn Sie fortfahren möchten, wählen Sie Yes, Delete aus, um die serviceverknüpfte Rolle zur Löschung zu übermitteln.
5. Sehen Sie sich die Benachrichtigungen in der IAM-Konsole an, um den Fortschritt der Löschung der serviceverknüpften Rolle zu überwachen. Da die Löschung der serviceverknüpften IAM-Rolle asynchron erfolgt, kann die Löschung nach dem Übermitteln der Rolle für die Löschung erfolgreich sein oder fehlschlagen.

   - Wenn der Vorgang erfolgreich ist, wird die Rolle aus der Liste entfernt und eine Benachrichtigung des Erfolgs oben auf der Seite angezeigt.
   - Wenn der Vorgang fehlschlägt, können Sie in den Benachrichtigungen View details oder View Resources auswählen, um zu erfahren, warum die Löschung fehlgeschlagen ist. Wenn die Löschung fehlschlägt, weil die Rolle Ressourcen des Services verwendet, enthält die Benachrichtigung eine Liste der Ressourcen – sofern der Service diese Informationen zurückgibt. Sie können dann die Ressourcen bereinigen und die Löschung erneut durchführen.

> **Note**
>
> Möglicherweise müssen Sie diesen Vorgang abhängig von den Informationen, die der Service zurückgibt, mehrmals wiederholen. Ihre serviceverknüpfte Rolle könnte beispielsweise sechs Ressourcen verwenden und Ihr Service Informationen zu fünf davon zurückgeben. Wenn Sie die fünf Ressourcen bereinigen und die Rolle erneut für den Löschvorgang übermitteln, schlägt die Löschung fehl und der Service gibt die eine verbleibende Ressource zurück. Ein Service kann alle Ressourcen zurückgeben oder nur einige bzw. gar keine.

- Wenn die Aufgabe fehlschlägt und die Benachrichtigung keine Liste der Ressourcen enthält, gibt der Service möglicherweise diese Informationen nicht zurück. Informationen zum Bereinigen von Ressourcen für diesen Service finden Sie unter AWS-Services , die mit IAM funktionieren. Um die Dokumentation der serviceverknüpften Rolle für den Service anzuzeigen, suchen Sie Ihren Service in der Tabelle und wählen Sie den Link Yes.

Löschen einer serviceverknüpften Rolle in IAM ()AWS CLI

Sie können IAM-Befehle von verwenden, um eine dienstverknüpfte AWS Command Line Interface Rolle zu löschen.

So löschen Sie eine serviceverknüpfte Rolle (CLI)

1. Da eine serviceverknüpfte Rolle nicht gelöscht werden kann, wenn sie verwendet wird oder ihr Ressourcen zugeordnet sind, müssen Sie eine Löschungsanforderung übermitteln. Diese Anforderung kann verweigert werden, wenn diese Bedingungen nicht erfüllt sind. Sie benötigen die `deletion-task-id` aus der Antwort, um den Status der Löschaufgabe zu überprüfen. Geben Sie den folgenden Befehl ein, um eine Löschanforderung für eine serviceverknüpfte Rolle zu übermitteln:

   ```
   $ aws iam delete-service-linked-role --role-name AWSServiceRoleForBatch
   ```

2. Verwenden Sie den folgenden Befehl, um den Status der Löschaufgabe zu überprüfen:

   ```
   $ aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
   ```

Der Status der Löschaufgabe kann `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED` oder `FAILED` lauten. Wenn die Löschung fehlschlägt, gibt der Aufruf den Grund zurück, sodass Sie das Problem beheben können. Wenn die Löschung fehlschlägt, weil die Rolle Ressourcen des Services verwendet, enthält die Benachrichtigung eine Liste der Ressourcen – sofern der Service diese Informationen zurückgibt. Sie können dann die Ressourcen bereinigen und die Löschung erneut durchführen.

> **Note**
>
> Möglicherweise müssen Sie diesen Vorgang abhängig von den Informationen, die der Service zurückgibt, mehrmals wiederholen. Ihre serviceverknüpfte Rolle könnte beispielsweise sechs Ressourcen verwenden und Ihr Service Informationen zu fünf davon zurückgeben. Wenn Sie die fünf Ressourcen bereinigen und die Rolle erneut für den Löschvorgang übermitteln, schlägt die Löschung fehl und der Service gibt die eine verbleibende Ressource zurück. Ein Dienst kann alle Ressourcen zurückgeben, einige davon. Oder es meldet möglicherweise keine Ressourcen. Informationen zum Bereinigen der Ressourcen für einen Dienst, der keine Ressourcen meldet, finden Sie unter AWS Dienste, die mit IAM funktionieren. Um die Dokumentation der serviceverknüpften Rolle für den Service anzuzeigen, suchen Sie Ihren Service in der Tabelle und wählen Sie den Link Yes.

Löschen einer dienstverknüpften Rolle in IAM (API)AWS

Sie können die IAM-API für das Löschen einer serviceverknüpften Rolle verwenden.

So löschen Sie eine serviceverknüpfte Rolle (API)

1. Rufen Sie an, um einen Löschantrag für eine dienstbezogene Rolle einzureichen. DeleteServiceLinkedRole Geben Sie in der Anfrage den AWSServiceRoleForBatch Rollennamen an.

   Da eine serviceverknüpfte Rolle nicht gelöscht werden kann, wenn sie verwendet wird oder ihr Ressourcen zugeordnet sind, müssen Sie eine Löschungsanforderung übermitteln. Diese Anforderung kann verweigert werden, wenn diese Bedingungen nicht erfüllt sind. Sie benötigen die `DeletionTaskId` aus der Antwort, um den Status der Löschaufgabe zu überprüfen.

2. Rufen Sie an, um den Status des Löschvorgangs zu überprüfen GetServiceLinkedRoleDeletionStatus. Geben Sie in der Anforderung die `DeletionTaskId` an.

   Der Status der Löschaufgabe kann `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED` oder `FAILED` lauten. Wenn die Löschung fehlschlägt, gibt der Aufruf den Grund zurück, sodass Sie das Problem beheben können. Wenn die Löschung fehlschlägt, weil die Rolle Ressourcen des Services verwendet, enthält die Benachrichtigung eine Liste der Ressourcen – sofern der Service diese Informationen zurückgibt. Sie können dann die Ressourcen bereinigen und die Löschung erneut durchführen.

   > **Note**
   >
   > Möglicherweise müssen Sie diesen Vorgang abhängig von den Informationen, die der Service zurückgibt, mehrmals wiederholen. Ihre serviceverknüpfte Rolle könnte beispielsweise sechs Ressourcen verwenden und Ihr Service Informationen zu fünf davon zurückgeben. Wenn Sie die fünf Ressourcen bereinigen und die Rolle erneut für den Löschvorgang übermitteln, schlägt die Löschung fehl und der Service gibt die eine verbleibende Ressource zurück. Ein Service kann alle Ressourcen zurückgeben oder nur einige bzw. gar keine. Informationen zum Bereinigen von Ressourcen für einen Service, der keine Ressourcen meldet, finden Sie unter AWS-Services , die mit IAM funktionieren. Um die Dokumentation der serviceverknüpften Rolle für den Service anzuzeigen, suchen Sie Ihren Service in der Tabelle und wählen Sie den Link Yes.

Unterstützte Regionen für AWS Batch serviceverknüpfte Rollen

AWS Batch unterstützt die Verwendung von dienstbezogenen Rollen in allen Regionen, in denen der Dienst verfügbar ist. Weitere Informationen finden Sie unter AWS Batch -Endpunkte.

AWS verwaltete Richtlinien für AWS Batch

Sie können AWS verwaltete Richtlinien verwenden, um die Verwaltung des Identitätszugriffs für Ihr Team und die bereitgestellten AWS Ressourcen zu vereinfachen. AWS verwaltete Richtlinien decken eine Vielzahl gängiger Anwendungsfälle ab, sind standardmäßig in Ihrem AWS Konto verfügbar und werden in Ihrem Namen verwaltet und aktualisiert. Sie können die Berechtigungen in AWS verwalteten Richtlinien nicht ändern. Wenn Sie mehr Flexibilität benötigen, können Sie alternativ vom

Kunden verwaltete IAM-Richtlinien erstellen. Auf diese Weise können Sie Ihrem Team bereitgestellte Ressourcen nur mit genau den Berechtigungen ausstatten, die es benötigt.

Weitere Informationen zu AWS verwalteten Richtlinien finden Sie im IAM-Benutzerhandbuch unter AWS Verwaltete Richtlinien.

AWS Services verwalten und aktualisieren AWS verwaltete Richtlinien in Ihrem Namen. In regelmäßigen Abständen fügen AWS Dienste einer AWS verwalteten Richtlinie zusätzliche Berechtigungen hinzu. AWS verwaltete Richtlinien werden höchstwahrscheinlich aktualisiert, wenn eine neue Funktion gestartet oder ein neuer Vorgang verfügbar wird. Diese Updates wirken sich automatisch auf alle Identitäten (Benutzer, Gruppen und Rollen) aus, denen die Richtlinie zugeordnet ist. Sie entfernen jedoch weder Berechtigungen noch beeinträchtigen sie Ihre bestehenden Berechtigungen.

AWS Unterstützt außerdem verwaltete Richtlinien für Jobfunktionen, die sich über mehrere Dienste erstrecken. Die `ReadOnlyAccess` AWS verwaltete Richtlinie bietet beispielsweise schreibgeschützten Zugriff auf alle AWS Dienste und Ressourcen. Wenn ein Dienst eine neue Funktion startet, werden nur Leseberechtigungen für neue Operationen und Ressourcen AWS hinzugefügt. Eine Liste und Beschreibungen der Richtlinien für Auftragsfunktionen finden Sie in Verwaltete AWS -Richtlinien für Auftragsfunktionen im IAM-Leitfaden.

AWS verwaltete Richtlinie: BatchServiceRolePolicy

Die BatchServiceRolePolicyverwaltete IAM-Richtlinie wird von der `AWSServiceRoleForBatch`serviceverknüpften Rolle verwendet. Auf diese Weise können AWS Batch Sie Aktionen in Ihrem Namen ausführen. Sie können diese Richtlinie nicht mit Ihren IAM-Entitäten verknüpfen. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für AWS Batch.

Diese Richtlinie ermöglicht AWS Batch es, die folgenden Aktionen an bestimmten Ressourcen durchzuführen:

- `autoscaling`— Ermöglicht das AWS Batch Erstellen und Verwalten von Amazon EC2 Auto Scaling Scaling-Ressourcen. AWS Batch erstellt und verwaltet Amazon EC2 Auto Scaling Scaling-Gruppen für die meisten Computerumgebungen.

- `ec2`— Ermöglicht AWS Batch die Kontrolle des Lebenszyklus von Amazon EC2 EC2-Instances sowie die Erstellung und Verwaltung von Startvorlagen und Tags. AWS Batch erstellt und verwaltet EC2 Spot Fleet-Anfragen für einige EC2-Spot-Computerumgebungen.
- `ecs`- Ermöglicht AWS Batch die Erstellung und Verwaltung von Amazon ECS-Clustern, Aufgabendefinitionen und Aufgaben für die Auftragsausführung.
- `eks`- Ermöglicht AWS Batch die Beschreibung der Amazon EKS-Cluster-Ressource für Validierungen.
- `iam`- Ermöglicht AWS Batch die Validierung und Weitergabe von Rollen, die vom Eigentümer bereitgestellt wurden, an Amazon EC2, Amazon EC2 Auto Scaling und Amazon ECS.
- `logs`— Ermöglicht AWS Batch das Erstellen und Verwalten von Protokollgruppen und Protokollströmen für AWS Batch Jobs.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSBatchPolicyStatement1",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeAccountAttributes",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeImages",
                "ec2:DescribeImageAttribute",
                "ec2:DescribeSpotInstanceRequests",
                "ec2:DescribeSpotFleetInstances",
                "ec2:DescribeSpotFleetRequests",
                "ec2:DescribeSpotPriceHistory",
                "ec2:DescribeSpotFleetRequestHistory",
                "ec2:DescribeVpcClassicLink",
                "ec2:DescribeLaunchTemplateVersions",
                "ec2:RequestSpotFleet",
                "autoscaling:DescribeAccountLimits",
                "autoscaling:DescribeAutoScalingGroups",
                "autoscaling:DescribeLaunchConfigurations",
                "autoscaling:DescribeAutoScalingInstances",
```

```
                "autoscaling:DescribeScalingActivities",
                "eks:DescribeCluster",
                "ecs:DescribeClusters",
                "ecs:DescribeContainerInstances",
                "ecs:DescribeTaskDefinition",
                "ecs:DescribeTasks",
                "ecs:ListClusters",
                "ecs:ListContainerInstances",
                "ecs:ListTaskDefinitionFamilies",
                "ecs:ListTaskDefinitions",
                "ecs:ListTasks",
                "ecs:DeregisterTaskDefinition",
                "ecs:TagResource",
                "ecs:ListAccountSettings",
                "logs:DescribeLogGroups",
                "iam:GetInstanceProfile",
                "iam:GetRole"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AWSBatchPolicyStatement2",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/batch/job*"
        },
        {
            "Sid": "AWSBatchPolicyStatement3",
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/batch/job*:log-stream:*"
        },
        {
            "Sid": "AWSBatchPolicyStatement4",
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateOrUpdateTags"
            ],
            "Resource": "*",
```

```
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement5",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn",
                        "ecs-tasks.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement6",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "spot.amazonaws.com",
                        "spotfleet.amazonaws.com",
                        "autoscaling.amazonaws.com",
                        "ecs.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement7",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateLaunchTemplate"
```

```
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement8",
            "Effect": "Allow",
            "Action": [
                "ec2:TerminateInstances",
                "ec2:CancelSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "ec2:DeleteLaunchTemplate"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement9",
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateLaunchConfiguration",
                "autoscaling:DeleteLaunchConfiguration"
            ],
            "Resource":
"arn:aws:autoscaling:*:*:launchConfiguration:*:launchConfigurationName/AWSBatch*"
        },
        {
            "Sid": "AWSBatchPolicyStatement10",
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:SetDesiredCapacity",
                "autoscaling:DeleteAutoScalingGroup",
                "autoscaling:SuspendProcesses",
                "autoscaling:PutNotificationConfiguration",
```

```
                "autoscaling:TerminateInstanceInAutoScalingGroup"
            ],
            "Resource":
"arn:aws:autoscaling:*:*:autoScalingGroup:*:autoScalingGroupName/AWSBatch*"
        },
        {
            "Sid": "AWSBatchPolicyStatement11",
            "Effect": "Allow",
            "Action": [
                "ecs:DeleteCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:StopTask"
            ],
            "Resource": "arn:aws:ecs:*:*:cluster/AWSBatch*"
        },
        {
            "Sid": "AWSBatchPolicyStatement12",
            "Effect": "Allow",
            "Action": [
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:StopTask"
            ],
            "Resource": "arn:aws:ecs:*:*:task-definition/*"
        },
        {
            "Sid": "AWSBatchPolicyStatement13",
            "Effect": "Allow",
            "Action": [
                "ecs:StopTask"
            ],
            "Resource": "arn:aws:ecs:*:*:task/*/*"
        },
        {
            "Sid": "AWSBatchPolicyStatement14",
            "Effect": "Allow",
            "Action": [
                "ecs:CreateCluster",
                "ecs:RegisterTaskDefinition"
            ],
            "Resource": "*",
            "Condition": {
```

```
                "Null": {
                    "aws:RequestTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement15",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:launch-template/*",
                "arn:aws:ec2:*:*:placement-group/*",
                "arn:aws:ec2:*:*:capacity-reservation/*",
                "arn:aws:ec2:*:*:elastic-gpu/*",
                "arn:aws:elastic-inference:*:*:elastic-inference-accelerator/*",
                "arn:aws:resource-groups:*:*:group/*"
            ]
        },
        {
            "Sid": "AWSBatchPolicyStatement16",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement17",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "*"
```

```
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": [
                        "RunInstances",
                        "CreateLaunchTemplate",
                        "RequestSpotFleet"
                    ]
                }
            }
        }
    ]
}
```

AWS verwaltete Richtlinie: AWSBatchServiceRoleRichtlinie

Die genannte Richtlinie für Rollenberechtigungen AWSBatchServiceRole AWS Batch ermöglicht es, die folgenden Aktionen für bestimmte Ressourcen durchzuführen:

Die AWSBatchServiceRoleverwaltete IAM-Richtlinie wird häufig von einer Rolle mit dem Namen verwendet AWSBatchServiceRoleund umfasst die folgenden Berechtigungen. Gemäß den standardmäßigen Sicherheitsempfehlungen zur Gewährung der geringsten Rechte kann die AWSBatchServiceRoleverwaltete Richtlinie als Leitfaden verwendet werden. Wenn eine der Berechtigungen, die in der verwalteten Richtlinie erteilt werden, für Ihren Anwendungsfall nicht erforderlich ist, erstellen Sie eine benutzerdefinierte Richtlinie, und fügen Sie nur die erforderlichen Berechtigungen hinzu. Diese AWS Batch verwaltete Richtlinie und Rolle können für die meisten Arten von Computerumgebungen verwendet werden, aber die Verwendung von dienstbezogenen Rollen wird bevorzugt less-error-prone, um einen besseren Umfang und eine bessere Verwaltungsoberfläche zu gewährleisten.

- `autoscaling`— Ermöglicht das AWS Batch Erstellen und Verwalten von Amazon EC2 Auto Scaling Scaling-Ressourcen. AWS Batch erstellt und verwaltet Amazon EC2 Auto Scaling Scaling-Gruppen für die meisten Computerumgebungen.
- `ec2`— Ermöglicht AWS Batch die Verwaltung des Lebenszyklus von Amazon EC2 EC2-Instances sowie die Erstellung und Verwaltung von Startvorlagen und Tags. AWS Batch erstellt und verwaltet EC2 Spot Fleet-Anfragen für einige EC2-Spot-Computerumgebungen.
- `ecs`- Ermöglicht AWS Batch die Erstellung und Verwaltung von Amazon ECS-Clustern, Aufgabendefinitionen und Aufgaben für die Auftragsausführung.

- `iam`- Ermöglicht AWS Batch die Validierung und Weitergabe von Rollen, die vom Eigentümer bereitgestellt wurden, an Amazon EC2, Amazon EC2 Auto Scaling und Amazon ECS.
- `logs`— Ermöglicht AWS Batch das Erstellen und Verwalten von Protokollgruppen und Protokollströmen für AWS Batch Jobs.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSBatchPolicyStatement1",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeAccountAttributes",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeImages",
                "ec2:DescribeImageAttribute",
                "ec2:DescribeSpotInstanceRequests",
                "ec2:DescribeSpotFleetInstances",
                "ec2:DescribeSpotFleetRequests",
                "ec2:DescribeSpotPriceHistory",
                "ec2:DescribeSpotFleetRequestHistory",
                "ec2:DescribeVpcClassicLink",
                "ec2:DescribeLaunchTemplateVersions",
                "ec2:CreateLaunchTemplate",
                "ec2:DeleteLaunchTemplate",
                "ec2:RequestSpotFleet",
                "ec2:CancelSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "ec2:TerminateInstances",
                "ec2:RunInstances",
                "autoscaling:DescribeAccountLimits",
                "autoscaling:DescribeAutoScalingGroups",
                "autoscaling:DescribeLaunchConfigurations",
                "autoscaling:DescribeAutoScalingInstances",
                "autoscaling:DescribeScalingActivities",
                "autoscaling:CreateLaunchConfiguration",
                "autoscaling:CreateAutoScalingGroup",
```

```
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:SetDesiredCapacity",
                "autoscaling:DeleteLaunchConfiguration",
                "autoscaling:DeleteAutoScalingGroup",
                "autoscaling:CreateOrUpdateTags",
                "autoscaling:SuspendProcesses",
                "autoscaling:PutNotificationConfiguration",
                "autoscaling:TerminateInstanceInAutoScalingGroup",
                "ecs:DescribeClusters",
                "ecs:DescribeContainerInstances",
                "ecs:DescribeTaskDefinition",
                "ecs:DescribeTasks",
                "ecs:ListAccountSettings",
                "ecs:ListClusters",
                "ecs:ListContainerInstances",
                "ecs:ListTaskDefinitionFamilies",
                "ecs:ListTaskDefinitions",
                "ecs:ListTasks",
                "ecs:CreateCluster",
                "ecs:DeleteCluster",
                "ecs:RegisterTaskDefinition",
                "ecs:DeregisterTaskDefinition",
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:UpdateContainerAgent",
                "ecs:DeregisterContainerInstance",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogGroups",
                "iam:GetInstanceProfile",
                "iam:GetRole"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AWSBatchPolicyStatement2",
            "Effect": "Allow",
            "Action": "ecs:TagResource",
            "Resource": [
                "arn:aws:ecs:*:*:task/*_Batch_*"
            ]
        },
```

```
        {
            "Sid": "AWSBatchPolicyStatement3",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn",
                        "ecs-tasks.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement4",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "spot.amazonaws.com",
                        "spotfleet.amazonaws.com",
                        "autoscaling.amazonaws.com",
                        "ecs.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement5",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
```

```
                    "ec2:CreateAction": "RunInstances"
                }
            }
        }
    ]
}
```

AWS verwaltete Richtlinie: AWSBatchFullAccess

Die AWSBatchFullAccessRichtlinie gewährt AWS Batch Aktionen uneingeschränkten Zugriff auf AWS Batch Ressourcen. Es gewährt auch Zugriff auf Beschreiben und Auflisten von Aktionen für Amazon EC2-, Amazon ECS-, Amazon EKS- und IAM-Services. CloudWatch Auf diese Weise können IAM-Identitäten, entweder Benutzer oder Rollen, AWS Batch verwaltete Ressourcen einsehen, die in ihrem Namen erstellt wurden. Schließlich ermöglicht diese Richtlinie auch, dass ausgewählte IAM-Rollen an diese Dienste übergeben werden.

Sie können eine Verbindung AWSBatchFullAccesszu Ihren IAM-Entitäten herstellen. AWS Batch ordnet diese Richtlinie auch einer Servicerolle zu, mit der Sie Aktionen AWS Batch in Ihrem Namen ausführen können.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "batch:*",
        "cloudwatch:GetMetricStatistics",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeVpcs",
        "ec2:DescribeImages",
        "ec2:DescribeLaunchTemplates",
        "ec2:DescribeLaunchTemplateVersions",
        "ecs:DescribeClusters",
        "ecs:Describe*",
        "ecs:List*",
        "eks:DescribeCluster",
        "eks:ListClusters",
        "logs:Describe*",
        "logs:Get*",
```

```
        "logs:TestMetricFilter",
        "logs:FilterLogEvents",
        "iam:ListInstanceProfiles",
        "iam:ListRoles"
      ],
      "Resource":"*"
    },
    {
      "Effect":"Allow",
      "Action":[
        "iam:PassRole"
      ],
      "Resource":[
        "arn:aws:iam::*:role/AWSBatchServiceRole",
        "arn:aws:iam::*:role/service-role/AWSBatchServiceRole",
        "arn:aws:iam::*:role/ecsInstanceRole",
        "arn:aws:iam::*:instance-profile/ecsInstanceRole",
        "arn:aws:iam::*:role/iaws-ec2-spot-fleet-role",
        "arn:aws:iam::*:role/aws-ec2-spot-fleet-role",
        "arn:aws:iam::*:role/AWSBatchJobRole*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "iam:CreateServiceLinkedRole"
      ],
      "Resource":"arn:aws:iam::*:role/*Batch*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "batch.amazonaws.com"
          }
      }
    }
  ]
}
```

AWS Batch Aktualisierungen der AWS verwalteten Richtlinien

Hier finden Sie Informationen zu Aktualisierungen der AWS verwalteten Richtlinien, die AWS Batch seit Beginn der Nachverfolgung dieser Änderungen durch diesen Dienst vorgenommen wurden.

Abonnieren Sie den RSS-Feed auf der Seite AWS Batch Dokumentenverlauf, um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhalten.

Änderung	Beschreibung	Datum
BatchServiceRolePolicyRichtlinie aktualisiert	Es wurde aktualisiert und bietet nun Unterstützung für die Beschreibung des Anforderungsverlaufs und der Amazon EC2 Auto Scaling Aktivitäten von Spot Fleet.	05. Dezember 20
AWSBatchServiceRoleRichtlinie hinzugefügt	Es wurden nun Kontoausweis-IDs hinzugefügt und AWS Batch Berechtigungen für `ec2:DescribeSpotFleetRequestHistory` und erteilt`autoscaling:DescribeScalingActivities` .	05. Dezember 20
BatchServiceRolePolicyRichtlinie aktualisiert	Aktualisiert, um Unterstützung für die Beschreibung von Amazon EKS-Clustern hinzuzufügen.	20. Oktober 2022
AWSBatchFullAccessRichtlinie aktualisiert	Aktualisiert, um Unterstützung für das Auflisten und Beschreiben von Amazon EKS-Clustern hinzuzufügen.	20. Oktober 2022
BatchServiceRolePolicyRichtlinie aktualisiert	Aktualisiert, um Unterstützung für Amazon EC2 EC2-Kapazitätsreservierungsgruppen hinzuzufügen, die von AWS Resource Groups verwaltet werden. Weitere Informationen finden Sie unter Arbeiten mit Kapazitätsreservierungsgruppen im Amazon EC2 EC2-Benutzerhandbuch.	18. Mai 2022
BatchServiceRolePolicyund die AWSBatchServiceRoleRichtlinien wurden aktualisiert	Es wurde aktualisiert, um Unterstützung für die Beschreibung des Status AWS Batch verwalteter Instances in Amazon	6. Dezember 2021

Änderung	Beschreibung	Datum
	EC2 hinzuzufügen, sodass fehlerhafte Instances ersetzt werden.	
BatchServiceRolePolicyRichtlinie aktualisiert	Aktualisiert, um Unterstützung für Platzierungsgruppen-, Kapazitätsreservierungs-, Elastic GPU- und Elastic Inference Inference-Ressourcen in Amazon EC2 hinzuzufügen.	26. März 2021
BatchServiceRolePolicyRichtlinie hinzugefügt	Mit der BatchServiceRolePolicyverwalteten Richtlinie für die AWSServiceRoleForBatchdienstverknüpfte Rolle können Sie eine dienstverknüpfte Rolle verwenden, die von verwaltet wird. AWS Batch Mit dieser Richtlinie müssen Sie Ihre eigene Rolle für die Verwendung in Ihren Computerumgebungen nicht beibehalten.	10. März 2021
AWSBatchFullAccess- Fügen Sie die Berechtigung hinzu, um eine serviceverknüpfte Rolle hinzuzufügen	Fügen Sie IAM-Berechtigungen hinzu, damit die AWSServiceRoleForBatchdienstverknüpfte Rolle dem Konto hinzugefügt werden kann.	10. März 2021
AWS Batch hat begonnen, Änderungen zu verfolgen	AWS Batch hat begonnen, Änderungen für die AWS verwalteten Richtlinien zu verfolgen.	10. März 2021

Zugriff AWS Batch über einen Schnittstellenendpunkt

Sie können verwenden AWS PrivateLink , um eine private Verbindung zwischen Ihrer VPC und AWS Batch herzustellen. Sie können darauf zugreifen, AWS Batch als ob es in Ihrer VPC wäre, ohne ein Internet-Gateway, ein NAT-Gerät, eine VPN-Verbindung oder AWS Direct Connect eine Verbindung zu verwenden. Instances in Ihrer VPC benötigen für den Zugriff AWS Batch keine öffentlichen IP-Adressen.

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. Hierbei handelt es sich um vom Anforderer verwaltete Netzwerkschnittstellen, die als Eingangspunkt für den Datenverkehr dienen, der für AWS Batch bestimmt ist.

Weitere Informationen finden Sie im Handbuch unter Interface VPC Endpoints.AWS PrivateLink

Überlegungen zu AWS Batch

Bevor Sie einen Schnittstellenendpunkt für einrichten AWS Batch, lesen Sie sich die Eigenschaften und Einschränkungen der Schnittstellenendpunkte im AWS PrivateLink Handbuch durch.

AWS Batch unterstützt Aufrufe aller API-Aktionen über den Schnittstellenendpunkt.

Bevor Sie Schnittstellen-VPC-Endpunkte für einrichten AWS Batch, sollten Sie die folgenden Überlegungen beachten:

- Jobs, die den Starttyp Fargate-Ressourcen verwenden, benötigen nicht die Schnittstellen-VPC-Endpunkte für Amazon ECS. Möglicherweise benötigen Sie jedoch Schnittstellen-VPC-Endpunkte für Amazon ECR AWS Batch, Secrets Manager oder Amazon CloudWatch Logs, die in den folgenden Punkten beschrieben werden.
  - Um Jobs auszuführen, müssen Sie die Schnittstellen-VPC-Endpunkte für Amazon ECS erstellen. Weitere Informationen finden Sie unter Interface VPC Endpoints (AWS PrivateLink) im Amazon Elastic Container Service Developer Guide.
  - Damit Ihre Jobs private Images von Amazon ECR abrufen können, müssen Sie die Schnittstellen-VPC-Endpunkte für Amazon ECR erstellen. Weitere Informationen finden Sie unter Schnittstellen-VPC-Endpunkte (AWS PrivateLink) im Amazon Elastic Container-Registry-Benutzerhandbuch.
  - Damit Ihre Jobs sensible Daten aus Secrets Manager abrufen können, müssen Sie die VPC-Schnittstellen-Endpunkte für Secrets Manager erstellen. Weitere Informationen finden Sie unter Verwenden von Secrets Manager mit VPC-Endpunkten im AWS Secrets Manager - Benutzerhandbuch.
  - Wenn Ihre VPC kein Internet-Gateway hat und Ihre Jobs den `awslogs` Protokolltreiber verwenden, um CloudWatch Protokollinformationen an Logs zu senden, müssen Sie einen VPC-Schnittstellen-Endpunkt für CloudWatch Logs erstellen. Weitere Informationen finden Sie unter Using CloudWatch Logs with Interface VPC Endpoints im Amazon CloudWatch Logs-Benutzerhandbuch.

- Jobs, die die EC2-Ressourcen verwenden, erfordern, dass die Container-Instances, auf denen sie gestartet werden, Version `1.25.1` oder höher des Amazon ECS-Container-Agenten ausführen. Weitere Informationen finden Sie unter Amazon ECS Linux Container Agent-Versionen im Amazon Elastic Container Service Developer Guide.
- VPC-Endpunkte unterstützen derzeit keine regionsübergreifenden Anforderungen. Stellen Sie sicher, dass Sie Ihren Endpunkt innerhalb derselben Region erstellen, in der Sie Ihre API-Aufrufe an AWS Batch ausgeben möchten.
- VPC-Endpunkte unterstützen nur von Amazon bereitgestellten DNS über Amazon Route 53. Wenn Sie Ihre eigene DNS verwenden möchten, können Sie die bedingte DNS-Weiterleitung nutzen. Weitere Informationen finden Sie unter DHCP Options Sets im Amazon VPC-Benutzerhandbuch.
- Die Sicherheitsgruppe für den VPC-Endpunkt müssen eingehende Verbindungen auf Port 443 aus dem privaten Subnetz der VPC zulassen.
- AWS Batch unterstützt im Folgenden keine VPC-Schnittstellenendpunkte: AWS-Regionen
  - Asien-Pazifik (Osaka) (`ap-northeast-3`)
  - Asien-Pazifik (Jakarta) (`ap-southeast-3`)

Erstellen Sie einen Schnittstellenendpunkt für AWS Batch

Sie können einen Schnittstellenendpunkt für die AWS Batch Verwendung entweder der Amazon VPC-Konsole oder der AWS Command Line Interface (AWS CLI) erstellen. Weitere Informationen finden Sie unter Erstellen eines Schnittstellenendpunkts im AWS PrivateLink -Leitfaden.

Erstellen Sie einen Schnittstellenendpunkt für die AWS Batch Verwendung des folgenden Servicenamens:

```
com.amazonaws.region.batch
```

Beispielsweise:

```
com.amazonaws.us-east-2.batch
```

In der `aws-cn` Partition ist das Format anders:

```
cn.com.amazonaws.region.batch
```

Beispielsweise:

```
cn.com.amazonaws.cn-northwest-1.batch
```

Wenn Sie privates DNS für den Schnittstellenendpunkt aktivieren, können Sie API-Anfragen an die AWS Batch Verwendung des standardmäßigen regionalen DNS-Namens stellen. z. B. `batch.us-east-1.amazonaws.com`.

Weitere Informationen finden Sie im AWS PrivateLink Handbuch unter Zugreifen auf einen Dienst über einen Schnittstellenendpunkt.

Erstellen einer Endpunktrichtlinie für Ihren Schnittstellen-Endpunkt

Eine Endpunktrichtlinie ist eine IAM-Ressource, die Sie an einen Schnittstellen-Endpunkt anfügen können. Die standardmäßige Endpunktrichtlinie ermöglicht den vollständigen Zugriff AWS Batch über den Schnittstellenendpunkt. Um den Zugriff AWS Batch von Ihrer VPC aus zu kontrollieren, fügen Sie dem Schnittstellenendpunkt eine benutzerdefinierte Endpunktrichtlinie hinzu.

Eine Endpunktrichtlinie gibt die folgenden Informationen an:

- Die Principals, die Aktionen ausführen können (AWS-Konten, Benutzer und IAM-Rollen).
- Aktionen, die ausgeführt werden können
- Die Ressourcen, auf denen die Aktionen ausgeführt werden können.

Weitere Informationen finden Sie unter Steuern des Zugriffs auf Services mit Endpunktrichtlinien im AWS PrivateLink -Leitfaden.

Beispiel: VPC-Endpunktrichtlinie für Aktionen AWS Batch

Im Folgenden finden Sie ein Beispiel für eine benutzerdefinierte Endpunktrichtlinie. Wenn Sie diese Richtlinie an Ihren Schnittstellenendpunkt anhängen, gewährt sie allen Prinzipalen auf allen Ressourcen Zugriff auf die aufgelisteten AWS Batch Aktionen.

```
{
   "Statement": [
      {
         "Principal": "*",
         "Effect": "Allow",
         "Action": [
            "batch:SubmitJob",
            "batch:ListJobs",
```

```
            "batch:DescribeJobs"
         ],
         "Resource":"*"
      }
   ]
}
```

Konformitätsprüfung für AWS Batch

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. AWS stellt die folgenden Ressourcen zur Verfügung, die Sie bei der Einhaltung der Vorschriften unterstützen:

- Schnellstartanleitungen zu Sicherheit und Compliance — In diesen Bereitstellungsleitfäden werden architektonische Überlegungen erörtert und Schritte für die Bereitstellung von Basisumgebungen beschrieben AWS , bei denen Sicherheit und Compliance im Mittelpunkt stehen.
- Architecting for HIPAA Security and Compliance on Amazon Web Services — In diesem Whitepaper wird beschrieben, wie Unternehmen HIPAA-fähige Anwendungen erstellen AWS können.

  > **Note**
  >
  > AWS-Services Nicht alle sind HIPAA-fähig. Weitere Informationen finden Sie in der Referenz für HIPAA-berechtigte Services.

- AWS Compliance-Ressourcen — Diese Sammlung von Arbeitsmappen und Leitfäden gilt möglicherweise für Ihre Branche und Ihren Standort.
- AWS Leitfäden zur Einhaltung von Vorschriften für Kunden — Verstehen Sie das Modell der gemeinsamen Verantwortung aus dem Blickwinkel der Einhaltung von Vorschriften. In den

Leitfäden werden die bewährten Verfahren zur Sicherung zusammengefasst AWS-Services und die Leitlinien den Sicherheitskontrollen in verschiedenen Frameworks (einschließlich des National Institute of Standards and Technology (NIST), des Payment Card Industry Security Standards Council (PCI) und der International Organization for Standardization (ISO)) zugeordnet.

- Evaluierung von Ressourcen anhand von Regeln im AWS Config Entwicklerhandbuch — Der AWS Config Service bewertet, wie gut Ihre Ressourcenkonfigurationen den internen Praktiken, Branchenrichtlinien und Vorschriften entsprechen.
- AWS Security Hub— Auf diese AWS-Service Weise erhalten Sie einen umfassenden Überblick über Ihren internen Sicherheitsstatus. AWS Security Hub verwendet Sicherheitskontrollen, um Ihre AWS -Ressourcen zu bewerten und Ihre Einhaltung von Sicherheitsstandards und bewährten Methoden zu überprüfen. Eine Liste der unterstützten Services und Kontrollen finden Sie in der Security-Hub-Steuerungsreferenz.
- Amazon GuardDuty — Dies AWS-Service erkennt potenzielle Bedrohungen für Ihre Workloads AWS-Konten, Container und Daten, indem es Ihre Umgebung auf verdächtige und böswillige Aktivitäten überwacht. GuardDuty kann Ihnen helfen, verschiedene Compliance-Anforderungen wie PCI DSS zu erfüllen, indem es die in bestimmten Compliance-Frameworks vorgeschriebenen Anforderungen zur Erkennung von Eindringlingen erfüllt.
- AWS Audit Manager— Auf diese AWS-Service Weise können Sie Ihre AWS Nutzung kontinuierlich überprüfen, um das Risikomanagement und die Einhaltung von Vorschriften und Industriestandards zu vereinfachen.

Sicherheit der Infrastruktur in AWS Batch

Als verwalteter Dienst AWS Batch ist er 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 für den Zugriff AWS Batch über das Netzwerk. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit AWS Security Token Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Sie können diese API-Operationen von jedem Netzwerkstandort aus aufrufen, AWS Batch unterstützt jedoch ressourcenbasierte Zugriffsrichtlinien, die Einschränkungen auf der Grundlage der Quell-IP-Adresse beinhalten können. Sie können auch AWS Batch Richtlinien verwenden, um den Zugriff von bestimmten Amazon Virtual Private Cloud (Amazon VPC) -Endpunkten oder bestimmten VPCs aus zu kontrollieren. Dadurch wird der Netzwerkzugriff auf eine bestimmte AWS Batch Ressource effektiv nur von der spezifischen VPC innerhalb des AWS Netzwerks isoliert.

Markieren Ihrer AWS Batch-Ressourcen

Um Sie bei der Verwaltung Ihrer AWS Batch-Ressourcen zu unterstützen, können Sie jeder Ressource eigene Metadaten in Form von Tags zuweisen. In diesem Thema werden Tags (Markierungen) und deren Erstellung beschrieben.

Inhalt

Grundlagen zu Tags (Markierungen)

Ein Tag (Markierung) ist eine Markierung, die Sie einer AWS-Ressource zuordnen. Jeder Tag (Markierung) besteht aus einem Schlüssel und einem optionalen Wert, beides können Sie bestimmen.

Mit Tags können Sie Ihre AWS-Ressourcen kategorisieren, z. B. nach Zweck, Eigentümer oder Umgebung. Wenn Sie viele Ressourcen desselben Typs haben, können Sie bestimmte Ressourcen basierend auf den zugewiesenen Tags schnell bestimmen. Sie können beispielsweise eine Reihe von Tags für Ihre AWS Batch-Cluster definieren. Diese helfen Ihnen, den Besitzer und die Stack-Ebene jedes einzelnen Clusters nachzuverfolgen. Sie sollten für jeden Ressourcentyp einen konsistenten Satz von Tag-Schlüsseln entwickeln.

Tags werden nicht automatisch Ihren Ressourcen zugewiesen. Nachdem Sie ein Tag hinzugefügt haben, können Sie jederzeit Tag-Schlüssel und -Werte bearbeiten oder Tags aus einer Ressource entfernen. Wenn Sie eine Ressource löschen, werden alle Tags (Markierungen) der Ressource ebenfalls gelöscht.

Tags haben keine semantische Bedeutung für AWS Batch und werden ausschließlich als Zeichenfolgen interpretiert. Sie können den Wert eines Tags (Markierung) zwar auf eine leere Zeichenfolge, jedoch nicht null festlegen. Wenn Sie ein Tag (Markierung) mit demselben Schlüssel wie ein vorhandener Tag (Markierung) für die Ressource hinzufügen, wird der alte Wert mit dem neuen überschrieben.

Sie können mit der AWS Management Console, AWS CLI und AWS Batch-API mit Tags arbeiten.

Wenn Sie AWS Identity and Access Management (IAM) verwenden, können Sie steuern, welche Benutzer in Ihrem AWS-Konto Tags erstellen, bearbeiten oder löschen dürfen.

Markieren Ihrer -Ressourcen

Sie können neue oder vorhandene AWS Batch Datenverarbeitungsumgebungen, Aufträge, Auftragsdefinitionen, Auftragswarteschlangen und Planungsrichtlinien markieren.

Wenn Sie die AWS Batch-Konsole verwenden, können Sie Tags auf neue Ressourcen anwenden, wenn sie erstellt werden, oder jederzeit auf bestehende Ressourcen, indem Sie die Registerkarte Tags auf der entsprechenden Ressourcenseite verwenden.

Wenn Sie die AWS Batch-API, die AWS CLI oder ein AWS-SDK verwenden, können Sie Tags mithilfe des Parameters `tags` auf neue Ressourcen oder mithilfe der API-Aktion `TagResource` auf vorhandene Ressourcen anwenden. Weitere Informationen finden Sie unter TagResource.

Bei einigen Aktionen zur Ressourcenerstellung können Sie Tags für eine Ressource angeben, wenn die Ressource erstellt wird. Wenn Tags während der Ressourcenerstellung nicht angewendet werden können, schlägt die Ressourcenerstellung fehl. Auf diese Weise wird sichergestellt, dass Ressourcen, die Sie bei der Erstellung markieren möchten, entweder mit angegebenen Tags oder gar nicht erstellt werden. Wenn Sie Ressourcen zum Zeitpunkt der Erstellung markieren, müssen Sie nach der Ressourcenerstellung keine benutzerdefinierten Tagging-Skripts ausführen.

In der folgenden Tabelle werden die markierbaren AWS Batch-Ressourcen und die bei Erstellung markierbaren Ressourcen beschrieben.

Markierungsunterstützung für AWS Batch-Ressourcen

Ressource	Unterstützt Tags (Markierungen)	Unterstützt Tag-Propagierung	Unterstützt das Markieren bei Erstellung (AWS Batch-API, AWS CLI, AWS-SDK)
AWS Batch Datenverarbeitungsumgebungen	Ja	Nein. Tags der Datenverarbeitungsumgebung werden nicht an andere	Ja

Ressource	Unterstützt Tags (Markierungen)	Unterstützt Tag-Propagierung	Unterstützt das Markieren bei Erstellung (AWS Batch-API, AWS CLI, AWS-SDK)
		Ressourcen weitergegeben. Tags für die Ressourcen werden im Tags-Mitglied des computeResources-Objekts angegeben, das in der CreateComputeEnvironment API-Operation übergeben wird.	
AWS Batch Aufträge	Ja	Ja	Ja
AWS Batch -Auftragsdefinitionen	Ja	Nein	Ja
AWS Batch-Auftragswarteschlangen	Ja	Nein	Ja
AWS Batch Planungsrichtlinien	Ja	Nein	Ja

Tag (Markierung)-Einschränkungen

Die folgenden grundlegenden Einschränkungen gelten für Tags (Markierungen):

- Maximale Anzahl von Tags (Markierungen) pro Ressource: 50
- Jeder Tag (Markierung) muss für jede Ressource eindeutig sein. Jeder Tag (Markierung) kann nur einen Wert haben.
- Maximale Schlüssellänge: 128 Unicode-Zeichen in UTF-8
- Maximale Wertlänge: 256 Unicode-Zeichen in UTF-8

- Wenn Ihr Markierungsschema für mehrere AWS-Services und -Ressourcen verwendet wird, denken Sie daran, dass andere Services möglicherweise Einschränkungen für zulässige Zeichen haben. Allgemein erlaubte Zeichen sind Buchstaben, Zahlen, Leerzeichen, die in UTF-8 darstellbar sind, sowie die folgenden Zeichen: + - = . _ : / @.
- Bei Tag-Schlüsseln und -Werten muss die Groß- und Kleinschreibung beachtet werden.
- Verwenden Sie weder `aws:` noch `AWS:` oder Kombinationen aus Groß- und Kleinbuchstaben von diesen als Präfix für Schlüssel oder Werte, da sie für die AWS-Verwendung reserviert sind. Sie können keine Tag-Schlüssel oder -Werte mit diesem Präfix bearbeiten oder löschen. Tags mit diesem Präfix werden nicht auf Ihr tags-per-resource Limit angerechnet.

Arbeiten mit Tags über die Konsole

Mit der AWS BatchKonsole können Sie die Tags verwalten, die neuen oder vorhandenen Datenverarbeitungsumgebungen, Aufträgen, Auftragsdefinitionen und Auftragswarteschlangen zugeordnet sind.

Hinzufügen von Tags zu einer einzelnen Ressource bei der Erstellung

Sie können AWS Batch Datenverarbeitungsumgebungen, Aufträgen, Auftragsdefinitionen, Auftragswarteschlangen und Planungsrichtlinien Tags hinzufügen, wenn Sie sie erstellen.

Hinzufügen und Löschen von Tags für einzelne Ressourcen

Mit AWS Batch können Sie Tags, die Ihren Clustern zugeordnet sind, direkt auf der Seite der Ressource hinzufügen oder löschen.

So fügen Sie ein Tag zu einer einzelnen Ressource hinzu oder löschen es

1. Öffnen Sie die -AWS BatchKonsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie in der Navigationsleiste die zu verwendende Region aus.
3. Wählen Sie im Navigationsbereich einen Ressourcentyp aus (z. B. Auftragswarteschlangen).
4. Wählen Sie eine bestimmte Ressource und dann Tags bearbeiten aus.
5. Fügen Sie Ihre Tags nach Bedarf hinzu oder löschen Sie sie.

    - So fügen Sie ein Tag hinzu – geben Sie den Schlüssel und den Wert in den leeren Textfeldern am Ende der Liste an.

- Um ein Tag zu löschen – wählen Sie die

  Delete icon
  Schaltfläche neben dem Tag.

6. Wiederholen Sie diesen Vorgang für jedes Tag, das Sie hinzufügen oder löschen möchten, und wählen Sie dann Tags bearbeiten aus, um den Vorgang abzuschließen.

Arbeiten mit Tags mittels CLI oder API

Verwenden Sie die folgenden AWS CLI-Befehle oder AWS Batch-API-Operationen, um die Tags für Ihre Ressourcen hinzuzufügen, zu aktualisieren, aufzulisten und zu löschen.

Markierungsunterstützung für AWS Batch-Ressourcen

Aufgabe	API-Aktion	AWS CLI	AWS Tools for Windows PowerShell
Fügen Sie einen oder mehrere Tags hinzu oder überschreiben Sie sie.	TagResource	tag-resource	Add-BATResourceTag
Löschen Sie ein oder mehrere Tags.	UntagResource	untag-resource	Remove-BATResourceTag
Listet Tags für eine Ressource auf	ListTagsForResource	list-tags-for-resource	Get-BATResourceTag

Die folgenden Beispiele zeigen, wie man Tags an Ressourcen mithilfe der AWS CLI hinzufügt oder entfernt.

Beispiel 1: Markieren einer vorhandenen Ressource

Der folgende Befehl markiert eine vorhandene Ressource.

```
aws batch tag-resource --resource-arn resource_ARN --tags team=devs
```

Beispiel 2: Entfernen der Markierung einer vorhandenen Ressource

Der folgende Befehl löscht ein Tag aus einer vorhandenen Ressource.

```
aws batch untag-resource --resource-arn resource_ARN --tag-keys tag_key
```

Beispiel 3: Tags für eine Ressource auflisten

Der folgende Befehl listet die Tags auf, die einer vorhandenen Ressource zugeordnet sind.

```
aws batch list-tags-for-resource --resource-arn resource_ARN
```

Mit einigen Aktionen zur Ressourcenerstellung können Sie Tags beim Erstellen der Ressource angeben. Die folgenden Aktionen unterstützen das Markieren bei der Erstellung.

Aufgabe	API-Aktion	AWS CLI	AWS Tools for Windows PowerShell
Erstellen einer Datenverarbeitungsumgebung	CreateComputeEnvironment	create-compute-environment	Neu-BATComputeEnvironment
Erstellen einer Auftragswarteschlange	CreateJobQueue	create-job-queue	Neu-BATJobQueue
Erstellen einer Planungsrichtlinie	CreateSchedulingPolicy	create-scheduling-policy	Neu-BATSchedulingPolicy
Registrieren einer Auftragsdefinition	RegisterJobDefinition	register-job-definition	Registrieren-BATJobDefinition
Übermitteln eines Auftrags	SubmitJob	Submit-Job	Submit-BATJob

AWS Batch-Servicekontingente

Die folgende Tabelle enthält die Service Quotas für AWS Batch, die nicht geändert werden können. Jedes Kontingent ist regionsspezifisch.

Ressource	Kontingent
Maximale Anzahl von Auftragswarteschlangen. Weitere Informationen finden Sie unter Warteschlangen für Job.	50
Maximale Anzahl von Datenverarbeitungsumgebungen in Amazon ECS und Amazon EKS. Weitere Informationen finden Sie unter Datenverarbeitungsumgebung.	50
Maximale Anzahl von Datenverarbeitungsumgebungen pro Amazon-EKS-Cluster.	5
Maximale Anzahl von Datenverarbeitungsumgebungen für jede Auftragswarteschlange	3
Maximale Anzahl von Aufgabenabhängigkeiten für eine Aufgabe	20
Maximale Größe der Auftragsdefinition (für `RegisterJobDefinition` API-Operationen)	24 KiB
Maximale Größe der Auftragsnutzlast (für `SubmitJob` API-Operationen)	30 KiB
Maximale Array-Größe für Array-Aufträge	10000
Maximale Anzahl von Aufträgen im Status `SUBMITTED`	1000000
Maximale Anzahl der Transaktionen pro Sekunde (TPS) für jedes Konto für `SubmitJob`-Operationen	50

Je nachdem, wie Sie verwendenAWS Batch, können zusätzliche Kontingente gelten. Weitere Informationen zu Amazon EC2-Kontingenten finden Sie unter Amazon EC2-Service Quotas im Allgemeine AWS-Referenz. Weitere Informationen zu Amazon-ECS-Kontingenten finden Sie unter Amazon-ECS-Service Quotas im Allgemeine AWS-Referenz. Weitere Informationen zu Amazon-EKS-Kontingenten finden Sie unter Amazon-EKS-Service Quotas im Allgemeine AWS-Referenz.

Problembehebung AWS Batch

Möglicherweise müssen Sie Probleme beheben, die mit Ihren Rechenumgebungen, Job-Warteschlangen, Jobdefinitionen oder Jobs zusammenhängen. In diesem Kapitel wird beschrieben, wie Sie solche Probleme in Ihrer AWS Batch Umgebung beheben und lösen können.

AWS Batch verwendet IAM-Richtlinien, -Rollen und -Berechtigungen und läuft auf der Infrastruktur von Amazon EC2, Amazon ECS und Amazon Elastic Kubernetes Service. AWS Fargate Informationen zur Behebung von Problemen im Zusammenhang mit diesen Diensten finden Sie im Folgenden:

- Problembehandlung bei IAM im IAM-Benutzerhandbuch
- Amazon ECS-Fehlerbehebung im Amazon Elastic Container Service Developer Guide
- Amazon EKS-Fehlerbehebung im Amazon EKS-Benutzerhandbuch
- Fehlerbehebung bei EC2-Instances im Amazon EC2 EC2-Benutzerhandbuch

Inhalt

    - Nicht unterstützte Version Kubernetes
    - Das Instanzprofil ist nicht vorhanden
    - Ungültiger Kubernetes Namespace
    - Computerumgebung wurde gelöscht
    - Knoten treten dem Amazon EKS-Cluster nicht bei
- AWS Batch bei Amazon EKS bleibt der Job im RUNNABLE Status hängen
- Stellen Sie sicher, dass der aws-auth ConfigMap richtig konfiguriert ist
- RBAC-Berechtigungen oder -Bindungen sind nicht richtig konfiguriert

AWS Batch

INVALIDDatenverarbeitungsumgebung

Es ist möglich, dass Sie eine verwaltete Rechenumgebung falsch konfiguriert haben. Wenn Sie das getan haben, wechselt die Computerumgebung in einen `INVALID` Status und kann keine Stellen zur Vermittlung annehmen. In den folgenden Abschnitten werden die möglichen Ursachen und die auf der Ursache beruhende Problembehebung beschrieben.

Falscher Rollenname oder ARN

Der häufigste Grund dafür, dass eine Rechenumgebung in einen `INVALID` Status wechselt, ist, dass die AWS Batch Service-Rolle oder die Amazon EC2-Spot-Flottenrolle einen falschen Namen oder Amazon Resource Name (ARN) hat. Dies ist häufiger bei Computerumgebungen der Fall, die mit den AWS CLI oder den AWS SDKs erstellt wurden. Wenn Sie in der eine Rechenumgebung erstellen AWS Management Console, AWS Batch hilft Ihnen das bei der Auswahl der richtigen Service- oder Spot-Flottenrollen. Nehmen wir jedoch an, Sie geben den Namen oder den ARN manuell ein und geben sie falsch ein. Dann ist es auch die resultierende Rechenumgebung`INVALID`.

Nehmen wir jedoch an, dass Sie den Namen oder den ARN für eine IAM-Ressource manuell in einen AWS CLI Befehl oder Ihren SDK-Code eingeben. In diesem Fall AWS Batch kann die Zeichenfolge nicht validiert werden. Stattdessen AWS Batch müssen Sie den falschen Wert akzeptieren und versuchen, die Umgebung zu erstellen. Wenn AWS Batch die Umgebung nicht erstellt werden kann, wechselt die Umgebung in einen `INVALID` Status, und es werden die folgenden Fehler angezeigt.

Bei einer ungültigen Servicerolle:

```
CLIENT_ERROR - Not authorized to perform sts:AssumeRole (Service:
AWSSecurityTokenService; Status Code: 403; Error Code: AccessDenied;
Request ID: dc0e2d28-2e99-11e7-b372-7fcc6fb65fe7)
```

Bei einer ungültigen Spot-Flottenrolle:

```
CLIENT_ERROR - Parameter: SpotFleetRequestConfig.IamFleetRole
is invalid. (Service: AmazonEC2; Status Code: 400; Error Code:
InvalidSpotFleetRequestConfig; Request ID: 331205f0-5ae3-4cea-
bac4-897769639f8d) Parameter: SpotFleetRequestConfig.IamFleetRole is
invalid
```

Eine häufige Ursache für dieses Problem ist das folgende Szenario. Sie geben nur den Namen einer IAM-Rolle an, wenn Sie die AWS CLI oder die AWS SDKs verwenden, anstelle des vollständigen Amazon-Ressourcennamens (ARN). Je nachdem, wie Sie die Rolle erstellt haben, kann der ARN ein `aws-service-role` Pfadpräfix enthalten. Wenn Sie die AWS Batch Servicerolle beispielsweise mithilfe der Verfahren unter manuell erstellenVerwenden von serviceverknüpften Rollen für AWS Batch, könnte Ihr ARN für die Servicerolle wie folgt aussehen.

```
arn:aws:iam::123456789012:role/AWSBatchServiceRole
```

Wenn Sie die Servicerolle heute jedoch als Teil des Assistenten für die erste Ausführung der Konsole erstellt haben, könnte Ihr ARN für die Servicerolle wie folgt aussehen.

```
arn:aws:iam::123456789012:role/aws-service-role/AWSBatchServiceRole
```

Dieses Problem kann auch auftreten, wenn Sie die AWS Batch Service-Level-Richtlinie (`AWSBatchServiceRole`) an eine Rolle anhängen, die nichts mit dem Dienst zu tun hat. In diesem Szenario erhalten Sie beispielsweise möglicherweise eine Fehlermeldung, die der folgenden ähnelt:

```
CLIENT_ERROR - User: arn:aws:sts::account_number:assumed-role/batch-replacement-role/
aws-batch is not
     authorized to perform: action on resource ...
```

Gehen Sie wie folgt vor, um dieses Problem zu beheben.

- Verwenden Sie eine leere Zeichenfolge für die Servicerolle, wenn Sie die AWS Batch Rechenumgebung erstellen.

- Geben Sie die Servicerolle im folgenden Format an:`arn:aws:iam::account_number:role/aws-service-role/batch.amazonaws.com/AWSServiceRoleForBatch`.

Wenn Sie nur den Namen einer IAM-Rolle angeben, wenn Sie die SDKs AWS CLI oder die AWS SDKs verwenden, AWS Batch wird davon ausgegangen, dass Ihr ARN das `aws-service-role` Pfadpräfix nicht verwendet. Aus diesem Grund empfehlen wir, dass Sie bei der Erstellung von Rechenumgebungen den vollständigen ARN für Ihre IAM-Rollen angeben.

Informationen zum Reparieren einer Computerumgebung, die auf diese Weise falsch konfiguriert ist, finden Sie unter. Reparieren einer Rechenumgebung `INVALID`

Reparieren einer Rechenumgebung **`INVALID`**

Wenn sich eine Rechenumgebung in einem `INVALID` Zustand befindet, aktualisieren Sie sie, um den ungültigen Parameter zu reparieren. Aktualisieren Sie in einem Falscher Rollenname oder ARN Fall die Rechenumgebung mit der richtigen Servicerolle.

So Reparieren Sie eine falsch konfigurierte Datenverarbeitungsumgebung

1. Öffnen Sie die AWS Batch Konsole unter https://console.aws.amazon.com/batch/.
2. Wählen Sie in der Navigationsleiste die aus, die Sie verwenden AWS-Region möchten.
3. Wählen Sie im Navigationsbereich Datenverarbeitungs-Umgebungen aus.
4. Wählen Sie auf der Seite Datenverarbeitungs-Umgebungen das Optionsfeld neben der zu bearbeitenden Datenverarbeitungsumgebung und dann Bearbeiten aus.
5. Wählen Sie auf der Seite Rechenumgebung aktualisieren unter Servicerolle die IAM-Rolle aus, die Sie mit Ihrer Rechenumgebung verwenden möchten. Die AWS Batch -Konsole zeigt nur mit der richtigen Vertrauensstellung für Datenverarbeitungsumgebungen an.
6. Wählen Sie Speichern aus, um Ihre Datenverarbeitungsumgebung zu aktualisieren.

Jobs stecken in einem **`RUNNABLE`** Status fest

Angenommen, Ihre Datenverarbeitungsumgebung enthält Rechenressourcen, aber Ihre Jobs werden nicht über diesen `RUNNABLE` Status hinaus bearbeitet. Dann ist es wahrscheinlich, dass etwas verhindert, dass die Jobs auf einer Rechenressource platziert werden, und dass Ihre Jobwarteschlangen blockiert werden. Hier erfahren Sie, ob Ihr Job darauf wartet, an der Reihe zu sein, oder ob er feststeckt und die Warteschlange blockiert.

Wenn AWS Batch festgestellt wird, dass Sie einen RUNNABLE Job an der Spitze haben und die Warteschlange blockiert haben, erhalten Sie von Amazon CloudWatch Events ein Ereignis mit dem Grund für die Warteschlange blockierter Jobs. Derselbe Grund wird auch im Rahmen von ListJobs DescribeJobs API-Aufrufen in das statusReason Feld aktualisiert.

Optional können Sie den jobStateTimeLimitActions Parameter über CreateJobQueue und UpdateJobQueueAPI-Aktionen konfigurieren.

> **Note**
>
> Derzeit besteht die einzige Aktion, die Sie mit verwenden können, jobStateLimitActions.action darin, einen Job abzubrechen.

Der jobStateTimeLimitActions Parameter wird verwendet, um eine Reihe von Aktionen anzugeben, die AWS Batch für Jobs in einem bestimmten Status ausgeführt werden. Sie können über das maxTimeSeconds Feld einen Zeitschwellenwert in Sekunden festlegen.

Wenn sich ein Job in einem RUNNABLE Zustand mit dem definierten Status befunden hat statusReason AWS Batch , wird die angegebene Aktion nach Ablauf maxTimeSeconds ausgeführt.

Sie können den jobStateTimeLimitActions Parameter beispielsweise so einstellen, dass bis zu 4 Stunden auf jeden Job in diesem RUNNABLE Status gewartet wird, der darauf wartet, dass genügend Kapazität verfügbar wird. Sie können dies tun, indem Sie statusReason den CAPACITY:INSUFFICIENT_INSTANCE_CAPACITY Wert maxTimeSeconds auf 144000 einstellen, bevor Sie den Job abbrechen und dem nächsten Job erlauben, an die Spitze der Job-Warteschlange zu gelangen.

Im Folgenden werden die Gründe aufgeführt, AWS Batch aus denen erkannt wird, dass eine Auftragswarteschlange blockiert ist. Diese Liste enthält die Nachrichten, die von den Aktionen ListJobs und der DescribeJobs API zurückgegeben wurden. Dies sind auch dieselben Werte, die Sie für den jobStateLimitActions.statusReason Parameter definieren können.

1. Grund: In allen verbundenen Rechenumgebungen treten Fehler aufgrund unzureichender Kapazitäten auf. AWS Batch Erkennt auf Anfrage Amazon EC2 EC2-Instances, bei denen Fehler mit unzureichender Kapazität auftreten. Wenn Sie den Auftrag entweder manuell oder durch Setzen des jobStateTimeLimitActions Parameters auf „abbrechen“statusReason, kann der nachfolgende Job an die Spitze der Warteschlange verschoben werden.

- **`statusReason`**Nachricht, während der Job feststeckt: `CAPACITY:INSUFFICIENT_INSTANCE_CAPACITY - Service cannot fulfill the capacity requested for instance type [instanceTypeName]`
- **`reason`**verwendet für**`jobStateTimeLimitActions`**: `CAPACITY:INSUFFICIENT_INSTANCE_CAPACITY`
- **`statusReason`**Nachricht, nachdem der Job abgebrochen wurde: `Canceled by JobStateTimeLimit action due to reason: CAPACITY:INSUFFICIENT_INSTANCE_CAPACITY`

Hinweis:

a. Die AWS Batch Servicerolle benötigt eine `autoscaling:DescribeScalingActivities` entsprechende Genehmigung, damit diese Erkennung funktioniert. Wenn Sie die `AWSServiceRoleForBatch`serviceverknüpfte Rolle (SLR) oder die `AWSBatchServiceRolePolicy`verwaltete Richtlinie verwenden, müssen Sie keine Maßnahmen ergreifen, da die entsprechenden Berechtigungsrichtlinien aktualisiert wurden.

b. Wenn Sie die SLR- oder die verwaltete Richtlinie verwenden, müssen Sie die `ec2:DescribeSpotFleetRequestHistory` Berechtigungen `autoscaling:DescribeScalingActivities` und hinzufügen, damit Sie blockierte Jobwarteschlangenereignisse und den aktualisierten Auftragsstatus erhalten können, wenn Sie angemeldet sind. `RUNNABLE` Darüber hinaus sind diese AWS Batch Berechtigungen erforderlich, um `cancellation` Aktionen über den `jobStateTimeLimitActions` Parameter ausführen zu können, auch wenn sie in der Auftragswarteschlange konfiguriert sind.

c. Wenn bei einem MNP-Job (Multi-Node Parallel) Fehler in der angehängten Amazon EC2 EC2-Rechenumgebung mit hoher Priorität auftreten, wird die Warteschlange blockiert, selbst wenn dieser `insufficient capacity` Fehler in einer Rechenumgebung mit niedrigerer Priorität auftritt.

2. Grund: Alle Computerumgebungen haben einen `maxvCpus`Parameter, der kleiner ist als die Jobanforderungen. Wenn Sie den Job entweder manuell oder durch Aktivieren des `jobStateTimeLimitActions` Parameters abbrechen`statusReason`, kann der nachfolgende Job an die Spitze der Warteschlange verschoben werden. Optional können Sie den `maxvCpus` Parameter der primären Rechenumgebung erhöhen, um den Anforderungen des blockierten Auftrags gerecht zu werden.

   - **`statusReason`**Nachricht, während der Job feststeckt: `MISCONFIGURATION:COMPUTE_ENVIRONMENT_MAX_RESOURCE - CE(s) associated with the job queue cannot meet the CPU requirement of the job.`

- **`reason`**verwendet für**`jobStateTimeLimitActions`**: `MISCONFIGURATION:COMPUTE_ENVIRONMENT_MAX_RESOURCE`
- **`statusReason`**Nachricht, nachdem der Job abgebrochen wurde: `Canceled by JobStateTimeLimit action due to reason: MISCONFIGURATION:COMPUTE_ENVIRONMENT_MAX_RESOURCE`

3. Grund: Keine der Rechenumgebungen verfügt über Instanzen, die die Jobanforderungen erfüllen. Wenn ein Job Ressourcen anfordert, wird AWS Batch erkannt, dass keine angeschlossene Rechenumgebung in der Lage ist, den eingehenden Job zu verarbeiten. Wenn Sie den Job entweder manuell oder durch Aktivieren des `jobStateTimeLimitActions` Parameters abbrechen`statusReason`, kann der nachfolgende Job an die Spitze der Warteschlange verschoben werden. Optional können Sie die zulässigen Instanztypen der Rechenumgebung neu definieren, um die erforderlichen Jobressourcen hinzuzufügen.
   - **`statusReason`**Nachricht, während der Job feststeckt: `MISCONFIGURATION:JOB_RESOURCE_REQUIREMENT - The job resource requirement (vCPU/memory/GPU) is higher than that can be met by the CE(s) attached to the job queue.`
   - **`reason`**verwendet für**`jobStateTimeLimitActions`**: `MISCONFIGURATION:JOB_RESOURCE_REQUIREMENT`
   - **`statusReason`**Nachricht, nachdem der Job abgebrochen wurde: `Canceled by JobStateTimeLimit action due to reason: MISCONFIGURATION:JOB_RESOURCE_REQUIREMENT`
4. Grund: In allen Computerumgebungen gibt es Probleme mit Servicerollen. Um dieses Problem zu beheben, vergleichen Sie Ihre Servicerollenberechtigungen mit den Berechtigungen für AWS Batch verwaltete Servicerollen und schließen Sie etwaige Lücken.

   Es hat sich bewährt, die AWS Batch Spiegelreflexkamera für Computerumgebungen zu verwenden, um ähnliche Fehler zu vermeiden.

   Wenn Sie den Job entweder manuell oder durch Aktivieren des `jobStateTimeLimitActions` Parameters abbrechen`statusReason`, kann der nachfolgende Job an die Spitze der Warteschlange verschoben werden. Ohne die Probleme mit der Servicerolle zu lösen, ist es wahrscheinlich, dass auch der nächste Auftrag blockiert wird. Es empfiehlt sich, dieses Problem manuell zu untersuchen und zu lösen.

- **`statusReason`**Nachricht, während der Job feststeckt: `MISCONFIGURATION:SERVICE_ROLE_PERMISSIONS – Batch service role has a permission issue.`
- **`reason`**verwendet für**`jobStateTimeLimitActions`**: `MISCONFIGURATION:SERVICE_ROLE_PERMISSIONS`
- **`statusReason`**Nachricht, nachdem der Job abgebrochen wurde: `Canceled by JobStateTimeLimit action due to reason: MISCONFIGURATION:SERVICE_ROLE_PERMISSIONS`

5. Grund: Alle Rechenumgebungen sind ungültig. Weitere Informationen finden Sie unter `INVALID`Rechenumgebung. Hinweis: Sie können über den `jobStateTimeLimitActions` Parameter keine programmierbare Aktion konfigurieren, um diesen Fehler zu beheben.
   - **`statusReason`**Meldung, während der Job feststeckt: `ACTION_REQUIRED - CE(s) associated with the job queue are invalid.`
6. Grund: AWS Batch hat eine blockierte Warteschlange erkannt, kann aber den Grund nicht ermitteln. Hinweis: Sie können über den `jobStateTimeLimitActions` Parameter keine programmierbare Aktion konfigurieren, um diesen Fehler zu beheben. Weitere Informationen zur Fehlerbehebung finden Sie unter Warum ist mein AWS Batch Job in RUNNABLE hängen AWS geblieben in re:POST.
   - **`statusReason`**Nachricht, während der Job feststeckt: `UNDETERMINED - Batch job is blocked, root cause is undetermined.`

Falls Sie kein Ereignis von CloudWatch Events erhalten haben oder ein Ereignis mit unbekanntem Grund erhalten haben, finden Sie hier einige der häufigsten Ursachen für dieses Problem.

Der **`awslogs`** Protokolltreiber ist auf Ihren Rechenressourcen nicht konfiguriert

AWS Batch Jobs senden ihre Protokollinformationen an CloudWatch Logs. Um die entsprechende Funktion zu aktivieren, müssen Sie Ihre Datenverarbeitungsressourcen so konfigurieren, dass sie den `awslogs`-Protokolltreiber verwenden. Nehmen wir an, dass Sie Ihr Datenressourcen-AMI auf dem für Amazon ECS optimierten AMI (oder Amazon Linux) aufbauen. Dann ist dieser Treiber standardmäßig im `ecs-init` Paket registriert. Nehmen wir nun an, Sie verwenden ein anderes Basis-AMI. Anschließend müssen Sie überprüfen, ob der `awslogs` Protokolltreiber als verfügbarer Protokolltreiber mit der `ECS_AVAILABLE_LOGGING_DRIVERS` Umgebungsvariablen angegeben ist, wenn der Amazon ECS-Container-Agent gestartet wird. Weitere Informationen

finden Sie unter AMI-Spezifikation für Rechenressourcen und Erstellen eines Compute-Ressourcen-AMI.

Unzureichende Ressourcen

Wenn in Ihren Jobdefinitionen mehr CPU- oder Speicherressourcen angegeben sind, als Ihre Rechenressourcen zuweisen können, werden Ihre Jobs niemals vergeben. Nehmen wir zum Beispiel an, dass Ihr Job 4 GiB Arbeitsspeicher spezifiziert und für Ihre Rechenressourcen weniger als der verfügbare Speicherplatz zur Verfügung steht. Dann ist es so, dass der Job nicht auf diesen Rechenressourcen platziert werden kann. In diesem Fall müssen Sie den angegebenen Speicher in Ihrer Auftragsdefinition reduzieren oder Ihrer Umgebung größere Datenverarbeitungsressourcen hinzufügen. Ein Teil des Speichers ist für den Amazon ECS-Container-Agenten und andere kritische Systemprozesse reserviert. Weitere Informationen finden Sie unter DatenverarbeitungsressourceSpeicherverwaltung.

Kein Internetzugang für Rechenressourcen

Datenverarbeitungsressourcen benötigen einen externen Netzwerkzugriff, um mit dem Amazon-ECS-Service-Endpunkt zu kommunizieren. Dies kann über einen Schnittstellen-VPC-Endpunkt oder über Ihre Datenverarbeitungsressourcen mit öffentlichen IP-Adressen sein.

Weitere Informationen zu Interface-VPC-Endpunkten finden Sie unter Amazon-ECS-Schnittstellen-VPC-Endpunkte (AWS PrivateLink) im Amazon Elastic Container Service-Entwicklerhandbuch.

Wenn Sie keinen Schnittstellen-VPC-Endpunkt konfiguriert haben und Ihre Datenverarbeitungsressourcen keine öffentlichen IP-Adressen haben, müssen Sie Netzwerkadressübersetzung (Network Address Translation, NAT) verwenden, um diesen Zugriff zur Verfügung zu stellen. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch. Weitere Informationen finden Sie unter the section called “Erstellen einer VPC”.

Amazon EC2 EC2-Instance-Limit erreicht

Die Anzahl der Amazon EC2 EC2-Instances, in denen Ihr Konto starten kann, AWS-Region wird durch Ihr EC2-Instance-Kontingent bestimmt. Bestimmte Instance-Typen haben auch ein per-instance-type Kontingent. Weitere Informationen zum Amazon EC2-Instance-Kontingent Ihres Kontos, einschließlich der Beantragung einer Limiterhöhung, finden Sie unter Amazon EC2 Service Limits im Amazon EC2 EC2-Benutzerhandbuch.

Der Amazon ECS-Container-Agent ist nicht installiert

Der Amazon ECS-Container-Agent muss auf dem Amazon Machine Image (AMI) installiert sein, damit Jobs AWS Batch ausgeführt werden können. Der Amazon ECS-Container-Agent ist standardmäßig auf Amazon ECS-optimierten AMIs installiert. Weitere Informationen zum Amazon ECS-Container-Agenten finden Sie unter Amazon ECS-Container-Agent im Amazon Elastic Container Service Developer Guide.

Weitere Informationen finden Sie unter Warum bleibt mein AWS Batch Job im RUNNABLE Status hängen? in re:POST.

Spot-Instances wurden bei der Erstellung nicht markiert

Spot-Instance-Tagging für AWS Batch Rechenressourcen wird ab dem 25. Oktober 2017 unterstützt. Zuvor enthielt die empfohlene IAM-verwaltete Richtlinie (`AmazonEC2SpotFleetRole`) für die Amazon EC2-Spot-Flottenrolle keine Berechtigungen zum Taggen von Spot-Instances beim Start. Die neue empfohlene IAM-verwaltete Richtlinie heißt. `AmazonEC2SpotFleetTaggingRole` Sie unterstützt das Taggen von Spot-Instances beim Start.

Um das Spot-Instance-Tagging bei der Erstellung zu korrigieren, gehen Sie wie folgt vor, um die aktuell empfohlene IAM-verwaltete Richtlinie auf Ihre Amazon EC2-Spot-Flotte-Rolle anzuwenden. Auf diese Weise sind alle future Spot-Instances, die mit dieser Rolle erstellt werden, berechtigt, Instance-Tags anzuwenden, wenn sie erstellt werden.

So wenden Sie die aktuelle IAM-verwaltete Richtlinie auf Ihre Amazon EC2 Spot Fleet-Rolle an

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Rollen und wählen Sie Ihre Amazon EC2 Spot Fleet-Rolle aus.
3. Wählen Sie Richtlinie anfügen aus.
4. Wählen Sie AmazonEC2 SpotFleet TaggingRole und dann Attach policy aus.
5. Wählen Sie erneut Ihre Amazon EC2 Spot Fleet-Rolle aus, um die vorherige Richtlinie zu entfernen.
6. Wählen Sie das X rechts neben der SpotFleetAmazonEC2-Rollenrichtlinie und wählen Sie Trennen aus.

Spot-Instances werden nicht herunterskaliert

AWS Batch hat am 10. März 2021 die AWSServiceRoleForBatchserviceverknüpfte Rolle eingeführt. Wenn im `serviceRole` Parameter der Rechenumgebung keine Rolle angegeben ist, wird diese dienstverknüpfte Rolle als Servicerolle verwendet. Nehmen wir jedoch an, dass die serviceverknüpfte Rolle in einer EC2-Spot-Computing-Umgebung verwendet wird, die verwendete Spot-Rolle jedoch nicht die von SpotFleetTaggingRoleAmazonEC2 verwaltete Richtlinie beinhaltet. Dann wird die Spot-Instance nicht herunterskaliert. Infolgedessen erhalten Sie eine Fehlermeldung mit der folgenden Meldung: „Sie sind nicht berechtigt, diesen Vorgang auszuführen.“ Gehen Sie wie folgt vor, um die Spot-Flottenrolle zu aktualisieren, die Sie im `spotIamFleetRole` Parameter verwenden. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen und Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS Service im IAM-Benutzerhandbuch.

Themen

- Fügen Sie Ihrer Spot-Flotte-Rolle eine von AmazonEC2 SpotFleet TaggingRole verwaltete Richtlinie hinzu in AWS Management Console
- Fügen Sie Ihrer Spot-Flotte-Rolle die von AmazonEC2 SpotFleet TaggingRole verwaltete Richtlinie hinzu, indem Sie AWS CLI

Fügen Sie Ihrer Spot-Flotte-Rolle eine von AmazonEC2 SpotFleet TaggingRole verwaltete Richtlinie hinzu in AWS Management Console

So wenden Sie die aktuelle IAM-verwaltete Richtlinie auf Ihre Amazon EC2 Spot Fleet-Rolle an

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Rollen und wählen Sie Ihre Amazon EC2 Spot Fleet-Rolle aus.
3. Wählen Sie Richtlinie anfügen aus.
4. Wählen Sie AmazonEC2 SpotFleet TaggingRole und dann Attach policy aus.
5. Wählen Sie erneut Ihre Amazon EC2 Spot Fleet-Rolle aus, um die vorherige Richtlinie zu entfernen.
6. Wählen Sie das X rechts neben der SpotFleetAmazonEC2-Rollenrichtlinie und wählen Sie Trennen aus.

Fügen Sie Ihrer Spot-Flotte-Rolle die von AmazonEC2 SpotFleet TaggingRole verwaltete Richtlinie hinzu, indem Sie AWS CLI

Bei den Beispielbefehlen wird davon ausgegangen, dass Ihre Amazon EC2 Spot Fleet-Rolle den Namen *SpotFleetAmazonEC2* Role trägt. Wenn Ihre Rolle einen anderen Namen verwendet, passen Sie die Befehle entsprechend an.

Um die von AmazonEC2 SpotFleet TaggingRole verwaltete Richtlinie Ihrer Spot-Flotte-Rolle zuzuordnen

1. Um die von AmazonEC2 SpotFleet TaggingRole verwaltete IAM-Richtlinie Ihrer *SpotFleetAmazonEC2-Rollenrolle* zuzuordnen, führen Sie den folgenden Befehl mit dem aus. AWS CLI

   ```
   $ aws iam attach-role-policy \
       --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEC2SpotFleetTaggingRole
    \
       --role-name AmazonEC2SpotFleetRole
   ```

2. Um die von AmazonEC2 SpotFleet Role verwaltete IAM-Richtlinie von Ihrer ***SpotFleetAmazonEC2-Rollenrolle*** zu trennen, führen Sie den folgenden Befehl mit dem aus. AWS CLI

   ```
   $ aws iam detach-role-policy \
       --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEC2SpotFleetRole \
       --role-name AmazonEC2SpotFleetRole
   ```

Secrets Manager können nicht abgerufen werden

Wenn Sie ein AMI mit einem Amazon ECS-Agenten verwenden, der älter als Version 1.16.0-1 ist, müssen Sie die Amazon ECS-Agenten-Konfigurationsvariable verwenden, um diese Funktion nutzen `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` zu können. Sie können es der `./etc/ecs/ecs.config` Datei einer neuen Container-Instance hinzufügen, wenn Sie diese Instance erstellen. Oder Sie können es zu einer vorhandenen Instanz hinzufügen. Wenn Sie es zu einer vorhandenen Instanz hinzufügen, müssen Sie den ECS-Agenten neu starten, nachdem Sie ihn hinzugefügt haben. Weitere Informationen finden Sie unter Amazon ECS Container Agent Configuration im Entwicklerhandbuch zum Amazon Elastic Container Service.

Die Ressourcenanforderungen für die Jobdefinition können nicht außer Kraft gesetzt werden

Die Speicher- und vCPU-Überschreibungen, die in der Struktur `memory` und `vcpus` den Elementen der ContainerOverrides-Struktur angegeben sind, die an übergeben wurden, können die Speicher- und vCPU-Anforderungen SubmitJob, die in der ResourceRequirements-Struktur in der Auftragsdefinition angegeben sind, nicht überschreiben.

Wenn Sie versuchen, diese Ressourcenanforderungen zu überschreiben, wird möglicherweise die folgende Fehlermeldung angezeigt:

„Dieser Wert wurde in einem veralteten Schlüssel übermittelt und kann zu einem Konflikt mit dem Wert führen, der in den Ressourcenanforderungen der Jobdefinition angegeben ist.“

Um dies zu korrigieren, geben Sie die Speicher- und vCPU-Anforderungen im Mitglied ResourceRequirements der ContainerOverrides an. Zum Beispiel, wenn Ihre Speicher- und vCPU-Überschreibungen in den folgenden Zeilen angegeben sind.

```
"containerOverrides": {
   "memory": 8192,
   "vcpus": 4
}
```

Ändern Sie sie wie folgt:

```
"containerOverrides": {
   "resourceRequirements": [
      {
         "type": "MEMORY",
         "value": "8192"
      },
      {
         "type": "VCPU",
         "value": "4"
      }
   ],
}
```

Nehmen Sie dieselbe Änderung an den Speicher- und vCPU-Anforderungen vor, die im ContainerProperties-Objekt in der Auftragsdefinition angegeben sind. Zum Beispiel, wenn Ihre Speicher- und vCPU-Anforderungen in den folgenden Zeilen angegeben sind.

```
{
   "containerProperties": {
      "memory": 4096,
      "vcpus": 2,
}
```

Ändern Sie sie wie folgt:

```
"containerProperties": {
   "resourceRequirements": [
      {
         "type": "MEMORY",
         "value": "4096"
      },
      {
         "type": "VCPU",
         "value": "2"
      }
   ],
}
```

Fehlermeldung beim Aktualisieren der **`desiredvCpus`** Einstellung

Wenn Sie die AWS Batch API verwenden, um die gewünschte vCPUs (`desiredvCpus`) -Einstellung zu aktualisieren, wird die folgende Fehlermeldung angezeigt.

`Manually scaling down compute environment is not supported. Disconnecting job queues from compute environment will cause it to scale-down to minvCpus.`

Dieses Problem tritt auf, wenn der aktualisierte `desiredvCpus` Wert unter dem aktuellen `desiredvCpus` Wert liegt. Wenn Sie den `desiredvCpus` Wert aktualisieren, müssen beide der folgenden Bedingungen zutreffen:

- Der `desiredvCpus` Wert muss zwischen den `maxvCpus` Werten `minvCpus` und liegen.
- Der aktualisierte `desiredvCpus` Wert muss größer oder gleich dem aktuellen `desiredvCpus` Wert sein.

AWS Batch auf Amazon EKS

Themen

INVALIDComputerumgebung

Es ist möglich, dass Sie eine verwaltete Rechenumgebung falsch konfiguriert haben. Wenn Sie das getan haben, wechselt die Computerumgebung in einen `INVALID` Status und kann keine Stellen zur Vermittlung annehmen. In den folgenden Abschnitten werden die möglichen Ursachen und die auf der Ursache beruhende Problembehebung beschrieben.

Nicht unterstützte Version Kubernetes

Möglicherweise wird eine Fehlermeldung angezeigt, die der folgenden ähnelt, wenn Sie den `CreateComputeEnvironment` API-Vorgang oder `UpdateComputeEnvironment` den API-Vorgang verwenden, um eine Rechenumgebung zu erstellen oder zu aktualisieren. Dieses Problem tritt auf, wenn Sie in `EC2Configuration` eine Kubernetes Version angeben, die nicht unterstützt wird.

```
At least one imageKubernetesVersion in EC2Configuration is not supported.
```

Um dieses Problem zu beheben, löschen Sie die Rechenumgebung und erstellen Sie sie dann mit einer unterstützten Kubernetes Version neu.

Sie können ein kleineres Versions-Upgrade auf Ihrem Amazon EKS-Cluster durchführen. Sie können den Cluster beispielsweise von `1.xx` auf aktualisieren, `1.yy` auch wenn die Nebenversion nicht unterstützt wird.

Der Status der Rechenumgebung kann sich jedoch `INVALID` nach einem Update der Hauptversion auf ändern. Dies ist beispielsweise der Fall, wenn Sie ein Upgrade einer Hauptversion von `1.xx` auf durchführen`2.yy`. Wenn die Hauptversion von nicht unterstützt wird AWS Batch, wird eine Fehlermeldung angezeigt, die der folgenden ähnelt.

```
reason=CLIENT_ERROR - ... EKS Cluster version [2.yy] is unsupported
```

Um dieses Problem zu beheben, geben Sie eine unterstützte Kubernetes Version an, wenn Sie einen API-Vorgang verwenden, um eine Rechenumgebung zu erstellen oder zu aktualisieren.

AWS Batch auf Amazon unterstützt EKS derzeit die folgenden Kubernetes Versionen:

- `1.29`
- `1.28`
- `1.27`
- `1.26`
- `1.25`
- `1.24`
- `1.23`

Das Instanzprofil ist nicht vorhanden

Wenn das angegebene Instance-Profil nicht existiert, wird der Status der Datenverarbeitungsumgebung AWS Batch auf Amazon EKS auf geändert`INVALID`. Im `statusReason` Parameter wird ein Fehler angezeigt, der dem folgenden ähnelt.

```
CLIENT_ERROR - Instance profile arn:aws:iam::...:instance-profile/<name> does not exist
```

Um dieses Problem zu beheben, geben Sie ein funktionierendes Instanzprofil an oder erstellen Sie es. Weitere Informationen finden Sie unter IAM-Rolle für Amazon EKS-Knoten im Amazon EKS-Benutzerhandbuch.

Ungültiger Kubernetes Namespace

Wenn EKS AWS Batch auf Amazon den Namespace für die Rechenumgebung nicht validieren kann, wird der Status der Rechenumgebung auf `INVALID` geändert. Dieses Problem kann beispielsweise auftreten, wenn der Namespace nicht existiert.

Im `statusReason` Parameter wird eine Fehlermeldung angezeigt, die der folgenden ähnelt.

```
CLIENT_ERROR - Unable to validate Kubernetes Namespace
```

Dieses Problem kann auftreten, wenn eine der folgenden Bedingungen zutrifft:

- Die Kubernetes Namespace-Zeichenfolge im `CreateComputeEnvironment` Aufruf ist nicht vorhanden. Weitere Informationen finden Sie unter CreateComputeUmgebung.
- Die erforderlichen RBAC-Berechtigungen (Role-Based Access Control) zur Verwaltung des Namespaces sind nicht richtig konfiguriert.
- AWS Batch hat keinen Zugriff auf den Amazon Kubernetes EKS-API-Serverendpunkt.

Informationen zum Beheben dieses Problems finden Sie unter Stellen Sie sicher, dass der `aws-auth ConfigMap` richtig konfiguriert ist. Weitere Informationen finden Sie unter Erste Schritte mit AWS Batch in Amazon EKS.

Computerumgebung wurde gelöscht

Angenommen, Sie löschen einen Amazon EKS-Cluster, bevor Sie die angehängte AWS Batch Amazon EKS-Rechenumgebung löschen. Dann wird der Status der Rechenumgebung auf geändert`INVALID`. In diesem Szenario funktioniert die Rechenumgebung nicht richtig, wenn Sie den Amazon EKS-Cluster mit demselben Namen neu erstellen.

Um dieses Problem zu beheben, löschen Sie die Datenverarbeitungsumgebung AWS Batch auf Amazon EKS und erstellen Sie sie anschließend neu.

Knoten treten dem Amazon EKS-Cluster nicht bei

AWS Batch auf Amazon EKS skaliert eine Rechenumgebung herunter, wenn festgestellt wird, dass nicht alle Knoten dem Amazon EKS-Cluster beigetreten sind. Wenn AWS Batch Amazon EKS die Rechenumgebung herunterskaliert, wird der Status der Rechenumgebung auf geändert`INVALID`.

> **Note**
>
> AWS Batch ändert den Status der Rechenumgebung nicht sofort, sodass Sie das Problem debuggen können.

Im `statusReason` Parameter wird eine Fehlermeldung angezeigt, die einer der folgenden ähnelt:

```
Your compute environment has been INVALIDATED and scaled down because
none of the instances joined the underlying ECS Cluster. Common issues
preventing instances joining are the following: VPC/Subnet configuration
```

```
preventing communication to ECS, incorrect Instance Profile policy
preventing authorization to ECS, or customized AMI or LaunchTemplate
configurations affecting ECS agent.

Your compute environment has been INVALIDATED and scaled down because
none of the nodes joined the underlying Amazon EKS Cluster. Common issues
preventing nodes joining are the following: networking configuration
preventing communication to Amazon EKS Cluster, incorrect Amazon EKS
Instance Profile or Kubernetes RBAC policy preventing authorization
to Amazon EKS Cluster, customized AMI or LaunchTemplate configurations
affecting Amazon EKS/Kubernetes node bootstrap.
```

Wenn Sie ein standardmäßiges Amazon EKS-AMI verwenden, sind die häufigsten Ursachen für dieses Problem die folgenden:

- Die Instance-Rolle ist nicht richtig konfiguriert. Weitere Informationen finden Sie unter IAM-Rolle für Amazon EKS-Knoten im Amazon EKS-Benutzerhandbuch.
- Die Subnetze sind nicht richtig konfiguriert. Weitere Informationen finden Sie unter Amazon EKS-VPC- und Subnetzanforderungen und Überlegungen im Amazon EKS-Benutzerhandbuch.
- Die Sicherheitsgruppe ist nicht richtig konfiguriert. Weitere Informationen finden Sie unter Anforderungen und Überlegungen zu Amazon EKS-Sicherheitsgruppen im Amazon EKS-Benutzerhandbuch.

> **Note**
>
> Möglicherweise wird Ihnen auch eine Fehlermeldung im Personal Health Dashboard (PHD) angezeigt.

AWS Batch bei Amazon EKS bleibt der Job im **RUNNABLE** Status hängen

Ein `aws-auth ConfigMap` wird automatisch erstellt und auf Ihren Cluster angewendet, wenn Sie eine verwaltete Knotengruppe oder eine Knotengruppe mithilfe von`eksctl`. Eine `aws-auth ConfigMap` wird zunächst erstellt, damit Knoten Ihrem Cluster beitreten können. Sie verwenden den jedoch auch, `aws-auth ConfigMap` um Benutzern und Rollen den Zugriff auf rollenbasierte Zugriffssteuerung (RBAC) hinzuzufügen.

Gehen Sie wie folgt vor, um zu überprüfen, ob der richtig konfiguriert ist `aws-authConfigMap`:

1. Rufen Sie die zugewiesenen Rollen ab in: `aws-auth ConfigMap`

```
$ kubectl get configmap -n kube-system aws-auth -o yaml
```

2. Stellen Sie sicher, dass der wie folgt konfiguriert `roleARN` ist.

```
rolearn: arn:aws:iam::aws_account_number:role/AWSServiceRoleForBatch
```

> **Note**
>
> Sie können auch die Protokolle der Amazon EKS-Kontrollebene überprüfen. Weitere Informationen finden Sie unter Protokollierung der Amazon EKS-Kontrollebene im Amazon EKS-Benutzerhandbuch.

Um ein Problem zu lösen, bei dem ein Job in einem `RUNNABLE` Status hängengeblieben ist, empfehlen wir Ihnen, das Manifest erneut anzuwenden. `kubectl` Weitere Informationen finden Sie unter Schritt 1: Vorbereiten Ihres Amazon-EKS-Clusters für AWS Batch. Oder Sie können es verwenden, `kubectl` um das manuell zu bearbeiten. `aws-auth ConfigMap` Weitere Informationen finden Sie unter Aktivieren des IAM-Benutzer- und Rollenzugriffs auf Ihren Cluster im Amazon EKS-Benutzerhandbuch.

Stellen Sie sicher, dass der **`aws-auth ConfigMap`** richtig konfiguriert ist

Gehen Sie wie folgt vor, um zu überprüfen, ob der richtig konfiguriert `aws-auth ConfigMap` ist:

1. Rufen Sie die zugewiesenen Rollen in der `aws-auth ConfigMap` ab.

```
$ kubectl get configmap -n kube-system aws-auth -o yaml
```

2. Stellen Sie sicher, dass der wie folgt konfiguriert `roleARN` ist.

```
rolearn: arn:aws:iam::aws_account_number:role/AWSServiceRoleForBatch
```

> **Note**
>
> Der Pfad `aws-service-role/batch.amazonaws.com/` wurde aus dem ARN der serviceverknüpften Rolle entfernt. Dies liegt an einem Problem mit der `aws-auth`

Konfigurationsübersicht. Weitere Informationen finden Sie unter Rollen mit Pfaden funktionieren nicht, wenn der Pfad in ihrem ARN in der enthalten ist aws-authconfigmap.

> **Note**
>
> Sie können auch die Protokolle der Amazon EKS-Kontrollebene überprüfen. Weitere Informationen finden Sie unter Protokollierung der Amazon EKS-Kontrollebene im Amazon EKS-Benutzerhandbuch.

Um ein Problem zu lösen, bei dem ein Job in einem `RUNNABLE` Status hängengeblieben ist, empfehlen wir Ihnen, das Manifest erneut anzuwenden. `kubectl` Weitere Informationen finden Sie unter Schritt 1: Vorbereiten Ihres Amazon-EKS-Clusters für AWS Batch. Oder Sie können es verwenden, `kubectl` um das manuell zu bearbeiten. `aws-auth` `ConfigMap` Weitere Informationen finden Sie unter Aktivieren des IAM-Benutzer- und Rollenzugriffs auf Ihren Cluster im Amazon EKS-Benutzerhandbuch.

RBAC-Berechtigungen oder -Bindungen sind nicht richtig konfiguriert

Wenn Sie Probleme mit RBAC-Berechtigungen oder -Bindungen haben, überprüfen Sie, ob die `aws-batch` Kubernetes Rolle auf den Namespace zugreifen kann: Kubernetes

```
$ kubectl get namespace namespace --as=aws-batch
```

```
$ kubectl auth can-i get ns --as=aws-batch
```

Sie können den **`kubectl describe`** Befehl auch verwenden, um die Autorisierungen für eine Clusterrolle oder einen Cluster-Namespace anzuzeigen. Kubernetes

```
$ kubectl describe clusterrole aws-batch-cluster-role
```

Es folgt eine Beispielausgabe.

```
Name:         aws-batch-cluster-role
Labels:       <none>
Annotations:  <none>
```

```
PolicyRule:
  Resources                                       Non-Resource URLs  Resource Names
 Verbs
  ---------                                       -----------------  --------------
 -----
  configmaps                                      []                 []
 [get list watch]
  nodes                                           []                 []
 [get list watch]
  pods                                            []                 []
 [get list watch]
  daemonsets.apps                                 []                 []
 [get list watch]
  deployments.apps                                []                 []
 [get list watch]
  replicasets.apps                                []                 []
 [get list watch]
  statefulsets.apps                               []                 []
 [get list watch]
  clusterrolebindings.rbac.authorization.k8s.io  []                 []
 [get list]
  clusterroles.rbac.authorization.k8s.io         []                 []
 [get list]
  namespaces                                      []                 []
 [get]
```

```
$ kubectl describe role aws-batch-compute-environment-role -n my-aws-batch-namespace
```

Es folgt eine Beispielausgabe.

```
Name:         aws-batch-compute-environment-role
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources                                Non-Resource URLs  Resource Names  Verbs
  ---------                                -----------------  --------------  -----
  pods                                     []                 []              [create
 get list watch delete patch]
  serviceaccounts                          []                 []              [get list]
  rolebindings.rbac.authorization.k8s.io  []                 []              [get list]
  roles.rbac.authorization.k8s.io         []                 []              [get list]
```

Wenden Sie die RBAC-Berechtigungen und -Befehle erneut an, um dieses Problem zu beheben. `rolebinding` Weitere Informationen finden Sie unter Schritt 1: Vorbereiten Ihres Amazon-EKS-Clusters für AWS Batch.

Bewährte Methoden für AWS Batch

Sie können verwendenAWS Batch, um eine Vielzahl anspruchsvoller Rechenworkloads in großem Umfang auszuführen, ohne eine komplexe Architektur zu verwalten. -AWS BatchAufträge können in einer Vielzahl von Anwendungsfällen in Bereichen wie Bioologie, Spiele und Machine Learning verwendet werden.

In diesem Thema werden die bewährten Methoden behandelt, die Sie bei der Verwendung von berücksichtigen sollten, AWS Batch sowie Anleitungen zum Ausführen und Optimieren Ihrer Workloads bei der Verwendung von AWS Batch.

Themen

Wann sollte man verwenden? AWS Batch

AWS Batch führt Aufträge in großem Umfang und zu geringen Kosten aus und bietet Warteschlangendienste und kostenoptimierte Skalierung. Allerdings eignet sich nicht jeder Workload für die Ausführung mit AWS Batch.

- Kurze Aufträge – Wenn ein Auftrag nur wenige Sekunden lang ausgeführt wird, kann der Overhead für die Planung des Batch-Auftrags länger dauern als die Laufzeit des Auftrags selbst. Um dieses Problem zu umgehen, können Sie binpack Ihre Aufgaben gemeinsam erledigen, bevor Sie sie in einreichenAWS Batch. Konfigurieren Sie dann Ihre AWS Batch Aufträge so, dass sie die Aufgaben durchlaufen. Stufen Sie beispielsweise die einzelnen Aufgabenargumente in einer Amazon-DynamoDB-Tabelle oder als Datei in einem Amazon S3-Bucket ein. Erwägen Sie, Aufgaben zu gruppieren, damit die Aufträge jeweils 3–5 Minuten laufen. Nachdem Sie binpack die Aufträge abgeschlossen haben, durchlaufen Sie Ihre Aufgabengruppen innerhalb Ihres AWS Batch Auftrags.

- Aufträge, die sofort ausgeführt werden müssen – AWS Batch kann Aufträge schnell verarbeiten. AWS Batch ist jedoch ein Scheduler und optimiert die Kostenleistung, die Auftragspriorität und den Durchsatz. benötigt AWS Batch möglicherweise Zeit, um Ihre Anforderungen zu verarbeiten. Wenn Sie innerhalb weniger Sekunden eine Antwort benötigen, ist ein servicebasierter Ansatz mit Amazon ECS oder Amazon EKS besser geeignet.

Checkliste für skalierbare Ausführung

Bevor Sie einen großen Workload auf 50 000 oder mehr vCPUs ausführen, sollten Sie die folgende Checkliste in Betracht ziehen.

> **Note**
>
> Wenn Sie einen großen Workload auf einer Million oder mehr vCPUs ausführen möchten oder Hilfe bei der Ausführung großer Mengen benötigen, wenden Sie sich an Ihr AWS Team.

- Überprüfen Ihrer Amazon EC2-Kontingente – Überprüfen Sie Ihre Amazon EC2-Kontingente (auch als Limits bezeichnet) im Bereich Service Quotas der AWS Management Console. Beantragen Sie bei Bedarf eine Kontingenterhöhung für Ihre Spitzenanzahl von Amazon EC2. Denken Sie daran, dass Amazon EC2 Spot- und Amazon On-Demand-Instances separate Kontingente haben. Weitere Informationen finden Sie unter Erste Schritte mit Service Quotas.
- Überprüfen Ihres Amazon Elastic Block Store-Kontingents für jede Region – Jede Instance verwendet ein GP2- oder GP3-Volume für das Betriebssystem. Standardmäßig AWS-Region beträgt das Kontingent für jede 300 TiB . Jede Instance, die verwendet, zählt jedoch als Teil dieses Kontingents. Stellen Sie daher sicher, dass Sie dies berücksichtigen, wenn Sie Ihr Amazon Elastic Block Store-Kontingent für jede Region überprüfen. Wenn Ihr Kontingent erreicht ist, können Sie keine weiteren Instances erstellen. Weitere Informationen finden Sie unter Endpunkte und Kontingente von Amazon Elastic Block Store.
- Verwenden von Amazon S3 für die Speicherung – Amazon S3 bietet einen hohen Durchsatz und trägt dazu bei, Vermutungen darüber zu vermeiden, wie viel Speicher basierend auf der Anzahl der Aufträge und Instances in jeder Availability Zone bereitgestellt werden soll. Weitere Informationen finden Sie unter Bewährte Methoden für Designmuster: Optimieren der Amazon S3-Leistung.
- Skalieren Sie schrittweise, um Engpässe frühzeitig zu identifizieren – Beginnen Sie bei einem Auftrag, der auf einer Million oder mehr vCPUs ausgeführt wird, mit einer niedrigeren und allmählichen Zunahme, damit Sie Engpässe frühzeitig erkennen können. Beginnen Sie

beispielsweise mit der Ausführung von auf 50 000 vCPUs. Erhöhen Sie dann die Anzahl auf 200 000 vCPUs und dann auf 500 000 vCPUs usw. Mit anderen Worten, erhöhen Sie die vCPU-Anzahl weiter schrittweise, bis Sie die gewünschte Anzahl von vCPUs erreichen.

- Überwachen Sie , um potenzielle Probleme frühzeitig zu identifizieren – Um potenzielle Unterbrechungen und Probleme bei der Ausführung in großem Umfang zu vermeiden, achten Sie darauf, sowohl Ihre Anwendung als auch Ihre Architektur zu überwachen. Unterbrechungen können auch bei der Skalierung von 1 000 auf 5 000 vCPUs auftreten. Sie können Amazon CloudWatch Logs verwenden, um Protokolldaten zu überprüfen, oder CloudWatch Embedded Metrics mit einer Client-Bibliothek verwenden. Weitere Informationen finden Sie in CloudWatch der Referenz zum Protokoll-Agenten und `aws-embedded-metrics`

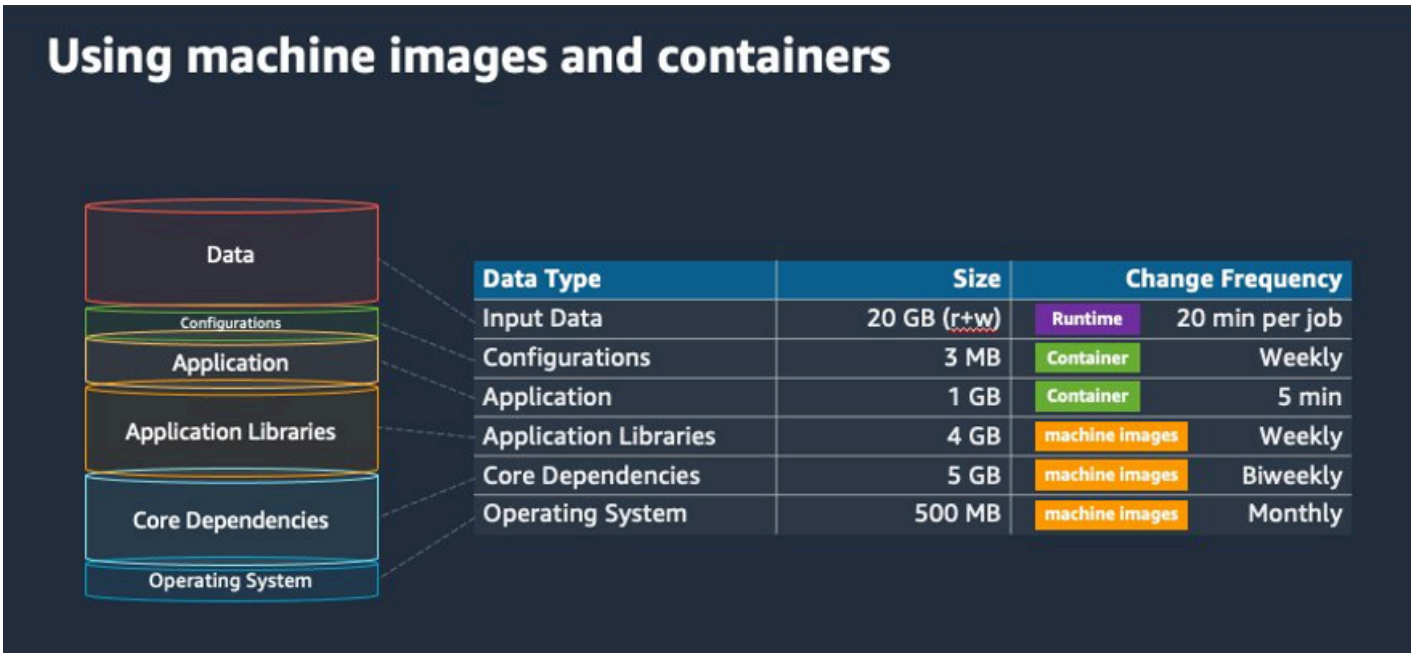
Optimieren von Containern und AMIs

Containergröße und -struktur sind für den ersten Satz von Aufträgen wichtig, die Sie ausführen. Dies gilt insbesondere, wenn der Container größer als 4 GB ist. Container-Images werden in Ebenen erstellt. Die Ebenen werden parallel von Docker mit drei gleichzeitigen Threads abgerufen. Sie können die Anzahl der gleichzeitigen Threads mit dem `-max-concurrent-downloads`Parameter erhöhen. Weitere Informationen finden Sie in der Dockerd-Dokumentation.

Obwohl Sie größere Container verwenden können, empfehlen wir Ihnen, die Containerstruktur und -größe für schnellere Startzeiten zu optimieren.

- Kleinere Container werden schneller abgerufen – Kleinere Container können zu schnelleren Anwendungsstartzeiten führen. Um die Containergröße zu verringern, entladen Sie Bibliotheken oder Dateien, die selten auf das Amazon Machine Image (AMI) aktualisiert werden. Sie können Bind-Mounts auch verwenden, um Zugriff auf Ihre Container zu gewähren. Weitere Informationen finden Sie unter Bind-Mounts.
- Erstellen von Ebenen mit gerader Größe und Aufteilen großer Ebenen – Jede Ebene wird von einem Thread abgerufen. Daher kann sich eine große Ebene erheblich auf Ihre Auftragsstartzeit auswirken. Wir empfehlen eine maximale Ebenengröße von 2 GB als guten Kompromiss zwischen einer größeren Containergröße und schnelleren Startzeiten. Sie können den `-docker history your_image_id`Befehl ausführen, um die Struktur Ihres Container-Images und die Ebenengröße zu überprüfen. Weitere Informationen finden Sie in der Docker-Dokumentation.
- Verwenden von Amazon Elastic Container Registry als Container-Repository – Wenn Sie Tausende von Aufträgen parallel ausführen, kann ein selbstverwaltetes Repository fehlschlagen

oder den Durchsatz drosseln. Amazon ECR funktioniert in großem Umfang und kann Workloads mit bis zu einer Million vCPUs verarbeiten.



Wählen Sie die richtige Ressource für die Datenverarbeitungsumgebung

AWS Fargate erfordert weniger Ersteinrichtung und -konfiguration als Amazon EC2 und ist wahrscheinlich einfacher zu verwenden, insbesondere wenn Sie es zum ersten Mal verwenden. Mit Fargate müssen Sie keine Server verwalten, Kapazitätsplanung durchführen oder Container-Workloads aus Sicherheitsgründen isolieren.

Wenn Sie die folgenden Anforderungen haben, empfehlen wir Ihnen, Fargate-Instances zu verwenden:

- Ihre Aufträge müssen schnell beginnen, insbesondere weniger als 30 Sekunden.
- Die Anforderungen Ihrer Aufträge liegen bei 16 vCPUs oder weniger, ohne GPUs und 120 GiB Arbeitsspeicher oder weniger.

Weitere Informationen finden Sie unter Wann Fargate verwendet werden sollte.

Wenn Sie die folgenden Anforderungen haben, empfehlen wir Ihnen, Amazon EC2 zu verwenden:

- Sie benötigen eine größere Kontrolle über die Instance-Auswahl oder erfordern die Verwendung bestimmter Instance-Typen.
- Ihre Aufträge erfordern Ressourcen, die nicht bereitstellen AWS Fargate kann, wie GPUs, mehr Arbeitsspeicher, ein benutzerdefiniertes AMI oder den Amazon Elastic Fabric Adapter.
- Sie benötigen einen hohen Durchsatz oder eine hohe Gleichzeitigkeit.
- Sie müssen Ihr AMI, Ihre Amazon EC2-Startvorlage oder den Zugriff auf spezielle Linux-Parameter anpassen.

Mit Amazon EC2 können Sie Ihren Workload besser an Ihre spezifischen Anforderungen anpassen und bei Bedarf skalierbar ausführen.

Amazon EC2 On-Demand oder Amazon EC2 Spot

Die meisten AWS Batch Kunden verwenden Amazon EC2-Spot-Instances aufgrund der Einsparungen gegenüber On-Demand-Instances. Wenn Ihr Workload jedoch mehrere Stunden lang ausgeführt wird und nicht unterbrochen werden kann, sind On-Demand-Instances möglicherweise besser für Sie geeignet. Sie können Spot-Instances immer zuerst ausprobieren und bei Bedarf zu On-Demand wechseln.

Wenn Sie die folgenden Anforderungen und Erwartungen haben, verwenden Sie Amazon EC2 On-Demand-Instances:

- Die Laufzeit Ihrer Aufträge beträgt mehr als eine Stunde und Sie können Unterbrechungen Ihres Workloads nicht tolerieren.
- Sie haben ein striktes SLO (Service-Level-Ziel) für Ihre gesamte Workload und können die Rechenzeit nicht verlängern.
- Die von Ihnen benötigten Instances weisen eine höhere Wahrscheinlichkeit auf, dass es zu Unterbrechungen kommt.

Wenn Sie die folgenden Anforderungen und Erwartungen haben, verwenden Sie Amazon EC2-Spot-Instances:

- Die Laufzeit für Ihre Aufträge beträgt in der Regel 30 Minuten oder weniger.
- Sie können potenzielle Unterbrechungen und die Neuplanung von Aufträgen als Teil Ihrer Workload tolerieren. Weitere Informationen finden Sie unter Spot-Instance-Advisor.

- Lang laufende Aufträge können von einem Checkpoint aus neu gestartet werden, wenn sie unterbrochen werden.

Sie können beide Kaufmodelle mischen, indem Sie zuerst auf der Spot-Instance einreichen und dann die On-Demand-Instance als Fallback-Option verwenden. Senden Sie beispielsweise Ihre Aufträge an eine Warteschlange, die mit Datenverarbeitungsumgebungen verbunden ist, die auf Amazon EC2-Spot-Instances ausgeführt werden. Wenn ein Auftrag unterbrochen wird, fangen Sie das Ereignis von Amazon ab EventBridge und korrelieren Sie es mit einer Spot-Instance-Rückrufung. Senden Sie dann den Auftrag erneut mit einer -AWS LambdaFunktion oder an eine On-Demand-WarteschlangeAWS Step Functions. Weitere Informationen finden Sie unter Tutorial: Senden von Amazon Simple Notification Service-Warnungen für fehlgeschlagene Auftragsereignisse, Bewährte Methoden für den Umgang mit Amazon EC2-Spot-Instance-Unterbrechungen und Mit Step Functions verwalten. AWS Batch

> **Important**
>
> Verwenden Sie verschiedene Instance-Typen, Größen und Availability Zones für Ihre On-Demand-Datenverarbeitungsumgebung, um die Verfügbarkeit des Amazon EC2 Spot-Instance-Pools aufrechtzuerhalten und die Unterbrechungsrate zu verringern.

Verwenden von bewährten Methoden für Amazon EC2 Spot für AWS Batch

Wenn Sie Amazon Elastic Compute Cloud (EC2) Spot-Instances wählen, können Sie Ihren Workflow wahrscheinlich optimieren, um Kosten zu sparen, manchmal erheblich. Weitere Informationen finden Sie unter Bewährte Methoden für Amazon EC2 Spot .

Um Ihren Workflow zur Kosteneinsparung zu optimieren, sollten Sie die folgenden bewährten Methoden für Amazon EC2 Spot für berücksichtigenAWS Batch:

- Wählen Sie die **`SPOT_CAPACITY_OPTIMIZED`** Zuweisungsstrategie – AWS Batch wählt Amazon EC2-Instances aus den tiefsten Amazon EC2-Spot-Kapazitätspools aus. Wenn Sie Bedenken wegen Unterbrechungen haben, ist dies eine geeignete Wahl. Weitere Informationen finden Sie unter Zuweisungsstrategien.
- Diversifizieren von Instance-Typen – Um Ihre Instance-Typen zu diversifizieren, sollten Sie kompatible Größen und Familien berücksichtigen und dann je nach Preis oder Verfügbarkeit AWS

Batch auswählen lassen. Betrachten Sie beispielsweise `c5.24xlarge` als Alternative zu den `m5d` Familien `c5.12xlarge` oder `c5a`, `c5n`, `m5`, und `c5d`. Weitere Informationen finden Sie unter Flexibel sein bei Instance-Typen und Availability Zones.

- Reduzieren der Auftragslaufzeit oder des Prüfpunkts – Wir raten davon ab, Aufträge auszuführen, die eine Stunde oder länger dauern, wenn Amazon EC2-Spot-Instances verwendet werden, um Unterbrechungen zu vermeiden. Wenn Sie Ihre Aufträge in kleinere Teile aufteilen oder überprüfen, die aus 30 Minuten oder weniger bestehen, können Sie die Möglichkeit von Unterbrechungen erheblich verringern.
- Automatisierte Wiederholungen verwenden – Um Unterbrechungen von AWS Batch Aufträgen zu vermeiden, legen Sie automatisierte Wiederholungen für Aufträge fest. Batch-Aufträge können aus einem der folgenden Gründe unterbrochen werden: Es wird ein Beendigungscode ungleich Null zurückgegeben, ein Servicefehler oder eine Instance-Rückrufung. Sie können bis zu 10 automatisierte Wiederholungen einrichten. Zu Beginn empfehlen wir Ihnen, mindestens 1–3 automatisierte Wiederholungen festzulegen. Informationen zum Nachverfolgen von Amazon EC2 Spot-Unterbrechungen finden Sie unter Spot Interruption Dashboard.

  Wenn Sie AWS Batchfür den Wiederholungsparameter festlegen, wird der Auftrag an der Spitze der Auftragswarteschlange platziert. Das heißt, der Auftrag hat Priorität. Wenn Sie die Auftragsdefinition erstellen oder den Auftrag in der absendenAWS CLI, können Sie eine Wiederholungsstrategie konfigurieren. Weitere Informationen finden Sie unter submit-job.

```
$ aws batch submit-job --job-name MyJob \
    --job-queue MyJQ \
    --job-definition MyJD \
    --retry-strategy attempts=2
```

- Benutzerdefinierte Wiederholungen verwenden – Sie können eine Auftragswiederholungsstrategie für einen bestimmten Anwendungs-Beendigungscode oder eine Instance-Wiederholung konfigurieren. Wenn im folgenden Beispiel der Host den Fehler verursacht, kann der Auftrag bis zu fünfmal wiederholt werden. Wenn der Auftrag jedoch aus einem anderen Grund fehlschlägt, wird der Auftrag beendet und der Status wird auf gesetzt`FAILED`.

```
"retryStrategy": {
    "attempts": 5,
    "evaluateOnExit":
    [{
        "onStatusReason" :"Host EC2*",
        "action": "RETRY"
    },{
```

```
      "onReason" : "*"
        "action": "EXIT"
    }]
}
```

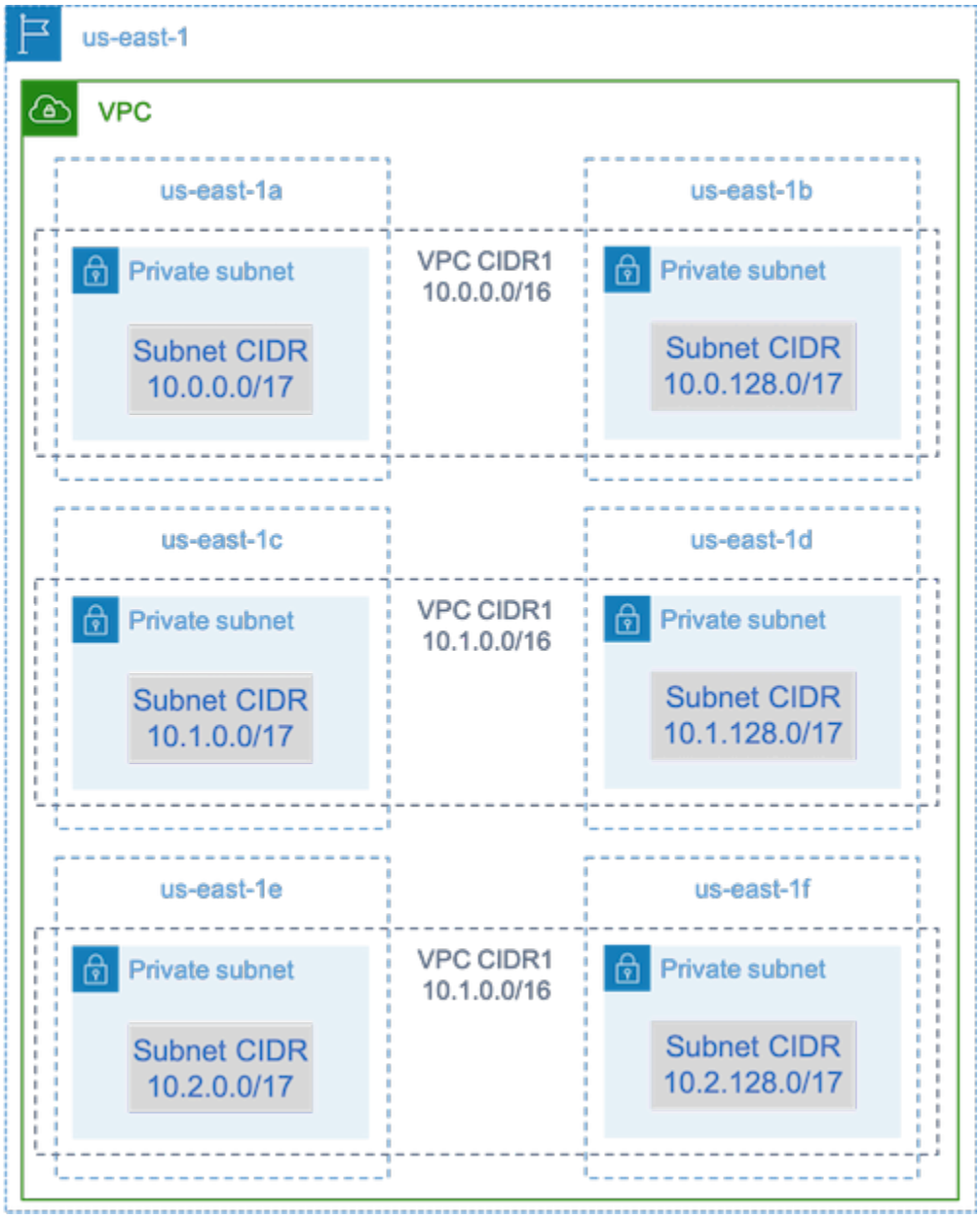
- Verwenden des Spot-Unterbrechungs-Dashboards – Sie können das Spot-Unterbrechungs-Dashboard verwenden, um Spot-Unterbrechungen zu verfolgen. Die Anwendung stellt Metriken zu Amazon EC2-Spot-Instances bereit, die zurückgewonnen werden, und welche Availability Zones sich in Spot-Instances befinden. Weitere Informationen finden Sie unter Spot-Unterbrechungs-Dashboard

Häufige Fehler und Fehlerbehebung

Fehler in treten AWS Batch häufig auf Anwendungsebene auf oder werden durch Instance-Konfigurationen verursacht, die Ihre spezifischen Auftragsanforderungen nicht erfüllen. Weitere Probleme sind Aufträge, die im `RUNNABLE` Status hängen bleiben, oder Datenverarbeitungsumgebungen, die im `INVALID` Status hängen bleiben. Weitere Informationen zur Fehlerbehebung bei Aufträgen, die im `RUNNABLE` Status hängen bleiben, finden Sie unter Jobs stecken in einem `RUNNABLE` Status fest. Informationen zur Fehlerbehebung bei Computing-Umgebungen in einem -`INVALID`Status finden Sie unter `INVALID`Datenverarbeitungsumgebung.

- Überprüfen der Amazon EC2 Spot vCPU-Kontingente – Stellen Sie sicher, dass Ihre aktuellen Servicekontingente die Auftragsanforderungen erfüllen. Angenommen, Ihr aktuelles Servicekontingent beträgt 256 vCPUs und der Auftrag erfordert 10 000 vCPUs. Dann erfüllt das Servicekontingent nicht die Auftragsanforderung. Weitere Informationen und Anweisungen zur Fehlerbehebung finden Sie unter Amazon EC2-Servicekontingente und Wie erhöhe ich das Servicekontingent meiner AmazonEC2resources?.
- Aufträge schlagen fehl, bevor die Anwendung ausgeführt wird – Einige Aufträge schlagen möglicherweise aufgrund eines `DockerTimeoutError` Fehlers oder eines `CannotPullContainerError` Fehlers fehl. Informationen zur Fehlerbehebung finden Sie unter Wie behebe ich den Fehler „DockerTimeoutError“ in AWS Batch?.
- Unzureichende IP-Adressen – Die Anzahl der IP-Adressen in Ihrer VPC und Ihren Subnetzen kann die Anzahl der Instances einschränken, die Sie erstellen können. Verwenden Sie Classless Inter-Domain Routings (CIDRs), um mehr IP-Adressen bereitzustellen, als zum Ausführen Ihrer Workloads erforderlich sind. Bei Bedarf können Sie auch eine dedizierte VPC mit einem großen Adressraum erstellen. Sie können beispielsweise eine VPC mit mehreren CIDRs in `10.x.0.0/16` und ein Subnetz in jeder Availability Zone mit einem CIDR von erstellen`10.x.y.0/17`. In diesem

Beispiel liegt x zwischen 1 und 4 und y ist entweder 0 oder 128. Diese Konfiguration bietet 36.000 IP-Adressen in jedem Subnetz.



- Stellen Sie sicher, dass Instances bei Amazon EC2 registriert sind – Wenn Sie Ihre Instances in der Amazon EC2-Konsole sehen, aber keine Amazon-Elastic-Container-Service-Container-Instances in Ihrem Amazon-ECS-Cluster, ist der Amazon-ECS-Agent möglicherweise nicht auf einem Amazon Machine Image (AMI) installiert. Der Amazon-ECS-Agent, die Amazon EC2-Daten in Ihrem AMI oder die Startvorlage sind möglicherweise ebenfalls nicht korrekt konfiguriert. Um die Ursache zu isolieren, erstellen Sie eine separate Amazon EC2-Instance oder stellen Sie mithilfe von SSH eine Verbindung zu einer vorhandenen Instance her. Weitere Informationen finden Sie unter Amazon-

ECS-Container-Agent-Konfiguration, Amazon-ECS-Protokolldateispeicherorte und Ressourcen-AMIs berechnen.

- Überprüfen Sie das AWS Dashboard – Überprüfen Sie das AWS Dashboard, um sicherzustellen, dass der erwartete Auftragsstatus und die Datenverarbeitungsumgebung wie erwartet skaliert werden. Sie können die Auftragsprotokolle auch in überprüfen CloudWatch.
- Überprüfen, ob Ihre Instance erstellt wurde – Wenn eine Instance erstellt wird, bedeutet dies, dass Ihre Datenverarbeitungsumgebung wie erwartet skaliert wurde. Wenn Ihre Instances nicht erstellt werden, suchen Sie die zugehörigen Subnetze in Ihrer Computing-Umgebung, die Sie ändern möchten. Weitere Informationen finden Sie unter Überprüfen einer Skalierungsaktivität für eine Auto Scaling-Gruppe.

  Wir empfehlen Ihnen außerdem, zu überprüfen, ob Ihre Instances Ihre zugehörigen Auftragsanforderungen erfüllen können. Beispielsweise könnte ein Auftrag 1 TiB Speicher benötigen, aber die Datenverarbeitungsumgebung verwendet einen C5-Instance-Typ, der auf 192 GB Speicher begrenzt ist.
- Stellen Sie sicher, dass Ihre Instances von angefordert werden AWS Batch – Überprüfen Sie den AutoAuto Scaling-Gruppenverlauf, um sicherzustellen, dass Ihre Instances von angefordert werdenAWS Batch. Dies ist ein Hinweis darauf, wie Amazon EC2 versucht, Instances zu erwerben. Wenn Sie die Fehlermeldung erhalten, dass Amazon EC2 Spot keine Instance in einer bestimmten Availability Zone erwerben kann, kann dies daran liegen, dass die Availability Zone keine bestimmte Instance-Familie anbietet.
- Überprüfen Sie, ob Instances bei Amazon ECS registriert sind – Wenn Sie Instances in der Amazon EC2-Konsole sehen, aber keine Amazon-ECS-Container-Instances in Ihrem Amazon-ECS-Cluster, ist der Amazon-ECS-Agent möglicherweise nicht auf dem Amazon Machine Image (AMI) installiert. Darüber hinaus sind der Amazon-ECS-Agent, die Amazon EC2-Daten in Ihrem AMI oder die Startvorlage möglicherweise nicht korrekt konfiguriert. Um die Ursache zu isolieren, erstellen Sie eine separate Amazon EC2-Instance oder stellen Sie mithilfe von SSH eine Verbindung zu einer vorhandenen Instance her. Weitere Informationen finden Sie unter CloudWatch Agent-Konfigurationsdatei: Abschnitt Protokolle, Speicherorte von Amazon-ECS-Protokolldateien und Ressourcen-AMIs berechnen.
- Öffnen eines Support-Tickets – Wenn nach der Fehlerbehebung immer noch Probleme auftreten und Sie über einen Support-Plan verfügen, öffnen Sie ein Support-Ticket. Stellen Sie sicher, dass Sie im Support-Ticket Informationen über das Problem, die Workload-Spezifikationen, die Konfiguration und die Testergebnisse angeben. Weitere Informationen finden Sie unter Vergleichen von -AWS SupportPlänen.

- Sehen Sie sich die HPC-Foren AWS Batch und an – Weitere Informationen finden Sie in den HPC-Foren AWS Batch und .
- Überprüfen Sie das AWS Batch Runtime Monitoring Dashboard – Dieses Dashboard verwendet eine Serverless-Architektur, um Ereignisse von Amazon ECSAWS Batch, und Amazon EC2 zu erfassen, um Einblicke in Aufträge und Instances zu erhalten. Weitere Informationen finden Sie unter AWS Batch Lösung für Laufzeitüberwachungs-Dashboards.

Dokumentverlauf

In der folgenden Tabelle werden die wichtigen Änderungen an der Dokumentation seit der ersten Veröffentlichung von beschrieben AWS Batch. Wir aktualisieren die Dokumentation regelmäßig, um das Feedback, das Sie uns senden, einzuarbeiten.

Änderung	Beschreibung	Datum
AWS Batch Unterstützte Amazon-EKS-Versionen aktualisiert	Die von AWS Batch unterstützten Amazon-EKS-Versionen wurden aktualisiert, um Version 1.22 zu entfernen.	11. März 2024
AWS Batch Unterstützte Amazon-EKS-Versionen aktualisiert	Die von AWS Batch unterstützten Amazon-EKS-Versionen wurden aktualisiert und enthalten nun Version 1.29.	29. Februar 2024
Automatisierte Auftragswiederholungen	Das Codebeispiel wurde korrigiert.	29. Februar 2024
Fügt Unterstützung für Aufträge mit mehreren Containern für hinzu AWS Batch	Fügt Unterstützung für Multi-Container-Aufträge für AWS Batch für Amazon Elastic Container Service, Amazon Elastic Kubernetes Service und hinzu AWS Fargate.	28. Februar 2024
AWS Batch Unterstützte Amazon-EKS-Versionen aktualisiert	Die von AWS Batch unterstützten Amazon-EKS-Versionen wurden aktualisiert und enthalten nun Version 1.28	27. Januar 2024
Aktualisiert BatchServiceRolePolicy und AWSBatchServiceRole	BatchServiceRolePolicy Aktualisiert für die Unterstützung der	05. Dezember 2023

	Beschreibung des Anforderungsverlaufs und Amazon EC2 Auto Scaling der Aktivitäten von Spot-Flotten. AWSBatchServiceRole Aktualisiert, um Anweisungs-IDs hinzuzufügen, erteilen Sie AWS Batch Berechtigungen für `ec2:DescribeSpotFleetRequestHistory` und `autoscaling:DescribeScalingActivities` .	
AWS Batch auf Amazon EKS	AWS Batch fügt Unterstützung für das Ausführen von Aufträgen auf Amazon-EKS-Clustern hinzu.	25. Oktober 2022
Serviceübergreifende Confused-Deputy-Prävention für AWS Batch	AWS Batch bietet jetzt eine Problemumgehung für das Sicherheitsproblem des verwirrten Stellvertreters, das auftritt, wenn eine Entität (ein Service oder ein Konto) von einer anderen Entität gezwungen wird, eine Aktion auszuführen.	6. Juni 2022

Schnittstellen-VPC-Endpunkte (AWS PrivateLink)	Unterstützung für die Konfiguration von Schnittstellen-VPC-Endpunkten hinzugefügt, die von unterstützt werden AWS PrivateLink. Das bedeutet, dass Sie eine private Verbindung zwischen Ihrer VPC und herstellen können, AWS Batch ohne dass ein Zugriff über eine NAT-Instance, eine VPN-Verbindung oder erforderlich ist AWS Direct Connect.	15. April 2022
Erweiterte Aktualisierungen der Datenverarbeitungsumgebung	AWS Batch erweitert e Support-Updates für Computing-Umgebungen.	14. April 2022
AWS Von verwaltete Richtlinienaktualisierungen – Aktualisierung vorhandener Richtlinien	AWS Batch hat vorhandene verwaltete Richtlinien aktualisiert.	6. Dezember 2021
Fair-Share-Planung	AWS Batch fügt Unterstützung für das Hinzufügen von Planungsrichtlinien zu Auftragswarteschlangen hinzu.	9. November 2021
Amazon EFS	AWS Batch fügt Unterstützung für das Hinzufügen von Amazon-EFS-Dateisystemen zu Ihren Auftragsdefinitionen hinzu.	01. April 2021
Serviceverknüpfte Rolle hinzugefügt	AWS Batch fügt die AWSServiceRoleForBatch serviceverknüpfte Rolle hinzu.	10. März 2021

AWS Fargate -Unterstützung	AWS Batch fügt Unterstützung für die Ausführung von Aufträgen auf Fargate-Ressourcen hinzu.	3. Dezember 2020
Amazon Linux 2-Unterstützung	AWS Batch fügt Unterstützung für die automatische Auswahl von Amazon Linux 2-AMIs in der Datenverarbeitungsumgebung mithilfe der EC2-Konfigurationsparameter hinzu.	24. November 2020
Verbesserte Wiederholungsstrategie	AWS Batch verbessert die Wiederholungsstrategie für -Aufträge. Jetzt können Aufträge erneut versucht oder weitere Wiederholungsversuche gestoppt werden`ExitCode`, indem die `Reason`, oder `StatusReason` eines Auftrags mit Mustern abgeglichen werden.	20. Oktober 2020
Ressourcen-Markierung	AWS Batch fügt Unterstützung für das Hinzufügen von Metadaten-Tags zu Ihren Datenverarbeitungsumgebungen, Auftragsdefinitionen, Auftragswarteschlangen und Aufträgen hinzu.	7. Oktober 2020
Secrets	AWS Batch fügt Unterstützung für die Übergabe von Secrets an -Aufträge hinzu.	1. Oktober 2020

Protokollierung	AWS Batch fügt Unterstützung für die Angabe zusätzlicher Protokolltreiber für -Aufträge hinzu.	1. Oktober 2020
Zuweisungsstrategien	AWS Batch fügt Unterstützung für mehrere Strategien zur Auswahl von Instance-Typen hinzu.	16. Oktober 2019
EFA-Unterstützung	AWS Batch fügt Unterstützung für Elastic Fabric Adapter (EFA)-Geräte hinzu.	2. August 2019
GPU-Planung	AWS Batch fügt GPU-Planung hinzu. Mit dieser Funktion können Sie die Anzahl der GPUs angeben, die jeder Auftrag benötigt, und Instances entsprechend AWS Batch hochskalieren.	4. April 2019
Parallele Aufträge mit mehreren Knoten	AWS Batch fügt Unterstützung für parallele Aufträge mit mehreren Knoten hinzu. Sie können diese Funktion verwenden, um einzelne Aufträge auszuführen, die sich über mehrere Amazon EC2-Instances erstrecken.	19. November 2018
Berechtigungen auf Ressourcenebene	AWS Batch unterstützt Berechtigungen auf Ressourcenebene für mehrere API-Operationen.	12. November 2018

Unterstützung für Amazon EC2-Startvorlagen	AWS Batch fügt Unterstützung für die Verwendung von Startvorlagen mit Datenverarbeitungsumgebungen hinzu.	12. November 2018
AWS Batch Auftrags-Timeouts	AWS Batch fügt Unterstützung für das Auftrags-Timeout hinzu. Mit dieser Unterstützung können Sie eine bestimmte Timeout-Dauer für Ihre Aufträge so konfigurieren, dass, wenn ein Auftrag länger ausgeführt wird als gewünscht, den Auftrag AWS Batch beendet.	5. April 2018
AWS Batch -Aufträge als EventBridge Ziele	AWS Batch -Aufträge werden als EventBridge Ziele zur Verfügung gestellt. Indem Sie einfache Regeln erstellen, können Sie Ereignisse abgleichen und als Reaktion darauf AWS Batch Aufträge senden.	1. März 2018
CloudTrail Prüfung für AWS Batch	CloudTrail kann Aufrufe von AWS Batch -API-Aktionen prüfen.	10. Januar 2018
Array-Aufträge	AWS Batch fügt Unterstützung für Array-Aufträge hinzu. Sie können Array-Aufträge für Parameter-Sweep- und Monte-Carlo-Workloads verwenden.	28. November 2017

Erweiterte AWS Batch Markierung	AWS Batch erweitert die Unterstützung für die Tagging-Funktion. Sie können diese Funktion verwenden, um Tags für Amazon EC2 Spot Instances anzugeben, die in verwalteten Datenverarbeitungsumgebungen gestartet werden.	26. Oktober 2017
AWS Batch -Ereignisstream für EventBridge	AWS Batch fügt den Ereignisstream für hinzu EventBridge. Sie können den AWS Batch Ereignisstream verwenden, um Benachrichtigungen über den Status von Aufträgen, die an Ihre Auftragswarteschlangen übermittelt werden, nahezu in Echtzeit zu erhalten.	24. Oktober 2017
Automatisierte Auftragswiederholungen	AWS Batch fügt Unterstützung für Auftragswiederholungen hinzu. Mit diesem Update können Sie eine Wiederholungsstrategie auf Ihre Aufträge und Auftragsdefinitionen anwenden, mit der Ihre Aufträge automatisch erneut versucht werden können, wenn sie fehlschlagen.	28. März 2017
AWS Batch allgemeine Verfügbarkeit	AWS Batch wird eingeführt und dient Ihnen zum Ausführen von Batch-Computing-Workloads auf der AWS Cloud.	5. Januar 2017

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.