aws

Benutzerhandbuch

EC2 Image Builder

EC2 Image Builder: Benutzerhandbuch

Table of Contents

Was ist EC2 Image Builder?

EC2 Image Builder ist ein vollständig verwaltetes AWS-Service Programm, mit dem Sie die Erstellung, Verwaltung und Bereitstellung von benutzerdefinierten, sicheren und up-to-date Server-Images automatisieren können. Sie können die APIs AWS Management Console, oder verwenden AWS Command Line Interface, um benutzerdefinierte Images in Ihrem AWS-Konto zu erstellen.

Sie besitzen die benutzerdefinierten Bilder, die Image Builder in Ihrem Konto erstellt. Sie können Pipelines so konfigurieren, dass Updates und System-Patches für die Images, die Sie besitzen, automatisiert werden. Sie können auch einen eigenständigen Befehl ausführen, um ein Image mit den von Ihnen definierten Konfigurationsressourcen zu erstellen.

Der Image Builder Builder-Pipeline-Assistent kann Sie wie folgt durch die Schritte zum Erstellen eines benutzerdefinierten Images führen:

1. Wählen Sie ein Basis-Image für Ihre Anpassungen aus.
2. Fügen Sie Ihrem Basis-Image Software hinzu oder entfernen Sie sie daraus.
3. Passen Sie Einstellungen und Skripts mit Build-Komponenten an.
4. Führen Sie ausgewählte Tests aus oder erstellen Sie benutzerdefinierte Testkomponenten.
5. Verteilen Sie AMIs an AWS-Regionen und AWS-Konten.
6. Wenn Ihre Image Builder Builder-Pipeline ein benutzerdefiniertes Amazon Machine Image (AMI) für die Verteilung erstellt, können Sie andere AWS-Konten Organisationen und Organisationseinheiten autorisieren, es von Ihrem Konto aus zu starten. Ihrem Konto werden Gebühren in Rechnung gestellt, die mit dem AMI verbunden sind.

Inhalt des Abschnitts

Funktionen von EC2 Image Builder

EC2 Image Builder bietet die folgenden Funktionen:

Steigern Sie die Produktivität und reduzieren Sie den Arbeitsaufwand für die Erstellung konformer Images up-to-date

Image Builder reduziert den Arbeitsaufwand für die Erstellung und Verwaltung von Images in großem Maßstab, indem es Ihre Build-Pipelines automatisiert. Sie können Ihre Builds automatisieren, indem Sie Ihren bevorzugten Zeitplan für die Build-Ausführung angeben. Durch die Automatisierung werden die Betriebskosten für die Wartung Ihrer Software mit den neuesten Betriebssystem-Patches gesenkt.

Erhöhen Sie die Verfügbarkeit Ihrer Dienste

Image Builder bietet Zugriff auf Testkomponenten, mit denen Sie Ihre Images vor der Bereitstellung testen können. Sie können auch benutzerdefinierte Testkomponenten mit AWS Task Orchestrator and Executor (AWSTOE) erstellen und diese verwenden. Image Builder verteilt Ihr Image nur, wenn alle konfigurierten Tests erfolgreich waren.

Legen Sie die Sicherheitslatte für Bereitstellungen höher

Mit Image Builder können Sie Images erstellen, die unnötige Sicherheitslücken bei Komponenten vermeiden. Sie können AWS Sicherheitseinstellungen anwenden, um sichere out-of-the-box Images zu erstellen, die branchenspezifischen und internen Sicherheitskriterien entsprechen. Image Builder bietet auch Sammlungen von Einstellungen für Unternehmen in regulierten Branchen. Mithilfe dieser Einstellungen können Sie schnell und einfach konforme Images für STIG-Standards erstellen. Eine vollständige Liste der über Image Builder verfügbaren STIG-Komponenten finden Sie unterVon Amazon verwaltete STIG-Härtungskomponenten für EC2 Image Builder.

Zentralisierte Durchsetzung und Nachverfolgung der Herkunft

Mithilfe der integrierten Integrationen mit können Sie mit Image Builder Richtlinien durchsetzen AWS Organizations, die Konten darauf beschränken, Instances nur von genehmigten AMIs auszuführen.

Vereinfachte gemeinsame Nutzung von Ressourcen zwischen AWS-Konten

EC2 Image Builder ist in AWS Resource Access Manager (AWS RAM) integriert, sodass Sie bestimmte Ressourcen mit anderen AWS-Konto oder über AWS Organizations diese gemeinsam nutzen können. EC2 Image Builder Builder-Ressourcen, die gemeinsam genutzt werden können, sind:

- Komponenten
- Bilder
- Image-Rezepte
- Container-Rezepte

Weitere Informationen finden Sie unter EC2 Image Builder Builder-Ressourcen teilen.

Unterstützte Betriebssysteme

Image Builder unterstützt die folgenden Betriebssystemversionen:

Betriebssystem/Distribution	Unterstützte Versionen
Amazon Linux	2 und 2023
CentOS	7 und 8
CentOS Stream	8
Red Hat Enterprise Linux (RHEL)	7 und 8
SUSE Linux Enterprise Server (SUSE)	12 und 15
Ubuntu	18.04 LTS, 20.04 LTS und 22.04 LTS
Windows Server	2012 R2, 2016, 2019 und 2022

Unterstützte Bildformate

Für Ihre benutzerdefinierten AMI-Images können Sie ein vorhandenes AMI als Ausgangspunkt wählen. Für Docker-Container-Images können Sie zwischen öffentlichen Images, die auf gehostet werden DockerHub, vorhandenen Container-Images in Amazon ECR oder von Amazon verwalteten Container-Images wählen.

Konzepte

Die folgenden Begriffe und Konzepte sind für Ihr Verständnis und Ihre Verwendung von EC2 Image Builder von zentraler Bedeutung.

AMI

Ein Amazon Machine Image (AMI) ist die grundlegende Bereitstellungseinheit in Amazon EC2 und gehört zu den Arten von Images, die Sie mit Image Builder erstellen können. Ein AMI ist ein vorkonfiguriertes VM-Image, das das Betriebssystem (OS) und die vorinstallierte Software zur Bereitstellung von EC2-Instances enthält. Weitere Informationen finden Sie unter Amazon Machine Images (AMI).

Image-Pipeline

Eine Image-Pipeline bietet ein Automatisierungs-Framework für die Erstellung sicherer AMIs und Container-Images AWS. Die Image Builder Builder-Image-Pipeline ist mit einem Image- oder Container-Rezept verknüpft, das die Erstellungs-, Validierungs- und Testphasen für einen Image-Build-Lebenszyklus definiert.

Eine Image-Pipeline kann einer Infrastrukturkonfiguration zugeordnet werden, die festlegt, wo Ihr Image erstellt wird. Sie können Attribute wie Instance-Typ, Subnetze, Sicherheitsgruppen, Protokollierung und andere infrastrukturbezogene Konfigurationen definieren. Sie können Ihre Image-Pipeline auch einer Verteilungskonfiguration zuordnen, um festzulegen, wie Sie Ihr Image bereitstellen möchten.

Verwaltetes Image

Ein verwaltetes Image ist eine Ressource in Image Builder, die aus einem AMI- oder Container-Image sowie Metadaten wie Version und Plattform besteht. Das verwaltete Image wird von Image Builder Builder-Pipelines verwendet, um zu bestimmen, welches Basis-Image für den Build verwendet werden soll. In diesem Handbuch werden verwaltete Images manchmal als „Images“ bezeichnet. Ein Image ist jedoch nicht dasselbe wie ein AMI.

Rezept für ein Bild

Ein Image Builder Builder-Image-Rezept ist ein Dokument, das das Basis-Image und die Komponenten definiert, die auf das Basis-Image angewendet werden, um die gewünschte Konfiguration für das Ausgabe-AMI-Image zu erzeugen. Sie können ein Image-Rezept verwenden, um Builds zu duplizieren. Image Builder Builder-Image-Rezepte können mit dem

Konsolenassistenten, der oder der API geteilt, verzweigt und bearbeitet werden. AWS CLI Sie können Image-Rezepte mit Ihrer Versionskontrollsoftware verwenden, um gemeinsam nutzbare, versionierte Image-Rezepte zu verwalten.

Rezept für Container

Ein Image Builder Builder-Container-Rezept ist ein Dokument, das das Basis-Image und die Komponenten definiert, die auf das Basis-Image angewendet werden, um die gewünschte Konfiguration für das Ausgabe-Container-Image zu erzeugen. Sie können ein Container-Rezept verwenden, um Builds zu duplizieren. Sie können Image Builder Builder-Image-Rezepte teilen, verzweigen und bearbeiten, indem Sie den Konsolenassistenten AWS CLI, die oder die API verwenden. Sie können Container-Rezepte mit Ihrer Versionskontrollsoftware verwenden, um gemeinsam nutzbare, versionierte Container-Rezepte zu verwalten.

Basis-Image

Das Basis-Image ist das ausgewählte Image und das Betriebssystem, das zusammen mit den Komponenten in Ihrem Image- oder Container-Rezeptdokument verwendet wird. Das Basisimage und die Komponentendefinitionen ergeben zusammen die gewünschte Konfiguration für das Ausgabebild.

Komponenten

Eine Komponente definiert die Reihenfolge der Schritte, die erforderlich sind, um entweder eine Instanz vor der Image-Erstellung anzupassen (eine Build-Komponente) oder um eine Instanz zu testen, die über das erstellte Image gestartet wurde (eine Testkomponente).

Eine Komponente wird aus einem deklarativen YAML- oder JSON-Dokument im Klartext erstellt, das die Laufzeitkonfiguration für das Erstellen und Validieren oder Testen einer Instanz beschreibt, die von Ihrer Pipeline erzeugt wird. Komponenten werden auf der Instanz mithilfe einer Komponentenverwaltungsanwendung ausgeführt. Die Komponentenverwaltungsanwendung analysiert die Dokumente und führt die gewünschten Schritte aus.

Nach ihrer Erstellung werden eine oder mehrere Komponenten mithilfe eines Image- oder Container-Rezepts gruppiert, um den Plan für die Erstellung und das Testen einer virtuellen Maschine oder eines Container-Images zu definieren. Sie können öffentliche Komponenten verwenden, die Eigentum sind und von denen diese verwaltet werden AWS, oder Sie können eigene Komponenten erstellen. Weitere Informationen zu Komponenten finden Sie unterAWS Task Orchestrator and Executor Komponentenmanager.

Dokument zu den Komponenten

Ein deklaratives YAML- oder JSON-Dokument im Klartext, das die Konfiguration für eine Anpassung beschreibt, die Sie auf Ihr Bild anwenden können. Das Dokument wird verwendet, um eine Build- oder Testkomponente zu erstellen.

Laufzeitphasen

EC2 Image Builder hat zwei Laufzeitphasen: Build und Test. Jede Laufzeitphase besteht aus einer oder mehreren Phasen, deren Konfiguration im Komponentendokument definiert ist.

Konfigurationsphasen

Die folgende Liste zeigt die Phasen, die während der Build - und Testphase ausgeführt werden:

Erstellungsphase:

Build-Phase

Eine Image-Pipeline beginnt mit der Buildphase der Build-Phase, wenn sie ausgeführt wird. Das Basis-Image wird heruntergeladen, und die für die Buildphase der Komponente angegebene Konfiguration wird angewendet, um eine Instanz zu erstellen und zu starten.

Phase validieren

Nachdem Image Builder die Instanz gestartet und alle Anpassungen der Buildphase angewendet hat, beginnt die Validierungsphase. Während dieser Phase stellt Image Builder sicher, dass alle Anpassungen wie erwartet funktionieren, basierend auf der Konfiguration, die die Komponente für die Validierungsphase angibt. Wenn die Instanzvalidierung erfolgreich ist, stoppt Image Builder die Instanz, erstellt ein Image und fährt dann mit der Testphase fort.

Testphase:

Testphase

Während dieser Phase startet Image Builder eine Instanz aus dem Image, das es nach erfolgreichem Abschluss der Überprüfungsphase erstellt hat. Image Builder führt während dieser Phase Testkomponenten aus, um zu überprüfen, ob die Instanz fehlerfrei ist und wie erwartet funktioniert.

Testphase für Container-Hosts

Nachdem Image Builder die Testphase für alle Komponenten ausgeführt hat, die Sie im Container-Rezept ausgewählt haben, führt Image Builder diese Phase für Container-Workflows

aus. In der Container-Host-Testphase können zusätzliche Tests ausgeführt werden, die das Containermanagement und benutzerdefinierte Laufzeitkonfigurationen validieren.

Workflow

Workflows definieren die Reihenfolge der Schritte, die Image Builder beim Erstellen eines neuen Images ausführt. Für alle Images gibt es Workflows zum Erstellen und Testen. Container verfügen über einen zusätzlichen Workflow für die Verteilung.

Workflow-Typen

`BUILD`

Deckt die Konfiguration in der Buildphase für jedes erstellte Image ab.

`TEST`

Deckt die Konfiguration der Testphase für jedes erstellte Image ab.

`DISTRIBUTION`

Deckt den Verteilungsworkflow für Container-Images ab.

Preisgestaltung

Die Verwendung von EC2 Image Builder zur Erstellung benutzerdefinierter AMI- oder Container-Images ist kostenlos. Für andere Dienste, die im Prozess verwendet werden, gelten jedoch die Standardpreise. Die folgende Liste enthält die Verwendung einiger AWS-Services , für die je nach Konfiguration Kosten anfallen können, wenn Sie Ihre benutzerdefinierten AMI- oder Container-Images erstellen, erstellen, speichern und verteilen.

- Starten einer EC2-Instance
- Speichern von Protokollen auf Amazon S3
- Validierung von Bildern mit Amazon Inspector
- Speichern von Amazon EBS-Snapshots für Ihre AMIs
- Speichern von Container-Images in Amazon ECR
- Container-Images in und aus Amazon ECR übertragen und daraus ziehen

- Wenn Systems Manager Advanced Tier aktiviert ist und Amazon EC2 EC2-Instances mit lokaler Aktivierung ausgeführt werden, werden Ihnen möglicherweise Ressourcen über Systems Manager in Rechnung gestellt

Verwandt AWS-Services

EC2 Image Builder verwendet andere AWS-Services , um Images zu erstellen. Abhängig von Ihrer Image Builder Builder-Image-Rezeptur oder Container-Rezeptkonfiguration können die folgenden Dienste verwendet werden.

AWS License Manager

AWS License Manager ermöglicht es Ihnen, Lizenzkonfigurationen aus einem Lizenzkonfigurationsspeicher für ein Konto zu erstellen und anzuwenden. Für jedes AMI können Sie Image Builder verwenden, um es an eine bereits bestehende Lizenzkonfiguration anzuhängen, auf die Sie AWS-Konto im Rahmen des Image Builder Builder-Workflows Zugriff haben. Lizenzkonfigurationen können nur auf AMIs angewendet werden. Image Builder kann nur bereits vorhandene Lizenzkonfigurationen verwenden und Lizenzkonfigurationen nicht direkt erstellen oder ändern. License Manager Manager-Einstellungen werden nicht repliziert AWS-Regionen , was in Ihrem Konto aktiviert sein muss, z. B. zwischen den Regionen `ap-east-1` (Asien-Pazifik: Hongkong) und `me-south-1` (Naher Osten: Bahrain).

AWS Organizations

AWS Organizations ermöglicht es Ihnen, Service Control Policies (SCP) auf Konten in Ihrer Organisation anzuwenden. Sie können einzelne Richtlinien erstellen, verwalten, aktivieren und deaktivieren. Wie bei allen anderen AWS Artefakten und Diensten beachtet Image Builder die in AWS Organizations definierten Richtlinien. AWS bietet Vorlagen-SCPs für gängige Szenarien, wie z. B. die Durchsetzung von Beschränkungen für Mitgliedskonten, Instances nur mit genehmigten AMIs zu starten.

Amazon Inspector

Image Builder verwendet Amazon Inspector als Standardagent für das Scannen von Sicherheitslücken, um Sicherheitsgrundlagen für Amazon Linux 2, Windows Server 2012 und Windows Server 2016 festzulegen. Weitere Informationen finden Sie unter Was ist Amazon Inspector?

AWS Resource Access Manager

AWS Resource Access Manager (AWS RAM) ermöglicht es Ihnen, Ihre Ressourcen mit anderen AWS-Konto oder über andere zu teilen AWS Organizations. Wenn Sie mehrere haben AWS-Konten, können Sie Ressourcen zentral erstellen und diese Ressourcen mit anderen Konten teilen. AWS RAM EC2 Image Builder ermöglicht die gemeinsame Nutzung der folgenden Ressourcen: Komponenten, Bilder und Bildrezepte. Weitere Informationen zu AWS RAM finden Sie im AWS Resource Access Manager Benutzerhandbuch. Informationen zur gemeinsamen Nutzung von Image Builder Builder-Ressourcen finden Sie unterEC2 Image Builder Builder-Ressourcen teilen.

CloudWatch Amazon-Protokolle

Sie können Amazon CloudWatch Logs verwenden, um Ihre Protokolldateien von EC2-Instances, Amazon Route 53 und anderen Quellen zu überwachen AWS CloudTrail, zu speichern und darauf zuzugreifen.

Amazon Elastic Container Registry (Amazon ECR)

Amazon ECR ist ein verwalteter AWS Container-Image-Registry-Service, der sicher, skalierbar und zuverlässig ist. Container-Images, die Sie mit Image Builder erstellen, werden in Amazon ECR in Ihrer Quellregion (in der Ihr Build ausgeführt wird) und in allen Regionen gespeichert, in denen Sie das Container-Image verteilen. Weitere Informationen zu Amazon ECR finden Sie im Amazon Elastic Container Registry User Guide.

So funktioniert EC2 Image Builder

Wenn Sie den Assistenten für die EC2 Image Builder Builder-Pipeline-Konsole verwenden, um ein benutzerdefiniertes Image zu erstellen, führt Sie ein Assistent durch die folgenden Schritte.

1. Pipeline-Details angeben — Geben Sie Informationen zu Ihrer Pipeline ein, z. B. einen Namen, eine Beschreibung, Tags und einen Zeitplan für die Ausführung automatisierter Builds. Sie können manuelle Builds wählen, wenn Sie dies bevorzugen.
2. Rezept wählen — Wählen Sie zwischen der Erstellung eines AMI oder der Erstellung eines Container-Images. Für beide Arten von Ausgabe-Images geben Sie einen Namen und eine Version für Ihr Rezept ein, wählen ein Basis-Image aus und wählen Komponenten aus, die zum Erstellen und Testen hinzugefügt werden sollen. Sie können auch die automatische Versionierung wählen, um sicherzustellen, dass Sie immer die neueste verfügbare Betriebssystemversion (OS) für Ihr Basis-Image verwenden. Container-Rezepte definieren zusätzlich Dockerfiles und das Amazon ECR-Ziel-Repository für Ihr Docker-Container-Ausgabe-Image.

   > **Note**
   >
   > Komponenten sind die Bausteine, die von einem Image-Rezept oder einem Container-Rezept verwendet werden. Zum Beispiel Pakete für die Installation, Schritte zur Stärkung der Sicherheit und Tests. Das ausgewählte Basisimage und die Komponenten bilden ein Image-Rezept.

3. Definieren Sie die Infrastrukturkonfiguration — Image Builder startet EC2-Instances in Ihrem Konto, um Images anzupassen und Validierungstests durchzuführen. In den Infrastrukturkonfigurationseinstellungen werden die Infrastrukturdetails für die Instances angegeben, die AWS-Konto während des Build-Prozesses in Ihrem ausgeführt werden.
4. Definieren Sie die Verteilungseinstellungen — Wählen Sie die AWS Regionen aus, in die Ihr Image verteilt werden soll, nachdem der Build abgeschlossen und alle Tests bestanden hat. Die Pipeline verteilt Ihr Image automatisch an die Region, in der der Build ausgeführt wird, und Sie können die Image-Verteilung für andere Regionen hinzufügen.

Die Images, die Sie aus Ihrem benutzerdefinierten Basis-Image erstellen, befinden sich in Ihrem AWS-Konto. Sie können Ihre Image-Pipeline so konfigurieren, dass aktualisierte und gepatchte Versionen Ihres Images erstellt werden, indem Sie einen Build-Zeitplan eingeben. Wenn der Build abgeschlossen ist, können Sie eine Benachrichtigung über Amazon Simple Notification Service

(SNS) erhalten. Neben der Erstellung eines endgültigen Images generiert der Image Builder Builder-Konsolenassistent ein Rezept, das mit vorhandenen Versionskontrollsystemen und CI/CD-Pipelines (Continuous Integration/Continuous Deployment) für wiederholbare Automatisierung verwendet werden kann. Sie können Ihr Rezept teilen und neue Versionen erstellen.

Inhalt des Abschnitts

AMI-Elemente

Ein Amazon Machine Image (AMI) ist ein vorkonfiguriertes Image einer virtuellen Maschine (VM), das das Betriebssystem und die Software für die Bereitstellung von EC2-Instances enthält.

Ein AMI umfasst die folgenden Elemente:

- Eine Vorlage für das Root-Volume der VM. Wenn Sie eine Amazon EC2 EC2-VM starten, enthält das Root-Geräte-Volume das Image zum Starten der Instance. Wenn der Instance-Speicher verwendet wird, ist das Root-Gerät ein Instance-Speicher-Volume, das anhand einer Vorlage in Amazon S3 erstellt wurde. Weitere Informationen finden Sie unter Amazon EC2 Root Device Volume.
- Wenn Amazon EBS verwendet wird, ist das Root-Gerät ein EBS-Volume, das aus einem EBS-Snapshot erstellt wurde.
- Startberechtigungen, die festlegen AWS-Konten , wer VMs mit dem AMI starten kann.
- Blockieren Sie Gerätezuordnungsdaten, die angeben, welche Volumes nach dem Start an die Instance angehängt werden sollen.
- Eine eindeutige Ressourcen-ID für jede Region, für jedes Konto.

- Metadaten-Nutzlasten wie Tags und Eigenschaften wie Region, Betriebssystem, Architektur, Root-Gerätetyp, Anbieter, Startberechtigungen, Speicherplatz für das Root-Gerät und Signaturstatus.
- Eine AMI-Signatur für Windows-Images zum Schutz vor unbefugter Manipulation. Weitere Informationen finden Sie unter Dokumente zur Identifizierung von Instanzen.

Standardkontingente

Die Standardkontingente für Image Builder finden Sie unter Image Builder Builder-Endpunkte und Kontingente.

AWS Regionen und Endpunkte

Informationen zur Anzeige der Dienstendpunkte für Image Builder finden Sie unter Image Builder Builder-Endpunkte und Kontingente.

Komponentenverwaltung

EC2 Image Builder verwendet eine Komponentenverwaltungsanwendung AWS Task Orchestrator and Executor (AWSTOE), mit der Sie komplexe Workflows orchestrieren, Systemkonfigurationen ändern und Ihre Systeme mit YAML-basierten Skriptkomponenten testen können. Da AWSTOE es sich um eine eigenständige Anwendung handelt, ist keine zusätzliche Einrichtung erforderlich. Es kann auf jeder Cloud-Infrastruktur und vor Ort ausgeführt werden. Informationen zu den ersten Schritten mit der Verwendung AWSTOE als eigenständige Anwendung finden Sie unterFangen Sie an mit AWSTOE.

Image Builder verwendet AWSTOE , um alle Aktivitäten auf der Instanz auszuführen. Dazu gehören die Erstellung und Validierung Ihres Images vor der Erstellung eines Snapshots sowie das Testen des Snapshots, um sicherzustellen, dass er erwartungsgemäß funktioniert, bevor das endgültige Image erstellt wird. Weitere Informationen darüber, wie Image Builder seine AWSTOE Komponenten verwaltet, finden Sie unterKomponenten mit Image Builder verwalten. Weitere Informationen zum Erstellen von Komponenten mit AWSTOE finden Sie unterAWS Task Orchestrator and Executor Komponentenmanager.

Testen von Bildern

Sie können AWSTOE Testkomponenten verwenden, um Ihr Image zu validieren und sicherzustellen, dass es erwartungsgemäß funktioniert, bevor Sie das endgültige Image erstellen.

Im Allgemeinen besteht jede Testkomponente aus einem YAML-Dokument, das ein Testskript, eine Test-Binärdatei und Testmetadaten enthält. Das Testskript enthält die Orchestrierungsbefehle zum Starten der Test-Binärdatei, die in jeder vom Betriebssystem unterstützten Sprache geschrieben werden kann. Exit-Statuscodes geben das Testergebnis an. Testmetadaten beschreiben den Test und sein Verhalten, z. B. den Namen, die Beschreibung, die Pfade zur Testbinärdatei und die erwartete Dauer.

Semantische Versionsverwaltung

Image Builder verwendet semantische Versionierung, um Ressourcen zu organisieren und sicherzustellen, dass sie eindeutige IDs haben. Die semantische Version hat vier Knoten:

<major>. <minor>. <patch>/<build>

Sie können Werte für die ersten drei zuweisen und nach allen filtern.

Semantische Versionierung ist im Amazon-Ressourcennamen (ARN) jedes Objekts enthalten, und zwar auf der Ebene, die für dieses Objekt gilt, wie folgt:

1. Versionlose ARNs und Name-ARNs enthalten in keinem der Knoten spezifische Werte. Die Knoten werden entweder komplett weggelassen oder als Platzhalter angegeben, zum Beispiel: x.x.x.
2. <major>Versions-ARNs haben nur die ersten drei Knoten:. <minor>. <patch>
3. ARNs der Build-Version haben alle vier Knoten und verweisen auf einen bestimmten Build für eine bestimmte Version eines Objekts.

Zuweisung: Für die ersten drei Knoten können Sie einen beliebigen positiven Ganzzahlwert, einschließlich Null, mit einer Obergrenze von 2^30-1 oder 1073741823 für jeden Knoten zuweisen. Image Builder weist die Build-Nummer automatisch dem vierten Knoten zu.

Muster: Sie können jedes numerische Muster verwenden, das den Zuweisungsanforderungen für die Knoten entspricht, die Sie zuweisen können. Sie können beispielsweise ein Muster für die Softwareversion wie 1.0.0 oder ein Datum wie 2021.01.01 wählen.

Auswahl: Bei der semantischen Versionierung können Sie bei der Auswahl des Basisimages oder der Komponenten für Ihr Rezept Platzhalter (x) verwenden, um die neuesten Versionen oder Knoten anzugeben. Wenn Sie in einem Knoten einen Platzhalter verwenden, müssen alle Knoten rechts vom ersten Platzhalter ebenfalls Platzhalter sein.

Beispiel: Bei den folgenden aktuellen Versionen: 2.2.4, 1.7.8 und 1.6.8 führt die Versionsauswahl mithilfe von Platzhaltern zu den folgenden Ergebnissen:

- `x.x.x`= 2.2.4
- `1.x.x`= 1,7.8
- `1.6.x`= 1,6.8
- `x.2.x`ist nicht gültig und erzeugt einen Fehler
- `1.x.8`ist nicht gültig und erzeugt einen Fehler

Ressourcen wurden erstellt

Wenn Sie eine Pipeline erstellen, werden keine Ressourcen außerhalb von Image Builder erstellt, sofern nicht Folgendes zutrifft:

- Wenn ein Image im Rahmen des Pipeline-Zeitplans erstellt wird
- Wenn Sie in der Image Builder Builder-Konsole im Menü Aktionen die Option Pipeline ausführen wählen
- Wenn Sie einen dieser Befehle über die API ausführen oder AWS CLI: StartImagePipelineExecution oder CreateImage

Die folgenden Ressourcen werden während der Image-Erstellung erstellt:

AMI-Image-Pipelines

- EC2-Instanz (temporär)
- Systems Manager Inventory Association (über Systems Manager State Manager, falls `EnhancedImageMetadata` aktiviert) auf der EC2-Instance
- Amazon EC2 EC2-AMI
- Der Amazon EBS-Snapshot, der mit Amazon EC2 AMI verknüpft ist

Pipelines für Container-Images

- Docker-Container, der auf einer EC2-Instance läuft (temporär)
- Systems Manager Inventory Association (über Systems Manager State Manager) `EnhancedImageMetadata` ist aktiviert) auf der EC2-Instance

- Docker-Container-Image
- Dockerfile

Nachdem das Image erstellt wurde, werden alle temporären Ressourcen gelöscht.

Distribution

EC2 Image Builder kann AMIs oder Container-Images in jede AWS Region verteilen. Das Image wird in jede Region kopiert, die Sie in dem Konto angeben, mit dem das Image erstellt wurde.

Für AMI-Ausgabebilder können Sie AMI-Startberechtigungen definieren, um zu steuern, AWS-Konten welche EC2-Instances mit dem erstellten AMI starten dürfen. Sie können das Image beispielsweise privat oder öffentlich machen oder es für bestimmte Konten freigeben. Wenn Sie sowohl das AMI an andere Regionen verteilen als auch Startberechtigungen für andere Konten definieren, werden die Startberechtigungen an die AMIs in allen Regionen weitergegeben, in denen das AMI verteilt ist.

Sie können Ihr AWS Organizations Konto auch verwenden, um Beschränkungen für Mitgliedskonten durchzusetzen, sodass nur Instances mit genehmigten und konformen AMIs gestartet werden dürfen. Weitere Informationen finden Sie unter Verwaltung der AWS-Konten in Ihrer Organisation.

Um Ihre Verteilungseinstellungen mit der Image Builder Builder-Konsole zu aktualisieren, folgen Sie den Schritten zuErstellen Sie eine neue Image-Rezeptversion (Konsole), oderErstellen Sie mit der Konsole eine neue Container-Rezeptversion.

Ressourcen teilen

Informationen zum Teilen von Komponenten, Rezepten oder Bildern mit anderen Konten oder innerhalb von AWS Organizations Accounts finden Sie unterEC2 Image Builder Builder-Ressourcen teilen.

-Compliance

Für CIS verwendet EC2 Image Builder Amazon Inspector, um Bewertungen im Hinblick auf Risiken, Sicherheitslücken und Abweichungen von bewährten Verfahren und Compliance-Standards durchzuführen. Image Builder bewertet beispielsweise unbeabsichtigten Netzwerkzugriff, ungepatchte CVEs, öffentliche Internetkonnektivität und Remote-Root-Login-Aktivierung. Amazon Inspector wird als Testkomponente angeboten, die Sie Ihrem Bildrezept hinzufügen können. Weitere Informationen

zu Amazon Inspector finden Sie im Amazon Inspector Inspector-Benutzerhandbuch. Für die Härtung validiert EC2 Image Builder mit STIG. Eine vollständige Liste der über Image Builder verfügbaren STIG-Komponenten finden Sie unterVon Amazon verwaltete STIG-Härtungskomponenten für EC2 Image Builder. Weitere Informationen finden Sie unter Benchmarks des Center for Internet Security (CIS).

Erste Schritte mit EC2 Image Builder

Dieses Kapitel hilft Ihnen beim Einrichten Ihrer Umgebung und beim ersten Erstellen einer automatisierten Image-Pipeline oder Container-Pipeline mithilfe des Konsolenassistenten Create Image Pipeline von EC2 Image Builder.

Inhalt

Voraussetzungen

Überprüfen Sie die folgenden Voraussetzungen, um eine Image-Pipeline mit EC2 Image Builder zu erstellen. Sofern nicht ausdrücklich anders angegeben, sind Voraussetzungen für alle Arten von Pipelines erforderlich.

Mit dem Dienst verknüpfte EC2 Image Builder Builder-Rolle

EC2 Image Builder verwendet eine dienstverknüpfte Rolle, um anderen AWS Diensten in Ihrem Namen Berechtigungen zu erteilen. Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie Ihre erste Image Builder-Ressource in der AWS Management Console, der AWS CLI oder der AWS API erstellen, erstellt Image Builder die serviceverknüpfte Rolle für Sie. Weitere Informationen zu der dienstbezogenen Rolle, die Image Builder in Ihrem Konto erstellt, finden Sie unterVerwenden von serviceverknüpften Rollen für EC2 Image Builder.

Konfigurationsanforderungen

- Image Builder unterstützt AWS PrivateLink. Weitere Informationen zur Konfiguration von VPC-Endpunkten für Image Builder finden Sie unter. EC2 Image Builder und VPC-Schnittstellen-Endpunkte ()AWS PrivateLink
- Die Instances, die Image Builder zum Erstellen von Container-Images verwendet, müssen über Internetzugang verfügen, um sie AWS CLI von Amazon S3 herunterzuladen und gegebenenfalls

ein Basis-Image aus dem Docker Hub-Repository herunterzuladen. Image Builder verwendet das AWS CLI , um das Dockerfile aus dem Container-Rezept abzurufen, wo es als Daten gespeichert ist.

- Die Instanzen, die Image Builder zum Erstellen von Images und zum Ausführen von Tests verwendet, müssen Zugriff auf den Systems Manager Manager-Dienst haben. Die Installationsanforderungen hängen von Ihrem Betriebssystem ab.

  Um die Installationsanforderungen für Ihr Basis-Image zu sehen, wählen Sie die Registerkarte, die Ihrem Basis-Image-Betriebssystem entspricht.

  Linux

  Für Amazon EC2 EC2-Linux-Instances installiert Image Builder den Systems Manager Agent auf der Build-Instance, falls er noch nicht vorhanden ist, und entfernt ihn, bevor das Image erstellt wird.

  Windows

  Image Builder installiert den Systems Manager Agent nicht auf Amazon EC2 Windows Server-Build-Instances. Wenn Ihr Basis-Image nicht mit dem Systems Manager Agent vorinstalliert war, müssen Sie eine Instance von Ihrem Quell-Image aus starten, Systems Manager manuell auf der Instance installieren und ein neues Basis-Image aus Ihrer Instance erstellen.

  Informationen zur manuellen Installation des Systems Manager Manager-Agenten auf Ihrer Amazon EC2 Windows Server-Instance finden Sie unter Manuelles Installieren des Systems Manager Manager-Agenten auf EC2-Instances für Windows Server im AWS Systems Manager Benutzerhandbuch.

Container-Repository (Container-Image-Pipelines)

Für Container-Image-Pipelines definiert das Rezept die Konfiguration für die Docker-Images, die im Ziel-Container-Repository erzeugt und gespeichert werden. Sie müssen das Ziel-Repository erstellen, bevor Sie das Container-Rezept für Ihr Docker-Image erstellen.

Image Builder verwendet Amazon ECR als Ziel-Repository für Container-Images. Um ein Amazon ECR-Repository zu erstellen, folgen Sie den unter Erstellen eines Repositorys im Amazon Elastic Container Registry-Benutzerhandbuch beschriebenen Schritte.

AWS Identity and Access Management (IAM)

Die IAM-Rolle, die Sie Ihrem Instanzprofil zuordnen, muss über Berechtigungen zum Ausführen der in Ihrem Image enthaltenen Build- und Testkomponenten verfügen. Die folgenden IAM-Rollenrichtlinien müssen der IAM-Rolle angehängt werden, die dem Instanzprofil zugeordnet ist:

- EC2InstanceProfileForImageBuilder
- EC2InstanceProfileForImageBuilderECRContainerBuilds
- AmazonSSM ManagedInstanceCore

Wenn Sie die Protokollierung konfigurieren, muss das in Ihrer Infrastrukturkonfiguration angegebene Instance-Profil über `s3:PutObject` Berechtigungen für den Ziel-Bucket () `arn:aws:s3:::BucketName/*` verfügen. Beispielsweise:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::bucket-name/*"
        }
    ]
}
```

Richtlinie anfügen

Die folgenden Schritte führen Sie durch den Prozess des Anfügens der IAM-Richtlinien an eine IAM-Rolle, um die oben genannten Berechtigungen zu gewähren.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Richtlinien aus.
3. Filtern Sie die Liste der Richtlinien mit EC2InstanceProfileForImageBuilder
4. Wählen Sie das bullet neben der Richtlinie aus und wählen Sie in der Dropdownliste Richtlinienaktionen die Option Anhängen aus.

5. Wählen Sie den Namen der IAM-Rolle aus, an die die Richtlinie angehängt werden soll.
6. Wählen Sie Richtlinie anfügen aus.
7. Wiederholen Sie die Schritte 3-6 für die Richtlinien EC2InstanceProfileForImageBuilderECRContainerBuildsund ManagedInstanceCoreAmazonSSM.

> **Note**
>
> Wenn Sie ein mit Image Builder erstelltes Image in ein anderes Konto kopieren möchten, müssen Sie die `EC2ImageBuilderDistributionCrossAccountRole` Rolle in allen Zielkonten erstellen und die Ec2ImageBuilderCrossAccountDistributionAccess-Richtlinie verwaltete Richtlinie an die Rolle anhängen. Weitere Informationen finden Sie unter EC2 Image Builder Builder-Ressourcen teilen.

Greifen Sie auf EC2 Image Builder zu

Sie können EC2 Image Builder über eine der folgenden Schnittstellen verwalten.

- Landingpage der EC2 Image Builder Builder-Konsole. Von der EC2 Image Builder Builder-Konsole aus.
- AWS Command Line Interface (AWS CLI). Sie können die verwenden AWS CLI , um auf AWS API-Operationen zuzugreifen. Weitere Informationen finden Sie unter Installation der AWS Befehlszeilenschnittstelle im AWS Command Line Interface Benutzerhandbuch.
- AWS Tools für SDKs. Sie können AWS SDKs und Tools verwenden, um in Ihrer bevorzugten Sprache auf Image Builder zuzugreifen und es zu verwalten.

Erstellen Sie eine Image-Pipeline mit dem EC2 Image Builder Builder-Konsolenassistenten

Dieses Tutorial führt Sie durch die Erstellung einer automatisierten Pipeline zum Erstellen und Verwalten eines benutzerdefinierten EC2 Image Builder mithilfe des Konsolenassistenten Create Image Pipeline. Damit Sie die Schritte effizient durchführen können, werden Standardeinstellungen verwendet, sofern sie verfügbar sind, und optionale Abschnitte werden übersprungen.

Workflow für die Image-Pipeline erstellen

Schritt 1: Geben Sie die Pipeline-Details an

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Um mit der Erstellung Ihrer Pipeline zu beginnen, wählen Sie Create Image Pipeline aus.
3. Geben Sie im Abschnitt Allgemein Ihren Pipeline-Namen ein (erforderlich).

   > **Tip**
   >
   > Die erweiterte Metadatensammlung ist standardmäßig aktiviert. Um die Kompatibilität zwischen Komponenten und Basis-Images sicherzustellen, sollten Sie sie aktiviert lassen.

4. Im Abschnitt Zeitplan erstellen können Sie die Standardeinstellungen für die Zeitplanoptionen beibehalten. Beachten Sie, dass die für den Standardzeitplan angezeigte Zeitzone die koordinierte Weltzeit (UTC) ist. Weitere Informationen zur UTC-Zeit und zur Ermittlung des Zeitversatzes für Ihre Zeitzone finden Sie unter Abkürzungen für Zeitzonen — Weltweite Liste.

   Wählen Sie für die Einstellungen für Abhängigkeitsupdates die Option Pipeline zur geplanten Zeit ausführen, falls es Abhängigkeitsupdates gibt. Diese Einstellung veranlasst Ihre Pipeline, vor dem Start des Builds nach Updates zu suchen. Wenn keine Updates vorhanden sind, wird der geplante Pipeline-Build übersprungen.

   > **Note**
   >
   > Um sicherzustellen, dass Ihre Pipeline Abhängigkeitsupdates und Builds wie erwartet erkennt, müssen Sie die semantische Versionierung (x.x.x) für Ihr Basis-Image und Ihre

Komponenten verwenden. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung

5. Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 2: Rezept wählen

1. Image Builder ist standardmäßig auf Bestehendes Rezept verwenden im Abschnitt Rezept eingestellt. Wählen Sie zum ersten Mal die Option Neues Rezept erstellen.
2. Wählen Sie im Abschnitt Image-Typ die Option Amazon Machine Image (AMI), um eine Image-Pipeline zu erstellen, die ein AMI erstellt und verteilt.
3. Geben Sie im Abschnitt Allgemein die folgenden erforderlichen Felder ein:
   - Name — Ihr Rezeptname
   - Version — deine Rezeptversion (verwende das Format <major>. <minor>. <patch>, wobei Major, Minor und Patch ganzzahlige Werte sind). Neue Rezepte beginnen in der Regel mit`1.0.0`.
4. Behalten Sie im Abschnitt Quell-Image die Standardwerte für Select image, Image Operating System (OS) und Image-Ursprung bei. Dies führt zu einer Liste von Amazon Linux 2-AMIs, die von Amazon verwaltet werden und aus der Sie für Ihr Basis-Image auswählen können.
   a. Wählen Sie aus der Drop-down-Liste mit dem Imagenamen ein Image aus.
   b. Behalten Sie die Standardeinstellungen für die automatische Versionierung bei (verwenden Sie die neueste verfügbare Betriebssystemversion).

   > **Note**
   >
   > Diese Einstellung stellt sicher, dass Ihre Pipeline die semantische Versionierung für das Basis-Image verwendet, um Abhängigkeitsupdates für automatisch geplante Jobs zu erkennen. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung
5. Behalten Sie im Abschnitt Instanzkonfiguration die Standardwerte für den Systems Manager Manager-Agent bei. Dies führt dazu, dass Image Builder den Systems Manager Manager-Agenten nach Abschluss des Builds und der Tests beibehält, um den Systems Manager Manager-Agent in Ihr neues Image aufzunehmen.

Lassen Sie die Benutzerdaten für dieses Tutorial leer. Sie können diesen Bereich zu anderen Zeiten verwenden, um Befehle oder ein Befehlsskript bereitzustellen, das ausgeführt wird, wenn Sie Ihre Build-Instance starten. Es ersetzt jedoch alle Befehle, die Image Builder möglicherweise hinzugefügt hat, um sicherzustellen, dass Systems Manager installiert ist. Wenn Sie ihn verwenden, stellen Sie sicher, dass der Systems Manager Agent auf Ihrem Basis-Image vorinstalliert ist, oder dass Sie die Installation in Ihren Benutzerdaten angeben.

6. Im Abschnitt Komponenten müssen Sie mindestens eine Build-Komponente auswählen.

   Im Bereich Komponenten erstellen — Amazon Linux können Sie die auf der Seite aufgelisteten Komponenten durchsuchen. Verwenden Sie die Seitennummerierung in der oberen rechten Ecke, um durch zusätzliche Komponenten zu navigieren, die für Ihr Basis-Image-Betriebssystem verfügbar sind. Sie können auch nach bestimmten Komponenten suchen oder mithilfe des Komponenten-Managers Ihre eigene Build-Komponente erstellen.

   Wählen Sie für dieses Tutorial wie folgt eine Komponente aus, die Linux mit den neuesten Sicherheitsupdates aktualisiert:

   a. Filtern Sie die Ergebnisse, indem Sie das Wort `update` in die Suchleiste eingeben, die sich oben im Bedienfeld befindet.

   b. Aktivieren Sie das Kontrollkästchen für die `update-linux` Build-Komponente.

   c. Scrollen Sie nach unten und wählen Sie in der oberen rechten Ecke der Liste Ausgewählte Komponenten die Option Alle erweitern aus.

   d. Behalten Sie die Standardeinstellung für die Versionierungsoptionen bei (Verwenden Sie die neueste verfügbare Komponentenversion).

      > **Note**
      >
      > Diese Einstellung stellt sicher, dass Ihre Pipeline die semantische Versionierung für die ausgewählte Komponente verwendet, um Abhängigkeitsupdates für automatisch geplante Jobs zu erkennen. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung

      Wenn Sie eine Komponente mit Eingabeparametern auswählen würden, würden Sie die Parameter auch in diesem Bereich sehen. Parameter werden in diesem Tutorial

nicht behandelt. Weitere Informationen zur Verwendung von Eingabeparametern in Ihren Komponenten und deren Einstellung in Ihren Rezepten finden Sie unterAWSTOE Komponentenparameter mit EC2 Image Builder verwalten.

Komponenten neu anordnen (optional)

Wenn Sie mehr als eine Komponente ausgewählt haben, die in Ihr Image aufgenommen werden soll, können Sie die drag-and-drop Aktion verwenden, um sie in der Reihenfolge anzuordnen, in der sie während des Erstellungsprozesses ausgeführt werden sollen.

> **Note**
>
> CIS-Härtungskomponenten folgen nicht den Standardregeln für die Reihenfolge der Komponenten in den Image Builder Builder-Rezepten. Die CIS-Härtungskomponenten werden immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests anhand Ihres Ausgabebilds ausgeführt werden.

1. Scrollen Sie zurück zur Liste der verfügbaren Komponenten.
2. Aktivieren Sie das Kontrollkästchen für die `update-linux-kernel-mainline` Build-Komponente (oder eine andere Komponente Ihrer Wahl).
3. Scrollen Sie nach unten zur Liste Ausgewählte Komponenten, um zu sehen, dass es mindestens zwei Ergebnisse gibt.
4. Bei neu hinzugefügten Komponenten wurden die Versionierungs- oder Eingabeparametereinstellungen möglicherweise nicht erweitert. Um die Einstellungen für Versionierungsoptionen oder Eingabeparameter zu erweitern, können Sie den Pfeil neben dem Namen der Einstellung wählen. Um alle Einstellungen für alle ausgewählten Komponenten zu erweitern, können Sie den Schalter Alle erweitern aus- und einschalten.
5. Wählen Sie eine der Komponenten aus und ziehen Sie sie nach oben oder unten, um die Reihenfolge zu ändern, in der die Komponenten ausgeführt werden.
6. Um die `update-linux-kernel-mainline` Komponente zu entfernen, wählen Sie in der oberen rechten Ecke des Komponentenfeldes eine Option `X` aus.
7. Wiederholen Sie den vorherigen Schritt, um alle anderen Komponenten zu entfernen, die Sie möglicherweise hinzugefügt haben, sodass nur die `update-linux` Komponente ausgewählt bleibt.

7. Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 3: Definieren Sie die Infrastrukturkonfiguration — optional

Image Builder startet EC2-Instances in Ihrem Konto, um Images anzupassen und Validierungstests durchzuführen. In den Infrastrukturkonfigurationseinstellungen werden die Infrastrukturdetails für die Instances angegeben, die AWS-Konto während des Build-Prozesses in Ihrem ausgeführt werden.

Im Abschnitt Infrastrukturkonfiguration sind die Konfigurationsoptionen standardmäßig auf eingestellt`Create infrastructure configuration using service defaults`. Dadurch werden eine IAM-Rolle und das zugehörige Instanzprofil für die EC2-Build- und Test-Instances erstellt, die zur Konfiguration Ihres Images verwendet werden. Weitere Informationen zu den Einstellungen für die Infrastrukturkonfiguration finden Sie CreateInfrastructureConfigurationin der EC2 Image Builder API-Referenz.

Für dieses Tutorial verwenden wir die Standardeinstellungen.

> **Note**
>
> Um ein Subnetz für eine private VPC anzugeben, können Sie Ihre eigene benutzerdefinierte Infrastrukturkonfiguration erstellen oder Einstellungen verwenden, die Sie bereits erstellt haben.

- Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 4: Definieren Sie die Verteilungseinstellungen — optional

Zu den Verteilungskonfigurationen gehören der Name des Ausgabe-AMIs, spezifische Regionseinstellungen für die Verschlüsselung, Startberechtigungen und AWS-Konten Organisationen und Organisationseinheiten (OUs), die das Ausgabe-AMI starten können, sowie Lizenzkonfigurationen.

Im Abschnitt Verteilungseinstellungen sind die Konfigurationsoptionen standardmäßig auf eingestellt`Create distribution settings using service defaults`. Diese Option verteilt das Ausgabe-AMI an die aktuelle Region. Weitere Informationen zur Konfiguration Ihrer Verteilungseinstellungen finden Sie unterEC2 Image Builder Builder-Verteilungseinstellungen verwalten.

Für dieses Tutorial verwenden wir die Standardeinstellungen.

- Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 5: Prüfen

Im Abschnitt Überprüfen werden alle Einstellungen angezeigt, die Sie konfiguriert haben. Um Informationen in einem bestimmten Abschnitt zu bearbeiten, wählen Sie die Schaltfläche Bearbeiten in der oberen rechten Ecke des Schrittabschnitts. Wenn Sie beispielsweise Ihren Pipeline-Namen ändern möchten, wählen Sie die Schaltfläche Bearbeiten in der oberen rechten Ecke des Abschnitts Schritt 1: Pipeline-Details.

1. Wenn Sie Ihre Einstellungen überprüft haben, wählen Sie Pipeline erstellen, um Ihre Pipeline zu erstellen.
2. Sie können Erfolgs- oder Fehlschlagsmeldungen oben auf der Seite sehen, während Ihre Ressourcen für die Verteilungseinstellungen, die Infrastrukturkonfiguration, Ihr neues Rezept und die Pipeline erstellt werden. Um Details zu einer Ressource, einschließlich der Ressourcen-ID, anzuzeigen, wählen Sie Details anzeigen.
3. Nachdem Sie die Details für eine Ressource angezeigt haben, können Sie Details zu anderen Ressourcen anzeigen, indem Sie den Ressourcentyp im Navigationsbereich auswählen. Um beispielsweise Details für Ihre neue Pipeline anzuzeigen, wählen Sie im Navigationsbereich Image-Pipelines aus. Wenn Ihr Build erfolgreich war, wird Ihre neue Pipeline in der Liste der Image-Pipelines angezeigt.

Schritt 6: Bereinigen

Ihre Image Builder Builder-Umgebung muss, genau wie Ihr Zuhause, regelmäßig gewartet werden, damit Sie das finden, was Sie benötigen, und Ihre Aufgaben erledigen können, ohne sich durch Unordnung zu wühlen. Stellen Sie sicher, dass Sie die temporären Ressourcen, die Sie zu Testzwecken erstellt haben, regelmäßig bereinigen. Andernfalls könnten Sie diese Ressourcen vergessen und sich später nicht mehr daran erinnern, wofür sie verwendet wurden. Bis dahin ist möglicherweise nicht klar, ob Sie sie sicher loswerden können.

> **Tip**
>
> Um Abhängigkeitsfehler beim Löschen von Ressourcen zu vermeiden, stellen Sie sicher, dass Sie Ihre Ressourcen in der folgenden Reihenfolge löschen:

1. Image-Pipeline
2. Bildrezept
3. Alle verbleibenden Ressourcen

Gehen Sie folgendermaßen vor, um die Ressourcen zu bereinigen, die Sie für dieses Tutorial erstellt haben:

Löschen Sie die Pipeline

1. Um eine Liste der Build-Pipelines anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Image-Pipelines aus.
2. Aktivieren Sie das Kontrollkästchen neben dem Namen der Pipeline, um die Pipeline auszuwählen, die Sie löschen möchten.
3. Wählen Sie oben im Bereich Image-Pipelines im Menü Aktionen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie den Text `Delete` in das Feld ein und wählen Sie „Löschen“.

Lösche das Rezept

1. Um eine Liste der unter Ihrem Konto erstellten Rezepte anzuzeigen, wählen Sie im Navigationsbereich Bildrezepte aus.
2. Aktivieren Sie das Kontrollkästchen neben Rezeptname, um das Rezept auszuwählen, das Sie löschen möchten.
3. Wählen Sie oben im Fenster Bildrezepte im Menü Aktionen die Option Rezept löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie den `Delete` Text in das Feld ein und wählen Sie „Löschen“.

Löschen Sie die Infrastrukturkonfiguration

1. Um eine Liste der unter Ihrem Konto erstellten Infrastrukturkonfigurationen anzuzeigen, wählen Sie im Navigationsbereich Infrastrukturkonfiguration aus.
2. Aktivieren Sie das Kontrollkästchen neben Konfigurationsname, um die Infrastrukturkonfiguration auszuwählen, die Sie löschen möchten.
3. Wählen Sie oben im Bereich Infrastrukturkonfigurationen die Option Löschen aus.

4. Um den Löschvorgang zu bestätigen, geben Sie den Text `Delete` in das Feld ein und wählen Sie Löschen.

Löschen Sie die Verteilungseinstellungen

1. Um eine Liste der unter Ihrem Konto erstellten Verteilungseinstellungen anzuzeigen, wählen Sie im Navigationsbereich Verteilungseinstellungen aus.
2. Aktivieren Sie das Kontrollkästchen neben dem Namen der Konfiguration, um die Verteilungseinstellungen auszuwählen, die Sie für dieses Tutorial erstellt haben.
3. Wählen Sie oben im Bereich „Verteilungseinstellungen“ die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie den Text `Delete` in das Feld ein und wählen Sie Löschen.

Lösche das Bild

Gehen Sie wie folgt vor, um sicherzustellen, dass Sie alle Bilder gelöscht haben, die in der Tutorial-Pipeline erstellt wurden. In diesem Tutorial wird wahrscheinlich kein Image erstellt, es sei denn, es ist genügend Zeit vergangen, seit Sie Ihre Pipeline erstellt haben, sodass sie gemäß dem Build-Zeitplan ausgeführt wird.

1. Um eine Liste der unter Ihrem Konto erstellten Images anzuzeigen, wählen Sie im Navigationsbereich Images aus.
2. Wählen Sie die Image-Version für das Bild aus, das Sie entfernen möchten. Dadurch wird die Seite mit den Image-Build-Versionen geöffnet.
3. Aktivieren Sie das Kontrollkästchen neben der Version für jedes Image, das Sie löschen möchten. Sie können mehrere Image-Versionen gleichzeitig auswählen.
4. Wählen Sie oben im Bereich Image-Build-Versionen die Option Version löschen aus.
5. Um den Löschvorgang zu bestätigen, geben Sie den Text `Delete` in das Feld ein und wählen Sie Löschen.

Erstellen Sie eine Container-Image-Pipeline mit dem EC2 Image Builder Builder-Konsolenassistenten

Dieses Tutorial führt Sie durch die Erstellung einer automatisierten Pipeline zum Erstellen und Verwalten eines benutzerdefinierten EC2 Image Builder Builder-Docker-Images mithilfe des

Konsolenassistenten Create Image Pipeline. Damit Sie die Schritte effizient durchführen können, werden Standardeinstellungen verwendet, sofern sie verfügbar sind, und optionale Abschnitte werden übersprungen.

Workflow für die Image-Pipeline erstellen

- Schritt 1: Geben Sie die Pipeline-Details an
- Schritt 2: Wählen Sie ein Rezept
- Schritt 3: Definieren Sie die Infrastrukturkonfiguration — optional
- Schritt 4: Definieren Sie die Verteilungseinstellungen — optional
- Schritt 5: Prüfen
- Schritt 6: Bereinigen

Schritt 1: Geben Sie die Pipeline-Details an

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Um mit der Erstellung Ihrer Pipeline zu beginnen, wählen Sie Create Image Pipeline aus.
3. Geben Sie im Abschnitt Allgemein Ihren Pipeline-Namen ein (erforderlich).
4. Im Abschnitt Zeitplan erstellen können Sie die Standardeinstellungen für die Zeitplanoptionen beibehalten. Beachten Sie, dass die für den Standardzeitplan angezeigte Zeitzone die koordinierte Weltzeit (UTC) ist. Weitere Informationen zur UTC-Zeit und zur Ermittlung des Zeitversatzes für Ihre Zeitzone finden Sie unter Abkürzungen für Zeitzonen — Weltweite Liste.

   Wählen Sie für die Einstellungen für Abhängigkeitsupdates die Option Pipeline zur geplanten Zeit ausführen, falls es Abhängigkeitsupdates gibt. Diese Einstellung veranlasst Ihre Pipeline, vor dem Start des Builds nach Updates zu suchen. Wenn keine Updates vorhanden sind, wird der geplante Pipeline-Build übersprungen.

   > **Note**
   >
   > Um sicherzustellen, dass Ihre Pipeline Abhängigkeitsupdates und Builds wie erwartet erkennt, müssen Sie die semantische Versionierung (x.x.x) für Ihr Basis-Image und Ihre Komponenten verwenden. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung

5. Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 2: Wählen Sie ein Rezept

1. Image Builder ist standardmäßig auf Bestehendes Rezept verwenden im Abschnitt Rezept eingestellt. Wählen Sie zum ersten Mal die Option Neues Rezept erstellen.
2. Wählen Sie im Abschnitt Image-Typ die Option Docker-Image, um eine Container-Pipeline zu erstellen, die ein Docker-Image erzeugt und es an Amazon ECR-Repositorys in Zielregionen verteilt.
3. Geben Sie im Abschnitt Allgemein die folgenden erforderlichen Felder ein:

   - Name — Ihr Rezeptname
   - Version — Ihre Rezeptversion (verwenden Sie das Format <major>. <minor>. <patch>, wobei Major, Minor und Patch ganzzahlige Werte sind). Neue Rezepte beginnen in der Regel mit`1.0.0`.
4. Behalten Sie im Abschnitt Quell-Image die Standardwerte für Select image, Image Operating System (OS) und Image-Ursprung bei. Dies führt zu einer Liste von Amazon Linux 2-Container-Images, die von Amazon verwaltet werden und aus denen Sie für Ihr Basis-Image auswählen können.

   a. Wählen Sie aus der Drop-down-Liste mit dem Imagenamen ein Bild aus.

   b. Behalten Sie die Standardeinstellungen für die automatische Versionierung bei (verwenden Sie die neueste verfügbare Betriebssystemversion).

   > **Note**
   >
   > Diese Einstellung stellt sicher, dass Ihre Pipeline die semantische Versionierung für das Basis-Image verwendet, um Abhängigkeitsupdates für automatisch geplante Jobs zu erkennen. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung
5. Im Abschnitt Komponenten müssen Sie mindestens eine Build-Komponente auswählen.

   Im Bereich Komponenten erstellen — Amazon Linux können Sie die auf der Seite aufgelisteten Komponenten durchsuchen. Verwenden Sie die Seitennummerierung in der oberen rechten Ecke, um durch zusätzliche Komponenten zu navigieren, die für Ihr Basis-Image-Betriebssystem verfügbar sind. Sie können auch nach bestimmten Komponenten suchen oder mithilfe des Komponenten-Managers Ihre eigene Build-Komponente erstellen.

Wählen Sie für dieses Tutorial wie folgt eine Komponente aus, die Linux mit den neuesten Sicherheitsupdates aktualisiert:

a. Filtern Sie die Ergebnisse, indem Sie das Wort `update` in die Suchleiste eingeben, die sich oben im Bedienfeld befindet.

b. Aktivieren Sie das Kontrollkästchen für die `update-linux` Build-Komponente.

c. Scrollen Sie nach unten und wählen Sie in der oberen rechten Ecke der Liste Ausgewählte Komponenten die Option Alle erweitern aus.

d. Behalten Sie die Standardeinstellung für die Versionierungsoptionen bei (Verwenden Sie die neueste verfügbare Komponentenversion).

> **Note**
>
> Diese Einstellung stellt sicher, dass Ihre Pipeline die semantische Versionierung für die ausgewählte Komponente verwendet, um Abhängigkeitsupdates für automatisch geplante Jobs zu erkennen. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung

Wenn Sie eine Komponente mit Eingabeparametern ausgewählt hätten, würden Sie die Parameter auch in diesem Bereich sehen. Parameter werden in diesem Tutorial nicht behandelt. Weitere Informationen zur Verwendung von Eingabeparametern in Ihren Komponenten und deren Einstellung in Ihren Rezepten finden Sie unterAWSTOE Komponentenparameter mit EC2 Image Builder verwalten.

Komponenten neu anordnen (optional)

Wenn Sie mehr als eine Komponente ausgewählt haben, die in Ihr Image aufgenommen werden soll, können Sie die drag-and-drop Aktion verwenden, um sie in der Reihenfolge anzuordnen, in der sie während des Erstellungsprozesses ausgeführt werden sollen.

> **Note**
>
> CIS-Härtungskomponenten folgen nicht den Standardregeln für die Reihenfolge der Komponenten in den Image Builder Builder-Rezepten. Die CIS-Härtungskomponenten

werden immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests anhand Ihres Ausgabebilds ausgeführt werden.

1. Scrollen Sie zurück zur Liste der verfügbaren Komponenten.
2. Aktivieren Sie das Kontrollkästchen für die `update-linux-kernel-mainline` Build-Komponente (oder eine andere Komponente Ihrer Wahl).
3. Scrollen Sie nach unten zur Liste Ausgewählte Komponenten, um zu sehen, dass es mindestens zwei Ergebnisse gibt.
4. Bei neu hinzugefügten Komponenten wurde die Versionierung möglicherweise nicht erweitert. Um die Versionierungsoptionen zu erweitern, können Sie entweder den Pfeil neben Versionierungsoptionen auswählen oder den Schalter Alle erweitern aus- und einschalten, um die Versionierung für alle ausgewählten Komponenten zu erweitern.
5. Wählen Sie eine der Komponenten aus und ziehen Sie sie nach oben oder unten, um die Reihenfolge zu ändern, in der die Komponenten ausgeführt werden.
6. Um die `update-linux-kernel-mainline` Komponente zu entfernen, wählen Sie in der oberen rechten Ecke des Komponentenfeldes eine Option `X` aus.
7. Wiederholen Sie den vorherigen Schritt, um alle anderen Komponenten zu entfernen, die Sie möglicherweise hinzugefügt haben, sodass nur die `update-linux` Komponente ausgewählt bleibt.

6. Wählen Sie im Abschnitt Dockerfile-Vorlage die Option Beispiel verwenden aus. Beachten Sie im Inhaltsbereich die Kontextvariablen, in denen Image Builder Build-Informationen oder Skripts platziert, die auf Ihrem Container-Image-Rezept basieren.

   Standardmäßig verwendet Image Builder die folgenden Kontextvariablen in Ihrem Dockerfile.

   ParentImage (erforderlich)

   Bei der Erstellung wird diese Variable in das Basis-Image für Ihr Rezept aufgelöst.

   Beispiel:

   ```
   FROM
   {{{ imagebuilder:parentImage }}}
   ```

Umgebungen (erforderlich, wenn Komponenten angegeben sind)

Diese Variable wird in ein Skript aufgelöst, das Komponenten ausführt.

Beispiel:

```
{{{ imagebuilder:environments }}}
```

Komponenten (optional)

Image Builder löst Build- und Testkomponentenskripts für die Komponenten auf, die das Container-Rezept enthält. Diese Variable kann an einer beliebigen Stelle im Dockerfile hinter der Umgebungsvariablen platziert werden.

Beispiel:

```
{{{ imagebuilder:components }}}
```

7. Geben Sie im Abschnitt Ziel-Repository den Namen des Amazon ECR-Repositorys an, das Sie als Voraussetzung für dieses Tutorial erstellt haben. Dieses Repository wird als Standardeinstellung für die Verteilungskonfiguration in der Region verwendet, in der die Pipeline ausgeführt wird (Region 1).

   > **Note**
   >
   > Das Ziel-Repository muss vor der Verteilung in Amazon ECR für alle Zielregionen vorhanden sein.

8. Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 3: Definieren Sie die Infrastrukturkonfiguration — optional

Image Builder startet EC2-Instances in Ihrem Konto, um Images anzupassen und Validierungstests durchzuführen. In den Infrastrukturkonfigurationseinstellungen werden die Infrastrukturdetails für die Instances angegeben, die AWS-Konto während des Build-Prozesses in Ihrem ausgeführt werden.

Im Abschnitt Infrastrukturkonfiguration sind die Konfigurationsoptionen standardmäßig auf eingestelltCreate infrastructure configuration using service defaults. Dadurch werden eine IAM-Rolle und ein zugehöriges Instanzprofil erstellt, die von Build-Instances

zur Konfiguration Ihrer Container-Images verwendet werden. Sie können auch Ihre eigene benutzerdefinierte Infrastrukturkonfiguration erstellen oder Einstellungen verwenden, die Sie bereits erstellt haben. Weitere Informationen zu den Einstellungen für die Infrastrukturkonfiguration finden Sie CreateInfrastructureConfigurationin der EC2 Image Builder API-Referenz.

Für dieses Tutorial verwenden wir die Standardeinstellungen.

- Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 4: Definieren Sie die Verteilungseinstellungen — optional

Die Verteilungseinstellungen bestehen aus den Zielregionen und dem Namen des Amazon ECR-Ziel-Repositorys. Docker-Ausgabe-Images werden im benannten Amazon ECR-Repository in jeder Region bereitgestellt.

Im Abschnitt Verteilungseinstellungen sind die Konfigurationsoptionen standardmäßig auf eingestellt. `Create distribution settings using service defaults` Diese Option verteilt das ausgegebene Docker-Image an das Amazon ECR-Repository, das in Ihrem Container-Rezept für die Region angegeben ist, in der Ihre Pipeline läuft (Region 1). Wenn Sie möchten`Create new distribution settings`, können Sie das ECR-Repository für die aktuelle Region überschreiben und weitere Regionen für die Verteilung hinzufügen.

Für dieses Tutorial verwenden wir die Standardeinstellungen.

- Wählen Sie Weiter, um mit dem nächsten Schritt fortzufahren.

Schritt 5: Prüfen

Im Abschnitt Überprüfen werden alle Einstellungen angezeigt, die Sie konfiguriert haben. Um Informationen in einem bestimmten Abschnitt zu bearbeiten, wählen Sie die Schaltfläche Bearbeiten in der oberen rechten Ecke des Schrittabschnitts. Wenn Sie beispielsweise Ihren Pipeline-Namen ändern möchten, wählen Sie die Schaltfläche Bearbeiten in der oberen rechten Ecke des Abschnitts Schritt 1: Pipeline-Details.

1. Wenn Sie Ihre Einstellungen überprüft haben, wählen Sie Pipeline erstellen, um Ihre Pipeline zu erstellen.
2. Sie können Erfolgs- oder Fehlschlagsmeldungen oben auf der Seite sehen, während Ihre Ressourcen für die Verteilungseinstellungen, die Infrastrukturkonfiguration, Ihr neues Rezept und

die Pipeline erstellt werden. Um Details zu einer Ressource, einschließlich der Ressourcen-ID, anzuzeigen, wählen Sie Details anzeigen.

3. Nachdem Sie die Details für eine Ressource angezeigt haben, können Sie Details zu anderen Ressourcen anzeigen, indem Sie den Ressourcentyp im Navigationsbereich auswählen. Um beispielsweise Details für Ihre neue Pipeline anzuzeigen, wählen Sie im Navigationsbereich Image-Pipelines aus. Wenn Ihr Build erfolgreich war, wird Ihre neue Pipeline in der Liste der Image-Pipelines angezeigt.

Schritt 6: Bereinigen

Ihre Image Builder Builder-Umgebung muss, genau wie Ihr Zuhause, regelmäßig gewartet werden, damit Sie das finden, was Sie benötigen, und Ihre Aufgaben erledigen können, ohne sich durch Unordnung zu wühlen. Stellen Sie sicher, dass Sie die temporären Ressourcen, die Sie zu Testzwecken erstellt haben, regelmäßig bereinigen. Andernfalls könnten Sie diese Ressourcen vergessen und sich später nicht mehr daran erinnern, wofür sie verwendet wurden. Bis dahin ist möglicherweise nicht klar, ob Sie sie sicher loswerden können.

> **Tip**
>
> Um Abhängigkeitsfehler beim Löschen von Ressourcen zu vermeiden, stellen Sie sicher, dass Sie Ihre Ressourcen in der folgenden Reihenfolge löschen:
>
> 1. Image-Pipeline
> 2. Bildrezept
> 3. Alle verbleibenden Ressourcen

Gehen Sie folgendermaßen vor, um die Ressourcen zu bereinigen, die Sie für dieses Tutorial erstellt haben:

Löschen Sie die Pipeline

1. Um eine Liste der Build-Pipelines anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Image-Pipelines aus.
2. Aktivieren Sie das Kontrollkästchen neben dem Namen der Pipeline, um die Pipeline auszuwählen, die Sie löschen möchten.
3. Wählen Sie oben im Bereich Image-Pipelines im Menü Aktionen die Option Löschen aus.

4. Um den Löschvorgang zu bestätigen, geben Sie den Text `Delete` in das Feld ein und wählen Sie „Löschen“.

Löschen Sie das Container-Rezept

1. Um eine Liste der unter Ihrem Konto erstellten Container-Rezepte zu sehen, wählen Sie im Navigationsbereich Container-Rezepte aus.
2. Aktivieren Sie das Kontrollkästchen neben Rezeptname, um das Rezept auszuwählen, das Sie löschen möchten.
3. Wählen Sie oben im Bereich „Container-Rezepte“ im Menü „Aktionen“ die Option „Rezept löschen“ aus.
4. Um den Löschvorgang zu bestätigen, geben Sie den `Delete` Text in das Feld ein und wählen Sie „Löschen“.

Löschen Sie die Infrastrukturkonfiguration

1. Um eine Liste der unter Ihrem Konto erstellten Infrastrukturkonfigurationen anzuzeigen, wählen Sie im Navigationsbereich Infrastrukturkonfiguration aus.
2. Aktivieren Sie das Kontrollkästchen neben Konfigurationsname, um die Infrastrukturkonfiguration auszuwählen, die Sie löschen möchten.
3. Wählen Sie oben im Bereich Infrastrukturkonfigurationen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie den Text `Delete` in das Feld ein und wählen Sie Löschen.

Löschen Sie die Verteilungseinstellungen

1. Um eine Liste der unter Ihrem Konto erstellten Verteilungseinstellungen anzuzeigen, wählen Sie im Navigationsbereich die Option Verteilungseinstellungen aus.
2. Aktivieren Sie das Kontrollkästchen neben dem Namen der Konfiguration, um die Verteilungseinstellungen auszuwählen, die Sie für dieses Tutorial erstellt haben.
3. Wählen Sie oben im Bereich „Verteilungseinstellungen“ die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie den Text `Delete` in das Feld ein und wählen Sie Löschen.

Lösche das Bild

Gehen Sie wie folgt vor, um sicherzustellen, dass Sie alle Bilder gelöscht haben, die in der Tutorial-Pipeline erstellt wurden. In diesem Tutorial wird wahrscheinlich kein Image erstellt, es sei denn, es ist genügend Zeit vergangen, seit Sie Ihre Pipeline erstellt haben, sodass sie gemäß dem Build-Zeitplan ausgeführt wird.

1. Um eine Liste der unter Ihrem Konto erstellten Images anzuzeigen, wählen Sie im Navigationsbereich Images aus.
2. Wählen Sie die Image-Version für das Bild aus, das Sie entfernen möchten. Dadurch wird die Seite mit den Image-Build-Versionen geöffnet.
3. Aktivieren Sie das Kontrollkästchen neben der Version für jedes Image, das Sie löschen möchten. Sie können mehr als eine Image-Version gleichzeitig auswählen.
4. Wählen Sie oben im Bereich Image-Build-Versionen die Option Version löschen aus.
5. Um den Löschvorgang zu bestätigen, geben Sie den Text `Delete` in das Feld ein und wählen Sie Löschen.

AWS Task Orchestrator and Executor Komponentenmanager

EC2 Image Builder verwendet die AWS Task Orchestrator and Executor (AWSTOE) -Anwendung, um komplexe Workflows zu orchestrieren, Systemkonfigurationen zu ändern und Ihre Systeme zu testen, ohne Code schreiben zu müssen. Diese Anwendung verwaltet und führt Komponenten aus, die ihr deklaratives Dokumentschema implementieren.

Da es sich um eine eigenständige Anwendung handelt, ist kein zusätzliches Server-Setup erforderlich. Es kann auf jeder Cloud-Infrastruktur und vor Ort ausgeführt werden.

Inhalt

AWSTOE lädt herunter

Wählen Sie zur Installation AWSTOE den Download-Link für Ihre Architektur und Plattform. Wenn Sie eine Verbindung zu einem VPC-Endpunkt für Ihren Service herstellen (z. B. Image Builder), muss diesem eine benutzerdefinierte Endpunktrichtlinie zugewiesen sein, die den Zugriff auf den S3-Bucket für AWSTOE Downloads beinhaltet. Andernfalls können Ihre Build- und Test-Instances das Bootstrap-Skript (`bootstrap.sh`) nicht herunterladen und die AWSTOE Anwendung nicht installieren. Weitere Informationen finden Sie unter Erstellen Sie eine VPC-Endpunktrichtlinie für Image Builder.

> **Important**
>
> AWS stellt die Unterstützung für die TLS-Versionen 1.0 und 1.1 schrittweise ein. Um auf den S3-Bucket für AWSTOE Downloads zuzugreifen, muss Ihre Client-Software TLS Version 1.2 oder höher verwenden. Weitere Informationen finden Sie in diesem AWS Sicherheits-Blogbeitrag.

Architektur	Plattform	Download-Link	Beispiel
386	AL 2 und 2023 RHEL 7 und 8 Ubuntu 16.04, 18.04, 20.04 und 22.04 CentOS 7 und 8 SUSE 12 und 15	`https://awstoe- <region>.s3.com/latest/linux/386/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/386/awstoe
AMD64	Windows Server 2012 R2, 2016, 2019 und 2022	`https://awstoe- <region>.s3.com/latest/windows/amd64/awstoe.exe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/windows/amd64/awstoe.exe
AMD64	AL 2 und 2023 RHEL 7 und 8 Ubuntu 16.04, 18.04, 20.04 und 22.04 CentOS 7 und 8 CentOS Stream 8 SUSE 12 und 15	`https://awstoe- <region>.s3.com/latest/linux/amd64/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/amd64/awstoe

Architektur	Plattform	Download-Link	Beispiel
ARM64	AL 2 und 2023 RHEL 7 und 8 Ubuntu 16.04, 18.04, 20.04 und 22.04 CentOS 7 und 8 CentOS Stream 8 SUSE 12 und 15	`https://awstoe- <region>.s3.s3.com/latest/linux/arm64/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/arm64/awstoe

Unterstützte Regionen

AWSTOE wird in den folgenden Regionen als eigenständige Anwendung unterstützt.

AWS-Region Name	AWS-Region
USA Ost (Ohio)	us-east-2
USA Ost (Nord-Virginia)	us-east-1
AWS GovCloud (US-Ost)	us-gov-east-1
AWS GovCloud (US-West)	us-gov-west-1
USA West (Nordkalifornien)	us-west-1
USA West (Oregon)	us-west-2
Afrika (Kapstadt)	af-south-1
Asien-Pazifik (Hongkong)	ap-east-1
Asien-Pazifik (Osaka)	ap-northeast-3
Asien-Pazifik (Seoul)	ap-northeast-2

AWS-Region Name	AWS-Region
Asien-Pazifik (Mumbai)	ap-south-1
Asien-Pazifik (Hyderabad)	ap-south-2
Asien-Pazifik (Singapur)	ap-southeast-1
Asien-Pazifik (Sydney)	ap-southeast-2
Asien-Pazifik (Jakarta)	ap-southeast-3
Asien-Pazifik (Tokio)	ap-northeast-1
Kanada (Zentral)	ca-central-1
Europa (Frankfurt)	eu-central-1
Europa (Zürich)	eu-central-2
Europa (Stockholm)	eu-north-1
Europa (Mailand)	eu-south-1
Europa (Spanien)	eu-south-2
Europa (Irland)	eu-west-1
Europa (London)	eu-west-2
Europa (Paris)	eu-west-3
Israel (Tel Aviv)	il-central-1
Naher Osten (VAE)	me-central-1
Naher Osten (Bahrain)	me-south-1
Südamerika (São Paulo)	sa-east-1
China (Beijing)	cn-north-1

AWS-Region Name	AWS-Region
China (Ningxia)	cn-northwest-1

Fangen Sie an mit AWSTOE

Die AWS Task Orchestrator and Executor (AWSTOE) -Anwendung ist eine eigenständige Anwendung, die Befehle innerhalb eines Komponentendefinitions-Frameworks erstellt, validiert und ausführt. AWS Dienste können verwendet werden, AWSTOE um Workflows zu orchestrieren, Software zu installieren, Systemkonfigurationen zu ändern und Image-Builds zu testen.

Gehen Sie wie folgt vor, um die AWSTOE Anwendung zu installieren und zum ersten Mal zu verwenden.

Überprüfen Sie die Signatur des AWSTOE Installationsdownloads

In diesem Abschnitt wird das empfohlene Verfahren zur Überprüfung der Gültigkeit des Installationsdownloads für AWSTOE Linux- und Windows-Betriebssysteme beschrieben.

Themen

Überprüfen Sie die Signatur des AWSTOE Installations-Downloads unter Linux

In diesem Thema wird das empfohlene Verfahren zur Überprüfung der Gültigkeit des Installationsdownloads für Linux-basierte AWSTOE Betriebssysteme beschrieben.

Wenn Sie eine Anwendung aus dem Internet herunterladen, empfehlen wir, dass Sie die Identität des Softwareherausgebers authentifizieren. Stellen Sie außerdem sicher, dass die Anwendung seit ihrer Veröffentlichung nicht verändert oder beschädigt wurde. Dies schützt Sie davor, eine Version der Anwendung zu installieren, die einen Virus oder einen anderen bösartigen Code enthält.

Wenn Sie nach Ausführung der Schritte in diesem Thema feststellen, dass die Software für geändert oder beschädigt AWSTOE ist, führen Sie die Installationsdatei nicht aus. Wenden Sie sich stattdessen an AWS Support Weitere Informationen zu Ihren Supportoptionen finden Sie unter AWS Support.

AWSTOE Dateien für Linux-basierte Betriebssysteme werden mit GnuPG einer Open-Source-Implementierung des Pretty Good Privacy (OpenPGP) -Standards für sichere digitale Signaturen signiert. GnuPG(auch bekannt alsGPG) ermöglicht Authentifizierung und Integritätsprüfung mithilfe einer digitalen Signatur. Amazon EC2 veröffentlicht einen öffentlichen Schlüssel und Signaturen, mit denen Sie die heruntergeladenen Amazon EC2 EC2-CLI-Tools überprüfen können. Weitere Informationen zu PGP und GnuPG (GPG) finden Sie unter http://www.gnupg.org.

Der erste Schritt besteht darin, eine Vertrauensstellung mit dem Software-Publisher zu schaffen. Laden Sie den öffentlichen Schlüssel des Softwareherausgebers herunter, überprüfen Sie, ob der Besitzer des öffentlichen Schlüssels derjenige ist, der er behauptet zu sein, und fügen Sie dann den öffentlichen Schlüssel zu Ihrem Schlüsselbund hinzu. Ihr Schlüsselbund ist eine Sammlung von bekannten öffentlichen Schlüsseln. Nachdem Sie die Echtheit des öffentlichen Schlüssels überprüft haben, können Sie ihn verwenden, um die Signatur der Anwendung zu überprüfen.

Themen

Installieren der GPG-Tools

Wenn Sie das Betriebssystem Linux oder Unix verwenden, sind die GPG-Tools wahrscheinlich bereits installiert. Um zu testen, ob die Tools auf Ihrem System installiert sind, geben Sie gpg in einer Eingabeaufforderung ein. Wenn die GPG-Tools installiert sind, sehen Sie eine Eingabeaufforderung. Wenn die GPG-Tools nicht installiert sind, wird eine Fehlermeldung angezeigt, die besagt, dass der Befehl nicht gefunden werden kann. Sie können das GnuPG-Paket von einem Repository aus installieren.

So installieren Sie GPG-Tools auf Debian-basiertem Linux

- Führen Sie von einem Terminal folgenden Befehl aus: apt-get install gnupg.

So installieren Sie GPG-Tools unter Red-Hat-basiertem Linux

- Führen Sie von einem Terminal folgenden Befehl aus: yum install gnupg.

Authentifizieren und Importieren des öffentlichen Schlüssels

Der nächste Schritt in diesem Prozess besteht darin, den AWSTOE öffentlichen Schlüssel zu authentifizieren und ihn als vertrauenswürdigen Schlüssel zu Ihrem GPG Schlüsselbund hinzuzufügen.

Um den öffentlichen Schlüssel zu authentifizieren und zu importieren AWSTOE

1. Besorgen Sie sich ein Exemplar unseres öffentlichen GPG-Schlüssels, indem Sie einen der folgenden Schritte ausführen:

   - Laden Sie den Schlüssel von https://awstoe - **<region>** .s3 herunter. **<region>**.amazonaws.com/assets/awstoe.gpg. z. B. https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/assets/awstoe.gpg.
   - Kopieren Sie den Schlüssel aus dem folgenden Text und fügen Sie ihn in eine Datei namens `awstoe.gpg` ein. Vergewissern Sie sich, alles Folgende einzubeziehen:

   ```
   -----BEGIN PGP PUBLIC KEY BLOCK-----
   Version: GnuPG v2

   mQENBF8UqwsBCACdiRF2bkZYaFSDPFC+LIkWLwFvtUCRwAHtD8KIwTJ6LVn3fHAU
   GhuK0ZH9mRrqRT2bq/xJjGsnF9VqTj2AJqndGJdDjz75YCZYM+ocZ+r5HSJaeW9i
   S5dykHj7Txti2zHe0G5+W0v7v5bPi2sPHsN7XWQ7+G2AMEPTz8PjxY//I0DvMQns
   Sle3l9hz6wCClz1l9LbBzTyHfSm5ucTXvNe88XX5Gmt37OCDM7vfli0Ctv8WFoLN
   6jbxuA/sV71yIkPm9IYp3+GvaKeT870+sn8/JOOKE/U4sJV1ppbqmuUzDfhrZUaw
   8eW8IN9A1FTIuWiZED/5L83UZuQs1S7s2PjlABEBAAG0GkFXU1RPRSA8YXdzdG9l
   QGFtYXpvbi5jb20+iQE5BBMBCAAjBQJfFKsLAhsDBwsJCAcDAgEGFQgCCQoLBBYC
   AwECHgECF4AACgkQ3r3BVvWuvFJGiwf9EVmrBR77+Qe/DUeXZJYoaFr7If/fVDZl
   6V3TC6p0J0Veme7uXleRUTFOjzbh+7e5sDX19HrnPquzCnzfMiqbp4lSoeUuNdOf
   FcpuTCQH+M+sIEIgPno4PLl0Uj2uE1o++mxmonBl/Krk+hly8hB2L/9n/vW3L7BN
   OMb1Ll9PmgGPbWipcT8KRdz4SUex9TXGYzjlWb3jU3uXetdaQY1M3kVKE1siRsRN
   YYDtpcjmwbhjpu4xm19aFqNoAHCDctEsXJA/mkU3erwIRocPyjAZE2dnlkL9ZkFZ
   z9DQkcIarbCnybDM5lemBbdhXJ6hezJE/b17VA0t1fY04MoEkn6oJg==
   =oyze
   -----END PGP PUBLIC KEY BLOCK-----
   ```

2. Verwenden Sie an einer Befehlszeile in dem Verzeichnis, in dem Sie awstoe.gpg gespeichert haben, den folgenden Befehl, um den öffentlichen Schlüssel in Ihren Schlüsselbund zu importieren. AWSTOE

   ```
   gpg --import awstoe.gpg
   ```

Der Befehl gibt Ergebnisse wie die folgenden zurück:

```
gpg: key F5AEBC52: public key "AWSTOE <awstoe@amazon.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
```

Notieren Sie sich den Schlüsselwert. Sie brauchen ihn im nächsten Schritt. Im vorangegangenen Beispiel ist der Schlüsselwert `F5AEBC52`.

3. Überprüfen Sie den Fingerabdruck, indem Sie den folgenden Befehl ausführen und Schlüssel-Wert durch den Wert des vorherigen Schritts ersetzen:

```
gpg --fingerprint key-value
```

Dieser Befehl gibt Ergebnisse wie die folgenden zurück:

```
pub   2048R/F5AEBC52 2020-07-19
      Key fingerprint = F6DD E01C 869F D639 15E5  5742 DEBD C156 F5AE BC52
uid       [ unknown] AWSTOE <awstoe@amazon.com>
```

Zusätzlich sollte der Fingerabdruck-String identisch mit `F6DD E01C 869F D639 15E5 5742 DEBD C156 F5AE BC52` sein, wie im voranstehenden Beispiel angezeigt. Vergleichen Sie den Schlüssel-Fingerabdruck mit demjenigen, der auf dieser Seite veröffentlicht ist.
Sie sollten übereinstimmen. Wenn sie nicht übereinstimmen, installieren Sie das AWSTOE Installationsskript nicht und setzen Sie sich mit uns in Verbindung. AWS Support

Verifizieren der Signatur des Pakets

Nachdem Sie die GPG-Tools installiert, den öffentlichen Schlüssel für AWSTOE authentifiziert und importiert und überprüfen haben, dass der öffentliche Schlüssel vertrauenswürdig ist, sind Sie bereit, die Signatur des Installationsskripts zu überprüfen.

So überprüfen Sie die Signatur des -Installationsskripts

1. Führen Sie an der Befehlszeile den folgenden Befehl aus, um die Binärdatei der Anwendung herunterzuladen:

```
curl -O https://awstoe-<region>.s3.<region>.amazonaws.com/latest/
linux/<architecture>/awstoe
```

Beispielsweise:

```
curl -O https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/amd64/
awstoe
```

Unterstützte Werte für **`architecture`** können `amd64` `386`, und sein `arm64`.

2. Führen Sie an einer Befehlszeile den folgenden Befehl aus, um die Signaturdatei für die entsprechende Anwendungsbinärdatei aus demselben S3-Schlüsselpräfixpfad herunterzuladen:

```
curl -O https://awstoe-<region>.s3.<region>.amazonaws.com/latest/
linux/<architecture>/awstoe.sig
```

Beispielsweise:

```
curl -O https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/amd64/
awstoe.sig
```

Unterstützte Werte für **`architecture`** können `amd64` `386`, und sein `arm64`.

3. Überprüfen Sie die Signatur, indem Sie in der Befehlszeile in dem Verzeichnis, in dem Sie die Datei gespeichert haben `awstoe.sig`, den folgenden AWSTOE Befehl ausführen. Beide Dateien müssen vorhanden sein.

```
gpg --verify ./awstoe.sig ~/awstoe
```

Die Ausgabe sollte wie folgt aussehen:

```
gpg: Signature made Mon 20 Jul 2020 08:54:55 AM IST using RSA key ID F5AEBC52
gpg: Good signature from "AWSTOE awstoe@amazon.com" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: F6DD E01C 869F D639 15E5 5742 DEBD C156 F5AE BC52
```

Wenn die Ausgabe den Begriff `Good signature from "AWSTOE <awstoe@amazon.com>"` enthält, bedeutet dies, dass die Signatur erfolgreich überprüft wurde und Sie mit der Ausführung des AWSTOE -Installationsskripts fortfahren können.

Wenn die Ausgabe die Bezeichnung `BAD signature`enthält, überprüfen Sie, ob Sie das Verfahren korrekt durchgeführt haben. Wenn Sie weiterhin diese Antwort erhalten, führen Sie die Installationsdatei, die Sie zuvor heruntergeladen haben, nicht aus und wenden Sie sich an AWS Support.

Im Folgenden finden Sie Details zu den Warnungen, die möglicherweise angezeigt werden:

- WARNUNG: Dieser Schlüssel ist nicht mit einer vertrauenswürdigen Signatur zertifiziert! Es gibt keinen Hinweis darauf, dass die Signatur dem Besitzer gehört. Idealerweise würden Sie ein AWS Büro aufsuchen und den Schlüssel persönlich entgegennehmen. Sie würden es jedoch höchstwahrscheinlich von einer Website herunterladen. In diesem Fall handelt es sich bei der Website um eine AWS Website.
- gpg: Keine endgültig vertrauenswürdigen Schlüssel gefunden Das bedeutet, dass Sie oder andere Personen, denen Sie vertrauen, dem spezifischen Schlüssel „letztendlich nicht vertrauen“.

Weitere Informationen finden Sie unter http://www.gnupg.org.

Überprüfen Sie die Signatur des AWSTOE Installationsdownloads unter Windows

In diesem Thema wird das empfohlene Verfahren zur Überprüfung der Gültigkeit der Installationsdatei für die AWS Task Orchestrator and Executor Anwendung auf Windows-Betriebssystemen beschrieben.

Wenn Sie eine Anwendung aus dem Internet herunterladen, empfehlen wir Ihnen, die Identität des Softwareverlegers zu authentifizieren und zu überprüfen, ob die Anwendung seit ihrer Veröffentlichung nicht verändert oder beschädigt wurde. Dies schützt Sie davor, eine Version der Anwendung zu installieren, die einen Virus oder einen anderen bösartigen Code enthält.

Wenn Sie nach Ausführung der Schritte in diesem Thema feststellen, dass die Software für die AWSTOE Anwendung verändert oder beschädigt ist, führen Sie die Installationsdatei nicht aus. Wenden Sie sich stattdessen an AWS Support.

Um die Gültigkeit der heruntergeladenen awstoe-Binärdatei auf Windows-basierten Betriebssystemen zu überprüfen, stellen Sie sicher, dass der Fingerabdruck des zugehörigen Amazon Services LLC-Unterzeichnerzertifikats diesem Wert entspricht:

F8 83 11 EE F0 4A A2 91 E3 79 21 BA 6B FC AF F8 19 92 12 D7

> **Note**
>
> Während des Rollout-Fensters für eine neue Binärdatei stimmt Ihr Unterzeichnerzertifikat möglicherweise nicht mit dem neuen Fingerabdruck überein. Wenn Ihr Unterzeichnerzertifikat nicht übereinstimmt, überprüfen Sie, ob der Fingerabdruckwert wie folgt lautet:
> 5B 77 F4 F0 C3 7A 8B 89 D9 A7 8F 54 B6 85 11 CE 9E A3 BF 17

Um diesen Wert zu überprüfen, gehen Sie wie folgt vor:

1. Klicken Sie mit der rechten Maustaste auf die heruntergeladen `awstoe.exe`, und öffnen Sie das Eigenschaften-Fenster.
2. Wählen Sie die Registerkarte Digital Signatures aus.
3. Wählen Sie in der Signature List die Option Amazon Services LLC und dann Details aus.
4. Falls die Registerkarte General nicht bereits ausgewählt ist, klicken Sie darauf und dann auf View Certificate.
5. Wählen Sie die Registerkarte Details aus, und anschließend die Option All (Alle) in der Dropdown-Liste Show (Zeigen), wenn diese nicht bereits ausgewählt ist.
6. Scrollen Sie nach unten zum Feld Thumbprint und wählen Sie Thumbprint aus. Der gesamte Thumbprint-Wert wird im unteren Fenster angezeigt.

   - Wenn der Thumbprint-Wert im unteren Fenster mit folgendem Wert identisch ist:

     F8 83 11 EE F0 4A A2 91 E3 79 21 BA 6B FC AF F8 19 92 12 D7

     dann ist Ihre heruntergeladene AWSTOE Binärdatei authentisch und kann sicher installiert werden.

     > **Note**
     >
     > Während des Rollout-Fensters für eine neue Binärdatei stimmt Ihr Unterzeichnerzertifikat möglicherweise nicht mit dem neuen Fingerabdruck überein.

Wenn Ihr Unterzeichnerzertifikat nicht übereinstimmt, überprüfen Sie, ob der Fingerabdruckwert wie folgt lautet:
5B 77 F4 F0 C3 7A 8B 89 D9 A7 8F 54 B6 85 11 CE 9E A3 BF 17

- Wenn der Fingerabdruckwert im unteren Detailfenster nicht mit dem vorherigen Wert identisch ist, führen Sie den Vorgang nicht aus. `awstoe.exe`

Schritte zum Einstieg

- Schritt 1: Installieren AWSTOE
- Schritt 2: Legen Sie die Anmeldeinformationen fest AWS
- Schritt 3: Komponentendokumente lokal entwickeln
- Schritt 4: AWSTOE Komponenten validieren
- Schritt 5: AWSTOE Komponenten ausführen

Schritt 1: Installieren AWSTOE

Um Komponenten lokal zu entwickeln, laden Sie die AWSTOE Anwendung herunter und installieren Sie sie.

1. Laden Sie die AWSTOE Anwendung herunter

   Wählen Sie zur Installation AWSTOE den entsprechenden Download-Link für Ihre Architektur und Plattform. Die vollständige Liste der Links zum Herunterladen von Anwendungen finden Sie unter AWSTOE lädt herunter

   > **Important**
   >
   > AWS stellt die Unterstützung für die TLS-Versionen 1.0 und 1.1 schrittweise ein. Um auf den S3-Bucket für AWSTOE Downloads zuzugreifen, muss Ihre Client-Software TLS Version 1.2 oder höher verwenden. Weitere Informationen finden Sie in diesem AWS Sicherheits-Blogbeitrag.

2. Überprüfen Sie die Signatur

   Die Schritte zur Überprüfung Ihres Downloads hängen von der Serverplattform ab, auf der Sie die AWSTOE Anwendung nach der Installation ausführen. Informationen zur Überprüfung

Ihres Downloads auf einem Linux-Server finden Sie unterÜberprüfen Sie die Signatur unter Linux. Informationen zur Überprüfung Ihres Downloads auf einem Windows-Server finden Sie unterÜberprüfen Sie die Signatur unter Windows.

> **Important**
>
> AWSTOE wird direkt von seinem Download-Speicherort aus aufgerufen. Ein separater Installationsschritt ist nicht erforderlich. Dies bedeutet auch, dass Änderungen an der lokalen Umgebung vorgenommen werden AWSTOE können.
> Um sicherzustellen, dass Sie Änderungen während der Komponentenentwicklung isolieren, empfehlen wir, für die Entwicklung und das Testen von AWSTOE Komponenten eine EC2-Instance zu verwenden.

Schritt 2: Legen Sie die Anmeldeinformationen fest AWS

AWSTOE erfordert AWS Anmeldeinformationen AWS-Services, um eine Verbindung zu anderen Geräten wie Amazon S3 und Amazon herzustellen CloudWatch, wenn Aufgaben ausgeführt werden, wie z. B.:

- AWSTOE Dokumente werden von einem vom Benutzer bereitgestellten Amazon S3-Pfad heruntergeladen.
- Laufende Module `S3Download` oder `S3Upload` Aktionsmodule.
- Streaming-Protokolle an CloudWatch, wenn aktiviert.

Wenn Sie AWSTOE auf einer EC2-Instance arbeiten, werden beim Ausführen dieselben Berechtigungen AWSTOE verwendet wie für die IAM-Rolle, die der EC2-Instance zugewiesen ist.

Weitere Informationen zu IAM-Rollen für EC2 finden Sie unter IAM-Rollen für Amazon EC2.

Die folgenden Beispiele zeigen, wie AWS Anmeldeinformationen mithilfe der Umgebungsvariablen und festgelegt werden. `AWS_ACCESS_KEY_ID` `AWS_SECRET_ACCESS_KEY`

Um diese Variablen unter Linux, macOS oder Unix festzulegen, verwenden Sie`export`.

```
$ export AWS_ACCESS_KEY_ID=your_access_key_id
```

```
$ export AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

Um diese Variablen unter Windows mit festzulegen PowerShell, verwenden Sie`$env`.

```
C:\> $env:AWS_ACCESS_KEY_ID=your_access_key_id
```

```
C:\> $env:AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

Um diese Variablen unter Windows über die Befehlszeile festzulegen, verwenden Sie`set`.

```
C:\> set AWS_ACCESS_KEY_ID=your_access_key_id
```

```
C:\> set AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

Schritt 3: Komponentendokumente lokal entwickeln

AWSTOE Komponenten werden mit Klartext-YAML-Dokumenten erstellt. Weitere Hinweise zur Dokumentensyntax finden Sie unter. Verwenden Sie Komponentendokumente in AWSTOE

Im Folgenden finden Sie Beispiele für Dokumente aus der Hello World-Komponente, die Sie verwenden können, um Ihre Dokumente lokal zu entwickeln.

`hello-world-windows.yml`.

```
name: Hello World
description: This is Hello World testing document for Windows.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host 'Hello World from the build phase.'
  - name: validate
    steps:
      - name: HelloWorldStep
```

```
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host 'Hello World from the validate phase.'
  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host 'Hello World from the test phase.'
```

`hello-world-linux.yml`.

```
name: Hello World
description: This is hello world testing document for Linux.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo 'Hello World from the build phase.'
  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo 'Hello World from the validate phase.'
  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo 'Hello World from the test phase.'
```

Schritt 4: AWSTOE Komponenten validieren

Sie können die Syntax von AWSTOE Komponenten lokal mit der AWSTOE Anwendung überprüfen. Die folgenden Beispiele zeigen den AWSTOE `validate` Anwendungsbefehl zur Validierung der Syntax einer Komponente, ohne sie auszuführen.

> **Note**
>
> Die AWSTOE Anwendung kann nur die Komponentensyntax für das aktuelle Betriebssystem überprüfen. Wenn Sie beispielsweise `awstoe.exe` unter Windows ausgeführt werden, können Sie die Syntax für ein Linux-Dokument, das das `ExecuteBash` Aktionsmodul verwendet, nicht überprüfen.

Windows

```
C:\> awstoe.exe validate --documents C:\Users\user\Documents\hello-world.yml
```

Linux

```
$ awstoe validate --documents /home/user/hello-world.yml
```

Schritt 5: AWSTOE Komponenten ausführen

Die AWSTOE Anwendung kann mithilfe des `--phases` Befehlszeilenarguments eine oder mehrere Phasen bestimmter Dokumente ausführen. Unterstützte Werte für `--phases` sind `build`, `validate`, und`test`. Mehrere Phasenwerte können als kommagetrennte Werte eingegeben werden.

Wenn Sie eine Liste von Phasen bereitstellen, führt die AWSTOE Anwendung nacheinander die angegebenen Phasen der einzelnen Dokumente aus. AWSTOE Führt beispielsweise die `validate` Phasen `build` und von aus`document1.yaml`, gefolgt von den `validate` Phasen `build` und von`document2.yaml`.

Um sicherzustellen, dass Ihre Protokolle sicher gespeichert und zur Fehlerbehebung aufbewahrt werden, empfehlen wir, die Protokollspeicherung in Amazon S3 zu konfigurieren. In Image Builder ist der Amazon S3 S3-Speicherort für die Veröffentlichung von Protokollen in der Infrastrukturkonfiguration angegeben. Weitere Informationen zur Infrastrukturkonfiguration finden Sie unter EC2 Image Builder Builder-Infrastrukturkonfiguration verwalten

Wenn keine Liste der Phasen bereitgestellt wird, führt die AWSTOE Anwendung alle Phasen in der Reihenfolge aus, die im YAML-Dokument aufgeführt ist.

Verwenden Sie die folgenden Befehle, um bestimmte Phasen in einem oder mehreren Dokumenten auszuführen.

Einphasig

```
awstoe run --documents hello-world.yml --phases build
```

Mehrere Phasen

```
awstoe run --documents hello-world.yml --phases build,test
```

Dokument ausführen

Führen Sie alle Phasen in einem einzigen Dokument aus

```
awstoe run --documents documentName.yaml
```

Führen Sie alle Phasen in mehreren Dokumenten aus

```
awstoe run --documents documentName1.yaml,documentName2.yaml
```

Geben Sie Amazon S3 S3-Informationen ein, um AWSTOE Protokolle von einem benutzerdefinierten lokalen Pfad hochzuladen (empfohlen)

```
awstoe run --documents documentName.yaml --log-s3-bucket-name <S3Bucket> --log-s3-key-
prefix <S3KeyPrefix> --log-s3-bucket-owner <S3BucketOwner> --log-directory <local_path>
```

Führen Sie alle Phasen in einem einzigen Dokument aus und zeigen Sie alle Protokolle auf der Konsole an

```
awstoe run --documents documentName.yaml --trace
```

-Beispielbefehl

```
awstoe run --documents s3://bucket/key/doc.yaml --phases build,validate
```

Dokument mit eindeutiger ID ausführen

```
awstoe run --documents <documentName>.yaml --execution-id <user provided id> --phases
 <comma separated list of phases>
```

Holen Sie sich Hilfe bei AWSTOE

```
awstoe --help
```

Verwenden Sie Komponentendokumente in AWSTOE

Um eine Komponente mit AWS Task Orchestrator and Executor (AWSTOE) zu erstellen, müssen Sie ein YAML-basiertes Dokument bereitstellen, das die Phasen und Schritte darstellt, die für die von Ihnen erstellte Komponente gelten. AWS-Services verwenden Sie Ihre Komponente, wenn sie ein neues Amazon Machine Image (AMI) oder Container-Image erstellen.

Themen

Workflow für Komponenten-Dokumente

Das AWSTOE Komponentendokument verwendet Phasen und Schritte, um verwandte Aufgaben zu gruppieren und diese Aufgaben in einem logischen Workflow für die Komponente zu organisieren.

> **Tip**
>
> Der Dienst, der Ihre Komponente zum Erstellen eines Images verwendet, implementiert möglicherweise Regeln darüber, welche Phasen für den Build-Prozess verwendet werden

sollen und wann diese Phasen ausgeführt werden dürfen. Dies ist wichtig, wenn Sie Ihre Komponente entwerfen.

Phasen

Phasen stellen den Verlauf Ihres Workflows durch den Image-Erstellungsprozess dar. Beispielsweise verwendet `build` der Image Builder Builder-Dienst während seiner Erstellungsphase *`validate`* Phasen für die von ihm erstellten Images. Es verwendet die `container-host-test` Phasen `test` und während der Testphase, um sicherzustellen, dass der Image-Snapshot oder das Container-Image die erwarteten Ergebnisse liefert, bevor das endgültige AMI erstellt oder das Container-Image verteilt wird.

Wenn die Komponente ausgeführt wird, werden die zugehörigen Befehle für jede Phase in der Reihenfolge angewendet, in der sie im Komponentendokument erscheinen.

Regeln für Phasen

- Jeder Phasenname muss innerhalb eines Dokuments eindeutig sein.
- Sie können viele Phasen in Ihrem Dokument definieren.
- Sie müssen mindestens eine der folgenden Phasen in Ihr Dokument aufnehmen:
  - build — für Image Builder wird diese Phase im Allgemeinen während der Buildphase verwendet.
  - validieren — für Image Builder wird diese Phase im Allgemeinen während der Erstellungsphase verwendet.
  - test — für Image Builder wird diese Phase im Allgemeinen während der Testphase verwendet.
- Phasen werden immer in der Reihenfolge ausgeführt, in der sie im Dokument definiert sind. Die Reihenfolge, in der sie für AWSTOE Befehle in angegeben sind, AWS CLI hat keine Auswirkung.

Schritte

Schritte sind einzelne Arbeitseinheiten, die den Arbeitsablauf innerhalb jeder Phase definieren. Die Schritte werden nacheinander ausgeführt. Die Eingabe oder Ausgabe für einen Schritt kann jedoch auch als Eingabe in einen nachfolgenden Schritt einfließen. Dies wird als „Verkettung“ bezeichnet.

Regeln für Schritte

- Der Schrittname muss für die Phase eindeutig sein.

- Der Schritt muss eine unterstützte Aktion (Aktionsmodul) verwenden, die einen Exit-Code zurückgibt.

  Eine vollständige Liste der unterstützten Aktionsmodule, deren Funktionsweise, Eingabe-/Ausgabewerte und Beispiele finden Sie unter. Aktionsmodule, die vom AWSTOE Komponentenmanager unterstützt werden

Protokollierung von Komponenten

AWSTOE erstellt bei jeder Ausführung Ihrer Komponente einen neuen Protokollordner auf den EC2-Instances, die zum Erstellen und Testen eines neuen Images verwendet werden. Bei Container-Images wird der Protokollordner im Container gespeichert.

Zur Unterstützung der Problembehandlung, falls bei der Erstellung des Images ein Fehler auftritt, werden das Eingabedokument und alle Ausgabedateien, die bei der Ausführung der Komponente AWSTOE erstellt werden, im Protokollordner gespeichert.

Der Name des Protokollordners besteht aus den folgenden Teilen:

1. Protokollverzeichnis — Wenn ein Dienst eine AWSTOE Komponente ausführt, übergibt sie das Protokollverzeichnis zusammen mit anderen Einstellungen für den Befehl. In den folgenden Beispielen zeigen wir das Protokolldateiformat, das Image Builder verwendet.
   - Linux: `/var/lib/amazon/toe/`
   - Windows: `$env:ProgramFiles\Amazon\TaskOrchestratorAndExecutor\`
2. Dateipräfix — Dies ist ein Standardpräfix, das für alle Komponenten verwendet wird: "`TOE_`“.
3. Laufzeit — Dies ist ein Zeitstempel im Format YYYY-MM-DD_HH-MM-SS_UTC-0.
4. Ausführungs-ID — Dies ist die GUID, die zugewiesen wird, wenn eine oder mehrere Komponenten ausgeführt werden. AWSTOE

Beispiel: */var/lib/amazon/toe/*`TOE`*_2021-07-01_12-34-56_UTC-0_a1bcd2e3-45f6-789a-bcde-0fa1b2c3def4*

AWSTOE speichert die folgenden Kerndateien im Protokollordner:

Eingabedateien

- document.yaml — Das Dokument, das als Eingabe für den Befehl verwendet wird. Nachdem die Komponente ausgeführt wurde, wird diese Datei als Artefakt gespeichert.

Ausgabedateien

- application.log — Das Anwendungsprotokoll enthält Informationen auf Debug-Ebene mit Zeitstempel AWSTOE darüber, was passiert, während die Komponente ausgeführt wird.
- detailedoutput.json — Diese JSON-Datei enthält detaillierte Informationen zum Ausführungsstatus, zu Eingaben, Ausgaben und Fehlern für alle Dokumente, Phasen und Schritte, die für die Komponente gelten, während sie ausgeführt wird.
- console.log — Das Konsolenprotokoll enthält alle Standardausgangsinformationen (stdout) und Standardfehlerinformationen (stderr), die in die Konsole AWSTOE geschrieben werden, während die Komponente ausgeführt wird.
- chaining.json — Diese JSON-Datei stellt Optimierungen dar, die zur Auflösung von Verkettungsausdrücken angewendet wurden. AWSTOE

> **Note**
>
> Der Protokollordner kann auch andere temporäre Dateien enthalten, die hier nicht behandelt werden.

Verkettung von Eingabe und Ausgabe

Die AWSTOE Konfigurationsverwaltungsanwendung bietet eine Funktion zum Verketten von Eingaben und Ausgaben, indem Verweise in den folgenden Formaten geschrieben werden:

```
{{ phase_name.step_name.inputs/outputs.variable }}
```

or

```
{{ phase_name.step_name.inputs/outputs[index].variable }}
```

Mit der Verkettungsfunktion können Sie Code recyceln und die Wartbarkeit des Dokuments verbessern.

Regeln für die Verkettung

- Verkettungsausdrücke können nur im Eingabebereich jedes Schritts verwendet werden.
- Anweisungen mit verketteten Ausdrücken müssen in Anführungszeichen gesetzt werden. Beispielsweise:

  - Ungültiger Ausdruck: `echo {{ phase.step.inputs.variable }}`
  - Gültiger Ausdruck: `"echo {{ phase.step.inputs.variable }}"`
  - Gültiger Ausdruck: `'echo {{ phase.step.inputs.variable }}'`
- Verkettete Ausdrücke können auf Variablen aus anderen Schritten und Phasen desselben Dokuments verweisen. Der aufrufende Dienst verfügt jedoch möglicherweise über Regeln, nach denen Verkettungsausdrücke nur im Kontext einer einzelnen Phase ausgeführt werden dürfen. Image Builder unterstützt beispielsweise keine Verkettung von der Buildphase zur Testphase, da jede Phase unabhängig ausgeführt wird.
- Indizes in der Verkettung von Ausdrücken folgen einer nullbasierten Indizierung. Der Index beginnt mit Null (0), um auf das erste Element zu verweisen.

Beispiele

Um im zweiten Eintrag des folgenden Beispielschritts auf die Quellvariable zu verweisen, lautet `{{ build.SampleS3Download.inputs[1].source }}` das Verkettungsmuster.

```
phases:
  -
    name: 'build'
    steps:
      -
        name: SampleS3Download
        action: S3Download
        timeoutSeconds: 60
        onFailure: Abort
        maxAttempts: 3
        inputs:
          -
            source: 's3://sample-bucket/sample1.ps1'
            destination: 'C:\sample1.ps1'
          -
            source: 's3://sample-bucket/sample2.ps1'
            destination: 'C:\sample2.ps1'
```

Um auf die Ausgangsvariable (entspricht „Hello“) des folgenden Beispielschritts zu verweisen, lautet das Verkettungsmuster. `{{ build.SamplePowerShellStep.outputs.stdout }}`

```
phases:
  -
```

```
    name: 'build'
    steps:
      -
        name: SamplePowerShellStep
        action: ExecutePowerShell
        timeoutSeconds: 120
        onFailure: Abort
        maxAttempts: 3
        inputs:
          commands:
            - 'Write-Host "Hello"'
```

Schema und Definitionen des Dokuments

Das Folgende ist das YAML-Schema für ein Dokument.

```
name: (optional)
description: (optional)
schemaVersion: "string"

phases:
  - name: "string"
    steps:
        - name: "string"
          action: "string"
          timeoutSeconds: integer
          onFailure: "Abort|Continue|Ignore"
          maxAttempts: integer
          inputs:
```

Die Schemadefinitionen für ein Dokument lauten wie folgt.

Feld	Beschreibung	Typ	Erforderlich
Name	Name des Dokuments.	String	Nein
description	Beschreibung des Dokuments.	String	Nein

Feld	Beschreibung	Typ	Erforderlich
schemaVersion	Schemaversion des Dokuments, derzeit 1.0.	String	Ja
phases	Eine Liste der Phasen mit ihren Schritten.	Auflisten	Ja

Die Schemadefinitionen für eine Phase lauten wie folgt.

Feld	Beschreibung	Typ	Erforderlich
Name	Name der Phase.	String	Ja
steps	Liste der Schritte in der Phase.	Auflisten	Ja

Die Schemadefinitionen für einen Schritt lauten wie folgt.

Feld	Beschreibung	Typ	Erforderlich	Standardwert
Name	Benutzerdefinierter Name für den Schritt.	String		
action	Schlüsselwort, das sich auf das Modul bezieht, das den Schritt ausführt.	String		
timeoutSeconds	Anzahl der Sekunden, die der Schritt ausgeführt wird, bevor	Ganzzahl	Nein	7.200 Sekunden (120 Minuten)

Feld	Beschreibung	Typ	Erforderlich	Standardwert
	er fehlschlägt oder es erneut versucht. Unterstützt auch den Wert -1, was auf ein unendliches Timeout hinweist. 0 und andere negative Werte sind nicht zulässig.			

Feld	Beschreibung	Typ	Erforderlich	Standardwert
onFailure	Gibt an, wie der Schritt im Falle eines Fehlers vorgehen soll. Gültige Werte sind: • Abbrechen — Der Schritt schlägt nach der maximalen Anzahl von Versuchen fehl und beendet die Ausführung. Setzt den Status für Phase und Dokument auf `Failed`. • Fortfahren — Der Schritt schlägt nach der maximalen Anzahl von Versuchen fehl und setzt die Ausführung der verbleibenden Schritte fort. Setzt den Status für Phase und Dokument auf `Failed`.	String	Nein	Abbrechen

Feld	Beschreibung	Typ	Erforderlich	Standardwert
	• Ignorieren — Legt fest, dass der `IgnoredFailure` Schritt die maximale Anzahl fehlgeschlagener Versuche erreicht hat, und setzt die Ausführung der verbleibenden Schritte fort. Setzt den Status für Phase und Dokument auf `SuccessWithIgnoredFailure` .			
maxAttempts	Höchstzahl zulässiger Versuche, bevor der Schritt fehlschlägt.	Ganzzahl	Nein	1
inputs	Enthält Parameter, die das Aktionsmodul benötigt, um den Schritt auszuführen.	Diktieren	Ja	

Beispielschemas für Dokumente

Im Folgenden finden Sie ein Beispieldokumentschema zum Installieren aller verfügbaren Windows-Updates, zum Ausführen eines Konfigurationsskripts, zum Überprüfen der Änderungen vor der Erstellung des AMI und zum Testen der Änderungen nach der Erstellung des AMI.

```
name: RunConfig_UpdateWindows
description: 'This document will install all available Windows updates and run a config
 script. It will then validate the changes before an AMI is created. Then after AMI
 creation, it will test all the changes.'
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: DownloadConfigScript
        action: S3Download
        timeoutSeconds: 60
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - source: 's3://customer-bucket/config.ps1'
            destination: 'C:\config.ps1'

      - name: RunConfigScript
        action: ExecutePowerShell
        timeoutSeconds: 120
        onFailure: Abort
        maxAttempts: 3
        inputs:
          file: '{{build.DownloadConfigScript.inputs[0].destination}}'

      - name: Cleanup
        action: DeleteFile
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - path: '{{build.DownloadConfigScript.inputs[0].destination}}'

      - name: RebootAfterConfigApplied
        action: Reboot
        inputs:
          delaySeconds: 60
```

```
      - name: InstallWindowsUpdates
        action: UpdateOS

  - name: validate
    steps:
      - name: DownloadTestConfigScript
        action: S3Download
        timeoutSeconds: 60
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - source: 's3://customer-bucket/testConfig.ps1'
            destination: 'C:\testConfig.ps1'

      - name: ValidateConfigScript
        action: ExecutePowerShell
        timeoutSeconds: 120
        onFailure: Abort
        maxAttempts: 3
        inputs:
          file: '{{validate.DownloadTestConfigScript.inputs[0].destination}}'

      - name: Cleanup
        action: DeleteFile
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - path: '{{validate.DownloadTestConfigScript.inputs[0].destination}}'

  - name: test
    steps:
      - name: DownloadTestConfigScript
        action: S3Download
        timeoutSeconds: 60
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - source: 's3://customer-bucket/testConfig.ps1'
            destination: 'C:\testConfig.ps1'

      - name: ValidateConfigScript
        action: ExecutePowerShell
        timeoutSeconds: 120
        onFailure: Abort
```

```
        maxAttempts: 3
        inputs:
          file: '{{test.DownloadTestConfigScript.inputs[0].destination}}'
```

Im Folgenden finden Sie ein Beispieldokumentschema zum Herunterladen und Ausführen einer benutzerdefinierten Linux-Binärdatei.

```
name: LinuxBin
description: Download and run a custom Linux binary file.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: Download
        action: S3Download
        inputs:
          - source: s3://<replaceable>mybucket</replaceable>/
<replaceable>myapplication</replaceable>
            destination: /tmp/<replaceable>myapplication</replaceable>
      - name: Enable
        action: ExecuteBash
        onFailure: Continue
        inputs:
          commands:
            - 'chmod u+x {{ build.Download.inputs[0].destination }}'
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: '{{ build.Download.inputs[0].destination }}'
          arguments:
            - '--install'
      - name: Delete
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

Im Folgenden finden Sie ein Beispiel für ein Dokumentschema zur Installation von AWS CLI auf einer Windows-Instanz mithilfe der Setup-Datei.

```
name: InstallCLISetUp
description: Install &CLI; using the setup file
```

```
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: Download
        action: S3Download
        inputs:
          - source: s3://aws-cli/AWSCLISetup.exe
            destination: C:\Windows\temp\AWSCLISetup.exe
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: '{{ build.Download.inputs[0].destination }}'
          arguments:
            - '/install'
            - '/quiet'
            - '/norestart'
      - name: Delete
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

Im Folgenden finden Sie ein Beispiel für ein Dokumentschema zur Installation AWS CLI mithilfe des MSI-Installationsprogramms.

```
name: InstallCLIMSI
description: Install &CLI; using the MSI installer
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: Download
        action: S3Download
        inputs:
          - source: s3://aws-cli/AWSCLI64PY3.msi
            destination: C:\Windows\temp\AWSCLI64PY3.msi
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: 'C:\Windows\System32\msiexec.exe'
          arguments:
```

```
            - '/i'
            - '{{ build.Download.inputs[0].destination }}'
            - '/quiet'
            - '/norestart'
      - name: Delete
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

Definieren und referenzieren Sie Variablen in AWSTOE

Variablen bieten eine Möglichkeit, Daten mit aussagekräftigen Namen zu versehen, die in der gesamten Anwendung verwendet werden können. Sie können benutzerdefinierte Variablen mit einfachen und lesbaren Formaten für komplexe Workflows definieren und sie im YAML-Anwendungskomponentendokument für eine AWSTOE Komponente referenzieren.

Dieser Abschnitt enthält Informationen, die Ihnen helfen, Variablen für Ihre AWSTOE Komponente im Dokument für die YAML-Anwendungskomponente zu definieren, einschließlich Syntax, Namenseinschränkungen und Beispielen.

Parameter

Parameter sind veränderbare Variablen mit Einstellungen, die die aufrufende Anwendung zur Laufzeit bereitstellen kann. Sie können Parameter im `Parameters` Abschnitt des YAML-Dokuments definieren.

Regeln für Parameternamen

- Der Name muss zwischen 3 und 128 Zeichen lang sein.
- Der Name darf nur alphanumerische Zeichen (a-z, A-Z, 0-9), Bindestriche (-) oder Unterstriche (_) enthalten.
- Der Name muss innerhalb des Dokuments eindeutig sein.
- Der Name muss als YAML-Zeichenfolge angegeben werden.

Syntax

```
parameters:
  - <name>:
      type: <parameter type>
      default: <parameter value>
```

```
      description: <parameter description>
```

Tastenname	Erforderlich	Beschreibung
`name`	Ja	Der Name des Parameters. Muss für das Dokument eindeutig sein (er darf nicht mit anderen Parameternamen oder Konstanten identisch sein).
`type`	Ja	Der Datentyp des Parameters. Zu den unterstützten Typen gehören:`string`.
`default`	Nein	Der Standardwert für den Parameter.
`description`	Nein	Beschreibt den Parameter.

Referenzparameterwerte in einem Dokument

Sie können Parameter in Step- oder Loop-Eingaben innerhalb Ihres YAML-Dokuments wie folgt referenzieren:

- Bei Parameterreferenzen wird zwischen Groß- und Kleinschreibung unterschieden, und der Name muss exakt übereinstimmen.
- Der Name muss in doppelte geschweifte Klammern eingeschlossen werden. `{{ MyParameter}}`
- Leerzeichen sind innerhalb der geschweiften Klammern zulässig und werden automatisch gekürzt. Beispielsweise sind alle der folgenden Verweise gültig:

  `{{ MyParameter }}`, `{{ MyParameter}}`, `{{MyParameter }}`, `{{MyParameter}}`
- Der Verweis im YAML-Dokument muss als Zeichenfolge (in einfache oder doppelte Anführungszeichen eingeschlossen) angegeben werden.

  Zum Beispiel: - `{{ MyParameter }}` ist nicht gültig, da es nicht als Zeichenfolge identifiziert wird.

Die folgenden Verweise sind jedoch beide gültig: - '{{ *MyParameter* }}' und- "{{ *MyParameter* }}".

Beispiele

Die folgenden Beispiele zeigen, wie Sie Parameter in Ihrem YAML-Dokument verwenden können:

- Beziehen Sie sich auf einen Parameter in den Schritteingaben:

```
name: Download AWS CLI version 2
schemaVersion: 1.0
parameters:
  - Source:
      type: string
      default: 'https://awscli.amazonaws.com/AWSCLIV2.msi'
      description: The AWS CLI installer source URL.
phases:
  - name: build
    steps:
      - name: Download
        action: WebDownload
        inputs:
          - source: '{{ Source }}'
            destination: 'C:\Windows\Temp\AWSCLIV2.msi'
```

- Beziehen Sie sich auf einen Parameter in Loop-Eingängen:

```
name: PingHosts
schemaVersion: 1.0
parameters:
  - Hosts:
      type: string
      default: 127.0.0.1,amazon.com
      description: A comma separated list of hosts to ping.
phases:
  - name: build
    steps:
      - name: Ping
        action: ExecuteBash
        loop:
          forEach:
            list: '{{ Hosts }}'
```

```
            delimiter: ','
        inputs:
          commands:
            - ping -c 4 {{ loop.value }}
```

Parameter zur Laufzeit überschreiben

Sie können die `--parameters` Option AWS CLI mit einem Schlüssel-Wert-Paar verwenden, um zur Laufzeit einen Parameterwert festzulegen.

- <name><value>Geben Sie das Schlüssel-Wert-Paar für den Parameter als Namen und Wert an, getrennt durch ein Gleichheitszeichen (=).
- Mehrere Parameter müssen durch ein Komma getrennt werden.
- Parameternamen, die im YAML-Komponentendokument nicht gefunden werden, werden ignoriert.
- Der Parametername und der Wert sind beide erforderlich.

> **Important**
>
> Bei den Komponentenparametern handelt es sich um reine Textwerte, die angemeldet sind AWS CloudTrail. Wir empfehlen, dass Sie AWS Secrets Manager oder den AWS Systems Manager Parameter Store verwenden, um Ihre Geheimnisse zu speichern. Weitere Informationen zu Secrets Manager finden Sie unter Was ist Secrets Manager? im AWS Secrets Manager Benutzerhandbuch. Weitere Informationen zum AWS Systems Manager Parameterspeicher finden Sie unter AWS Systems Manager Parameterspeicher im AWS Systems Manager Benutzerhandbuch.

Syntax

```
--parameters name1=value1,name2=value2...
```

CLI-Option	Erforderlich	Beschreibung
--parameters *Name* = *Wert*,...	Nein	Diese Option verwendet eine Liste von Schlüssel-Wert-Paa

CLI-Option	Erforderlich	Beschreibung
		ren mit dem Parameternamen als Schlüssel.

Beispiele

Die folgenden Beispiele zeigen, wie Sie Parameter in Ihrem YAML-Dokument verwenden können:

- Das in dieser `--parameter` Option angegebene Parameter-Schlüssel-Wert-Paar ist nicht gültig:

```
--parameters ntp-server=
```

- Legen Sie ein Parameter-Schlüssel-Wert-Paar mit der `--parameter` folgenden Option fest: AWS CLI

```
--parameters ntp-server=ntp-server-windows-qe.us-east1.amazon.com
```

- Legen Sie mehrere Parameter-Schlüssel-Wert-Paare mit der Option im `--parameter` Feld fest: AWS CLI

```
--parameters ntp-server=ntp-server.amazon.com,http-url=https://internal-us-
east1.amazon.com
```

Konstanten

Konstanten sind unveränderliche Variablen, die nach ihrer Definition nicht geändert oder überschrieben werden können. Konstanten können mithilfe von Werten im Abschnitt eines Dokuments definiert werden. `constants` AWSTOE

Regeln für Konstantennamen

- Der Name muss zwischen 3 und 128 Zeichen lang sein.
- Der Name darf nur alphanumerische Zeichen (a-z, A-Z, 0-9), Bindestriche (-) oder Unterstriche (_) enthalten.
- Der Name muss innerhalb des Dokuments eindeutig sein.
- Der Name muss als YAML-Zeichenfolge angegeben werden.

Syntax

```
constants:
  - <name>:
      type: <constant type>
      value: <constant value>
```

Tastenname	Erforderlich	Beschreibung
`name`	Ja	Name der Konstante. Muss für das Dokument eindeutig sein (er darf nicht mit anderen Parameternamen oder Konstanten identisch sein).
`value`	Ja	Wert der Konstante.
`type`	Ja	Typ der Konstante. Der unterstützte Typ ist `string`.

Verweisen Sie auf konstante Werte in einem Dokument

Sie können in Step- oder Loop-Eingaben in Ihrem YAML-Dokument wie folgt auf Konstanten verweisen:

- Bei konstanten Verweisen wird zwischen Groß- und Kleinschreibung unterschieden, und der Name muss exakt übereinstimmen.
- Der Name muss in doppelte geschweifte Klammern eingeschlossen werden. `{{ MyConstant}}`
- Leerzeichen sind innerhalb der geschweiften Klammern zulässig und werden automatisch gekürzt. Beispielsweise sind alle der folgenden Verweise gültig:

  `{{ MyConstant }}`, `{{ MyConstant}}`, `{{MyConstant }}`, `{{MyConstant}}`
- Der Verweis im YAML-Dokument muss als Zeichenfolge (in einfache oder doppelte Anführungszeichen eingeschlossen) angegeben werden.

  Zum Beispiel: - `{{ MyConstant }}` ist nicht gültig, da es nicht als Zeichenfolge identifiziert wird.

Die folgenden Verweise sind jedoch beide gültig: - '{{ *MyConstant* }}' und- "{{ *MyConstant* }}".

Beispiele

In Schritteingaben referenzierte Konstante

```
name: Download AWS CLI version 2
schemaVersion: 1.0
constants:
  - Source:
      type: string
      value: https://awscli.amazonaws.com/AWSCLIV2.msi
phases:
  - name: build
    steps:
      - name: Download
        action: WebDownload
        inputs:
          - source: '{{ Source }}'
            destination: 'C:\Windows\Temp\AWSCLIV2.msi'
```

In Loop-Eingängen wird auf eine Konstante verwiesen

```
name: PingHosts
schemaVersion: 1.0
constants:
  - Hosts:
      type: string
      value: 127.0.0.1,amazon.com
phases:
  - name: build
    steps:
      - name: Ping
        action: ExecuteBash
        loop:
          forEach:
            list: '{{ Hosts }}'
            delimiter: ','
        inputs:
          commands:
```

```
            - ping -c 4 {{ loop.value }}
```

Verwenden Sie Looping-Konstrukte in AWSTOE

Dieser Abschnitt enthält Informationen, die Ihnen beim Erstellen von Schleifenkonstrukten in der helfen. AWSTOE Schleifenkonstrukte definieren eine sich wiederholende Abfolge von Befehlen. Sie können die folgenden Typen von Schleifenkonstrukten verwenden in: AWSTOE

- `for`Konstrukte — Iteriert über eine begrenzte Folge von ganzen Zahlen.
- `forEach`Konstrukte
  - `forEach`Schleife mit Eingabeliste — Iteriert über eine endliche Sammlung von Zeichenketten.
  - `forEach`Schleife mit begrenzter Liste — Iteriert über eine endliche Sammlung von Zeichenketten, die durch ein Trennzeichen verbunden sind.

> **Note**
>
> Schleifenkonstrukte unterstützen nur Zeichenkettendatentypen.

Themen, die sich in einer Schleife wiederholen

- Referenz-Iterationsvariablen
- Arten von Looping-Konstrukten
- Felder für Schritte
- Ausgaben für Schritt und Iteration

Referenz-Iterationsvariablen

Um auf den Index und den Wert der aktuellen Iterationsvariablen zu verweisen, `{{ loop.* }}` muss der Referenzausdruck im Eingabekörper eines Schritts verwendet werden, der ein Schleifenkonstrukt enthält. Dieser Ausdruck kann nicht verwendet werden, um auf die Iterationsvariablen des Schleifenkonstrukts eines anderen Schritts zu verweisen.

Der Referenzausdruck besteht aus den folgenden Elementen:

- `{{ loop.index }}`— Die Ordinalposition der aktuellen Iteration, die indexiert ist. `0`
- `{{ loop.value }}`— Der Wert, der der aktuellen Iterationsvariablen zugeordnet ist.

Namen von Schleifen

Alle Schleifenkonstrukte haben ein optionales Namensfeld zur Identifizierung. Wenn ein Schleifenname angegeben wird, kann er verwendet werden, um auf Iterationsvariablen im Eingabekörper des Schritts zu verweisen. Um auf die Iterationsindizes und Werte einer benannten Schleife zu verweisen, verwenden Sie `{{ <loop_name>.* }}` with `{{ loop.* }}` im Eingabebereich des Schritts. Dieser Ausdruck kann nicht verwendet werden, um auf das benannte Schleifenkonstrukt eines anderen Schritts zu verweisen.

Der Referenzausdruck besteht aus den folgenden Elementen:

- `{{ <loop_name>.index }}`— Die Ordinalposition der aktuellen Iteration der benannten Schleife, die indexiert ist. `0`
- `{{ <loop_name>.value }}`— Der Wert, der der aktuellen Iterationsvariablen der benannten Schleife zugeordnet ist.

Referenzausdrücke auflösen

Der AWSTOE löst Referenzausdrücke wie folgt auf:

- `{{ <loop_name>.* }}`— AWSTOE löst diesen Ausdruck mit der folgenden Logik auf:
  - Wenn die Schleife des aktuell laufenden Schritts mit dem `<loop_name>` Wert übereinstimmt, wird der Referenzausdruck in das Schleifenkonstrukt des aktuell laufenden Schritts aufgelöst.
  - `<loop_name>`wird in das benannte Schleifenkonstrukt aufgelöst, wenn es im aktuell laufenden Schritt vorkommt.
- `{{ loop.* }}`— AWSTOE löst den Ausdruck unter Verwendung des im aktuell ausgeführten Schritt definierten Schleifenkonstrukts auf.

Wenn Referenzausdrücke innerhalb eines Schritts verwendet werden, der keine Schleife enthält, werden die Ausdrücke AWSTOE nicht aufgelöst und sie erscheinen im Schritt ohne Ersatz.

> **Note**
>
> Referenzausdrücke müssen in doppelte Anführungszeichen eingeschlossen werden, damit sie vom YAML-Compiler korrekt interpretiert werden.

Arten von Looping-Konstrukten

Dieser Abschnitt enthält Informationen und Beispiele zu Schleifenkonstrukttypen, die in der verwendet werden können. AWSTOE

Typen von Looping-Konstrukten

- forSchleife
- forEachSchleife mit Eingabeliste
- forEachSchleife mit begrenzter Liste

forSchleife

Die `for` Schleife iteriert über einen Bereich von ganzen Zahlen, die innerhalb einer Grenze angegeben sind, die durch den Anfang und das Ende der Variablen umrissen wird. Die iterierenden Werte befinden sich in der Menge `[start, end]` und enthalten Grenzwerte.

AWSTOE überprüft die `updateBy` Werte`start`, und`end`, um sicherzustellen, dass die Kombination nicht zu einer Endlosschleife führt.

`for`Schleifenschema

```
  - name: "StepName"
    action: "ActionModule"
    loop:
      name: "string"
      for:
        start: int
        end: int
        updateBy: int
inputs:
  ...
```

forSchleifeneingabe

Feld	Beschreibung	Typ	Erforderlich	Standard
`name`	Eindeutiger Name der Schleife. Er muss im	String	Nein	""

Feld	Beschreibung	Typ	Erforderlich	Standard
	Vergleich zu anderen Schleifennamen in derselben Phase eindeutig sein.			
`start`	Startwert der Iteration. Akzeptiert keine Verkettung von Ausdrücken.	Ganzzahl	Ja	–
`end`	Endwert der Iteration. Akzeptiert keine Verkettung von Ausdrücken.	Ganzzahl	Ja	–
`updateBy`	Unterschied, um den ein iterierender Wert durch Addition aktualisiert wird. Es muss ein negativer oder positiver Wert ungleich Null sein. Akzeptiert keine Verkettung von Ausdrücken.	Ganzzahl	Ja	–

`for`Beispiel für eine Schleifeneingabe

```
  - name: "CalculateFileUploadLatencies"
```

```
    action: "ExecutePowerShell"
    loop:
      for:
        start: 100000
        end: 1000000
        updateBy: 100000
    inputs:
      commands:
        - |
          $f = new-object System.IO.FileStream c:\temp\test{{ loop.index }}.txt,
 Create, ReadWrite
          $f.SetLength({{ loop.value }}MB)
          $f.Close()
        - c:\users\administrator\downloads\latencyTest.exe --file c:\temp
\test{{ loop.index }}.txt
        - AWS s3 cp c:\users\administrator\downloads\latencyMetrics.json s3://bucket/
latencyMetrics.json
        - |
          Remove-Item -Path c:\temp\test{{ loop.index }}.txt
          Remove-Item -Path c:\users\administrator\downloads\latencyMetrics.json
```

`forEach`Schleife mit Eingabeliste

Die `forEach` Schleife iteriert anhand einer expliziten Werteliste, bei der es sich um Zeichenketten und verkettete Ausdrücke handeln kann.

`forEach`Schleife mit Eingabelistenschema

```
  - name: "StepName"
    action: "ActionModule"
    loop:
      name: "string"
      forEach:
        - "string"
    inputs:
  ...
```

`forEach`Schleife mit Eingabe in die Eingabeliste

Feld	Beschreibung	Typ	Erforderlich	Standard
`name`	Eindeutiger Name der	String	Nein	""

Feld	Beschreibung	Typ	Erforderlich	Standard
	Schleife. Er muss im Vergleich zu anderen Schleifennamen in derselben Phase eindeutig sein.			
Liste der Zeichenketten der `forEach` Schleife	Liste der Zeichenketten für die Iteration . Akzeptiert verkettete Ausdrücke als Zeichenketten in der Liste. Verkettete Ausdrücke müssen in doppelte Anführungszeichen eingeschlossen werden, damit der YAML-Compiler sie korrekt interpretieren kann.	Liste von Zeichenfolgen	Ja	–

`forEach`Schleife mit Eingabeliste, Beispiel 1

```
- name: "ExecuteCustomScripts"
  action: "ExecuteBash"
```

```
    loop:
      name: BatchExecLoop
      forEach:
        - /tmp/script1.sh
        - /tmp/script2.sh
        - /tmp/script3.sh
    inputs:
      commands:
        - echo "Count {{ BatchExecLoop.index }}"
        - sh "{{ loop.value }}"
        - |
          retVal=$?
          if [ $retVal -ne 0 ]; then
            echo "Failed"
          else
            echo "Passed"
         fi
```

`forEach`Schleife mit Eingabeliste, Beispiel 2

```
- name: "RunMSIWithDifferentArgs"
  action: "ExecuteBinary"
  loop:
    name: MultiArgLoop
    forEach:
      - "ARG1=C:\Users ARG2=1"
      - "ARG1=C:\Users"
      - "ARG1=C:\Users ARG3=C:\Users\Administrator\Documents\f1.txt"
  inputs:
    commands:
      path: "c:\users\administrator\downloads\runner.exe"
      args:
        - "{{ MultiArgLoop.value }}"
```

`forEach`Schleife mit Eingabeliste, Beispiel 3

```
- name: "DownloadAllBinaries"
  action: "S3Download"
  loop:
    name: MultiArgLoop
    forEach:
      - "bin1.exe"
      - "bin10.exe"
```

```
        - "bin5.exe"
    inputs:
      - source: "s3://bucket/{{ loop.value }}"
        destination: "c:\temp\{{ loop.value }}"
```

`forEach`Schleife mit begrenzter Liste

Die Schleife iteriert über eine Zeichenfolge, die Werte enthält, die durch ein Trennzeichen getrennt sind. Um über die Bestandteile der Zeichenfolge zu iterieren, AWSTOE verwendet sie das Trennzeichen, um die Zeichenfolge in ein Array aufzuteilen, das für die Iteration geeignet ist.

`forEach`Schleife mit einem durch Trennzeichen getrennten Listenschema

```
- name: "StepName"
  action: "ActionModule"
  loop:
    name: "string"
    forEach:
      list: "string"
      delimiter: ".,;:\n\t -_"
  inputs:
...
```

`forEach`Schleife mit begrenzter Listeneingabe

Feld	Beschreibung	Typ	Erforderlich	Standard
`name`	Der Schleife wurde ein eindeutiger Name gegeben. Er sollte im Vergleich zu anderen Schleifennamen in derselben Phase eindeutig sein.	String	Nein	""
`list`	Eine Zeichenfolge, die aus	String	Ja	–

Feld	Beschreibung	Typ	Erforderlich	Standard
	einzelnen Zeichenketten besteht, die durch ein gemeinsames Trennzeichen miteinander verbunden sind. Akzeptiert auch verkettete Ausdrücke. Stellen Sie bei verketteten Ausdrücken sicher, dass diese in doppelte Anführungszeichen eingeschlossen sind, damit sie vom YAML-Compiler korrekt interpretiert werden können.			

Feld	Beschreibung	Typ	Erforderlich	Standard
`delimiter`	Zeichen, das verwendet wird, um Zeichenketten innerhalb eines Blocks voneinander zu trennen. Die Standardeinstellung ist das Kommazeichen. Aus der angegebenen Liste ist nur ein Trennzeichen zulässig: • Punkt: `"."` • Komma: `","` • Semikolon: `";"` • Doppelpunkt: `":"` • Neue Zeile: `"\n"` • Registerkarte: `"\t"` • Leertaste: `" "` •	String	Nein	Komma: `","`

Feld	Beschreibung	Typ	Erforderlich	Standard
	Bindestrich: "-" • Unterstrich: "_" Verkettungsausdrücke können nicht verwendet werden.			

> **Note**
>
> Der Wert von `list` wird als unveränderliche Zeichenfolge behandelt. Wenn die Quelle von während der Laufzeit geändert `list` wird, wird dies während der Ausführung nicht berücksichtigt.

`forEach`Schleife mit begrenzter Liste, Beispiel 1

In diesem Beispiel wird das folgende Verkettungsausdrucksmuster verwendet, um auf die Ausgabe eines anderen Schritts zu verweisen:. `<phase_name>.<step_name>.[inputs | outputs].<var_name>`

```
- name: "RunMSIs"
  action: "ExecuteBinary"
  loop:
    forEach:
      list: "{{ build.GetAllMSIPathsForInstallation.outputs.stdout }}"
      delimiter: "\n"
  inputs:
    commands:
      path: "{{ loop.value }}"
```

`forEach`Schleife mit begrenzter Liste, Beispiel 2

```
- name: "UploadMetricFiles"
  action: "S3Upload"
  loop:
    forEach:
      list: "/tmp/m1.txt,/tmp/m2.txt,/tmp/m3.txt,..."
  inputs:
    commands:
      - source: "{{ loop.value }}"
        destination: "s3://bucket/key/{{ loop.value }}"
```

Felder für Schritte

Loops sind Teil eines Schritts. Felder, die sich auf die Ausführung eines Schritts beziehen, werden nicht auf einzelne Iterationen angewendet. Schrittfelder gelten wie folgt nur auf Schrittebene:

- timeoutSeconds — Alle Iterationen der Schleife müssen innerhalb des in diesem Feld angegebenen Zeitraums ausgeführt werden. Wenn bei der Schleifenausführung eine AWSTOE Zeitüberschreitung eintritt, wird die Wiederholungsrichtlinie des Schritts ausgeführt und der Timeout-Parameter für jeden neuen Versuch zurückgesetzt. Wenn der Schleifenlauf nach Erreichen der maximalen Anzahl von Wiederholungen den Timeout-Wert überschreitet, gibt die Fehlermeldung des Schritts an, dass für den Schleifenlauf eine Zeitüberschreitung aufgetreten ist.
- onFailure — Die Fehlerbehandlung wird wie folgt auf den Schritt angewendet:
  - Wenn onFailure auf gesetzt ist`Abort`, wird die Schleife AWSTOE beendet und der Schritt gemäß der Wiederholungsrichtlinie wiederholt. AWSTOE Markiert den aktuellen Schritt nach der maximalen Anzahl von Wiederholungsversuchen als fehlgeschlagen und beendet die Ausführung des Prozesses.

    AWSTOE setzt den Statuscode für die übergeordnete Phase und das Dokument auf`Failed`.

    > **Note**
    >
    > Nach dem fehlgeschlagenen Schritt werden keine weiteren Schritte ausgeführt.

  - Wenn onFailure auf gesetzt ist`Continue`, wird die Schleife AWSTOE beendet und der Schritt gemäß der Wiederholungsrichtlinie wiederholt. AWSTOE Markiert den aktuellen Schritt nach der maximalen Anzahl von Wiederholungsversuchen als fehlgeschlagen und fährt mit der Ausführung des nächsten Schritts fort.

AWSTOE setzt den Statuscode für die übergeordnete Phase und das Dokument auf`Failed`.

- Wenn onFailure auf gesetzt ist`Ignore`, wird die Schleife AWSTOE beendet und der Schritt gemäß der Wiederholungsrichtlinie wiederholt. AWSTOE Markiert den aktuellen Schritt nach der maximalen Anzahl von Wiederholungsversuchen als `IgnoredFailure` und fährt mit der Ausführung des nächsten Schritts fort.

  AWSTOE setzt den Statuscode für die übergeordnete Phase und das Dokument auf`SuccessWithIgnoredFailure`.

  > **Note**
  >
  > Dies wird immer noch als erfolgreiche Ausführung angesehen, enthält jedoch Informationen, die Sie darüber informieren, dass ein oder mehrere Schritte fehlgeschlagen sind und ignoriert wurden.

- maxAttempts — Bei jeder Wiederholung werden der gesamte Schritt und alle Iterationen von Anfang an ausgeführt.
- Status — Der Gesamtstatus der Ausführung eines Schritts. `status`stellt nicht den Status einzelner Iterationen dar. Der Status eines Schritts mit Schleifen wird wie folgt bestimmt:
  - Wenn eine einzelne Iteration nicht ausgeführt werden kann, weist der Status eines Schritts auf einen Fehler hin.
  - Wenn alle Iterationen erfolgreich sind, deutet der Status eines Schritts auf Erfolg hin.
- StartTime — Die Gesamtstartzeit der Ausführung eines Schritts. Stellt nicht die Startzeit einzelner Iterationen dar.
- EndTime — Die Gesamtendzeit der Ausführung eines Schritts. Stellt nicht die Endzeit einzelner Iterationen dar.
- FailureMessage — Schließt die Iterationsindizes ein, die bei Fehlern ohne Timeout fehlgeschlagen sind. Bei Timeoutfehlern gibt die Meldung an, dass der Schleifenlauf fehlgeschlagen ist. Es werden keine individuellen Fehlermeldungen für jede Iteration bereitgestellt, um die Größe der Fehlermeldungen zu minimieren.

Ausgaben für Schritt und Iteration

Jede Iteration enthält eine Ausgabe. Am Ende eines Schleifenlaufs AWSTOE konsolidiert es alle erfolgreichen Iterationsausgaben in. `detailedOutput.json` Die konsolidierten Ausgaben sind eine

Zusammenstellung von Werten, die zu den entsprechenden Ausgabeschlüsseln gehören, wie sie im Ausgabeschema des Aktionsmoduls definiert sind. Das folgende Beispiel zeigt, wie die Ausgaben konsolidiert werden:

Ausgabe von **`ExecuteBash`** für Iteration 1

```
{
 "stdout":"Hello"
}
```

Ausgabe von **`ExecuteBash`** für Iteration 2

```
{
 "stdout":"World"
}
```

Ausgabe von **`ExecuteBash`** für Schritt

```
{
 "stdout":"Hello\nWorld"
}
```

Zum Beispiel `ExecuteBinary` sind `ExecuteBash``ExecutePowerShell`, und Aktionsmodule, die `STDOUT` als Aktionsmodulausgabe zurückgegeben werden. `STDOUT`Nachrichten werden mit dem neuen Zeilenzeichen verknüpft, um die Gesamtausgabe des Step-In zu erzeugen`detailedOutput.json`.

AWSTOE konsolidiert nicht die Ausgaben erfolgloser Iterationen.

Aktionsmodule, die vom AWSTOE Komponentenmanager unterstützt werden

Image-Building-Services wie EC2 Image Builder verwenden AWSTOE Aktionsmodule, um die EC2-Instances zu konfigurieren, die zum Erstellen und Testen von benutzerdefinierten Computer-Images verwendet werden. In diesem Abschnitt werden die Funktionen häufig verwendeter AWSTOE Aktionsmodule und deren Konfiguration sowie Beispiele beschrieben.

AWSTOE Komponenten werden mit Klartext-YAML-Dokumenten verfasst. Weitere Hinweise zur Dokumentensyntax finden Sie unter. Verwenden Sie Komponentendokumente in AWSTOE

> **Note**
>
> Alle Aktionsmodule verwenden dasselbe Konto wie der Systems Manager Manager-Agent, wenn sie ausgeführt werden, was `root` unter Linux und `NT Authority\SYSTEM` unter Windows der Fall ist.

Typen von Aktionsmodulen

- Allgemeine Ausführungsmodule
- Module zum Herunterladen und Hochladen von Dateien
- Betriebsmodule für das Dateisystem
- Aktionen zur Softwareinstallation
- Aktionsmodule des Systems

Allgemeine Ausführungsmodule

Der folgende Abschnitt enthält Einzelheiten zu Aktionsmodulen, die allgemeine Befehle und Anweisungen zur Ausführung ausführen.

Allgemeine Aktionsmodule für die Ausführung

- ExecuteBash
- ExecuteBinary
- ExecuteDocument
- ExecutePowerShell

ExecuteBash

Das ExecuteBashAktionsmodul ermöglicht es Ihnen, Bash-Skripte mit Inline-Shell-Code/Befehlen auszuführen. Dieses Modul unterstützt Linux.

Alle Befehle und Anweisungen, die Sie im Befehlsblock angeben, werden in eine Datei konvertiert (zum Beispiel`input.sh`) und mit der Bash-Shell ausgeführt. Das Ergebnis der Ausführung der Shell-Datei ist der Exit-Code des Schritts.

Das ExecuteBashModul verarbeitet Systemneustarts, wenn das Skript mit dem Exit-Code beendet wird. 194 Wenn das Programm initiiert wird, führt es eine der folgenden Aktionen aus:

- Die Anwendung übergibt dem Aufrufer den Exit-Code, wenn er vom Systems Manager Agent ausgeführt wird. Der Systems Manager Agent führt den Systemneustart durch und führt denselben Schritt aus, der den Neustart initiiert hat, wie unter Managed Instance from Scripts neu starten beschrieben.
- Die Anwendung speichert die aktuellen Daten`executionstate`, konfiguriert einen Neustart-Trigger, um die Anwendung erneut auszuführen, und startet das System neu.

Nach dem Systemneustart führt die Anwendung denselben Schritt aus, der den Neustart initiiert hat. Wenn Sie diese Funktionalität benötigen, müssen Sie idempotente Skripten schreiben, die mehrere Aufrufe desselben Shell-Befehls verarbeiten können.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich
`commands`	Enthält eine Liste von Anweisungen oder Befehlen, die gemäß der Bash-Syntax ausgeführt werden sollen. Mehrzeiliges YAML ist zulässig.	Auflisten	Ja

Eingabebeispiel: Vor und nach einem Neustart

```
name: ExitCode194Example
description: This shows how the exit code can be used to restart a system with
 ExecuteBash
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: RestartTrigger
        action: ExecuteBash
        inputs:
          commands:
            - |
              REBOOT_INDICATOR=/var/tmp/reboot-indicator
              if [ -f "${REBOOT_INDICATOR}" ]; then
```

```
              echo 'The reboot file exists. Deleting it and exiting with success.'
              rm "${REBOOT_INDICATOR}"
              exit 0
            fi
            echo 'The reboot file does not exist. Creating it and triggering a
 restart.'
            touch "${REBOOT_INDICATOR}"
            exit 194
```

Output

Feld	Beschreibung	Typ
`stdout`	Standardausgabe der Befehlsausführung.	Zeichenfolge

Wenn Sie einen Neustart starten und den Exit-Code `194` als Teil des Aktionsmoduls zurückgeben, wird der Build mit demselben Aktionsmodulschritt fortgesetzt, der den Neustart initiiert hat. Wenn Sie einen Neustart ohne den Exit-Code starten, schlägt der Build-Prozess möglicherweise fehl.

Ausgabebeispiel: Vor dem Neustart (zum ersten Mal über das Dokument)

```
{
 “stdout”: “The reboot file does not exist. Creating it and triggering a restart."
}
```

Ausgabebeispiel: Nach dem Neustart (zweites Mal durch das Dokument)

```
{
 “stdout”: “The reboot file exists. Deleting it and exiting with success."
}
```

ExecuteBinary

Das ExecuteBinaryAktionsmodul ermöglicht es Ihnen, Binärdateien mit einer Liste von Befehlszeilenargumenten auszuführen.

Das ExecuteBinaryModul verarbeitet Systemneustarts, wenn die Binärdatei mit dem Exit-Code `194` (Linux) oder `3010` (Windows) beendet wird. In diesem Fall führt das Programm eine der folgenden Aktionen aus:

- Die Anwendung übergibt dem Aufrufer den Exit-Code, wenn er vom Systems Manager Agent ausgeführt wird. Der Systems Manager Agent übernimmt den Neustart des Systems und führt denselben Schritt aus, der den Neustart initiiert hat, wie unter Managed Instance from Scripts neu starten beschrieben.
- Die Anwendung speichert den aktuellen Status`executionstate`, konfiguriert einen Neustart-Trigger, um die Anwendung erneut auszuführen, und startet das System neu.

Nach dem Neustart des Systems führt die Anwendung denselben Schritt aus, der den Neustart initiiert hat. Wenn Sie diese Funktionalität benötigen, müssen Sie idempotente Skripten schreiben, die mehrere Aufrufe desselben Shell-Befehls verarbeiten können.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich
`path`	Der Pfad zur Binärdatei für die Ausführung.	String	Ja
`arguments`	Enthält eine Liste von Befehlsze ilenargumenten, die beim Ausführen der Binärdatei verwendet werden sollen.	Liste der Zeichenke tten	Nein

Eingabebeispiel: .NET installieren

```
- name: "InstallDotnet"
  action: ExecuteBinary
  inputs:
    path: C:\PathTo\dotnet_installer.exe
    arguments:
      - /qb
      - /norestart
```

Output

Feld	Beschreibung	Typ
`stdout`	Standardausgabe der Befehlsausführung.	Zeichenfolge

Output example

```
{
 "stdout": "success"
}
```

ExecuteDocument

Das ExecuteDocumentAktionsmodul bietet Unterstützung für verschachtelte Komponentendokumente, sodass mehrere Komponentendokumente von einem Dokument aus ausgeführt werden. AWSTOE validiert das Dokument, das zur Laufzeit im Eingabeparameter übergeben wird.

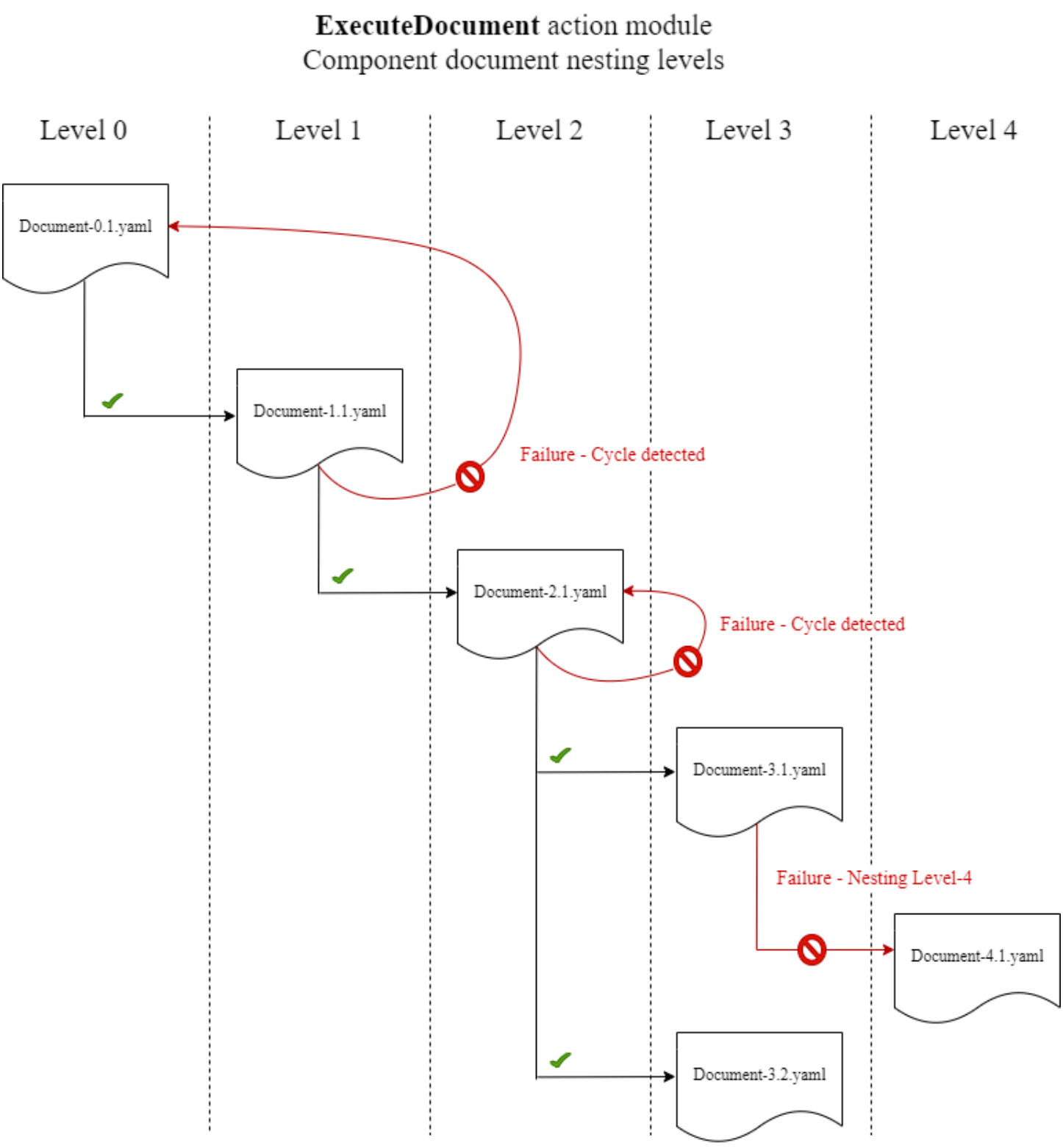
Einschränkungen

- Dieses Aktionsmodul wird einmal ausgeführt. Wiederholungen sind nicht zulässig und es besteht keine Option zum Festlegen von Timeoutlimits. ExecuteDocumentlegt die folgenden Standardwerte fest und gibt einen Fehler zurück, wenn Sie versuchen, sie zu ändern.
  - `timeoutSeconds`: -1
  - `maxAttempts`: 1

  > **Note**
  >
  > Sie können diese Werte leer lassen und AWSTOE die Standardwerte verwenden.

- Das Verschachteln von Dokumenten ist bis zu drei Ebenen zulässig, jedoch nicht mehr. Drei Verschachtelungsebenen bedeuten vier Dokumentebenen, da die oberste Ebene nicht verschachtelt ist. In diesem Szenario darf das Dokument der untersten Ebene keine anderen Dokumente aufrufen.
- Die zyklische Ausführung von Komponentendokumenten ist nicht zulässig. Jedes Dokument, das sich selbst außerhalb eines sich wiederholenden Konstrukts aufruft oder ein anderes Dokument

aufruft, das in der aktuellen Ausführungskette weiter oben steht, leitet einen Zyklus ein, der zu einer Endlosschleife führen kann. Wenn eine zyklische Ausführung AWSTOE erkannt wird, stoppt es die Ausführung und zeichnet den Fehler auf.



Wenn ein Komponentendokument versucht, sich selbst oder eines der Komponentendokumente auszuführen, die in der aktuellen Ausführungskette weiter oben stehen, schlägt die Ausführung fehl.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich
`document`	Pfad des Komponent endokuments. Gültige Optionen sind unter anderem: • Lokale Dateipfade • S3-URIs • ARNs der Build-Version der EC2 Image Builder-K omponente	String	Ja
`document-s3-bucket-owner`	Die Konto-ID des S3-Bucket-Besitzer s für den S3-Bucket , in dem Komponent endokumente gespeichert sind. (Empfohlen, wenn Sie S3-URIs in Ihrem Komponent endokument verwenden.)	String	Nein
`phases`	Phasen, die im Komponent endokument ausgeführt werden sollen, ausgedrückt als kommagetrennte Liste. Wenn keine	String	Nein

Primitiv	Beschreibung	Typ	Erforderlich
	Phasen angegeben sind, werden alle Phasen ausgeführt.		
`parameters`	Eingabeparameter, die zur Laufzeit als Schlüssel-Wert-Paare an das Komponentendokument übergeben werden.	Liste der Parameterzuordnungen	Nein

Eingabe der Parameterzuweisung

Primitiv	Beschreibung	Typ	Erforderlich
`name`	Der Name des Eingabeparameters, der an das Komponentendokument übergeben werden soll, das das ExecuteDocumentAktionsmodul ausführt.	String	Ja
`value`	Der Wert des Eingabeparameters.	String	Ja

Beispiele für die Eingabe

Die folgenden Beispiele zeigen Variationen der Eingaben für Ihr Komponentendokument, abhängig von Ihrem Installationspfad.

Eingabebeispiel: Lokaler Dokumentpfad

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        inputs:
          document: Sample-1.yaml
          phases: build
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Eingabebeispiel: S3-URI als Dokumentenpfad

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        inputs:
          document: s3://my-bucket/Sample-1.yaml
          document-s3-bucket-owner: 123456789012
          phases: build,validate
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Eingabebeispiel: ARN der EC2 Image Builder Builder-Komponente als Dokumentpfad

```
# main.yaml
schemaVersion: 1.0
```

```
phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        inputs:
          document: arn:aws:imagebuilder:us-west-2:aws:component/Sample-Test/1.0.0
          phases: test
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Verwenden einer ForEach Schleife zum Ausführen von Dokumenten

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        loop:
          name: 'myForEachLoop'
          forEach:
            - Sample-1.yaml
            - Sample-2.yaml
        inputs:
          document: "{{myForEachLoop.value}}"
          phases: test
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Verwenden einer For-Schleife zum Ausführen von Dokumenten

```
# main.yaml
schemaVersion: 1.0
```

```
phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        loop:
          name: 'myForLoop'
          for:
            start: 1
            end: 2
            updateBy: 1
        inputs:
          document: "Sample-{{myForLoop.value}}.yaml"
          phases: test
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

Output

AWSTOE erstellt eine Ausgabedatei, die `detailedoutput.json` bei jeder Ausführung aufgerufen wird. Die Datei enthält Details zu jeder Phase und jedem Schritt jedes Komponentendokuments, das während der Ausführung aufgerufen wird. Für das ExecuteDocumentAktionsmodul finden Sie in dem `outputs` Feld eine kurze Zusammenfassung der Laufzeit sowie Einzelheiten zu den Phasen, Schritten und Dokumenten, in denen es ausgeführt wird. `detailedOutput`

```
{
 \"executedStepCount\":1,\"executionId\":\"97054e22-06cc-11ec-9b14-acde48001122\",
\"failedStepCount\":0,\"failureMessage\":\"\",\"ignoredFailedStepCount\":0,\"logUrl\":
\"\",\"status\":\"success\"
}",
```

Das Ausgabe-Übersichtsobjekt jedes Komponentendokuments enthält die folgenden Details, wie hier gezeigt, mit Beispielwerten:

- executedStepCount„:1
- „Ausführungs-ID“: „12345a67-89bc-01de-2f34-abcd56789012"
- failedStepCount„: 0

- „Fehlermeldung“: ""
- „ignoredFailedStepAnzahl“ :0
- „Protokoll-URL“: ""
- „status“: „erfolgreich“

Ausgabebeispiel

Das folgende Beispiel zeigt die Ausgabe des ExecuteDocumentAktionsmoduls, wenn eine verschachtelte Ausführung stattfindet. In diesem Beispiel führt das `main.yaml` Komponentendokument das Komponentendokument erfolgreich aus. `Sample-1.yaml`


```
{
    "executionId": "12345a67-89bc-01de-2f34-abcd56789012",
    "status": "success",
    "startTime": "2021-08-26T17:20:31-07:00",
    "endTime": "2021-08-26T17:20:31-07:00",
    "failureMessage": "",
    "documents": [
        {
            "name": "",
            "filePath": "main.yaml",
            "status": "success",
            "description": "",
            "startTime": "2021-08-26T17:20:31-07:00",
            "endTime": "2021-08-26T17:20:31-07:00",
            "failureMessage": "",
            "phases": [
                {
                    "name": "build",
                    "status": "success",
                    "startTime": "2021-08-26T17:20:31-07:00",
                    "endTime": "2021-08-26T17:20:31-07:00",
                    "failureMessage": "",
                    "steps": [
                        {
                            "name": "ExecuteNestedDocument",
                            "status": "success",
                            "failureMessage": "",
                            "timeoutSeconds": -1,
                            "onFailure": "Abort",
                            "maxAttempts": 1,
```

```
                            "action": "ExecuteDocument",
                            "startTime": "2021-08-26T17:20:31-07:00",
                            "endTime": "2021-08-26T17:20:31-07:00",
                            "inputs": "[{\"document\":\"Sample-1.yaml\",\"document-s3-
bucket-owner\":\"\",\"phases\":\"\",\"parameters\":null}]",
                            "outputs": "[{\"executedStepCount\":1,\"executionId\":
\"98765f43-21ed-09cb-8a76-fedc54321098\",\"failedStepCount\":0,\"failureMessage\":\"\",
\"ignoredFailedStepCount\":0,\"logUrl\":\"\",\"status\":\"success\"}]",
                            "loop": null,
                            "detailedOutput": [
                                {
                                    "executionId": "98765f43-21ed-09cb-8a76-
fedc54321098",
                                    "status": "success",
                                    "startTime": "2021-08-26T17:20:31-07:00",
                                    "endTime": "2021-08-26T17:20:31-07:00",
                                    "failureMessage": "",
                                    "documents": [
                                        {
                                            "name": "",
                                            "filePath": "Sample-1.yaml",
                                            "status": "success",
                                            "description": "",
                                            "startTime": "2021-08-26T17:20:31-07:00",
                                            "endTime": "2021-08-26T17:20:31-07:00",
                                            "failureMessage": "",
                                            "phases": [
                                                {
                                                    "name": "build",
                                                    "status": "success",
                                                    "startTime":
 "2021-08-26T17:20:31-07:00",
                                                    "endTime":
 "2021-08-26T17:20:31-07:00",
                                                    "failureMessage": "",
                                                    "steps": [
                                                        {
                                                            "name": "ExecuteBashStep",
                                                            "status": "success",
                                                            "failureMessage": "",
                                                            "timeoutSeconds": 7200,
                                                            "onFailure": "Abort",
                                                            "maxAttempts": 1,
                                                            "action": "ExecuteBash",
```

```
                                                                "startTime":
 "2021-08-26T17:20:31-07:00",
                                                                "endTime":
 "2021-08-26T17:20:31-07:00",
                                                                "inputs": "[{\"commands\":
[\"echo \\\"Hello World!\\\"\"]}]",
                                                                "outputs": "[{\"stdout\":
\"Hello World!\"}]",
                                                                "loop": null,
                                                                "detailedOutput": null
                                                            }]
                                                        }]
                                                    }]
                                                }]
                                            }]

                                    }]
                                }]
}
```

ExecutePowerShell

Das ExecutePowerShellAktionsmodul ermöglicht es Ihnen, PowerShell Skripts mit Inline-Shell-Code/Befehlen auszuführen. Dieses Modul unterstützt die Windows-Plattform und Windows. PowerShell

Alle im Befehlsblock angegebenen Befehle/Anweisungen werden in eine Skriptdatei konvertiert (z. B.`input.ps1`) und unter Windows ausgeführt. PowerShell Das Ergebnis der Ausführung der Shell-Datei ist der Exit-Code.

Das ExecutePowerShellModul verarbeitet Systemneustarts, wenn der Shell-Befehl mit dem Exit-Code von beendet wird. `3010` Nach der Initiierung führt das Programm eine der folgenden Aktionen aus:

- Übergibt dem Anrufer den Exit-Code, wenn er vom Systems Manager Agent ausgeführt wird. Der Systems Manager Agent führt den Systemneustart durch und führt denselben Schritt aus, der den Neustart initiiert hat, wie unter Managed Instance from Scripts neu starten beschrieben.
- Speichert den aktuellen Status`executionstate`, konfiguriert einen Neustart-Trigger, um die Anwendung erneut auszuführen, und startet das System neu.

Nach dem Systemneustart führt die Anwendung denselben Schritt aus, der den Neustart initiiert hat. Wenn Sie diese Funktionalität benötigen, müssen Sie idempotente Skripten schreiben, die mehrere Aufrufe desselben Shell-Befehls verarbeiten können.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich
`commands`	Enthält eine Liste von Anweisungen oder Befehlen, die gemäß der PowerShell Syntax ausgeführt werden sollen. Mehrzeiliges YAML ist zulässig.	String-Liste	Ja. Muss `commands` oder angeben`file`, nicht beides.
`file`	Enthält den Pfad zu einer PowerShell Skriptdatei. PowerShell wird mit dem `-file` Befehlszeilenargument gegen diese Datei ausgeführt. Der Pfad muss auf eine `.ps1` Datei verweisen.	String	Ja. Muss `commands` oder`file`, nicht beides, angeben.

Eingabebeispiel: Vor und nach einem Neustart

```
name: ExitCode3010Example
description: This shows how the exit code can be used to restart a system with
 ExecutePowerShell
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: RestartTrigger
        action: ExecutePowerShell
        inputs:
          commands:
            - |
```

```
              $rebootIndicator = Join-Path -Path $env:SystemDrive -ChildPath 'reboot-
indicator'
              if (Test-Path -Path $rebootIndicator) {
                Write-Host 'The reboot file exists. Deleting it and exiting with
 success.'
                Remove-Item -Path $rebootIndicator -Force | Out-Null
                [System.Environment]::Exit(0)
              }
              Write-Host 'The reboot file does not exist. Creating it and triggering a
 restart.'
              New-Item -Path $rebootIndicator -ItemType File | Out-Null
              [System.Environment]::Exit(3010)
```

Output

Feld	Beschreibung	Typ
`stdout`	Standardausgabe der Befehlsausführung.	Zeichenfolge

Wenn Sie einen Neustart ausführen und den Exit-Code `3010` als Teil des Aktionsmoduls zurückgeben, wird der Build mit demselben Aktionsmodulschritt fortgesetzt, der den Neustart initiiert hat. Wenn Sie einen Neustart ohne den Exit-Code ausführen, schlägt der Build-Prozess möglicherweise fehl.

Ausgabebeispiel: Vor dem Neustart (zum ersten Mal über das Dokument)

```
{
 “stdout”: “The reboot file does not exist. Creating it and triggering a restart."
}
```

Ausgabebeispiel: Nach dem Neustart (zweites Mal durch das Dokument)

```
{
 “stdout”: “The reboot file exists. Deleting it and exiting with success."
}
```

Module zum Herunterladen und Hochladen von Dateien

Der folgende Abschnitt enthält Informationen zu Aktionsmodulen, die Befehle und Anweisungen zum Herunterladen und Hochladen ausführen.

Laden Sie Aktionsmodule herunter und laden Sie sie hoch

- S3 herunterladen
- S3 Upload
- WebDownload

S3 herunterladen

Mit dem `S3Download` Aktionsmodul können Sie ein Amazon S3 S3-Objekt oder eine Reihe von Objekten in eine lokale Datei oder einen Ordner herunterladen, den Sie mit dem `destination` Pfad angeben. Wenn am angegebenen Speicherort bereits eine Datei vorhanden ist und das `overwrite` Flag auf true gesetzt ist, wird die Datei `S3Download` überschrieben.

Ihr `source` Standort kann auf ein bestimmtes Objekt in Amazon S3 verweisen, oder Sie können ein key prefix mit einem Sternchen als Platzhalter (*) verwenden, um eine Reihe von Objekten herunterzuladen, die dem Schlüsselpräfixpfad entsprechen. Wenn Sie an Ihrem `source` Standort ein key prefix angeben, lädt das `S3Download` Aktionsmodul alles herunter, was dem Präfix entspricht (einschließlich Dateien und Ordner). Stellen Sie sicher, dass das key prefix mit einem Schrägstrich endet, gefolgt von einem Sternchen (/*), damit Sie alles herunterladen, was dem Präfix entspricht. Zum Beispiel: *`s3://my-bucket/my-folder/*`*.

> **Note**
>
> Alle Ordner im Zielpfad müssen vor dem Herunterladen vorhanden sein, andernfalls schlägt der Download fehl.

Wenn die `S3Download` Aktion für ein bestimmtes key prefix während eines Downloads fehlschlägt, wird der Ordnerinhalt nicht auf den Zustand vor dem Fehler zurückgesetzt. Der Zielordner bleibt so, wie er zum Zeitpunkt des Fehlers war.

Unterstützte Anwendungsfälle

Das `S3Download` Aktionsmodul unterstützt die folgenden Anwendungsfälle:

- Das Amazon S3 S3-Objekt wird in einen lokalen Ordner heruntergeladen, wie im Download-Pfad angegeben.
- Amazon S3 S3-Objekte (mit einem key prefix im Amazon S3 S3-Dateipfad) werden in den angegebenen lokalen Ordner heruntergeladen, der rekursiv alle Amazon S3 S3-Objekte, die dem key prefix entsprechen, in den lokalen Ordner kopiert.

IAM-Anforderungen

Die IAM-Rolle, die Sie Ihrem Instanzprofil zuordnen, muss über Berechtigungen zum Ausführen des `S3Download` Aktionsmoduls verfügen. Die folgenden IAM-Richtlinien müssen der IAM-Rolle angehängt werden, die dem Instanzprofil zugeordnet ist:

- Einzelne Datei: für den `s3:GetObject` Bucket/das Objekt (z. B.). `arn:aws:s3:::BucketName/*`
- Mehrere Dateien: `s3:ListBucket` gegen den Bucket/das Objekt (zum Beispiel`arn:aws:s3:::BucketName`) und `s3:GetObject` gegen den Bucket/das Objekt (zum Beispiel). `arn:aws:s3:::BucketName/*`

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`source`	Der Amazon S3 S3-Bucket, der die Quelle für Ihren Download ist. Sie können einen Pfad zu einem bestimmten Objekt angeben oder ein key prefix verwenden , das mit einem Schrägstrich endet, gefolgt	String	Ja	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standard
	von einem Sternchen-Platzhalter (`/*`), um eine Gruppe von Objekten herunterzuladen, die dem key prefix entsprechen.			
`destination`	Der lokale Pfad, in den die Amazon S3 S3-Objekte heruntergeladen werden. Um eine einzelne Datei herunterzuladen, müssen Sie den Dateinamen als Teil des Pfads angeben. z. B. *`/myfolder/package.zip`* .	String	Ja	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`expectedBucketOwner`	Erwartete Besitzerkonto-ID des im `source` Pfad angegebenen Buckets. Wir empfehlen Ihnen, die Inhaberschaft des in der Quelle angegebenen Amazon S3 S3-Buckets zu überprüfen.	String	Nein	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`overwrite`	Wenn diese Einstellung auf true gesetzt ist und bereits eine Datei mit demselben Namen im Zielordner für den angegeben en lokalen Pfad existiert , überschreibt die Download-Datei die lokale Datei. Wenn der Wert auf false gesetzt ist, ist die vorhanden e Datei auf dem lokalen System vor dem Überschreiben geschützt, und das Aktionsmo dul schlägt mit einem Download-Fehler fehl. Beispiel: `Error: S3Download: File already exists and`	Boolesch	Nein	true

Primitiv	Beschreibung	Typ	Erforderlich	Standard
	"overwrite" property for "destination" file is set to false. Cannot download.			

> **Note**
>
> In den folgenden Beispielen kann der Windows-Ordnerpfad durch einen Linux-Pfad ersetzt werden. *C:\myfolder\package.zip*Kann beispielsweise durch ersetzt werden*/myfolder/package.zip*.

Eingabebeispiel: Kopieren Sie ein Amazon S3 S3-Objekt in eine lokale Datei

Das folgende Beispiel zeigt, wie ein Amazon S3 S3-Objekt in eine lokale Datei kopiert wird.

```
- name: DownloadMyFile
  action: S3Download
  inputs:
    - source: s3://mybucket/path/to/package.zip
      destination: C:\myfolder\package.zip
      expectedBucketOwner: 123456789022
      overwrite: false
    - source: s3://mybucket/path/to/package.zip
      destination: C:\myfolder\package.zip
      expectedBucketOwner: 123456789022
      overwrite: true
    - source: s3://mybucket/path/to/package.zip
      destination: C:\myfolder\package.zip
      expectedBucketOwner: 123456789022
```

Eingabebeispiel: Alle Amazon S3 S3-Objekte in einem Amazon S3 S3-Bucket mit key prefix in einen lokalen Ordner kopieren

Das folgende Beispiel zeigt, wie alle Amazon S3 S3-Objekte in einem Amazon S3-Bucket mit dem key prefix in einen lokalen Ordner kopiert werden. Amazon S3 hat kein Ordnerkonzept, daher werden alle Objekte kopiert, die dem key prefix entsprechen. Die maximale Anzahl von Objekten, die heruntergeladen werden können, beträgt 1000.

```
- name: MyS3DownloadKeyprefix
  action: S3Download
  maxAttempts: 3
  inputs:
    - source: s3://mybucket/path/to/*
      destination: C:\myfolder\
      expectedBucketOwner: 123456789022
      overwrite: false
    - source: s3://mybucket/path/to/*
      destination: C:\myfolder\
      expectedBucketOwner: 123456789022
      overwrite: true
    - source: s3://mybucket/path/to/*
      destination: C:\myfolder\
      expectedBucketOwner: 123456789022
```

Output

Keine.

S3 Upload

Mit dem Aktionsmodul S3Upload können Sie eine Datei aus einer Quelldatei oder einem Quellordner an einen Amazon S3 S3-Speicherort hochladen. Sie können einen Platzhalter (*) in dem für Ihren Quellspeicherort angegebenen Pfad verwenden, um alle Dateien hochzuladen, deren Pfad dem Platzhaltermuster entspricht.

Wenn die rekursive S3Upload-Aktion fehlschlägt, verbleiben alle Dateien, die bereits hochgeladen wurden, im Amazon S3-Ziel-Bucket.

Unterstützte Anwendungsfälle

- Lokale Datei zum Amazon S3 S3-Objekt.

- Lokale Dateien im Ordner (mit Platzhalter) zum Amazon S3 S3-Schlüsselpräfix.
- Kopieren Sie den lokalen Ordner (muss auf `recurse` eingestellt sein`true`) in das Amazon S3 S3-Schlüsselpräfix.

IAM-Anforderungen

Die IAM-Rolle, die Sie Ihrem Instanzprofil zuordnen, muss über Berechtigungen zum Ausführen des `S3Upload` Aktionsmoduls verfügen. Die folgende IAM-Richtlinie muss der IAM-Rolle angehängt werden, die dem Instanzprofil zugeordnet ist. Die Richtlinie muss dem Amazon S3 S3-Ziel-Bucket `s3:PutObject` Berechtigungen gewähren. Beispiel, `arn:aws:s3:::`**`BucketName`**`/*`).

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`source`	Der lokale Pfad, aus dem die Quelldateien/-ordner stammen. Der `source` unterstützt einen Sternchen-Platzhalter (). *	String	Ja	N/A
`destination`	Der Pfad für den Amazon S3-Ziel-Bucket, in den Quelldateien/-ordner hochgeladen werden.	String	Ja	N/A
`recurse`	Wenn auf gesetzt, wird `true` S3Upload	String	Nein	`false`

Primitiv	Beschreibung	Typ	Erforderlich	Standard
	rekursiv ausgeführt.			
`expectedBucketOwner`	Die erwartete Besitzerkonto-ID für den Amazon S3 S3-Bucket, der im Zielpfad angegeben ist. Wir empfehlen Ihnen, die Inhaberschaft des Amazon S3 S3-Buckets zu überprüfen, der im Ziel angegeben ist.	String	Nein	N/A

Eingabebeispiel: Eine lokale Datei in ein Amazon S3 S3-Objekt kopieren

Das folgende Beispiel zeigt, wie eine lokale Datei in ein Amazon S3 S3-Objekt kopiert wird.

```
- name: MyS3UploadFile
  action: S3Upload
  onFailure: Abort
  maxAttempts: 3
  inputs:
    - source: C:\myfolder\package.zip
      destination: s3://mybucket/path/to/package.zip
      expectedBucketOwner: 123456789022
```

Eingabebeispiel: Kopieren Sie alle Dateien in einem lokalen Ordner in einen Amazon S3 S3-Bucket mit key prefix

Das folgende Beispiel zeigt, wie Sie alle Dateien im lokalen Ordner in einen Amazon S3 S3-Bucket mit key prefix kopieren. In diesem Beispiel werden keine Unterordner oder deren Inhalt kopiert, da dies nicht angegeben `recurse` ist. Die Standardeinstellung ist. `false`

```
- name: MyS3UploadMultipleFiles
  action: S3Upload
  onFailure: Abort
  maxAttempts: 3
  inputs:
    - source: C:\myfolder\*
      destination: s3://mybucket/path/to/
      expectedBucketOwner: 123456789022
```

Eingabebeispiel: Alle Dateien und Ordner rekursiv von einem lokalen Ordner in einen Amazon S3 S3-Bucket kopieren

Das folgende Beispiel zeigt, wie alle Dateien und Ordner rekursiv von einem lokalen Ordner in einen Amazon S3 S3-Bucket mit key prefix kopiert werden.

```
- name: MyS3UploadFolder
  action: S3Upload
  onFailure: Abort
  maxAttempts: 3
  inputs:
    - source: C:\myfolder\*
      destination: s3://mybucket/path/to/
      recurse: true
      expectedBucketOwner: 123456789022
```

Output

Keine.

WebDownload

Das WebDownloadAktionsmodul ermöglicht es Ihnen, Dateien und Ressourcen von einem entfernten Standort über das HTTP/HTTPS-Protokoll herunterzuladen (HTTPS wird empfohlen). Die Anzahl oder Größe der Downloads ist unbegrenzt. Dieses Modul verarbeitet Wiederholungsversuche und exponentielle Backoff-Logik.

Jedem Download-Vorgang werden je nach Benutzereingabe maximal 5 Versuche zugewiesen, um erfolgreich zu sein. Diese Versuche unterscheiden sich von den im `maxAttempts` Dokumentfeld angegebenen Versuchen`steps`, die sich auf Fehler im Aktionsmodul beziehen.

Dieses Aktionsmodul verarbeitet implizit Weiterleitungen. Alle HTTP-Statuscodes mit Ausnahme `200` von führen zu einem Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`source`	Die gültige HTTP/HTTPS-URL (HTTPS wird empfohlen), die dem RFC 3986-Standard entspricht. Verkettungsausdrücke sind zulässig.	String	Ja	N/A
`destination`	Ein absoluter oder relativer Datei- oder Ordnerpfad auf dem lokalen System. Ordnerpfade müssen mit enden`/`. Wenn sie nicht mit enden`/`, werden sie als Dateipfade behandelt. Das Modul erstellt alle erforderlichen Dateien	String	Ja	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standard
	oder Ordner für erfolgreiche Downloads. Verkettungsausdrücke sind zulässig.			

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`overwrite`	Wenn diese Option aktiviert ist, werden alle vorhandenen Dateien auf dem lokalen System mit der heruntergeladenen Datei oder Ressource überschrieben. Wenn diese Option nicht aktiviert ist, werden alle vorhandenen Dateien auf dem lokalen System nicht überschrieben, und das Aktionsmodul schlägt mit einem Fehler fehl. Wenn Überschreiben aktiviert ist und Prüfsumme und Algorithmus angegeben sind, lädt das Aktionsmodul die Datei nur herunter, wenn die Prüfsumme und der Hash	Boolesch	Nein	`true`

Primitiv	Beschreibung	Typ	Erforderlich	Standard
	aller bereits vorhandenen Dateien nicht übereinstimmen.			
`checksum`	Wenn Sie die Prüfsumme angeben, wird sie mit dem Hash der heruntergeladenen Datei verglichen, der mit dem mitgelieferten Algorithmus generiert wurde. Damit die Dateiüberprüfung aktiviert werden kann, müssen sowohl die Prüfsumme als auch der Algorithmus angegeben werden. Verkettungsausdrücke sind zulässig.	String	Nein	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`algorithm`	Der zur Berechnung der Prüfsumme verwendete Algorithmus. Die Optionen sind MD5, SHA1, SHA256 und SHA512. Damit die Dateiüber prüfung aktiviert werden kann, müssen sowohl die Prüfsumme als auch der Algorithmus angegeben werden. Verkettun gsausdrücke sind zulässig.	String	Nein	N/A
`ignoreCer tificateE rrors`	Die SSL-Zerti fikatsvalidierung wird ignoriert, wenn sie aktiviert ist.	Boolesch	Nein	`false`

Output

Primitiv	Beschreib ung	Typ				
`destinati on`	Zeichenfo lge durch	String				

Primitiv	Beschreibung	Typ				
	Zeilenumbruch, die den Zielpfad angibt, in dem die heruntergeladenen Dateien oder Ressourcen gespeichert sind.					

Eingabebeispiel: Laden Sie die Remote-Datei in ein lokales Ziel herunter

```
  - name: DownloadRemoteFile
    action: WebDownload
    maxAttempts: 3
    inputs:
      - source: https://testdomain/path/to/java14.zip
        destination: C:\testfolder\package.zip
```

Ausgabe:

```
{
 "destination": "C:\\testfolder\\package.zip"
}
```

Eingabebeispiel: Laden Sie mehr als eine Remote-Datei an mehr als ein lokales Ziel herunter

```
  - name: DownloadRemoteFiles
    action: WebDownload
    maxAttempts: 3
```

```
    inputs:
      - source: https://testdomain/path/to/java14.zip
        destination: /tmp/java14_renamed.zip
      - source: https://testdomain/path/to/java14.zip
        destination: /tmp/create_new_folder_and_add_java14_as_zip/
```

Ausgabe:

```
{
 "destination": "/tmp/create_new_folder/java14_renamed.zip\n/tmp/
create_new_folder_and_add_java14_as_zip/java14.zip"
}
```

Eingabebeispiel: Laden Sie eine Remote-Datei herunter, ohne das lokale Ziel zu überschreiben, und laden Sie eine weitere Remote-Datei mit Dateiüberprüfung herunter

```
  - name: DownloadRemoteMultipleProperties
    action: WebDownload
    maxAttempts: 3
    inputs:
      - source: https://testdomain/path/to/java14.zip
        destination: C:\create_new_folder\java14_renamed.zip
        overwrite: false
      - source: https://testdomain/path/to/java14.zip
        destination: C:\create_new_folder_and_add_java14_as_zip\
        checksum: ac68bbf921d953d1cfab916cb6120864
        algorithm: MD5
        overwrite: true
```

Ausgabe:

```
{
 "destination": "C:\\create_new_folder\\java14_renamed.zip\nC:\
\create_new_folder_and_add_java14_as_zip\\java14.zip"
}
```

Eingabebeispiel: Remote-Datei herunterladen und SSL-Zertifizierungsvalidierung ignorieren

```
  - name: DownloadRemoteIgnoreValidation
    action: WebDownload
```

```
    maxAttempts: 3
    inputs:
      - source: https://www.bad-ssl.com/resource
        destination: /tmp/downloads/
        ignoreCertificateErrors: true
```

Ausgabe:

```
{
 "destination": "/tmp/downloads/resource"
}
```

Betriebsmodule für das Dateisystem

Der folgende Abschnitt enthält Einzelheiten zu Aktionsmodulen, die Befehle und Anweisungen für den Betrieb des Dateisystems ausführen.

Aktionsmodule für den Betrieb des Dateisystems

- AppendFile
- CopyFile
- CopyFolder
- CreateFile
- CreateFolder
- CreateSymlink
- DeleteFile
- DeleteFolder
- ListFiles
- MoveFile
- MoveFolder
- ReadFile
- SetFileEncoding
- SetFileOwner
- SetFolderOwner

- SetFilePermissions
- SetFolderPermissions

AppendFile

Das AppendFileAktionsmodul fügt dem bereits vorhandenen Inhalt einer Datei den angegebenen Inhalt hinzu.

Wenn sich der Dateicodierungswert vom Standardwert encoding (`utf-8`) unterscheidet, können Sie den Dateicodierungswert mithilfe der `encoding` Option angeben. Standardmäßig wird davon ausgegangen, `utf-16` dass `utf-32` sie die Little-Endian-Kodierung verwenden.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Die angegebene Datei ist zur Laufzeit nicht vorhanden.
- Sie haben keine Schreibberechtigungen, um den Dateiinhalt zu ändern.
- Das Modul stößt während des Dateivorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`path`	Der Dateipfad.	String	Ja	–	N/A	Ja
`content`	Der Inhalt, der an die Datei angehängt werden soll.	String	Nein	Leere Zeichenfolge	N/A	Ja
`encoding`	Der Kodierungsstandard.	String	Nein	`utf8`	`utf8`,`utf-8-LE`, `utf-16-`	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
					LE utf16-BE,utf-16-BE ,utf32,utfLE,utf-32-LE ,utf32-BE, und utf-32-BE . Der Wert der Kodierungsoption unterscheidet nicht zwischen Groß- und Kleinschreibung.	

Eingabebeispiel: Datei ohne Kodierung anhängen (Linux)

```
- name: AppendingFileWithOutEncodingLinux
  action: AppendFile
  inputs:
    - path: ./Sample.txt
      content: "The string to be appended to the file"
```

Eingabebeispiel: Datei ohne Kodierung anhängen (Windows)

```
- name: AppendingFileWithOutEncodingWindows
```

```
  action: AppendFile
  inputs:
    - path: C:\MyFolder\MyFile.txt
      content: "The string to be appended to the file"
```

Eingabebeispiel: Datei mit Kodierung anhängen (Linux)

```
- name: AppendingFileWithEncodingLinux
  action: AppendFile
  inputs:
    - path: /FolderName/SampleFile.txt
      content: "The string to be appended to the file"
      encoding: UTF-32
```

Eingabebeispiel: Datei mit Kodierung anhängen (Windows)

```
- name: AppendingFileWithEncodingWindows
  action: AppendFile
  inputs:
    - path: C:\MyFolderName\SampleFile.txt
      content: "The string to be appended to the file"
      encoding: UTF-32
```

Eingabebeispiel: Datei mit leerer Zeichenfolge anhängen (Linux)

```
- name: AppendingEmptyStringLinux
  action: AppendFile
  inputs:
    - path: /FolderName/SampleFile.txt
```

Eingabebeispiel: Datei mit leerer Zeichenfolge anhängen (Windows)

```
- name: AppendingEmptyStringWindows
  action: AppendFile
  inputs:
    - path: C:\MyFolderName\SampleFile.txt
```

Output

Keine.

CopyFile

Das CopyFileAktionsmodul kopiert Dateien von der angegebenen Quelle zum angegebenen Ziel. Standardmäßig erstellt das Modul den Zielordner rekursiv, wenn er zur Laufzeit nicht vorhanden ist.

Wenn eine Datei mit dem angegebenen Namen bereits im angegebenen Ordner vorhanden ist, überschreibt das Aktionsmodul standardmäßig die vorhandene Datei. Sie können dieses Standardverhalten außer Kraft setzen, indem Sie die Option Überschreiben auf setzen. `false` Wenn die Option zum Überschreiben auf `false` gesetzt ist und sich am angegebenen Speicherort bereits eine Datei mit dem angegebenen Namen befindet, gibt das Aktionsmodul einen Fehler zurück. Diese Option funktioniert genauso wie der `cp` Befehl unter Linux, der standardmäßig überschreibt.

Der Name der Quelldatei kann einen Platzhalter () * enthalten. Platzhalterzeichen werden nur nach dem letzten Dateipfad-Trennzeichen (/oder\) akzeptiert. Wenn der Quelldateiname Platzhalterzeichen enthält, werden alle Dateien, die dem Platzhalter entsprechen, in den Zielordner kopiert. Wenn Sie mehr als eine Datei mithilfe eines Platzhalterzeichens verschieben möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (/oder\) enden, das angibt, dass es sich bei der Zieleingabe um einen Ordner handelt.

Wenn sich der Zieldateiname vom Quelldateinamen unterscheidet, können Sie den Zieldateinamen mithilfe der `destination` Option angeben. Wenn Sie keinen Zieldateinamen angeben, wird der Name der Quelldatei verwendet, um die Zieldatei zu erstellen. Jeder Text, der auf das letzte Dateipfadtrennzeichen (/oder\) folgt, wird als Dateiname behandelt. Wenn Sie denselben Dateinamen wie die Quelldatei verwenden möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (/oder\) enden.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, eine Datei im angegebenen Ordner zu erstellen.
- Die Quelldateien sind zur Laufzeit nicht vorhanden.
- Es gibt bereits einen Ordner mit dem angegebenen Dateinamen und die `overwrite` Option ist auf eingestellt`false`.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`source`	Der Pfad der Quelldatei.	String	Ja	–	N/A	Ja
`destination`	Der Zieldateipfad.	String	Ja	–	N/A	Ja
`overwrite`	Wenn der Wert auf false gesetzt ist, werden die Zieldateien nicht ersetzt, wenn sich am angegebenen Speicherort bereits eine Datei mit dem angegebenen Namen befindet.	Boolesch	Nein	`true`	N/A	Ja

Eingabebeispiel: Eine Datei kopieren (Linux)

```
- name: CopyingAFileLinux
  action: CopyFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/destinationFile.txt
```

Eingabebeispiel: Eine Datei kopieren (Windows)

```
- name: CopyingAFileWindows
  action: CopyFile
  inputs:
    - source: C:\MyFolder\Sample.txt
      destination: C:\MyFolder\destinationFile.txt
```

Eingabebeispiel: Kopiert eine Datei unter Verwendung des Quelldateinamens (Linux)

```
- name: CopyingFileWithSourceFileNameLinux
  action: CopyFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/
```

Eingabebeispiel: Kopieren Sie eine Datei unter Verwendung des Quelldateinamens (Windows)

```
- name: CopyingFileWithSourceFileNameWindows
  action: CopyFile
  inputs:
    - source: C:\Sample\MyFolder\Sample.txt
      destination: C:\MyFolder\
```

Eingabebeispiel: Kopiert eine Datei mit dem Platzhalterzeichen (Linux)

```
- name: CopyingFilesWithWildCardLinux
  action: CopyFile
  inputs:
    - source: /Sample/MyFolder/Sample*
      destination: /MyFolder/
```

Eingabebeispiel: Kopiert eine Datei mit dem Platzhalterzeichen (Windows)

```
- name: CopyingFilesWithWildCardWindows
  action: CopyFile
  inputs:
    - source: C:\Sample\MyFolder\Sample*
      destination: C:\MyFolder\
```

Eingabebeispiel: Eine Datei kopieren, ohne sie zu überschreiben (Linux)

```
- name: CopyingFilesWithoutOverwriteLinux
  action: CopyFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/destinationFile.txt
      overwrite: false
```

Eingabebeispiel: Eine Datei kopieren, ohne sie zu überschreiben (Windows)

```
- name: CopyingFilesWithoutOverwriteWindows
  action: CopyFile
  inputs:
    - source: C:\Sample\MyFolder\Sample.txt
      destination: C:\MyFolder\destinationFile.txt
      overwrite: false
```

Output

Keine.

CopyFolder

Das CopyFolderAktionsmodul kopiert einen Ordner von der angegebenen Quelle zum angegebenen Ziel. Die Eingabe für die `source` Option ist der zu kopierende Ordner, und die Eingabe für die `destination` Option ist der Ordner, in den der Inhalt des Quellordners kopiert wird. Standardmäßig erstellt das Modul den Zielordner rekursiv, wenn er zur Laufzeit nicht vorhanden ist.

Wenn im angegebenen Ordner bereits ein Ordner mit dem angegebenen Namen vorhanden ist, überschreibt das Aktionsmodul standardmäßig den vorhandenen Ordner. Sie können dieses Standardverhalten außer Kraft setzen, indem Sie die Option Überschreiben auf setzen. `false` Wenn die Option zum Überschreiben auf `false` gesetzt ist und sich am angegebenen Speicherort bereits ein Ordner mit dem angegebenen Namen befindet, gibt das Aktionsmodul einen Fehler zurück.

Der Name des Quellordners kann einen Platzhalter () * enthalten. Platzhalterzeichen werden nur nach dem letzten Dateipfad-Trennzeichen (/oder\) akzeptiert. Wenn Platzhalterzeichen im Quellordnernamen enthalten sind, werden alle Ordner, die dem Platzhalter entsprechen, in den Zielordner kopiert. Wenn Sie mehr als einen Ordner mithilfe eines Platzhalterzeichens kopieren möchten, muss die Eingabe für die `destination` Option mit einem Dateipfadtrennzeichen (/oder\) enden, was darauf hinweist, dass es sich bei der Zieleingabe um einen Ordner handelt.

Wenn sich der Name des Zielordners vom Namen des Quellordners unterscheidet, können Sie den Namen des Zielordners mithilfe der `destination` Option angeben. Wenn Sie keinen Zielordnernamen angeben, wird der Name des Quellordners verwendet, um den Zielordner zu erstellen. Jeder Text, der auf das letzte Dateipfadtrennzeichen (/oder\) folgt, wird als Ordnername behandelt. Wenn Sie denselben Ordnernamen wie den Quellordner verwenden möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (/oder\) enden.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, einen Ordner im angegebenen Ordner zu erstellen.
- Die Quellordner sind zur Laufzeit nicht vorhanden.
- Es gibt bereits einen Ordner mit dem angegebenen Ordnernamen und die `overwrite` Option ist auf gesetzt`false`.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`source`	Der Pfad des Quellordners.	String	Ja	–	N/A	Ja
`destination`	Der Pfad des	String	Ja	–	N/A	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
	Zielordners.					
`overwrite`	Wenn der Wert auf False gesetzt ist, werden die Zielordner nicht ersetzt, wenn sich am angegebenen Speicherort bereits ein Ordner mit dem angegebenen Namen befindet.	Boolesch	Nein	`true`	N/A	Ja

Eingabebeispiel: Einen Ordner kopieren (Linux)

```
- name: CopyingAFolderLinux
  action: CopyFolder
  inputs:
    - source: /Sample/MyFolder/SampleFolder
      destination: /MyFolder/destinationFolder
```

Eingabebeispiel: Einen Ordner kopieren (Windows)

```
- name: CopyingAFolderWindows
  action: CopyFolder
  inputs:
    - source: C:\Sample\MyFolder\SampleFolder
      destination: C:\MyFolder\destinationFolder
```

Eingabebeispiel: Kopieren Sie einen Ordner unter Verwendung des Quellordnernamens (Linux)

```
- name: CopyingFolderSourceFolderNameLinux
  action: CopyFolder
  inputs:
    - source: /Sample/MyFolder/SourceFolder
      destination: /MyFolder/
```

Eingabebeispiel: Kopieren Sie einen Ordner unter Verwendung des Quellordnernamens (Windows)

```
- name: CopyingFolderSourceFolderNameWindows
  action: CopyFolder
  inputs:
    - source: C:\Sample\MyFolder\SampleFolder
      destination: C:\MyFolder\
```

Eingabebeispiel: Kopieren Sie einen Ordner mit dem Platzhalterzeichen (Linux)

```
- name: CopyingFoldersWithWildCardLinux
  action: CopyFolder
  inputs:
    - source: /Sample/MyFolder/Sample*
      destination: /MyFolder/
```

Eingabebeispiel: Einen Ordner mit dem Platzhalterzeichen kopieren (Windows)

```
- name: CopyingFoldersWithWildCardWindows
  action: CopyFolder
  inputs:
    - source: C:\Sample\MyFolder\Sample*
      destination: C:\MyFolder\
```

Eingabebeispiel: Einen Ordner kopieren, ohne ihn zu überschreiben (Linux)

```
- name: CopyingFoldersWithoutOverwriteLinux
```

```
    action: CopyFolder
    inputs:
      - source: /Sample/MyFolder/SourceFolder
        destination: /MyFolder/destinationFolder
        overwrite: false
```

Eingabebeispiel: Einen Ordner kopieren, ohne ihn zu überschreiben (Windows)

```
- name: CopyingFoldersWithoutOverwrite
  action: CopyFolder
  inputs:
    - source: C:\Sample\MyFolder\SourceFolder
      destination: C:\MyFolder\destinationFolder
      overwrite: false
```

Output

Keine.

CreateFile

Das CreateFileAktionsmodul erstellt eine Datei an einem bestimmten Ort. Standardmäßig erstellt das Modul bei Bedarf auch rekursiv die übergeordneten Ordner.

Wenn die Datei bereits im angegebenen Ordner vorhanden ist, kürzt oder überschreibt das Aktionsmodul standardmäßig die vorhandene Datei. Sie können dieses Standardverhalten außer Kraft setzen, indem Sie die Option Überschreiben auf setzen. `false` Wenn die Option zum Überschreiben auf `false` gesetzt ist und sich am angegebenen Speicherort bereits eine Datei mit dem angegebenen Namen befindet, gibt das Aktionsmodul einen Fehler zurück.

Wenn sich der Wert für die Dateikodierung vom Standardwert für die Kodierung (`utf-8`) unterscheidet, können Sie den Wert für die Dateikodierung mithilfe der `encoding` Option angeben. Standardmäßig wird davon ausgegangen, `utf-16` dass `utf-32` sie die Little-Endian-Kodierung verwenden.

`owner`, und sind `group` optionale Eingaben. `permissions` Die Eingabe für `permissions` muss ein Zeichenkettenwert sein. Dateien werden mit Standardwerten erstellt, wenn sie nicht angegeben werden. Diese Optionen werden auf Windows-Plattformen nicht unterstützt. Dieses Aktionsmodul validiert und gibt einen Fehler zurück, wenn die `permissions` Optionen `ownergroup`, und auf Windows-Plattformen verwendet werden.

Dieses Aktionsmodul kann eine Datei mit Berechtigungen erstellen, die durch den `umask` Standardwert des Betriebssystems definiert sind. Sie müssen den `umask` Wert festlegen, wenn Sie den Standardwert überschreiben möchten.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, eine Datei oder einen Ordner im angegebenen übergeordneten Ordner zu erstellen.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`path`	Der Dateipfad.	String	Ja	–	N/A	Ja
`content`	Der Textinhalt der Datei.	String	Nein	N/A	N/A	Ja
`encoding`	Der Kodierungsstandard.	String	Nein	`utf8`	`utf8`,`utf-8-LE`, `utf-16-LE` `utf16-BE`,`utf-16-BE` ,`utf32`,`utfLE`,`utf-32-LE` ,`utf32-BE`, und `utf-32-BE` . Der	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
					Wert der Kodierungsoption unterscheidet nicht zwischen Groß- und Kleinschreibung.	
`owner`	Der Benutzername oder die ID.	String	Nein	N/A	N/A	Wird unter Windows nicht unterstützt.
`group`	Der Gruppenname oder die ID.	String	Nein	Der aktuelle Benutzer.	N/A	Wird unter Windows nicht unterstützt.
`permissions`	Die Dateiberechtigungen.	String	Nein	`0666`	N/A	Wird unter Windows nicht unterstützt.

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`overwrite`	Wenn der Name der angegebenen Datei bereits existiert, wird durch das Einstellen dieses Werts auf `false` verhindert, dass die Datei standardmäßig gekürzt oder überschrieben wird.	Boolesch	Nein	`true`	N/A	Ja

Eingabebeispiel: Eine Datei ohne Überschreiben erstellen (Linux)

```
- name: CreatingFileWithoutOverwriteLinux
  action: CreateFile
  inputs:
    - path: /home/UserName/Sample.txt
      content: The text content of the sample file.
      overwrite: false
```

Eingabebeispiel: Erstellen Sie eine Datei ohne Überschreiben (Windows)

```
- name: CreatingFileWithoutOverwriteWindows
  action: CreateFile
  inputs:
    - path: C:\Temp\Sample.txt
      content: The text content of the sample file.
      overwrite: false
```

Eingabebeispiel: Erstellen Sie eine Datei mit Dateieigenschaften

```
- name: CreatingFileWithFileProperties
  action: CreateFile
  inputs:
    - path: SampleFolder/Sample.txt
      content: The text content of the sample file.
      encoding: UTF-16
      owner: Ubuntu
      group: UbuntuGroup
      permissions: 0777
   - path: SampleFolder/SampleFile.txt
      permissions: 755
    - path: SampleFolder/TextFile.txt
      encoding: UTF-16
      owner: root
      group: rootUserGroup
```

Eingabebeispiel: Erstellen Sie eine Datei ohne Dateieigenschaften

```
- name: CreatingFileWithoutFileProperties
  action: CreateFile
  inputs:
    - path: ./Sample.txt
    - path: Sample1.txt
```

Eingabebeispiel: Erstellen Sie eine leere Datei, um einen Abschnitt im Linux-Bereinigungsskript zu überspringen

```
- name: CreateSkipCleanupfile
  action: CreateFile
  inputs:
    - path: <skip section file name>
```

Weitere Informationen finden Sie unter Überschreiben Sie das Linux-Bereinigungsskript.

Output

Keine.

CreateFolder

Das CreateFolderAktionsmodul erstellt einen Ordner an einem bestimmten Ort. Standardmäßig erstellt das Modul bei Bedarf auch rekursiv die übergeordneten Ordner.

Wenn der Ordner bereits im angegebenen Ordner vorhanden ist, kürzt oder überschreibt das Aktionsmodul standardmäßig den vorhandenen Ordner. Sie können dieses Standardverhalten außer Kraft setzen, indem Sie die Option Überschreiben auf setzen. `false` Wenn die Option zum Überschreiben auf `false` gesetzt ist und sich am angegebenen Speicherort bereits ein Ordner mit dem angegebenen Namen befindet, gibt das Aktionsmodul einen Fehler zurück.

`ownergroup`, und `permissions` sind optionale Eingaben. Die Eingabe für `permissions` muss ein Zeichenkettenwert sein. Diese Optionen werden auf Windows-Plattformen nicht unterstützt. Dieses Aktionsmodul validiert und gibt einen Fehler zurück, wenn die `permissions` Optionen `ownergroup`, und auf Windows-Plattformen verwendet werden.

Dieses Aktionsmodul kann einen Ordner mit Berechtigungen erstellen, die durch den `umask` Standardwert des Betriebssystems definiert sind. Sie müssen den `umask` Wert festlegen, wenn Sie den Standardwert überschreiben möchten.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, einen Ordner am angegebenen Speicherort zu erstellen.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`path`	Der Ordnerpfad.	String	Ja	–	N/A	Ja
`owner`	Der Benutzername oder die ID.	String	Nein	Der aktuelle Benutzer.	N/A	Wird unter Windows nicht unterstützt.
`group`	Der Gruppenname oder die ID.	String	Nein	Die Gruppe des aktuellen Benutzers.	N/A	Wird unter Windows nicht unterstützt.
`permissions`	Die Ordnerberechtigungen.	String	Nein	`0777`	N/A	Wird unter Windows nicht unterstützt.
`overwrite`	Wenn der Name der angegebenen Datei bereits existiert, wird durch das Einstellen dieses Werts auf `false`	Boolesch	Nein	`true`	N/A	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
	verhindert, dass die Datei standardmäßig gekürzt oder überschrieben wird.					

Eingabebeispiel: Erstellen Sie einen Ordner (Linux)

```
- name: CreatingFolderLinux
  action: CreateFolder
  inputs:
    - path: /Sample/MyFolder/
```

Eingabebeispiel: Erstellen Sie einen Ordner (Windows)

```
- name: CreatingFolderWindows
  action: CreateFolder
  inputs:
    - path: C:\MyFolder
```

Eingabebeispiel: Erstellen Sie einen Ordner, in dem die Ordnereigenschaften angegeben sind

```
- name: CreatingFolderWithFolderProperties
  action: CreateFolder
  inputs:
    - path: /Sample/MyFolder/Sample/
      owner: SampleOwnerName
      group: SampleGroupName
      permissions: 0777
    - path: /Sample/MyFolder/SampleFoler/
```

```
        permissions: 777
```

Eingabebeispiel: Erstellen Sie einen Ordner, der den vorhandenen Ordner überschreibt, falls es einen gibt.

```
- name: CreatingFolderWithOverwrite
  action: CreateFolder
  inputs:
    - path: /Sample/MyFolder/Sample/
      overwrite: true
```

Output

Keine.

CreateSymlink

Das CreateSymlinkAktionsmodul erstellt symbolische Links oder Dateien, die einen Verweis auf eine andere Datei enthalten. Dieses Modul wird auf Windows-Plattformen nicht unterstützt.

Die Eingabe für die `target` Optionen `path` und kann entweder ein absoluter oder ein relativer Pfad sein. Handelt es sich bei der Eingabe für die `path` Option um einen relativen Pfad, wird dieser bei der Erstellung des Links durch den absoluten Pfad ersetzt.

Wenn ein Link mit dem angegebenen Namen bereits im angegebenen Ordner vorhanden ist, gibt das Aktionsmodul standardmäßig einen Fehler zurück. Sie können dieses Standardverhalten außer Kraft setzen, indem Sie die `force` Option auf setzen`true`. Wenn die `force` Option auf gesetzt ist`true`, überschreibt das Modul den vorhandenen Link.

Wenn kein übergeordneter Ordner vorhanden ist, erstellt das Aktionsmodul den Ordner standardmäßig rekursiv.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Die Zieldatei ist zur Laufzeit nicht vorhanden.
- Eine nichtsymbolische Linkdatei mit dem angegebenen Namen ist bereits vorhanden.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`path`	Der Dateipfad.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`target`	Der Zieldateipfad, auf den der symbolische Link verweist.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`force`	Erzwingt die Erstellung eines Links, wenn bereits ein Link mit demselben Namen vorhanden ist.	Boolesch	Nein	`false`	N/A	Wird unter Windows nicht unterstützt.

Eingabebeispiel: Erstellen Sie einen symbolischen Link, der die Erstellung eines Links erzwingt

```
- name: CreatingSymbolicLinkWithForce
  action: CreateSymlink
  inputs:
```

```
      - path: /Folder2/Symboliclink.txt
        target: /Folder/Sample.txt
        force: true
```

Eingabebeispiel: Erstellen Sie einen symbolischen Link, der die Erstellung eines Links nicht erzwingt

```
- name: CreatingSymbolicLinkWithOutForce
  action: CreateSymlink
  inputs:
    - path: Symboliclink.txt
      target: /Folder/Sample.txt
```

Output

Keine.

DeleteFile

Das DeleteFileAktionsmodul löscht eine oder mehrere Dateien an einem bestimmten Ort.

Die Eingabe von `path` sollte ein gültiger Dateipfad oder ein Dateipfad mit einem Platzhalterzeichen (*) im Dateinamen sein. Wenn Platzhalterzeichen im Dateinamen angegeben werden, werden alle Dateien innerhalb desselben Ordners, die dem Platzhalter entsprechen, gelöscht.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, Löschvorgänge durchzuführen.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`path`	Der Dateipfad.	String	Ja	–	N/A	Ja

Eingabebeispiel: Eine einzelne Datei löschen (Linux)

```
- name: DeletingSingleFileLinux
  action: DeleteFile
  inputs:
    - path: /SampleFolder/MyFolder/Sample.txt
```

Eingabebeispiel: Eine einzelne Datei löschen (Windows)

```
- name: DeletingSingleFileWindows
  action: DeleteFile
  inputs:
    - path: C:\SampleFolder\MyFolder\Sample.txt
```

Eingabebeispiel: Lösche eine Datei, die mit „log“ endet (Linux)

```
- name: DeletingFileEndingWithLogLinux
  action: DeleteFile
  inputs:
    - path: /SampleFolder/MyFolder/*log
```

Eingabebeispiel: Lösche eine Datei, die mit „log“ endet (Windows)

```
- name: DeletingFileEndingWithLogWindows
  action: DeleteFile
  inputs:
    - path: C:\SampleFolder\MyFolder\*log
```

Eingabebeispiel: lösche alle Dateien in einem angegebenen Ordner (Linux)

```
- name: DeletingAllFilesInAFolderLinux
  action: DeleteFile
  inputs:
    - path: /SampleFolder/MyFolder/*
```

Eingabebeispiel: lösche alle Dateien in einem angegebenen Ordner (Windows)

```
- name: DeletingAllFilesInAFolderWindows
  action: DeleteFile
  inputs:
    - path: C:\SampleFolder\MyFolder\*
```

Output

Keine.

DeleteFolder

Das DeleteFolderAktionsmodul löscht Ordner.

Wenn der Ordner nicht leer ist, müssen Sie die `force` Option auf setzen, `true` um den Ordner und seinen Inhalt zu entfernen. Wenn Sie die `force` Option nicht auf setzen und der Ordner`true`, den Sie löschen möchten, nicht leer ist, gibt das Aktionsmodul einen Fehler zurück. Der Standardwert der `force` Option ist`false`.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, Löschvorgänge durchzuführen.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`path`	Der Ordnerpfad.	String	Ja	–	N/A	Ja
`force`	Löscht den Ordner, unabhängig davon, ob der Ordner leer ist oder nicht.	Boolesch	Nein	`false`	N/A	Ja

Eingabebeispiel: Löschen Sie einen Ordner, der nicht leer ist, mit der **`force`** Option (Linux)

```
  - name: DeletingFolderWithForceOptionLinux
```

```
    action: DeleteFolder
    inputs:
      - path: /Sample/MyFolder/Sample/
        force: true
```

Eingabebeispiel: Löschen Sie einen Ordner, der nicht leer ist, mit der **`force`** Option (Windows)

```
- name: DeletingFolderWithForceOptionWindows
  action: DeleteFolder
  inputs:
    - path: C:\Sample\MyFolder\Sample\
      force: true
```

Eingabebeispiel: einen Ordner löschen (Linux)

```
- name: DeletingFolderWithOutForceLinux
  action: DeleteFolder
  inputs:
    - path: /Sample/MyFolder/Sample/
```

Eingabebeispiel: Löschen eines Ordners (Windows)

```
- name: DeletingFolderWithOutForce
  action: DeleteFolder
  inputs:
    - path: C:\Sample\MyFolder\Sample\
```

Output

Keine.

ListFiles

Das ListFilesAktionsmodul listet die Dateien in einem angegebenen Ordner auf. Wenn die Option rekursiv auf gesetzt ist`true`, listet sie die Dateien in Unterordnern auf. Dieses Modul listet standardmäßig keine Dateien in Unterordnern auf.

Um alle Dateien aufzulisten, deren Namen einem bestimmten Muster entsprechen, verwenden Sie die `fileNamePattern` Option zur Angabe des Musters. Die `fileNamePattern` Option akzeptiert den Platzhalterwert (`*`). Wenn der angegeben `fileNamePattern` ist, werden alle Dateien zurückgegeben, die dem angegebenen Dateinamenformat entsprechen.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Der angegebene Ordner ist zur Laufzeit nicht vorhanden.
- Sie sind nicht berechtigt, eine Datei oder einen Ordner im angegebenen übergeordneten Ordner zu erstellen.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`path`	Der Ordnerpfad.	String	Ja	–	N/A	Ja
`fileNamePattern`	Das Muster, das abgeglichen werden soll, um alle Dateien aufzulisten, deren Namen dem Muster entsprechen.	String	Nein	N/A	N/A	Ja
`recursive`	Listet die Dateien im Ordner	Boolesch	Nein	`false`	N/A	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
	rekursiv auf.					

Eingabebeispiel: listet Dateien im angegebenen Ordner auf (Linux)

```
- name: ListingFilesInSampleFolderLinux
  action: ListFiles
  inputs:
    - path: /Sample/MyFolder/Sample
```

Eingabebeispiel: Dateien im angegebenen Ordner auflisten (Windows)

```
- name: ListingFilesInSampleFolderWindows
  action: ListFiles
  inputs:
    - path: C:\Sample\MyFolder\Sample
```

Eingabebeispiel: listet Dateien auf, die mit „log“ enden (Linux)

```
- name: ListingFilesWithEndingWithLogLinux
  action: ListFiles
  inputs:
    - path: /Sample/MyFolder/
      fileNamePattern: *log
```

Eingabebeispiel: listet Dateien auf, die mit „log“ enden (Windows)

```
- name: ListingFilesWithEndingWithLogWindows
  action: ListFiles
  inputs:
    - path: C:\Sample\MyFolder\
      fileNamePattern: *log
```

Eingabebeispiel: Dateien rekursiv auflisten

```
  - name: ListingFilesRecursively
    action: ListFiles
    inputs:
      - path: /Sample/MyFolder/
        recursive: true
```

Output

Primitiv	Beschreibung	Typ
`files`	Die Liste der Dateien.	String

Output example

```
{
 "files": "/sample1.txt,/sample2.txt,/sample3.txt"
}
```

MoveFile

Das MoveFileAktionsmodul verschiebt Dateien von der angegebenen Quelle zum angegebenen Ziel.

Wenn die Datei bereits im angegebenen Ordner vorhanden ist, überschreibt das Aktionsmodul standardmäßig die vorhandene Datei. Sie können dieses Standardverhalten außer Kraft setzen, indem Sie die Option Überschreiben auf setzen. `false` Wenn die Option zum Überschreiben auf `false` gesetzt ist und sich am angegebenen Speicherort bereits eine Datei mit dem angegebenen Namen befindet, gibt das Aktionsmodul einen Fehler zurück. Diese Option funktioniert genauso wie der `mv` Befehl unter Linux, der standardmäßig überschreibt.

Der Name der Quelldatei kann einen Platzhalter () * enthalten. Platzhalterzeichen werden nur nach dem letzten Dateipfad-Trennzeichen (/oder\) akzeptiert. Wenn der Quelldateiname Platzhalterzeichen enthält, werden alle Dateien, die dem Platzhalter entsprechen, in den Zielordner kopiert. Wenn Sie mehr als eine Datei mithilfe eines Platzhalterzeichens verschieben möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (/oder\) enden, das angibt, dass es sich bei der Zieleingabe um einen Ordner handelt.

Wenn sich der Zieldateiname vom Quelldateinamen unterscheidet, können Sie den Zieldateinamen mithilfe der `destination` Option angeben. Wenn Sie keinen Zieldateinamen angeben, wird der Name der Quelldatei verwendet, um die Zieldatei zu erstellen. Jeder Text, der auf das letzte Dateipfadtrennzeichen (/oder\) folgt, wird als Dateiname behandelt. Wenn Sie denselben Dateinamen wie die Quelldatei verwenden möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (/oder\) enden.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, eine Datei im angegebenen Ordner zu erstellen.
- Die Quelldateien sind zur Laufzeit nicht vorhanden.
- Es gibt bereits einen Ordner mit dem angegebenen Dateinamen und die `overwrite` Option ist auf eingestellt`false`.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`source`	Der Pfad der Quelldatei.	String	Ja	–	N/A	Ja
`destination`	Der Zieldateipfad.	String	Ja	–	N/A	Ja
`overwrite`	Wenn der Wert auf false gesetzt ist, werden die Zieldateien nicht ersetzt,	Boolesch	Nein	`true`	N/A	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
	wenn sich am angegebenen Speicherort bereits eine Datei mit dem angegebenen Namen befindet.					

Eingabebeispiel: Eine Datei verschieben (Linux)

```
- name: MovingAFileLinux
  action: MoveFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/destinationFile.txt
```

Eingabebeispiel: Eine Datei verschieben (Windows)

```
- name: MovingAFileWindows
  action: MoveFile
  inputs:
    - source: C:\Sample\MyFolder\Sample.txt
      destination: C:\MyFolder\destinationFile.txt
```

Eingabebeispiel: Verschieben Sie eine Datei unter Verwendung des Quelldateinamens (Linux)

```
- name: MovingFileWithSourceFileNameLinux
  action: MoveFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
```

```
      destination: /MyFolder/
```

Eingabebeispiel: Verschieben Sie eine Datei unter Verwendung des Quelldateinamens (Windows)

```
- name: MovingFileWithSourceFileNameWindows
  action: MoveFile
  inputs:
    - source: C:\Sample\MyFolder\Sample.txt
      destination: C:\MyFolder
```

Eingabebeispiel: Verschieben Sie eine Datei mit einem Platzhalterzeichen (Linux)

```
- name: MovingFilesWithWildCardLinux
  action: MoveFile
  inputs:
    - source: /Sample/MyFolder/Sample*
      destination: /MyFolder/
```

Eingabebeispiel: Verschieben Sie eine Datei mit einem Platzhalterzeichen (Windows)

```
- name: MovingFilesWithWildCardWindows
  action: MoveFile
  inputs:
    - source: C:\Sample\MyFolder\Sample*
      destination: C:\MyFolder
```

Eingabebeispiel: Eine Datei verschieben, ohne sie zu überschreiben (Linux)

```
- name: MovingFilesWithoutOverwriteLinux
  action: MoveFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/destinationFile.txt
      overwrite: false
```

Eingabebeispiel: Eine Datei verschieben, ohne sie zu überschreiben (Windows)

```
- name: MovingFilesWithoutOverwrite
  action: MoveFile
  inputs:
```

```
      - source: C:\Sample\MyFolder\Sample.txt
        destination: C:\MyFolder\destinationFile.txt
        overwrite: false
```

Output

Keine.

MoveFolder

Das MoveFolderAktionsmodul verschiebt Ordner von der angegebenen Quelle zum angegebenen Ziel. Die Eingabe für die `source` Option ist der zu verschiebende Ordner, und die Eingabe für die `destination` Option ist der Ordner, in den der Inhalt der Quellordner verschoben wird.

Wenn der übergeordnete Zielordner oder die Eingabe für die `destination` Option zur Laufzeit nicht vorhanden sind, besteht das Standardverhalten des Moduls darin, den Ordner rekursiv am angegebenen Ziel zu erstellen.

Wenn im Zielordner bereits ein Ordner vorhanden ist, der mit dem Quellordner identisch ist, überschreibt das Aktionsmodul standardmäßig den vorhandenen Ordner. Sie können dieses Standardverhalten außer Kraft setzen, indem Sie die Option Überschreiben auf setzen. `false` Wenn die Option zum Überschreiben auf `false` gesetzt ist und sich am angegebenen Speicherort bereits ein Ordner mit dem angegebenen Namen befindet, gibt das Aktionsmodul einen Fehler zurück.

Der Name des Quellordners kann einen Platzhalter () * enthalten. Platzhalterzeichen werden nur nach dem letzten Dateipfad-Trennzeichen (/oder\) akzeptiert. Wenn Platzhalterzeichen im Quellordnernamen enthalten sind, werden alle Ordner, die dem Platzhalter entsprechen, in den Zielordner kopiert. Wenn Sie mehr als einen Ordner mithilfe eines Platzhalterzeichens verschieben möchten, muss die Eingabe für die `destination` Option mit einem Dateipfadtrennzeichen (/oder\) enden, was darauf hinweist, dass es sich bei der Zieleingabe um einen Ordner handelt.

Wenn sich der Name des Zielordners vom Namen des Quellordners unterscheidet, können Sie den Namen des Zielordners mithilfe der `destination` Option angeben. Wenn Sie keinen Zielordnernamen angeben, wird der Name des Quellordners verwendet, um den Zielordner zu erstellen. Jeder Text, der auf das letzte Dateipfadtrennzeichen (/oder\) folgt, wird als Ordnername behandelt. Wenn Sie denselben Ordnernamen wie den Quellordner verwenden möchten, muss die Eingabe der `destination` Option mit einem Dateipfadtrennzeichen (/oder\) enden.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, einen Ordner im Zielordner zu erstellen.
- Die Quellordner sind zur Laufzeit nicht vorhanden.
- Es gibt bereits einen Ordner mit dem angegebenen Namen und die `overwrite` Option ist auf gesetzt`false`.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`source`	Der Pfad des Quellordners.	String	Ja	–	N/A	Ja
`destination`	Der Pfad des Zielordners.	String	Ja	–	N/A	Ja
`overwrite`	Wenn der Wert auf False gesetzt ist, werden die Zielordner nicht ersetzt, wenn sich am angegebenen Speicherort bereits	Boolesch	Nein	`true`	N/A	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
	ein Ordner mit dem angegebenen Namen befindet.					

Eingabebeispiel: Einen Ordner verschieben (Linux)

```
- name: MovingAFolderLinux
  action: MoveFolder
  inputs:
    - source: /Sample/MyFolder/SourceFolder
      destination: /MyFolder/destinationFolder
```

Eingabebeispiel: Einen Ordner verschieben (Windows)

```
- name: MovingAFolderWindows
  action: MoveFolder
  inputs:
    - source: C:\Sample\MyFolder\SourceFolder
      destination: C:\MyFolder\destinationFolder
```

Eingabebeispiel: Verschieben Sie einen Ordner unter Verwendung des Quellordnernamens (Linux)

```
- name: MovingFolderWithSourceFolderNameLinux
  action: MoveFolder
  inputs:
    - source: /Sample/MyFolder/SampleFolder
      destination: /MyFolder/
```

Eingabebeispiel: Verschieben Sie einen Ordner mithilfe des Quellordnernamens (Windows)

```
- name: MovingFolderWithSourceFolderNameWindows
```

```
  action: MoveFolder
  inputs:
    - source: C:\Sample\MyFolder\SampleFolder
      destination: C:\MyFolder\
```

Eingabebeispiel: Verschieben Sie einen Ordner mithilfe eines Platzhalterzeichens (Linux)

```
- name: MovingFoldersWithWildCardLinux
  action: MoveFolder
  inputs:
    - source: /Sample/MyFolder/Sample*
      destination: /MyFolder/
```

Eingabebeispiel: Verschieben Sie einen Ordner mithilfe eines Platzhalterzeichens (Windows)

```
- name: MovingFoldersWithWildCardWindows
  action: MoveFolder
  inputs:
    - source: C:\Sample\MyFolder\Sample*
      destination: C:\MyFolder\
```

Eingabebeispiel: Einen Ordner verschieben, ohne ihn zu überschreiben (Linux)

```
- name: MovingFoldersWithoutOverwriteLinux
  action: MoveFolder
  inputs:
    - source: /Sample/MyFolder/SampleFolder
  destination: /MyFolder/destinationFolder
  overwrite: false
```

Eingabebeispiel: Einen Ordner verschieben, ohne ihn zu überschreiben (Windows)

```
- name: MovingFoldersWithoutOverwriteWindows
  action: MoveFolder
  inputs:
    - source: C:\Sample\MyFolder\SampleFolder
      destination: C:\MyFolder\destinationFolder
      overwrite: false
```

Output

Keine.

ReadFile

Das ReadFileAktionsmodul liest den Inhalt einer Textdatei vom Typ Zeichenfolge. Dieses Modul kann verwendet werden, um den Inhalt einer Datei zur Verwendung in nachfolgenden Schritten durch Verkettung oder zum Lesen von Daten in die `console.log` Datei zu lesen. Wenn der angegebene Pfad ein symbolischer Link ist, gibt dieses Modul den Inhalt der Zieldatei zurück. Dieses Modul unterstützt nur Textdateien.

Wenn sich der Dateicodierungswert vom Standardwert encoding (`utf-8`) unterscheidet, können Sie den Dateicodierungswert mithilfe der `encoding` Option angeben. Standardmäßig wird davon ausgegangen, `utf-16` dass `utf-32` sie die Little-Endian-Kodierung verwenden.

Standardmäßig kann dieses Modul den Dateiinhalt nicht in die Datei drucken. `console.log` Sie können diese Einstellung überschreiben, indem Sie die `printFileContent` Eigenschaft auf setzen`true`.

Dieses Modul kann nur den Inhalt einer Datei zurückgeben. Es kann keine Dateien wie Excel- oder JSON-Dateien analysieren.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Die Datei ist zur Laufzeit nicht vorhanden.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`path`	Der Dateipfad.	String	Ja	–	N/A	Ja
`encoding`	Der Kodierungsstandard.	String	Nein	`utf8`	`utf8`,`utf-8-LE`, `utf-16-LE`	Ja

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
					`utf16-BE`,`utf-16-BE`,`utf32`,`utfLE`,`utf-32-LE`,`utf32-BE`, und `utf-32-BE`. Der Wert der Kodierungsoption unterscheidet nicht zwischen Groß- und Kleinschreibung.	
`printFileContent`	Druckt den Dateiinhalt in die `console.log` Datei.	Boolesch	Nein	false	N/A	Ja.

Eingabebeispiel: Eine Datei lesen (Linux)

```
- name: ReadingFileLinux
  action: ReadFile
  inputs:
    - path: /home/UserName/SampleFile.txt
```

Eingabebeispiel: eine Datei lesen (Windows)

```
- name: ReadingFileWindows
  action: ReadFile
  inputs:
    - path: C:\Windows\WindowsUpdate.log
```

Eingabebeispiel: Eine Datei lesen und den Kodierungsstandard angeben

```
- name: ReadingFileWithFileEncoding
  action: ReadFile
  inputs:
    - path: /FolderName/SampleFile.txt
      encoding: UTF-32
```

Eingabebeispiel: Eine Datei lesen und in die **`console.log`** Datei drucken

```
- name: ReadingFileToConsole
  action: ReadFile
  inputs:
    - path: /home/UserName/SampleFile.txt
      printFileContent: true
```

Output

Feld	Beschreibung	Typ
`content`	Der Inhalt der Datei.	Zeichenfolge

Output example

```
{
 "content" : "The file content"
}
```

SetFileEncoding

Das SetFileEncodingAktionsmodul ändert die Kodierungseigenschaft einer vorhandenen Datei. Dieses Modul kann die Dateikodierung von einem bestimmten Kodierungsstandard `utf-8` in einen

bestimmten Kodierungsstandard konvertieren. Standardmäßig wird davon ausgegangen, `utf-16` dass `utf-32` es sich um Little-Endian-Kodierung handelt.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, die angegebene Änderung durchzuführen.
- Die Datei ist zur Laufzeit nicht vorhanden.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt	
`path`	Der Dateipfad.	String	Ja	–	N/A	Ja	
`encoding`	Der Kodierungsstandard.	String	Nein	`utf8`	`utf8`,`utf-8-LE`, `utf-16-LE` `utf16-BE`,`utf-16-BE` ,`utf32`,`utf-LE`,`utf-32-LE` ,`utf32-BE`, und `utf-32-BE` . Der Wert der Kodierungsoption untersche	Ja	6,utf1

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
					idet nicht zwischen Groß- und Kleinschreibung.	

Eingabebeispiel: Legen Sie die Eigenschaft für die Dateikodierung fest

```
- name: SettingFileEncodingProperty
  action: SetFileEncoding
  inputs:
    - path: /home/UserName/SampleFile.txt
      encoding: UTF-16
```

Output

Keine.

SetFileOwner

Das SetFileOwnerAktionsmodul ändert die Eigenschaften `owner` und `group` Eigentümer einer vorhandenen Datei. Wenn es sich bei der angegebenen Datei um einen symbolischen Link handelt, ändert das Modul die `owner` Eigenschaft der Quelldatei. Dieses Modul wird auf Windows-Plattformen nicht unterstützt.

Dieses Modul akzeptiert Benutzer- und Gruppennamen als Eingaben. Wenn der Gruppenname nicht angegeben wird, weist das Modul den Gruppeneigentümer der Datei der Gruppe zu, zu der der Benutzer gehört.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, die angegebene Änderung durchzuführen.
- Der angegebene Benutzer- oder Gruppenname ist zur Laufzeit nicht vorhanden.
- Die Datei ist zur Laufzeit nicht vorhanden.

- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`path`	Der Dateipfad.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`owner`	Der Benutzername	Zeichenfolge	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`group`	Der Name der Benutzergruppe.	String	Nein	Der Name der Gruppe, zu der der Benutzer gehört.	N/A	Wird unter Windows nicht unterstützt.

Eingabebeispiel: Legen Sie die Eigenschaft des Dateibesitzers fest, ohne den Namen der Benutzergruppe anzugeben

```
- name: SettingFileOwnerPropertyNoGroup
  action: SetFileOwner
  inputs:
    - path: /home/UserName/SampleText.txt
      owner: LinuxUser
```

Eingabebeispiel: Legen Sie die Eigenschaft des Dateibesitzers fest, indem Sie den Eigentümer und die Benutzergruppe angeben

```
- name: SettingFileOwnerProperty
  action: SetFileOwner
  inputs:
    - path: /home/UserName/SampleText.txt
      owner: LinuxUser
      group: LinuxUserGroup
```

Output

Keine.

SetFolderOwner

Das SetFolderOwnerAktionsmodul ändert rekursiv die Eigenschaften `owner` und `group` Eigentümer eines vorhandenen Ordners. Standardmäßig kann das Modul die Eigentümerschaft für den gesamten Inhalt eines Ordners ändern. Sie können die `recursive` Option auf setzen, `false` um dieses Verhalten zu überschreiben. Dieses Modul wird auf Windows-Plattformen nicht unterstützt.

Dieses Modul akzeptiert Benutzer- und Gruppennamen als Eingaben. Wenn der Gruppenname nicht angegeben wird, weist das Modul den Gruppeneigentümer der Datei der Gruppe zu, zu der der Benutzer gehört.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, die angegebene Änderung durchzuführen.
- Der angegebene Benutzer- oder Gruppenname ist zur Laufzeit nicht vorhanden.
- Der Ordner ist zur Laufzeit nicht vorhanden.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`path`	Der Ordnerpfad.	String	Ja	–	N/A	Wird unter Windows

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
						nicht unterstützt.
`owner`	Der Benutzername	Zeichenfolge	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`group`	Der Name der Benutzergruppe.	String	Nein	Der Name der Gruppe, zu der der Benutzer gehört.	N/A	Wird unter Windows nicht unterstützt.
`recursive`	Setzt das Standardverhalten beim Ändern der Eigentümerschaft für den gesamten Inhalt eines Ordners außer Kraft, wenn diese Einstellung auf `false` gesetzt ist.	Boolesch	Nein	`true`	N/A	Wird unter Windows nicht unterstützt.

Eingabebeispiel: Legen Sie die Eigenschaft des Ordnerbesitzers fest, ohne den Namen der Benutzergruppe anzugeben

```
- name: SettingFolderPropertyWithOutGroup
  action: SetFolderOwner
  inputs:
    - path: /SampleFolder/
      owner: LinuxUser
```

Eingabebeispiel: Legen Sie die Eigenschaft des Ordnerbesitzers fest, ohne die Eigentümerschaft aller Inhalte in einem Ordner zu überschreiben

```
- name: SettingFolderPropertyWithOutRecursively
  action: SetFolderOwner
  inputs:
    - path: /SampleFolder/
      owner: LinuxUser
      recursive: false
```

Eingabebeispiel: Legen Sie die Eigenschaft „Dateibesitz“ fest, indem Sie den Namen der Benutzergruppe angeben

```
- name: SettingFolderPropertyWithGroup
  action: SetFolderOwner
  inputs:
    - path: /SampleFolder/
      owner: LinuxUser
      group: LinuxUserGroup
```

Output

Keine.

SetFilePermissions

Das SetFilePermissionsAktionsmodul ändert die `permissions` einer vorhandenen Datei. Dieses Modul wird auf Windows-Plattformen nicht unterstützt.

Die Eingabe für `permissions` muss ein Zeichenkettenwert sein.

Dieses Aktionsmodul kann eine Datei mit Berechtigungen erstellen, die durch den Standard-Umask-Wert des Betriebssystems definiert sind. Sie müssen den `umask` Wert festlegen, wenn Sie den Standardwert überschreiben möchten.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, die angegebene Änderung durchzuführen.
- Die Datei ist zur Laufzeit nicht vorhanden.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`path`	Der Dateipfad.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`permissions`	Die Dateiberechtigungen.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.

Eingabebeispiel: Dateiberechtigungen ändern

```
- name: ModifyingFilePermissions
  action: SetFilePermissions
  inputs:
    - path: /home/UserName/SampleFile.txt
      permissions: 766
```

Output

Keine.

SetFolderPermissions

Das SetFolderPermissionsAktionsmodul ändert rekursiv den `permissions` eines vorhandenen Ordners und all seiner Unterdateien und Unterordner. Standardmäßig kann dieses Modul die Berechtigungen für den gesamten Inhalt des angegebenen Ordners ändern. Sie können die `recursive` Option auf setzen, `false` um dieses Verhalten zu überschreiben. Dieses Modul wird auf Windows-Plattformen nicht unterstützt.

Die Eingabe für `permissions` muss ein Zeichenkettenwert sein.

Dieses Aktionsmodul kann Berechtigungen entsprechend dem Standard-Umask-Wert des Betriebssystems ändern. Sie müssen den `umask` Wert festlegen, wenn Sie den Standardwert überschreiben möchten.

Das Aktionsmodul gibt einen Fehler zurück, wenn Folgendes eintritt:

- Sie sind nicht berechtigt, die angegebene Änderung durchzuführen.
- Der Ordner ist zur Laufzeit nicht vorhanden.
- Das Aktionsmodul stößt bei der Ausführung des Vorgangs auf einen Fehler.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
`path`	Der Ordnerpfad.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`permissions`	Die Ordnerberechtigungen.	String	Ja	–	N/A	Wird unter Windows nicht unterstützt.
`recursive`	Setzt das Standardv	Boolesch	Nein	`true`	N/A	Wird unter Windows

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte	Wird auf allen Plattformen unterstützt
	erhalten beim Ändern von Berechtigungen für den gesamten Inhalt eines Ordners außer Kraft, wenn diese Einstellung auf `false` gesetzt ist.					nicht unterstützt.

Eingabebeispiel: Ordnerberechtigungen festlegen

```
- name: SettingFolderPermissions
  action: SetFolderPermissions
  inputs:
    - path: SampleFolder/
      permissions: 0777
```

Eingabebeispiel: Ordnerberechtigungen festlegen, ohne die Berechtigungen für den gesamten Inhalt eines Ordners zu ändern

```
- name: SettingFolderPermissionsNoRecursive
  action: SetFolderPermissions
  inputs:
    - path: /home/UserName/SampleFolder/
      permissions: 777
      recursive: false
```

Output

Keine.

Aktionen zur Softwareinstallation

In diesem Abschnitt werden Aktionsmodule beschrieben, die Aktionsbefehle und Anweisungen zur Softwareinstallation ausführen.

IAM-Anforderungen

Wenn es sich bei Ihrem Installations-Download-Pfad um einen S3-URI handelt, muss die IAM-Rolle, die Sie Ihrem Instanzprofil zuordnen, über die Berechtigung zum Ausführen des `S3Download` Aktionsmoduls verfügen. Um die erforderliche Berechtigung zu erteilen, fügen Sie die `S3:GetObject` IAM-Richtlinie der IAM-Rolle bei, die Ihrem Instance-Profil zugeordnet ist, und geben Sie den Pfad für Ihren Bucket an. Beispiel, *`arn:aws:s3:::BucketName/*`*).

Komplexe MSI-Eingaben

Wenn Ihre Eingabezeichenfolgen doppelte Anführungszeichen (") enthalten, müssen Sie eine der folgenden Methoden verwenden, um sicherzustellen, dass sie korrekt interpretiert werden:

- Sie können einfache Anführungszeichen (') außerhalb Ihrer Zeichenfolge verwenden, um sie zu enthalten, und doppelte Anführungszeichen („) innerhalb Ihrer Zeichenfolge verwenden, wie im folgenden Beispiel gezeigt.

  ```
  properties:
    COMPANYNAME: '"Acme ""Widgets"" and ""Gizmos."""'
  ```

  Wenn Sie in diesem Fall einen Apostroph innerhalb Ihrer Zeichenfolge verwenden müssen, müssen Sie ihn maskieren. Das bedeutet, vor dem Apostroph ein weiteres einfaches Anführungszeichen (') zu verwenden.
- Sie können doppelte Anführungszeichen („) an der Außenseite Ihrer Zeichenfolge verwenden, um sie zu enthalten. Und Sie können alle doppelten Anführungszeichen innerhalb Ihrer Zeichenfolge maskieren, indem Sie den umgekehrten Schrägstrich (\) verwenden, wie im folgenden Beispiel gezeigt.

  ```
  properties:
    COMPANYNAME: "\"Acme \"\"Widgets\"\" and \"\"Gizmos.\"\"\""
  ```

Beide Methoden übergeben den Wert `COMPANYNAME="Acme ""Widgets"" and ""Gizmos."""` an den msiexec Befehl.

Aktionsmodule für die Softwareinstallation

- Installieren Sie MSI
- Deinstallieren Sie MSI

Installieren Sie MSI

Das `InstallMSI` Aktionsmodul installiert eine Windows-Anwendung mithilfe einer MSI-Datei. Sie können die MSI-Datei mithilfe eines lokalen Pfads, einer S3-Objekt-URI oder einer Web-URL angeben. Die Neustartoption konfiguriert das Neustartverhalten des Systems.

AWSTOE generiert den msiexec Befehl auf der Grundlage der Eingabeparameter für das Aktionsmodul. Werte für die Eingabeparameter `path` (Speicherort der MSI-Datei) und `logFile` (Speicherort der Protokolldatei) müssen in Anführungszeichen („) eingeschlossen werden.

Die folgenden MSI-Exitcodes gelten als erfolgreich:

- 0 (Erfolg)
- 1614 (ERROR_PRODUCT_UNINSTALL)
- 1641 (Neustart eingeleitet)
- 3010 (Neustart erforderlich)

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`path`	Geben Sie den Speicherort der MSI-Datei mit einer der folgenden Methoden an:	String	Ja	–	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	• Der lokale Dateipfad. Der Pfad kann absolut oder relativ sein • Ein gültiger S3-Objekt-URI. • Eine gültige HTTP-/HTTPS-Web-URL (HTTPS wird empfohlen), die dem RFC 3986-Standard entspricht. Verkettungsausdrücke sind zulässig.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`reboot`	Konfigurieren Sie das Verhalten beim Systemneustart, das auf eine erfolgreiche Ausführung des Aktionsmoduls folgt. Einstellungen: • `Force`— Leitet einen Systemneustart ein, nachdem der msiexec Befehl erfolgreich ausgeführt wurde. • `Allow`— Leitet einen Systemneustart ein, wenn der	String	Nein	`Allow`	`Allow`, `Force`, `Skip`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	msiexec Befehl einen Exit-Code zurückgibt, der angibt, dass ein Neustart erforderlich ist. • `Skip`— Protokolliert eine Informationsmeldung in der `console.log` Datei, die darauf hinweist, dass ein Neustart übersprungen wurde. Diese Option verhindert einen Neustart, auch wenn der msiexec Befehl				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	einen Exit-Code zurückgibt, der darauf hinweist, dass ein Neustart erforderlich ist.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`logOptions`	Geben Sie die Optionen an, die für die MSI-Installationsprotokollierung verwendet werden sollen. Die angegebenen Flags werden zusammen mit den /L Befehlszeilenparametern zur Aktivierung der Protokollierung an das MSI-Installationsprogramm übergeben. Wenn keine Flags angegeben sind, AWSTOE wird der Standardwert verwendet.	String	Nein	`*VX`	`i,w,e,a,r,u,c,m,o,p,v,x,+,!,*`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	Weitere Informationen zu Protokoll optionen für MSI finden Sie unter Befehlsze ilenoptio nen in der Microsoft Windows Installer-Produktd okumentat ion.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`logFile`	Ein absoluter oder relativer Pfad zum Speicherort der Protokoll datei. Wenn der Protokoll dateipfad nicht existiert , wird er erstellt. Wenn der Protokoll dateipfad nicht angegeben wird, AWSTOE wird das MSI-Installationsp rotokoll nicht gespeichert.	String	Nein	N/A	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`properties`	Schlüssel-Wert-Paare für MSI-Logging-Eigenschaften, zum Beispiel: `TARGETDIR: "C:\target\location"` Hinweis: Die Änderung der folgenden Eigenschaften ist nicht zulässig: • `REBOOT="ReallySupress"` • `REINSTALLMODE="ecmus"` • `REINSTALL="ALL"`	Map [Zeichenfolge] Zeichenfolge	Nein	N/A	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`ignoreAuthenticodeSignatureErrors`	Markierung zum Ignorieren von Fehlern bei der Überprüfung der Authenticode-Signatur für das im Pfad angegebene Installationsprogramm. Der Get-AuthenticodeSignature Befehl wird verwendet, um Installationsprogramme zu validieren. Einstellungen: • `true`— Validierungsfehler werden ignoriert und das Installat	Boolesch	Nein	`false`	`true`, `false`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	ionsprogramm wird ausgeführt. • `false`— Validierungsfehler werden nicht ignoriert. Das Installationsprogramm wird nur ausgeführt, wenn die Validierung erfolgreich ist. Dies ist das Standardverhalten.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`allowUnsignedInstaller`	Markierung, die die Ausführung des im Pfad angegebenen unsignierten Installationsprogramms ermöglicht. Der Get-AuthenticodeSignature Befehl wird verwendet, um Installationsprogramme zu validieren. Einstellungen: • `true` — Ignoriert den vom Get-AuthenticodeSignature Befehl zurückgegebenen	Boolesch	Nein	`false`	`true`, `false`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	`NotSigned` Status und führt das Installationsprogramm aus. • `false`— Erfordert, dass das Installationsprogramm signiert ist. Unsignierte Installationsprogramme können nicht ausgeführt werden. Dies ist das Standardverhalten.				

Beispiele

Die folgenden Beispiele zeigen Varianten des Eingabeabschnitts für Ihr Komponentendokument, abhängig von Ihrem Installationspfad.

Eingabebeispiel: Installation im lokalen Dokumentpfad

```
- name: local-path-install
  steps:
    - name: LocalPathInstaller
      action: InstallMSI
      inputs:
        path: C:\sample.msi
        logFile: C:\msilogs\local-path-install.log
        logOptions: '*VX'
        reboot: Allow
        properties:
          COMPANYNAME: '"Amazon Web Services"'
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: true
```

Eingabebeispiel: Amazon S3 S3-Pfadinstallation

```
- name: s3-path-install
  steps:
    - name: S3PathInstaller
      action: InstallMSI
      inputs:
        path: s3://<bucket-name>/sample.msi
        logFile: s3-path-install.log
        reboot: Force
        ignoreAuthenticodeSignatureErrors: false
        allowUnsignedInstaller: true
```

Eingabebeispiel: Installation eines Webpfads

```
- name: web-path-install
  steps:
    - name: WebPathInstaller
      action: InstallMSI
      inputs:
        path: https://<some-path>/sample.msi
        logFile: web-path-install.log
        reboot: Skip
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: false
```

Output

Das Folgende ist ein Beispiel für die Ausgabe des `InstallMSI` Aktionsmoduls.

```
{
 "logFile": "web-path-install.log",
 "msiExitCode": 0,
 "stdout": ""
}
```

Deinstallieren Sie MSI

Das `UninstallMSI` Aktionsmodul ermöglicht es Ihnen, eine Windows-Anwendung mithilfe einer MSI-Datei zu entfernen. Sie können den Speicherort der MSI-Datei mithilfe eines lokalen Dateipfads, einer S3-Objekt-URI oder einer Web-URL angeben. Die Neustartoption konfiguriert das Neustartverhalten des Systems.

AWSTOE generiert den msiexec Befehl auf der Grundlage der Eingabeparameter für das Aktionsmodul. Der Speicherort der MSI-Datei (`path`) und der Speicherort der Protokolldatei (`logFile`) werden bei der Generierung des msiexec Befehls explizit in doppelte Anführungszeichen („) eingeschlossen.

Die folgenden MSI-Exitcodes werden als erfolgreich angesehen:

- 0 (Erfolg)
- 1605 (ERROR_UNKNOWN_PRODUCT)
- 1614 (ERROR_PRODUCT_DEINSTALLIERT)
- 1641 (Neustart eingeleitet)
- 3010 (Neustart erforderlich)

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`path`	Geben Sie den Speicherort der MSI-Datei mit einer der	String	Ja	–	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	folgenden Methoden an: • Der lokale Dateipfad. Der Pfad kann absolut oder relativ sein. • Ein gültiger S3-Objekt-URI. • Eine gültige HTTP-/HTTPS-Web-URL (HTTPS wird empfohlen), die dem RFC 3986-Standard entspricht. Verkettungsausdrücke sind zulässig.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`reboot`	Konfiguriert das Verhalten beim Systemneustart, das auf eine erfolgreiche Ausführung des Aktionsmoduls folgt. Einstellungen: • `Force`— Leitet einen Systemneustart ein, nachdem der msiexec Befehl erfolgreich ausgeführt wurde. • `Allow`— Leitet einen Systemneustart ein, wenn der msiexec	String	Nein	`Allow`	`Allow`, `Force`, `Skip`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	Befehl einen Exit-Code zurückgibt, der angibt, dass ein Neustart erforderlich ist. • `Skip`— Protokolliert eine Informationsmeldung in der `console.log` Datei, die darauf hinweist, dass ein Neustart übersprungen wurde. Diese Option verhindert einen Neustart, auch wenn der msiexec Befehl einen				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	Exit-Code zurückgibt, der darauf hinweist, dass ein Neustart erforderlich ist.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`logOptions`	Geben Sie die Optionen an, die für die MSI-Installationsprotokollierung verwendet werden sollen. Die angegebenen Flags werden zusammen mit den /L Befehlszeilenparametern zur Aktivierung der Protokollierung an das MSI-Installationsprogramm übergeben. Wenn keine Flags angegeben sind, AWSTOE wird der Standardwert verwendet.	String	Nein	`*VX`	`i,w,e,a,r,u,c,m,o,p,v,x,+,!,*`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	Weitere Informationen zu Protokoll optionen für MSI finden Sie unter Befehlsze ilenoptio nen in der Microsoft Windows Installer-Produktd okumentat ion.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`logFile`	Ein absoluter oder relativer Pfad zum Speicherort der Protokoll datei. Wenn der Protokoll dateipfad nicht existiert , wird er erstellt. Wenn der Protokoll dateipfad nicht angegeben wird, AWSTOE wird das MSI-Installationsp rotokoll nicht gespeichert.	String	Nein	N/A	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`properties`	Schlüssel-Wert-Paare für MSI-Logging-Eigenschaften, zum Beispiel: `TARGETDIR: "C:\target\location"` Hinweis: Die Änderung der folgenden Eigenschaften ist nicht zulässig: • `REBOOT="ReallySupress"` • `REINSTALLMODE="ecmus"` • `REINSTALL="ALL"`	Map [Zeichenfolge] Zeichenfolge	Nein	N/A	N/A

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`ignoreAuthenticodeSignatureErrors`	Markierung zum Ignorieren von Fehlern bei der Überprüfung der Authenticode-Signatur für das im Pfad angegebene Installationsprogramm. Der Get-AuthenticodeSignature Befehl wird verwendet, um Installationsprogramme zu validieren. Einstellungen: • `true`— Validierungsfehler werden ignoriert und das Installat	Boolesch	Nein	`false`	`true`, `false`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	ionsprogramm wird ausgeführt. • `false`— Validierungsfehler werden nicht ignoriert. Das Installationsprogramm wird nur ausgeführt, wenn die Validierung erfolgreich ist. Dies ist das Standardverhalten.				

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
`allowUnsignedInstaller`	Markierung, die die Ausführung des im Pfad angegebenen unsignierten Installationsprogramms ermöglicht. Der Get-AuthenticodeSignature Befehl wird verwendet, um Installationsprogramme zu validieren. Einstellungen: • `true`— Ignoriert den vom Get-AuthenticodeSignature Befehl zurückgegebenen	Boolesch	Nein	`false`	`true`, `false`

Primitiv	Beschreibung	Typ	Erforderlich	Standardwert	Zulässige Werte
	`NotSigned` Status und führt das Installationsprogramm aus. • `false`— Erfordert, dass das Installationsprogramm signiert ist. Unsignierte Installationsprogramme können nicht ausgeführt werden. Dies ist das Standardverhalten.				

Beispiele

Die folgenden Beispiele zeigen Varianten des Eingabeabschnitts für Ihr Komponentendokument, abhängig von Ihrem Installationspfad.

Eingabebeispiel: Installation des lokalen Dokumentpfads entfernen

```
- name: local-path-uninstall
  steps:
    - name: LocalPathUninstaller
      action: UninstallMSI
      inputs:
        path: C:\sample.msi
        logFile: C:\msilogs\local-path-uninstall.log
        logOptions: '*VX'
        reboot: Allow
        properties:
          COMPANYNAME: '"Amazon Web Services"'
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: true
```

Eingabebeispiel: Amazon S3 S3-Pfadinstallation entfernen

```
- name: s3-path-uninstall
  steps:
    - name: S3PathUninstaller
      action: UninstallMSI
      inputs:
        path: s3://<bucket-name>/sample.msi
        logFile: s3-path-uninstall.log
        reboot: Force
        ignoreAuthenticodeSignatureErrors: false
        allowUnsignedInstaller: true
```

Eingabebeispiel: Webpfad-Installation entfernen

```
- name: web-path-uninstall
  steps:
    - name: WebPathUninstaller
      action: UninstallMSI
      inputs:
        path: https://<some-path>/sample.msi
        logFile: web-path-uninstall.log
        reboot: Skip
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: false
```

Output

Das Folgende ist ein Beispiel für die Ausgabe des `UninstallMSI` Aktionsmoduls.

```
{
 "logFile": "web-path-uninstall.log",
 "msiExitCode": 0,
 "stdout": ""
}
```

Aktionsmodule des Systems

Im folgenden Abschnitt werden Aktionsmodule beschrieben, die Befehle und Anweisungen für Dateisystemaktionen ausführen.

Module für Systemaktionen

- Neustart
- SetRegistry
- OS aktualisieren

Neustart

Das Aktionsmodul Reboot startet die Instanz neu. Es verfügt über eine konfigurierbare Option, um den Start des Neustarts zu verzögern. Standardmäßig `delaySeconds` ist auf eingestellt, was bedeutet`0`, dass es keine Verzögerung gibt. Das Schrittzeitlimit wird für das Aktionsmodul Neustart nicht unterstützt, da es nicht gilt, wenn die Instanz neu gestartet wird.

Wenn die Anwendung vom Systems Manager Agent aufgerufen wird, übergibt sie den Exit-Code (`3010`für Windows, `194` für Linux) an den Systems Manager Agent. Der Systems Manager Agent führt den Systemneustart wie unter Managed Instance from Scripts neu starten beschrieben durch.

Wenn die Anwendung auf dem Host als eigenständiger Prozess aufgerufen wird, speichert sie den aktuellen Ausführungsstatus, konfiguriert einen Auslöser für die automatische Ausführung nach dem Neustart, sodass die Anwendung nach dem Neustart erneut ausgeführt wird, und startet dann das System neu.

Automatischer Start-Trigger nach dem Neustart:

- Windows. AWSTOE erstellt einen Windows Taskplaner-Eintrag mit einem Trigger, der automatisch ausgeführt wird unter `SystemStartup`

- Linux. AWSTOE fügt einen Job in Crontab hinzu, der nach dem Neustart des Systems automatisch ausgeführt wird.

```
@reboot /download/path/awstoe run --document s3://bucket/key/doc.yaml
```

Dieser Trigger wird bereinigt, wenn die Anwendung gestartet wird.

> **Wiederholversuche**
>
> Standardmäßig ist die maximale Anzahl von Wiederholungen auf den Systems Manager `CommandRetryLimit` festgelegt. Wenn die Anzahl der Neustarts das Wiederholungslimit überschreitet, schlägt die Automatisierung fehl. Sie können das Limit ändern, indem Sie die Konfigurationsdatei (`Mds.CommandRetryLimit`) des Systems Manager Manager-Agenten bearbeiten. Weitere Informationen finden Sie unter Laufzeitkonfiguration im Open-Source-System Manager-Agent.

Um das Reboot-Aktionsmodul für Schritte zu verwenden, die einen Neustart beinhalten `exitcode` (z. B.`3010`), müssen Sie die Anwendungsbinärdatei als ausführen`sudo user`.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich	Standard
`delaySeconds`	Verzögert eine bestimmte Zeit, bevor ein Neustart eingeleitet wird.	Ganzzahl	Nein	`0`

Eingabebeispiel: Neustart-Schritt

```
- name: RebootStep
  action: Reboot
  onFailure: Abort
  maxAttempts: 2
  inputs:
```

```
      delaySeconds: 60
```

Ausgabe

Keine.

Wenn das Reboot-Modul abgeschlossen ist, fährt Image Builder mit dem nächsten Schritt im Build fort.

SetRegistry

Das SetRegistryAktionsmodul akzeptiert eine Liste von Eingaben und ermöglicht es Ihnen, den Wert für den angegebenen Registrierungsschlüssel festzulegen. Wenn ein Registrierungsschlüssel nicht existiert, wird er im definierten Pfad erstellt. Diese Funktion gilt nur für Windows.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich
`path`	Pfad des Registrierungsschlüssels.	String	Ja
`name`	Name des Registrierungsschlüssels.	String	Ja
`value`	Wert des Registrierungsschlüssels.	Zeichenfolge/Zahl/Array	Ja
`type`	Werttyp des Registrierungsschlüssels.	String	Ja

Unterstützte Pfadpräfixe

- `HKEY_CLASSES_ROOT / HKCR:`
- `HKEY_USERS / HKU:`
- `HKEY_LOCAL_MACHINE / HKLM:`
- `HKEY_CURRENT_CONFIG / HKCC:`
- `HKEY_CURRENT_USER / HKCU:`

Unterstützte -Typen

- BINARY
- DWORD
- QWORD
- SZ
- EXPAND_SZ
- MULTI_SZ

Eingabebeispiel: Legen Sie die Werte der Registrierungsschlüssel fest

```
- name: SetRegistryKeyValues
  action: SetRegistry
  maxAttempts: 3
  inputs:
    - path: HKLM:\SOFTWARE\MySoftWare
      name: MyName
      value: FirstVersionSoftware
      type: SZ
    - path: HKEY_CURRENT_USER\Software\Test
      name: Version
      value: 1.1
      type: DWORD
```

Ausgabe

Keine.

OS aktualisieren

Das UpdateOS-Aktionsmodul bietet Unterstützung für die Installation von Windows- und Linux-Updates. Es installiert standardmäßig alle verfügbaren Updates. Alternativ können Sie eine Liste mit einem oder mehreren spezifischen Updates konfigurieren, die das Aktionsmodul installieren soll. Sie können auch Updates angeben, die von der Installation ausgeschlossen werden sollen.

Wenn sowohl die Listen „Einschließen“ als auch „Ausschließen“ bereitgestellt werden, kann die resultierende Liste von Updates nur die Updates enthalten, die in der „Einschluss“ -Liste aufgeführt sind und nicht in der Ausschlussliste aufgeführt sind.

> **Note**
>
> UpdateOS unterstützt Amazon Linux 2023 (AL2023) nicht. Wir empfehlen Ihnen, Ihr Basis-AMI auf die neue Version zu aktualisieren, die mit jeder Version geliefert wird. Weitere Alternativen finden Sie unter Steuern der Updates, die Sie von Haupt- und Nebenversionen erhalten haben, im Amazon Linux 2023-Benutzerhandbuch.

- Windows. Updates werden von der auf dem Zielcomputer konfigurierten Update-Quelle installiert.
- Linux. Die Anwendung sucht nach dem unterstützten Paketmanager auf der Linux-Plattform und verwendet entweder den `apt-get` Paketmanager `yum` oder. Wenn keiner von beiden unterstützt wird, wird ein Fehler zurückgegeben. Sie sollten über die `sudo` erforderlichen Berechtigungen verfügen, um das UpdateOS-Aktionsmodul auszuführen. Wenn Sie keine `sudo` Berechtigungen haben, `error.Input` wird ein zurückgegeben.

Eingabe

Primitiv	Beschreibung	Typ	Erforderlich
`include`	Für Windows können Sie Folgendes angeben: • Eine oder mehrere Artikel-IDs der Microsoft Knowledge Base (KB), die in die Liste der Updates aufgenommen werden sollen, die installiert werden können. Gültige Formate sind `KB1234567` oder `1234567`.	Liste der Zeichenketten	Nein

Primitiv	Beschreibung	Typ	Erforderlich
	• Ein Updatename, der einen Platzhalterwert (*) verwendet. Gültige Formate sind `Security*` oder `*Security*` . Für Linux können Sie ein oder mehrere Pakete angeben, die in die Liste der zu installierenden Updates aufgenommen werden sollen.		

Primitiv	Beschreibung	Typ	Erforderlich
`exclude`	Für Windows können Sie Folgendes angeben: • Eine oder mehrere Artikel-IDs der Microsoft Knowledge Base (KB), die in die Liste der Updates aufgenommen werden sollen, die von der Installation ausgeschlossen werden sollen. Gültige Formate sind `KB1234567` oder `1234567`. • Ein Updatename, der einen Platzhalterwert (*) verwendet. Gültige Formate sind: `Security*` oder`*Security*` . Für Linux können Sie ein oder mehrere Pakete angeben, die von der Liste der Updates für die	Liste der Zeichenketten	Nein

Primitiv	Beschreibung	Typ	Erforderlich
	Installation ausgeschl ossen werden sollen.		

Eingabebeispiel: Unterstützung für die Installation von Linux-Updates hinzufügen

```
- name: UpdateMyLinux
  action: UpdateOS
  onFailure: Abort
  maxAttempts: 3
  inputs:
    exclude:
      - ec2-hibinit-agent
```

Eingabebeispiel: Unterstützung für die Installation von Windows-Updates hinzufügen

```
- name: UpdateWindowsOperatingSystem
  action: UpdateOS
  onFailure: Abort
  maxAttempts: 3
  inputs:
    include:
      - KB1234567
      - '*Security*'
```

Ausgabe

Keine.

Eingabe für den Befehl AWSTOE run konfigurieren

Um die Befehlszeileneingabe für Ihren AWSTOE run Befehl zu optimieren, können Sie Einstellungen für Befehlsparameter und Optionen in eine Eingabekonfigurationsdatei im JSON-Format mit einer `.json` Dateierweiterung aufnehmen. AWSTOE kann Ihre Datei von einem der folgenden Orte aus lesen:

- Ein lokaler Dateipfad (*`. /config.json`*).
- <bucket-name>Ein S3-Bucket (*`s3:////config.json`*<bucket-path>).

Wenn Sie den run Befehl eingeben, können Sie die Eingabekonfigurationsdatei mithilfe des Parameters angeben. --config Beispielsweise:

```
awstoe run --config <file-path>/config.json
```

Eingabe-Konfigurationsdatei

Die JSON-Datei für die Eingabekonfiguration enthält Schlüssel-Wert-Paare für alle Einstellungen, die Sie direkt über run Befehlsparameter und Optionen angeben können. Wenn Sie eine Einstellung sowohl in der Eingabekonfigurationsdatei als auch im run Befehl als Parameter oder Option angeben, gelten die folgenden Prioritätsregeln:

Regeln der Rangfolge

1. Eine Einstellung, die über einen Parameter oder eine Option direkt an den run Befehl in der AWS CLIübergeben wird, überschreibt jeden Wert, der in der Eingabekonfigurationsdatei für dieselbe Einstellung definiert ist.
2. Eine Einstellung in der Eingabekonfigurationsdatei überschreibt den Standardwert einer Komponente.
3. Wenn keine anderen Einstellungen an das Komponentendokument übergeben werden, kann es einen Standardwert anwenden, sofern einer existiert.

Es gibt zwei Ausnahmen von dieser Regel: Dokumente und Parameter. Diese Einstellungen funktionieren in der Eingabekonfiguration und als Befehlsparameter unterschiedlich. Wenn Sie die Eingabekonfigurationsdatei verwenden, dürfen Sie diese Parameter nicht direkt für den run Befehl angeben. Andernfalls wird ein Fehler generiert.

Einstellungen der Komponenten

Die Eingabekonfigurationsdatei enthält die folgenden Einstellungen. Um Ihre Datei zu optimieren, können Sie alle optionalen Einstellungen weglassen, die nicht benötigt werden. Alle Einstellungen sind optional, sofern nicht anders angegeben.

- cwIgnoreFailures(Boolean) — Ignoriert Protokollfehler in den CloudWatch Protokollen.
- cwLogGroup(String) — Der `LogGroup` Name für die CloudWatch Protokolle.
- cwLogRegion(Zeichenfolge) — Die AWS Region, die für die CloudWatch Protokolle gilt.
- cwLogStream(String) — Der `LogStream` Name für die CloudWatch Logs, der angibt, AWSTOE wohin die `console.log` Datei gestreamt werden soll.

- documentS3 BucketOwner (String) — Die Konto-ID des Bucket-Besitzers für S3-URI-basierte Dokumente.
- documents (Array von Objekten, erforderlich) — Ein Array von JSON-Objekten, die die Dokumente der YAML-Komponente darstellen, die den AWSTOE run Befehl ausführt. Es muss mindestens ein Komponentendokument angegeben werden.

  Jedes Objekt besteht aus den folgenden Feldern:

  - path (String, erforderlich) — Der Dateispeicherort des YAML-Komponentendokuments. Dabei muss es sich um einen der folgenden Werte handeln:
    - Ein lokaler Dateipfad (*. /component-doc-example.yaml).*
    - Eine S3-URI ()`s3://`*bucket/key*.
    - *Ein ARN für die Build-Version einer Image Builder-Komponente (arn:aws:imagebuilder:us-west-2:123456789012:component/ /2021.12.02/1). my-example-component*
  - parameters (Array von Objekten) — Ein Array von Objekten mit Schlüssel-Wert-Paaren, von denen jedes einen komponentenspezifischen Parameter darstellt, den der Befehl bei der Ausführung des Komponentendokuments übergibt. run Parameter sind für Komponenten optional. Für das Komponentendokument können Parameter definiert sein oder nicht.

    Jedes Objekt besteht aus den folgenden Feldern:

    - name (Zeichenfolge, erforderlich) — Der Name des Komponentenparameters.
    - value (String, erforderlich) — Der Wert, der für den benannten Parameter an das Komponentendokument übergeben werden soll.

    Weitere Informationen zu Komponentenparametern finden Sie im Abschnitt Parameter Definieren und referenzieren Sie Variablen in AWSTOE auf der Seite.
- executonId (String) — Dies ist die eindeutige ID, die für die Ausführung des aktuellen Befehls gilt. run Diese ID ist in den Namen der Ausgabe- und Protokolldateien enthalten, um diese Dateien eindeutig zu identifizieren und sie mit der aktuellen Befehlsausführung zu verknüpfen. Wenn diese Einstellung weggelassen wird, wird eine GUID AWSTOE generiert.
- LogDirectory (Zeichenfolge) — Das Zielverzeichnis, in dem alle Protokolldateien dieser Befehlsausführung AWSTOE gespeichert werden. Standardmäßig befindet sich dieses Verzeichnis im folgenden übergeordneten Verzeichnis:`TOE_<DATETIME>_<EXECUTIONID>`. Wenn Sie das Protokollverzeichnis nicht angeben, AWSTOE wird das aktuelle Arbeitsverzeichnis (`.`) verwendet.

- logS3 BucketName (Zeichenfolge) — Wenn Komponentenprotokolle in Amazon S3 gespeichert sind (empfohlen), werden die Komponentenanwendungsprotokolle in den in diesem Parameter genannten S3-Bucket AWSTOE hochgeladen.
- logS3 BucketOwner (Zeichenfolge) — Wenn Komponentenprotokolle in Amazon S3 gespeichert werden (empfohlen), ist dies die Eigentümerkonto-ID für den Bucket, in den die Protokolldateien AWSTOE geschrieben werden.
- logS3 KeyPrefix (String) — Wenn Komponentenprotokolle in Amazon S3 gespeichert werden (empfohlen), ist dies das S3-Objektschlüsselpräfix für den Protokollspeicherort im Bucket.
- parameters (Array von Objekten) — Ein Array von Objekten mit Schlüssel-Wert-Paaren, die Parameter darstellen, die global für alle Komponenten gelten, die in der aktuellen run Befehlsausführung enthalten sind.
  - name (Zeichenfolge, erforderlich) — Der Name des globalen Parameters.
  - value (String, erforderlich) — Der Wert, der an alle Komponentendokumente für den benannten Parameter übergeben werden soll.
- phases (String) — Eine durch Kommas getrennte Liste, die angibt, welche Phasen in den Dokumenten der YAML-Komponente ausgeführt werden sollen. Wenn ein Komponentendokument zusätzliche Phasen enthält, werden diese nicht ausgeführt.
- StateDirectory (Zeichenfolge) — Der Dateipfad, in dem State-Tracking-Dateien gespeichert werden.
- trace (Boolean) — Aktiviert die ausführliche Protokollierung in der Konsole.

Beispiele

Das folgende Beispiel zeigt eine Eingabekonfigurationsdatei, die die `test` Phasen `build` und für zwei Komponentendokumente ausführt:`sampledoc.yaml`, und. `conversation-intro.yaml` Jedes Komponentendokument hat einen Parameter, der nur für sich selbst gilt, und beide verwenden einen gemeinsamen Parameter. Der `project` Parameter gilt für beide Komponentendokumente.

```
{
   "documents": [
     {
       "path": "<file path>/awstoe/sampledoc.yaml>",
       "parameters": [
         {
           "name": "dayofweek",
           "value": "Monday"
         }
```

```
        ]
      },
      {
        "path": "<file path>/awstoe/conversation-intro.yaml>",
        "parameters": [
          {
            "name": "greeting",
            "value": "Hello, HAL."
          }
        ]
      }
    ],
    "phases": "build,test",
    "parameters": [
      {
        "name": "project",
        "value": "examples"
      }
    ],
    "cwLogGroup": "<log_group_name>",
    "cwLogStream": "<log_stream_name>",
    "documentS3BucketOwner": "<owner_aws_account_number>",
    "executionId": "<id_number>",
    "logDirectory": "<local_directory_path>",
    "logS3BucketName": "<bucket_name_for_log_files>",
    "logS3KeyPrefix": "<key_prefix_for_log_files>",
    "logS3BucketOwner": "<owner_aws_account_number>"
}
```

Verwaltete Komponenten von Distributor-Paketen für Windows

AWS Systems Manager Der Distributor unterstützt Sie beim Verpacken und Veröffentlichen von Software auf AWS Systems Manager verwalteten Knoten. Sie können Ihre eigene Software paketieren und veröffentlichen oder den Distributor verwenden, um von Ihnen AWS bereitgestellte Agenten-Softwarepakete zu finden und zu veröffentlichen. Weitere Informationen zu Systems Manager Distributor finden Sie unter AWS Systems Manager Distributor im AWS Systems Manager Benutzerhandbuch.

Verwaltete Komponenten für Distributor

Die folgenden von Image Builder verwalteten Komponenten verwenden AWS Systems Manager Distributor, um Anwendungspakete auf Windows-Instanzen zu installieren.

- Die `distributor-package-windows` verwaltete Komponente verwendet AWS Systems Manager Distributor, um Anwendungspakete zu installieren, die Sie auf Ihrer Windows-Image-Build-Instanz angeben. Informationen zur Konfiguration von Parametern, wenn Sie diese Komponente in Ihr Rezept aufnehmen, finden Sie unter`distributor-package-windows`Als eigenständige Komponente konfigurieren.
- Die `aws-vss-components-windows` Komponente verwendet AWS Systems Manager Distributor, um das `AwsVssComponents` Paket auf Ihrer Windows-Image-Build-Instanz zu installieren. Informationen zum Konfigurieren von Parametern, wenn Sie diese Komponente in Ihr Rezept aufnehmen, finden Sie unter`aws-vss-components-windows`Als eigenständige Komponente konfigurieren.

Weitere Informationen zur Verwendung verwalteter Komponenten in Ihrem Image Builder Builder-Rezept finden Sie unter Erstellen Sie eine neue Version eines Bildrezepts Für Image-Rezepte oder Eine neue Version eines Container-Rezepts erstellen für Container-Rezepte. Weitere Informationen über das `AwsVssComponents` Paket finden Sie unter Erstellen eines anwendungskonsistenten VSS-Snapshots im Amazon EC2 EC2-Benutzerhandbuch für Windows-Instances.

Voraussetzungen

Bevor Sie Image Builder Builder-Komponenten verwenden, die auf Systems Manager Distributor zur Installation von Anwendungspaketen angewiesen sind, müssen Sie sicherstellen, dass die folgenden Voraussetzungen erfüllt sind.

- Image Builder Builder-Komponenten, die Systems Manager Distributor verwenden, um Anwendungspakete auf Ihrer Instanz zu installieren, benötigen die Berechtigung, die Systems Manager Manager-API aufzurufen. Bevor Sie die Komponenten in einem Image Builder Builder-Rezept verwenden, müssen Sie die IAM-Richtlinie und die Rolle erstellen, die die Berechtigung gewähren. Informationen zum Konfigurieren von Berechtigungen finden Sie unterSystems Manager Manager-Verteilerberechtigungen konfigurieren.

> **Note**
>
> Image Builder unterstützt derzeit keine Systems Manager Distributor-Pakete, die die Instanz neu starten. Beispielsweise starten die Pakete `AWSNVMeAWSPVDrivers`, und `AwsEnaNetworkDriver` Distributor die Instanz neu und sind daher nicht zulässig.

Systems Manager Manager-Verteilerberechtigungen konfigurieren

Für die Ausführung der `distributor-package-windows` Komponente und anderer Komponenten, die sie verwendenaws-vss-components-windows, sind zusätzliche Berechtigungen für die Build-Instanz erforderlich. Die Build-Instanz muss in der Lage sein, die Systems Manager Manager-API aufzurufen, um eine Distributor-Installation zu starten und das Ergebnis abzufragen.

Gehen Sie wie folgt vor AWS Management Console , um eine benutzerdefinierte IAM-Richtlinie und -Rolle zu erstellen, die Image Builder Builder-Komponenten die Erlaubnis erteilen, Systems Manager Distributor-Pakete von der Build-Instanz aus zu installieren.

Schritt 1: Erstellen Sie eine Richtlinie

Erstellen Sie eine IAM-Richtlinie für Verteilerberechtigungen.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Policies (Richtlinien) und dann Create policy (Richtlinie erstellen).
3. Wählen Sie auf der Seite Richtlinie erstellen die Registerkarte JSON aus und ersetzen Sie dann den Standardinhalt durch die folgende JSON-Richtlinie. Ersetzen Sie dabei gegebenenfalls Partition, Region und Konto-ID oder verwenden Sie Platzhalter.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowDistributorSendCommand",
   "Effect": "Allow",
   "Action": [
    "ssm:SendCommand"
   ],
   "Resource": [
    "arn:${AWS::Partition}:ssm:${AWS::Region}::document/AWS-ConfigureAWSPackage",
    "arn:${AWS::Partition}:ec2:${AWS::Region}:${AWS::AccountId}:instance/*"
   ]
  },
  {
   "Sid": "AllowGetCommandInvocation",
   "Effect": "Allow",
   "Action": [
```

```
      "ssm:GetCommandInvocation"
    ],
    "Resource": [
      "*"
    ]
   }
  ]
}
```

4. Wählen Sie Review policy (Richtlinie prüfen) aus.
5. Geben Sie für Name einen Namen zum Identifizieren der Richtlinie ein, wie z. B. *InvokeDistributor* oder einen anderen von Ihnen bevorzugten Namen.
6. (Optional) Geben Sie für Description (Beschreibung) eine Beschreibung des Zwecks der Rolle ein.
7. Klicken Sie auf Create Policy.

Schritt 2: Erstellen Sie eine Rolle

Erstellen Sie eine IAM-Rolle für Distributor-Berechtigungen.

1. Wählen Sie im Navigationsbereich der IAM-Konsole Rollen und anschließend Rolle erstellen aus.
2. Wählen Sie unter Select type of trusted entity (Typ der vertrauenswürdigen Entität auswählen) die Option AWS-Service Service aus.
3. Wählen Sie für Choose the service that will use this role (Wählen Sie den Service aus, der diese Rolle verwendet) die Option EC2 und danach Next: Permissions (Nächster Schritt: Berechtigungen) aus.
4. Wählen Sie unter Select your use case (Anwendungsfall auswählen) die Option EC2 und anschließend Next: Permissions (Weiter: Berechtigungen) aus.
5. Aktivieren Sie in der Liste der Richtlinien das Kontrollkästchen neben ManagedInstanceCoreAmazonSSM. (Geben Sie `SSM` in das Suchfeld ein, wenn Sie die Liste eingrenzen müssen.)
6. Aktivieren Sie in dieser Richtlinienliste das Kästchen neben EC2. InstanceProfileForImageBuilder (Geben Sie `ImageBuilder` in das Suchfeld ein, wenn Sie die Liste eingrenzen müssen.)
7. Wählen Sie Next: Tags (Weiter: Tags (Markierungen)) aus.
8. (Optional) Fügen Sie ein oder mehrere Tag-Schlüssel-Wertepaare hinzu, um den Zugriff für diese Rolle zu organisieren, nachzuverfolgen oder zu kontrollieren, und wählen Sie dann Weiter: Überprüfen aus.

9. Geben Sie unter Role name (Rollenname) einen Namen für die Rolle, wie z. B. *InvokeDistributor* oder einen anderen von Ihnen bevorzugten Namen ein.
10. (Optional) Ersetzen Sie für Role description (Rollenbeschreibung) den Standardtext mit einer Beschreibung des Zwecks der Rolle.
11. Wählen Sie Create role (Rolle erstellen) aus. Das System leitet Sie zur Seite Roles (Rollen) zurück.

Schritt 3: Hängen Sie die Richtlinie an die Rolle an

Der letzte Schritt zur Einrichtung Ihrer Vertriebspartnerberechtigungen besteht darin, die IAM-Richtlinie an die IAM-Rolle anzuhängen.

1. Wählen Sie auf der Seite Rollen in der IAM-Konsole die Rolle aus, die Sie gerade erstellt haben. Die Seite mit der Rollenzusammenfassung wird geöffnet.
2. Wählen Sie Attach Policies (Richtlinien hinzufügen) aus.
3. Suchen Sie nach der Richtlinie, die Sie im vorherigen Verfahren erstellt haben, und aktivieren Sie das Kontrollkästchen neben dem Namen.
4. Wählen Sie Richtlinie anfügen aus.

Verwenden Sie diese Rolle in der Image Builder Builder-Infrastrukturkonfigurationsressource für jedes Image, das Komponenten enthält, die Systems Manager Distributor verwenden. Weitere Informationen finden Sie unter Erstellen einer Infrastrukturkonfiguration.

`distributor-package-windows`Als eigenständige Komponente konfigurieren

Um die `distributor-package-windows` Komponente in einem Rezept zu verwenden, legen Sie die folgenden Parameter fest, mit denen das zu installierende Paket konfiguriert wird.

> **Note**
>
> Bevor Sie die `distributor-package-windows` Komponente in einem Rezept verwenden, müssen Sie sicherstellen, dass alle Bedingungen erfüllt Voraussetzungen sind.

- Aktion (erforderlich) — Geben Sie an, ob das Paket installiert oder deinstalliert werden soll. Gültige Werte sind `Install` und `Uninstall`. Der Standardwert ist. `Install`
- PackageName(Erforderlich) — Der Name des zu installierenden oder zu deinstallierenden Distributor-Pakets. Eine Liste der gültigen Paketnamen finden Sie unterFinden Sie Vertriebspartner-Pakete.
- PackageVersion(Optional) — Die Version des zu installierenden Distributor-Pakets. PackageVersion ist standardmäßig auf die empfohlene Version eingestellt.
- AdditionalArguments(Optional) — Eine JSON-Zeichenfolge, die die zusätzlichen Parameter enthält, die Sie Ihrem Skript zur Installation, Deinstallation oder Aktualisierung eines Pakets zur Verfügung stellen müssen. Weitere Informationen finden Sie unter AdditionalArguments im Abschnitt aws:ConfigurePackage Inputs der Referenzseite des Systems Manager Command-Dokument-Plug-ins.

`aws-vss-components-windows`Als eigenständige Komponente konfigurieren

Wenn Sie die `aws-vss-components-windows` Komponente in einem Rezept verwenden, können Sie den `PackageVersion` Parameter optional so einstellen, dass eine bestimmte Version des `AwsVssComponents` Pakets verwendet wird. Wenn Sie diesen Parameter weglassen, verwendet die Komponente standardmäßig die empfohlene Version des `AwsVssComponents` Pakets.

> **Note**
>
> Bevor Sie die `aws-vss-components-windows` Komponente in einem Rezept verwenden, müssen Sie sicherstellen, dass alle Kriterien erfüllt Voraussetzungen sind.

Finden Sie Vertriebspartner-Pakete

Amazon und Drittanbieter stellen öffentliche Pakete zur Verfügung, die Sie mit Systems Manager Distributor installieren können.

Um die verfügbaren Pakete in der anzuzeigen AWS Management Console, melden Sie sich an der AWS Systems Manager Konsole an und wählen Sie im Navigationsbereich die Option Distributor aus. Auf der Händlerseite werden alle Pakete angezeigt, die für Sie verfügbar sind. Weitere Informationen

zum Auflisten verfügbarer Pakete mit dem AWS CLI finden Sie unter Pakete anzeigen (Befehlszeile) im AWS Systems Manager Benutzerhandbuch.

Sie können auch Ihre eigenen privaten Systems Manager Distributor-Pakete erstellen. Weitere Informationen finden Sie im AWS Systems Manager Benutzerhandbuch unter Paket erstellen.

CIS-Härtungskomponenten

Das Center for Internet Security (CIS) ist eine gemeinnützige Organisation mit gemeinnütziger Ausrichtung. Ihre Cybersicherheitsexperten arbeiten zusammen, um IT-Sicherheitsrichtlinien zu entwickeln, die öffentliche und private Organisationen vor Cyberbedrohungen schützen. Ihre weltweit anerkannten Best Practices, die als CIS Benchmarks bekannt sind, helfen IT-Organisationen auf der ganzen Welt, ihre Systeme sicher zu konfigurieren. Aktuelle Artikel, Blogbeiträge, Podcasts, Webinare und Whitepapers finden Sie auf der Website CIS Insights on the Center for Internet Security.

CIS Benchmarks

CIS erstellt und verwaltet eine Reihe von Konfigurationsrichtlinien, die als CIS-Benchmarks bezeichnet werden und bewährte Methoden für die Konfiguration bestimmter Technologien enthalten, darunter Betriebssysteme, Cloud-Plattformen, Anwendungen, Datenbanken und mehr. CIS-Benchmarks werden von Organisationen und Standards wie PCI DSS, HIPAA, DoD Cloud Computing SRG, FISMA, DFARS und FEDRAMP als Industriestandard anerkannt. Weitere Informationen finden Sie unter CIS Benchmarks auf der Website des Center for Internet Security.

CIS-Härtungskomponenten

Wenn Sie ein CIS Hardened Image abonnieren AWS Marketplace, erhalten Sie auch Zugriff auf die zugehörige Hardening-Komponente, die ein Skript ausführt, um die CIS Benchmarks Level 1-Richtlinien für Ihre Konfiguration durchzusetzen. Die CIS-Organisation besitzt und verwaltet die CIS-Härtungskomponenten, um sicherzustellen, dass sie den neuesten Richtlinien entsprechen.

> **Note**
>
> CIS-Härtungskomponenten folgen nicht den Standardregeln für die Reihenfolge der Komponenten in den Image Builder Builder-Rezepten. Die CIS-Härtungskomponenten werden immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests anhand Ihres Ausgabebilds ausgeführt werden.

Von Amazon verwaltete STIG-Härtungskomponenten für EC2 Image Builder

Die Security Technical Implementation Guides (STIGs) sind die von der Defense Information Systems Agency (DISA) entwickelten Standards zur Sicherung von Informationssystemen und Software zur Absicherung von Informationssystemen und Software. Um Ihre Systeme mit STIG-Standards konform zu machen, müssen Sie eine Vielzahl von Sicherheitseinstellungen installieren, konfigurieren und testen.

Image Builder bietet STIG-Härtungskomponenten, mit denen Sie effizienter konforme Images für STIG-Grundstandards erstellen können. Diese STIG-Komponenten suchen nach Fehlkonfigurationen und führen ein Korrekturskript aus. Für die Verwendung von STIG-kompatiblen Komponenten fallen keine zusätzlichen Gebühren an.

> **Important**
>
> Mit wenigen Ausnahmen installieren STIG-Hardening-Komponenten keine Pakete von Drittanbietern. Wenn Pakete von Drittanbietern bereits auf der Instanz installiert sind und wenn es verwandte STIGs gibt, die Image Builder für dieses Paket unterstützt, wendet die Hardening-Komponente sie an.

Auf dieser Seite sind alle STIGs aufgeführt, die Image Builder unterstützt und die auf die EC2-Instances angewendet werden, die Image Builder startet, wenn Sie ein neues Image erstellen und testen. Wenn Sie zusätzliche STIG-Einstellungen auf Ihr Image anwenden möchten, können Sie eine benutzerdefinierte Komponente erstellen, um es zu konfigurieren. Weitere Informationen zu benutzerdefinierten Komponenten und deren Erstellung finden Sie unterKomponenten mit Image Builder verwalten.

Wenn Sie ein Image erstellen, protokollieren die STIG-Härtungskomponenten, ob unterstützte STIGs angewendet oder übersprungen wurden. Wir empfehlen Ihnen, die Image Builder Builder-Protokolle für Ihre Images zu überprüfen, die STIG-Hardening-Komponenten verwenden. Weitere Informationen zum Zugreifen auf und Überprüfen von Image Builder Builder-Protokollen finden Sie unterProblembehandlung bei Pipeline-Buil.

Compliance-Stufen

- Hoch (Kategorie I)

Das schwerwiegendste Risiko. Schließt jede Schwachstelle ein, die zu einem Verlust der Vertraulichkeit, Verfügbarkeit oder Integrität führen kann.

- Mittel (Kategorie II)

  Umfasst alle Sicherheitslücken, die zum Verlust von Vertraulichkeit, Verfügbarkeit oder Integrität führen können. Die Risiken können jedoch gemindert werden.

- Niedrig (Kategorie III)

  Jede Schwachstelle, die Maßnahmen zum Schutz vor einem Verlust der Vertraulichkeit, Verfügbarkeit oder Integrität beeinträchtigt.

Themen

Komponenten zur Härtung von Windows STIG

AWSTOE Die Windows STIG-Härtungskomponenten sind für eigenständige Server konzipiert und wenden lokale Gruppenrichtlinien an. STIG-konforme Härtungskomponenten werden InstallRoot vom Verteidigungsministerium (DoD) auf der Windows-Infrastruktur installiert, um die DoD-Zertifikate herunterzuladen, zu installieren und zu aktualisieren. Sie entfernen auch unnötige Zertifikate, um die STIG-Konformität aufrechtzuerhalten. Derzeit werden STIG-Baselines für die folgenden Versionen von Windows Server unterstützt: 2012 R2, 2016, 2019 und 2022.

In diesem Abschnitt werden die aktuellen Einstellungen für jede der Windows STIG-Hardening-Komponenten aufgeführt, gefolgt von einem Versionsverlaufsprotokoll.

STIG-Build-Windows-Low Version 2022.4.x

Die folgende Liste enthält STIG-Einstellungen, die die Hardening-Komponente auf Ihre Infrastruktur anwendet. Wenn eine unterstützte Einstellung für Ihre Infrastruktur nicht zutrifft, überspringt die Härtungskomponente diese Einstellung und fährt fort. Beispielsweise gelten einige STIG-

Einstellungen möglicherweise nicht für eigenständige Server. Unternehmensspezifische Richtlinien können sich auch darauf auswirken, welche Einstellungen die Härtungskomponente anwendet, z. B. die Anforderung, dass Administratoren die Dokumenteinstellungen überprüfen müssen.

Eine vollständige Liste der Windows-STIGs finden Sie in der STIGs-Dokumentbibliothek. Informationen zum Anzeigen der vollständigen Liste finden Sie unter STIG Viewing Tools.

- Windows Server 2022 STIG Version 1 Version 1

  V-254335, V-254336, V-254337, V-254338, V-254351, V-254357, V-254363 und V-254481
- Windows Server 2019 STIG Version 2 Version 5

  V-205691, V-205819, V-205858, V-205859, V-205860, V-205870, V-205871 und V-205923
- Windows Server 2016 STIG Version 2 Version 5

  V-224916, V-224917, V-224918, V-224919, V-224931, V-224942 und V-225060
- Windows Server 2012 R2 MS STIG Version 3 Version 5

  V-225537, V-225536, V-225526, V-225525, V-225514, V-225511, V-225490, V-225489, V-225488, V-225487, V-225485, V-225484, V-225483, V-225482, V-225481, V-225480, V-225479, V-225476, V-225473, V-225468, V-225462, V-225460, V-225459, V-225412, V-225394, V-225392, V-225376, V-225363, V-225362, V-225360, V-225359, V-225358, V-225357, V-225355, V-225343, V-225342, V-225336, V-225335, V-225334, V-225333, V-225332, V-225331, V-225330, V-225328, V-225327, V-225324, V-225319, V-225318 und V-225250
- Microsoft.NET Framework 4.0 STIG Version 2 Version 2

  Auf das Microsoft .NET Framework für Sicherheitslücken der Kategorie III werden keine STIG-Einstellungen angewendet.
- Windows Firewall STIG Version 2 Version 1

  V-241994, V-241995, V-241996, V-241999, V-242000, V-242001, V-242006, V-242007 und V-242008
- Internet Explorer 11 STIG Version 2 Version 3

  V-46477, V-46629 und V-97527
- Microsoft Edge STIG Version 1 Version 6 (nur Windows Server 2022)

  V-235727, V-235731, V-235751, V-235752 und V-235765

STIG-Build-Windows-Medium versie 2022.4.x

Die folgende Liste enthält STIG-Einstellungen, die die Hardening-Komponente auf Ihre Infrastruktur anwendet. Wenn eine unterstützte Einstellung für Ihre Infrastruktur nicht zutrifft, überspringt die Härtungskomponente diese Einstellung und fährt fort. Beispielsweise gelten einige STIG-Einstellungen möglicherweise nicht für eigenständige Server. Unternehmensspezifische Richtlinien können sich auch darauf auswirken, welche Einstellungen die Härtungskomponente anwendet, z. B. die Anforderung, dass Administratoren die Dokumenteinstellungen überprüfen müssen.

Eine vollständige Liste der Windows-STIGs finden Sie in der STIGs-Dokumentbibliothek. Informationen zum Anzeigen der vollständigen Liste finden Sie unter STIG Viewing Tools.

> **Note**
>
> Die Komponenten STIG-Build-Windows-Medium Hardening beinhalten alle aufgelisteten STIG-Einstellungen, die für STIG-Build-Windows-Low Hardening-Komponenten AWSTOE gelten, zusätzlich zu den STIG-Einstellungen, die speziell für Sicherheitslücken der Kategorie II aufgeführt sind.

- Windows Server 2022 STIG Version 1 Version 1

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) anwendet, sowie:

  V-254247, V-254265, V-254269, V-254270, V-254271, V-254272, V-254273, V-254274, V-254276, V-254277, V-254278, V-254285, V-254286, V-254287, V-254288, V-254289, V-254290, V-254291, V-254292, V-254300, V-254301, V-254302, V-254303, V-254304, V-254305, V-254306, V-254307, V-254308, V-254309, V-254310, V-254311, V-254312, V-254313, V-254314, V-254315, V-254316, V-254317, V-254318, V-254319, V-254320, V-254321, V-254322, V-254323, V-254324, V-254325, V-254326, V-254327, V-254328, V-254329, V-254330, V-254331, V-254332, V-254333, V-254334, V-254339, V-254341, V-2543422, V-254344, V-254345, V-254346, V-254347, V-254348, V-254349, V-254350, V-254355, V-254356, V-254358, V-254359, V-254360, V-254361, V-254362, V-254364, V-254365, V-254366, V-254367, V-254368 V-254369, V-254370, V-254371, V-254372, V-254373, V-254375, V-254376, V-254377, V-254379, V-254380, V-254382, V-254383, V-254431, V-254432, V-254433, V-254434, V-254435, V-254436, V-254438, V-254439, V-254442, V-254443, V-254444, V-254445, V-254449, V-254450, V-254451, V-254452, V-254453, V-254454, V-254455, V-254456, V-254459, V-254460, V-254461, V-254462, V-254463, V-2544646 4, V-254468,

V-254470, V-254471, V-254472, V-254473, V-254476, V-254477, V-254478, V-254479, V-254480, V-254482, V-254483, V-254484, V-254485, V-254486, V-254487, V-254488, V-254489, V-254490, V-254493, V-254494, V-254495, V-254497, V-254499, V-254501, V-254502, V-254503, V-254504, V-254505, V-254507, V-254508, V-254509, V-254510, V-254511 und V-254512

- Windows Server 2019 STIG Version 2 Version 5

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) anwendet, sowie:

  V-205625, V-205626, V-205627, V-205629, V-205630, V-205633, V-205634, V-205635, V-205636, V-205637, V-205638, V-205639, V-205643, V-205644, V-205648, V-205649, V-205650, V-205651, V-205652, V-205655, V-205656, V-205659, V-205660, V-205662, V-205671, V-205672, V-205673, V-205675, V-205676, V-205678, V-205679, V-205680, V-205681, V-205682, V-205683, V-205684, V-205685, V-205686, V-205686, 205687, V-205688, V-205689, V-205690, V-205692, V-205693, V-205694, V-205697, V-205698, V-205708, V-205709, V-205712, V-205714, V-205716, V-205717, V-205718, V-205719, V-205720, V-205722, V-205729, V-205730, V-205733, V-205747, V-205751, V-205752, V-205754, V-205756, V-205758, V-205759, V-205760, V-205761, V-205762, V-205764, V-205765, V-205766, V-205767, V-205768, V-205769, V-205770, V-205771, V-205772, V-205773, V-205774, V-205775, V-205776, V-205777, V-205778, V-205779, V-205780, V-205781, V-205782, V-205783, V-205784, V-205795, V-205796, V-205797, V-205798, V-205798, V-205798 205801, V-205808, V-205809, V-205810, V-205811, V-205812, V-205813, V-205814, V-205815, V-205816, V-205817, V-205821, V-205822, V-205823, V-205824, V-205825, V-205826, V-205827, V-205828, V-205830, V-205832, V-205833, V-205834, V-205835, V-205836, V-205837, V-205838, V-205839, V-205840, V-205841, V-205861, V-205863, V-205865, V-205866, V-205867, V-205868, V-205869, V-205872, V-205873, V-205874, V-205911, V-205912, V-205915, V-205916, V-205917, V-205918, V-205920, V-205921, V-205922, V-205924, V-205925 und V-236001

- Windows Server 2016 STIG Version 2 Version 5

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) anwendet, sowie:

  V-224850, V-224852, V-224853, V-224854, V-224855, V-224856, V-224857, V-224858, V-224859, V-224866, V-224867, V-224868, V-224869, V-224870, V-224871, V-224872, V-224873, V-224881, V-224882, V-224882, V-224882, V-224882, V-224882 V-224883, V-224884, V-224885, V-224886, V-224887, V-224888, V-224889, V-224890, V-224891, V-224892, V-224893, V-224894, V-224895, V-224896, V-224897, V-224898, V-224899, V-224900, V-224901, V-224902, V-224903, V-224904, V-224905, V-224906, V-224907, V-224908, V-224909, V-224910, V-224911, V-224912, V-224913,

V-224914, V-224915, V-224920, V-224922, V-224924, V-224925, V-224926, V-224927, V-224928, V-224929, V-224930, V-224935, V-224936, V-224937, V-224938, V-224939, V-224940, V-224941, V-224943, V-224944, V-224945, V-224946, V-224947, V-224948, V-224949, V-224951, V-224951, V-224951, V-224951, V-224951, V-224951, V-224951 V-224952, V-224953, V-224955, V-224956, V-224957, V-224959, V-224960, V-224962, V-224963, V-225010, V-225013, V-225014, V-225015, V-225016, V-225017, V-225018, V-225019, V-225021, V-225022, V-225022 25023, V-225024, V-225028, V-225029, V-225030, V-225031, V-225032, V-225033, V-225034, V-225035, V-225038, V-225039, V-225040, V-225041, V-225042, V-225043, V-225047, V-225049, V-225050, V-225051, V-225052, V-225055, V-225056, V-225057, V-225058, V-225061, V-225062, V-225063, V-225064, V-225065, V-225066, V-225067, V-225068, V-225069, V-225072, V-225073, V-225074, V-225076, V-225078, V-225080, V-225081, V-225082, V-225083, V-225084, V-225086, V-225087, V-225088, V-225089, V-225092, V-225093 und V-236000

- Windows Server 2012 R2 MS STIG Version 3 Version 5

  Enthält alle unterstützten STIG-Einstellungen, die von der Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) verwendet werden, sowie:

  V-225574, V-225573, V-225572, V-225571, V-225570, V-225569, V-225568, V-225567, V-225566, V-225565, V-225564, V-225563, V-225562, V-225561, V-225560, V-225559, V-225558, V-225557, V-225555, V-225557 4, V-225553, V-225551, V-225550, V-225549, V-225548, V-225546, V-225545, V-225544, V-225543, V-225542, V-225541, V-225540, V-225539, V-225538, V-225535, V-225534, V-225533, V-225532, V-225531, V-225531, V-225531, V-225531, V-225531 V-225530, V-225529, V-225528, V-225527, V-225524, V-225523, V-225522, V-225521, V-225520, V-225519, V-225518, V-225517, V-225516, V-225515, V-225513, V-225510, V-225509, V-225508, V-225506, V-225504, V-225503, V-225502, V-225501, V-225500, V-225494, V-225486, V-225478, V-225477, V-225475, V-225474, V-225472, V-225471, V-225470, V-225469, V-225464, V-225463, V-225461, V-225458, V-225457, V-225456, V-225455, V-225454, V-225453, V-225452, V-225448, V-225443, V-225442, V-225441, V-225415, V-225414, V-225413, V-225411, V-225410, V-225409, V-225408, V-225407, V-225406, V-225405, V-225404, V-225402, V-225401, V-225400, V-225398, V-225397, V-225395, V-225393, V-225391, V-225389, V-225386, V-225385, V-225384, V-225383, V-225382, V-225381, V-225380, V-225379, V-225378, V-225377, V-225375, V-225374, V-225373, V-225372, V-225371, V-225370, V-225369, V-225368, V-225367, V-225356, V-225353, V-225352, V-225351, V-225350, V-225349, V-225348, V-225347, V-225346, V-225345, V-225344, V-225341, V-225340, V-225339, V-225338, V-225337, V-225329, V-225326, V-225325, V-225317, V-225316, V-225315, V-225314, V-225305, V-225304, V-225303, V-225302, V-225301, V-225300, V-225299, V-225298, V-225297, V-225296, V-225295, V-225294, V-225293, V-225292, V-225291, V-225290, V-225289, V-225288, V-225287, V-225286, V-225285, V-225284, V-225283, V-225282, V-225281, V-225280,

V-225279, V-225278, V-225277, V-225276, V-225275, V-225273, V-225272, V-225271, V-225270, V-225269, V-225268, V-225267, V-225266, V-225265, V-225264, V-225263, V-225263, V-225261, V-225260, V-225259 und V-225259 239

- Microsoft.NET Framework 4.0 STIG Version 2 Version 2

  Enthält alle unterstützten STIG-Einstellungen, die von der Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) verwendet werden, sowie V-225238

- Windows Firewall STIG Version 2 Version 1

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) anwendet, sowie:

  V-241989, V-241990, V-241991, V-241993, V-241998 und V-242003

- Internet Explorer 11 STIG Version 2 Version 3

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) anwendet, sowie:

  V-46473, V-46475, V-46481, V-46483, V-46501, V-46507, V-46509, V-46511, V-46513, V-46515, V-46517, V-46521, V-46523, V-46525, V-46543, V-46545, V-46547, V-46549, V-46553, V-46555, V-46573, V-46575, V-46577, V-46579, V-46581, V-46583, V-46587, V-46589, V-46591, V-46593, V-46597, V-46599, V-46601, V-46603, V-46605, V-46607, V-46609, V-46615, V-46617, V-46619, V-46621, V-46625, V-46633, V-46635, V-46637, V-46639, V-46641, V-46643, V-46645, V-46647, V-46649, V-46653, V-46663, V-46665, V-46669, V-46681, V-46685, V-46689, V-46691, V-46693, V-46695, V-46701, V-46705, V-46709, V-46711, V-46713, V-46715, V-46717, V-46719, V-46721, V-46723, V-46725, V-46727, V-46729, V-46731, V-46733, V-46779, V-46781, V-46787, V-46789, V-46791, V-46797, V-46799, V-46801, V-46807, V-46811, V-46815, V-46819, V-46829, V-46841, V-46847, V-46849, V-46853, V-46857, V-46859, V-46861, V-46865, V-46869, V-46879, V-46883, V-46885, V-46889, V-46893, V-46895, V-46897, V-46903, V-46907, V-46921, V-46927, V-46939, V-46975, V-46981, V-46987, V-46995, V-46997, V-46999, V-47003, V-47005, V-47009, V-64711, V-64713, V-64715, V-64717, V-64719, V-64721, V-64723, V-64725, V-64729, V-72757, V-72759, V-72761, V-72763, V-75169 und V-75171

- Microsoft Edge STIG Version 1 Version 6 (nur Windows Server 2022)

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) anwendet, sowie:

V-235720, V-235721, V-235723, V-235724, V-235725, V-235726, V-235728, V-235729, V-235730, V-235732, V-235733, V-235734, V-235735, V-235736, V-235737, V-235738, V-235739, V-235740, V-235741, V-235742, V-235743, V-235744, V-235745, V-235746, V-235747, V-235748, V-235749, V-235750, V-235754, V-235756, V-235760, V-235761, V-235763, V-235764, V-235766, V-235767, V-235768, V-235768, V-235768 69, V-235770, V-235771, V-235772, V-235773, V-235774 und V-246736

- Defender STIG Version 2 Version 4 (nur Windows Server 2022)

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) anwendet, sowie:

  V-213427, V-213429, V-213430, V-213431, V-213432, V-213433, V-213434, V-213435, V-213436, V-213437, V-213438, V-213439, V-213440, V-213441, V-213442, V-213443, V-213444, V-213445, V-213446, V-213447, V-213447, V-213447 48, V-213449, V-213450, V-213451, V-213455, V-213464, V-213465 und V-213466

STIG-Build-Windows-hohe Version 2022.4.x

Die folgende Liste enthält STIG-Einstellungen, die die Hardening-Komponente auf Ihre Infrastruktur anwendet. Wenn eine unterstützte Einstellung für Ihre Infrastruktur nicht zutrifft, überspringt die Härtungskomponente diese Einstellung und fährt fort. Beispielsweise gelten einige STIG-Einstellungen möglicherweise nicht für eigenständige Server. Unternehmensspezifische Richtlinien können sich auch darauf auswirken, welche Einstellungen die Härtungskomponente anwendet, z. B. die Anforderung, dass Administratoren die Dokumenteinstellungen überprüfen müssen.

Eine vollständige Liste der Windows-STIGs finden Sie in der STIGs-Dokumentbibliothek. Informationen zum Anzeigen der vollständigen Liste finden Sie unter STIG Viewing Tools.

> **Note**
>
> Die Härtungskomponenten STIG-Build-Windows-High umfassen alle aufgelisteten STIG-Einstellungen, die für die Härtungskomponenten STIG-Build-Windows-Low und STIG-Build-Windows-Medium AWSTOE gelten, zusätzlich zu den STIG-Einstellungen, die speziell für Sicherheitslücken der Kategorie I aufgeführt sind.

- Windows Server 2022 STIG Version 1 Version 1

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) anwendet, sowie:

V-254293, V-254352, V-254353, V-254354, V-254374, V-254378, V-254381, V-254446, V-254465, V-254466, V-254467, V-254469, V-254474, V-254475 und V-254500

- Windows Server 2019 STIG Version 2 Version 5

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) anwendet, sowie:

  V-205653, V-205654, V-205711, V-205713, V-205724, V-205725, V-205757, V-205802, V-205804, V-205805, V-205806, V-205849, V-205908, V-205913, V-205914 und V-205919

- Windows Server 2016 STIG Version 2, Version 5

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) anwendet, sowie:

  V-224874, V-224932, V-224933, V-224934, V-224954, V-224958, V-224961, V-225025, V-225044, V-225045, V-225046, V-225048, V-225053, V-225054 und V-225079

- Windows Server 2012 R2 MS STIG Version 3, Version 5

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) anwendet, sowie:

  V-225556, V-225552, V-225547, V-225507, V-225505, V-225498, V-225497, V-225496, V-225493, V-225492, V-225491, V-225449, V-225444, V-225399, V-225396, V-225390, V-225366, V-225365, V-225364, V-225354 und V-225274

- Microsoft.NET Framework 4.0 STIG Version 2 Version 2

  Beinhaltet alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) für das Microsoft.NET Framework anwendet. Für Sicherheitslücken der Kategorie I gelten keine zusätzlichen STIG-Einstellungen.

- Windows Firewall STIG Version 2 Version 1

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) anwendet, sowie:

  V-241992, V-241997 und V-242002

- Internet Explorer 11 STIG Version 2 Version 3

  Enthält alle unterstützten STIG-Einstellungen, die von der Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) in Internet Explorer 11 verwendet werden. Für Sicherheitslücken der Kategorie I gelten keine zusätzlichen STIG-Einstellungen.

- Microsoft Edge STIG Version 1 Version 6 (nur Windows Server 2022)

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) anwendet, sowie:

  V-235758 und V-235759

- Defender STIG Version 2 Version 4 (nur Windows Server 2022)

  Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) anwendet, sowie:

  V-213426, V-213452 und V-213453

STIG-Versionsverlaufsprotokoll für Windows

In diesem Abschnitt wird der Versionsverlauf der Windows-Hardening-Komponente für die vierteljährlichen STIG-Updates protokolliert. Um die Änderungen und veröffentlichten Versionen für ein Quartal zu sehen, wählen Sie den Titel aus, um die Informationen zu erweitern.

Änderungen im ersten Quartal 2024 — 06.02.2024 (keine Änderungen):

In der Version des ersten Quartals 2024 gab es keine Änderungen für die Windows-Komponente STIG.

Änderungen im vierten Quartal 2023 — 12.04.2023 (keine Änderungen):

Für die Version des vierten Quartals 2023 gab es keine Änderungen für die Windows-Komponente STIGS.

Änderungen im dritten Quartal 2023 — 04.10.2023 (keine Änderungen):

Für die Version des dritten Quartals 2023 gab es keine Änderungen für die Windows-Komponente STIGS.

Änderungen im zweiten Quartal 2023 — 05/03/2023 (keine Änderungen):

In der Version des zweiten Quartals 2023 gab es keine Änderungen für die Windows-Komponente STIG.

Änderungen im ersten Quartal 2023 — 27.03.2023 (keine Änderungen):

In der Version des ersten Quartals 2023 gab es keine Änderungen für die Windows-Komponente STIG.

Änderungen im vierten Quartal 2022 — 01.02.2023:

Die STIG-Versionen wurden aktualisiert und STIGS für die Q4-Version 2022 wie folgt angewendet:

STIG-Build-Windows-Low-Version 2022.4.x

- Windows Server 2022 STIG Version 1 Release 1
- Windows Server 2019 STIG Version 2 Release 5
- Windows Server 2016 STIG Version 2 Release 5
- Windows Server 2012 R2 MS STIG Version 3 Release 5
- Microsoft .NET Framework 4.0 STIG Version 2 Release 2
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 2 Release 3
- Microsoft Edge STIG Version 1 Version 6 (nur Windows Server 2022)

STIG-Build-Windows-Medium Version 2022.4.x

- Windows Server 2022 STIG Version 1 Release 1
- Windows Server 2019 STIG Version 2 Release 5
- Windows Server 2016 STIG Version 2 Release 5
- Windows Server 2012 R2 MS STIG Version 3 Release 5
- Microsoft .NET Framework 4.0 STIG Version 2 Release 2
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 2 Release 3
- Microsoft Edge STIG Version 1 Version 6 (nur Windows Server 2022)
- Defender STIG Version 2 Version 4 (nur Windows Server 2022)

STIG-Build-Windows-hohe Version 2022.4.x

- Windows Server 2022 STIG Version 1 Release 1
- Windows Server 2019 STIG Version 2 Release 5
- Windows Server 2016 STIG Version 2 Release 5
- Windows Server 2012 R2 MS STIG Version 3 Release 5
- Microsoft .NET Framework 4.0 STIG Version 2 Release 2
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 2 Release 3
- Microsoft Edge STIG Version 1 Version 6 (nur Windows Server 2022)
- Defender STIG Version 2 Version 4 (nur Windows Server 2022)

Änderungen im dritten Quartal 2022 — 30.09.2022 (keine Änderungen):

Für die Version des dritten Quartals 2022 gab es keine Änderungen für die Windows-Komponente STIGS.

Änderungen im zweiten Quartal 2022 — 08.02.2022:

Die STIG-Versionen wurden aktualisiert und STIGS für die Q2-Version 2022 angewendet.

STIG-Build-Windows-Low Version 1.5.x

- Windows Server 2019 STIG Version 2 Version 4
- Windows Server 2016 STIG Version 2 Version 4
- Windows Server 2012 R2 MS STIG Version 3 Version 3
- Microsoft.NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 1 Version 19

STIG-Build-Windows-Medium versie 1.5.x

- Windows Server 2019 STIG Version 2 Version 4
- Windows Server 2016 STIG Version 2 Version 4
- Windows Server 2012 R2 MS STIG Version 3 Version 3

- Microsoft.NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 1 Version 19

STIG-Build-Windows-hohe Version 1.5.x

- Windows Server 2019 STIG Version 2 Version 4
- Windows Server 2016 STIG Version 2 Version 4
- Windows Server 2012 R2 MS STIG Version 3 Version 3
- Microsoft.NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 1 Version 19

Änderungen im ersten Quartal 2022 — 08.02.2022 (keine Änderungen):

In der Version des ersten Quartals 2022 gab es keine Änderungen für die Windows-Komponente STIG.

Änderungen im vierten Quartal 2021 — 20.12.2021:

Die STIG-Versionen wurden aktualisiert und STIGS für die Version des vierten Quartals 2021 angewendet.

STIG-Build-Windows-Low Version 1.5.x

- Windows Server 2019 STIG Version 2 Version 3
- Windows Server 2016 STIG Version 2 Version 3
- Windows Server 2012 R2 MS STIG Version 3 Version 3
- Microsoft.NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 1 Version 19

STIG-Build-Windows-Medium versie 1.5.x

- Windows Server 2019 STIG Version 2 Version 3
- Windows Server 2016 STIG Version 2 Version 3

- Windows Server 2012 R2 MS STIG Version 3 Version 3
- Microsoft.NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 1 Version 19

STIG-Build-Windows-hohe Version 1.5.x

- Windows Server 2019 STIG Version 2 Version 3
- Windows Server 2016 STIG Version 2 Version 3
- Windows Server 2012 R2 MS STIG Version 3 Version 3
- Microsoft.NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 2 Release 1
- Internet Explorer 11 STIG Version 1 Version 19

Änderungen im dritten Quartal 2021 — 30.09.2021:

Die STIG-Versionen wurden aktualisiert und STIGS für die Version des dritten Quartals 2021 angewendet.

STIG-Build-Windows-Low Version 1.4.x

- Windows Server 2019 STIG Version 2 Version 2
- Windows Server 2016 STIG Version 2 Version 2
- Windows Server 2012 R2 MS STIG Version 3 Version 2
- Microsoft.NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 1 Version 7
- Internet Explorer 11 STIG Version 1 Version 19

STIG-Build-Windows-Medium versie 1.4.x

- Windows Server 2019 STIG Version 2 Version 2
- Windows Server 2016 STIG Version 2 Version 2
- Windows Server 2012 R2 MS STIG Version 3 Version 2
- Microsoft.NET Framework 4.0 STIG Version 2 Version 1

- Windows Firewall STIG Version 1 Version 7
- Internet Explorer 11 STIG Version 1 Version 19

STIG-Build-Windows-hohe Version 1.4.x

- Windows Server 2019 STIG Version 2 Version 2
- Windows Server 2016 STIG Version 2 Version 2
- Windows Server 2012 R2 MS STIG Version 3 Version 2
- Microsoft.NET Framework 4.0 STIG Version 2 Version 1
- Windows Firewall STIG Version 1 Version 7
- Internet Explorer 11 STIG Version 1 Version 19

Komponenten zum Härten von Linux STIG

Dieser Abschnitt enthält Informationen über Linux STIG-Hardening-Komponenten, gefolgt von einem Versionsverlauf. Wenn die Linux-Distribution keine eigenen STIG-Einstellungen hat, wendet die Hardening-Komponente die RHEL-Einstellungen an. Die Hardening-Komponente wendet unterstützte STIG-Einstellungen wie folgt auf die Infrastruktur an, die auf der Linux-Distribution basiert:

STIG-Einstellungen für Red Hat Enterprise Linux (RHEL) 7

- RHEL 7
- CentOS 7
- Amazon Linux 2 (AL2)

RHEL 8 STIG-Einstellungen

- RHEL 8
- CentOS 8
- Amazon Linux 2023 (AL 2023)

STIG-Build-Linux-Low-Version 2024.1.x

Die folgende Liste enthält STIG-Einstellungen, die die Hardening-Komponente auf Ihre Infrastruktur anwendet. Wenn eine unterstützte Einstellung für Ihre Infrastruktur nicht zutrifft, überspringt

die Härtungskomponente diese Einstellung und fährt fort. Beispielsweise gelten einige STIG-Einstellungen möglicherweise nicht für eigenständige Server. Unternehmensspezifische Richtlinien können sich auch darauf auswirken, welche Einstellungen die Härtungskomponente anwendet, z. B. die Anforderung, dass Administratoren die Dokumenteinstellungen überprüfen müssen.

Eine vollständige Liste finden Sie in der STIGs-Dokumentbibliothek. Informationen zum Anzeigen der vollständigen Liste finden Sie unter STIG Viewing Tools.

RHEL 7 STIG Version 3, Version 14

- RHEL 7/CentOS 7

  V-204452, V-204576 und V-204605
- AL 2

  V-204452, V-204576 und V-204605

RHEL 8 STIG Version 1 Version 13

- RHEL 8/CentOS 8/AL 2023

  V-230241, V-244527, V-230269, V-230270, V-230285, V-230253, V-230346, V-230381, V-230395, V-230468, V-230469, V-230491, V-230485, V-230486, V-230494, V-230495, V-230496, V-230497, V-230498, V-230499 und V-230281

Ubuntu 18.04 STIG Version 2 Version 13

V-219172, V-219173, V-219174, V-219175, V-219210, V-219164, V-219165, V-219178, V-219180, V-219301, V-219163, V-219332, V-219327 und V-219333

Ubuntu 20.04 STIG Version 1 Version 11

V-238202, V-238234, V-238235, V-238237, V-238323, V-238373, V-238221, V-238222, V-238223, V-238224, V-238226, V-238362, V-238357 und V-238308

STIG-Build-Linux-Medium, Version 2024.1.x

Die folgende Liste enthält STIG-Einstellungen, die die Hardening-Komponente auf Ihre Infrastruktur anwendet. Wenn eine unterstützte Einstellung für Ihre Infrastruktur nicht zutrifft, überspringt

die Härtungskomponente diese Einstellung und fährt fort. Beispielsweise gelten einige STIG-Einstellungen möglicherweise nicht für eigenständige Server. Unternehmensspezifische Richtlinien können sich auch darauf auswirken, welche Einstellungen die Härtungskomponente anwendet, z. B. die Anforderung, dass Administratoren die Dokumenteinstellungen überprüfen müssen.

Eine vollständige Liste finden Sie in der STIGs-Dokumentbibliothek. Informationen zum Anzeigen der vollständigen Liste finden Sie unter STIG Viewing Tools.

> **Note**
>
> Die Komponenten des Typs STIG-Build-Linux-Medium Hardening beinhalten alle aufgelisteten STIG-Einstellungen, die für STIG-Build-Linux-Low Hardening-Komponenten AWSTOE gelten, zusätzlich zu den STIG-Einstellungen, die speziell für Sicherheitslücken der Kategorie II aufgeführt sind.

RHEL 7 STIG Version 3, Version 14

Enthält alle unterstützten STIG-Einstellungen, die von der Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) in dieser Linux-Distribution verwendet werden, sowie:

- RHEL 7/CentOS 7

  V-204585, V-204490, V-204491, V-255928, V-204405, V-204406, V-204407, V-204408, V-204409, V-204410, V-204411, V-204412, V-204413, V-204414, V-204415, V-204422, V-204423, V-204427, V-204416, V-204418, V-204426, V-204431, V-204457, V-204466, V-204417, V-204434, V-204435, V-204587, V-204588, V-204589, V-204590, V-204591, V-204592, V-204593, V-204596, V-204597, V-204598, V-4599, V-204600, V-204601, V-204602, V-204622, V-233307, V-255925, V-204578, V-204595, V-204437, V-204503, V-204507, V-204508, V-204510, V-204511, V-204512, V-204514, V-204515, V-204516, V-20451717, V-204521, V-204524, V-204531, V-204536, V-204537, V-204538, V-204539, V-204540, V-204541, V-204542, V-204543, V-204544, V-204545, V-204546, V-204547, V-204548, V-204549, V-204550, V-204551, V-204552, V-204553, V-204554, V-204555, V-204556, V-204557, V-204558, V-204559, V-204560, V-204562, V-204563, V-204564, V-204565, V-204566, V-204567, V-204568, V-204572, V-204584, V-204609, V-204610, V-204611, V-204612, V-204613, V-204614, V-204615, V-204616, V-204617, V-204625, V-204630, V-255927, V-237634, V-237635, V-251703, V-204449, V-204450, V-204451, V-204619, V-204579, V-204631, V-204633 und V-256970

- AL 2:

V-204585, V-204490, V-204491, V-255928, V-204405, V-204406, V-204407, V-204408, V-204409, V-204410, V-204411, V-204412, V-204413, V-204414, V-204415, V-204422, V-204423, V-204427, V-204416, V-204418, V-204426, V-204431, V-204457, V-204466, V-204417, V-204434, V-204435, V-204587, V-204588, V-204589, V-204590, V-204591, V-204592, V-204593, V-204596, V-204597, V-204598, V-4599, V-204600, V-204601, V-204602, V-204622, V-233307, V-255925, V-204578, V-204595, V-204437, V-204503, V-204507, V-204508, V-204510, V-204511, V-204512, V-204514, V-204515, V-204516, V-20451717, V-204521, V-204524, V-204531, V-204536, V-204537, V-204538, V-204539, V-204540, V-204541, V-204542, V-204543, V-204544, V-204545, V-204546, V-204547, V-204548, V-204549, V-204550, V-204551, V-204552, V-204553, V-204554, V-204555, V-204556, V-204557, V-204558, V-204559, V-204560, V-204562, V-204563, V-204564, V-204565, V-204566, V-204567, V-204568, V-204572, V-204584, V-204609, V-204610, V-204611, V-204612, V-204613, V-204614, V-204615, V-204616, V-204617, V-204625, V-204630, V-255927, V-237634, V-237635, V-251703, V-204449, V-204450, V-204451, V-204619, V-204579, V-204631, V-204633 und V-256970

RHEL 8 STIG Version 1 Version 13

Enthält alle unterstützten STIG-Einstellungen, die von der Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) in dieser Linux-Distribution verwendet werden, sowie:

- RHEL 8/CentOS 8/AL 2023

  V-230257, V-230258, V-230259, V-230550, V-230248, V-230249, V-230250, V-230245, V-230246, V-230247, V-230397, V-230399, V-230400, V-230401, V-230228, V-230298, V-230387, V-230231, V-230233, V-230324, V-230365, V-230370, V-230378, V-230383, V-230236, V-230314, V-230315, V-244523, V-230266, V-230267, V-230268, V-230280, V-230310, V-230311, V-230312, V-230502, V-230532, V-230535, V-230536, V-230537, V-230538, V-230539, V-230540, V-230541, V-230542, V-230543, V-230544, V-230545, V-230546, V-230547, V-230548, V-230549, V-244550, V-244551, V-244552, V-244553, V-244554, V-250317, V-251718, V-230237, V-230313, V-230356, V-230357, V-230358, V-230359, V-230360, V-230361, V-230362, V-230363, V-230368, V-230369, V-230375, V-230376, V-230377, V-244524, V-244533, V-251713, V-251717, V-251714, V-251715, V-251716, V-230332, V-230334, V-230336, V-230338, V-230340, V-230342, V-230344, V-230333, V-230335, V-230337, V-230339, V-230341, V-230343, V-230345, V-230240, V-230282, V-250315, V-250316, V-230255, V-230277, V-230277, V-230277, V-230277, V-230277, V-230277, V-230230 278, V-230348, V-230353, V-230386,

V-230390, V-230392, V-230394, V-230396, V-230393, V-230398, V-230402, V-230403, V-230404, V-230405, V-230406, V-230407, V-230408, V-230409, V-230410, V-230411, V-230412, V-230413, V-230418, V-230419, V-230421, V-230422, V-230423, V-230424, V-230425, V-230426, V-230427, V-230428, V-230429, V-230430, V-230430, V-230430, V-230430, V-230430, V-230430, V-230430, V-230430, V-230430, V-230427 431, V-230432, V-230433, V-230434, V-230435, V-230436, V-230437, V-230438, V-230439, V-230444, V-230446, V-230447, V-230448, V-230449, V-230455, V-230456, V-230462, V-230463, V-230464, V-230465, V-230466, V-230467, V-230471, V-230472, V-230473, V-230474, V-230480, V-230483, V-244542, V-230503, V-230244, V-230286, V-230287, V-230288, V-230290, V-230291, V-230296, V-230330, V-230382, V-230526, V-230527, V-230555, V-230556, V-244526, V-244528, V-237642, V-237643, V-251711, V-230238, V-230239, V-230273, V-230275, V-230478, V-230488, V-230489, V-230559, V-230560, V-230560, V-230560, V-230230 561, V-237640 und V-256974

Ubuntu 18.04 STIG Version 2 Version 13

Enthält alle unterstützten STIG-Einstellungen, die von der Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) in dieser Linux-Distribution verwendet werden, sowie:

V-219188, V-219190, V-219191, V-219198, V-219199, V-219200, V-219201, V-219202, V-219203, V-219204, V-219205, V-219206, V-219207, V-219208, V-219209, V-219303, V-219326, V-219328, V-219330 V-219342, V-219189, V-219192, V-219193, V-219194, V-219315, V-219195, V-219196, V-219197, V-219213, V-219214, V-219215, V-219216, V-219217, V-219218, V-219219, V-219220, V-219221, V-219222 V-219223, V-219224, V-219227, V-219228, V-219229, V-219230, V-219231, V-219232, V-219233, V-219234, V-219235, V-219236, V-219238, V-219239, V-219240, V-219241, V-219242, V-219243, V-21924442, V-219250, V-219254, V-219257, V-219263, V-219264, V-219265, V-219266, V-219267, V-219268, V-219269, V-219270, V-219271, V-219272, V-219273, V-219274, V-219275, V-219276, V-219277, V-219279, V-219281, V-219287, V-219291, V-219297, V-219298, V-219299, V-219300, V-219309, V-219310, V-219311, V-219312, V-233779, V-233780, V-255906, V-219336, V-219338, V-219344, V-219181, V-219184, V-219186, V-219155, V-219156, V-219160, V-219306, V-219149, V-219166, V-219176, V-219339, V-219331, V-219337 und V-219335

Ubuntu 20.04 STIG Version 1 Version 11

Enthält alle unterstützten STIG-Einstellungen, die von der Hardening-Komponente für Sicherheitslücken der Kategorie III (Niedrig) in dieser Linux-Distribution verwendet werden, sowie:

V-238205, V-238207, V-238329, V-238337, V-238339, V-238340, V-238344, V-238345, V-238346, V-238347, V-238348, V-238349, V-238350, V-238351, V-238352, V-238376, V-238377, V-238378,

V-238209, V-238325, V-238330, V-238333, V-238369, V-238338, V-238341, V-238342, V-238343, V-238324, V-238353, V-238228, V-238225, V-238227, V-238299, V-238238, V-238239, V-238240, V-238241, V-238242, V-238244, V-238245, V-238246, V-238247, V-238248, V-238249, V-238250, V-238251, V-238252, V-238253, V-238254, V-238255, V-238256, V-238257, V-238258, V-238264, V-238268, V-238271, V-2382727 7, V-238278, V-238279, V-238280, V-238281, V-238282, V-238283, V-238284, V-238285, V-238286, V-238287, V-238288, V-238289, V-238290, V-238291, V-238292, V-238293, V-238294, V-238295, V-238297, V-238300, V-238301, V-238302, V-238304, V-238309, V-238310, V-238315, V-238316, V-238317, V-238318, V-238319, V-238320, V-251505, V-238360, V-238211, V-238212, V-238213, V-238216, V-238220, V-255912, V-238355, V-238236, V-238303, V-238358, V-238356, V-238359, V-238370 und V-238334

STIG-Build-Linux-Hochversion 2024.1.x

Die folgende Liste enthält STIG-Einstellungen, die die Hardening-Komponente auf Ihre Infrastruktur anwendet. Wenn eine unterstützte Einstellung für Ihre Infrastruktur nicht zutrifft, überspringt die Härtungskomponente diese Einstellung und fährt fort. Beispielsweise gelten einige STIG-Einstellungen möglicherweise nicht für eigenständige Server. Unternehmensspezifische Richtlinien können sich auch darauf auswirken, welche Einstellungen die Härtungskomponente anwendet, z. B. die Anforderung, dass Administratoren die Dokumenteinstellungen überprüfen müssen.

Eine vollständige Liste finden Sie in der STIGs-Dokumentbibliothek. Informationen zum Anzeigen der vollständigen Liste finden Sie unter STIG Viewing Tools.

> **Note**
>
> Die Härtungskomponenten STIG-Build-Linux-High beinhalten alle aufgelisteten STIG-Einstellungen, die für die Hardening-Komponenten STIG-Build-Linux-Low und STIG-Build-Linux-Medium AWSTOE gelten, zusätzlich zu den aufgelisteten STIG-Einstellungen, die speziell für Sicherheitslücken der Kategorie I gelten.

RHEL 7 STIG Version 3, Version 14

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) in dieser Linux-Distribution anwendet, sowie:

- RHEL 7/CentOS 7

V-204425, V-204594, V-204455, V-204424, V-204442, V-204443, V-204447, V-204448, V-204502, V-204620 und V-204621

- AL2:

  V-204425, V-204594, V-204455, V-204424, V-204442, V-204443, V-204447, V-204448, V-204502, V-204620 und V-204621

RHEL 8 STIG Version 1 Version 13

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) in dieser Linux-Distribution anwendet, sowie:

- RHEL 8/CentOS 8/AL 2023

  V-230265, V-230529, V-230531, V-230264, V-230487, V-230492, V-230533 und V-230558

Ubuntu 18.04 STIG Version 2 Version 13

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) in dieser Linux-Distribution anwendet, sowie:

V-219157, V-219158, V-219177, V-219212, V-219308, V-219314, V-219316 und V-251507

Ubuntu 20.04 STIG Version 1 Version 11

Enthält alle unterstützten STIG-Einstellungen, die die Hardening-Komponente für Sicherheitslücken der Kategorien II und III (Mittel und Niedrig) in dieser Linux-Distribution anwendet, sowie:

V-238218, V-238219, V-238201, V-238326, V-238327, V-238380 und V-251504

STIG-Versionsverlaufsprotokoll für Linux

In diesem Abschnitt wird der Versionsverlauf der Linux-Komponenten protokolliert. Um die Änderungen und veröffentlichten Versionen für ein Quartal zu sehen, wählen Sie den Titel aus, um die Informationen zu erweitern.

Änderungen im ersten Quartal 2024 — 06.02.2024:

Die STIG-Versionen wurden aktualisiert und STIGS für die Version des ersten Quartals 2024 wie folgt angewendet:

STIG-Build-Linux-Low Version 2024.1.x

- RHEL 7 STIG Version 3 Version 14
- RHEL 8 STIG Version 1 Version 13
- Ubuntu 18.04 STIG Version 2 Version 13
- Ubuntu 20.04 STIG Version 1 Version 11

STIG-Build-Linux-Medium versie 2024.1.x

- RHEL 7 STIG Version 3 Version 14
- RHEL 8 STIG Version 1 Version 13
- Ubuntu 18.04 STIG Version 2 Version 13
- Ubuntu 20.04 STIG Version 1 Version 11

STIG-Build-Linux-hohe Version 2024.1.x

- RHEL 7 STIG Version 3 Version 14
- RHEL 8 STIG Version 1 Version 13
- Ubuntu 18.04 STIG Version 2 Version 13
- Ubuntu 20.04 STIG Version 1 Version 11

Änderungen im vierten Quartal 2023 — 12/07/2023:

Die STIG-Versionen wurden aktualisiert und STIGS für die Version des vierten Quartals 2023 wie folgt angewendet:

STIG-Build-Linux-Low Version 2023.4.x

- RHEL 7 STIG Version 3 Version 13
- RHEL 8 STIG Version 1 Version 12
- Ubuntu 18.04 STIG Version 2 Version 12
- Ubuntu 20.04 STIG Version 1 Version 10

STIG-Build-Linux-Medium versie 2023.4.x

- RHEL 7 STIG Version 3 Version 13
- RHEL 8 STIG Version 1 Version 12
- Ubuntu 18.04 STIG Version 2 Version 12
- Ubuntu 20.04 STIG Version 1 Version 10

STIG-Build-Linux-hohe Version 2023.4.x

- RHEL 7 STIG Version 3 Version 13
- RHEL 8 STIG Version 1 Version 12
- Ubuntu 18.04 STIG Version 2 Version 12
- Ubuntu 20.04 STIG Version 1 Version 10

Änderungen im dritten Quartal 2023 — 04.10.2023:

Die STIG-Versionen wurden aktualisiert und STIGS für die Version des dritten Quartals 2023 wie folgt angewendet:

STIG-Build-Linux-Low Version 2023.3.x

- RHEL 7 STIG Version 3 Version 12
- RHEL 8 STIG Version 1 Version 11
- Ubuntu 18.04 STIG Version 2 Version 11
- Ubuntu 20.04 STIG Version 1 Version 9

STIG-Build-Linux-Medium versie 2023.3.x

- RHEL 7 STIG Version 3 Version 12
- RHEL 8 STIG Version 1 Version 11
- Ubuntu 18.04 STIG Version 2 Version 11
- Ubuntu 20.04 STIG Version 1 Version 9

STIG-Build-Linux-hohe Version 2023.3.x

- RHEL 7 STIG Version 3 Version 12
- RHEL 8 STIG Version 1 Version 11
- Ubuntu 18.04 STIG Version 2 Version 11
- Ubuntu 20.04 STIG Version 1 Version 9

Änderungen im zweiten Quartal 2023 — 05/03/2023:

Die STIG-Versionen wurden aktualisiert und STIGS für die Version des zweiten Quartals 2023 wie folgt angewendet:

STIG-Build-Linux-Low Version 2023.2.x

- RHEL 7 STIG Version 3 Version 11
- RHEL 8 STIG Version 1 Version 10
- Ubuntu 18.04 STIG Version 2 Version 11
- Ubuntu 20.04 STIG Version 1 Veröffentlichung 8

STIG-Build-Linux-Medium versie 2023.2.x

- RHEL 7 STIG Version 3 Version 11
- RHEL 8 STIG Version 1 Version 10
- Ubuntu 18.04 STIG Version 2 Version 11
- Ubuntu 20.04 STIG Version 1 Veröffentlichung 8

STIG-Build-Linux-hohe Version 2023.2.x

- RHEL 7 STIG Version 3 Version 11
- RHEL 8 STIG Version 1 Version 10
- Ubuntu 18.04 STIG Version 2 Version 11
- Ubuntu 20.04 STIG Version 1 Veröffentlichung 8

Änderungen im ersten Quartal 2023 — 27.03.2023:

Die STIG-Versionen wurden wie folgt aktualisiert und STIGS für die Version des ersten Quartals 2023 angewendet:

STIG-Build-Linux-Low Version 2023.1.x

- RHEL 7 STIG Version 3 Version 10
- RHEL 8 STIG Version 1 Version 9
- Ubuntu 18.04 STIG Version 2 Version 10
- Ubuntu 20.04 STIG Version 1 Veröffentlichung 7

STIG-Build-Linux-Medium versie 2023.1.x

- RHEL 7 STIG Version 3 Version 10
- RHEL 8 STIG Version 1 Version 9
- Ubuntu 18.04 STIG Version 2 Version 10
- Ubuntu 20.04 STIG Version 1 Veröffentlichung 7

STIG-Build-Linux-hohe Version 2023.1.x

- RHEL 7 STIG Version 3 Version 10
- RHEL 8 STIG Version 1 Version 9
- Ubuntu 18.04 STIG Version 2 Version 10
- Ubuntu 20.04 STIG Version 1 Veröffentlichung 7

Änderungen im vierten Quartal 2022 — 01.02.2023:

Die STIG-Versionen wurden aktualisiert und STIGS für die Version des vierten Quartals 2022 wie folgt angewendet:

STIG-Build-Linux-Low Version 2022.4.x

- RHEL 7 STIG Version 3 Version 9
- RHEL 8 STIG Version 1 Version 8
- Ubuntu 18.04 STIG Version 2 Version 9

- Ubuntu 20.04 STIG Version 1 Veröffentlichung 6

STIG-Build-Linux-Medium versie 2022.4.x

- RHEL 7 STIG Version 3 Version 9
- RHEL 8 STIG Version 1 Version 8
- Ubuntu 18.04 STIG Version 2 Version 9
- Ubuntu 20.04 STIG Version 1 Veröffentlichung 6

STIG-Build-Linux-hohe Version 2022.4.x

- RHEL 7 STIG Version 3 Version 9
- RHEL 8 STIG Version 1 Version 8
- Ubuntu 18.04 STIG Version 2 Version 9
- Ubuntu 20.04 STIG Version 1 Veröffentlichung 6

Änderungen im dritten Quartal 2022 — 30.09.2022 (keine Änderungen):

Für die Version des dritten Quartals 2022 gab es keine Änderungen für die Linux-Komponente STIGS.

Änderungen im zweiten Quartal 2022 — 08.02.2022:

Die Ubuntu-Unterstützung wurde eingeführt, die STIG-Versionen aktualisiert und STIGS für die Version des zweiten Quartals 2022 wie folgt angewendet:

STIG-Build-Linux-Low Version 2022.2.x

- RHEL 7 STIG Version 3 Version 7
- RHEL 8 STIG Version 1 Version 6
- Ubuntu 18.04 STIG Version 2 Version 6 (neu)
- Ubuntu 20.04 STIG Version 1 Version 4 (neu)

STIG-Build-Linux-Medium Version 2022.2.x

- RHEL 7 STIG Version 3 Version 7

- RHEL 8 STIG Version 1 Version 6
- Ubuntu 18.04 STIG Version 2 Version 6 (neu)
- Ubuntu 20.04 STIG Version 1 Version 4 (neu)

STIG-Build-Linux-Hochversion 2022.2.x

- RHEL 7 STIG Version 3 Version 7
- RHEL 8 STIG Version 1 Version 6
- Ubuntu 18.04 STIG Version 2 Version 6 (neu)
- Ubuntu 20.04 STIG Version 1 Version 4 (neu)

Änderungen im ersten Quartal 2022 — 26.04.2022:

Umgestaltet, um nun eine bessere Unterstützung für Container zu bieten. Kombiniert das vorherige AL2-Skript mit RHEL 7. Die STIG-Versionen wurden aktualisiert und STIGS für die Version des ersten Quartals 2022 wie folgt angewendet:

STIG-Build-Linux-Low Version 3.6.x

- RHEL 7 STIG Version 3 Version 6
- RHEL 8 STIG Version 1 Version 5

STIG-Build-Linux-Medium, Version 3.6.x

- RHEL 7 STIG Version 3 Version 6
- RHEL 8 STIG Version 1 Version 5

STIG-Build-Linux-High Version 3.6.x

- RHEL 7 STIG Version 3 Version 6
- RHEL 8 STIG Version 1 Version 5

Änderungen im vierten Quartal 2021 — 20.12.2021:

Die STIG-Versionen wurden aktualisiert und STIGS für die Version des vierten Quartals 2021 wie folgt angewendet:

STIG-Build-Linux-Low Version 3.5.x

- RHEL 7 STIG Version 3 Version 5
- RHEL 8 STIG Version 1 Version 4

STIG-Build-Linux-Medium, Version 3.5.x

- RHEL 7 STIG Version 3 Version 5
- RHEL 8 STIG Version 1 Version 4

STIG-Build-Linux-hohe Version 3.5.x

- RHEL 7 STIG Version 3 Version 5
- RHEL 8 STIG Version 1 Version 4

Änderungen im dritten Quartal 2021 — 30.09.2021:

Die STIG-Versionen wurden aktualisiert und STIGS für die Version des dritten Quartals 2021 wie folgt angewendet:

STIG-Build-Linux-Low Version 3.4.x

- RHEL 7 STIG Version 3 Version 4
- RHEL 8 STIG Version 1 Version 3

STIG-Build-Linux-Medium, Version 3.4.x

- RHEL 7 STIG Version 3 Version 4
- RHEL 8 STIG Version 1 Version 3

STIG-Build-Linux-High Version 3.4.x

- RHEL 7 STIG Version 3 Version 4
- RHEL 8 STIG Version 1 Version 3

Komponente zur Validierung der SCAP-Konformität

Das Security Content Automation Protocol (SCAP) besteht aus einer Reihe von Standards, anhand derer IT-Experten Sicherheitslücken in Anwendungen identifizieren können, um die Einhaltung der Vorschriften zu gewährleisten. Der SCAP Compliance Checker (SCC) ist ein SCAP-validiertes Scan-Tool, das vom Naval Information Warfare Center (NIWC) Atlantic veröffentlicht wurde. Weitere Informationen finden Sie unter Security Content Automation Protocol (SCAP) Compliance Checker (SCC) auf der Website von NIWC Atlantic.

Die AWSTOE `scap-compliance-checker-windows` `scap-compliance-checker-linux` Komponenten laden den SCC-Scanner herunter und installieren ihn auf den Build- und Testinstanzen der Pipeline. Wenn der Scanner ausgeführt wird, führt er authentifizierte Konfigurationsscans mithilfe von DISA SCAP-Benchmarks durch und erstellt einen Bericht, der die folgenden Informationen enthält. AWSTOE schreibt die Informationen auch in Ihre Anwendungsprotokolle.

- STIG-Einstellungen, die auf die Instanz angewendet werden.
- Eine allgemeine Konformitätsbewertung für die Instanz.

Wir empfehlen, dass Sie die SCAP-Validierung als letzten Schritt Ihres Build-Prozesses ausführen, um sicherzustellen, dass Sie genaue Ergebnisse der Konformitätsvalidierung melden.

> **Note**
>
> Sie können die Berichte mit einem der STIG Viewing Tools überprüfen. Diese Tools sind online über den DoD Cyber Exchange verfügbar.

In den folgenden Abschnitten werden die Benchmarks beschrieben, die die SCAP-Validierungskomponenten beinhalten.

scap-compliance-checker-linux Version 2021.04.0

Die `scap-compliance-checker-linux` Komponente wird auf den Build- und Testinstanzen der Image Builder Builder-Pipeline ausgeführt. AWSTOE protokolliert sowohl den Bericht als auch das Ergebnis, das die SCC-Anwendung generiert.

Die Komponente führt die folgenden Workflow-Schritte aus:

1. Lädt die SCC-Anwendung herunter und installiert sie.

2. Importiert die Konformitäts-Benchmarks.
3. Führt die Validierung mithilfe der SCC-Anwendung aus.
4. Speichert den Konformitätsbericht und die Bewertung lokal auf dem Desktop der Build-Instanz.
5. Protokolliert den Konformitätswert aus dem lokalen Bericht in den AWSTOE Anwendungsprotokolldateien.

> **Note**
>
> AWSTOE unterstützt derzeit die SCAP-Konformitätsprüfung für Windows Server 2012 R2, 2016 und 2019.

Die Komponente SCAP-Konformitätsprüfung für Windows umfasst die folgenden Benchmarks:

SCC-Version: 5.4.2

Benchmarks für das vierte Quartal 2021:

- U_MS__Framework_4-0_V2R1_STIG_SCAP_1-2_Benchmark DotNet
- u_MS_IE11_V2R1_STIG_SCAP_1-2_Benchmark
- u_MS_Windows_2012_und_2012_R2_MS_V3R2_STIG_SCAP_1-2_Benchmark
- u_MS_Windows_Defender_AV_V2R2_STIG_SCAP_1-2_Benchmark
- u_MS_Windows_Server_2016_V2R1_STIG_SCAP_1-2_Benchmark
- u_MS_Windows_Server_2019_V2R1_STIG_SCAP_1-2_Benchmark
- u_MS_Windows_Firewall_v2R1_STIG_SCAP_1-2_Benchmark
- u_CAN_Ubuntu_18-04_v2R4_STIG_SCAP_1-2_Benchmark
- u_RHEL_7_v3R5_STIG_SCAP_1-2_Benchmark
- u_RHEL_8_V1R3_STIG_SCAP_1-2_Benchmark

scap-compliance-checker-linux Ausführung 2021.04.0

Die `scap-compliance-checker-linux` Komponente wird auf den Build- und Testinstanzen der Image Builder Builder-Pipeline ausgeführt. AWSTOE protokolliert sowohl den Bericht als auch das Ergebnis, das die SCC-Anwendung generiert.

Die Komponente führt die folgenden Workflow-Schritte aus:

1. Lädt die SCC-Anwendung herunter und installiert sie.
2. Importiert die Konformitäts-Benchmarks.
3. Führt die Validierung mithilfe der SCC-Anwendung aus.
4. Speichert den Konformitätsbericht und die Bewertung lokal am folgenden Speicherort auf der Build-Instanz:`/opt/scc/SCCResults`.
5. Protokolliert die Konformitätsbewertung aus dem lokalen Bericht in den AWSTOE Anwendungsprotokolldateien.

> **Note**
>
> AWSTOE unterstützt derzeit die SCAP-Konformitätsvalidierung für RHEL 7/8 und Ubuntu 18. Die SCC-Anwendung unterstützt derzeit die x86-Architektur für die Validierung.

Die SCAP-Compliance-Checker-Komponente für Linux umfasst die folgenden Benchmarks:

SCC-Version: 5.4.2

Benchmarks für das vierte Quartal 2021:

- u_CAN_Ubuntu_18-04_v2R4_STIG_SCAP_1-2_Benchmark
- u_RHEL_7_v3R5_STIG_SCAP_1-2_Benchmark
- u_RHEL_8_V1R3_STIG_SCAP_1-2_Benchmark
- U_MS__Framework_4-0_V2R1_STIG_SCAP_1-2_Benchmark DotNet
- u_MS_IE11_V2R1_STIG_SCAP_1-2_Benchmark
- u_MS_Windows_2012_und_2012_R2_MS_V3R2_STIG_SCAP_1-2_Benchmark
- u_MS_Windows_Defender_AV_V2R2_STIG_SCAP_1-2_Benchmark
- u_MS_Windows_Server_2016_V2R1_STIG_SCAP_1-2_Benchmark
- u_MS_Windows_Server_2019_V2R1_STIG_SCAP_1-2_Benchmark
- u_MS_Windows_Firewall_v2R1_STIG_SCAP_1-2_Benchmark

Versionsgeschichte von SCAP

In der folgenden Tabelle werden wichtige Änderungen an der SCAP-Umgebung und die in diesem Dokument beschriebenen Einstellungen beschrieben.

Änderung	Beschreibung	Datum
SCAP-Komponenten wurden hinzugefügt	Die folgenden SCAP-Komponenten wurden eingeführt: • scap-compliance-checker-linux Version 2021.04.0 wurde erstellt (SCC-Version: 5.4.2) • scap-compliance-checker-linux Version 2021.04.0 wurde erstellt (SCC-Version: 5.4.2)	20. Dezember 2021

AWSTOE Befehlsreferenz

AWSTOE ist eine Komponentenverwaltungsanwendung, die in der ausgeführt wird AWS CLI.

> **Note**
>
> Für einige AWSTOE Aktionsmodule sind erhöhte Berechtigungen erforderlich, um auf einem Linux-Server ausgeführt zu werden. Um erhöhte Berechtigungen zu verwenden, stellen Sie der Befehlssyntax ein Präfix voransudo, oder führen Sie den sudo su Befehl einmal aus, wenn Sie sich anmelden, bevor Sie die unten verlinkten Befehle ausführen. Weitere Informationen zu AWSTOE Aktionsmodulen finden Sie unterAktionsmodule, die vom AWSTOE Komponentenmanager unterstützt werden.

[run](#)

Verwenden Sie den run Befehl, um die YAML-Dokumentskripts für ein oder mehrere Komponentendokumente auszuführen.

[validieren](#)

Führen Sie den validate Befehl aus, um die YAML-Dokumentsyntax für ein oder mehrere Komponentendokumente zu überprüfen.

Befehl awstoe run

Mit diesem Befehl werden die Dokumentskripts der YAML-Komponente in der Reihenfolge ausgeführt, in der sie in der durch den `--config` Parameter angegebenen Konfigurationsdatei oder in der durch den `--documents` Parameter angegebenen Liste der Komponentendokumente enthalten sind.

> **Note**
>
> Sie müssen genau einen der folgenden Parameter angeben, niemals beide:
> --config
> --dokumente

Syntax

```
awstoe run [--config <file path>] [--cw-ignore-failures <?>]
      [--cw-log-group <?>] [--cw-log-region us-west-2] [--cw-log-stream <?>]
      [--document-s3-bucket-owner <owner>] [--documents <file path,file path,...>]
      [--execution-id <?>] [--log-directory <file path>]
      [--log-s3-bucket-name <name>] [--log-s3-bucket-owner <owner>]
      [--log-s3-key-prefix <?>] [--parameters name1=value1,name2=value2...]
      [--phases <phase name>] [--state-directory <directory path>] [--version <?>]
      [--help] [--trace]
```

Parameter und Optionen

Parameter

--config ***./config-example.json***

Kurzform: -c *./config-example.json*

Die Konfigurationsdatei (bedingt). Dieser Parameter enthält den Dateispeicherort für die JSON-Datei, die Konfigurationseinstellungen für die Komponenten enthält, die dieser Befehl ausführt. Wenn Sie run Befehlseinstellungen in einer Konfigurationsdatei angeben, dürfen Sie den `--documents` Parameter nicht angeben. Weitere Hinweise zur Eingabekonfiguration finden Sie unterEingabe für den Befehl AWSTOE run konfigurieren.

Zu den gültigen Standorten gehören:

- Ein lokaler Dateipfad (*./config-example.json*)
- Ein S3-URI (`s3://`*bucket/key*)

--cw-ignore-failures

Kurzform: N/A

Ignorieren Sie Fehler bei der Protokollierung in den CloudWatch Protokollen.

--cw-log-group

Kurzform: N/A

Der `LogGroup` Name für die CloudWatch Logs.

--cw-log-region

Kurzform: N/A

Die AWS Region, die für die CloudWatch Logs gilt.

--cw-log-stream

Kurzform: N/A

Der `LogStream` Name für die CloudWatch Logs, der angibt, AWSTOE wohin die `console.log` Datei gestreamt werden soll.

--document-s3-bucket-owner

Kurzform: N/A

Die Konto-ID des Bucket-Besitzers für S3-URI-basierte Dokumente.

--Dokumente, ./doc-1.yaml ./doc-n.yaml

Kurzform: *./doc-1.yaml*-d, *./doc-n*

Die Komponentendokumente (bedingt). Dieser Parameter enthält eine durch Kommas getrennte Liste von Dateispeicherorten, an denen die YAML-Komponentendokumente ausgeführt werden sollen. Wenn Sie mithilfe des Parameters YAML-Dokumente für den run Befehl angeben, dürfen Sie den `--documents` Parameter nicht angeben. `--config`

Zu den gültigen Speicherorten gehören:

- lokale Dateipfade (*. /component-doc-example.yaml).*

- 3S-URIs (). `s3://bucket/key`
- *ARNs für die Build-Version der Image Builder-Komponente (arn:aws:imagebuilder:us-west-2:123456789012:component/ /2021.12.02/1). my-example-component*

> **Note**
>
> Zwischen den Elementen in der Liste gibt es keine Leerzeichen, nur Kommas.

--execution-id

Kurzform: -i

Dies ist die eindeutige ID, die für die Ausführung des aktuellen run Befehls gilt. Diese ID ist in den Namen der Ausgabe- und Protokolldateien enthalten, um diese Dateien eindeutig zu identifizieren und sie mit der aktuellen Befehlsausführung zu verknüpfen. Wenn diese Einstellung weggelassen wird, wird eine GUID AWSTOE generiert.

--log-directory

Kurzform: -l

Das Zielverzeichnis, in dem alle Protokolldateien dieser Befehlsausführung AWSTOE gespeichert werden. Standardmäßig befindet sich dieses Verzeichnis im folgenden übergeordneten Verzeichnis:`TOE_<DATETIME>_<EXECUTIONID>`. Wenn Sie das Protokollverzeichnis nicht angeben, AWSTOE wird das aktuelle Arbeitsverzeichnis (`.`) verwendet.

--log-s3-Bucket-Name

Kurzform: -b

Wenn Komponentenprotokolle in Amazon S3 gespeichert sind (empfohlen), werden die Komponentenanwendungsprotokolle in den in diesem Parameter genannten S3-Bucket AWSTOE hochgeladen.

--log-s3-bucket-owner

Kurzform: N/A

Wenn Komponentenprotokolle in Amazon S3 gespeichert werden (empfohlen), ist dies die Eigentümerkonto-ID für den Bucket, in den die Protokolldateien AWSTOE geschrieben werden.

--log-s3-Schlüsselpräfix

Kurzform: -k

Wenn Komponentenprotokolle in Amazon S3 gespeichert werden (empfohlen), ist dies das S3-Objektschlüsselpräfix für den Protokollspeicherort im Bucket.

--parameters Name1 = Wert1, Name2 = Wert2...

Kurzform: N/A

Parameter sind veränderbare Variablen, die im Komponentendokument definiert sind und deren Einstellungen die aufrufende Anwendung zur Laufzeit bereitstellen kann.

--phasen

Kurzform: -p

Eine durch Kommas getrennte Liste, die angibt, welche Phasen in den Dokumenten der YAML-Komponente ausgeführt werden sollen. Wenn ein Komponentendokument zusätzliche Phasen enthält, werden diese nicht ausgeführt.

--state-directory

Kurzform: -s

Der Dateipfad, in dem State-Tracking-Dateien gespeichert werden.

--version

Kurzform: -v

Gibt die Version der Komponentenanwendung an.

Optionen

-h

Kurzform: -h

Zeigt ein Hilfehandbuch zur Verwendung der Anwendungsoptionen für die Komponentenverwaltung an.

--trace

Kurzform: -t

Aktiviert die ausführliche Protokollierung auf der Konsole.

Befehl awstoe validate

Wenn Sie diesen Befehl ausführen, validiert er die YAML-Dokumentsyntax für jedes der im Parameter angegebenen Komponentendokumente. `--documents`

Syntax

```
awstoe validate [--document-s3-bucket-owner <owner>]
      --documents <file path,file path,...> [--help] [--trace]
```

Parameter und Optionen

Parameter

--document-s3-bucket-owner

Kurzform: N/A

Quellkonto-ID der bereitgestellten S3-URI-basierten Dokumente.

--Dokumente, ./doc-1.yaml ./doc-n.yaml

Kurzform: *./doc-1.yaml*-d, *./doc-n*

Die Komponentendokumente (erforderlich). Dieser Parameter enthält eine durch Kommas getrennte Liste von Dateispeicherorten für die auszuführenden YAML-Komponentendokumente. Zu den gültigen Speicherorten gehören:

- lokale Dateipfade (*. /component-doc-example.yaml*)
- 3S-URIs () `s3://`*bucket/key*
- *ARNs für die Build-Version der Image Builder-Komponente (arn:aws:imagebuilder:us-west-2:123456789012:component/ /2021.12.02/1) my-example-component*

> **Note**
>
> Zwischen den Elementen in der Liste gibt es keine Leerzeichen, nur Kommas.

Optionen

-h

Kurzform: -h

Zeigt ein Hilfehandbuch zur Verwendung der Anwendungsoptionen für die Komponentenverwaltung an.

--trace

Kurzform: -t

Aktiviert die ausführliche Protokollierung auf der Konsole.

EC2 Image Builder Builder-Ressourcen verwalten

Ressourcen sind die Bausteine, aus denen Image-Pipelines bestehen, sowie die Bilder, die diese Pipelines erzeugen. In diesem Kapitel werden die Erstellung, Verwaltung und gemeinsame Nutzung von Image Builder Builder-Ressourcen, einschließlich Komponenten, Rezepten und Images, sowie die Einstellungen für die Konfiguration und Verteilung der Infrastruktur behandelt.

> **Note**
>
> Um Ihnen bei der Verwaltung Ihrer Image Builder Builder-Ressourcen zu helfen, können Sie jeder Ressource Ihre eigenen Metadaten in Form von Tags zuweisen. Sie verwenden Tags, um Ihre AWS Ressourcen auf unterschiedliche Weise zu kategorisieren, z. B. nach Zweck, Eigentümer oder Umgebung. Dies ist nützlich, wenn Sie viele Ressourcen desselben Typs haben. Sie können eine bestimmte Ressource anhand der Tags, die Sie ihr zugewiesen haben, leichter identifizieren.
> Weitere Informationen zum Taggen Ihrer Ressourcen mithilfe von Image Builder Builder-Befehlen finden Sie im AWS CLIMarkieren von Ressourcen Abschnitt dieses Handbuchs.

Inhalt

Komponenten mit Image Builder verwalten

Image Builder verwendet die Komponentenverwaltungsanwendung AWS Task Orchestrator and Executor (AWSTOE), um komplexe Workflows zu orchestrieren. Komponenten zum Erstellen und Testen, die mit der AWSTOE Anwendung funktionieren, basieren auf YAML-Dokumenten, in denen die Skripts zum Anpassen oder Testen Ihres Images definiert sind. Für AMI-Images installiert Image Builder Komponenten und die AWSTOE Komponentenverwaltungsanwendung auf seinen Amazon EC2 EC2-Build- und Test-Instances. Bei Container-Images werden die Komponenten und die AWSTOE Komponentenverwaltungsanwendung innerhalb des laufenden Containers installiert.

Image Builder verwendet AWSTOE , um alle Aktivitäten auf der Instanz auszuführen. Es ist kein zusätzliches Setup erforderlich, mit dem Sie interagieren können AWSTOE , wenn Sie Image Builder Builder-Befehle ausführen oder die Image Builder Builder-Konsole verwenden.

> **Note**
>
> Wenn eine von Amazon verwaltete Komponente das Ende ihrer Support-Laufzeit erreicht, wird sie nicht mehr gewartet. Ungefähr vier Wochen zuvor erhalten alle Konten, die die Komponente verwenden, eine Benachrichtigung und eine Liste der betroffenen Rezepte in ihrem Konto von ihrem AWS Health Dashboard Konto. Weitere Informationen AWS Health dazu finden Sie im AWS Health Benutzerhandbuch.

Workflow-Phasen für die Erstellung eines neuen Images

Der Image Builder Builder-Workflow zum Erstellen neuer Images umfasst die folgenden zwei unterschiedlichen Phasen.

1. Build-Phase (Pre-Snapshot) — Während der Build-Phase nehmen Sie Änderungen an der Amazon EC2 EC2-Build-Instance vor, auf der Ihr Basis-Image ausgeführt wird, um die Baseline für Ihr neues Image zu erstellen. Ihr Rezept kann beispielsweise Komponenten enthalten, die eine Anwendung installieren oder die Firewall-Einstellungen des Betriebssystems ändern.

   Die folgenden Komponentenphasen werden während der Erstellungsphase ausgeführt:

   - build
   - validieren

   Nach erfolgreichem Abschluss dieser Phase erstellt Image Builder einen Snapshot oder ein Container-Image, das für die Testphase und darüber hinaus verwendet wird.

2. Testphase (nach dem Snapshot) — Während der Testphase gibt es einige Unterschiede zwischen Images, die AMIs erstellen, und Container-Images. Für AMI-Workflows startet Image Builder eine EC2-Instance aus dem Snapshot, den es als letzten Schritt der Buildphase erstellt hat. Tests werden auf der neuen Instance ausgeführt, um die Einstellungen zu überprüfen und sicherzustellen, dass die Instance wie erwartet funktioniert. Bei Container-Workflows werden die Tests auf derselben Instanz ausgeführt, die für die Erstellung verwendet wurde.

   Die folgende Komponentenphase wird für jede Komponente ausgeführt, die während der Testphase im Rezept enthalten ist:

   - Test

   Diese Komponentenphase gilt sowohl für den Build- als auch für den Test-Komponententyp. Nach erfolgreichem Abschluss dieser Phase kann Image Builder Ihr endgültiges Image aus dem Snapshot oder dem Container-Image erstellen und verteilen.

> **Note**
>
> Sie AWSTOE können zwar viele Phasen in einem Komponentendokument definieren, aber Image Builder hat strenge Regeln dafür, welche Phasen es ausführt und in welchen Phasen es sie ausführt. Damit eine Komponente während der Erstellungsphase ausgeführt werden kann, muss das Komponentendokument mindestens eine dieser Phasen definieren: `build` oder`validate`. Damit eine Komponente während der Testphase ausgeführt werden kann, muss das Komponentendokument die `test` Phase und keine anderen Phasen definieren. Da Image Builder die Stufen unabhängig voneinander ausführt, kann die Verkettung von Verweisen in Komponentendokumenten keine Stufengrenzen überschreiten. Es ist nicht möglich, einen Wert aus einer Phase, die in der Buildphase ausgeführt wird, mit einer Phase, die in der Testphase ausgeführt wird, zu verknüpfen. Sie können jedoch Eingabeparameter für das vorgesehene Ziel definieren und Werte über die Befehlszeile übergeben. Weitere Informationen zum Einstellen von Komponentenparametern in Ihren Image Builder Builder-Rezepten finden Sie unterAWSTOE Komponentenparameter mit EC2 Image Builder verwalten.

Zur Unterstützung bei der Problembehandlung auf Ihrer Build- oder Testinstanz AWSTOE wird ein Protokollordner erstellt, der das Eingabedokument und die Protokolldateien enthält, um zu verfolgen, was bei jeder Ausführung einer Komponente passiert. Wenn Sie in Ihrer Pipeline-Konfiguration einen Amazon S3 S3-Bucket konfiguriert haben, werden die Protokolle auch dort geschrieben. Weitere

Informationen zu YAML-Dokumenten und der Protokollausgabe finden Sie unterVerwenden Sie Komponentendokumente in AWSTOE.

> **Tip**
>
> Wenn Sie viele Komponenten im Auge behalten müssen, hilft Ihnen das Tagging dabei, eine bestimmte Komponente oder Version anhand der Tags zu identifizieren, die Sie ihr zugewiesen haben. Weitere Informationen zum Taggen Ihrer Ressourcen mithilfe von Image Builder Builder-Befehlen finden Sie im AWS CLIMarkieren von Ressourcen Abschnitt dieses Handbuchs.

In diesem Abschnitt wird beschrieben, wie Sie Komponenten mithilfe der Image Builder Builder-Konsole oder mit Befehlen in der auflisten, anzeigen, erstellen und importieren AWS CLI.

Inhalt

Erstellen Sie ein YAML-Komponentendokument

Um eine Komponente zu erstellen, stellen Sie ein Dokument mit einer YAML-Anwendungskomponente bereit. Dies stellt die Phasen und Schritte dar, die Sie zum Erstellen der Komponente benötigen.

Die Beispiele in diesem Abschnitt erstellen eine Build-Komponente, die das `UpdateOS` Aktionsmodul in der AWSTOE Komponentenverwaltungsanwendung aufruft. Das Modul aktualisiert das Betriebssystem. Weitere Hinweise zum `UpdateOS` Aktionsmodul finden Sie unterOS aktualisieren. Weitere Informationen zu den Phasen, Schritten und Syntax für Dokumente mit AWSTOE YAML-Anwendungskomponenten finden Sie unter Dokumente verwenden in AWSTOE.

> **Note**
>
> Image Builder bestimmt die Komponententypen im Pipeline-Workflow. Dieser Workflow entspricht der Build-Phase und der Testphase im Build-Prozess. Image Builder bestimmt den Komponententyp wie folgt:
>
> - Build — Dies ist der Standardkomponententyp. Alles, was nicht als Testkomponente klassifiziert ist, ist eine Build-Komponente. Diese Art von Komponente wird während der Buildphase ausgeführt. Wenn für diese Build-Komponente eine `test` Phase definiert ist, wird diese Phase während der Testphase ausgeführt.
> - Test — Um als Testkomponente zu gelten, darf das Komponentendokument nur eine Phase mit dem Namen enthalten`test`. Für Tests, die sich auf Build-Komponentenkonfigurationen beziehen, empfehlen wir, keine eigenständige Testkomponente zu verwenden. Verwenden Sie stattdessen die `test` Phase in der zugehörigen Build-Komponente.
>
> Weitere Informationen darüber, wie Image Builder Phasen und Phasen verwendet, um den Komponenten-Workflow im Build-Prozess zu verwalten, finden Sie unterKomponenten mit Image Builder verwalten.

Um ein Dokument mit einer YAML-Anwendungskomponente für eine Beispielanwendung zu erstellen, folgen Sie den Schritten auf der Registerkarte, die Ihrem Image-Betriebssystem entspricht.

Linux

Erstellen Sie eine YAML-Komponentendatei

Verwenden Sie ein Dateibearbeitungstool, um eine Datei mit dem Namen *`update-linux-os.yaml`* zu erstellen. Fügen Sie den folgenden Inhalt hinzu:

```
# Copyright 2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: MIT-0
#
# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
# without restriction, including without limitation the rights to use, copy, modify,
```

```
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
# permit persons to whom the Software is furnished to do so.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
name: update-linux-os
description: Updates Linux with the latest security updates.
schemaVersion: 1
phases:
  - name: build
    steps:
    - name: UpdateOS
      action: UpdateOS
# Document End
```

> **Tip**
>
> Verwenden Sie ein Tool wie diesen Online-YAML-Validator oder eine YAML-Lint-Erweiterung in Ihrer Codeumgebung, um zu überprüfen, ob Ihr YAML wohlgeformt ist.

Windows

Erstellen Sie eine YAML-Komponentendatei

Verwenden Sie ein Dateibearbeitungstool, um eine Datei mit dem Namen *update-windows-os.yaml* zu erstellen. Fügen Sie den folgenden Inhalt hinzu:

```
# Copyright 2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: MIT-0
#
# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
# without restriction, including without limitation the rights to use, copy, modify,
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
```

```
# permit persons to whom the Software is furnished to do so.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
name: update-windows-os
description: Updates Windows with the latest security updates.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: UpdateOS
        action: UpdateOS
# Document End
```

> **Tip**
>
> Verwenden Sie ein Tool wie diesen Online-YAML-Validator oder eine YAML-Lint-Erweiterung in Ihrer Codeumgebung, um zu überprüfen, ob Ihr YAML wohlgeformt ist.

AWSTOE Komponentenparameter mit EC2 Image Builder verwalten

Sie können AWSTOE Komponenten, einschließlich der Erstellung und Einstellung von Komponentenparametern, direkt über die EC2 Image Builder Builder-Konsole oder mithilfe von AWS CLI Befehlen oder einem der Image Builder-SDKs verwalten. In diesem Abschnitt behandeln wir das Erstellen und Verwenden von Parametern in Ihrer Komponente und das Einstellen von Komponentenparametern über die Image Builder Builder-Konsole und AWS CLI Befehle.

> **Important**
>
> Bei Komponentenparametern handelt es sich um reine Textwerte, die angemeldet sind AWS CloudTrail. Wir empfehlen, dass Sie AWS Secrets Manager oder den AWS Systems Manager Parameter Store verwenden, um Ihre Geheimnisse zu speichern. Weitere Informationen zu Secrets Manager finden Sie unter Was ist Secrets Manager? im AWS Secrets Manager Benutzerhandbuch. Weitere Informationen zum AWS Systems Manager Parameterspeicher

finden Sie unter AWS Systems Manager Parameterspeicher im AWS Systems Manager Benutzerhandbuch.

Verwenden Sie Parameter in Ihrem YAML-Komponentendokument

Um eine Komponente zu erstellen, stellen Sie ein Dokument mit einer YAML-Anwendungskomponente bereit. Dies stellt die Phasen und Schritte dar, die Sie zum Erstellen der Komponente benötigen. Das Rezept, das auf die Komponente verweist, kann die Parameter so einstellen, dass die Werte zur Laufzeit angepasst werden. Standardwerte werden wirksam, wenn der Parameter nicht auf einen bestimmten Wert festgelegt ist.

Erstellen Sie ein Komponentendokument mit Eingabeparametern

In diesem Abschnitt erfahren Sie, wie Sie Eingabeparameter in Ihrem YAML-Komponentendokument definieren und verwenden.

Um ein YAML-Anwendungskomponentendokument zu erstellen, das Parameter verwendet und Befehle in Ihren Image Builder Builder-Build- oder Testinstanzen ausführt, folgen Sie den Schritten, die Ihrem Image-Betriebssystem entsprechen:

Linux

Erstellen Sie ein Dokument mit einer YAML-Komponente

Verwenden Sie ein Dateibearbeitungstool, um eine Datei mit dem Namen *hello-world-test.yaml* zu erstellen. Fügen Sie den folgenden Inhalt hinzu:

```
# Document Start
#
name: "HelloWorldTestingDocument-Linux"
description: "Hello world document to demonstrate parameters."
schemaVersion: 1.0
parameters:
  - MyInputParameter:
      type: string
      default: "It's me!"
      description: This is an input parameter.
phases:
  - name: build
    steps:
      - name: HelloWorldStep
```

```
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Build phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Validate phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Test phase. My input parameter value is
 {{ MyInputParameter }}"
# Document End
```

> **Tip**
>
> Verwenden Sie ein Tool wie diesen Online-YAML-Validator oder eine YAML-Lint-Erweiterung in Ihrer Codeumgebung, um zu überprüfen, ob Ihr YAML wohlgeformt ist.

Windows

Erstellen Sie ein Dokument mit einer YAML-Komponente

Verwenden Sie ein Dateibearbeitungstool, um eine Datei mit dem Namen *`hello-world-test.yaml`* zu erstellen. Fügen Sie den folgenden Inhalt hinzu:

```
# Document Start
#
name: "HelloWorldTestingDocument-Windows"
description: "Hello world document to demonstrate parameters."
```

```
schemaVersion: 1.0
parameters:
  - MyInputParameter:
      type: string
      default: "It's me!"
      description: This is an input parameter.
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host "Hello World! Build phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host "Hello World! Validate phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host "Hello World! Test phase. My input parameter value is
 {{ MyInputParameter }}"
# Document End
```

Tip

Verwenden Sie ein Tool wie diesen Online-YAML-Validator oder eine YAML-Lint-Erweiterung in Ihrer Codeumgebung, um zu überprüfen, ob Ihr YAML wohlgeformt ist.

Weitere Informationen zu den Phasen, Schritten und der Syntax von Dokumenten mit AWSTOE YAML-Anwendungskomponenten finden Sie unter Dokumente verwenden in. AWSTOE Weitere Informationen zu Parametern und ihren Anforderungen finden Sie auf Parameter der AWSTOE Seite Variablen definieren und referenzieren.

Erstellen Sie eine Komponente aus dem YAML-Komponentendokument

Unabhängig davon, mit welcher Methode Sie eine AWSTOE Komponente erstellen, ist das Dokument mit der YAML-Anwendungskomponente immer als Grundlage erforderlich.

- Informationen dazu, wie Sie mit der Image Builder Builder-Konsole eine Komponente direkt aus Ihrem YAML-Dokument erstellen können, finden Sie unterErstellen Sie eine Komponente mit der Image Builder Builder-Konsole.
- Informationen zur Verwendung von Image Builder Builder-Befehlen in der AWS CLI , um Ihre Komponente zu erstellen, finden Sie unterErstellen Sie AWSTOE Komponenten mit Image Builder mit dem AWS CLI. Ersetzen Sie in diesen Beispielen den Namen des YAML-Dokuments durch den Namen Ihres Hello World YAML-Dokuments ()*`hello-world-test.yaml`*.

Einstellen von Komponentenparametern in einem Image Builder Builder-Rezept (Konsole)

Das Einstellen von Komponentenparametern funktioniert für Image-Rezepte und Container-Rezepte auf dieselbe Weise. Wenn Sie ein neues Rezept oder eine neue Version eines Rezepts erstellen, wählen Sie aus den Listen Komponenten erstellen und Testkomponenten aus, welche Komponenten aufgenommen werden sollen. Die Komponentenlisten enthalten Komponenten, die für das Basisbetriebssystem gelten, das Sie für Ihr Image ausgewählt haben.

Nachdem Sie eine Komponente ausgewählt haben, wird sie im Abschnitt Ausgewählte Komponenten direkt unter den Komponentenlisten angezeigt. Die Konfigurationsoptionen werden für jede ausgewählte Komponente angezeigt. Wenn für Ihre Komponente Eingabeparameter definiert sind, werden diese als erweiterbarer Abschnitt mit der Bezeichnung Eingabeparameter angezeigt.

Die folgenden Parametereinstellungen werden für jeden Parameter angezeigt, der für Ihre Komponente definiert ist:

- Parametername (nicht editierbar) — Der Name des Parameters.
- Beschreibung (nicht editierbar) — Die Beschreibung des Parameters
- Typ (nicht editierbar) — Der Datentyp für den Parameterwert.

- Wert — Der Wert für den Parameter. Wenn Sie diese Komponente zum ersten Mal in diesem Rezept verwenden und ein Standardwert für den Eingabeparameter definiert wurde, wird der Standardwert im Feld Wert mit ausgegrautem Text angezeigt. Wenn kein anderer Wert eingegeben wird, verwendet Image Builder den Standardwert.

Komponentendetails auflisten und anzeigen

In diesem Abschnitt wird beschrieben, wie Sie Informationen zu den AWS Task Orchestrator and Executor (AWSTOE) -Komponenten finden und Details anzeigen können, die Sie in Ihren EC2 Image Builder Builder-Rezepten verwenden.

Einzelheiten zu den Komponenten

- AWSTOE Komponenten auflisten
- Listet die Build-Versionen der Komponenten auf (AWS CLI)
- Komponentendetails abrufen ()AWS CLI
- Details zur Komponentenrichtlinie abrufen (AWS CLI)

AWSTOE Komponenten auflisten

Sie können eine der folgenden Methoden verwenden, um AWSTOE Komponenten aufzulisten und zu filtern.

AWS Management Console

Gehen Sie folgendermaßen vor AWS Management Console, um eine Liste der Komponenten in der anzuzeigen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Komponenten aus. Standardmäßig zeigt Image Builder eine Liste der Komponenten an, die Ihrem Konto gehören.
3. Sie können optional nach dem Besitz der Komponenten filtern. Um Komponenten zu sehen, die Sie nicht besitzen, auf die Sie aber Zugriff haben, erweitern Sie die Dropdownliste Besitzertyp und wählen Sie einen der Werte aus. Die Besitzertypliste befindet sich in der Suchleiste neben dem Suchtextfeld. Sie können aus den folgenden Werten auswählen:

- Schnellstart (von Amazon verwaltet) — Öffentlich verfügbare Komponenten, die Amazon erstellt und verwaltet.
- Gehört mir — Komponenten, die Sie erstellt haben. Dies ist die Standardauswahl.
- Mit mir geteilt — Komponenten, die andere über ihr Konto erstellt und mit Ihnen geteilt haben.
- Von Dritten verwaltet — Komponenten, die einem Dritten gehören und die Sie abonniert AWS Marketplace haben.

AWS CLI

Das folgende Beispiel zeigt, wie Sie den list-components Befehl verwenden, um eine Liste der AWSTOE Komponenten zurückzugeben, die Ihrem Konto gehören.

```
aws imagebuilder list-components
```

Sie können optional nach dem Besitz der Komponenten filtern. Das Eigentümerattribut definiert, wem die Komponenten gehören, die Sie auflisten möchten. Standardmäßig gibt diese Anfrage eine Liste der Komponenten zurück, die Ihrem Konto gehören. Um die Ergebnisse nach dem Eigentümer der Komponente zu filtern, geben Sie bei der Ausführung des list-components Befehls einen der folgenden Werte mit dem `--owner` Parameter an.

Werte für den Eigentümer der Komponente

- Selbst
- Amazon
- ThirdParty
- Freigegeben

Die folgenden Beispiele zeigen den list-components Befehl mit dem `--owner` Parameter zum Filtern von Ergebnissen.

```
aws imagebuilder list-components --owner Self
{
    "requestId": "012a3456-b789-01cd-e234-fa5678b9012b",
    "componentVersionList": [
        {
```

```
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/sample-
component01/1.0.0",
            "name": "sample-component01",
            "version": "1.0.0",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2020-09-24T16:58:24.444Z"
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/sample-
component01/1.0.1",
            "name": "sample-component01",
            "version": "1.0.1",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2021-07-10T03:38:46.091Z"
        }
    ]
}
```

```
aws imagebuilder list-components --owner Amazon
```

```
aws imagebuilder list-components --owner Shared
```

```
aws imagebuilder list-components --owner ThirdParty
```

Listet die Build-Versionen der Komponenten auf (AWS CLI)

Das folgende Beispiel zeigt, wie der list-component-build-versions Befehl verwendet wird, um Komponenten-Build-Versionen aufzulisten, die eine bestimmte semantische Version haben. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung

```
aws imagebuilder list-component-build-versions --component-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:component/example-component/1.0.1
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "componentSummaryList": [
```

```
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
examplecomponent/1.0.1/1",
            "name": "examplecomponent",
            "version": "1.0.1",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "description": "An example component that builds, validates and tests an
 image",
            "changeDescription": "Updated version.",
            "dateCreated": "2020-02-19T18:53:45.940Z",
            "tags": {
                "KeyName": "KeyValue"
            }
        }
    ]
}
```

Komponentendetails abrufen ()AWS CLI

Das folgende Beispiel zeigt, wie Sie den get-component Befehl verwenden, um Komponentendetails abzurufen, wenn Sie den Amazon-Ressourcennamen (ARN) der Komponente angeben.

```
aws imagebuilder get-component --component-build-version-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/example-component/1.0.1/1
   {
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11112",
    "component": {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
examplecomponent/1.0.1/1",
        "name": "examplecomponent",
        "version": "1.0.1",
        "type": "BUILD",
        "platform": "Linux",
        "owner": "123456789012",
        "data": "name: HelloWorldTestingDocument\ndescription: This is hello world
 testing document... etc.\"\n",
        "encrypted": true,
        "dateCreated": "2020-09-24T16:58:24.444Z",
        "tags": {}
    }
```

```
}
```

Details zur Komponentenrichtlinie abrufen (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den get-component-policy Befehl verwenden, um Details zu einer Komponentenrichtlinie abzurufen, wenn Sie den ARN der Komponente angeben.

```
aws imagebuilder get-component-policy --component-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/example-component/1.0.1
```

Erstellen Sie eine Komponente mit der Image Builder Builder-Konsole

Gehen Sie folgendermaßen vor, um eine AWSTOE Anwendungskomponente von der Image Builder Builder-Konsole aus zu erstellen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Komponenten aus. Wählen Sie dann Komponente erstellen aus.
3. Geben Sie auf der Seite Komponente erstellen unter Komponentendetails Folgendes ein:

   a. Image-Betriebssystem (OS). Geben Sie das Betriebssystem an, mit dem die Komponente kompatibel ist.

   b. Komponentenkategorie. Wählen Sie aus der Dropdownliste den Typ der Build- oder Testkomponente aus, die Sie erstellen.

   c. Name der Komponente. Geben Sie einen Namen für die Komponente ein.

   d. Version der Komponente. Geben Sie die Versionsnummer der Komponente ein.

   e. Beschreibung. Geben Sie optional eine Beschreibung an, die Ihnen bei der Identifizierung der Komponente hilft.

   f. Beschreibung ändern. Geben Sie optional eine Beschreibung an, damit Sie die Änderungen verstehen, die an dieser Version der Komponente vorgenommen wurden.
4. Im Abschnitt Definitionsdokument lautet die Standardoption Dokumentinhalt definieren. Das Komponentendokument definiert die Aktionen, die Image Builder auf den Build- und Testinstanzen ausführt, um Ihr Image zu erstellen.

   Geben Sie im Feld Inhalt den Inhalt Ihres YAML-Komponentendokuments ein. Um mit einem Hello World-Beispiel für Linux zu beginnen, wählen Sie die Option Beispiel verwenden. Weitere

Informationen zum Erstellen eines YAML-Komponentendokuments oder zum Kopieren und Einfügen des UpdateOS-Beispiels von dieser Seite finden Sie unter. Erstellen Sie ein YAML-Komponentendokument

5. Nachdem Sie die Komponentendetails eingegeben haben, wählen Sie Komponente erstellen aus.

> **Note**
>
> Wenn Sie Ihre neue Komponente sehen möchten, wenn Sie ein Rezept erstellen oder aktualisieren, wenden Sie den Filter „Mein Eigentum“ auf die Liste der Build- oder Testkomponenten an. Der Filter befindet sich oben in der Komponentenliste neben dem Suchfeld.

6. Um eine Komponente zu löschen, aktivieren Sie auf der Seite Komponenten das Kontrollkästchen neben der Komponente, die Sie löschen möchten. Wählen Sie in der Dropdownliste Aktionen die Option Komponente löschen aus.

Gehen Sie folgendermaßen vor, um eine neue Komponentenversion zu erstellen:

1. Je nachdem, wo Sie beginnen:

   - Auf der Seite mit der Komponentenliste — Aktivieren Sie das Kontrollkästchen neben dem Komponentennamen und wählen Sie dann im Menü Aktionen die Option Neue Version erstellen aus.
   - Auf der Detailseite der Komponente — Wählen Sie in der oberen rechten Ecke der Überschrift die Schaltfläche Neue Version erstellen.

2. Die Komponenteninformationen sind bereits mit den aktuellen Werten gefüllt, wenn die Seite „Komponente erstellen“ angezeigt wird. Folgen Sie den Schritten zum Erstellen einer Komponente, um die Komponente zu aktualisieren. Dadurch wird sichergestellt, dass Sie eine eindeutige semantische Version in die Komponentenversion eingeben. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung

Erstellen Sie eine Komponente mit dem AWS CLI

In diesem Abschnitt wird beschrieben, wie Sie mit Image Builder Builder-Befehlen AWS Task Orchestrator and Executor (AWSTOE) Komponenten aus dem erstellen AWS Command Line Interface. Um eine Komponente zu erstellen, stellen Sie ein Dokument mit einer YAML-Anwendungskomponente bereit. Dies stellt die Phasen und Schritte dar, die Sie zum Erstellen der Komponente benötigen. Informationen zum Erstellen eines neuen YAML-Komponentendokuments finden Sie unterErstellen Sie ein YAML-Komponentendokument.

Erstellen Sie AWSTOE Komponenten mit Image Builder mit dem AWS CLI

In diesem Abschnitt erfahren Sie, wie Sie Image Builder Builder-Befehle in der Komponente AWS CLI So erstellen Sie eine AWSTOE Anwendung wie folgt einrichten und verwenden.

- Laden Sie Ihr YAML-Komponentendokument in einen S3-Bucket hoch, auf den Sie von der Befehlszeile aus verweisen können.
- Erstellen Sie die AWSTOE Anwendungskomponente mit dem create-component Befehl.
- Listet die Komponentenversionen mit dem list-components Befehl und einem Namensfilter auf, um zu sehen, welche Versionen bereits existieren. Anhand der Ausgabe können Sie bestimmen, welche Version als nächste Version für Updates verwendet werden soll.

Um eine AWSTOE Anwendungskomponente aus einem YAML-Eingabedokument zu erstellen, folgen Sie den Schritten, die Ihrer Image-Betriebssystemplattform entsprechen.

Linux

Speichern Sie Ihr Anwendungskomponentendokument in Amazon S3

Sie können einen S3-Bucket als Repository für das Quelldokument Ihrer AWSTOE Anwendungskomponente verwenden. Gehen Sie folgendermaßen vor, um Ihr Komponentendokument zu speichern:

- Laden Sie das Dokument auf Amazon S3 hoch

  Wenn Ihr Dokument kleiner als 64 KB ist, können Sie diesen Schritt überspringen. Dokumente mit einer Größe von 64 KB oder mehr müssen in Amazon S3 gespeichert werden.

```
aws s3 cp update-linux-os.yaml s3://my-s3-bucket/my-path/update-linux-os.yaml
```

Erstellen Sie eine Komponente aus dem YAML-Dokument

Um den create-component Befehl, den Sie in der verwenden, zu optimieren AWS CLI, erstellen Sie eine JSON-Datei, die alle Komponentenparameter enthält, die Sie an den Befehl übergeben möchten. Geben Sie den Speicherort des *update-linux-os.yaml* Dokuments an, das Sie in den vorherigen Schritten erstellt haben. Das `uri` Schlüssel-Wert-Paar enthält die Dateireferenz.

> **Note**
>
> Die Benennungskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Aktionsanforderungsparameter der Image Builder Builder-API angegeben ist. Informationen zu den API-Befehlsanforderungsparametern finden Sie unter dem CreateComponentBefehl in der EC2 Image Builder API-Referenz.
> Informationen zur Bereitstellung der Datenwerte als Befehlszeilenparameter finden Sie in den Parameternamen, die in der AWS CLI Befehlsreferenz angegeben sind.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie ein Dateibearbeitungstool, um eine Datei mit dem Namen zu erstellen*create-update-linux-os-component.json*. Fügen Sie den folgenden Inhalt hinzu:

   ```
   {
    "name": "update-linux-os",
    "semanticVersion": "1.1.2",
    "description": "An example component that updates the Linux operating system",
    "changeDescription": "Initial version.",
    "platform": "Linux",
    "uri": "s3://my-s3-bucket/my-path/update-linux-os.yaml",
    "kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/98765432-
   b123-456b-7f89-0123456f789c",
    "tags": {
     "MyTagKey-purpose": "security-updates"
    }
   }
   ```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

2. Erstellen Sie die Komponente

   Verwenden Sie den folgenden Befehl, um die Komponente zu erstellen, und verweisen Sie dabei auf den Dateinamen der JSON-Datei, die Sie im vorherigen Schritt erstellt haben:

```
aws imagebuilder create-component --cli-input-json file://create-update-linux-os-component.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Windows

Speichern Sie Ihr Anwendungskomponentendokument in Amazon S3

Sie können einen S3-Bucket als Repository für das Quelldokument Ihrer AWSTOE Anwendungskomponente verwenden. Gehen Sie folgendermaßen vor, um Ihr Komponentendokument zu speichern:

- Laden Sie das Dokument auf Amazon S3 hoch

  Wenn Ihr Dokument kleiner als 64 KB ist, können Sie diesen Schritt überspringen. Dokumente mit einer Größe von 64 KB oder mehr müssen in Amazon S3 gespeichert werden.

  ```
  aws s3 cp update-windows-os.yaml s3://my-s3-bucket/my-path/update-windows-os.yaml
  ```

Erstellen Sie eine Komponente aus dem YAML-Dokument

Um den create-component Befehl, den Sie in der verwenden, zu optimieren AWS CLI, erstellen Sie eine JSON-Datei, die alle Komponentenparameter enthält, die Sie an den Befehl übergeben möchten. Geben Sie den Speicherort des *update-windows-os.yaml* Dokuments an, das Sie in den vorherigen Schritten erstellt haben. Das `uri` Schlüssel-Wert-Paar enthält die Dateireferenz.

> **Note**
>
> Die Benennungskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Aktionsanforderungsparameter der Image Builder Builder-API angegeben ist. Informationen zu den API-Befehlsanforderungsparametern finden Sie unter dem CreateComponentBefehl in der EC2 Image Builder API-Referenz.
> Informationen zur Bereitstellung der Datenwerte als Befehlszeilenparameter finden Sie in den Parameternamen, die in der AWS CLI Befehlsreferenz angegeben sind.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie ein Dateibearbeitungstool, um eine Datei mit dem Namen zu erstellen*create-update-windows-os-component.json*. Fügen Sie den folgenden Inhalt hinzu:

   ```
   {
    "name": "update-windows-os",
    "semanticVersion": "1.1.2",
    "description": "An example component that updates the Windows operating system.",
    "changeDescription": "Initial version.",
    "platform": "Windows",
   ```

```
 "uri": "s3://my-s3-bucket/my-path/update-windows-os.yaml",
 "kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/98765432-
b123-456b-7f89-0123456f789c",
 "tags": {
  "MyTagKey-purpose": "security-updates"
 }
}
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

2. Erstellen Sie die Komponente

   Verwenden Sie den folgenden Befehl, um die Komponente zu erstellen, und verweisen Sie dabei auf den Dateinamen der JSON-Datei, die Sie im vorherigen Schritt erstellt haben:

```
aws imagebuilder create-component --cli-input-json file://create-update-windows-
os-component.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

AWSTOE Versionierung von Komponenten für Updates ()AWS CLI

AWSTOE Komponentennamen und Versionen werden in den Amazon-Ressourcennamen (ARN) der Komponente nach dem Komponentenpräfix eingebettet. Jede neue Version einer Komponente

hat ihren eigenen eindeutigen ARN. Die Schritte zum Erstellen einer neuen Version sind genau dieselben wie zum Erstellen einer neuen Komponente, solange die semantische Version für diesen Komponentennamen eindeutig ist. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung

Um sicherzustellen, dass Sie die nächste logische Version zuweisen, rufen Sie zunächst eine Liste der vorhandenen Versionen für die Komponente ab, die Sie ändern möchten. Verwenden Sie den list-components Befehl mit dem AWS CLI und filtern Sie nach dem Namen.

In diesem Beispiel filtern Sie nach dem Namen der Komponente, die Sie in den vorherigen Linux-Beispielen erstellt haben. Um die Komponente aufzulisten, die Sie erstellt haben, verwenden Sie den Wert des `name` Parameters aus der JSON-Datei, die Sie im create-component Befehl verwendet haben.

```
aws imagebuilder list-components --filters name="name",values="update-linux-os"
{
    "requestId": "123a4567-b890-123c-45d6-ef789ab0cd1e",
    "componentVersionList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:1234560087789012:component/update-
linux-os/1.0.0",
            "name": "update-linux-os",
            "version": "1.0.0",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2020-09-24T16:58:24.444Z"
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:1234560087789012:component/update-
linux-os/1.0.1",
            "name": "update-linux-os",
            "version": "1.0.1",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2021-07-10T03:38:46.091Z"
        }
    ]
}
```

Anhand Ihrer Ergebnisse können Sie bestimmen, welche Version die nächste Version sein soll.

Eine Komponente importieren (AWS CLI)

In einigen Szenarien ist es möglicherweise einfacher, mit einem bereits vorhandenen Skript zu beginnen. Für dieses Szenario können Sie das folgende Beispiel verwenden.

In diesem Beispiel wird davon ausgegangen, dass Sie eine Datei mit dem Namen *import-component.json* (wie abgebildet) haben. Beachten Sie, dass die Datei direkt auf ein PowerShell aufgerufenes Skript verweist, in `AdminConfig.ps1` das bereits hochgeladen wurde*my-s3-bucket*. Wird derzeit für die Komponente unterstütztformat. `SHELL`

```
{
"name": "MyImportedComponent",
"semanticVersion": "1.0.0",
"description": "An example of how to import a component",
"changeDescription": "First commit message.",
"format": "SHELL",
"platform": "Windows",
"type": "BUILD",
"uri": "s3://my-s3-bucket/AdminConfig.ps1",
"kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/60763706-
b131-418b-8f85-3420912f020c"
}
```

Führen Sie den folgenden Befehl aus, um die Komponente zu importieren.

```
aws imagebuilder import-component --cli-input-json file://import-component.json
```

Bereinigen von -Ressourcen

Um unerwartete Gebühren zu vermeiden, sollten Sie die Ressourcen und Pipelines, die Sie anhand der Beispiele in diesem Handbuch erstellt haben, bereinigen. Weitere Informationen zum Löschen von Ressourcen in Image Builder finden Sie unterEC2 Image Builder Builder-Ressourcen löschen.

Rezepte verwalten

Ein EC2 Image Builder Builder-Rezept definiert das Basis-Image, das als Ausgangspunkt für die Erstellung eines neuen Images verwendet werden soll, zusammen mit den Komponenten, die Sie hinzufügen, um Ihr Image anzupassen und zu überprüfen, ob alles wie erwartet funktioniert.

Image Builder bietet automatische Versionsauswahlen für jede Komponente. Standardmäßig können Sie bis zu 20 Komponenten auf ein Rezept anwenden. Dies umfasst sowohl Build- als auch Testkomponenten.

Nachdem Sie ein Rezept erstellt haben, können Sie es weder ändern noch ersetzen. Um Komponenten zu aktualisieren, nachdem Sie ein Rezept erstellt haben, müssen Sie ein neues Rezept oder eine neue Rezeptversion erstellen. Sie können jederzeit Tags auf Ihre vorhandenen Rezepte anwenden. Weitere Informationen zum Taggen Ihrer Ressourcen mithilfe von Image Builder Builder-Befehlen finden Sie im AWS CLIMarkieren von Ressourcen Abschnitt dieses Handbuchs.

> **Tip**
>
> Sie können von Amazon verwaltete Komponenten in Ihren Rezepten verwenden oder mit der Anwendung AWS Task Orchestrator and Executor (AWSTOE) Ihre eigenen benutzerdefinierten Komponenten entwickeln. Um zu beginnen, sehen Sie sich Fangen Sie an mit AWSTOE an.

In diesem Abschnitt erfahren Sie, wie Sie Rezepte auflisten, ansehen und erstellen.

Inhalt

Rezeptdetails auf Bildern auflisten und anzeigen

In diesem Abschnitt werden die verschiedenen Möglichkeiten beschrieben, wie Sie Informationen finden und Details zu Ihren EC2 Image Builder Builder-Image-Rezepten anzeigen können.

Einzelheiten zu den Bildrezepten

- Rezeptdetails im Bild abrufen (AWS CLI)
- Details zur Bildrezeptrichtlinie abrufen (AWS CLI)

Bildrezepte auflisten (Konsole)

Gehen Sie wie folgt vor, um eine Liste der unter Ihrem Konto erstellten Image-Rezepte in der Image Builder Builder-Konsole anzuzeigen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Bildrezepte aus. Daraufhin wird eine Liste der Bildrezepte angezeigt, die unter Ihrem Konto erstellt wurden.
3. Um Details anzuzeigen oder eine neue Rezeptversion zu erstellen, wählen Sie den Link Rezeptname. Dadurch wird die Detailansicht für das Rezept geöffnet.

> **Note**
>
> Sie können auch das Kontrollkästchen neben dem Rezeptnamen aktivieren und dann Details anzeigen auswählen.

Bildrezepte auflisten (AWS CLI)

Das folgende Beispiel zeigt, wie Sie alle Ihre Bildrezepte auflisten können, indem Sie den verwenden AWS CLI.

```
aws imagebuilder list-image-recipes
```

Rezeptdetails für Bilder anzeigen (Konsole)

Um Details für ein bestimmtes Image-Rezept mit der Image Builder Builder-Konsole anzuzeigen, wählen Sie das Bildrezept aus, das Sie überprüfen möchten. Gehen Sie dabei wie unter beschrieben vorBildrezepte auflisten (Konsole).

Auf der Seite mit den Rezeptdetails können Sie:

- Löschen Sie das Rezept. Weitere Informationen zum Löschen von Ressourcen in Image Builder finden Sie unterEC2 Image Builder Builder-Ressourcen löschen.

- Erstellen Sie eine neue Version.
- Erstellen Sie eine Pipeline aus dem Rezept. Nachdem Sie „Pipeline aus diesem Rezept erstellen“ ausgewählt haben, werden Sie zum Pipeline-Assistenten weitergeleitet. Weitere Informationen zum Erstellen einer Image Builder Builder-Pipeline mithilfe des Pipeline-Assistenten finden Sie unter Erstellen Sie eine Image-Pipeline mit dem EC2 Image Builder Builder-Konsolenassistenten

> **Note**
>
> Wenn Sie eine Pipeline aus einem vorhandenen Rezept erstellen, ist die Option zum Erstellen eines neuen Rezepts nicht verfügbar.

Rezeptdetails im Bild abrufen (AWS CLI)

Das folgende Beispiel zeigt, wie ein imagebuilder CLI-Befehl verwendet wird, um die Details eines Bildrezepts abzurufen, indem der Amazon-Ressourcenname (ARN) angegeben wird.

```
aws imagebuilder get-image-recipe --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-example-recipe/2020.12.03
```

Details zur Bildrezeptrichtlinie abrufen (AWS CLI)

Das folgende Beispiel zeigt, wie Sie mit einem imagebuilder CLI-Befehl die Details einer Bildrezeptrichtlinie abrufen, indem Sie deren ARN angeben.

```
aws imagebuilder get-image-recipe-policy --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-example-recipe/2020.12.03
```

Rezeptdetails für Container auflisten und anzeigen

In diesem Abschnitt wird beschrieben, wie Sie Informationen finden und Details zu Ihren EC2 Image Builder Builder-Container-Rezepten anzeigen können.

Einzelheiten zum Container-Rezept

- Listet Container-Rezepte in der Konsole auf
- Listet Container-Rezepte mit dem auf AWS CLI
- Rezeptdetails für Container in der Konsole anzeigen

- Rezeptdetails für Container erhalten Sie mit dem AWS CLI
- Details zur Container-Rezept-Richtlinie erhalten Sie mit dem AWS CLI

Listet Container-Rezepte in der Konsole auf

Gehen Sie wie folgt vor, um eine Liste der Container-Rezepte anzuzeigen, die unter Ihrem Konto in der Image Builder Builder-Konsole erstellt wurden:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Container-Rezepte aus. Daraufhin wird eine Liste der Container-Rezepte angezeigt, die unter Ihrem Konto erstellt wurden.
3. Um Details anzuzeigen oder eine neue Rezeptversion zu erstellen, wählen Sie den Link Rezeptname. Dadurch wird die Detailansicht für das Rezept geöffnet.

> **Note**
>
> Sie können auch das Kontrollkästchen neben dem Rezeptnamen aktivieren und dann Details anzeigen auswählen.

Listet Container-Rezepte mit dem auf AWS CLI

Das folgende Beispiel zeigt, wie Sie alle Ihre Container-Rezepte auflisten, indem Sie den verwenden AWS CLI.

```
aws imagebuilder list-container-recipes
```

Rezeptdetails für Container in der Konsole anzeigen

Um Details für ein bestimmtes Container-Rezept mit der Image Builder Builder-Konsole anzuzeigen, wählen Sie das zu überprüfende Container-Rezept aus und führen Sie die unter beschriebenen Schritte ausListet Container-Rezepte in der Konsole auf.

Auf der Seite mit den Rezeptdetails können Sie Folgendes tun:

- Löschen Sie das Rezept. Weitere Informationen zum Löschen von Ressourcen in Image Builder finden Sie unterEC2 Image Builder Builder-Ressourcen löschen.

- Erstellen Sie eine neue Version.
- Erstellen Sie eine Pipeline aus dem Rezept. Nachdem Sie „Pipeline aus diesem Rezept erstellen“ ausgewählt haben, werden Sie zum Pipeline-Assistenten weitergeleitet. Weitere Informationen zum Erstellen einer Image Builder Builder-Pipeline mithilfe des Pipeline-Assistenten finden Sie unter Erstellen Sie eine Image-Pipeline mit dem EC2 Image Builder Builder-Konsolenassistenten

> **Note**
>
> Wenn Sie eine Pipeline aus einem vorhandenen Rezept erstellen, ist die Option zum Erstellen eines neuen Rezepts nicht verfügbar.

Rezeptdetails für Container erhalten Sie mit dem AWS CLI

Das folgende Beispiel zeigt, wie Sie mit einem imagebuilder CLI-Befehl die Details eines Container-Rezepts abrufen, indem Sie dessen ARN angeben.

```
aws imagebuilder get-container-recipe --container-recipe-arn arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-example-recipe/2020.12.03
```

Details zur Container-Rezept-Richtlinie erhalten Sie mit dem AWS CLI

Das folgende Beispiel zeigt, wie Sie mit einem imagebuilder CLI-Befehl die Details einer Container-Rezeptrichtlinie abrufen, indem Sie deren ARN angeben.

```
aws imagebuilder get-container-recipe-policy --container-recipe-arn
 arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-example-recipe/2020.12.03
```

Erstellen Sie eine neue Version eines Bildrezepts

In diesem Abschnitt wird beschrieben, wie Sie eine neue Version eines Image-Rezepts erstellen.

Inhalt

Erstellen Sie eine neue Image-Rezeptversion (Konsole)

Wenn Sie eine neue Rezeptversion erstellen, ist das praktisch dasselbe wie das Erstellen eines neuen Rezepts. Der Unterschied besteht darin, dass in den meisten Fällen bestimmte Details so ausgewählt sind, dass sie dem Basisrezept entsprechen. In der folgenden Liste werden die Unterschiede zwischen dem Erstellen eines neuen Rezepts und dem Erstellen einer neuen Version eines vorhandenen Rezepts beschrieben.

Einzelheiten zum Basisrezept in der neuen Version

- Name — Nicht editierbar.
- Version — Erforderlich. Dieses Basisdetail ist nicht mit der aktuellen Version oder einer Sequenz vorausgefüllt. <major>Geben Sie die Versionsnummer ein, die Sie in dem Format erstellen möchten. <minor>. <patch>. Wenn die Version bereits existiert, tritt ein Fehler auf.
- Die Option Bild auswählen — Vorausgewählt, aber Sie können sie bearbeiten. Wenn Sie Ihre Wahl für die Quelle Ihres Basisimages ändern, gehen möglicherweise weitere Details verloren, die von der ursprünglich ausgewählten Option abhängen.

  Um Details zu sehen, die mit Ihrer Auswahl des Basisimages verknüpft sind, wählen Sie die Registerkarte, die Ihrer Auswahl entspricht.

  Managed image

  - Image-Betriebssystem (OS) — Nicht editierbar.
  - Bildname — Vorausgewählt, basierend auf der Kombination von Basis-Image-Optionen, die Sie für das bestehende Rezept getroffen haben. Wenn Sie jedoch die Option Bild auswählen ändern, verlieren Sie den Namen des vorausgewählten Images.
  - Optionen für die automatische Versionierung — Entspricht nicht Ihrem Grundrezept. Diese Image-Option ist standardmäßig auf die Option Ausgewählte Betriebssystemversion verwenden eingestellt.

  > **Important**
  >
  > Wenn Sie semantische Versionierung verwenden, um Pipeline-Builds zu starten, stellen Sie sicher, dass Sie diesen Wert auf Letzte verfügbare Betriebssystemversion verwenden ändern. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung

AWS Marketplace image

- Abonnements — Diese Registerkarte sollte geöffnet sein und das abonnierte Image von AWS Marketplace sollte so ausgewählt sein, dass es Ihrem Basisrezept entspricht. Wenn du das Bild änderst, das dein Rezept als Basisbild verwendet, gehen möglicherweise weitere Details verloren, die vom ausgewählten Originalbild abhängen.

Weitere Informationen zu AWS Marketplace Produkten finden Sie unter Produkte kaufen im AWS Marketplace Einkaufsführer.

Custom AMI

- AMI-ID — Erforderlich. Diese Einstellung ist jedoch nicht mit Ihrem ursprünglichen Eintrag vorausgefüllt. Sie müssen die AMI-ID für Ihr Basis-Image eingeben.

- Instanzkonfiguration — Die Einstellungen sind vorausgewählt, aber Sie können sie bearbeiten.
  - Systems Manager Manager-Agent — Sie können dieses Kontrollkästchen aktivieren oder deaktivieren, um die Installation des Systems Manager Manager-Agenten auf dem neuen Image zu steuern. Das Kontrollkästchen ist standardmäßig deaktiviert, um den Systems Manager Manager-Agenten in Ihr neues Image aufzunehmen. Um den Systems Manager Manager-Agenten aus dem endgültigen Image zu entfernen, aktivieren Sie das Kontrollkästchen, sodass der Agent nicht in Ihrem AMI enthalten ist.
  - Benutzerdaten — In diesem Bereich können Sie Befehle oder ein Befehlsskript angeben, das ausgeführt werden soll, wenn Sie Ihre Build-Instance starten. Dieser Wert ersetzt jedoch alle Befehle, die Image Builder möglicherweise hinzugefügt hat, um sicherzustellen, dass Systems Manager installiert ist. Zu diesen Befehlen gehört das Bereinigungsskript, das Image Builder normalerweise für Linux-Images ausführt, bevor das neue Image erstellt wird.

> **Note**
>
> - Wenn Sie Benutzerdaten eingeben, stellen Sie sicher, dass der Systems Manager Agent auf Ihrem Basis-Image vorinstalliert ist oder dass Sie die Installation in Ihren Benutzerdaten angeben.
> - Stellen Sie bei Linux-Images sicher, dass die Bereinigungsschritte ausgeführt werden, indem Sie einen Befehl hinzufügen, um eine leere Datei mit dem Namen `perform_cleanup` in Ihrem Benutzerdatenskript zu erstellen. Image Builder erkennt diese Datei und führt das Bereinigungsskript aus, bevor das neue Image erstellt wird. Weitere Informationen und ein Beispielskript finden Sie unterBewährte Sicherheitsmethoden für EC2 Image Builder.

- Arbeitsverzeichnis — Vorausgewählt, aber Sie können es bearbeiten.
- Komponenten — Komponenten, die bereits in der Rezeptur enthalten sind, werden im Abschnitt Ausgewählte Komponenten am Ende jeder Komponentenliste (Build und Test) angezeigt. Sie können die ausgewählten Komponenten je nach Bedarf entfernen oder neu anordnen.

  CIS-Härtungskomponenten folgen nicht den Standardregeln für die Reihenfolge der Komponenten in den Image Builder Builder-Rezepten. Die CIS-Härtungskomponenten werden immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests anhand Ihres Ausgabebilds ausgeführt werden.

  > **Note**
  >
  > In den Listen der Build- und Testkomponenten werden die verfügbaren Komponenten basierend auf dem Typ des Komponentenbesitzers angezeigt. Um Komponenten für Ihr Rezept hinzuzufügen oder zu aktualisieren, wählen Sie den Besitzertyp für die Komponente aus, nach der Sie suchen. Wenn Sie beispielsweise eine Komponente hinzufügen möchten, die mit einem Basis-Image verknüpft ist, das Sie abonniert haben AWS Marketplace, wählen Sie in der Liste mit dem Besitzertyp neben der Suchleiste einen Eintrag `Third party managed` aus.

  Sie können die folgenden Einstellungen für die ausgewählte Komponente konfigurieren:

  - Versionierungsoptionen — Vorausgewählt, aber Sie können sie ändern. Wir empfehlen Ihnen, die Option Letzte verfügbare Komponentenversion verwenden zu wählen, um sicherzustellen, dass Ihre Image-Builds immer die neueste Version der Komponente verwenden. Wenn Sie in Ihrem Rezept eine bestimmte Komponentenversion verwenden müssen, können Sie Komponentenversion angeben wählen und die Version in das angezeigte Feld Komponentenversion eingeben.
  - Eingabeparameter — Zeigt Eingabeparameter an, die die Komponente akzeptiert. Der Wert ist bereits mit dem Wert aus der vorherigen Version des Rezepts gefüllt. Wenn Sie diese Komponente zum ersten Mal in diesem Rezept verwenden und ein Standardwert für den Eingabeparameter definiert wurde, wird der Standardwert im Feld Wert mit ausgegrautem Text angezeigt. Wenn kein anderer Wert eingegeben wird, verwendet Image Builder den Standardwert.

    Wenn ein Eingabeparameter erforderlich ist, für den jedoch kein Standardwert in der Komponente definiert ist, müssen Sie einen Wert angeben. Image Builder erstellt die

Rezeptversion nicht, wenn erforderliche Parameter fehlen und für die kein Standardwert definiert ist.

> **Important**
>
> Bei den Komponentenparametern handelt es sich um reine Textwerte, die angemeldet sind AWS CloudTrail. Wir empfehlen, dass Sie AWS Secrets Manager oder den AWS Systems Manager Parameter Store verwenden, um Ihre Geheimnisse zu speichern. Weitere Informationen zu Secrets Manager finden Sie unter Was ist Secrets Manager? im AWS Secrets Manager Benutzerhandbuch. Weitere Informationen zum AWS Systems Manager Parameterspeicher finden Sie unter AWS Systems Manager Parameterspeicher im AWS Systems Manager Benutzerhandbuch.

Um die Einstellungen für Versionierungsoptionen oder Eingabeparameter zu erweitern, können Sie den Pfeil neben dem Namen der Einstellung wählen. Um alle Einstellungen für alle ausgewählten Komponenten zu erweitern, können Sie den Schalter Alle erweitern aus- und einschalten.

- Speicher (Volumen) — sind vorgefüllt. Das Root-Volume, der Gerätename, der Snapshot und die IOPS-Auswahl können nicht bearbeitet werden. Sie können jedoch alle übrigen Einstellungen ändern, z. B. die Größe. Sie können auch neue Volumes hinzufügen und neue oder bestehende Volumes verschlüsseln.

  Um Volumes für die Images zu verschlüsseln, die Image Builder unter Ihrem Konto in der Quellregion (in der der Build ausgeführt wird) erstellt, müssen Sie die Speichervolumenverschlüsselung im Image-Rezept verwenden. Die Verschlüsselung, die während der Verteilungsphase des Builds ausgeführt wird, gilt nur für Images, die an andere Konten oder Regionen verteilt werden.

  > **Note**
  >
  > Wenn Sie Verschlüsselung für Ihre Volumes verwenden, müssen Sie den Schlüssel für jedes Volume separat auswählen, auch wenn der Schlüssel derselbe ist, der für das Root-Volume verwendet wird.

So erstellen Sie eine neue Image-Rezeptversion:

1. Wählen Sie oben auf der Seite mit den Rezeptdetails die Option Neue Version erstellen aus. Dadurch gelangen Sie zur Seite „Bildrezept erstellen“.
2. Um die neue Version zu erstellen, nehmen Sie Ihre Änderungen vor und wählen Sie dann Image-Rezept erstellen.

Weitere Informationen zum Erstellen eines Image-Rezepts beim Erstellen einer Image-Pipeline finden Sie Schritt 2: Rezept wählen im Abschnitt Erste Schritte dieses Handbuchs.

Erstellen Sie ein Bildrezept mit dem AWS CLI

Gehen Sie wie folgt vor, um ein Image-Rezept mit dem `create-image-recipe` AWS CLI Befehl Image Builder im zu erstellen:

Voraussetzungen

Bevor Sie die Image Builder Builder-Befehle in diesem Abschnitt ausführen, um ein Image-Rezept aus dem zu erstellen AWS CLI, müssen Sie die Komponenten erstellen, die das Rezept verwendet. Das Beispiel für ein Bildrezept im folgenden Schritt bezieht sich auf Beispielkomponenten, die im Erstellen Sie eine Komponente mit dem AWS CLI Abschnitt dieses Handbuchs erstellt wurden.

Nachdem Sie Ihre Komponenten erstellt haben oder wenn Sie vorhandene Komponenten verwenden, notieren Sie sich die ARNs, die Sie in das Rezept aufnehmen möchten.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Sie können die gesamte Eingabe für den create-image-recipe Befehl mit Inline-Befehlsparametern bereitstellen. Der resultierende Befehl kann jedoch ziemlich lang sein. Um den Befehl zu vereinfachen, können Sie stattdessen eine JSON-Datei bereitstellen, die alle Rezepteinstellungen enthält.

   > **Note**
   >
   > Die Benennungskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Aktionsanforderungsparameter der Image Builder Builder-API angegeben ist. Informationen zu den API-Befehlsanforderungsparametern finden Sie unter dem CreateImageRecipeBefehl in der EC2 Image Builder API-Referenz.

Informationen zur Bereitstellung der Datenwerte als Befehlszeilenparameter finden Sie in den Parameternamen, die in der AWS CLI Befehlsreferenz angegeben sind.

Im Folgenden finden Sie eine Zusammenfassung der Parameter, die in diesen Beispielen angegeben werden:

- name (string, erforderlich) — Der Name des Bildrezepts.
- description (string) — Die Beschreibung des Bildrezepts.
- parentImage (string, erforderlich) — Das Bild, das das Bildrezept als Grundlage für Ihr benutzerdefiniertes Bild verwendet. Der Wert kann der ARN des Basis-Images oder eine AMI-ID sein.

  > **Note**
  >
  > Das Linux-Beispiel verwendet ein Image Builder Builder-AMI und das Windows-Beispiel verwendet einen ARN.

- SemanticVersion <major>(string, erforderlich) — Die semantische Version des Bildrezepts, ausgedrückt im folgenden Format, mit numerischen Werten an jeder Position, um eine bestimmte Version anzugeben:. <minor>. <patch>. Zum Beispiel könnte ein Wert sein`1.0.0`. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung
- components (Array, erforderlich) — Enthält ein Array von `ComponentConfiguration` Objekten. Es muss mindestens eine Build-Komponente angegeben werden:

  > **Note**
  >
  > Image Builder installiert Komponenten in der Reihenfolge, in der Sie sie im Rezept angegeben haben. CIS-Härtungskomponenten werden jedoch immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests anhand Ihres Ausgabe-Images ausgeführt werden.

  - componentArn (string, required) — Der Komponenten-ARN.

> **Tip**
>
> Um eines der Beispiele zu verwenden, um Ihr eigenes Bildrezept zu erstellen, müssen Sie die Beispiel-ARNs durch die ARNs für die Komponenten ersetzen, die Sie für Ihr Rezept verwenden.

- Parameter (Array von Objekten) — Enthält ein Array von `ComponentParameter` Objekten. Wenn ein Eingabeparameter erforderlich ist, aber in der Komponente kein Standardwert definiert ist, müssen Sie einen Wert angeben. Image Builder erstellt die Rezeptversion nicht, wenn erforderliche Parameter fehlen und für die kein Standardwert definiert ist.

  > **Important**
  >
  > Bei den Komponentenparametern handelt es sich um reine Textwerte, die angemeldet sind AWS CloudTrail. Wir empfehlen, dass Sie AWS Secrets Manager oder den AWS Systems Manager Parameter Store verwenden, um Ihre Geheimnisse zu speichern. Weitere Informationen zu Secrets Manager finden Sie unter Was ist Secrets Manager? im AWS Secrets Manager Benutzerhandbuch. Weitere Informationen zum AWS Systems Manager Parameterspeicher finden Sie unter AWS Systems Manager Parameterspeicher im AWS Systems Manager Benutzerhandbuch.

  - name (string, erforderlich) — Der Name des festzulegenden Komponentenparameters.
  - value (Zeichenkettenarray, erforderlich) — Enthält ein Array von Zeichenketten, um den Wert für den benannten Komponentenparameter festzulegen. Wenn für die Komponente ein Standardwert definiert ist und kein anderer Wert angegeben wird, wird der Standardwert AWSTOE verwendet.

- additionalInstanceConfiguration(Objekt) — Geben Sie zusätzliche Einstellungen und Startskripts für Ihre Build-Instances an.
  - systemsManagerAgent(Objekt) — Enthält Einstellungen für den Systems Manager Manager-Agenten auf Ihrer Build-Instanz.
    - uninstallAfterBuild(Boolean) — Steuert, ob der Systems Manager Manager-Agent vor der Erstellung des neuen AMI aus Ihrem endgültigen Build-Image entfernt wird. Wenn diese Option auf gesetzt ist`true`, wird der Agent aus dem endgültigen Image entfernt. Wenn

die Option auf gesetzt ist`false`, bleibt der Agent aktiviert, sodass er in das neue AMI aufgenommen wird. Der Standardwert ist `false`.

> **Note**
>
> Wenn das `uninstallAfterBuild` Attribut nicht in der JSON-Datei enthalten ist und die folgenden Bedingungen zutreffen, entfernt Image Builder den Systems Manager Manager-Agenten aus dem endgültigen Image, sodass er im AMI nicht verfügbar ist:
>
> - Das `userDataOverride` ist leer oder wurde in der JSON-Datei weggelassen.
> - Image Builder installierte den Systems Manager Manager-Agenten automatisch auf der Build-Instanz für ein Betriebssystem, bei dem der Agent nicht auf dem Basis-Image vorinstalliert war.

- userDataOverride(string) — Geben Sie Befehle oder ein Befehlsskript an, das ausgeführt werden soll, wenn Sie Ihre Build-Instance starten.

> **Note**
>
> Die Benutzerdaten sind immer Basis-64-codiert.. Zum Beispiel sind die folgenden Befehle als `IyEvYmluL2Jhc2gKbWtkaXIgLXAgL3Zhci9iYi8KdG91Y2ggL3Zhcg==` codiert:
>
> ```
> #!/bin/bash
> mkdir -p /var/bb/
> touch /var
> ```
>
> Das Linux-Beispiel verwendet diesen codierten Wert.

Linux

Das Basis-Image (`parentImage`Eigenschaft) im folgenden Beispiel ist ein AMI. Wenn Sie ein AMI verwenden, müssen Sie Zugriff auf das AMI haben, und das AMI muss sich in der Quellregion befinden (dieselbe Region, in der Image Builder den Befehl ausführt). Speichern Sie die Datei `create-image-recipe.json` unter und verwenden Sie sie im create-image-recipe Befehl.

```
{
"name": "BB Ubuntu Image recipe",
"description": "Hello World image recipe for Linux.",
"parentImage": "ami-0a01b234c5de6fabc",
"semanticVersion": "1.0.0",
"components": [
 {
  "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/bb$"
 }
],
"additionalInstanceConfiguration": {
 "systemsManagerAgent": {
   "uninstallAfterBuild": true
 },
 "userDataOverride": "IyEvYmluL2Jhc2gKbWtkaXIgLXAgL3Zhci9iYi8KdG91Y2ggL3Zhcg=="
}
}
```

Windows

Das folgende Beispiel bezieht sich auf die neueste Version des englischen Full Base-Images von Windows Server 2016. Der ARN in diesem Beispiel verweist auf das neueste Bild in der SKU, basierend auf den von Ihnen angegebenen semantischen Versionsfiltern:`arn:aws:imagebuilder:us-west-2:aws:image/windows-server-2016-english-full-base-x86/x.x.x`.

```
{
"name": "MyBasicRecipe",
"description": "This example image recipe creates a Windows 2016 image.",
"parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/windows-server-2016-english-full-base-x86/x.x.x",
"semanticVersion": "1.0.0",
"components": [
 {
  "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/my-example-component/2019.12.02/1"
 },
 {
  "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/my-imported-component/1.0.0/1"
 }
```

```
]
}
```

> **Note**
>
> Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung

2. Erstellen Sie das Rezept

   Verwenden Sie den folgenden Befehl, um das Rezept zu erstellen. Geben Sie den Namen der JSON-Datei, die Sie im vorherigen Schritt erstellt haben, in den `--cli-input-json` Parameter ein:

   ```
   aws imagebuilder create-image-recipe --cli-input-json file://create-image-recipe.json
   ```

   > **Note**
   >
   > - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
   > - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Importieren Sie eine VM als Ihr Basis-Image in die Konsole

In diesem Abschnitt konzentrieren wir uns darauf, wie Sie eine virtuelle Maschine (VM) als Basisimage für Ihr Image-Rezept importieren. Andere Schritte im Zusammenhang mit der Erstellung eines Rezepts oder einer Rezeptversion werden hier nicht behandelt. Weitere Schritte zum Erstellen eines neuen Image-Rezepts mit dem Assistenten zur Pipelineerstellung in der Image Builder Builder-Konsole finden Sie unterErstellen Sie eine Image-Pipeline (AMI). Weitere Schritte zum Erstellen eines neuen Image-Rezepts oder einer neuen Rezeptversion finden Sie unterErstellen Sie eine neue Version eines Bildrezepts.

Um eine VM als Basis-Image für Ihr Image-Rezept in der Image Builder Builder-Konsole zu importieren, führen Sie diese Schritte zusammen mit allen anderen erforderlichen Schritten aus, um Ihr Rezept oder Ihre Rezeptversion zu erstellen.

1. Wählen Sie im Abschnitt Image auswählen für das Basis-Image die Option Basis-Image importieren aus.
2. Wählen Sie das Image-Betriebssystem (OS) und die Betriebssystemversion wie gewohnt aus.

Konfiguration für den VM-Import

Wenn Sie Ihre VM aus ihrer Virtualisierungsumgebung exportieren, erstellt dieser Prozess einen Satz von einer oder mehreren Festplattencontainerdateien, die als Snapshots Ihrer VM-Umgebung, Einstellungen und Daten dienen. Sie können diese Dateien verwenden, um Ihre VM als Basisimage für Ihr Image-Rezept zu importieren. Weitere Informationen zum Importieren von VMs in Image Builder finden Sie unter VM-Images importieren und exportieren

Gehen Sie folgendermaßen vor, um den Speicherort Ihrer Importquelle anzugeben:

Quelle importieren

Geben Sie die Quelle für den ersten VM-Image-Festplattencontainer oder Snapshot, der importiert werden soll, im Abschnitt Festplattencontainer 1 an.

1. Quelle — Dies kann entweder ein S3-Bucket oder ein EBS-Snapshot sein.
2. Wählen Sie den S3-Speicherort der Festplatte — Geben Sie den Speicherort in Amazon S3 ein, an dem Ihre Festplatten-Images gespeichert sind. Um nach dem Speicherort zu suchen, wählen Sie Browse S3.
3. Um einen Festplattencontainer hinzuzufügen, wählen Sie Festplattencontainer hinzufügen.

IAM-Rolle

Um Ihrer VM-Importkonfiguration eine IAM-Rolle zuzuordnen, wählen Sie die Rolle aus der Dropdownliste der IAM-Rolle aus oder wählen Sie Neue Rolle erstellen, um eine neue zu erstellen. Wenn Sie eine neue Rolle erstellen, wird die Konsolenseite für IAM-Rollen auf einer separaten Registerkarte geöffnet.

Erweiterte Einstellungen — optional

Die folgenden Einstellungen sind optional. Mit diesen Einstellungen können Sie Verschlüsselung, Lizenzierung, Tags und mehr für das Basis-Image konfigurieren, das durch den Import erstellt wird.

Allgemeines

1. Geben Sie einen eindeutigen Namen für das Basis-Image an. Wenn Sie keinen Wert eingeben, erbt das Basis-Image den Rezeptnamen.
2. Geben Sie eine Version für das Basis-Image an. Verwenden Sie das folgende Format: *<major>.<minor>.<patch>*. Wenn Sie keinen Wert eingeben, erbt das Basis-Image die Rezeptversion.
3. Sie können auch eine Beschreibung für das Basis-Image eingeben.

Architektur des Basis-Images

Um die Architektur Ihrer VM-Importquelle anzugeben, wählen Sie einen Wert aus der Architekturliste aus.

Verschlüsselung

Wenn Ihre VM-Disk-Images verschlüsselt sind, müssen Sie einen Schlüssel angeben, den Sie für den Importvorgang verwenden können. Um einen AWS KMS key für den Import anzugeben, wählen Sie einen Wert aus der Liste Verschlüsselung (KMS-Schlüssel) aus. Die Liste enthält KMS-Schlüssel, auf die Ihr Konto in der aktuellen Region Zugriff hat.

Lizenzverwaltung

Wenn Sie eine VM importieren, erkennt der Importvorgang automatisch das VM-Betriebssystem und wendet die entsprechende Lizenz auf das Basis-Image an. Abhängig von Ihrer Betriebssystemplattform lauten die Lizenztypen wie folgt:

- Lizenz enthalten — Eine entsprechende AWS Lizenz für Ihre Plattform wird auf Ihr Basis-Image angewendet.
- Bringen Sie Ihre eigene Lizenz mit (BYOL) — Behält die Lizenz von Ihrer VM bei, falls zutreffend.

Um Lizenzkonfigurationen, die mit erstellt wurden, AWS License Manager an Ihr Basis-Image anzuhängen, wählen Sie aus der Liste mit den Namen der Lizenzkonfiguration aus. Weitere Informationen zu License Manager finden Sie unter Arbeiten mit AWS License Manager

> **Note**
>
> - Lizenzkonfigurationen enthalten Lizenzregeln, die auf den Bedingungen Ihrer Unternehmensvereinbarungen basieren.
> - Linux unterstützt nur BYOL-Lizenzen.

Tags (Basisbild)

Tags verwenden Schlüssel-Wert-Paare, um Ihrer Image Builder Builder-Ressource durchsuchbaren Text zuzuweisen. Um Tags für das importierte Basisbild anzugeben, geben Sie Schlüssel-Wert-Paare in die Felder Schlüssel und Wert ein.

Um einen Tag hinzuzufügen, wählen Sie Add tag (Tag hinzufügen). Klicken Sie zum Entfernen eines Tags auf Tag entfernen.

Eine neue Version eines Container-Rezepts erstellen

In diesem Abschnitt erfahren Sie, wie Sie eine neue Version eines Container-Rezepts erstellen.

Inhalt

Erstellen Sie mit der Konsole eine neue Container-Rezeptversion

Das Erstellen einer neuen Version eines Container-Rezepts entspricht praktisch dem Erstellen eines neuen Rezepts. Der Unterschied besteht darin, dass in den meisten Fällen bestimmte Details so ausgewählt sind, dass sie dem Basisrezept entsprechen. In der folgenden Liste werden die Unterschiede zwischen dem Erstellen eines neuen Rezepts und dem Erstellen einer neuen Version eines vorhandenen Rezepts beschrieben.

Einzelheiten zum Rezept

- Name — nicht editierbar.
- Version — Erforderlich. Dieses Detail ist nicht mit der aktuellen Version oder einer Sequenz vorausgefüllt. Geben Sie die Versionsnummer, die Sie erstellen möchten, im Format major.minor.patch ein. Wenn die Version bereits existiert, tritt ein Fehler auf.

Basisbild

- Bildoption auswählen — Vorausgewählt, aber editierbar. Wenn Sie Ihre Wahl für die Quelle Ihres Basisimages ändern, gehen möglicherweise weitere Details verloren, die von der ursprünglich ausgewählten Option abhängen.

  Um Details zu sehen, die mit Ihrer Auswahl des Basisimages verknüpft sind, wählen Sie die Registerkarte, die Ihrer Auswahl entspricht.

  Managed images

  - Image-Betriebssystem (OS) — Nicht editierbar.
  - Bildname — Vorausgewählt, basierend auf der Kombination von Basis-Image-Optionen, die Sie für das bestehende Rezept getroffen haben. Wenn Sie jedoch die Option Bild auswählen ändern, verlieren Sie den Namen des vorausgewählten Images.
  - Optionen für die automatische Versionierung — Entspricht nicht Ihrem Grundrezept. Die Optionen für die automatische Versionierung sind standardmäßig auf die Option Ausgewählte Betriebssystemversion verwenden eingestellt.

    > **Important**
    >
    > Wenn Sie semantische Versionierung verwenden, um Pipeline-Builds zu starten, stellen Sie sicher, dass Sie diesen Wert auf Letzte verfügbare Betriebssystemversion verwenden ändern. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung

  ECR image

  - Image-Betriebssystem (OS) — Vorausgewählt, aber editierbar.
  - Betriebssystemversion — Vorausgewählt, aber editierbar.
  - ECR-Bild-ID — Vorausgefüllt, aber editierbar.

  Docker Hub image

  - Image-Betriebssystem (OS) — Nicht editierbar.
  - Betriebssystemversion — Vorausgewählt, aber editierbar.
  - Docker-Image-ID — Vorausgefüllt, aber editierbar.

Instance-Konfiguration

- AMI-ID — Vorausgefüllt, aber editierbar.
- Speicher (Volumen)

  EBS-Volumen 1 (AMI-Root) — Vorgefüllt. Sie können den Gerätenamen, den Snapshot oder die IOPS-Auswahl des Root-Volumes nicht bearbeiten. Sie können jedoch alle übrigen Einstellungen ändern, z. B. die Größe. Sie können auch neue Volumes hinzufügen.

  > **Note**
  >
  > Wenn Sie ein Basis-AMI angegeben haben, das von einem anderen Konto für Sie freigegeben wurde, müssen die Snapshots für alle angegebenen sekundären Volumes auch mit Ihrem Konto geteilt werden.

Arbeitsverzeichnis

- Pfad zum Arbeitsverzeichnis — Vorausgefüllt, aber editierbar.

Komponenten

- Komponenten — Komponenten, die bereits in der Rezeptur enthalten sind, werden im Abschnitt Ausgewählte Komponenten am Ende jeder Komponentenliste (Build und Test) angezeigt. Sie können die ausgewählten Komponenten je nach Bedarf entfernen oder neu anordnen.

  CIS-Härtungskomponenten folgen nicht den Standardregeln für die Reihenfolge der Komponenten in den Image Builder Builder-Rezepten. Die CIS-Härtungskomponenten werden immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests anhand Ihres Ausgabebilds ausgeführt werden.

  > **Note**
  >
  > In den Listen der Build- und Testkomponenten werden die verfügbaren Komponenten basierend auf dem Typ des Komponentenbesitzers angezeigt. Um Komponenten für Ihr Rezept hinzuzufügen oder zu aktualisieren, wählen Sie den Besitzertyp für die Komponente aus, nach der Sie suchen. Wenn Sie beispielsweise eine Komponente hinzufügen möchten, die mit einem Basis-Image verknüpft ist, das Sie abonniert haben

AWS Marketplace, wählen Sie in der Liste mit dem Besitzertyp neben der Suchleiste einen Eintrag `Third party managed` aus.

Sie können die folgenden Einstellungen für die ausgewählte Komponente konfigurieren:

- Versionierungsoptionen — Vorausgewählt, aber Sie können sie ändern. Wir empfehlen Ihnen, die Option Letzte verfügbare Komponentenversion verwenden zu wählen, um sicherzustellen, dass Ihre Image-Builds immer die neueste Version der Komponente verwenden. Wenn Sie in Ihrem Rezept eine bestimmte Komponentenversion verwenden müssen, können Sie Komponentenversion angeben wählen und die Version in das angezeigte Feld Komponentenversion eingeben.
- Eingabeparameter — Zeigt Eingabeparameter an, die die Komponente akzeptiert. Der Wert ist bereits mit dem Wert aus der vorherigen Version des Rezepts gefüllt. Wenn Sie diese Komponente zum ersten Mal in diesem Rezept verwenden und ein Standardwert für den Eingabeparameter definiert wurde, wird der Standardwert im Feld Wert mit ausgegrautem Text angezeigt. Wenn kein anderer Wert eingegeben wird, verwendet Image Builder den Standardwert.

  Wenn ein Eingabeparameter erforderlich ist, aber in der Komponente kein Standardwert definiert ist, müssen Sie einen Wert angeben. Image Builder erstellt die Rezeptversion nicht, wenn erforderliche Parameter fehlen und für die kein Standardwert definiert ist.

  Important

  Komponentenparameter sind reine Textwerte und sind angemeldet AWS CloudTrail. Wir empfehlen, dass Sie AWS Secrets Manager oder den AWS Systems Manager Parameter Store verwenden, um Ihre Geheimnisse zu speichern. Weitere Informationen zu Secrets Manager finden Sie unter Was ist Secrets Manager? im AWS Secrets Manager Benutzerhandbuch. Weitere Informationen zum AWS Systems Manager Parameterspeicher finden Sie unter AWS Systems Manager Parameterspeicher im AWS Systems Manager Benutzerhandbuch.

Um die Einstellungen für Versionierungsoptionen oder Eingabeparameter zu erweitern, können Sie den Pfeil neben dem Namen der Einstellung wählen. Um alle Einstellungen für alle ausgewählten Komponenten zu erweitern, können Sie den Schalter Alle erweitern aus- und einschalten.

Dockerfile-Vorlage

- Dockerfile-Vorlage — Vorgefüllt, aber editierbar. Sie können jede der folgenden Kontextvariablen angeben, die Image Builder zur Laufzeit durch Buildinformationen ersetzt.

  ParentImage (erforderlich)

  Bei der Erstellung wird diese Variable in das Basis-Image für Ihr Rezept aufgelöst.

  Beispiel:

  ```
  FROM
  {{{ imagebuilder:parentImage }}}
  ```

  Umgebungen (erforderlich, wenn Komponenten angegeben sind)

  Diese Variable wird in ein Skript aufgelöst, das Komponenten ausführt.

  Beispiel:

  ```
  {{{ imagebuilder:environments }}}
  ```

  Komponenten (optional)

  Image Builder löst Build- und Testkomponentenskripts für die Komponenten auf, die das Container-Rezept enthält. Diese Variable kann an einer beliebigen Stelle im Dockerfile hinter der Umgebungsvariablen platziert werden.

  Beispiel:

  ```
  {{{ imagebuilder:components }}}
  ```

Ziel-Repository

- Name des Ziel-Repositorys — Das Amazon ECR-Repository, in dem Ihr Ausgabebild gespeichert ist, sofern in der Verteilungskonfiguration Ihrer Pipeline für die Region, in der die Pipeline ausgeführt wird (Region 1), kein anderes Repository angegeben ist.

So erstellen Sie eine neue Container-Rezeptversion:

1. Wählen Sie oben auf der Seite mit den Container-Rezeptdetails die Option Neue Version erstellen aus. Sie werden zur Seite „Rezept erstellen“ für Container-Rezepte weitergeleitet.
2. Um die neue Version zu erstellen, nehmen Sie Ihre Änderungen vor und wählen Sie dann Rezept erstellen.

Weitere Informationen zum Erstellen eines Container-Rezepts beim Erstellen einer Image-Pipeline finden Sie Schritt 2: Wählen Sie ein Rezept im Abschnitt Erste Schritte dieses Handbuchs.

Erstellen Sie ein Container-Rezept mit dem AWS CLI

Gehen Sie wie folgt vor, um ein Image Builder Builder-Container-Rezept mit dem `imagebuilder create-container-recipe` Befehl in der AWS CLI zu erstellen:

Voraussetzungen

Bevor Sie die Image Builder Builder-Befehle in diesem Abschnitt ausführen, um ein Container-Rezept mit dem zu erstellen AWS CLI, müssen Sie die Komponenten erstellen, die das Rezept verwenden soll. Das Beispiel für ein Container-Rezept im folgenden Schritt bezieht sich auf Beispielkomponenten, die im Erstellen Sie eine Komponente mit dem AWS CLI Abschnitt dieses Handbuchs erstellt wurden.

Nachdem Sie Ihre Komponenten erstellt haben oder wenn Sie vorhandene Komponenten verwenden, notieren Sie sich die ARNs, die Sie in das Rezept aufnehmen möchten.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Sie können die gesamte Eingabe für den create-container-recipe Befehl mit Inline-Befehlsparametern bereitstellen. Der resultierende Befehl kann jedoch ziemlich lang sein. Um den Befehl zu vereinfachen, können Sie stattdessen eine JSON-Datei bereitstellen, die alle Rezepteinstellungen des Containers enthält

   > **Note**
   >
   > Die Benennungskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Aktionsanforderungsparameter der Image Builder Builder-API angegeben ist. Informationen zu den API-Befehlsanforderungsparametern finden Sie unter dem CreateContainerRecipeBefehl in der EC2 Image Builder API-Referenz.

Informationen zur Bereitstellung der Datenwerte als Befehlszeilenparameter finden Sie in den Parameternamen, die in der AWS CLI Befehlsreferenz angegeben sind.

Im Folgenden finden Sie eine Zusammenfassung der Parameter in diesem Beispiel:

- components (Array von Objekten, erforderlich) — Enthält ein Array von `ComponentConfiguration` Objekten. Es muss mindestens eine Build-Komponente angegeben werden:

  > **Note**
  >
  > Image Builder installiert Komponenten in der Reihenfolge, in der Sie sie im Rezept angegeben haben. CIS-Härtungskomponenten werden jedoch immer zuletzt ausgeführt, um sicherzustellen, dass die Benchmark-Tests anhand Ihres Ausgabe-Images ausgeführt werden.

  - componentArn (string, required) — Der Komponenten-ARN.

    > **Tip**
    >
    > Um das Beispiel zu verwenden, um Ihr eigenes Container-Rezept zu erstellen, ersetzen Sie die Beispiel-ARNs durch die ARNs für die Komponenten, die Sie für Ihr Rezept verwenden,. Dazu gehören jeweils der Name AWS-Region, der Name und die Versionsnummer.

  - Parameter (Anordnung von Objekten) — Enthält eine Reihe von `ComponentParameter` Objekten. Wenn ein Eingabeparameter erforderlich ist, aber in der Komponente kein Standardwert definiert ist, müssen Sie einen Wert angeben. Image Builder erstellt die Rezeptversion nicht, wenn erforderliche Parameter fehlen und für die kein Standardwert definiert ist.

    > **Important**
    >
    > Komponentenparameter sind reine Textwerte und sind angemeldet AWS CloudTrail. Wir empfehlen, dass Sie AWS Secrets Manager oder den AWS Systems Manager Parameter Store verwenden, um Ihre Geheimnisse zu speichern. Weitere

Informationen zu Secrets Manager finden Sie unter Was ist Secrets Manager? im AWS Secrets Manager Benutzerhandbuch. Weitere Informationen zum AWS Systems Manager Parameterspeicher finden Sie unter AWS Systems Manager Parameterspeicher im AWS Systems Manager Benutzerhandbuch.

    - name (string, erforderlich) — Der Name des festzulegenden Komponentenparameters.
    - value (Zeichenkettenarray, erforderlich) — Enthält ein Array von Zeichenketten, um den Wert für den benannten Komponentenparameter festzulegen. Wenn für die Komponente ein Standardwert definiert ist und kein anderer Wert angegeben wird, wird der Standardwert AWSTOE verwendet.
- containerType (string, required) — Der Typ des zu erstellenden Containers. Zu den gültigen Werten gehören. `DOCKER`
- dockerfileTemplateData(string) — Die Dockerfile-Vorlage, die zum Erstellen Ihres Images verwendet wird, ausgedrückt als Inline-Datenblob.
- name (string, erforderlich) — Der Name des Container-Rezepts.
- description (string) — Die Beschreibung des Container-Rezepts.
- parentImage (string, erforderlich) — Bild, das das Container-Rezept als Grundlage für Ihr benutzerdefiniertes Bild verwendet. Der Wert kann der ARN des Basis-Images oder eine AMI-ID sein.
- PlatformOverride (Zeichenfolge) — Gibt die Betriebssystemplattform an, wenn Sie ein benutzerdefiniertes Basisimage verwenden.
- semanticVersion <major>(string, erforderlich) — Die semantische Version des Container-Rezepts, angegeben im folgenden Format, mit numerischen Werten an jeder Position, um eine bestimmte Version anzugeben:. <minor>. <patch>. Ein Beispiel wäre `1.0.0`. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung
- tags (String-Map) — Tags, die an das Container-Rezept angehängt sind.
- instanceConfiguration (object) — Eine Gruppe von Optionen, die verwendet werden können, um eine Instanz zum Erstellen und Testen von Container-Images zu konfigurieren.
    - image (string) — Die AMI-ID, die als Basisimage für eine Container-Build- und Test-Instance verwendet werden soll. Wenn Sie diesen Wert nicht angeben, verwendet Image Builder das entsprechende Amazon ECS-optimierte AMI als Basis-Image.

- blockDeviceMappings(Array von Objekten) — Definiert die Blockgeräte, die zum Erstellen einer Instanz aus dem im image Parameter angegebenen Image Builder Builder-AMI angeschlossen werden sollen.
  - deviceName (string) — Das Gerät, für das diese Zuordnungen gelten.
  - ebs (object) — Wird verwendet, um die Amazon EBS-spezifische Konfiguration für diese Zuordnung zu verwalten.
    - deleteOnTermination(Boolean) — Wird verwendet, um das Löschen bei Beendigung des zugehörigen Geräts zu konfigurieren.
    - encrypted (Boolean) — Wird zur Konfiguration der Geräteverschlüsselung verwendet.
    - volumeSize (Ganzzahl) — Wird verwendet, um die Volumengröße des Geräts zu überschreiben.
    - volumeType (string) — Wird verwendet, um den Lautstärketyp des Geräts zu überschreiben.
- targetRepository (object, required) — Das Ziel-Repository für das Container-Image, falls in der Verteilungskonfiguration Ihrer Pipeline für die Region, in der die Pipeline ausgeführt wird (Region 1), kein anderes Repository angegeben ist.
  - repositoryName (string, erforderlich) — Der Name des Container-Repositorys, in dem das Ausgabe-Container-Image gespeichert ist. Diesem Namen wird der Speicherort des Repositorys vorangestellt.
  - service (string, erforderlich) — Gibt den Dienst an, in dem dieses Image registriert wurde.
- workingDirectory (string) — Das Arbeitsverzeichnis für die Verwendung bei Build- und Test-Workflows.

```
{
 "components": [
   {
   "componentArn": "arn:aws:imagebuilder:us-east-1:123456789012:component/
helloworldal2/x.x.x"
   }
 ],
 "containerType": "DOCKER",
 "description": "My Linux Docker container image",
 "dockerfileTemplateData": "FROM
 {{{ imagebuilder:parentImage }}}\n{{{ imagebuilder:environments }}}\n{{{ imagebuilder:comp
 "name": "amazonlinux-container-recipe",
 "parentImage": "amazonlinux:latest",
```

```
 "platformOverride": "Linux",
 "semanticVersion": "1.0.2",
 "tags": {
   "sometag" : "Tag detail"
 },
 "instanceConfiguration": {
   "image": "ami-1234567890",
   "blockDeviceMappings": [
   {
   "deviceName": "/dev/xvda",
   "ebs": {
    "deleteOnTermination": true,
    "encrypted": false,
    "volumeSize": 8,
    "volumeType": "gp2"
    }
    }
   ]
 },
 "targetRepository": {
   "repositoryName": "myrepo",
   "service": "ECR"
 },
 "workingDirectory": "/tmp"
}
```

2. Erstellen Sie das Rezept

   Verwenden Sie den folgenden Befehl, um das Rezept zu erstellen. Geben Sie den Namen der JSON-Datei, die Sie im vorherigen Schritt erstellt haben, in den `--cli-input-json` Parameter ein:

   ```
   aws imagebuilder create-container-recipe --cli-input-json file://create-container-recipe.json
   ```

   > **Note**
   >
   > - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
   > - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet

beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Bereinigen von -Ressourcen

Um unerwartete Gebühren zu vermeiden, sollten Sie die Ressourcen und Pipelines, die Sie anhand der Beispiele in diesem Handbuch erstellt haben, bereinigen. Weitere Informationen zum Löschen von Ressourcen in Image Builder finden Sie unterEC2 Image Builder Builder-Ressourcen löschen.

EC2 Image Builder Builder-Images verwalten

Nachdem Sie mit Image Builder Image-Ressourcen für AMI- oder Container-Images erstellt haben, können Sie diese mit der Image Builder Builder-Konsole, über die Image Builder Builder-API oder mit imagebuilder Befehlen in der verwalten AWS CLI.

> **Tip**
>
> Wenn Sie über mehrere Ressourcen desselben Typs verfügen, hilft Ihnen das Tagging dabei, eine bestimmte Ressource anhand der Tags zu identifizieren, die Sie ihr zugewiesen haben. Weitere Informationen zum Taggen Ihrer Ressourcen mithilfe von Image Builder Builder-Befehlen finden Sie im AWS CLIMarkieren von Ressourcen Abschnitt dieses Handbuchs.

In diesem Abschnitt wird beschrieben, wie Sie Bilder auflisten, anzeigen und erstellen. Informationen zu Bild-Workflows und deren Verwaltung finden Sie unterVerwaltung von Build- und Test-Workflows für EC2 Image Builder Builder-Images.

Inhalt

Images und Build-Versionen auflisten

Auf der Seite Images in der Image Builder Builder-Konsole finden Sie Listen aller Image Builder Builder-Image-Ressourcen, die Sie besitzen, die mit Ihnen geteilt wurden und auf die Sie Zugriff haben. Die Listenergebnisse enthalten einige wichtige Details zu diesen Ressourcen.

Sie können auch alle Bilder in Ihrem Konto sehen, für die Workflow-Aktionen ausstehen.

Inhalt

Bilder auflisten

In diesem Abschnitt werden die verschiedenen Möglichkeiten beschrieben, wie Sie Informationen zu Ihren Bildern auflisten können.

Sie können eine der folgenden Methoden verwenden, um Image Builder Builder-Image-Ressourcen aufzulisten, auf die Sie Zugriff haben. Informationen zur API-Aktion finden Sie ListImagesin der EC2 Image Builder API-Referenz. Die zugehörige SDK-Anfrage finden Sie unter dem Link „Siehe auch“ auf derselben Seite.

Inhalt

Bilder in der Konsole auflisten

Gehen Sie wie folgt vor, um die Seite mit der Bilderliste in der Konsole zu öffnen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich „Bilder“ aus.

Die Seite „Bilder“ in der Konsole ist je nach Eigentümerschaft oder ausstehenden Workflow-Aktionen in Tabs unterteilt. In diesem Abschnitt werden die ersten drei Tabs behandelt, auf denen Bilder angezeigt werden, die Ihnen gehören oder auf die Sie Zugriff haben.

Registerkarte „Konsole“: Gehört mir

Auf der Registerkarte „Mein Eigentum“ können Sie die folgenden Filter verwenden, um die Ergebnisse der Bilderliste zu optimieren.

- Sie können in der Suchleiste nach dem gesamten Namen oder einem Teil davon suchen.
- Sie können Bilder nach ihrer Betriebssystemplattform (Windows oder Linux) filtern.
- Sie können Bilder nach der Art der Ausgabe filtern, die sie erzeugen (AMI oder Container-Image).
- Sie können die Filterquelle verwenden, um Bilder zu finden, die mit der VMIE von einer virtuellen Maschine importiert wurden.

Nach den Filtersteuerungen wird auf der Registerkarte „Mein Eigentum“ eine Liste der Image Builder Builder-Images angezeigt, die Sie erstellt haben, mit den folgenden Details für die aufgelisteten Ressourcen:

Name/Version

Die Namen der Image Builder Builder-Bildressourcen beginnen mit dem Rezeptnamen und der Version, aus der sie erstellt wurden. Wählen Sie den Link aus, um alle zugehörigen Image-Build-Versionen anzuzeigen.

Typ

Der Typ des Ausgabebilds, das Image Builder für diese Image-Ressource erstellt (ein AMI oder ein Container-Image).

Plattform

Die Betriebssystemplattform der Image-Ressourcenversion, z. B. „Windows“ oder „Linux“.

Quelle des Bilds

Der Ursprung des Basis-Images, mit dem Image Builder diese Image-Ressource erstellt hat. Dies wird hauptsächlich verwendet, um Ergebnisse für Images zu filtern, die von einer virtuellen Maschine (VMIE) importiert wurden.

Zeitpunkt der Erstellung

Datum und Uhrzeit der Erstellung der aktuellen Version der Bildressource durch Image Builder.

ARN

Der Amazon-Ressourcenname (ARN) der aktuellen Version der Bildressource.

Registerkarte „Konsole“: Mit mir geteilt

Auf der Registerkarte „Für mich freigegeben“ können Sie die folgenden Filter verwenden, um die Ergebnisse der Bilderliste zu optimieren.

- Sie können in der Suchleiste nach dem gesamten Namen oder einem Teil davon suchen.
- Sie können Bilder nach ihrer Betriebssystemplattform (Windows oder Linux) filtern.
- Sie können Bilder nach der Art der Ausgabe filtern, die sie erzeugen (AMI oder Container-Image).
- Sie können die Filterquelle verwenden, um Bilder zu finden, die mit der VMIE von einer virtuellen Maschine importiert wurden.

Nach den Filtersteuerungen wird auf der Registerkarte Für mich freigegeben eine Liste der Image Builder Builder-Bilder angezeigt, die für Sie freigegeben wurden, mit den folgenden Details für die aufgelisteten Ressourcen:

Name des Bilds

Der Name der Bildressource, die mit Ihnen geteilt wurde. Um ein geteiltes Bild in einem Rezept zu verwenden, wählen Sie die Option Verwaltete Bilder auswählen und ändern den Ursprung des Bilds in Mit mir geteilte Bilder.

Typ

Der Typ des Ausgabebilds, das Image Builder für diese Image-Ressource erstellt (ein AMI oder ein Container-Image).

Version

Die Betriebssystemplattform der Image-Ressourcenversion, z. B. „Windows“ oder „Linux“.

Quelle des Bilds

Der Ursprung des Basis-Images, mit dem Image Builder diese Image-Ressource erstellt hat, falls zutreffend. Dies wird hauptsächlich verwendet, um Ergebnisse für Images zu filtern, die von einer virtuellen Maschine (VMIE) importiert wurden.

Plattform

Die Betriebssystemplattform der Image-Ressourcenversion, zum Beispiel „Windows“ oder „Linux“.

Zeitpunkt der Erstellung

Datum und Uhrzeit, an dem Image Builder die Version der Bildressource erstellt hat, die für Sie freigegeben wurde.

Eigentümer

Der Besitzer der gemeinsam genutzten Bildressource.

ARN

Der Amazon-Ressourcenname (ARN) der Image-Ressourcenversion, die mit Ihnen geteilt wurde.

Registerkarte „Konsole“: Von Amazon verwaltet

Auf der Registerkarte „Von Amazon verwaltet“ können Sie die folgenden Filter verwenden, um die Ergebnisse der Bilderliste zu optimieren.

- Sie können in der Suchleiste nach dem gesamten Namen oder einem Teil davon suchen.
- Sie können Bilder nach ihrer Betriebssystemplattform (Windows oder Linux) filtern.
- Sie können Bilder nach der Art der Ausgabe filtern, die sie erzeugen (AMI oder Container-Image).
- Sie können die Filterquelle verwenden, um Bilder zu finden, die mit der VMIE von einer virtuellen Maschine importiert wurden.

Nach den Filtersteuerungen wird auf der Registerkarte „Von Amazon verwaltet“ eine Liste der von Amazon verwalteten Image Builder Builder-Images angezeigt, die Sie als Basisbilder für Ihre Rezepte verwenden können. Image Builder zeigt die folgenden Details für die aufgelisteten Ressourcen an:

Name des Bildes

Der Name des verwalteten Images. Wenn Sie ein Rezept erstellen, ist die Standardeinstellung für Ihr Basis-Image Quick Start (von Amazon verwaltet). Die auf dieser Registerkarte aufgelisteten Bilder füllen die Liste mit den Imagenamen, die der Betriebssystemplattform zugeordnet ist, die Sie bei der Erstellung eines Rezepts für Ihr Basis-Image ausgewählt haben.

Typ

Der Typ des Ausgabebilds, das Image Builder für diese Image-Ressource erstellt (ein AMI oder ein Container-Image).

Version

Die Betriebssystemplattform der Image-Ressourcenversion, z. B. „Windows“ oder „Linux“.

Plattform

Die Betriebssystemplattform der Image-Ressourcenversion, zum Beispiel „Windows“ oder „Linux“.

Zeitpunkt der Erstellung

Datum und Uhrzeit, an dem Image Builder die Version der Bildressource erstellt hat, die für Sie freigegeben wurde.

Eigentümer

Amazon ist Eigentümer der verwalteten Bilder.

ARN

Der Amazon-Ressourcenname (ARN) der Image-Ressourcenversion, die mit Ihnen geteilt wurde.

Bilder mit AWS CLI Befehlen auflisten

Wenn Sie den list-images Befehl in der ausführen AWS CLI, können Sie eine Liste der Bilder abrufen, die Ihnen gehören oder auf die Sie Zugriff haben.

Das folgende Befehlsbeispiel zeigt, wie Sie den list-images Befehl ohne Filter verwenden, um alle Image Builder Builder-Bildressourcen aufzulisten, die Sie besitzen.

Beispiel: Alle Bilder auflisten

```
aws imagebuilder list-images
```

Ausgabe:

```
{
 "requestId": "1abcd234-e567-8fa9-0123-4567b890cd12",
 "imageVersionList": [
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-name/1.0.0",
    "name": "image-recipe-name",
    "type": "AMI",
```

```
    "version": "1.0.0",
    "platform": "Linux",
    "owner": "123456789012",
    "dateCreated": "2022-04-28T01:38:23.286Z"
  },
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-win/1.0.1",
    "name": "image-recipe-win",
    "type": "AMI",
    "version": "1.0.1",
    "platform": "Windows",
    "owner": "123456789012",
    "dateCreated": "2022-04-28T01:38:23.286Z"
  }
 ]
}
```

Wenn Sie den list-images Befehl ausführen, können Sie Filter anwenden, um die Ergebnisse zu optimieren, wie das folgende Beispiel zeigt. Weitere Informationen zum Filtern Ihrer Ergebnisse finden Sie unter dem Befehl list-images in der AWS CLI Befehlsreferenz.

Beispiel: Filter für Linux-Bilder

```
aws imagebuilder list-images --filters name="platform",values="Linux"
```

Ausgabe:

```
{
 "requestId": "1abcd234-e567-8fa9-0123-4567b890cd12",
 "imageVersionList": [
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-name/1.0.0",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0",
    "platform": "Linux",
    "owner": "123456789012",
    "dateCreated": "2022-04-28T01:38:23.286Z"
  }
 ]
}
```

Listet Bilder auf, die auf eine Aktion warten

Wenn Sie die `WaitForAction` Schrittaktion in Ihrem Bild-Workflow verwenden, wird der Workflow angehalten, bis Sie ihr das Signal senden, die Verarbeitung fortzusetzen, oder der Workflow schlägt fehl. Sie können diese Schrittaktion verwenden, wenn Sie einen externen Prozess haben, der ausgeführt werden muss, bevor Sie fortfahren können. Sie können dann die verwenden`SendWorkflowStepAction`, um ein Signal an den unterbrochenen Schritt an oder zu `RESUME` senden. `STOP` Sie können Ihren Workflow auch von der Konsole aus beenden oder fortsetzen.

Die folgenden Tabs zeigen, wie Sie eine Liste aller Bildressourcen in Ihrem Konto mit Workflow-Schritten abrufen können, die derzeit pausiert sind, um auf ein Signal zu warten, das wieder aufgenommen oder beendet wird. Die Registerkarten behandeln die Schritte in der Konsole und den AWS CLI Befehl.

Sie können auch die API oder ein SDK verwenden, um eine Liste der Workflow-Schritte abzurufen, die noch ausgeführt werden müssen. Informationen zur API-Aktion finden Sie ListWaitingWorkflowStepsin der EC2 Image Builder API-Referenz. Die zugehörige SDK-Anfrage finden Sie unter dem Link „Siehe auch“ auf derselben Seite.

Console

Gehen Sie wie folgt vor, um in der Konsole zur Registerkarte Warten auf Aktion zu gelangen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich „Bilder“ aus. Dadurch wird die Seite mit der Bilderliste geöffnet.
3. Wählen Sie auf der Listenseite den Tab Warten auf Aktion aus.
4. (optional) Um einen Schritt zu beenden oder fortzusetzen, aktivieren Sie das Kontrollkästchen neben dem Namen und wählen Sie dann Schritt beenden oder Schritt fortsetzen aus. Sie können mehr als ein Kontrollkästchen aktivieren, um dieselbe Aktion für alle ausgewählten Schritte auszuführen.

Details zu ausstehenden Workflow-Schritten

Zu den Workflow-Details für den ausstehenden Schritt gehören:

- Bildname — Der Name der Bildressource, die den ausstehenden Schritt enthält. Sie können den Namenslink auswählen, um die Detailseite für dieses Bild anzuzeigen.
- Name des ausstehenden Schritts — Der Name des Workflow-Schritts, der auf eine Aktion wartet.
- Ausführungs-ID des Schritts — Identifiziert eindeutig die Laufzeitinstanz des Workflow-Schritts. Sie können die verknüpfte ID auswählen, um Laufzeitdetails für den Schritt anzuzeigen.
- Schrittstart — Der Zeitstempel, zu dem die Runtime-Instanz des Workflow-Schritts gestartet wurde.
- Workflow-ARN — Der Amazon-Ressourcenname (ARN) des Workflows mit dem ausstehenden Schritt.
- Aktionen — Die Schrittaktion, die sich im Wartestatus befindet.

AWS CLI

Wenn Sie den list-waiting-workflow-steps Befehl in der ausführen AWS CLI, erhalten Sie eine Liste aller Bilder in Ihrem Konto, deren Workflow-Schritte noch auf eine Aktion warten, bevor der Image-Erstellungsprozess abgeschlossen wird.

Das folgende Befehlsbeispiel zeigt, wie Sie den list-waiting-workflow-steps Befehl verwenden, um alle Bilder in Ihrem Konto mit Workflow-Schritten aufzulisten, die noch auf eine Aktion warten.

Beispiel: listet Bilder in Ihrem Konto auf, bei denen die Workflow-Schritte noch warten

```
aws imagebuilder list-waiting-workflow-steps
```

Ausgabe:

Die Ausgabe für dieses Beispiel zeigt ein Bild im Konto mit einem Schritt, der auf eine Aktion wartet.

```
{
    "steps": [
        {
            "imageBuildVersionArn": "arn:aws:imagebuilder:us-
west-2:111122223333:image/example-image/1.0.0/8",
            "name": "WaitForAction",
            "workflowExecutionId": "wf-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
```

```
            "stepExecutionId": "step-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "workflowBuildVersionArn": "arn:aws:imagebuilder:us-
west-2:111122223333:workflow/test/wait-for-action/1.0.0/1",
            "startTime": "2023-11-21T23:21:23.609Z",
            "action": "WaitForAction"
        }
    ]
}
```

Listet die Build-Versionen der Images auf

Auf der Seite Image-Build-Versionen der Image Builder Builder-Konsole finden Sie eine Liste der Build-Versionen und zusätzliche Details für eine Image-Ressource, die Sie besitzen. Sie können Befehle oder Aktionen auch mit der Image Builder Builder-API, den SDKs oder AWS CLI zum Auflisten von Image-Build-Versionen verwenden.

Sie können eine der folgenden Methoden verwenden, um Image-Build-Versionen für Image-Ressourcen aufzulisten, die Sie besitzen. Informationen zur API-Aktion finden Sie ListImageBuildVersionsin der EC2 Image Builder API-Referenz. Die zugehörige SDK-Anfrage finden Sie unter dem Link „Siehe auch“ auf derselben Seite.

Console

Einzelheiten zur Version

Auf der Seite mit den Image-Build-Versionen in der Image Builder Builder-Konsole finden Sie unter anderem die folgenden Informationen:

- Version — Die Build-Version der Image-Ressource. In der Image Builder Builder-Konsole ist die Version mit einer Image-Detailseite verknüpft.
- Typ — Der Ausgabetyp, den Image Builder bei der Erstellung dieser Image-Ressource (ein AMI oder ein Container-Image) verteilt hat.
- Erstellungsdatum — Datum und Uhrzeit der Erstellung der Image-Build-Version durch Image Builder.
- Image-Status — Der aktuelle Status der Image-Build-Version. Der Status kann sich auf die Erstellung oder Disposition des Images beziehen. Während des Erstellungsvorgangs wird Ihnen beispielsweise der Status `Building` oder angezeigt`Distributing`. Bei der Disposition des Images wird möglicherweise der Status `Deprecated` oder angezeigt`Deleted`.

- Grund für den Fehler — Der Grund für den Image-Status. In der Image Builder-Konsole wird nur der Grund angezeigt, wenn der Build fehlschlägt (Image-Status ist gleich`Failed`).
- Sicherheitsergebnisse — Die aggregierten Image-Scan-Ergebnisse für die referenzierte Image-Build-Version.
- ARN — Der Amazon-Ressourcenname (ARN) für die referenzierte Version der Bildressource.
- Log-Stream — Ein Link zu den Log-Stream-Details für die referenzierte Image-Build-Version.

Versionen auflisten

Gehen Sie wie folgt vor, um Image-Build-Versionen in der Image Builder-Konsole aufzulisten:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich die Option Images aus. Standardmäßig zeigt die Bilderliste die aktuelle Version aller Bilder, die Sie besitzen.
3. Um eine Liste aller Versionen für ein Bild zu sehen, wählen Sie den Link zur aktuellen Version. Der Link öffnet die Seite mit den Image-Build-Versionen, auf der alle Build-Versionen für ein bestimmtes Image aufgeführt sind.

AWS CLI

Wenn Sie den list-image-build-versions Befehl in der ausführen AWS CLI, erhalten Sie eine vollständige Liste der Build-Versionen für die angegebene Image-Ressource. Sie müssen Eigentümer des Images sein, um diesen Befehl ausführen zu können.

Das folgende Befehlsbeispiel zeigt, wie der list-image-build-versions Befehl verwendet wird, um alle Build-Versionen für das angegebene Image aufzulisten.

Beispiel: listet die Build-Versionen für ein bestimmtes Image auf

```
aws imagebuilder list-image-build-versions --image-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-name/1.0.0
```

Ausgabe:

Die Ausgabe für dieses Beispiel enthält zwei Build-Versionen für das angegebene Image-Rezept.

```
{
```

```
  "requestId": "12f3e45d-67cb-8901-af23-45ed678c9b01",
  "imageSummaryList": [
    {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-
name/1.0.0/2",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0/2",
    "platform": "Linux",
    "osVersion": "Amazon Linux 2",
    "state": {
      "status": "AVAILABLE"
    },
    "owner": "123456789012",
    "dateCreated": "2023-03-10T01:04:40.609Z",
    "outputResources": {
      "amis": [
        {
        "region": "us-west-2",
        "image": "ami-012b3456789012c3d",
        "name": "image-recipe-name 2023-03-10T01-05-12.541Z",
        "description": "First verison of image-recipe-name",
        "accountId": "123456789012"
        }
      ]
    },
    "tags": {}
    },
    {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-
name/1.0.0/1",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0/1",
    "platform": "Linux",
    "osVersion": "Amazon Linux 2",
    "state": {
      "status": "AVAILABLE"
    },
    "owner": "123456789012",
    "dateCreated": "2023-03-10T00:07:16.384Z",
    "outputResources": {
      "amis": [
        {
```

```
        "region": "us-west-2",
        "image": "ami-0d1e23456789f0a12",
        "name": "image-recipe-name 2023-03-10T00-07-18.146132Z",
        "description": "First verison of image-recipe-name",
        "accountId": "123456789012"
        }
      ]
    },
    "tags": {}
    }
  ]
}
```

> **Note**
>
> Die Ausgabe des list-image-build-versions Befehls enthält derzeit keine Sicherheitsergebnisse oder Protokollstreams.

Bilddetails anzeigen

Auf der Seite mit den Image-Details in der Image Builder Builder-Konsole können Sie Details für eine bestimmte Image-Ressource anzeigen, die Sie besitzen. Sie können Befehle oder Aktionen auch mit der Image Builder Builder-API, SDKs oder AWS CLI zum Abrufen von Bilddetails verwenden.

Weitere Informationen zu Ressourcen, die ein anderer Benutzer über eine AWS Resource Access Manager (AWS RAM) -Ressourcenfreigabe für Sie AWS-Konto freigegeben hat, finden Sie im AWS RAM Benutzerhandbuch unter Zugreifen auf AWS Ressourcen, die für Sie freigegeben wurden.

Inhalt

Bilddetails in der Image Builder Builder-Konsole anzeigen

Die Image-Detailseite in der Image Builder Builder-Konsole enthält einen Übersichtsabschnitt mit zusätzlichen Informationen, die in Registerkarten gruppiert sind. Die Seitenüberschrift enthält den Namen und die Build-Version des Rezepts, mit dem das Bild erstellt wurde.

Abschnitte und Registerkarten mit Konsolendetails

- Abschnitt „Zusammenfassung“
- Registerkarte Ausgaberessourcen
- Registerkarte „Konfiguration der Infrastruktur“
- Registerkarte „Vertriebseinstellungen“
- Registerkarte „Arbeitsablauf“
- Registerkarte „Sicherheitsergebnisse“
- Registerkarte „Tags“

Abschnitt „Zusammenfassung“

Der Abschnitt mit der Zusammenfassung erstreckt sich über die gesamte Breite der Seite und enthält die folgenden Details. Diese Details werden immer angezeigt.

Rezept

Der Name und die Version des Rezepts, die die Build-Version nicht enthalten. Wenn die Build-Version beispielsweise ist`sample-linux-recipe | 1.0.1/2`, ist das Rezept und die Build-Version ist2. `sample-linux-recipe | 1.0.1`

Erstellungsdatum

Datum und Uhrzeit der Erstellung der Image-Build-Version durch Image Builder.

Image-Status

Der aktuelle Status der Image-Build-Version. Der Status kann sich auf die Erstellung oder Disposition des Images beziehen. Während des Erstellungsvorgangs wird Ihnen beispielsweise der Status `Building` oder angezeigt`Distributing`. Bei der Disposition des Images wird möglicherweise der Status `Deprecated` oder angezeigt`Deleted`.

Grund für das Scheitern

Der Grund für den Image-Status. In der Image Builder-Konsole wird nur der Grund angezeigt, wenn der Build fehlschlägt (Image-Status ist gleich`Failed`).

Registerkarte Ausgaberessourcen

Auf der Registerkarte Ausgaberessourcen werden Ausgabe- und Verteilungsdetails für die Bildressource aufgeführt, die derzeit angezeigt wird. Die Informationen, die Image Builder anzeigt, hängen wie folgt von der Art des Rezepts ab, mit dem die Pipeline das Image erstellt hat.

Rezept für das Bild

- Region — Die Vertriebsregion für das ausgegebene Amazon Machine Image (AMI), das in der Spalte Image angegeben ist.
- Image — Die ID des AMI, das Image Builder an das Ziel verteilt hat. Diese ID ist mit der Amazon Machine Images (AMIs) -Seite in der Amazon EC2 EC2-Konsole verknüpft.

> **Note**
>
> Image Builder erstellt das AMI, nachdem es die Ausgabe-Image-Ressource erstellt hat und bevor es das AMI an das Ziel verteilt.

- Name — Der Name des AMI, das Image Builder an das Ziel verteilt hat.
- Beschreibung — Die optionale Beschreibung aus dem Image-Rezept, mit dem die Pipeline die Ausgabe-Image-Ressource erstellt hat.
- Konto — Das Konto AWS-Konto , dem die aktuell angezeigte Image Builder Builder-Bildressource gehört.

Rezept für Container

Image Builder zeigt die folgenden Details für die Ausgabe an, die aus einem Container-Rezept erstellt wurde.

- Region — Die Vertriebsregion für das Container-Image, das in der Spalte Image-URI angegeben ist.
- Bild-URI — Der URI des Ausgabecontainer-Images, das Image Builder an das ECR-Repository in der Zielregion verteilt hat.

> **Note**
>
> Image Builder zeigt eine Zeile pro Ziel an. Das Ausgabebild enthält immer mindestens einen Eintrag für die Verteilung an das Konto, mit dem das Bild erstellt wurde. Zusätzliche Ziele können Verteilungen über Regionen AWS-Konten, oder AWS Organizations umfassen. Weitere Informationen finden Sie unter EC2 Image Builder Builder-Verteilungseinstellungen verwalten.

Registerkarte „Konfiguration der Infrastruktur“

Auf der Registerkarte Infrastrukturkonfiguration werden die Amazon EC2 EC2-Infrastruktureinstellungen angezeigt, die Image Builder zum Erstellen und Testen des aktuell angezeigten Images verwendet hat. Image Builder zeigt immer den Namen der Infrastrukturkonfigurationsressource (Konfigurationsname) und ihren Amazon-Ressourcennamen (ARN) an. Wenn Ihre Infrastrukturkonfiguration die Werte festlegt, können zusätzliche Infrastrukturdetails Folgendes beinhalten

- Instance-Typen
- Ein Instanzprofil
- Netzwerk-Infrastruktur
- Einstellungen für Sicherheitsgruppen
- Ein Amazon S3 S3-Speicherort, an dem Image Builder Anwendungsprotokolle speichert
- Ein Amazon EC2 EC2-Schlüsselpaar zur Fehlerbehebung
- Ein Amazon SNS SNS-Thema für Ereignisbenachrichtigungen

Weitere Informationen finden Sie unter EC2 Image Builder Builder-Infrastrukturkonfiguration verwalten.

Registerkarte „Vertriebseinstellungen“

Auf der Registerkarte Verteilungseinstellungen werden Einstellungen angezeigt, die Image Builder zur Verteilung Ihrer Ausgabebilder verwendet hat. Image Builder zeigt immer den Namen der Distributionskonfigurationsressource (Konfigurationsname) und ihren Amazon-Ressourcennamen (ARN) an. Zusätzliche Verteilungsdetails hängen von der Art des Rezepts ab, das die Image Builder Builder-Pipeline zur Erstellung des Images verwendet hat, und zwar wie folgt:

Rezept für das Bild

Wenn Ihre Ressource für die Verteilungskonfiguration die Werte festlegt, können zusätzliche Verteilungsdetails Folgendes beinhalten:

- Region — Die Vertriebsregion für das ausgegebene Amazon Machine Image (AMI).
- Name des Ausgabe-AMIs — Der Name des AMI, das Image Builder an das Ziel verteilt hat.
- Verschlüsselung (KMS-Schlüssel) — Falls konfiguriert, die, AWS KMS key die Image Builder verwendet, um das Image für die Verteilung in die Zielregion zu verschlüsseln.
- Zielkonten für die Verteilung — Wenn Sie die kontoübergreifende Verteilung konfiguriert haben, wird in dieser Spalte eine durch Kommas getrennte Liste der Konten angezeigt, für die das Ausgabebild in der Zielregion freigegeben werden AWS-Konten soll.
- Principals mit gemeinsamen Rechten — Eine durch Kommas getrennte Liste der AWS Principals, die zum Beispiel berechtigt sind, Ihr Image zu starten, AWS-Konten oder Gruppen oder Organisationseinheiten (OUs). AWS Organizations

> **Note**
>
> Wenn Sie anderen Principals die Erlaubnis erteilen, Ihr Image zu starten, sind Sie immer noch Eigentümer des Images. AWS stellt Ihrem Konto alle Instances in Rechnung, die Amazon EC2 von Ihrem Image aus startet.

- Zielkonten für eine schnellere Startkonfiguration —
- Zugeordnete Lizenzkonfigurationen — Die License Manager Manager-Lizenzkonfiguration ARNs zur Verknüpfung mit dem AMI in der angegebenen Region.
- Vorlagenkonfiguration starten —
- Standardversion der Startvorlage festlegen —

Rezept für Container

Container-Distributionen enthalten immer die folgenden Details:

- Region — Die Verteilungsregion für das Container-Image, das in der Spalte Image-URI angegeben ist.
- Bild-URI — Der URI des Ausgabecontainer-Images, das Image Builder an das Amazon ECR-Repository in der Zielregion verteilt hat.

> **Note**
>
> Image Builder zeigt eine Zeile pro Ziel an. Das Ausgabebild enthält immer mindestens einen Eintrag für die Verteilung an das Konto, mit dem das Bild erstellt wurde. Zusätzliche Ziele können Verteilungen über Regionen AWS-Konten, oder AWS Organizations umfassen. Weitere Informationen finden Sie unter EC2 Image Builder Builder-Verteilungseinstellungen verwalten.

Registerkarte „Arbeitsablauf“

Workflows definieren die Reihenfolge der Schritte, die Image Builder beim Erstellen eines neuen Images ausführt. Für alle Images gibt es Workflows zum Erstellen und Testen. Container verfügen über einen zusätzlichen Workflow für die Verteilung. Auf der Registerkarte Workflow werden die entsprechenden Workflows angezeigt, die Image Builder für Ihr Image ausgeführt hat.

Workflowtypen filtern

Image Builder zeigt zunächst standardmäßig die Zusammenfassung des Build-Workflows und die Workflow-Schritte an. Der Workflow-Filter zeigt jedoch alle Workflows an, die für Ihr Image gerade ausgeführt oder abgeschlossen sind. Um einen anderen Workflow anzuzeigen, wählen Sie ihn wie folgt aus der Liste aus:

Image-Workflows (AMI-Ausgabe)

- `build-image`
- `test-image`

Container-Workflows (Container-Ausgabe)

- `build-container`
- `test-container`
- `distribute-container`

> **Note**
>
> Wenn der Workflow noch nicht gestartet wurde, erscheint er nicht in der Liste. Wenn Ihr Image-Build beispielsweise gerade gestartet wurde, `build-image` ist dies der einzige

Workflowtyp, der in der Liste angezeigt wird. Wenn der nächste Workflow beginnt, fügt Image Builder ihn `test-image` in diesem Fall der Liste hinzu.

Nach dem Workflow-Filter zeigt der ausgewählte Workflow eine Laufzeitübersicht an, die die folgenden Details für jeden Workflowtyp enthält:

Workflow-Status

Der aktuelle Laufzeitstatus für diesen Workflow. Die Werte können Folgendes beinhalten:

- Ausstehend
- Übersprungen
- In Ausführung
- Completed
- Fehlgeschlagen
- R ollback-in-progress
- Rollback abgeschlossen

Ausführungs-ID

Eine eindeutige Kennung, die Image Builder zuweist, um bei jeder Ausführung eines Workflows den Überblick über die Laufzeitressourcen zu behalten.

Starten

Der Zeitstempel, als die Runtime-Instanz dieses Workflows gestartet wurde.

Ende

Der Zeitstempel, zu dem diese Laufzeitinstanz des Workflows beendet wurde.

Schritte insgesamt

Die Gesamtzahl der Schritte im Workflow. Dies sollte der Summe der Schrittzahlen für erfolgreiche, übersprungene und fehlgeschlagene Schritte entsprechen.

Schritte waren erfolgreich

Eine Laufzeitanzahl für die Anzahl der Schritte im Workflow, die erfolgreich ausgeführt wurden.

Schritte sind fehlgeschlagen

Eine Laufzeitanzahl für die Anzahl der fehlgeschlagenen Schritte im Workflow.

Übersprungene Schritte

Eine Laufzeitanzahl für die Anzahl der Schritte im Workflow, die übersprungen wurden.

Die Details in der folgenden Liste geben den aktuellen Status für alle Schritte in dieser Laufzeitinstanz des Workflows an. Image Builder zeigt dieselben Details für alle Bildtypen an.

Schritt #

Eine Zahl, die die Reihenfolge angibt, in der Image Builder die Workflow-Schritte ausführt.

ID des Schritts

Eine eindeutige Kennung für den Workflow-Schritt, die zur Laufzeit zugewiesen wird.

Status des Schritts

Der aktuelle Laufzeitstatus des angegebenen Workflow-Schritts.

Rollback-Status

Der aktuelle Rollback-Status, falls diese Runtime-Instanz des Workflows fehlgeschlagen ist.

Name des Schritts

Der Name des angegebenen Workflow-Schritts.

Starten

Der Zeitstempel, zu dem der angegebene Schritt für diese Laufzeitinstanz des Workflows gestartet wurde.

Ende

Der Zeitstempel, zu dem der angegebene Schritt für diese Laufzeitinstanz des Workflows abgeschlossen wurde.

Registerkarte „Sicherheitsergebnisse“

Wenn Sie das Scannen aktiviert haben, werden auf der Registerkarte Sicherheitsergebnisse die Ergebnisse von Common Vulnerabilities and Exposures (CVE) angezeigt. Amazon Inspector hat diese Ergebnisse in der Test-Instance identifiziert, die Image Builder gestartet hat, um Ihr neues Image zu erstellen. Um sicherzustellen, dass Image Builder Ergebnisse für Ihr Bild erfasst, müssen Sie das Scannen wie folgt konfigurieren:

1. Aktivieren Sie Amazon Inspector-Scans für Ihr Konto. Weitere Informationen finden Sie unter Erste Schritte mit Amazon Inspector im Amazon Inspector Inspector-Benutzerhandbuch.
2. Aktivieren Sie die Sicherheitserkenntnisse für die Pipeline, die dieses Image erstellt. Wenn Sie Sicherheitsergebnisse für Ihre Pipeline aktivieren, speichert Image Builder eine Momentaufnahme der Ergebnisse, bevor die Testinstanz beendet wird. Weitere Informationen finden Sie unter Konfigurieren Sie Sicherheitsscans für Image Builder Builder-Images in der AWS Management Console.

Die Registerkarte Sicherheitsergebnisse enthält die folgenden Details für jede Sicherheitslücke, die Amazon Inspector für Ihr Bild identifiziert hat.

Schweregrad

Der Schweregrad des CVE-Befundes. Werte sind wie folgt:

- Untrifiziert
- Informativ
- Niedrig
- Medium
- Hoch
- Kritisch

Die ID des Ergebnisses

Die eindeutige Kennung für den CVE-Befund, den Amazon Inspector beim Scannen der Test-Instance für Ihr Bild erkannt hat. Die ID ist mit der Seite „Sicherheitsergebnisse“ > „Nach Sicherheitslücke“ verknüpft. Weitere Informationen finden Sie unter Verwalten Sie die Sicherheitsergebnisse für Image Builder Builder-Images in der AWS Management Console.

Quelle

Die Quelle der Informationen zur Sicherheitslücke für die CVE-Entdeckung.

Alter

Die Anzahl der Tage, seit der Befund zum ersten Mal für Ihr Bild beobachtet wurde.

Ergebnis des Inspector

Die Punktzahl, die Amazon Inspector dem CVE-Ergebnis zugewiesen hat.

Registerkarte „Tags“

Auf der Registerkarte „Tags“ werden alle Tags angezeigt, die Sie für Ihr Bild definiert haben.

Details zur Bild-Richtlinie abrufen (AWS CLI)

Das folgende Beispiel zeigt, wie Sie die Details einer Image-Richtlinie mit ihrem Amazon-Ressourcennamen (ARN) abrufen.

```
aws imagebuilder get-image-policy --image-arn arn:aws:imagebuilder:us-
west-2:123456789012:image/example-image/2019.12.02
```

Bilder erstellen

In diesem Abschnitt erfahren Sie, wie Sie Image Builder Builder-Images erstellen und einen laufenden Build abbrechen.

Inhalt

Erstellen eines Images

Es gibt verschiedene Möglichkeiten, ein neues Image Builder zu erstellen. Sie können beispielsweise eine der folgenden Methoden verwenden, um ein Bild mit dem AWS Management Console oder zu erstellen AWS CLI. Sie können auch die CreateImageAPI-Aktion verwenden. Die zugehörige SDK-Anfrage finden Sie unter dem Link „Siehe auch“ für diesen Befehl in der EC2 Image Builder API-Referenz.

AWS Management Console

Um ein neues Image aus einer vorhandenen Pipeline zu erstellen, können Sie die Pipeline wie folgt manuell ausführen. Sie können auch den Pipeline-Assistenten verwenden, um ein neues Image von Grund auf neu zu erstellen. Je nach Art des BildsErstellen Sie eine Image-Pipeline (Docker), das Sie erstellen möchten, finden Sie unter Erstellen Sie eine Image-Pipeline (AMI) oder.

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.

2. Wählen Sie im Navigationsbereich die Option Image-Pipelines aus.
3. Aktivieren Sie das Kontrollkästchen neben dem Namen der Pipeline, die Sie ausführen möchten.
4. Um das Image zu erstellen, wählen Sie im Menü Aktionen die Option Pipeline ausführen aus. Dadurch wird die Pipeline gestartet.

Sie können auch einen Zeitplan für den Betrieb Ihrer Pipeline angeben oder Amazon verwenden, EventBridge um Ihre Pipeline auf der Grundlage der von Ihnen konfigurierten Regeln auszuführen.

AWS CLI

Bevor Sie den create-image Befehl in der ausführen AWS CLI, müssen Sie die folgenden Ressourcen erstellen, sofern sie noch nicht vorhanden sind:

Erforderliche -Ressourcen

- Rezept — Sie müssen genau ein Rezept für Ihr Bild angeben, und zwar wie folgt:

  Bildrezept

  Geben Sie mit dem `--image-recipe-arn` Parameter den Amazon-Ressourcennamen (ARN) für Ihre Bildrezeptressource an.

  Rezept für Container

  Geben Sie den ARN für Ihre Container-Rezeptressource mit dem `--container-recipe-arn` Parameter an.
- Infrastrukturkonfiguration — Geben Sie den ARN für Ihre Infrastrukturkonfigurationsressource mit dem `--infrastructure-configuration-arn` Parameter an.

Sie können auch jede der folgenden Ressourcen angeben, die Ihr Image benötigt:

Optionale Ressourcen und Konfiguration

- Verteilungskonfiguration — Standardmäßig verteilt Image Builder die Ausgabe-Image-Ressource an Ihr Konto in der Region, in der Sie den create-image Befehl ausführen. Um zusätzliche Ziele oder Konfigurationen für Ihre Distribution bereitzustellen, geben Sie den ARN für Ihre Distributionskonfigurationsressource mit dem `--distribution-configuration-arn` Parameter an.

- Scannen von Bildern — Verwenden Sie den `--image-scanning-configuration` Parameter, um Schnappschüsse für Amazon Inspector Inspector-Ergebnisse auf Ihrer Image- oder Container-Testinstanz zu konfigurieren. Für Container-Images geben Sie auch das ECR-Repository an, das Amazon Inspector für seine Scans verwendet.
- Image-Tests — Verwenden Sie den `--image-tests-configuration` Parameter, um die Image Builder Builder-Testphase zu unterdrücken. Alternativ können Sie ein Timeout für die Dauer der Ausführung festlegen.
- Bild-Tags — Verwenden Sie den `--tags` Parameter, um Ihrem Ausgabebild Tags hinzuzufügen.
- Image-Workflows — Wenn Sie keine Build- oder Test-Workflows angeben, erstellt Image Builder Ihr Image mit seinem Standard-Image-Workflow. Verwenden Sie den `--workflows` Parameter, um die von Ihnen erstellten Workflows anzugeben.

  > **Note**
  >
  > Wenn Sie Image-Workflows angeben, müssen Sie im `--execution-role` Parameter auch den Namen oder ARN der IAM-Rolle angeben, die Image Builder zur Ausführung Ihrer Workflow-Aktionen verwendet.

Das folgende Beispiel zeigt, wie Sie mit dem Befehl AWS CLI create-image ein Image erstellen. Weitere Informationen finden Sie in der AWS CLI -Befehlsreferenz.

Beispiel: Erstellen Sie ein einfaches Image mit Standardverteilung

```
aws imagebuilder create-image --image-recipe-arn arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/simple-recipe-linux/1.0.0 --infrastructure-configuration-arn arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/simple-infra-config-linux
```

Ausgabe:

```
{
    "requestId": "1abcd234-e567-8fa9-0123-4567b890cd12",
    "imageVersionList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/simple-recipe-linux/1.0.0",
            "name": "simple-recipe-linux",
```

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

Image-Erstellung abbrechen (AWS CLI)

Um einen laufenden Image-Build abzubrechen, verwenden Sie den cancel-image-creation Befehl wie folgt:

```
aws imagebuilder cancel-image-creation --image-build-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-recipe/2019.12.03/1
```

Ein VM-Image importieren

Image Builder ist in die Amazon EC2 VM Import/Export API integriert, sodass der Importvorgang asynchron im Hintergrund ausgeführt werden kann. Image Builder verweist auf die Aufgaben-ID aus dem VM-Import, um dessen Fortschritt zu verfolgen, und erstellt eine Image Builder Builder-Image-Ressource als Ausgabe. Auf diese Weise können Sie in Ihren Rezepten auf die Image Builder Builder-Image-Ressource verweisen, bevor der VM-Import abgeschlossen ist.

Importieren Sie eine VM (Konsole)

Gehen Sie folgendermaßen vor, um eine VM mit der Image Builder Builder-Konsole zu importieren:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Images aus.
3. Klicken Sie auf Packet importieren.
4. Geben Sie auf der Seite Bild importieren Einzelheiten zu den folgenden Abschnitten an. Wählen Sie dann Bild importieren, wenn Sie fertig sind.

Allgemeines

1. Geben Sie einen eindeutigen Namen für das Basis-Image an.
2. Geben Sie eine Version für das Basis-Image an. Verwenden Sie das folgende Format: *major.minor.patch*.
3. Sie können auch eine optionale Beschreibung für das Basis-Image eingeben.

Betriebssystem für das Basis-Image

1. Wählen Sie die Option Image-Betriebssystem (OS) aus, die Ihrer VM-OS-Plattform entspricht.
2. Wählen Sie aus der Liste die Betriebssystemversion aus, die der Version für Ihre VM entspricht.

Konfiguration für den VM-Import

Wenn Sie Ihre VM aus ihrer Virtualisierungsumgebung exportieren, erstellt dieser Prozess einen Satz von einer oder mehreren Festplattencontainerdateien. Diese dienen als Snapshots der Umgebung, der Einstellungen und Daten Ihrer VM. Sie können diese Dateien verwenden, um Ihre VM als Basisimage für Ihr Image-Rezept zu importieren. Weitere Informationen zum Importieren von VMs in Image Builder finden Sie unterVM-Images importieren und exportieren.

Gehen Sie folgendermaßen vor, um den Speicherort Ihrer Importquelle anzugeben:

Quelle importieren

Geben Sie die Quelle für den ersten VM-Image-Festplattencontainer oder Snapshot, der importiert werden soll, im Abschnitt Festplattencontainer 1 an.

1. Quelle — Dies kann entweder ein S3-Bucket oder ein EBS-Snapshot sein.
2. Wählen Sie den S3-Speicherort der Festplatte — Geben Sie den Speicherort in Amazon S3 ein, an dem Ihre Festplatten-Images gespeichert sind. Um nach dem Speicherort zu suchen, wählen Sie Browse S3.
3. Um einen Festplattencontainer hinzuzufügen, wählen Sie Festplattencontainer hinzufügen.

IAM-Rolle

Um Ihrer VM-Importkonfiguration eine IAM-Rolle zuzuordnen, wählen Sie die Rolle aus der Dropdownliste der IAM-Rolle aus, oder wählen Sie Neue Rolle erstellen, um eine neue Rolle zu erstellen. Wenn Sie eine neue Rolle erstellen, wird die Konsolenseite für IAM-Rollen auf einer separaten Registerkarte geöffnet.

Erweiterte Einstellungen — optional

Die folgenden Einstellungen sind optional. Mit diesen Einstellungen können Sie Verschlüsselung, Lizenzierung, Tags und mehr für das Basis-Image konfigurieren, das durch den Import erstellt wird.

Architektur des Basis-Images

Um die Architektur Ihrer VM-Importquelle anzugeben, wählen Sie einen Wert aus der Architekturliste aus.

Verschlüsselung

Wenn Ihre VM-Disk-Images verschlüsselt sind, müssen Sie einen Schlüssel angeben, den Sie für den Importvorgang verwenden können. Um einen KMS-Schlüssel für den Import anzugeben, wählen Sie einen Wert aus der Liste Verschlüsselung (KMS-Schlüssel) aus. Die Liste enthält KMS-Schlüssel, auf die Ihr Konto in der aktuellen Region Zugriff hat.

Lizenzverwaltung

Wenn Sie eine VM importieren, erkennt der Importvorgang automatisch das VM-Betriebssystem und wendet die entsprechende Lizenz auf das Basis-Image an. Abhängig von Ihrer Betriebssystemplattform lauten die Lizenztypen wie folgt:

- Lizenz enthalten — Eine entsprechende AWS Lizenz für Ihre Plattform wird auf Ihr Basis-Image angewendet.
- Bringen Sie Ihre eigene Lizenz mit (BYOL) — Behält die Lizenz von Ihrer VM bei, falls zutreffend.

Um Lizenzkonfigurationen, die mit erstellt wurden, AWS License Manager an Ihr Basis-Image anzuhängen, wählen Sie aus der Liste mit den Namen der Lizenzkonfiguration aus. Weitere Informationen zu License Manager finden Sie unter Arbeiten mit AWS License Manager

> **Note**
>
> - Lizenzkonfigurationen enthalten Lizenzregeln, die auf den Bedingungen Ihrer Unternehmensvereinbarungen basieren.
> - Linux unterstützt nur BYOL-Lizenzen.

Tags (Basisbild)

Tags verwenden Schlüssel-Wert-Paare, um Ihrer Image Builder Builder-Ressource durchsuchbaren Text zuzuweisen. Um Tags für das importierte Basisbild anzugeben, geben Sie Schlüssel-Wert-Paare in die Felder Schlüssel und Wert ein.

Um einen Tag hinzuzufügen, wählen Sie Add tag (Tag hinzufügen). Klicken Sie zum Entfernen eines Tags auf Tag entfernen.

Importieren Sie eine VM ()AWS CLI

Um eine VM von Festplatten in ein AMI zu importieren und eine Image Builder Builder-Image-Ressource zu erstellen, auf die Sie sofort verweisen können, gehen Sie wie folgt vor AWS CLI:

1. Initiieren Sie einen VM-Import mit dem import-image Befehl Amazon EC2 VM Import/Export in der. AWS CLI Notieren Sie sich die Aufgaben-ID, die in der Befehlsantwort zurückgegeben wird. Sie benötigen sie für den nächsten Schritt. Weitere Informationen finden Sie unter Importieren einer VM als Image mithilfe von VM Import/Export im VM Import/Export-Benutzerhandbuch.
2. Erstellen einer CLI-Eingabe-JSON-Datei

   Um den in der verwendeten Image Builder import-vm-image Builder-Befehl zu optimieren AWS CLI, erstellen wir eine JSON-Datei, die die gesamte Importkonfiguration enthält, die wir an den Befehl übergeben möchten.

   > **Note**
   >
   > Die Benennungskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Aktionsanforderungsparameter der Image Builder Builder-API angegeben ist. Informationen zu den API-Befehlsanforderungsparametern finden Sie unter dem ImportVmImageBefehl in der EC2 Image Builder API-Referenz.
   > Um die Datenwerte als Befehlszeilenparameter bereitzustellen, beziehen Sie sich auf die Parameternamen, die in der AWS CLI Befehlsreferenz angegeben sind. Auf den Image Builder import-vm-image Builder-Befehl als Optionen.

   Im Folgenden finden Sie eine Zusammenfassung der Parameter, die wir in diesem Beispiel angeben:

   - name (string, erforderlich) — Der Name für die Image Builder Builder-Bildressource, die als Ausgabe aus dem Import erstellt werden soll.
   - semanticVersion <major>(string, erforderlich) — Die semantische Version für das Ausgabebild, die die Version im folgenden Format angibt, mit numerischen Werten an jeder Position, um eine bestimmte Version anzugeben:. <minor>. <patch>. z. B. `1.0.0`. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung
   - description (string) — Die Beschreibung des Image-Rezepts.

- platform (string, erforderlich) — Die Betriebssystemplattform für die importierte VM.
- vmImportTaskId (string, required) — Das `ImportTaskId` (AWS CLI) aus dem Amazon EC2 EC2-VM-Importprozess. Image Builder überwacht den Importvorgang, um das von ihm erstellte AMI abzurufen und eine Image Builder Builder-Image-Ressource zu erstellen, die sofort in Rezepten verwendet werden kann.
- clientToken (string, required) — Eine eindeutige Kennung, bei der Groß- und Kleinschreibung berücksichtigt wird, die Sie angeben, um die Idempotenz der Anfrage sicherzustellen. Weitere Informationen finden Sie unter Ensuring Idempotenz in der Amazon EC2 API-Referenz.
- tags (String-Map) — Tags sind Schlüssel-Wert-Paare, die an die Importressourcen angehängt werden. Bis zu 50 Schlüssel-Wert-Paare sind zulässig.

Speichern Sie die Datei unter`import-vm-image.json`, um sie im Image Builder import-vm-image Builder-Befehl zu verwenden.

```
{
    "name": "example-request",
    "semanticVersion": "1.0.0",
    "description": "vm-import-test",
    "platform": "Linux",
    "vmImportTaskId": "import-ami-01ab234567890cd1e",
    "clientToken": "asz1231231234cs3z",
    "tags": {
     "Usage": "VMIE"
    }
}
```

3. Importiert das Bild

Führen Sie den import-vm-image Befehl mit der Datei aus, die Sie als Eingabe erstellt haben:

```
aws imagebuilder import-vm-image --cli-input-json file://import-vm-image.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet

beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Sicherheitsergebnisse für Image Builder Builder-Images verwalten

Wenn Sie Sicherheitsscans mit Amazon Inspector aktivieren, scannt Amazon Inspector kontinuierlich Computerbilder und laufende Instances in Ihrem Konto auf Sicherheitslücken in Betriebssystem und Programmiersprachen. Wenn aktiviert, erfolgt der Sicherheitsscan automatisch, und Image Builder kann eine Momentaufnahme der Ergebnisse Ihrer Testinstanz speichern, wenn Sie ein neues Image erstellen. Amazon Inspector ist ein kostenpflichtiger Service.

Wenn Amazon Inspector Sicherheitslücken in Ihrer Software oder Ihren Netzwerkeinstellungen entdeckt, ergreift Amazon Inspector die folgenden Maßnahmen:

- Informiert Sie darüber, dass ein Befund gefunden wurde.
- Bewertet den Schweregrad des Befundes. Bei der Bewertung des Schweregrads werden Sicherheitslücken kategorisiert, um Ihnen bei der Priorisierung Ihrer Ergebnisse zu helfen. Sie umfasst die folgenden Werte:
  - Ungeprüft
  - Informativ
  - Niedrig
  - Medium
  - Hoch
  - Kritisch
- Enthält Informationen zu den Ergebnissen und Links zu weiteren Ressourcen für weitere Informationen.
- Bietet Anleitungen zur Problembehebung, die Ihnen bei der Behebung der Probleme helfen sollen, die zu dem Ergebnis geführt haben.

Konfigurieren Sie Sicherheitsscans für Image Builder Builder-Images in der AWS Management Console

Wenn Sie Amazon Inspector für Ihr Konto aktiviert haben, scannt Amazon Inspector automatisch die EC2-Instances, die Image Builder startet, um ein neues Image zu erstellen und zu testen.

Diese Instances haben während des Build- und Testprozesses eine kurze Lebensdauer, und ihre Ergebnisse laufen normalerweise ab, sobald diese Instances heruntergefahren werden. Um Sie bei der Untersuchung und Behebung der Ergebnisse für Ihr neues Image zu unterstützen, kann Image Builder optional alle Ergebnisse, die Amazon Inspector während des Erstellungsprozesses auf Ihrer Test-Instance identifiziert hat, als Snapshot speichern.

Schritt 1: Aktivieren Sie die Amazon Inspector-Sicherheitsscans für Ihr Konto

Gehen Sie wie folgt vor, um Amazon Inspector-Sicherheitsscans für Ihr Konto von der Image Builder Builder-Konsole aus zu aktivieren:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich die Option Einstellungen für Sicherheitsscans aus. Dadurch wird das Dialogfeld Sicherheitsscan geöffnet.

   Das Dialogfeld zeigt den Scanstatus für Ihr Konto an. Wenn Amazon Inspector bereits für Ihr Konto aktiviert ist, wird der Status Aktiviert angezeigt.
3. Folgen Sie den Schritten 1 und 2 der Anweisungen, um den Amazon Inspector-Scan zu aktivieren.

> **Note**
>
> Für Amazon Inspector fallen Gebühren an. Weitere Informationen erhalten Sie unter Amazon Inspector: Preise.

Wenn Sie das Scannen für Ihre Pipeline aktiviert haben, erstellt Image Builder beim Erstellen eines neuen Images eine Momentaufnahme der Ergebnisse für Ihre Build-Instanz. Auf diese Weise können Sie auf die Ergebnisse zugreifen, nachdem Image Builder die Build-Instanz beendet hat.

Schritt 2: Konfigurieren Sie Ihre Pipeline so, dass Snapshots für gefundene Sicherheitslücken gespeichert werden

Gehen Sie wie folgt vor, um Snapshots zur Suche nach Sicherheitslücken für Ihre Pipeline zu konfigurieren:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Image-Pipelines aus.
3. Wählen Sie eine der folgenden Methoden, um Pipeline-Details anzugeben:

Erstellen Sie eine neue Pipeline

1. Wählen Sie auf der Seite Image-Pipelines die Option Image-Pipeline erstellen aus. Dadurch wird die Seite „Pipeline-Details angeben“ im Pipeline-Assistenten geöffnet.

Aktualisieren Sie eine bestehende Pipeline

1. Wählen Sie auf der Seite Image-Pipelines den Link Pipeline-Namen für die Pipeline aus, die Sie aktualisieren möchten. Dadurch wird die Pipeline-Detailseite geöffnet.

> **Note**
>
> Sie können auch das Kontrollkästchen neben dem Namen der Pipeline aktivieren, die Sie aktualisieren möchten, und dann Details anzeigen auswählen.

2. Wählen Sie auf der Seite mit den Pipeline-Details im Aktionsmenü die Option Pipeline bearbeiten aus. Dadurch gelangen Sie zur Seite „Pipeline bearbeiten“.

4. Aktivieren Sie im Abschnitt Allgemein des Pipeline-Assistenten oder auf der Seite Pipeline bearbeiten das Kontrollkästchen Sicherheitsscan aktivieren.

> **Note**
>
> Wenn Sie die Snapshots später deaktivieren möchten, können Sie Ihre Pipeline bearbeiten, um das Kontrollkästchen zu deaktivieren. Dadurch wird das Scannen von Amazon Inspector für Ihr Konto nicht deaktiviert. Informationen zur Deaktivierung von Amazon Inspector-Scans finden Sie unter Deaktivierung von Amazon Inspector im Amazon Inspector Inspector-Benutzerhandbuch.

Verwalten Sie die Sicherheitsergebnisse für Image Builder Builder-Images in der AWS Management Console

Auf den Seiten mit der Liste der Sicherheitsergebnisse werden allgemeine Informationen zu den Ergebnissen Ihrer Ressourcen angezeigt. Die Ansichten basieren auf verschiedenen Filtern, die Sie anwenden können. Jede Ansicht enthält oben die folgenden Optionen, mit denen Sie Ihre Ansicht ändern können:

- Alle Sicherheitsergebnisse — Dies ist die Standardansicht, wenn Sie im Navigationsbereich der Image Builder Builder-Konsole die Seite mit den Sicherheitsergebnissen auswählen.
- Nach Sicherheitslücken sortiert — In dieser Ansicht wird eine allgemeine Liste aller Image-Ressourcen in Ihrem Konto angezeigt, für die Sicherheitslücken vorliegen. Die Fund-ID ist mit detaillierteren Informationen über das Ergebnis verknüpft. Diese Informationen werden in einem Fenster angezeigt, das sich auf der rechten Seite der Seite öffnet. Das Panel enthält die folgenden Informationen:
  - Eine detaillierte Beschreibung des Ergebnisses.
  - Eine Registerkarte mit den Details zu den Ergebnissen. Diese Registerkarte enthält eine Übersicht über die Ergebnisse, die betroffenen Pakete, zusammenfassende Hinweise zur Problembehebung, Details zu Sicherheitslücken und zugehörige Sicherheitslücken. Die Schwachstellen-ID enthält Links zu detaillierten Informationen zu Sicherheitslücken in der National Vulnerability Database.
  - Eine Registerkarte mit einer Aufschlüsselung der Ergebnisse. Diese Registerkarte enthält einen side-by-side Vergleich der CVSS- und Amazon Inspector-Scores, sodass Sie sehen können, wo Amazon Inspector gegebenenfalls eine Bewertung geändert hat.
- Nach Image-Pipeline — In dieser Ansicht wird die Anzahl der Ergebnisse für jede Image-Pipeline in Ihrem Konto angezeigt. Image Builder zeigt Zählungen für Ergebnisse mit mittlerem Schweregrad und höherem Schweregrad sowie die Gesamtsumme für alle Ergebnisse an. Alle Daten in der Liste sind wie folgt verknüpft:
  - Die Spalte mit dem Namen der Image-Pipeline ist mit der Detailseite für die angegebene Image-Pipeline verknüpft.
  - Die Links in der Spalte „Schweregrad“ öffnen die Ansicht „Alle Sicherheitsergebnisse“, gefiltert nach dem Namen und Schweregrad der zugehörigen Image-Pipeline.

  Sie können Ihre Ergebnisse auch anhand von Suchkriterien verfeinern.
- Nach Bild — In dieser Ansicht wird die Anzahl der Ergebnisse für jedes in Ihrem Konto erstellte Image angezeigt. Image Builder zeigt Zählungen für Ergebnisse mit mittlerem Schweregrad und höherem Schweregrad sowie die Gesamtsumme für alle Ergebnisse an. Alle Daten in der Liste sind wie folgt verknüpft:
  - Die Spalte mit dem Imagenamen ist mit der Image-Detailseite für den angegebenen Image-Build verknüpft. Weitere Informationen finden Sie unter Bilddetails anzeigen.
  - Über die Links in der Spalte „Schweregrad“ wird die Ansicht „Alle Sicherheitsergebnisse“ geöffnet, die nach dem Namen und dem Schweregrad des zugehörigen Image-Builds gefiltert ist.

Sie können Ihre Ergebnisse auch anhand von Suchkriterien verfeinern.

Image Builder zeigt die folgenden Details im Abschnitt Ergebnisliste der Standardansicht Alle Sicherheitsergebnisse an.

Schweregrad

Der Schweregrad des CVE-Ergebnisses. Werte sind wie folgt:

- Untrifiziert
- Informativ
- Niedrig
- Medium
- Hoch
- Kritisch

Die ID des Ergebnisses

Die eindeutige Kennung für den CVE-Befund, den Amazon Inspector beim Scannen der Build-Instance für Ihr Bild erkannt hat. Die ID ist mit der Seite „Sicherheitsergebnisse“ **>** „Nach Sicherheitslücke“ verknüpft.

Bild ARN

Der Amazon-Ressourcenname (ARN) für das Bild mit dem in der Spalte Finding ID angegebenen Ergebnis.

Pipeline

Die Pipeline, die das in der Spalte Image ARN angegebene Image erstellt hat.

Beschreibung

Eine kurze Beschreibung des Ergebnisses.

Ergebnis des Inspector

Die Punktzahl, die Amazon Inspector dem CVE-Ergebnis zugewiesen hat.

Behebung

Links zu Einzelheiten zur empfohlenen Vorgehensweise zur Behebung des Fehlers.

Datum der Veröffentlichung

Datum und Uhrzeit, an dem diese Sicherheitsanfälligkeit zum ersten Mal zur Datenbank des Anbieters hinzugefügt wurde.

Bereinigen von -Ressourcen

Um unerwartete Gebühren zu vermeiden, sollten Sie die Ressourcen und Pipelines, die Sie anhand der Beispiele in diesem Handbuch erstellt haben, bereinigen. Weitere Informationen zum Löschen von Ressourcen in Image Builder finden Sie unterEC2 Image Builder Builder-Ressourcen löschen.

EC2 Image Builder Builder-Infrastrukturkonfiguration verwalten

Sie können Infrastrukturkonfigurationen verwenden, um die Amazon EC2 EC2-Infrastruktur anzugeben, die Image Builder zum Erstellen und Testen Ihres EC2 Image Builder Builder-Images verwendet. Zu den Infrastruktureinstellungen gehören:

- Instanztypen für Ihre Build- und Testinfrastruktur. Es wird empfohlen, mehr als einen Instanztyp anzugeben, da Image Builder dadurch eine Instanz aus einem Pool mit ausreichender Kapazität starten kann. Dies kann Ihre vorübergehenden Build-Fehler reduzieren.
- Ein Instanzprofil, das Ihren Build- und Test-Instances die Berechtigungen gewährt, die für die Durchführung von Anpassungsaktivitäten erforderlich sind. Wenn Sie beispielsweise über eine Komponente verfügen, die Ressourcen von Amazon S3 abruft, benötigt das Instance-Profil Berechtigungen für den Zugriff auf diese Dateien. Das Instanzprofil erfordert außerdem einen Mindestsatz an Berechtigungen, damit EC2 Image Builder erfolgreich mit der Instance kommunizieren kann. Weitere Informationen finden Sie unter Voraussetzungen.
- Die VPC, das Subnetz und die Sicherheitsgruppen für die Build- und Test-Instances Ihrer Pipeline.
- Der Amazon S3 S3-Speicherort, an dem Image Builder Anwendungsprotokolle von Ihrem Build und Ihren Tests speichert. Wenn Sie die Protokollierung konfigurieren, muss das in Ihrer Infrastrukturkonfiguration angegebene Instance-Profil über `s3:PutObject` Berechtigungen für den Ziel-Bucket (`arn:aws:s3:::BucketName/*`) verfügen.
- Ein Amazon EC2 EC2-Schlüsselpaar, mit dem Sie sich bei Ihrer Instance anmelden können, um Fehler zu beheben, falls Ihr Build fehlschlägt und Sie `terminateInstanceOnFailure` auf `false` einstellen.

- Ein SNS-Thema, bei dem Image Builder Ereignisbenachrichtigungen sendet. Weitere Informationen zur Integration von Image Builder in Amazon SNS finden Sie unterAmazon SNS SNS-Integration in Image Builder.

> **Note**
>
> Wenn Ihr SNS-Thema verschlüsselt ist, muss sich der Schlüssel, der dieses Thema verschlüsselt, in dem Konto befinden, auf dem der Image Builder Builder-Dienst ausgeführt wird. Image Builder kann keine Benachrichtigungen an SNS-Themen senden, die mit Schlüsseln anderer Konten verschlüsselt sind.

Sie können Infrastrukturkonfigurationen mit der Image Builder Builder-Konsole, über die Image Builder Builder-API oder mit imagebuilder Befehlen in der erstellen und verwalten AWS CLI.

Inhalt

> **Tip**
>
> Wenn Sie über mehrere Ressourcen desselben Typs verfügen, hilft Ihnen das Tagging dabei, eine bestimmte Ressource anhand der Tags zu identifizieren, die Sie ihr zugewiesen haben. Weitere Informationen zum Taggen Ihrer Ressourcen mithilfe von Image Builder Builder-Befehlen finden Sie im AWS CLIMarkieren von Ressourcen Abschnitt dieses Handbuchs.

Details zur Infrastrukturkonfiguration auflisten und anzeigen

In diesem Abschnitt werden die verschiedenen Möglichkeiten beschrieben, wie Sie Informationen finden und Details zu Ihren EC2 Image Builder Builder-Infrastrukturkonfigurationen anzeigen können.

Einzelheiten zur Infrastrukturkonfiguration

- Listet die Infrastrukturkonfigurationen auf (AWS CLI)

- Rufen Sie Details zur Infrastrukturkonfiguration ab (AWS CLI)

Listet die Infrastrukturkonfigurationen auf (AWS CLI)

Das folgende Beispiel zeigt, wie Sie mit dem list-infrastructure-configurations Befehl in der alle Ihre Infrastrukturkonfigurationen auflisten können AWS CLI.

```
aws imagebuilder list-infrastructure-configurations
```

Rufen Sie Details zur Infrastrukturkonfiguration ab (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den get-infrastructure-configuration Befehl in verwenden AWS CLI , um die Details einer Infrastrukturkonfiguration abzurufen, indem Sie ihren Amazon-Ressourcennamen (ARN) angeben.

```
aws imagebuilder get-infrastructure-configuration --infrastructure-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-
infrastructure-configuration
```

Erstellen einer Infrastrukturkonfiguration

In diesem Abschnitt wird beschrieben, wie Sie die Image Builder Builder-Konsole oder imagebuilder Befehle in der verwenden können, AWS CLI um eine Infrastrukturkonfiguration zu erstellen.

Console

Gehen Sie folgendermaßen vor, um eine Infrastrukturkonfigurationsressource von der Image Builder Builder-Konsole aus zu erstellen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich die Option Infrastrukturkonfiguration aus.
3. Wählen Sie Infrastrukturkonfiguration erstellen aus.
4. Geben Sie im Abschnitt Allgemein die folgenden erforderlichen Informationen ein:
   - Geben Sie den Namen Ihrer Infrastrukturkonfigurationsressource ein.
   - Wählen Sie eine IAM-Rolle aus, die Sie dem Instanzprofil für Komponentenberechtigungen für Ihre Build- und Test-Instances zuordnen möchten. Image Builder verwendet diese

Berechtigungen, um Ihre Komponenten herunterzuladen und auszuführen, Protokolle hochzuladen und alle zusätzlichen Aktionen auszuführen CloudWatch, die in den Komponenten in Ihrem Rezept angegeben sind.

5. Im AWS Infrastrukturfenster können Sie die übrigen verfügbaren Infrastruktureinstellungen konfigurieren. Geben Sie die folgenden erforderlichen Informationen ein:

   - Instanztyp — Sie können einen oder mehrere Instance-Typen angeben, die für diesen Build verwendet werden sollen. Der Service wählt je nach Verfügbarkeit einen dieser Instanztypen aus.
   - SNS-Thema (optional) — Wählen Sie ein SNS-Thema aus, um Benachrichtigungen und Warnungen von EC2 Image Builder zu erhalten.

   Wenn Sie keine Werte für die folgenden Einstellungen angeben, verwenden sie gegebenenfalls dienstspezifische Standardwerte.

   - VPC, Subnetz und Sicherheitsgruppen — Image Builder verwendet Ihre Standard-VPC und Ihr Standard-Subnetz. Weitere Informationen zur Konfiguration von VPC-Schnittstellenendpunkten finden Sie unter. EC2 Image Builder und VPC-Schnittstellen-Endpunkte ()AWS PrivateLink
   - Im Abschnitt Einstellungen zur Fehlerbehebung können Sie die folgenden Werte konfigurieren:
     - Standardmäßig ist das Kontrollkästchen Instanz bei Ausfall beenden aktiviert. Wenn jedoch ein Build fehlschlägt, können Sie sich zur Fehlerbehebung bei der EC2-Instance anmelden. Wenn Sie möchten, dass Ihre Instance nach einem Build-Fehler weiter ausgeführt wird, deaktivieren Sie das Kontrollkästchen.
     - key pair — Wenn Ihre EC2-Instance nach einem Build-Fehler weiter läuft, können Sie ein key pair erstellen oder ein vorhandenes Schlüsselpaar verwenden, um sich bei der Instance anzumelden und Fehler zu beheben.
     - Logs — Sie können einen S3-Bucket angeben, in den Image Builder Anwendungsprotokolle schreiben kann, um Sie bei der Fehlerbehebung bei Ihrem Build und Ihren Tests zu unterstützen. Wenn Sie keinen S3-Bucket angeben, schreibt Image Builder die Anwendungsprotokolle in die Instanz.
   - Im Abschnitt Einstellungen für Instanz-Metadaten können Sie die folgenden Werte so konfigurieren, dass sie auf die EC2-Instances angewendet werden, die Image Builder zum Erstellen und Testen Ihres Images verwendet:

- Wählen Sie die Metadaten-Version aus, um festzustellen, ob EC2 einen signierten Token-Header für Anfragen zum Abrufen von Instance-Metadaten benötigt.
  - V1 und V2 (Token optional) — Standardwert, wenn Sie nichts auswählen.
  - V2 (Token erforderlich)

  > **Note**
  >
  > Wir empfehlen, dass Sie alle EC2-Instances, die Image Builder aus einem Pipeline-Build startet, für die Verwendung von IMDSv2 konfigurieren, sodass für Anfragen zum Abrufen von Instance-Metadaten ein signierter Token-Header erforderlich ist.

- Antwort-Hop-Limit für Metadatentokens — Die Anzahl der Netzwerk-Hops, die das Metadatentoken zurücklegen kann. Minimale Hops: 1, maximale Anzahl Hops: 64, mit einem Standardwert von einem Hop.

6. Im Abschnitt Infrastruktur-Tags (optional) können Sie der Amazon EC2 EC2-Instance, die Image Builder während des Build-Prozesses startet, Metadaten-Tags zuweisen. Tags werden als Schlüsselwertpaare eingegeben.
7. Im Abschnitt Tags (optional) können Sie der Infrastrukturkonfigurationsressource, die Image Builder als Ausgabe erstellt, Metadaten-Tags zuweisen. Tags werden als Schlüssel-Wert-Paare eingegeben.

AWS CLI

Das folgende Beispiel zeigt, wie Sie die Infrastruktur für Ihr Image mit dem Image Builder create-infrastructure-configuration Builder-Befehl in der konfigurieren AWS CLI.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   In diesem Beispiel für eine Infrastrukturkonfiguration werden zwei Instanztypen angegeben, `m5.large` und`m5.xlarge`. Es wird empfohlen, mehr als einen Instanztyp anzugeben, da Image Builder dadurch eine Instanz aus einem Pool mit ausreichender Kapazität starten kann. Dies kann Ihre vorübergehenden Build-Fehler reduzieren.

   Das `instanceProfileName` gibt das Instanzprofil an, das der Instanz die Berechtigungen gewährt, die das Profil für die Durchführung von Anpassungsaktivitäten benötigt. Wenn Sie beispielsweise über eine Komponente verfügen, die Ressourcen von Amazon S3

abruft, benötigt das Instance-Profil Berechtigungen für den Zugriff auf diese Dateien. Das Instanzprofil erfordert außerdem einen Mindestsatz an Berechtigungen, damit EC2 Image Builder erfolgreich mit der Instance kommunizieren kann. Weitere Informationen finden Sie unter Voraussetzungen.

Verwenden Sie ein Dateibearbeitungstool, um eine JSON-Datei mit den im folgenden Beispiel gezeigten Schlüsseln sowie Werten, die für Ihre Umgebung gültig sind, zu erstellen. In diesem Beispiel wird eine Datei mit dem Namen`create-infrastructure-configuration.json`:

```
{
    "name": "MyExampleInfrastructure",
    "description": "An example that will retain instances of failed builds",
    "instanceTypes": [
        "m5.large", "m5.xlarge"
    ],
    "instanceProfileName": "myIAMInstanceProfileName",
    "securityGroupIds": [
        "sg-12345678"
    ],
    "subnetId": "sub-12345678",
    "logging": {
        "s3Logs": {
            "s3BucketName": "my-logging-bucket",
            "s3KeyPrefix": "my-path"
        }
    },
    "keyPair": "myKeyPairName",
    "terminateInstanceOnFailure": false,
    "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:MyTopic"
}
```

2. Verwenden Sie die Datei, die Sie erstellt haben, als Eingabe, wenn Sie den folgenden Befehl ausführen.

```
aws imagebuilder create-infrastructure-configuration --cli-input-json
 file://create-infrastructure-configuration.json
```

Aktualisieren Sie eine Infrastrukturkonfiguration

In diesem Abschnitt wird beschrieben, wie Sie die Image Builder Builder-Konsole oder imagebuilder Befehle in der Ressource AWS CLI zum Aktualisieren einer Infrastrukturkonfiguration verwenden können. Um Ihre Ressourcen nachzuverfolgen, können Sie Tags wie folgt anwenden. Tags werden als Schlüsselwertpaare eingegeben.

- Ressourcen-Tags weisen der Amazon EC2 EC2-Instance, die Image Builder während des Build-Prozesses startet, Metadaten-Tags zu.
- Tags weisen der Infrastrukturkonfigurationsressource, die Image Builder als Ausgabe erstellt, Metadaten-Tags zu.

Console

Sie können die folgenden Infrastrukturkonfigurationsdetails von der Image Builder Builder-Konsole aus bearbeiten:

- Die Beschreibung für Ihre Infrastrukturkonfiguration.
- Die IAM-Rolle, die dem Instanzprofil zugeordnet werden soll.
- AWS Infrastruktur, einschließlich des Instanztyps und eines SNS-Themas für Benachrichtigungen.
- VPC, Subnetz und Sicherheitsgruppen.
- Einstellungen zur Fehlerbehebung, einschließlich Instanz bei Ausfall beenden, Schlüsselpaar für die Verbindung und optionaler S3-Bucket-Speicherort für Instanzprotokolle.

Gehen Sie folgendermaßen vor, um eine Infrastrukturkonfigurationsressource von der Image Builder Builder-Konsole aus zu aktualisieren:

Wählen Sie eine bestehende Image Builder Builder-Infrastrukturkonfiguration

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Um eine Liste der Ressourcen für die Infrastrukturkonfiguration unter Ihrem Konto anzuzeigen, wählen Sie im Navigationsbereich Infrastrukturkonfiguration aus.

3. Um Details anzuzeigen oder eine Infrastrukturkonfiguration zu bearbeiten, wählen Sie den Link „Konfigurationsname“. Dadurch wird die Detailansicht für die Infrastrukturkonfiguration geöffnet.

   > **Note**
   >
   > Sie können auch das Kontrollkästchen neben dem Namen der Konfiguration aktivieren und dann Detail anzeigen auswählen.

4. Wählen Sie in der oberen rechten Ecke des Bereichs Infrastrukturdetails die Option Bearbeiten aus.
5. Wenn Sie bereit sind, die an Ihrer Infrastrukturkonfiguration vorgenommenen Aktualisierungen zu speichern, wählen Sie Änderungen speichern.

AWS CLI

Das folgende Beispiel zeigt, wie Sie die Infrastrukturkonfiguration für Ihr Image mit dem Image Builder update-infrastructure-configuration Builder-Befehl in der aktualisieren AWS CLI.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Dieses Beispiel für eine Infrastrukturkonfiguration verwendet dieselben Einstellungen wie das Create-Beispiel, mit der Ausnahme, dass wir die `terminateInstanceOnFailure` Einstellung auf aktualisiert haben`false`. Nachdem wir den update-infrastructure-configuration Befehl ausgeführt haben, beenden Pipelines, die diese Infrastrukturkonfiguration verwenden, die Build- und Testinstanzen, wenn der Build fehlschlägt.

   Verwenden Sie ein Dateibearbeitungstool, um eine JSON-Datei mit den im folgenden Beispiel gezeigten Schlüsseln und Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen`update-infrastructure-configuration.json`:

   ```
   {
   "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
   west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
   configuration",
   "description": "An example that will terminate instances of failed builds",
   "instanceTypes": [
   ```

```
    "m5.large", "m5.2xlarge"
],
"instanceProfileName": "myIAMInstanceProfileName",
"securityGroupIds": [
    "sg-12345678"
],
"subnetId": "sub-12345678",
"logging": {
    "s3Logs": {
        "s3BucketName": "my-logging-bucket",
        "s3KeyPrefix": "my-path"
    }
},
"terminateInstanceOnFailure": true,
"snsTopicArn": "arn:aws:sns:us-west-2:123456789012:MyTopic"
}
```

2. Verwenden Sie die Datei, die Sie erstellt haben, als Eingabe, wenn Sie den folgenden Befehl ausführen.

```
aws imagebuilder update-infrastructure-configuration --cli-input-json
 file://update-infrastructure-configuration.json
```

EC2 Image Builder und VPC-Schnittstellen-Endpunkte ()AWS PrivateLink

Sie können eine private Verbindung zwischen Ihrer VPC und EC2 Image Builder herstellen, indem Sie einen VPC-Schnittstellen-Endpunkt erstellen. Schnittstellenendpunkte basieren auf einer Technologie AWS PrivateLink, mit der Sie privat auf Image Builder Builder-APIs zugreifen können, ohne dass ein Internet-Gateway, ein NAT-Gerät, eine VPN-Verbindung oder AWS Direct Connect eine Verbindung erforderlich ist. Instances in Ihrer VPC benötigen keine öffentlichen IP-Adressen, um mit Image Builder Builder-APIs zu kommunizieren. Der Datenverkehr zwischen Ihrer VPC und Image Builder verlässt das Amazon-Netzwerk nicht.

Jeder Schnittstellenendpunkt wird durch eine oder mehrere Elastic-Network-Schnittstellen in Ihren Subnetzen dargestellt. Wenn Sie ein neues Image erstellen, können Sie die VPC-Subnetz-ID in Ihrer Infrastrukturkonfiguration angeben.

> **Note**
>
> Jeder Service, auf den Sie innerhalb einer VPC zugreifen, hat seinen eigenen Schnittstellenendpunkt mit eigener Endpunktrichtlinie. Image Builder lädt die AWSTOE Component Manager-Anwendung herunter und greift auf verwaltete Ressourcen aus S3-Buckets zu, um benutzerdefinierte Images zu erstellen. Um Zugriff auf diese Buckets zu gewähren, müssen Sie die S3-Endpunktrichtlinie aktualisieren, um dies zuzulassen. Weitere Informationen finden Sie unter Benutzerdefinierte Richtlinien für den S3-Bucket-Zugriff.

Weitere Informationen zu VPC-Endpunkten finden Sie unter Schnittstellen-VPC-Endpunkte (AWS PrivateLink) im Amazon VPC Benutzerhandbuch.

Überlegungen zu Image Builder Builder-VPC-Endpoints

Bevor Sie einen Schnittstellen-VPC-Endpunkt für Image Builder einrichten, sollten Sie die Eigenschaften und Einschränkungen der Schnittstellen-Endpunkte im Amazon VPC-Benutzerhandbuch lesen.

Image Builder unterstützt Aufrufe aller API-Aktionen von Ihrer VPC aus.

Erstellen Sie einen VPC-Schnittstellen-Endpunkt für Image Builder

Um einen VPC-Endpunkt für den Image Builder Builder-Service zu erstellen, können Sie entweder die Amazon VPC-Konsole oder die AWS Command Line Interface ()AWS CLI verwenden. Weitere Informationen finden Sie unter Erstellung eines Schnittstellenendpunkts im Benutzerhandbuch für Amazon VPC.

Erstellen Sie einen VPC-Endpunkt für Image Builder mit dem folgenden Dienstnamen:

- com.amazonaws.*region*.imagebuilder

Wenn Sie privates DNS für den Endpunkt aktivieren, können Sie API-Anfragen an Image Builder stellen, indem Sie dessen Standard-DNS-Namen für die Region verwenden, zum Beispiel:`imagebuilder.us-east-1.amazonaws.com`. Informationen zum Endpunkt, der für Ihre Zielregion gilt, finden Sie unter EC2 Image Builder Builder-Endpunkte und Kontingente in der. Allgemeine Amazon Web Services-Referenz

Weitere Informationen finden Sie unter Zugriff auf einen Service über einen Schnittstellenendpunkt im Benutzerhandbuch für Amazon VPC.

Erstellen Sie eine VPC-Endpunktrichtlinie für Image Builder

Sie können Ihrem VPC-Endpunkt eine Endpunktrichtlinie hinzufügen, die den Zugriff auf Image Builder steuert. Die Richtlinie gibt die folgenden Informationen an:

- Prinzipal, der die Aktionen ausführen kann.
- Aktionen, die ausgeführt werden können
- Die Ressourcen, für die Aktionen ausgeführt werden können.

Wenn Sie in Ihrem Rezept von Amazon verwaltete Komponenten verwenden, muss der VPC-Endpunkt für Image Builder den Zugriff auf die folgende Komponentenbibliothek ermöglichen, die dem Service gehört:

`arn:aws:imagebuilder:region:aws:component/*`

> **Important**
>
> Wenn eine nicht standardmäßige Richtlinie auf einen VPC-Schnittstellen-Endpunkt für EC2 Image Builder angewendet wird, werden bestimmte fehlgeschlagene API-Anfragen, z. B. solche, die von fehlschlagen`RequestLimitExceeded`, möglicherweise nicht bei Amazon protokolliert. AWS CloudTrail CloudWatch

Weitere Informationen finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten im Amazon-VPC-Benutzerhandbuch.

Benutzerdefinierte Richtlinien für den S3-Bucket-Zugriff

Image Builder verwendet einen öffentlich verfügbaren S3-Bucket, um verwaltete Ressourcen wie Komponenten zu speichern und darauf zuzugreifen. Außerdem wird die AWSTOE Komponentenverwaltungsanwendung aus einem separaten S3-Bucket heruntergeladen. Wenn Sie in Ihrer Umgebung einen VPC-Endpunkt für Amazon S3 verwenden, müssen Sie sicherstellen, dass Ihre S3-VPC-Endpunktrichtlinie Image Builder den Zugriff auf die folgenden S3-Buckets ermöglicht. *Die Bucket-Namen sind pro AWS Region (Region) und Anwendungsumgebung (Umgebung) eindeutig.* Image Builder und AWSTOE unterstützt die folgenden Anwendungsumgebungen: `prod`preprod, und`beta`.

- Der AWSTOE Component Manager-Bucket:

```
s3://ec2imagebuilder-toe-region-environment
```

Beispiel: s3://ec2 imagebuilder-toe-us-west -2-prod/*

- Der Bucket für verwaltete Ressourcen in Image Builder:

```
s3://ec2imagebuilder-managed-resources-region-environment/components
```

Beispiel: s3://ec2 imagebuilder-managed-resources-us -west-2-prod/components/*

Beispiele für VPC-Endpunktrichtlinien

Dieser Abschnitt enthält Beispiele für benutzerdefinierte VPC-Endpunktrichtlinien.

Allgemeine VPC-Endpunktrichtlinie für Image Builder Builder-Aktionen

Die folgende Beispiel-Endpunktrichtlinie für Image Builder verweigert die Erlaubnis, Image Builder Builder-Images und -Komponenten zu löschen. Die Beispielrichtlinie gewährt auch die Erlaubnis, alle anderen EC2 Image Builder Builder-Aktionen auszuführen.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Action": "imagebuilder:*",
        "Effect": "Allow",
        "Resource": "*"
    },
    {
        "Action": [
            "imagebuilder: DeleteImage"
        ],
        "Effect": "Deny",
        "Resource": "*",
    },
    {
        "Action": [
            "imagebuilder: DeleteComponent"
        ],
        "Effect": "Deny",
        "Resource": "*",
```

```
    }]
}
```

Beschränken Sie den Zugriff nach Organisation, erlauben Sie den Zugriff auf verwaltete Komponenten

Die folgende Beispiel-Endpunktrichtlinie zeigt, wie Sie den Zugriff auf Identitäten und Ressourcen einschränken, die zu Ihrer Organisation gehören, und wie Sie Zugriff auf die von Amazon AWSTOE verwalteten Komponenten gewähren können. Ersetzen Sie *Region* und *principal-org-id*durch die *resource-org-id*Werte Ihrer Organisation.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowRequestsByOrgsIdentitiesToOrgsResources",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "principal-org-id",
          "aws:ResourceOrgID": "resource-org-id"
        }
      }
    },
    {
      "Sid": "AllowAccessToEC2ImageBuilderComponents",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "imagebuilder:GetComponent"
      ],
      "Resource": [
        "arn:aws:imagebuilder:region:aws:component/*"
      ]
    }
  ]
```

```
}
```

VPC-Endpunktrichtlinie für Amazon S3 S3-Bucket-Zugriff

Das folgende Beispiel für eine S3-Endpunktrichtlinie zeigt, wie Sie Zugriff auf die S3-Buckets gewähren, die Image Builder zum Erstellen benutzerdefinierter Images verwendet. Ersetzen Sie *Region* und *Umgebung* durch die Werte Ihres Unternehmens. Fügen Sie der Richtlinie alle weiteren erforderlichen Berechtigungen hinzu, die Ihren Anwendungsanforderungen entsprechen.

> **Note**
>
> Wenn Sie bei Linux-Images keine Benutzerdaten in Ihrem Image-Rezept angeben, fügt Image Builder ein Skript zum Herunterladen und Installieren des Systems Manager Manager-Agenten auf den Build- und Testinstanzen für Ihr Image hinzu. Um den Agenten herunterzuladen, greift Image Builder auf den S3-Bucket für Ihre Build-Region zu.
> Um sicherzustellen, dass Image Builder die Build- und Testinstanzen booten kann, fügen Sie Ihrer S3-Endpunktrichtlinie die folgende zusätzliche Ressource hinzu:
> `"arn:aws:s3:::amazon-ssm-region/*"`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowImageBuilderAccessToAppAndComponentBuckets",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::ec2imagebuilder-toe-region-environment/*",
        "arn:aws:s3:::ec2imagebuilder-managed-resources-region-environment/components/
*"
      ]
    }
  ]
}
```

EC2 Image Builder Builder-Verteilungseinstellungen verwalten

Nachdem Sie Verteilungseinstellungen mit Image Builder erstellt haben, können Sie sie mit der Image Builder Builder-Konsole, der Image Builder Builder-API oder imagebuilder Befehlen in der verwalten AWS CLI. Mit den Verteilungseinstellungen können Sie die folgenden Aktionen ausführen:

AMI-Vertrieb

- Geben Sie den Namen und die Beschreibung Ihres Ausgabe-AMI an.
- Autorisieren Sie andere AWS-Konten Organisationen und Organisationseinheiten, das AMI vom Konto des Besitzers aus zu starten. Dem Besitzerkonto werden Gebühren in Rechnung gestellt, die mit dem AMI verbunden sind.

  > **Note**
  >
  > Um ein AMI öffentlich zu machen, setzen Sie die Startberechtigung für autorisierte Konten auf `all`. Sehen Sie sich die Beispiele für die Veröffentlichung eines AMI auf EC2 ModifyImageAttribute an.

- Erstellen Sie eine Kopie des Ausgabe-AMI für jedes der angegebenen Zielkonten, Organisationen und Organisationseinheiten in der Zielregion. Die Zielkonten, Organisationen und Organisationseinheiten sind Eigentümer ihrer AMI-Kopien und es werden ihnen alle damit verbundenen Gebühren in Rechnung gestellt. Weitere Informationen zur Verteilung Ihres AMI an AWS Organizations und OUs finden Sie unter Teilen eines AMI mit Organisationen oder OUs.
- Kopieren Sie das AMI auf das Konto des Besitzers in einem anderen Bereich AWS-Regionen.
- Exportieren Sie VM-Image-Festplatten nach Amazon Simple Storage Service (Amazon S3). Weitere Informationen finden Sie unter Erstellen Sie Verteilungseinstellungen für Ausgabe-VM-Festplatten (AWS CLI).

Verteilung von Container-Images

- Geben Sie das ECR-Repository in der Vertriebsregion an, in dem Image Builder das Ausgabebild speichert.

Sie können Ihre Verteilungseinstellungen auf folgende Weise verwenden, um Bilder einmalig oder bei jedem Pipeline-Build an Zielregionen, Konten AWS Organizations und Organisationseinheiten (OUs) bereitzustellen:

- Um automatisch aktualisierte Images für bestimmte Regionen, Konten, Organizations und Organisationseinheiten bereitzustellen, verwenden Sie Verteilungseinstellungen mit einer Image Builder Builder-Pipeline, die nach einem Zeitplan ausgeführt wird.
- Um ein neues Image zu erstellen und es an die angegebenen Regionen, Konten, Organizations und Organisationseinheiten bereitzustellen, verwenden Sie Verteilungseinstellungen mit einer Image Builder Builder-Pipeline, die Sie einmal von der Image Builder Builder-Konsole aus ausführen, indem Sie im Menü Aktionen die Option Pipeline ausführen verwenden.
- Um ein neues Image zu erstellen und es an die angegebenen Regionen, Konten, Organizations und Organisationseinheiten bereitzustellen, verwenden Sie die Verteilungseinstellungen mit der folgenden API-Aktion oder dem folgenden Image Builder Builder-Befehl im AWS CLI:
  - Die CreateImage Aktion in der Image Builder Builder-API.
  - Der create-image Befehl in der AWS CLI.
- Um Image-Festplatten für virtuelle Maschinen (VM) im Rahmen Ihres regulären Image-Build-Prozesses in S3-Buckets in Zielregionen zu exportieren.

> **Tip**
>
> Wenn Sie über mehrere Ressourcen desselben Typs verfügen, hilft Ihnen das Tagging dabei, eine bestimmte Ressource anhand der Tags zu identifizieren, die Sie ihr zugewiesen haben. Weitere Informationen zum Taggen Ihrer Ressourcen mithilfe von Image Builder Builder-Befehlen finden Sie im AWS CLIMarkieren von Ressourcen Abschnitt dieses Handbuchs.

In diesem Thema wird beschrieben, wie Sie Verteilungseinstellungen auflisten, anzeigen und erstellen.

Inhalt

Details zu den Verteilungseinstellungen auflisten und anzeigen

In diesem Abschnitt werden die verschiedenen Möglichkeiten beschrieben, wie Sie Informationen finden und Details zu Ihren EC2 Image Builder Builder-Verteilungseinstellungen anzeigen können.

Einzelheiten zu den Verteilungseinstellungen

- Listet die Verteilungskonfigurationen auf (Konsole)
- Details zur Distributionskonfiguration anzeigen (Konsole)
- Distributionen auflisten ()AWS CLI
- Details zur Distributionskonfiguration abrufen ()AWS CLI

Listet die Verteilungskonfigurationen auf (Konsole)

Gehen Sie folgendermaßen vor, um eine Liste der unter Ihrem Konto erstellten Verteilungskonfigurationen in der Image Builder Builder-Konsole anzuzeigen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich die Option Verteilungseinstellungen aus. Daraufhin wird eine Liste der Verteilungskonfigurationen angezeigt, die unter Ihrem Konto erstellt wurden.
3. Um Details anzuzeigen oder eine neue Verteilungskonfiguration zu erstellen, wählen Sie den Link „Konfigurationsname“. Dadurch wird die Detailansicht für die Verteilungseinstellungen geöffnet.

> **Note**
>
> Sie können auch das Kontrollkästchen neben dem Namen der Konfiguration aktivieren und dann Details anzeigen auswählen.

Details zur Distributionskonfiguration anzeigen (Konsole)

Um Details für eine bestimmte Verteilungskonfiguration mit der Image Builder Builder-Konsole anzuzeigen, wählen Sie die zu überprüfende Konfiguration aus. Gehen Sie dabei wie unter beschrieben vorListet die Verteilungskonfigurationen auf (Konsole).

Auf der Seite mit den Verteilungsdetails können Sie:

- Löschen Sie die Verteilungskonfiguration. Weitere Informationen zum Löschen von Ressourcen in Image Builder finden Sie unterEC2 Image Builder Builder-Ressourcen löschen.
- Bearbeiten Sie die Verteilungsdetails.

Distributionen auflisten ()AWS CLI

Das folgende Beispiel zeigt, wie Sie den list-distribution-configurations Befehl in verwenden, AWS CLI um alle Ihre Distributionen aufzulisten.

```
aws imagebuilder list-distribution-configurations
```

Details zur Distributionskonfiguration abrufen ()AWS CLI

Das folgende Beispiel zeigt, wie Sie den get-distribution-configuration Befehl in verwenden, AWS CLI um die Details einer Distributionskonfiguration abzurufen, indem Sie ihren Amazon-Ressourcennamen (ARN) angeben.

```
aws imagebuilder get-distribution-configuration --distribution-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-
distribution-configuration
```

AMI-Verteilungskonfigurationen erstellen und aktualisieren

In diesem Abschnitt wird das Erstellen und Aktualisieren von Verteilungskonfigurationen für ein Image Builder Builder-AMI behandelt.

Inhalt

- Erstellen Sie eine AMI-Verteilungskonfiguration (Konsole)
- Erstellen Sie Verteilungseinstellungen für Ausgabe-AMIs (AWS CLI)
- AMI-Verteilungseinstellungen aktualisieren (Konsole)
- Verteilungseinstellungen für ein Windows-AMI mit aktiviertem EC2 Fast Launch erstellen ()AWS CLI
- Erstellen Sie Verteilungseinstellungen für Ausgabe-VM-Festplatten (AWS CLI)
- AMI-Verteilungseinstellungen aktualisieren (AWS CLI)

Erstellen Sie eine AMI-Verteilungskonfiguration (Konsole)

Zu den Verteilungskonfigurationen gehören der Name des Ausgabe-AMIs, spezifische Regionseinstellungen für die Verschlüsselung, Startberechtigungen und AWS-Konten Organisationen und Organisationseinheiten (OUs), die das Ausgabe-AMI starten können, sowie Lizenzkonfigurationen.

So erstellen Sie eine neue AMI-Verteilungskonfiguration:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich die Option Verteilungseinstellungen aus. Daraufhin wird eine Liste der Verteilungskonfigurationen angezeigt, die unter Ihrem Konto erstellt wurden.
3. Wählen Sie oben im Bereich Verteilungseinstellungen die Option Verteilungseinstellungen erstellen aus.
4. Wählen Sie im Abschnitt Image-Typ den Ausgabetyp Amazon Machine Image (AMI) aus.
5. Geben Sie im Abschnitt Allgemein einen Namen für Ihre Distributionskonfiguration und optional eine Beschreibung ein.
6. Geben Sie im Abschnitt Regionseinstellungen die folgenden Details für jede Region ein, in der Sie Ihr AMI verteilen:

    a. Das AMI wird standardmäßig auf die aktuelle Region (Region 1) verteilt. Region 1 ist die Quelle für die Verteilung. Einige Einstellungen für Region 1 können nicht bearbeitet werden. Für alle Regionen, die Sie hinzufügen, können Sie eine Region aus der Dropdownliste Region auswählen.

    Der Kms-Schlüssel identifiziert den AWS KMS key , der zur Verschlüsselung der EBS-Volumes für Ihr Image in der Zielregion verwendet wird. Es ist wichtig zu beachten, dass dies nicht für das ursprüngliche AMI gilt, das der Build unter Ihrem Konto in der Quellregion (Region 1) erstellt. Die Verschlüsselung, die während der Verteilungsphase des Builds ausgeführt wird, gilt nur für Images, die an andere Konten oder Regionen verteilt werden.

    Um die EBS-Volumes für das AMI zu verschlüsseln, das in der Quellregion für Ihr Konto erstellt wurde, müssen Sie den KMS-Schlüssel in der Image-Rezeptblock-Gerätezuordnung (Speicher (Volumes) in der Konsole) festlegen.

    Image Builder kopiert das AMI auf die Target-Konten, die Sie für die Region angeben.

> **Voraussetzung**
>
> Um ein Image kontenübergreifend zu kopieren, müssen Sie die `EC2ImageBuilderDistributionCrossAccountRole` Rolle in allen Zielkonten in den Zielregionen erstellen und die Ec2ImageBuilderCrossAccountDistributionAccess-Richtlinie verwaltete Richtlinie an die Rolle anhängen.

Der Name des Ausgabe-AMIs ist optional. Wenn Sie einen Namen angeben, enthält der endgültige Ausgabe-AMI-Name einen angehängten Zeitstempel, der angibt, wann das AMI erstellt wurde. Wenn Sie keinen Namen angeben, hängt Image Builder den Build-Zeitstempel an den Rezeptnamen an. Dadurch werden eindeutige AMI-Namen für jeden Build gewährleistet.

i. Mit AMI-Sharing können Sie bestimmten AWS Principals Zugriff gewähren, um Instances von Ihrem AMI aus zu starten. Wenn Sie den Abschnitt AMI-Sharing erweitern, können Sie die folgenden Details eingeben:

   - Startberechtigungen — Wählen Sie Privat aus, wenn Sie Ihr AMI privat halten und bestimmten AWS Principals Zugriff gewähren möchten, um eine Instance von Ihrem privaten AMI aus zu starten. Wählen Sie Öffentlich aus, wenn Sie Ihr AMI veröffentlichen möchten. Jeder AWS Principal kann eine Instance von Ihrem öffentlichen AMI aus starten.
   - Principals — Sie können den folgenden AWS Principal-Typen Zugriff gewähren, um Instances zu starten:
     - AWS Konto — Gewähren Sie Zugriff auf ein bestimmtes Konto AWS
     - Organisationseinheit (OU) — Gewähren Sie Zugriff auf eine Organisationseinheit und all ihre untergeordneten Entitäten. Zu den untergeordneten Entitäten gehören Organisationseinheiten und AWS Konten.
     - Organisation — Gewähren Sie Zugriff auf Ihre AWS Organizations und alle untergeordneten Entitäten. Zu den untergeordneten Entitäten gehören Organisationseinheiten und AWS Konten.

Wählen Sie zunächst den Prinzipaltyp aus. Geben Sie dann die ID für den AWS Principal, dem Sie Zugriff gewähren möchten, in das Feld rechts neben der Dropdownliste ein. Sie können mehrere IDs unterschiedlichen Typs eingeben.

ii. Sie können den Abschnitt Lizenzkonfiguration erweitern, um Lizenzkonfigurationen, die mit erstellt wurden AWS License Manager , an Ihre Image Builder Builder-Images anzuhängen. Lizenzkonfigurationen enthalten Lizenzregeln, die auf den Bedingungen Ihrer Unternehmensvereinbarungen basieren. Image Builder schließt automatisch Lizenzkonfigurationen ein, die mit Ihrem Basis-AMI verknüpft waren.

iii. Sie können den Abschnitt Konfiguration der Startvorlage erweitern, um eine EC2-Startvorlage anzugeben, die zum Starten von Instances über das von Ihnen erstellte AMI verwendet werden soll.

Wenn Sie eine EC2-Startvorlage verwenden, können Sie Image Builder anweisen, nach Abschluss des Builds eine neue Version Ihrer Startvorlage zu erstellen, die die neueste AMI-ID enthält. Um die Startvorlage zu aktualisieren, konfigurieren Sie die Einstellungen wie folgt:

- Name der Startvorlage — Wählen Sie den Namen der Startvorlage aus, die Image Builder aktualisieren soll.
- Standardversion festlegen — Aktivieren Sie dieses Kontrollkästchen, um die Standardversion der Startvorlage auf die neue Version zu aktualisieren.

Um eine weitere Startvorlagenkonfiguration hinzuzufügen, wählen Sie Startvorlagenkonfiguration hinzufügen. Sie können bis zu fünf Startvorlagenkonfigurationen pro Region einrichten.

b. Um Vertriebseinstellungen für eine andere Region hinzuzufügen, wählen Sie Region hinzufügen.

7. Wählen Sie Einstellungen erstellen, wenn Sie fertig sind.

Erstellen Sie Verteilungseinstellungen für Ausgabe-AMIs (AWS CLI)

Eine Verteilungskonfiguration ermöglicht es Ihnen, den Namen und die Beschreibung Ihres Ausgabe-AMI anzugeben, andere AWS-Konten zum Starten des AMI zu autorisieren, das AMI auf andere Konten zu kopieren und das AMI in andere AWS Regionen zu replizieren. Außerdem können Sie das AMI nach Amazon Simple Storage Service (Amazon S3) exportieren oder EC2 Fast Launch

für die Ausgabe von Windows-AMIs konfigurieren. Um ein AMI öffentlich zu machen, setzen Sie die Startberechtigung für autorisierte Konten auf `all`. Sehen Sie sich die Beispiele für die Veröffentlichung eines AMI auf EC2 ModifyImageAttribute an.

Das folgende Beispiel zeigt, wie Sie den create-distribution-configuration Befehl verwenden, um eine neue Distributionskonfiguration für Ihr AMI zu erstellen, indem Sie den verwenden AWS CLI.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie ein Tool zur Dateibearbeitung, um eine JSON-Datei mit Schlüsseln, die in einem der folgenden Beispiele gezeigt werden, und Werten, die für Ihre Umgebung gültig sind, zu erstellen. In diesen Beispielen wird definiert AWS-Konten, welche AWS Organizations oder welche Organisationseinheiten (OUs) berechtigt sind, das AMI zu starten, das Sie an die angegebenen Regionen verteilen. Benennen Sie die Datei`create-ami-distribution-configuration.json`, die im nächsten Schritt verwendet werden soll:

   Accounts

   In diesem Beispiel wird ein AMI auf zwei Regionen verteilt und angegeben AWS-Konten , welche Regionen über Startberechtigungen verfügen.

```
{
    "name": "MyExampleAccountDistribution",
    "description": "Copies AMI to eu-west-1, and specifies accounts that can
 launch instances in each Region.",
    "distributions": [
        {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
                "name": "Name {{imagebuilder:buildDate}}",
                "description": "An example image name with parameter
 references",
                "amiTags": {
                    "KeyName": "Some Value"
                },
                "launchPermission": {
                    "userIds": [
                        "987654321012"
                    ]
                }
            }
        },
```

```
        {
            "region": "eu-west-1",
            "amiDistributionConfiguration": {
                "name": "My {{imagebuilder:buildVersion}} image
 {{imagebuilder:buildDate}}",
                "amiTags": {
                    "KeyName": "Some value"
                },
                "launchPermission": {
                    "userIds": [
                        "100000000001"
                    ]
                }
            }
        }
    ]
}
```

Organizations and OUs

In diesem Beispiel wird ein AMI an die Quellregion verteilt und die Startberechtigungen für Organisationen und Organisationseinheiten angegeben.

```
{
  "name": "MyExampleAWSOrganizationDistribution",
  "description": "Shares AMI with the Organization and OU",
  "distributions": [
    {
      "region": "us-west-2",
      "amiDistributionConfiguration": {
        "name": "Name {{ imagebuilder:buildDate }}",
        "launchPermission": {
          "organizationArns": [
            "arn:aws:organizations::123456789012:organization/o-
myorganization123"
          ],
          "organizationalUnitArns": [
            "arn:aws:organizations::123456789012:ou/o-123example/ou-1234-
myorganizationalunit"
          ]
        }
      }
    }
```

```
  ]
}
```

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder create-distribution-configuration --cli-input-json file://create-ami-distribution-configuration.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Ausführlichere Informationen finden Sie create-distribution-configuration in der AWS CLI Befehlsreferenz.

AMI-Verteilungseinstellungen aktualisieren (Konsole)

Sie können Ihre AMI-Verteilungseinstellungen mit der Image Builder Builder-Konsole ändern. Die aktualisierten Verteilungseinstellungen werden künftig für alle automatisierten und manuellen Pipeline-Bereitstellungen verwendet. Die Änderungen, die Sie vornehmen, gelten jedoch nicht für Ressourcen, die Image Builder bereits verteilt hat. Wenn Sie beispielsweise ein AMI an eine Region verteilt haben, das Sie später aus Ihrer Distribution entfernen, verbleibt das bereits verteilte AMI in dieser Region, bis Sie es manuell entfernen.

AMI-Verteilungskonfiguration aktualisieren

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich die Option Verteilungseinstellungen aus. Daraufhin wird eine Liste der Verteilungskonfigurationen angezeigt, die unter Ihrem Konto erstellt wurden.
3. Um Details anzuzeigen oder eine Verteilungskonfiguration zu aktualisieren, wählen Sie den Link „Konfigurationsname“. Dadurch wird die Detailansicht für die Verteilungseinstellungen geöffnet.

> **Note**
>
> Sie können auch das Kontrollkästchen neben dem Namen der Konfiguration aktivieren und dann Details anzeigen auswählen.

4. Um die Verteilungskonfiguration zu bearbeiten, wählen Sie in der oberen rechten Ecke des Abschnitts Verteilungsdetails die Option Bearbeiten aus. Einige Felder sind gesperrt, z. B. der Name der Verteilungskonfiguration und die Standardregion, die als Region 1 angezeigt wird. Weitere Informationen zu den Einstellungen für die Verteilungskonfiguration finden Sie unterErstellen Sie eine AMI-Verteilungskonfiguration (Konsole).
5. Wählen Sie abschließend Änderungen speichern aus.

Verteilungseinstellungen für ein Windows-AMI mit aktiviertem EC2 Fast Launch erstellen ()AWS CLI

Das folgende Beispiel zeigt, wie Sie den create-distribution-configuration Befehl verwenden, um Verteilungseinstellungen zu erstellen, für die EC2 Fast Launch für Ihr AMI konfiguriert ist, aus dem AWS CLI.

> **Note**
>
> Image Builder unterstützt keine kontoübergreifende Verteilung für AMIs, für die EC2 Fast Launch vorab aktiviert ist. EC2 Fast Launch muss vom Zielkonto aus aktiviert werden.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie ein Dateibearbeitungstool, um eine JSON-Datei mit Schlüsseln zu erstellen, wie im folgenden Beispiel gezeigt, plus Werten, die für Ihre Umgebung gültig sind.

   In diesem Beispiel werden Instances für alle Zielressourcen gleichzeitig gestartet, da die maximale Anzahl parallel Starts größer ist als die Anzahl der Zielressourcen. Diese Datei ist `ami-dist-config-win-fast-launch.json` in dem Befehlsbeispiel benannt, das im nächsten Schritt gezeigt wird.

   ```
   {
   "name": "WinFastLaunchDistribution",
   ```

```
"description": "An example of Windows AMI EC2 Fast Launch settings in the
 distribution configuration.",
"distributions": [
    {
        "region": "us-west-2",
        "amiDistributionConfiguration": {
            "name": "Name {{imagebuilder:buildDate}}",
            "description": "Includes Windows AMI EC2 Fast Launch settings.",
            "amiTags": {
                "KeyName": "Some Value"
            }
        },
        "fastLaunchConfigurations": [{
            "enabled": true,
            "snapshotConfiguration": {
                "targetResourceCount": 5
            },
            "maxParallelLaunches": 6,
            "launchTemplate": {
                "launchTemplateId": "lt-0ab1234c56d789012",
                "launchTemplateVersion": "1"
             }
        }],
        "launchTemplateConfigurations": [{
                    "launchTemplateId": "lt-0ab1234c56d789012",
                    "setDefaultVersion": true
          }]
    }]
}
```

> **Note**
>
> Sie können `launchTemplateName` statt des `launchTemplateId` im `launchTemplate` Abschnitt angeben, aber Sie können nicht gleichzeitig den Namen und die ID angeben.

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder create-distribution-configuration --cli-input-json file://ami-
dist-config-win-fast-launch.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Ausführlichere Informationen finden Sie create-distribution-configuration in der AWS CLI Befehlsreferenz.

Erstellen Sie Verteilungseinstellungen für Ausgabe-VM-Festplatten (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den create-distribution-configuration Befehl verwenden, um Verteilungseinstellungen zu erstellen, die VM-Image-Festplatten bei jedem Image-Build nach Amazon S3 exportieren.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Sie können den create-distribution-configuration Befehl, den Sie in der verwenden, rationalisieren. AWS CLI Erstellen Sie dazu eine JSON-Datei, die die gesamte Exportkonfiguration enthält, die Sie an den Befehl übergeben möchten.

   > **Note**
   >
   > Die Benennungskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Aktionsanforderungsparameter der Image Builder Builder-API angegeben ist. Informationen zu den API-Befehlsanforderungsparametern finden Sie unter dem CreateDistributionConfiguration Befehl in der EC2 Image Builder API-Referenz. Informationen zur Bereitstellung der Datenwerte als Befehlszeilenparameter finden Sie in den Parameternamen, die in der AWS CLI Befehlsreferenz angegeben sind, und den create-distribution-configuration Befehl als Optionen.

Im Folgenden finden Sie eine Zusammenfassung der Parameter, die wir im `s3ExportConfiguration` JSON-Objekt für dieses Beispiel angeben:

- roleName (string, erforderlich) — Der Name der Rolle, die VM Import/Export-Berechtigungen zum Exportieren von Bildern in Ihren S3-Bucket erteilt.
- diskImageFormat(string, erforderlich) — Exportiert das aktualisierte Festplatten-Image in eines der folgenden unterstützten Formate:
  - Virtuelle Festplatte (VHD) — kompatibel mit Citrix Xen- und Microsoft Hyper-V-Virtualisierungsprodukten.
  - Stream-optimierte ESX Virtual Machine Disk (VMDK) — kompatibel mit den Versionen 4, 5 und 6 von VMware ESX und VMware vSphere.
  - Raw — Rohformat.
- s3Bucket (string, erforderlich) — Der S3-Bucket, in dem die ausgegebenen Disk-Images für Ihre VM gespeichert werden sollen.

Speichern Sie die Datei als `export-vm-disks.json`. Verwenden Sie den Dateinamen im Befehlcreate-distribution-configuration.

```
{
    "name": "example-distribution-configuration-with-vm-export",
    "description": "example",
    "distributions": [
        {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
             "description": "example-with-vm-export"

            },
            "s3ExportConfiguration": {
                "roleName": "vmimport",
                "diskImageFormat": "RAW",
                "s3Bucket": "vm-bucket-export"
            }
        }],
    "clientToken": "abc123def4567ab"
}
```

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder create-distribution-configuration --cli-input-json file://export-vm-disks.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Ausführlichere Informationen finden Sie create-distribution-configuration in der AWS CLI Befehlsreferenz.

AMI-Verteilungseinstellungen aktualisieren (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den update-distribution-configuration Befehl verwenden, um die Verteilungseinstellungen für Ihr AMI zu aktualisieren, indem Sie den verwenden AWS CLI.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den im folgenden Beispiel gezeigten Schlüsseln sowie Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen verwendet. `update-ami-distribution-configuration.json`

```
{
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/update-ami-distribution-
configuration.json",
    "description": "Copies AMI to eu-west-2, and specifies accounts that can launch
 instances in each Region.",
    "distributions": [
      {
```

```
            "region": "us-west-2",
            "amiDistributionConfiguration": {
                "name": "Name {{imagebuilder:buildDate}}",
                "description": "An example image name with parameter references",
                "launchPermissions": {
                    "userIds": [
                        "987654321012"
                    ]
                }
            }
        },
        {
            "region": "eu-west-2",
            "amiDistributionConfiguration": {
                "name": "My {{imagebuilder:buildVersion}} image
 {{imagebuilder:buildDate}}",
                "tags": {
                    "KeyName": "Some value"
                },
                "launchPermissions": {
                    "userIds": [
                        "100000000001"
                    ]
                }
            }
        }
    ]
}
```

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder update-distribution-configuration --cli-input-json file://update-ami-distribution-configuration.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet

beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Ausführlichere Informationen finden Sie update-distribution-configuration in der AWS CLI Befehlsreferenz. Informationen zum Aktualisieren von Tags für Ihre Distributionskonfigurationsressource finden Sie im Markieren von Ressourcen Abschnitt.

Verteilungseinstellungen für Container-Images erstellen und aktualisieren

In diesem Abschnitt wird das Erstellen und Aktualisieren von Verteilungseinstellungen für Image Builder Builder-Container-Images beschrieben.

Inhalt

Verteilungseinstellungen für Image Builder Builder-Container-Images erstellen (AWS CLI)

Eine Verteilungskonfiguration ermöglicht es Ihnen, den Namen und die Beschreibung Ihres Ausgabe-Container-Images anzugeben und das Container-Image in andere AWS Regionen zu replizieren. Sie können auch separate Tags auf die Distributionskonfigurationsressource und auf die Container-Images in jeder Region anwenden.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den im folgenden Beispiel gezeigten Schlüsseln sowie Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen`create-container-distribution-configuration.json`:

```
{
 "name": "distribution-configuration-name",
 "description": "Distributes container image to Amazon ECR repository in two
 regions.",
 "distributions": [
     {
```

```
        "region": "us-west-2",
        "containerDistributionConfiguration": {
            "description": "My test image.",
            "targetRepository": {
                "service": "ECR",
                "repositoryName": "testrepo"
            },
            "containerTags": ["west2", "image1"]
        }
    },
    {
        "region": "us-east-1",
        "containerDistributionConfiguration": {
            "description": "My test image.",
            "targetRepository": {
                "service": "ECR",
                "repositoryName": "testrepo"
            },
           "containerTags": ["east1", "imagedist"]
        }
    }
 ],
 "tags": {
    "DistributionConfigurationTestTagKey1":
 "DistributionConfigurationTestTagValue1",
    "DistributionConfigurationTestTagKey2":
 "DistributionConfigurationTestTagValue2"
 }
}
```

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder create-distribution-configuration --cli-input-json file://create-container-distribution-configuration.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet

beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Ausführlichere Informationen finden Sie create-distribution-configuration in der AWS CLI Befehlsreferenz.

Aktualisieren Sie die Verteilungseinstellungen für Ihr Container-Image (AWS CLI)

Das folgende Beispiel zeigt, wie Sie den update-distribution-configuration Befehl verwenden, um die Verteilungseinstellungen für Ihr Container-Image zu aktualisieren, indem Sie den AWS CLI. Sie können auch Tags für die Container-Images in jeder Region aktualisieren.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den im folgenden Beispiel gezeigten Schlüsseln und Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen`update-container-distribution-configuration.json`:

```
{
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/update-container-distribution-
configuration.json",
 "description": "Distributes container image to Amazon ECR repository in two
 regions.",
 "distributions": [
     {
         "region": "us-west-2",
         "containerDistributionConfiguration": {
             "description": "My test image.",
             "targetRepository": {
                 "service": "ECR",
                 "repositoryName": "testrepo"
             },
             "containerTags": ["west2", "image1"]
         }
     },
     {
         "region": "us-east-2",
         "containerDistributionConfiguration": {
```

```
            "description": "My test image.",
            "targetRepository": {
                "service": "ECR",
                "repositoryName": "testrepo"
            },
           "containerTags": ["east2", "imagedist"]
        }
    }
 ]
}
```

2. Führen Sie den folgenden Befehl aus und verwenden Sie dabei die Datei, die Sie als Eingabe erstellt haben:

```
aws imagebuilder update-distribution-configuration --cli-input-json file://update-container-distribution-configuration.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Ausführlichere Informationen finden Sie update-distribution-configuration in der AWS CLI Befehlsreferenz. Informationen zum Aktualisieren von Tags für Ihre Distributionskonfigurationsressource finden Sie im Markieren von Ressourcen Abschnitt.

Richten Sie die kontenübergreifende AMI-Verteilung mit Image Builder ein

In diesem Abschnitt wird beschrieben, wie Sie Verteilungseinstellungen konfigurieren können, um ein Image Builder Builder-AMI für andere von Ihnen angegebene Konten bereitzustellen.

Das Zielkonto kann dann das AMI nach Bedarf starten oder ändern.

> **Note**
>
> AWS CLI Bei den Befehlsbeispielen in diesem Abschnitt wird davon ausgegangen, dass Sie zuvor JSON-Dateien für Image-Rezepte und Infrastrukturkonfigurationen erstellt haben. Informationen zum Erstellen der JSON-Datei für ein Image-Rezept finden Sie unterErstellen Sie ein Bildrezept mit dem AWS CLI. Informationen zum Erstellen der JSON-Datei für eine Infrastrukturkonfiguration finden Sie unterErstellen einer Infrastrukturkonfiguration.

Voraussetzungen

Um sicherzustellen, dass Zielkonten erfolgreich Instances von Ihrem Image Builder Builder-Image aus starten können, müssen Sie die entsprechenden Berechtigungen für alle Zielkonten in allen Regionen konfigurieren.

Wenn Sie Ihr AMI mit AWS Key Management Service (AWS KMS) verschlüsseln, müssen Sie AWS KMS key für Ihr Konto eine konfigurieren, die zum Verschlüsseln des neuen Images verwendet wird.

Wenn Image Builder eine kontoübergreifende Verteilung für verschlüsselte AMIs durchführt, wird das Image im Quellkonto entschlüsselt und in die Zielregion übertragen, wo es mit dem angegebenen Schlüssel für diese Region erneut verschlüsselt wird. Da Image Builder im Namen des Zielkontos handelt und eine IAM-Rolle verwendet, die Sie in der Zielregion erstellen, muss dieses Konto Zugriff auf Schlüssel sowohl in der Quell- als auch in der Zielregion haben.

Verschlüsselungsschlüssel

Die folgenden Voraussetzungen sind erforderlich, wenn Ihr Image mit AWS KMS verschlüsselt ist. Die IAM-Voraussetzungen werden im nächsten Abschnitt behandelt.

Anforderungen an das Quellkonto

- Erstellen Sie in Ihrem Konto in allen Regionen, in denen Sie Ihr AMI erstellen und verteilen, einen KMS-Schlüssel. Sie können auch einen vorhandenen Schlüssel verwenden.
- Aktualisieren Sie die Schlüsselrichtlinie für all diese Schlüssel, damit Zielkonten Ihren Schlüssel verwenden können.

Anforderungen an das Zielkonto

- Fügen Sie eine Inline-Richtlinie hinzu`EC2ImageBuilderDistributionCrossAccountRole`, die es der Rolle ermöglicht, die erforderlichen Aktionen zur Verteilung eines verschlüsselten AMI durchzuführen. Die Schritte zur IAM-Konfiguration finden Sie im Abschnitt IAM-Richtlinien Voraussetzungen.

Weitere Informationen zum kontoübergreifenden Zugriff finden Sie unter Zulassen AWS KMS, dass Benutzer mit anderen Konten einen KMS-Schlüssel verwenden können im AWS Key Management Service Entwicklerhandbuch.

Geben Sie Ihren Verschlüsselungsschlüssel im Image-Rezept wie folgt an:

- Wenn Sie die Image Builder Builder-Konsole verwenden, wählen Sie Ihren Verschlüsselungsschlüssel aus der Dropdownliste Verschlüsselung (KMS-Alias) im Abschnitt Speicher (Volumes) Ihres Rezepts aus.
- Wenn Sie die CreateImageRecipe API-Aktion oder den create-image-recipe Befehl in verwenden AWS CLI, konfigurieren Sie Ihren Schlüssel im `ebs` Abschnitt unter `blockDeviceMappings` Ihrer JSON-Eingabe.

  Das folgende JSON-Snippet zeigt die Verschlüsselungseinstellungen für ein Bildrezept. Sie müssen nicht nur Ihren Verschlüsselungsschlüssel angeben, sondern auch das `encrypted` Flag auf setzen. `true`

```
{
 ...
 "blockDeviceMappings": [
 {
  "deviceName": "Example root volume",
  "ebs": {
   "deleteOnTermination": true,
   "encrypted": true,
   "iops": 100,
   "kmsKeyId": "image-owner-key-id",
   ...
  },
  ...
 }],
 ...
}
```

IAM-Richtlinien

Gehen Sie wie folgt vor, um die kontoübergreifenden Verteilungsberechtigungen in AWS Identity and Access Management (IAM) zu konfigurieren:

1. Um Image Builder Builder-AMIs zu verwenden, die auf Konten verteilt sind, muss der Besitzer des Zielkontos eine neue IAM-Rolle in seinem Konto namens `EC2ImageBuilderDistributionCrossAccountRole` erstellen.
2. Sie müssen das der Rolle zuordnenEc2ImageBuilderCrossAccountDistributionAccess-Richtlinie, um die kontoübergreifende Verteilung zu ermöglichen. Weitere Informationen zu verwalteten Richtlinien finden Sie unter Verwaltete Richtlinien und Inline-Richtlinien im AWS Identity and Access Management Benutzerhandbuch.
3. Stellen Sie sicher, dass die Quellkonto-ID der Vertrauensrichtlinie hinzugefügt wurde, die der IAM-Rolle des Zielkontos zugeordnet ist. Weitere Informationen zu Vertrauensrichtlinien finden Sie im Benutzerhandbuch unter Ressourcenbasierte Richtlinien.AWS Identity and Access Management
4. Wenn das von Ihnen verteilte AMI verschlüsselt ist, muss der Besitzer des Zielkontos die folgende Inline-Richtlinie zu seinem Konto hinzufügen, damit er Ihre KMS-Schlüssel verwenden kann. `EC2ImageBuilderDistributionCrossAccountRole` Der `Principal` Abschnitt enthält ihre Kontonummer. Dadurch kann Image Builder in ihrem Namen handeln, wenn es das AMI mit den entsprechenden Schlüsseln für jede Region ver- und entschlüsselt. AWS KMS

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowRoleToPerformKMSOperationsOnBehalfOfTheDestinationAccount",
   "Effect": "Allow",
   "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey",
    "kms:CreateGrant",
    "kms:ListGrants",
    "kms:RevokeGrant"
   ],
   "Resource": "*"
  }
```

```
 ]
}
```

Weitere Informationen zu Inline-Richtlinien finden Sie unter Inline-Richtlinien im AWS Identity and Access Management Benutzerhandbuch.

5. Wenn Sie eine Amazon EC2 EC2-Startvorlage angeben, müssen Sie außerdem die folgende Richtlinie zu Ihrem `EC2ImageBuilderDistributionCrossAccountRole` Zielkonto hinzufügen. `launchTemplateConfigurations`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateLaunchTemplateVersion",
                "ec2:ModifyLaunchTemplate"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeLaunchTemplates"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*:*:launch-template/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "EC2 Image Builder"
                }
```

```
            }
        }
    ]
}
```

Grenzwerte für den kontoübergreifenden Vertrieb

Bei der Verteilung von Image Builder Builder-Images auf mehrere Konten gibt es einige Einschränkungen:

- Das Zielkonto ist auf 50 gleichzeitige AMI-Kopien für jede Zielregion begrenzt.
- Wenn Sie ein paravirtuelles Virtualisierungs-AMI (PV) in eine andere Region kopieren möchten, muss die Zielregion PV-Virtualisierungs-AMIs unterstützen. Weitere Informationen finden Sie unter Linux AMI-Virtualisierungstypen.
- Sie können keine unverschlüsselte Kopie eines verschlüsselten Snapshots erstellen. Wenn Sie keinen vom Kunden verwalteten Schlüssel AWS Key Management Service (AWS KMS) für den `KmsKeyId` Parameter angeben, verwendet Image Builder den Standardschlüssel für Amazon Elastic Block Store (Amazon EBS). Weitere Informationen finden Sie unter Amazon EBS-Verschlüsselung im Amazon Elastic Compute Cloud-Benutzerhandbuch.

Weitere Informationen finden Sie CreateDistributionConfigurationin der EC2 Image Builder API-Referenz.

Kontenübergreifende Verteilung für ein Image Builder Builder-AMI (Konsole) konfigurieren

In diesem Abschnitt wird beschrieben, wie Sie Verteilungseinstellungen für die kontoübergreifende Verteilung Ihrer Image Builder Builder-AMIs mithilfe von erstellen und konfigurieren. AWS Management Console Für die Konfiguration der kontenübergreifenden Verteilung sind spezielle IAM-Berechtigungen erforderlich. Sie müssen den Abschnitt Voraussetzungen für diesen Abschnitt ausfüllen, bevor Sie fortfahren können.

Gehen Sie folgendermaßen vor, um Verteilungseinstellungen in der Image Builder Builder-Konsole zu erstellen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.

2. Wählen Sie im Navigationsbereich Verteilungseinstellungen aus. Daraufhin wird eine Liste der Verteilungseinstellungen angezeigt, die unter Ihrem Konto erstellt wurden.
3. Wählen Sie oben auf der Seite mit den Verteilungseinstellungen die Option Verteilungseinstellungen erstellen aus. Dadurch gelangen Sie zur Seite „Verteilungseinstellungen erstellen“.
4. Wählen Sie im Abschnitt Image-Typ Amazon Machine Image (AMI) als Ausgabetyp aus. Dies ist die Standardeinstellung.
5. Geben Sie im Abschnitt Allgemein den Namen der Ressource für die Verteilungseinstellungen ein, die Sie erstellen möchten (erforderlich).
6. Geben Sie im Bereich Regionseinstellungen unter Zielkonten für die ausgewählte Region eine 12-stellige Konto-ID ein, an die Sie Ihr AMI verteilen möchten, und drücken Sie die Eingabetaste. Dadurch wird geprüft, ob die Formatierung korrekt ist, und dann wird die Konto-ID angezeigt, die Sie unter dem Feld eingegeben haben. Wiederholen Sie den Vorgang, um weitere Konten hinzuzufügen.

   Um ein von Ihnen eingegebenes Konto zu entfernen, wählen Sie das X, das rechts neben der Konto-ID angezeigt wird.

   Geben Sie den Namen des Ausgabe-AMI für jede Region ein.
7. Geben Sie weitere Einstellungen an, die Sie benötigen, und wählen Sie Einstellungen erstellen, um Ihre neue Ressource für Verteilungseinstellungen zu erstellen.

Kontenübergreifende Verteilung für ein Image Builder Builder-AMI konfigurieren ()AWS CLI

In diesem Abschnitt wird beschrieben, wie Sie eine Datei mit Verteilungseinstellungen konfigurieren und den create-image Befehl in der verwenden AWS CLI , um ein Image Builder Builder-AMI zu erstellen und kontenübergreifend zu verteilen.

Für die Konfiguration der kontenübergreifenden Verteilung sind spezielle IAM-Berechtigungen erforderlich. Sie müssen den Abschnitt Voraussetzungen für diesen Abschnitt abschließen, bevor Sie den create-image Befehl ausführen.

1. Konfigurieren Sie eine Datei mit den Verteilungseinstellungen

   Bevor Sie den create-image Befehl in verwenden, AWS CLI um ein Image Builder Builder-AMI zu erstellen, das an ein anderes Konto verteilt wird, müssen Sie eine

`DistributionConfiguration` JSON-Struktur erstellen, die die Zielkonto-IDs in den `AmiDistributionConfiguration` Einstellungen angibt. Sie müssen mindestens eine `AmiDistributionConfiguration` in der Quellregion angeben.

Die folgende Beispieldatei mit dem Namen `create-distribution-configuration.json` zeigt die Konfiguration für die kontoübergreifende Bildverteilung in der Quellregion.

```
{
 "name": "cross-account-distribution-example",
 "description": "Cross Account Distribution Configuration Example",
 "distributions": [
  {
   "amiDistributionConfiguration": {
    "targetAccountIds": ["123456789012", "987654321098"],
    "name": "Name {{ imagebuilder:buildDate }}",
    "description": "ImageCopy Ami Copy Configuration"
   },
   "region": "us-west-2"
  }
 ]
}
```

2. Erstellen Sie die Verteilungseinstellungen

Um eine Image Builder Builder-Verteilungseinstellungsressource mit dem create-distribution-configurationBefehl in zu erstellen AWS CLI, geben Sie die folgenden Parameter im Befehl an:

- Geben Sie den Namen der Verteilung in den `--name` Parameter ein.
- Hängen Sie die JSON-Datei für die Verteilungskonfiguration an, die Sie im `--cli-input-json` Parameter erstellt haben.

```
aws imagebuilder create-distribution-configuration --name my distribution name --
cli-input-json file://create-distribution-configuration.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet

beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Mithilfe des `--distributions` Parameters können Sie JSON auch direkt im Befehl angeben.

Konfigurieren Sie die AMI-Verteilungseinstellungen für die Verwendung einer Amazon EC2 EC2-Startvorlage

Um ein einheitliches Starterlebnis für Ihr Image Builder Builder-AMI in Zielkonten und Regionen zu gewährleisten, können Sie in Ihren Vertriebseinstellungen eine Amazon EC2 EC2-Startvorlage angeben, indem `launchTemplateConfigurations` Sie Wenn sie während des Verteilungsprozesses vorhanden `launchTemplateConfigurations` sind, erstellt Image Builder eine neue Version der Startvorlage, die alle ursprünglichen Einstellungen aus der Vorlage und die neue AMI-ID aus dem Build enthält. Weitere Informationen zum Starten einer EC2-Instance mithilfe einer Startvorlage finden Sie je nach Zielbetriebssystem unter einem der folgenden Links.

- Starten Sie eine Linux-Instance über eine Startvorlage
- Starten Sie eine Windows-Instance über eine Startvorlage

> **Note**
>
> Wenn Sie Ihrem Image eine Startvorlage hinzufügen, um Windows Fast Launch zu aktivieren, muss die Startvorlage das folgende Tag enthalten, damit Image Builder Windows Fast Launch in Ihrem Namen aktivieren kann.
> `CreatedBy: EC2 Image Builder`

Fügen Sie eine Amazon EC2 EC2-Startvorlage zu Ihren AMI-Verteilungseinstellungen hinzu (Konsole)

Gehen Sie in der Konsole wie folgt vor, um eine Startvorlage mit Ihrem Ausgabe-AMI bereitzustellen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich die Option Verteilungseinstellungen aus. Daraufhin wird eine Liste der Verteilungseinstellungen angezeigt, die unter Ihrem Konto erstellt wurden.

3. Wählen Sie oben auf der Seite mit den Verteilungseinstellungen die Option Verteilungseinstellungen erstellen aus. Dadurch wird die Seite Verteilungseinstellungen erstellen geöffnet.
4. Wählen Sie im Abschnitt Image-Typ den Ausgabetyp Amazon Machine Image (AMI) aus. Dies ist die Standardeinstellung.
5. Geben Sie im Abschnitt Allgemein den Namen der Ressource für die Verteilungseinstellungen ein, die Sie erstellen möchten (erforderlich).
6. Wählen Sie im Abschnitt Regionseinstellungen den Namen einer EC2-Startvorlage aus der Liste aus. Wenn Ihr Konto keine Startvorlagen enthält, wählen Sie Neue Startvorlage erstellen aus, wodurch die Startvorlagen im EC2-Dashboard geöffnet werden.

   Aktivieren Sie das Kontrollkästchen Standardversion festlegen, um die Standardversion der Startvorlage auf die neue Version zu aktualisieren, die Image Builder mit Ihrem Ausgabe-AMI erstellt.

   Um der ausgewählten Region eine weitere Startvorlage hinzuzufügen, wählen Sie Startvorlagenkonfiguration hinzufügen.

   Um eine Startvorlage zu entfernen, wählen Sie Entfernen.
7. Geben Sie weitere Einstellungen an, die Sie benötigen, und wählen Sie Einstellungen erstellen aus, um Ihre neue Ressource für Verteilungseinstellungen zu erstellen.

Fügen Sie eine Amazon EC2 EC2-Startvorlage zu Ihren AMI-Verteilungseinstellungen hinzu ()AWS CLI

In diesem Abschnitt wird beschrieben, wie Sie eine Datei mit Verteilungseinstellungen mit einer Startvorlage konfigurieren und den create-image Befehl in verwenden, AWS CLI um ein Image Builder Builder-AMI und eine neue Version der Startvorlage, die es verwendet, zu erstellen und zu verteilen.

1. Konfigurieren Sie eine Datei mit den Verteilungseinstellungen

   Bevor Sie mithilfe von ein Image Builder Builder-AMI mit einer Startvorlage erstellen können, müssen Sie eine JSON-Struktur für die AWS CLI Verteilungskonfiguration erstellen, die die `launchTemplateConfigurations` Einstellungen spezifiziert. Sie müssen mindestens einen `launchTemplateConfigurations` Eintrag in der Quellregion angeben.

Die folgende Beispieldatei mit dem Namen `create-distribution-config-launch-template.json` zeigt einige mögliche Szenarien für die Konfiguration der Startvorlage in der Quellregion.

```
{
    "name": "NewDistributionConfiguration",
    "description": "This is just a test",
    "distributions": [
        {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
                "name": "test-{{imagebuilder:buildDate}}-
{{imagebuilder:buildVersion}}",
                "description": "description"
            },
            "launchTemplateConfigurations": [
                {
                    "launchTemplateId": "lt-0a1bcde2fgh34567",
                    "accountId": "935302948087",
                    "setDefaultVersion": true
                },
                {
                    "launchTemplateId": "lt-0aaa1bcde2ff3456"
                },
                {
                    "launchTemplateId": "lt-12345678901234567",
                    "accountId": "123456789012"
                }
            ]
        }
    ],
    "clientToken": "clientToken1"
}
```

2. Erstellen Sie die Verteilungseinstellungen

   Um eine Image Builder Builder-Verteilungseinstellungsressource mit dem create-distribution-configurationBefehl in zu erstellen AWS CLI, geben Sie die folgenden Parameter im Befehl an:

   - Geben Sie den Namen der Verteilung in den `--name` Parameter ein.

- Hängen Sie die JSON-Datei für die Verteilungskonfiguration an, die Sie im `--cli-input-json` Parameter erstellt haben.

```
aws imagebuilder create-distribution-configuration --name my distribution name--
cli-input-json file://create-distribution-config-launch-template.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Sie können JSON auch direkt im Befehl angeben, indem Sie den *`--distributions`* Parameter verwenden.

Lebenszyklusrichtlinien für EC2 Image Builder Builder-Images verwalten

Wenn Sie benutzerdefinierte Images erstellen, ist es wichtig, dass Sie einen Plan haben, um diese Images außer Betrieb zu nehmen, bevor sie veraltet sind. Image Builder Builder-Pipelines können Updates und Sicherheitspatches automatisch anwenden. Jeder Build erstellt jedoch eine neue Version des Images und aller zugehörigen Ressourcen, die es verteilt. Frühere Versionen verbleiben in Ihrem Konto, bis Sie sie manuell löschen oder ein Skript zur Ausführung der Aufgabe erstellen.

Mit den Lebenszyklusmanagement-Richtlinien von Image Builder können Sie den Prozess des Verfalls, Deaktivierens und Löschens veralteter Images und der zugehörigen Ressourcen automatisieren. Zu den zugehörigen Ressourcen können Ausgabebilder gehören, die Sie an andere AWS-Konten Organisationen und Organisationseinheiten (OUs) verteilt haben. AWS-Regionen Sie definieren die Regeln dafür, wie und wann jeder Schritt im Lebenszyklusprozess durchgeführt werden soll und welche Schritte in Ihre Richtlinie aufgenommen werden sollen.

Vorteile des automatisierten Lebenszyklusmanagements

Zu den allgemeinen Vorteilen des automatisierten Lebenszyklusmanagements gehören:

- Vereinfacht das Lebenszyklusmanagement für Ihre benutzerdefinierten Images mit einer automatisierten Methode zur Außerbetriebnahme von Images und zugehörigen Ressourcen.
- Hilft dabei, Compliance-Risiken zu vermeiden, die sich aus der Verwendung veralteter Images für den Start neuer Instances ergeben.
- Hält die Bildbestände auf dem neuesten Stand, indem veraltete Bilder entfernt werden.
- Kann die Speicher- und Datenübertragungskosten senken, indem optional die zugehörigen Ressourcen für gelöschte Bilder entfernt werden.

Realisieren Sie Kosteneinsparungen

Die Verwendung von EC2 Image Builder zur Erstellung benutzerdefinierter AMI- oder Container-Images ist kostenlos. Für andere Dienste, die im Prozess verwendet werden, gelten jedoch die Standardpreise. Wenn Sie ungenutzte oder veraltete Bilder und die zugehörigen Ressourcen aus Ihrem entfernen AWS-Konto, können Sie auf folgende Weise Zeit- und Kosteneinsparungen erzielen:

- Reduzieren Sie die Zeit, die zum Patchen vorhandener Bilder benötigt wird, wenn Sie nicht auch unbenutzte oder veraltete Bilder patchen.
- Für AMI-Image-Ressourcen, die Sie löschen, können Sie festlegen, dass auch verteilte AMIs und die zugehörigen Snapshots entfernt werden. Durch diesen Ansatz können Kosten für das Speichern von Snapshots eingespart werden.
- Für Container-Image-Ressourcen, die Sie löschen, können Sie wählen, ob Sie die zugrunde liegenden Ressourcen löschen möchten. Dieser Ansatz kann Amazon ECR-Speicherkosten und Datenübertragungsraten für Ihre Docker Bilder, die in ECR-Repositorys gespeichert sind, sparen.

> **Note**
>
> Image Builder kann die potenziellen Auswirkungen nicht für alle möglichen Downstream-Abhängigkeiten wie Auto Scaling Scaling-Gruppen oder Startvorlagen bewerten. Bei der Konfiguration von Richtlinienaktionen müssen Sie Downstream-Abhängigkeiten für Ihre Images berücksichtigen.

Inhalt

- Lifecycle-Management-Richtlinien für EC2 Image Builder Builder-Image-Ressourcen
- So funktionieren Lebenszyklusmanagementregeln für EC2 Image Builder Builder-Image-Ressourcen

Voraussetzungen für das Lebenszyklusmanagement für EC2 Image Builder Builder-Images

Bevor Sie EC2 Image Builder Builder-Lebenszyklusmanagement-Richtlinien und -Regeln für Ihre Image-Ressourcen definieren können, müssen Sie die folgenden Voraussetzungen erfüllen.

- Erstellen Sie eine IAM-Rolle, die Image Builder die Erlaubnis erteilt, Lebenszyklusrichtlinien auszuführen. Weitere Informationen zum Erstellen der Rolle finden Sie unter Erstellen Sie eine IAM-Rolle für Image Builder Builder-Lebenszyklusmanagement.
- Erstellen Sie eine IAM-Rolle im Zielkonto für zugeordnete Ressourcen, die auf mehrere Konten verteilt wurden. Die Rolle erteilt Image Builder die Berechtigung, Lebenszyklusaktionen im Zielkonto für zugeordnete Ressourcen durchzuführen. Weitere Informationen zum Erstellen der Rolle finden Sie unter Erstellen Sie eine IAM-Rolle für das kontoübergreifende Lebenszyklusmanagement von Image Builder.

  > **Note**
  >
  > Diese Voraussetzung gilt nicht, wenn Sie Startberechtigungen für ein Ausgabe-AMI erteilt haben. Mit Startberechtigungen besitzt das Konto, mit dem Sie geteilt haben, die Instances, die über das gemeinsam genutzte AMI gestartet werden, aber alle AMI-Ressourcen verbleiben in Ihrem Konto.

- Für Container-Images müssen Sie Ihren ECR-Repositorys das folgende Tag hinzufügen, um Image Builder Zugriff auf die Ausführung von Lebenszyklusaktionen für die im Repository gespeicherten Container-Images zu gewähren:. `LifecycleExecutionAccess: EC2 Image Builder`

Erstellen Sie eine IAM-Rolle für Image Builder Builder-Lebenszyklusmanagement

Um Image Builder die Berechtigung zur Ausführung von Lebenszyklusrichtlinien zu erteilen, müssen Sie zunächst die IAM-Rolle erstellen, die für die Ausführung von Lebenszyklusaktionen verwendet wird. Gehen Sie wie folgt vor, um die Servicerolle zu erstellen, die die Berechtigung erteilt.

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

2. Wählen Sie im Navigationsbereich Roles (Rollen).
3. Wählen Sie Rolle erstellen aus. Dies öffnet den ersten Schritt des Prozesses. Wählen Sie eine vertrauenswürdige Entität aus, um Ihre Rolle zu erstellen.
4. Wählen Sie die Option Benutzerdefinierte Vertrauensrichtlinie für den Entitätstyp Vertrauenswürdige aus.
5. Kopieren Sie die folgende JSON-Vertrauensrichtlinie und fügen Sie sie in den Textbereich Benutzerdefinierte Vertrauensrichtlinie ein, wobei Sie den Beispieltext ersetzen. Diese Vertrauensrichtlinie ermöglicht es Image Builder, die Rolle anzunehmen, die Sie für die Ausführung von Lebenszyklusaktionen erstellen.

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

6. Wählen Sie die folgende verwaltete Richtlinie aus der Liste aus: EC2ImageBuilderLifecycleExecutionPolicyund klicken Sie dann auf Weiter. Dadurch wird die Seite „Name“, „Überprüfen“ und „Erstellen“ geöffnet.

> **Tip**
>
> Filtern Sie nach`image`, um die Ergebnisse zu optimieren.

7. Geben Sie einen Role name ein.
8. Nachdem Sie Ihre Einstellungen überprüft haben, wählen Sie Rolle erstellen aus.

Erstellen Sie eine IAM-Rolle für das kontoübergreifende Lebenszyklusmanagement von Image Builder

Um Image Builder die Erlaubnis zu erteilen, Lebenszyklusaktionen in Zielkonten für zugeordnete Ressourcen auszuführen, müssen Sie zunächst die IAM-Rolle erstellen, mit der Lebenszyklusaktionen in diesen Konten ausgeführt werden. Sie müssen die Rolle im Zielkonto erstellen.

Gehen Sie wie folgt vor, um die Servicerolle zu erstellen, die Berechtigungen für das Zielkonto gewährt.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Roles (Rollen).
3. Wählen Sie Rolle erstellen aus. Dies öffnet den ersten Schritt des Prozesses. Wählen Sie eine vertrauenswürdige Entität aus, um Ihre Rolle zu erstellen.
4. Wählen Sie die Option Benutzerdefinierte Vertrauensrichtlinie für den Entitätstyp Vertrauenswürdige aus.
5. Kopieren Sie die folgende JSON-Vertrauensrichtlinie und fügen Sie sie in den Textbereich Benutzerdefinierte Vertrauensrichtlinie ein, wobei Sie den Beispieltext ersetzen. Diese Vertrauensrichtlinie ermöglicht es Image Builder, die Rolle anzunehmen, die Sie für die Ausführung von Lebenszyklusaktionen erstellen.

> **Note**
>
> Wenn Image Builder diese Rolle im Zielkonto verwendet, um auf zugeordnete Ressourcen zu reagieren, die auf Konten verteilt wurden, handelt es im Namen des Besitzers des Zielkontos. Das Konto AWS-Konto , das Sie `aws:SourceAccount` in der Vertrauensrichtlinie konfigurieren, ist das Konto, über das Image Builder diese Ressourcen verteilt hat.

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

```
                "Service": [
                    "imagebuilder.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "444455556666"
                },
                "StringLike": {
                    "aws:SourceArn": "arn:*:imagebuilder:*:*:image/*/*/*"
                }
            }
        }
    ]
}
```

6. Wählen Sie die folgende verwaltete Richtlinie aus der Liste aus: EC2ImageBuilderLifecycleExecutionPolicyund klicken Sie dann auf Weiter. Dadurch wird die Seite „Name“, „Überprüfen“ und „Erstellen“ geöffnet.

> **Tip**
>
> Filtern Sie nachimage, um die Ergebnisse zu optimieren.

7. Geben Sie `Ec2ImageBuilderCrossAccountLifecycleAccess` als Rollennamen ein.

> **Important**
>
> `Ec2ImageBuilderCrossAccountLifecycleAccess`muss der Name dieser Rolle sein.

8. Nachdem Sie Ihre Einstellungen überprüft haben, wählen Sie Rolle erstellen aus.

Lifecycle-Management-Richtlinien für EC2 Image Builder Builder-Image-Ressourcen

Mit Richtlinien für den Image-Lebenszyklus können Sie Ihre Strategie für das Ressourcenmanagement so definieren, dass veraltete Images und die zugehörigen Ressourcen außer Betrieb genommen werden, indem veraltete Images und die zugehörigen Ressourcen

deaktiviert, deaktiviert und gelöscht werden. In diesem Abschnitt erfahren Sie, wie Sie Richtlinien auflisten, Richtliniendetails anzeigen und neue Richtlinien für AMI- und Container-Images erstellen.

Inhalt

Lifecycle-Management-Richtlinien für Image Builder Builder-Image-Ressourcen auflisten

Sie können eine Liste Ihrer Image-Lifecycle-Management-Richtlinien mit wichtigen Detailspalten auf der Listenseite für Lebenszyklusrichtlinien im AWS Management Console oder mit Befehlen oder Aktionen in der Image Builder Builder-API, den SDKs oder AWS CLI abrufen.

Sie können eine der folgenden Methoden verwenden, um Image Builder Builder-Image-Lebenszyklus-Richtlinienressourcen in Ihrem aufzulisten AWS-Konto. Informationen zur API-Aktion finden Sie ListLifecyclePoliciesin der EC2 Image Builder API-Referenz. Die zugehörige SDK-Anfrage finden Sie unter dem Link „Siehe auch“ auf derselben Seite.

AWS Management Console

Die folgenden Details werden in der Konsole für Ihre vorhandenen Richtlinien angezeigt. Sie können eine beliebige Spalte auswählen, um die Sortierreihenfolge Ihrer Ergebnisse zu ändern. Die Richtlinienliste ist zunächst nach dem Richtliniennamen sortiert. Der Spaltenname für die aktuelle Sortierreihenfolge ist fett gedruckt.

Wenn Sie mehr als eine Ergebnisseite haben, werden die Seitenpfeile in der oberen rechten Ecke des Bedienfelds aktiv. Mit der Suchleiste können Sie Ergebnisse nach Richtlinienname, Richtlinienstatus, Ausgabebildtyp und Bildressourcen-ARN filtern.

- Richtlinienname — Der Name der Richtlinie.
- Richtlinienstatus — Gibt an, ob die Richtlinie aktiv oder inaktiv ist.
- Typ — Der Typ des Ausgabebilds, das Image Builder verteilt, wenn Sie eine neue Image-Version (AMI oder Container-Image) erstellen.
- Datum der letzten Ausführung — Das letzte Mal, dass die Lebenszyklus-Richtlinie ausgeführt wurde.

- Erstellungsdatum — Der Zeitstempel seit der Erstellung der Lebenszyklus-Richtlinie.
- ARN — Der Amazon-Ressourcenname (ARN) der Lifecycle-Policy-Ressource.

Gehen Sie wie folgt vor AWS Management Console, um Lebenszyklus-Richtlinien in der aufzulisten:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich die Option Lebenszyklusrichtlinien aus. Daraufhin wird eine Liste der Image-Lebenszyklus-Richtlinien in Ihrem Konto angezeigt.

Verfügbare Aktionen

Auf der Listenseite mit den Lebenszyklusrichtlinien können Sie auch die folgenden Aktionen für Ihre Lebenszyklusrichtlinie ausführen.

Um eine neue Image-Lebenszyklus-Richtlinie zu erstellen, wählen Sie Lebenszyklus-Richtlinie erstellen. Weitere Informationen zum Erstellen einer Richtlinie finden Sie unterLebenszyklusrichtlinien erstellen.

Für alle folgenden Aktionen müssen Sie zuerst die Richtlinie auswählen. Um eine Richtlinie auszuwählen, können Sie das Kontrollkästchen neben dem Richtliniennamen aktivieren.

- Um die Richtlinie ein- oder auszuschalten, wählen Sie im Menü Aktionen die Option Richtlinie deaktivieren oder Richtlinie aktivieren aus.
- Um die Richtlinie zu ändern, wählen Sie im Menü Aktionen die Option Richtlinie bearbeiten aus.
- Um eine Richtlinie zu löschen, wählen Sie im Menü Aktionen die Option Richtlinie löschen aus.
- Um eine neue Richtlinie zu erstellen, die Ihre gewählte Richtlinie als Basiseinstellungen verwendet, wählen Sie im Menü Aktionen die Option Richtlinie klonen aus.

AWS CLI

Das folgende Befehlsbeispiel zeigt, wie Sie mithilfe von Image-Lebenszyklus-Richtlinien für ein bestimmtes Objekt auflisten können AWS-Region. AWS CLI Weitere Informationen zu den Parametern und Optionen, die Sie mit diesem Befehl verwenden können, finden Sie unter dem list-lifecycle-policiesBefehl in der AWS CLI Befehlsreferenz.

Beispiel:

```
aws imagebuilder list-lifecycle-policies \
--region us-west-1
```

Ausgabe:

```
{
    "lifecyclePolicySummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:lifecycle-policy/
sample-lifecycle-policy1",
            "name": "sample-lifecycle-policy1",
            "status": "DISABLED",
            "executionRole": "arn:aws:iam::111122223333:role/sample-lifecycle-role",
            "resourceType": "AMI_IMAGE",
            "dateCreated": "2023-11-07T14:57:01.603000-08:00",
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:lifecycle-policy/
sample-lifecycle-policy2",
            "name": "sample-lifecycle-policy2",
            "status": "ENABLED",
            "executionRole": "arn:aws:iam::111122223333:role/sample-lifecycle-role",
            "resourceType": "AMI_IMAGE",
            "dateCreated": "2023-09-06T10:43:21.436000-07:00",
            "dateLastRun": "2023-11-13T04:43:46.106000-08:00",
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:lifecycle-policy/
sample-lifecycle-policy3",
            "name": "sample-lifecycle-policy3",
            "status": "ENABLED",
            "executionRole": "arn:aws:iam::111122223333:role/sample-lifecycle-role",
            "resourceType": "AMI_IMAGE",
            "dateCreated": "2023-10-19T15:16:40.046000-07:00",
            "dateUpdated": "2023-10-21T20:07:15.958000-07:00",
            "dateLastRun": "2023-11-12T09:27:45.830000-08:00"
}]}
```

> **Note**
>
> Um Ihre Standardeinstellung zu verwenden AWS-Region, führen Sie diesen Befehl ohne den `--region` Parameter aus.

Details zur Lebenszyklusrichtlinie anzeigen

Die Detailseite der Lebenszyklusrichtlinie in der Image Builder Builder-Konsole enthält einen Übersichtsabschnitt mit zusätzlichen Informationen, die in Registerkarten gruppiert sind. Die Seitenüberschrift ist der Name der Richtlinie.

Auf der Detailseite der Lebenszyklusrichtlinie in der Image Builder Builder-Konsole können Sie Details für eine bestimmte Lebenszyklusrichtlinie anzeigen. Sie können Befehle oder Aktionen auch mit der Image Builder Builder-API, SDKs oder AWS CLI zum Abrufen von Richtliniendetails verwenden.

Inhalt

Details zur Lebenszyklusrichtlinie in der Image Builder Builder-Konsole anzeigen

Die Image-Detailseite in der Image Builder Builder-Konsole enthält einen Übersichtsabschnitt mit zusätzlichen Informationen, die in Registerkarten gruppiert sind. Die Seitenüberschrift enthält den Namen und die Build-Version des Rezepts, mit dem das Bild erstellt wurde.

Abschnitte und Registerkarten mit Konsolendetails

Abschnitt „Zusammenfassung“

Der Abschnitt mit der Zusammenfassung erstreckt sich über die gesamte Breite der Seite und enthält die folgenden Details. Diese Details werden immer angezeigt.

Status der Richtlinie

Ob die Richtlinie aktiv oder inaktiv ist.

Typ

Der Typ des Ausgabebilds, das Image Builder verteilt, wenn Sie eine neue Image-Version (AMI oder Container-Image) erstellen.

Erstellungsdatum

Der Zeitstempel von der Erstellung der Lebenszyklus-Richtlinie.

Datum geändert

Das letzte Mal, als die Lebenszyklusrichtlinie aktualisiert wurde.

Datum der letzten Ausführung

Das letzte Mal, als die Lebenszyklus-Richtlinie ausgeführt wurde.

IAM role (IAM-Rolle)

Die IAM-Rolle, die Image Builder zur Ausführung von Lebenszyklusaktionen verwendet.

ARN

Der Amazon-Ressourcenname (ARN) der Lifecycle-Policy-Ressource.

Beschreibung

Die Beschreibung für die Lebenszyklus-Richtlinie, sofern eingegeben.

Registerkarte „Regeln“

Auf der Registerkarte Regeln werden die Lebenszyklusregeln angezeigt, die Sie für die angezeigte Richtlinie konfiguriert haben. Die Registerkarte enthält die folgenden Details:

- Name — Der Name der Regel. Diese Namen sind statisch und basieren auf Richtlinienaktionen, die Sie konfigurieren können.
  - `Deprecation rule`
  - `Disable rule`
  - `Deletion rule`

- Regel — Eine kurze Beschreibung der Aktion, die für die Regel konfiguriert ist.
- Regelbedingungen — Listet die Konfiguration für die zugehörige Ressourcenverwaltung, Ausnahmen von der Regel und gegebenenfalls Aufbewahrungseinstellungen auf.

Weitere Informationen zur Regelkonfiguration finden Sie unterWie funktionieren Lebenszyklusregeln.

Registerkarte „Geltungsbereich“

Auf der Registerkarte Geltungsbereich werden die Kriterien für die Ressourcenauswahl angezeigt, die für die angezeigte Richtlinie konfiguriert sind. Die Registerkarte enthält die folgenden Details:

- Filter: ***Filtertyp – Der Filtertyp***, den Sie zur Definition des Bereichs verwendet haben. Der Filtertyp kann einer der folgenden sein:
  - `recipes`— Die Rezepte, die verwendet wurden, um die Images zu erstellen, für die die Lebenszyklusrichtlinie gilt.
  - `tags`— Eine Reihe von Tags, die Image Builder verwendet, um Image-Ressourcen auszuwählen, für die die Lebenszyklusrichtlinie gilt.
- Eine Suchleiste — Sie können die Liste nach Namen filtern, um die Ergebnisse zu optimieren, die auf der Registerkarte angezeigt werden.
- Name — Jede Zeile enthält einen Namen oder ein Tag, das Sie für die Filterkriterien konfiguriert haben.
- Version — Wenn Sie einen Rezeptfilter konfiguriert haben, zeigt Image Builder die Rezeptversion an.

RunLog Registerkarte

Jedes Mal, wenn Sie die Richtlinie für Ihre konfigurierten Ressourcen ausführen, speichert Image Builder Laufzeitdetails. Jede Zeile in der Tabelle steht für eine einzelne Laufzeitinstanz. Die Registerkarte enthält die folgenden Details:

- Ausführungs-ID — Identifiziert die Laufzeitinstanz der Lebenszyklusrichtlinie.
- Ausführungsstatus — Laufzeitstatus, der meldet, ob die Richtlinienaktion derzeit ausgeführt wird, erfolgreich ausgeführt wurde, fehlgeschlagen ist oder abgebrochen wurde.
- Betroffene Ressource — Gibt an, ob die Runtime-Instanz Image-Ressourcen für Lebenszyklusaktionen identifiziert hat.
- Startdatum — Der Zeitstempel, als die Runtime-Instanz gestartet wurde.

- Enddatum — Der Zeitstempel, zu dem die Runtime-Instanz beendet wurde.

Lebenszyklusrichtlinien erstellen

Wenn Sie eine neue EC2 Image Builder Builder-Lebenszyklusrichtlinie erstellen, hängt die Konfiguration davon ab, für welche Art von Image die Richtlinie bestimmt ist. Die API-Aktion zum Erstellen einer Lebenszyklusrichtlinie für AMI-Image-Ressourcen und Container-Image-Ressourcen ist dieselbe (CreateLifecyclePolicy). Die Konfiguration für die Image-Ressourcen und die zugehörigen Ressourcen ist jedoch unterschiedlich. In diesem Abschnitt erfahren Sie, wie Sie Lebenszyklusmanagement-Richtlinien für beide erstellen.

> **Note**
>
> Bevor Sie eine Lebenszyklusrichtlinie erstellen, stellen Sie sicher, dass Sie alle Anforderungen erfüllt habenVoraussetzungen.

Lebenszyklusmanagement-Richtlinien für Image Builder AMI-Image-Ressourcen erstellen

Sie können eine der folgenden Methoden verwenden, um eine AMI-Image-Lebenszyklusrichtlinie mit dem AWS Management Console oder zu erstellen AWS CLI. Sie können auch die CreateLifecyclePolicyAPI-Aktion verwenden. Die zugehörige SDK-Anfrage finden Sie unter dem Link „Siehe auch“ für diesen Befehl in der EC2 Image Builder API-Referenz.

AWS Management Console

Gehen Sie folgendermaßen vor, um eine Lebenszyklusrichtlinie für AMI-Image-Ressourcen in der AWS Management Console zu erstellen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich die Option Lebenszyklusrichtlinien aus.
3. Wählen Sie Lebenszyklusrichtlinie erstellen aus.
4. Konfigurieren Sie die in den folgenden Verfahren beschriebenen Richtlinieneinstellungen.
5. Um die Lebenszyklusrichtlinie zu erstellen, nachdem Sie die Einstellungen konfiguriert haben, wählen Sie Richtlinie erstellen aus.

Konfigurieren Sie die allgemeinen Einstellungen für Ihre Richtlinie.

1. Wählen Sie unter Richtlinientyp die Option AMI aus.
2. Geben Sie den Namen der Richtlinie ein.
3. Geben Sie optional eine Beschreibung für Ihre Lebenszyklusrichtlinie ein.
4. Standardmäßig ist Activate aktiviert. Die Standardeinstellung aktiviert die Lebenszyklusrichtlinie und fügt sie sofort dem Zeitplan hinzu. Um eine Richtlinie zu erstellen, die zunächst deaktiviert ist, können Sie Activate deaktivieren.
5. Wählen Sie die IAM-Rolle aus, die Sie für Lebenszyklus-Richtlinienberechtigungen erstellt haben. Wenn Sie diese Rolle noch nicht erstellt haben, finden Sie Voraussetzungen weitere Informationen unter.

Konfigurieren Sie den Regelbereich für Ihre Richtlinie.

In diesem Abschnitt wird die Ressourcenauswahl für Ihre Lebenszyklusrichtlinie auf der Grundlage des von Ihnen verwendeten Filtertyps konfiguriert.

1. Filtertyp: Rezepte — Um Lebenszyklusregeln auf Image-Ressourcen anzuwenden, die auf dem Rezept basieren, mit dem sie erstellt wurden, wählen Sie bis zu 50 Rezeptversionen für die Richtlinie aus.
2. Filtertyp: Tags — Um Lebenszyklusregeln auf Bildressourcen anzuwenden, die auf Ressourcen-Tags basieren, geben Sie eine Liste mit bis zu 50 Schlüsselwertpaaren ein, anhand derer die Richtlinie einen Abgleich vornehmen soll.

Aktivieren Sie eine oder mehrere der folgenden Lebenszyklusregeln, die auf die Ressourcen angewendet werden sollen, die von der Lebenszyklusrichtlinie ausgewählt werden. Wenn eine Ressource bei der Ausführung der Richtlinie mit mehr als einer Lebenszyklusregel übereinstimmt, führt Image Builder die Regelaktionen in der folgenden Reihenfolge aus: 1) Veraltet, 2) Deaktivieren, 3) Löschen.

Regel verwerfen

Setzt den Image Builder Builder-Image-Ressourcenstatus auf`Deprecated`. Image Builder Builder-Pipelines werden weiterhin für veraltete Images ausgeführt. Sie können optional die Verfallszeit für zugehörige AMIs festlegen, ohne dass Ihre Fähigkeit, neue Instances zu starten, beeinträchtigt wird.

- Anzahl der Einheiten — Geben Sie den ganzzahligen Wert für den Zeitraum an, der nach der Erstellung einer Image-Ressource vergehen muss, bevor sie als markiert wird. `Deprecated`

- Einheit — Wählen Sie den zu verwendenden Zeitraum aus. Der Bereich kann`Days`, `WeeksMonths`, oder sein`Years`.
- AMIs verwerfen — Aktivieren Sie das Kontrollkästchen, um zugehörige Amazon EC2 EC2-AMIs mit einem Verfallsdatum zu kennzeichnen. Die AMIs bleiben verfügbar, und Sie können weiterhin neue Instances von ihnen aus starten.

Regel deaktivieren

Setzt den Image Builder Builder-Image-Ressourcenstatus auf`Disabled`. Dadurch wird verhindert, dass Image Builder Builder-Pipelines für dieses Image ausgeführt werden. Sie können optional das zugehörige AMI deaktivieren, um den Start neuer Instances zu verhindern.

- Anzahl der Einheiten — Geben Sie den ganzzahligen Wert für den Zeitraum an, der nach der Erstellung einer Image-Ressource vergehen muss, bevor sie als markiert wird`Disabled`.
- Einheit — Wählen Sie den zu verwendenden Zeitraum aus. Der Bereich kann`Days`, `WeeksMonths`, oder sein`Years`.
- AMIs deaktivieren — Aktivieren Sie das Kontrollkästchen, um die zugehörigen Amazon EC2 EC2-AMIs zu deaktivieren. Sie können die AMIs nicht mehr verwenden oder neue Instances von ihnen aus starten.

Regel löschen

Löscht die Bildressourcen nach Alter oder Anzahl. Sie definieren den Schwellenwert, der Ihren Anforderungen entspricht. Wenn eine Image Builder Builder-Image-Ressource den Schwellenwert überschreitet, wird sie entfernt. Sie können die Registrierung der zugehörigen AMIs optional aufheben oder die Snapshots für diese AMIs löschen. Sie können auch Tags für Ressourcen angeben, die Sie über den Schwellenwert hinaus behalten möchten.

Wenn Sie die Löschregel nach Alter konfigurieren, löscht Image Builder die Image-Ressource nach einem von Ihnen konfigurierten Zeitraum. Löschen Sie beispielsweise Bildressourcen nach 6 Monaten. Wenn Sie nach Anzahl konfigurieren, behält Image Builder die neueste Anzahl von Bildern bei, die Sie angeben, oder so nahe wie möglich an dieser Anzahl, und löscht frühere Versionen.

- Nach Alter
  - Anzahl der Einheiten — Geben Sie den ganzzahligen Wert für den Zeitraum an, der nach der Erstellung einer Bildressource vergehen muss, bevor sie gelöscht wird.

- Einheit — Wählen Sie den zu verwendenden Zeitraum aus. Der Bereich kann`Days`, `Weeks`Months, oder sein`Years`.
- Mindestens ein Bild pro Rezept beibehalten — Aktivieren Sie das Kontrollkästchen, um die neueste verfügbare Bildressource für jede Rezeptversion beizubehalten, auf die sich diese Regel auswirkt.

Nach Anzahl

- Anzahl Bilder — Geben Sie den ganzzahligen Wert für die Anzahl der aktuellen Bildressourcen an, die für jede Rezeptversion beibehalten werden sollen.

- AMIs deregistrieren — Aktivieren Sie das Kontrollkästchen, um die zugehörigen Amazon EC2-AMIs abzumelden. Sie können die AMIs nicht mehr verwenden oder neue Instances von ihnen aus starten.
- Images, AMIs und Snapshots mit zugehörigen Tags beibehalten — Aktivieren Sie das Kontrollkästchen, um eine Liste von Tags für Image-Ressourcen einzugeben, die Sie behalten möchten. Tags gelten für Image-Ressourcen und Amazon EC2 EC2-AMIs. Sie können bis zu 50 Schlüssel-Wert-Paare eingeben.

Markierungen (optional)

Fügen Sie Ihrer Lebenszyklusrichtlinie Tags hinzu.

AWS CLI

Um eine neue Image Builder Builder-Lebenszyklusrichtlinie zu erstellen, können Sie den create-lifecycle-policy Befehl in der AWS CLI verwenden.

Lebenszyklusmanagement-Richtlinien für Image Builder Builder-Container-Image-Ressourcen erstellen

Sie können eine der folgenden Methoden verwenden, um eine Container-Image-Lebenszyklusrichtlinie mit dem AWS Management Console oder zu erstellen AWS CLI. Sie können auch die CreateLifecyclePolicyAPI-Aktion verwenden. Die zugehörige SDK-Anfrage finden Sie unter dem Link „Siehe auch“ für diesen Befehl in der EC2 Image Builder API-Referenz.

AWS Management Console

Gehen Sie folgendermaßen vor, um eine Lebenszyklusrichtlinie für Container-Image-Ressourcen in der AWS Management Console zu erstellen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich die Option Lebenszyklusrichtlinien aus.
3. Wählen Sie Lebenszyklusrichtlinie erstellen aus.
4. Konfigurieren Sie die in den folgenden Verfahren beschriebenen Richtlinieneinstellungen.
5. Um die Lebenszyklusrichtlinie zu erstellen, nachdem Sie die Einstellungen konfiguriert haben, wählen Sie Richtlinie erstellen aus.

Richtlinienkonfiguration: Allgemeine Einstellungen

Konfigurieren Sie die allgemeinen Einstellungen für Ihre Richtlinie.

1. Wählen Sie unter Richtlinientyp die Option AMI aus.
2. Geben Sie den Namen der Richtlinie ein.
3. Geben Sie optional eine Beschreibung für Ihre Lebenszyklusrichtlinie ein.
4. Standardmäßig ist Activate aktiviert. Die Standardeinstellung aktiviert die Lebenszyklusrichtlinie und fügt sie sofort dem Zeitplan hinzu. Um eine Richtlinie zu erstellen, die zunächst deaktiviert ist, können Sie Activate deaktivieren.
5. Wählen Sie die IAM-Rolle aus, die Sie für Lebenszyklus-Richtlinienberechtigungen erstellt haben. Wenn Sie diese Rolle noch nicht erstellt haben, finden Sie Voraussetzungen weitere Informationen unter.

Konfigurieren Sie den Regelbereich für Ihre Richtlinie.

In diesem Abschnitt wird die Ressourcenauswahl für Ihre Lebenszyklusrichtlinie auf der Grundlage des von Ihnen verwendeten Filtertyps konfiguriert.

1. Filtertyp: Rezepte — Um Lebenszyklusregeln auf Image-Ressourcen anzuwenden, die auf dem Rezept basieren, mit dem sie erstellt wurden, wählen Sie bis zu 50 Rezeptversionen für die Richtlinie aus.
2. Filtertyp: Tags — Um Lebenszyklusregeln auf Bildressourcen anzuwenden, die auf Ressourcen-Tags basieren, geben Sie eine Liste mit bis zu 50 Schlüsselwertpaaren ein, anhand derer die Richtlinie einen Abgleich vornehmen soll.

Regel löschen

Bei Container-Images löscht diese Regel die Image Builder Builder-Container-Image-Ressource. Sie können optional Docker-Images entfernen, die an ECR-Repositorys verteilt wurden, um zu verhindern, dass sie zum Ausführen neuer Container verwendet werden.

Wenn Sie die Löschregel nach Alter konfigurieren, löscht Image Builder die Image-Ressource nach einem von Ihnen konfigurierten Zeitraum. Löschen Sie beispielsweise Bildressourcen nach 6 Monaten. Wenn Sie nach Anzahl konfigurieren, behält Image Builder die neueste Anzahl von Bildern bei, die Sie angeben, oder so nahe wie möglich an dieser Anzahl, und löscht frühere Versionen.

- Nach Alter
  - Anzahl der Einheiten — Geben Sie den ganzzahligen Wert für den Zeitraum an, der nach der Erstellung einer Bildressource vergehen muss, bevor sie gelöscht wird.
  - Einheit — Wählen Sie den zu verwendenden Zeitraum aus. Der Bereich kann`Days`, `WeeksMonths`, oder sein`Years`.
  - Mindestens ein Bild beibehalten — Aktivieren Sie das Kontrollkästchen, um nur die neueste verfügbare Bildressource für jede Rezeptversion beizubehalten, auf die sich diese Regel auswirkt.

  Nach Anzahl

  - Anzahl Bilder — Geben Sie den ganzzahligen Wert für die Anzahl der aktuellen Bildressourcen an, die für jede Rezeptversion beibehalten werden sollen.
- ECR-Container-Images löschen — Aktivieren Sie das Kontrollkästchen, um zugehörige Container-Images zu löschen, die in einem ECR-Repository gespeichert sind. Sie können das Container-Image nicht mehr als Basis verwenden, um neue Images zu erstellen oder neue Container auszuführen.
- Bilder mit zugehörigen Tags beibehalten — Aktivieren Sie das Kontrollkästchen, um eine Liste von Tags für Bildressourcen einzugeben, die Sie behalten möchten.

Markierungen (optional)

Fügen Sie Ihrer Lebenszyklus-Richtlinie Tags hinzu.

AWS CLI

Um eine neue Image Builder Builder-Lebenszyklusrichtlinie zu erstellen, können Sie den create-lifecycle-policy Befehl in der AWS CLI verwenden.

So funktionieren Lebenszyklusmanagementregeln für EC2 Image Builder Builder-Image-Ressourcen

Image-Lebenszyklusrichtlinien verwenden die Lebenszyklusregeln, die Sie zur Implementierung Ihrer gesamten Ressourcenmanagementstrategie definieren. Die von Ihnen definierten Regeln tragen dazu bei, die Aktualität Ihrer verfügbaren Images sicherzustellen und die Kosten für die zugrunde liegende Infrastruktur wie Snapshot-Speicher für Ausgabe-AMIs oder ECR-Repository-Speicher- und Datenübertragungsraten für Container-Images zu minimieren.

Sie können die folgenden Regeltypen für Ihre Richtlinien konfigurieren.

Regel verwerfen

Setzt den Image Builder Builder-Image-Ressourcenstatus auf`Deprecated`. Image Builder Builder-Pipelines werden weiterhin für veraltete Images ausgeführt. Sie können optional die Verfallszeit für zugehörige AMIs festlegen, ohne dass Ihre Fähigkeit, neue Instances zu starten, beeinträchtigt wird.

Wenn ein AMI veraltet ist, wird es bei allgemeinen Suchanfragen ignoriert. Wenn Sie beispielsweise den Amazon EC2 describe-images EC2-Befehl in der ausführen AWS CLI, werden keine veralteten AMIs in der Ergebnismenge zurückgegeben. Sie können jedoch immer noch veraltete AMIs mit ihrer AMI-ID finden.

Diese Regel ist für Container-Images nicht verfügbar.

Regel deaktivieren

Setzt den Image Builder Builder-Image-Ressourcenstatus auf`Disabled`. Dadurch wird verhindert, dass Image Builder Builder-Pipelines für dieses Image ausgeführt werden. Sie können optional das zugehörige AMI deaktivieren, um den Start neuer Instances zu verhindern.

Wenn ein AMI deaktiviert ist, wird es privat und kann nicht zum Starten neuer Instances verwendet werden. Wenn Sie das AMI mit Konten, Organisationen oder Organisationseinheiten geteilt haben, verlieren diese den Zugriff auf Ihr AMI, wenn es privat wird.

Diese Regel ist für Container-Images nicht verfügbar.

Regel löschen

Löscht die Bildressourcen nach Alter oder Anzahl. Sie definieren den Schwellenwert, der Ihren Anforderungen entspricht. Wenn eine Image Builder Builder-Image-Ressource den Schwellenwert überschreitet, wird sie entfernt. Sie können die Registrierung der zugehörigen AMIs optional

aufheben oder die Snapshots für diese AMIs löschen. Sie können auch Tags für Ressourcen angeben, die Sie über den Schwellenwert hinaus behalten möchten.

Bei Container-Images löscht diese Regel die Image Builder Builder-Container-Image-Ressource. Sie können optional Container-Images entfernen, die an ECR-Repositorys verteilt wurden, um zu verhindern, dass sie zum Ausführen neuer Container verwendet werden.

Inhalt

Ausschlussregeln (API/SDK/CLI)

Die folgenden Ausnahmeregeln definieren Ausnahmen von den Lebenszyklusregeln für AMIs. AMIs, die die in den Ausschlussregeln festgelegten Kriterien erfüllen, sind von Lebenszyklusaktionen ausgeschlossen. Ausschlussregeln sind in der nicht verfügbar AWS Management Console.

Die folgenden Begriffe verwenden die API-Notation des `LifecyclePolicyDetailExclusionRules` Datentyps.

Ausschlussregeln

Amis

Enthält die Einstellungen, die in der folgenden Liste aufgeführt sind. `LifecyclePolicyDetailExclusionRulesAmis`

TagMap

Sie können eine Liste mit bis zu 50 Tags bereitstellen, mit denen Lebenszyklusaktionen für jede Art von Ressource übersprungen werden.

Die folgenden Begriffe verwenden die API-Notation des `LifecyclePolicyDetailExclusionRulesAmis` Datentyps.

AMI-Ausschlussregeln

isPublic

Konfiguriert, ob öffentliche AMIs von der Lifecycle-Aktion ausgeschlossen werden.

Zuletzt gestartet

Gibt Konfigurationsdetails für Image Builder an, um die neuesten Ressourcen von Lebenszyklusaktionen auszuschließen.

Regionen

Konfiguriert AWS-Regionen , die von der Lebenszyklusaktion ausgeschlossen sind.

Gemeinsam genutzte Konten

Gibt an AWS-Konten , wessen Ressourcen von der Lebenszyklusaktion ausgeschlossen sind.

TagMap

Führt Tags auf, die von den Lebenszyklusaktionen der AMIs ausgeschlossen werden sollten, die sie enthalten.

Zeigt die Regeldetails für das Lebenszyklusmanagement für eine Richtlinie an

Regeln werden in den Lifecycle Management-Richtlinien definiert, die Sie für Ihre Image Builder Builder-Image-Ressourcen erstellen. In der Konsole enthält die Detailseite der Lebenszyklus-Richtlinie eineRegisterkarte „Regeln“, auf der die Details der Regeln angezeigt werden, die Sie für die Richtlinie konfiguriert haben.

Um die Richtliniendetails in der AWS CLI abzurufen, können Sie den get-lifecycle-policyBefehl ausführen. Die Richtliniendetails in der Antwort enthalten eine Liste der Aktionen (Regeln), die Sie für die Richtlinie definiert haben, einschließlich all Ihrer konfigurierten Einstellungen.

Verwaltung von Build- und Test-Workflows für EC2 Image Builder Builder-Images

Ein Image-Workflow definiert die Reihenfolge der Schritte, die EC2 Image Builder während der Build- und Testphase des Image-Erstellungsprozesses ausführt. Dies ist Teil des gesamten Image Builder Builder-Workflow-Frameworks.

Vorteile des Image-Workflows

- Mit Image-Workflows haben Sie mehr Flexibilität, Transparenz und Kontrolle über den Image-Erstellungsprozess.

- Sie können benutzerdefinierte Workflow-Schritte hinzufügen, wenn Sie Ihr Workflow-Dokument definieren, oder Sie können den Image Builder Builder-Standardworkflow verwenden.
- Sie können Workflow-Schritte ausschließen, die in den Standard-Bild-Workflows enthalten sind.
- Sie können reine Test-Workflows erstellen, bei denen der Erstellungsprozess vollständig übersprungen wird. Sie können dasselbe tun, um reine Build-Workflows zu erstellen.

> **Note**
>
> Sie können einen vorhandenen Workflow nicht ändern, aber Sie können ihn klonen oder eine neue Version erstellen.

Workflow-Framework: Phasen

Für die Anpassung von Image-Workflows ist es wichtig, die Workflow-Phasen zu verstehen, aus denen sich das Workflow-Framework für die Image-Erstellung zusammensetzt.

Das Workflow-Framework für die Image-Erstellung umfasst die folgenden zwei unterschiedlichen Phasen.

1. Build-Phase (Pre-Snapshot) — Während der Build-Phase nehmen Sie Änderungen an der Amazon EC2 EC2-Build-Instance vor, auf der Ihr Basis-Image ausgeführt wird, um die Baseline für Ihr neues Image zu erstellen. Ihr Rezept kann beispielsweise Komponenten enthalten, die eine Anwendung installieren oder die Firewall-Einstellungen des Betriebssystems ändern.

   Nach erfolgreichem Abschluss dieser Phase erstellt Image Builder einen Snapshot oder ein Container-Image, das für die Testphase und darüber hinaus verwendet wird.
2. Testphase (nach dem Snapshot) — Während der Testphase gibt es einige Unterschiede zwischen Images, die AMIs erstellen, und Container-Images. Für AMI-Workflows startet Image Builder eine EC2-Instance aus dem Snapshot, den es als letzten Schritt der Buildphase erstellt hat. Tests werden auf der neuen Instance ausgeführt, um die Einstellungen zu überprüfen und sicherzustellen, dass die Instance wie erwartet funktioniert. Bei Container-Workflows werden die Tests auf derselben Instanz ausgeführt, die für die Erstellung verwendet wurde.

Das Workflow-Framework umfasst auch eine Verteilungsphase. Image Builder verarbeitet jedoch die Workflows für diese Phase.

Zugriff auf Services

Um Image-Workflows auszuführen, benötigt Image Builder die Erlaubnis, Workflow-Aktionen auszuführen. Sie können die AWSServiceRoleForImageBuilderdienstbezogene Rolle angeben, oder Sie können Ihre eigene benutzerdefinierte Rolle für den Dienstzugriff wie folgt angeben.

- Konsole — Wählen Sie im Pipeline-Assistenten Schritt 3: Prozess zur Image-Erstellung definieren die serviceverknüpfte Rolle oder Ihre eigene benutzerdefinierte Rolle aus der IAM-Rollenliste im Bereich Dienstzugriff aus.
- Image Builder Builder-API — Geben Sie in der CreateImageAktionsanforderung die serviceverknüpfte Rolle oder Ihre eigene benutzerdefinierte Rolle als Wert für den `executionRole` Parameter an.

Weitere Informationen zum Erstellen einer Servicerolle finden Sie im AWS Identity and Access Management Benutzerhandbuch unter Erstellen einer Rolle zum Delegieren von Berechtigungen für einen AWS Dienst.

Inhalt

Bild-Workflows auflisten

Auf der Seite mit der Liste der Image-Workflows in der Image Builder-Konsole finden Sie eine Liste der Image-Workflow-Ressourcen, die Sie besitzen oder auf die Sie Zugriff haben, zusammen mit einigen wichtigen Details zu diesen Ressourcen. Sie können Befehle oder Aktionen auch mit der Image Builder Builder-API oder den SDKs verwenden oder AWS CLI um Image-Workflows in Ihrem Konto aufzulisten.

Sie können eine der folgenden Methoden verwenden, um Image-Workflow-Ressourcen aufzulisten, die Ihnen gehören oder auf die Sie Zugriff haben. Informationen zur API-Aktion finden Sie ListWorkflowsin der EC2 Image Builder API-Referenz. Die zugehörige SDK-Anfrage finden Sie unter dem Link „Siehe auch“ auf derselben Seite.

Console

Einzelheiten zum Arbeitsablauf

Zu den Details auf der Listenseite für Image-Workflows in der Image Builder Builder-Konsole gehören:

- Workflow — Der Name der neuesten Version der Image-Workflow-Ressource. In der Image Builder Builder-Konsole ist die Workflow-Spalte mit der Workflow-Detailseite verknüpft.
- Version — Die neueste Version der Image-Workflow-Ressource.
- Typ — Der Workflow-Typ: `BUILD` oder`TEST`.
- Besitzer — Der Besitzer der Workflow-Ressource.
- Erstellungszeit — Datum und Uhrzeit, an dem Image Builder die neueste Version der Image-Workflow-Ressource erstellt hat.
- ARN — Der Amazon-Ressourcenname (ARN) der aktuellen Version der Image-Workflow-Ressource.

Image-Workflows auflisten

Gehen Sie wie folgt vor, um Image-Workflow-Ressourcen in der Image Builder Builder-Konsole aufzulisten:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Image-Workflows aus.

Ergebnisse filtern

Auf der Listenseite mit Bild-Workflows können Sie nach bestimmten Bild-Workflows suchen, um Ihre Ergebnisse zu filtern. Die folgenden Filter sind für Bild-Workflows verfügbar:

`Workflow`

Sie können einen Workflow-Namen ganz oder teilweise eingeben, um die Ergebnisse zu optimieren. Standardmäßig werden alle Workflows in der Liste angezeigt.

`Version`

Sie können eine Versionsnummer ganz oder teilweise eingeben, um die Ergebnisse zu optimieren. Standardmäßig werden alle Versionen in der Liste angezeigt.

`Type`

Sie können nach dem Workflowtyp filtern oder alle Typen anzeigen. Standardmäßig werden alle Workflowtypen in der Liste angezeigt.

- BUILD
- TESTEN

`Owner`

Wenn Sie den Besitzerfilter in der Suchleiste auswählen, zeigt Image Builder eine Liste der Eigentümer für die Image-Workflows in Ihrem Konto an. Sie können einen Besitzer aus der Liste auswählen, um die Ergebnisse zu optimieren. Standardmäßig werden alle Besitzer in der Liste angezeigt.

- AWS-Konto— Das Konto, dem die Workflow-Ressource gehört.
- Amazon — Workflow-Ressourcen, die Amazon besitzt und verwaltet.

AWS CLI

Wenn Sie den list-workflows Befehl in der ausführen AWS CLI, können Sie eine Liste der Image-Workflows abrufen, die Ihnen gehören oder auf die Sie Zugriff haben.

Das folgende Befehlsbeispiel zeigt, wie Sie den list-workflows Befehl ohne Filter verwenden, um alle Image Builder Builder-Image-Workflow-Ressourcen aufzulisten, die Sie besitzen oder auf die Sie Zugriff haben.

Beispiel: Alle Image-Workflows auflisten

```
aws imagebuilder list-workflows
```

Ausgabe:

```
{
 "workflowVersionList": [
  {
    "name": "example-test-workflow",
    "dateCreated": "2023-11-21T22:53:14.347Z",
    "version": "1.0.0",
    "owner": "111122223333",
    "type": "TEST",
```

```
    "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/test/example-test-
workflow/1.0.0"
  },
  {
    "name": "example-build-workflow",
    "dateCreated": "2023-11-20T12:26:10.425Z",
    "version": "1.0.0",
    "owner": "111122223333",
    "type": "BUILD",
    "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/example-
build-workflow/1.0.0"
  }
 ]
}
```

Wenn Sie den list-workflows Befehl ausführen, können Sie Filter anwenden, um die Ergebnisse zu optimieren, wie das folgende Beispiel zeigt. Weitere Informationen zum Filtern Ihrer Ergebnisse finden Sie unter dem Befehl list-workflows in der AWS CLI Befehlsreferenz.

Beispiel: Filter für Build-Workflows

```
aws imagebuilder list-workflows --filters name="type",values="BUILD"
```

Ausgabe:

```
{
 "workflowVersionList": [
  {
    "name": "example-build-workflow",
    "dateCreated": "2023-11-20T12:26:10.425Z",
    "version": "1.0.0",
    "owner": "111122223333",
    "type": "BUILD",
    "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/example-
build-workflow/1.0.0"
  }
 ]
}
```

Erstellen Sie einen Image-Workflow

Wenn Sie einen Image-Workflow erstellen, haben Sie mehr Kontrolle über Ihren Image-Erstellungsprozess. Sie können angeben, welcher Workflow ausgeführt wird, wenn Image Builder Ihr Image erstellt, und welche Workflows ausgeführt werden, wenn das Image getestet wird. Sie können auch einen vom Kunden verwalteten Schlüssel angeben, um Ihre Workflow-Ressourcen zu verschlüsseln. Weitere Informationen zur Verschlüsselung Ihrer Workflow-Ressourcen finden Sie unterVerschlüsselung und Schlüsselverwaltung in EC2 Image Builder.

Für die Image-Erstellung können Sie einen Workflow in der Buildphase und einen oder mehrere Workflows in der Testphase angeben. Je nach Bedarf können Sie die Build- oder Testphase sogar komplett überspringen. Sie konfigurieren die Aktionen, die Ihr Workflow ausführt, in dem YAML-Definitionsdokument, das Ihr Workflow verwendet. Weitere Informationen zur Syntax für Ihr YAML-Dokument finden Sie unter. Erstellen Sie ein YAML-Workflow-Dokument

Für Schritte zum Erstellen eines neuen Build- oder Test-Workflows wählen Sie die Registerkarte aus, die der Umgebung entspricht, die Sie verwenden möchten.

AWS Management Console

Sie können den folgenden Prozess verwenden, um einen neuen Workflow in der Image Builder Builder-Konsole zu erstellen.

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich Image-Workflows aus. Daraufhin wird eine Liste der Image-Workflows angezeigt, die Ihrem Konto gehören oder auf die Ihr Konto Zugriff hat.

   > **Note**
   >
   > In Ihrer Liste werden immer die von Amazon verwalteten Workflow-Ressourcen angezeigt, die Image Builder für seine Standard-Workflows verwendet. Um Details zu diesen Workflows anzuzeigen, können Sie den Workflow-Link auswählen.

3. Um einen neuen Workflow zu erstellen, wählen Sie Image-Workflow erstellen. Daraufhin wird die Seite „Image-Workflow erstellen“ angezeigt.
4. Konfigurieren Sie die Details für Ihren neuen Workflow. Um einen Build-Workflow zu erstellen, wählen Sie oben im Formular die Option Erstellen aus. Um einen Test-Workflow zu erstellen, wählen Sie oben im Formular die Option Test aus. Image Builder füllt die Vorlagenliste auf

der Grundlage dieser Option aus. Alle anderen Schritte sind für Build- und Test-Workflows identisch.

Allgemeines

Der allgemeine Abschnitt enthält Einstellungen, die für Ihre Workflow-Ressource gelten, wie Name und Beschreibung. Zu den allgemeinen Einstellungen gehören die folgenden:

- Name des Image-Workflows (erforderlich) — Der Name für Ihren Image-Workflow. Der Name muss in Ihrem Konto eindeutig sein. Der Name kann bis zu 128 Zeichen lang sein. Zu den gültigen Zeichen gehören Buchstaben, Zahlen-, Leerzeichen und_.
- Version (erforderlich) — Die semantische Version für die zu erstellende Workflow-Ressource (major.minor.patch).
- Beschreibung (optional) — Fügen Sie optional eine Beschreibung für Ihren Workflow hinzu.
- KMS-Schlüssel (optional) — Sie können Ihre Workflow-Ressourcen mit einem vom Kunden verwalteten Schlüssel verschlüsseln. Weitere Informationen finden Sie unter Verschlüsseln Sie Image-Workflows mit einem vom Kunden verwalteten Schlüssel.

Definitionsdokument

Das YAML-Workflow-Dokument enthält die gesamte Konfiguration für Ihren Workflow.

Erste Schritte

- Um mit einer Image Builder Builder-Standardvorlage als Grundlage für Ihren Workflow zu beginnen, wählen Sie die Option Mit Vorlagen beginnen. Diese Option ist standardmäßig ausgewählt. Nachdem Sie aus der Liste Vorlagen ausgewählt haben, welche Vorlage verwendet werden soll, wird die Standardkonfiguration aus der ausgewählten Vorlage in den Inhalt Ihres neuen Workflow-Dokuments kopiert, wo Sie Änderungen vornehmen können.
- Um Ihr Workflow-Dokument von Grund auf neu zu definieren, wählen Sie die Option Von Grund auf neu beginnen. Dadurch wird der Inhalt mit einem kurzen Überblick über einige wichtige Teile des Dokumentformats gefüllt, um Ihnen den Einstieg zu erleichtern.

Das Inhaltsfenster enthält unten eine Statusleiste, in der Warnungen oder Fehler für Ihr YAML-Dokument angezeigt werden. Weitere Informationen zum Erstellen eines YAML-Workflow-Dokuments finden Sie unter. Erstellen Sie ein YAML-Workflow-Dokument

5. Wenn Sie Ihren Workflow abgeschlossen haben oder wenn Sie den Fortschritt speichern und später darauf zurückkommen möchten, wählen Sie Workflow erstellen.

AWS CLI

Bevor Sie den create-workflow Befehl in der ausführen AWS CLI, müssen Sie das YAML-Dokument erstellen, das die gesamte Konfiguration für Ihren Workflow enthält. Weitere Informationen finden Sie unter Erstellen Sie ein YAML-Workflow-Dokument.

Das folgende Beispiel zeigt, wie Sie mit dem Befehl AWS CLI create-workflow einen Build-Workflow erstellen. Der `--data` Parameter bezieht sich auf ein YAML-Dokument, das die Build-Konfiguration für den von Ihnen erstellten Workflow enthält.

Beispiel: Workflow erstellen

```
aws imagebuilder create-workflow --name example-build-workflow --semantic-
version 1.0.0 --type BUILD --data file://example-build-workflow.yml
```

Ausgabe:

```
{
"workflowBuildVersionArn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
build/example-build-workflow/1.0.0/1",
"clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

Das folgende Beispiel zeigt, wie Sie mit dem Befehl create-workflow einen Test-Workflow AWS CLI erstellen. Der `--data` Parameter bezieht sich auf ein YAML-Dokument, das die Build-Konfiguration für den von Ihnen erstellten Workflow enthält.

Beispiel: Test-Workflow erstellen

```
aws imagebuilder create-workflow --name example-test-workflow --semantic-
version 1.0.0 --type TEST --data file://example-test-workflow.yml
```

Ausgabe:

```
{
"workflowBuildVersionArn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
test/example-test-workflow/1.0.0/1",
```

```
"clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

Erstellen Sie ein YAML-Workflow-Dokument

Das Dokument mit der Definition des YAML-Formats konfiguriert Eingabe-, Ausgabe- und Workflow-Schritte für die Erstellungs- und Testphasen des Image-Erstellungsprozesses. Sie können mit Vorlagen beginnen, die standardisierte Schritte enthalten, oder Sie können ganz von vorne beginnen, um Ihren eigenen Workflow zu definieren. Unabhängig davon, ob Sie eine Vorlage verwenden oder ganz von vorne beginnen, können Sie den Workflow an Ihre Bedürfnisse anpassen.

Struktur eines YAML-Workflow-Dokuments

Das YAML-Workflow-Dokument, das Image Builder verwendet, um Image-Build- und Testaktionen durchzuführen, ist wie folgt strukturiert.

- Identifizierung
- Eingabeparameter
- Schritte
- Outputs

Identifizierung

Identifiziert den Workflow eindeutig. Dieser Abschnitt kann die folgenden Attribute enthalten.

Feld	Beschreibung	Typ	Erforderlich
Name	Der Name des Workflow-Dokuments.	String	Nein
description	Die Beschreibung des Dokuments.	String	Nein
schemaVersion		String	Ja

Feld	Beschreibung	Typ	Erforderlich
	Die Version des Dokumentschemas, derzeit 1.0.		

Beispiel

```
---
name: sample-test-image
description: Workflow for a sample image, with extra configuration options exposed
 through workflow parameters.
schemaVersion: 1.0
```

Eingabeparameter

Dieser Teil des Workflow-Dokuments definiert Eingabeparameter, die der Aufrufer angeben kann. Wenn Sie keine Parameter haben, können Sie diesen Abschnitt weglassen. Wenn Sie Parameter angeben, kann jeder Parameter die folgenden Attribute enthalten.

Feld	Beschreibung	Typ	Erforderlich	Beschränkungen
Name	Der Name des Parameters.	String	Ja	
description	Die Beschreibung des Parameters.	String	Nein	
default	Der Standardwert des Parameters, wenn kein Wert angegeben wird. Wenn Sie keinen	Entspricht dem Parameter-Datentyp.	Nein	

Feld	Beschreibung	Typ	Erforderlich	Beschränkungen
	Standardwert in die Parameter definition aufnehmen, ist der Parameter wert zur Laufzeit erforderlich.			
Typ	Der Datentyp des Parameter s. Wenn Sie den Datentyp nicht in die Parameter definition aufnehmen , verwendet der Parameter typ standardm äßig einen zur Laufzeit erforderl ichen Zeichenke ttenwert.	String	Ja	Der Datentyp des Parameters muss einer der folgenden sein: • `string` • `integer` • `boolean` • `stringList`

Beispiel

Geben Sie den Parameter im Workflow-Dokument an.

```
parameters:
  - name: waitForActionAtEnd
    type: boolean
    default: true
    description: "Wait for an external action at the end of the workflow"
```

Verwenden Sie den Parameterwert im Workflow-Dokument.

```
$.parameters.waitForActionAtEnd
```

Schritte

Gibt bis zu 15 Schrittaktionen für den Workflow an. Die Schritte werden in der Reihenfolge ausgeführt, in der sie im Workflow-Dokument definiert sind. Im Falle eines Fehlers wird ein Rollback in umgekehrter Reihenfolge ausgeführt, wobei mit dem fehlgeschlagenen Schritt begonnen und die vorherigen Schritte rückwärts durchgearbeitet werden.

Jeder Schritt kann sich auf die Ausgabe aller Aktionen eines vorherigen Schritts beziehen. Dies wird als Verkettung oder Referenzierung bezeichnet. Um auf die Ausgabe einer Aktion aus einem vorherigen Schritt zu verweisen, können Sie einen JSONPath-Selektor verwenden. Beispielsweise:

```
$.stepOutputs.step-name.output-name
```

Weitere Informationen finden Sie unter Verwenden Sie dynamische Variablen in Ihrem Workflow-Dokument.

> **Note**
>
> Der Schritt selbst hat zwar kein Ausgabeattribut, aber jede Ausgabe einer Schrittaktion ist in `stepOutput` diesem Schritt enthalten.

Jeder Schritt kann die folgenden Attribute enthalten.

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
action	Die Workflow-Aktion, die durch diesen Schritt ausgeführt wird.	String	Ja		Muss eine unterstützte Schrittaktion für Image Builder Builder-Workflow-D

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
					okumente sein.

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
`if`, gefolgt von einer Reihe von bedingten Anweisungen, die den `if` Operator ändern.	Bedingte Anweisungen erweitern den Hauptteil Ihrer Workflow-Schritte um Entscheidungspunkte für den Kontrollfluss.	Diktieren	Nein		Image Builder unterstützt die folgenden bedingten Anweisungen als Modifikatoren für den `if` Operator: • Verzweigungsbedingungen und Modifikatoren:`if`,,,`and`.`or` `not` Verzweigungsbedingungen werden in einer Zeile selbst angegeben. • Vergleichsoperatoren:`booleanEquals` ,`numberEquals` ,`numberGreaterThan` ,`numberGreaterThanEquals` ,

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
					`numberLessThan` `numberLessThanEquals` ,`stringEquals` .
description	Die Beschreibung des Schritts.	String	Nein		Leere Zeichenketten sind nicht zulässig. Falls angegeben, muss die Länge 1—1024 Zeichen betragen.

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
inputs	Enthält Parameter, die die Schrittaktion zur Ausführung benötigt. Sie können Schlüsselwerte als statische Werte oder mit einer JsonPath-Variablen angeben, die in den richtigen Datentyp aufgelöst wird.	Diktieren	Ja		

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
Name	Der Name des Schritts. Dieser Name muss innerhalb des Workflow-Dokuments eindeutig sein.	String	Ja		Die Länge muss zwischen 3 und 128 Zeichen liegen. Kann alphanumerische Zeichen und enthalten . _ Keine Leerzeichen.

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
onFailure	Konfiguriert die Aktion, die ausgeführt werden soll, wenn der Schritt fehlschlägt, wie folgt. Behavior • `Abort`— Der Schritt schlägt fehl, der Workflow schlägt fehl und führt nach dem fehlgeschlagenen Schritt keine weiteren Schritte aus. Wenn Rollback aktiviert ist, beginnt das Rollback mit dem Schritt, der	String	Nein	`Abort`	`Abort` | `Continue`

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
	fehlgeschlagen ist, und wird fortgesetzt, bis alle Schritte, die es zulassen, rückgängig gemacht wurden. • `Continue`— Der Schritt schlägt fehl, führt aber die verbleibenden Schritte nach dem fehlgeschlagenen Schritt weiter aus. In diesem Fall erfolgt kein Rollback.				

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
Rollback aktiviert	Konfiguriert, ob der Schritt zurückgesetzt wird, falls ein Fehler auftritt. Sie können einen statischen booleschen Wert oder eine dynamische JSONPath-Variable verwenden, die in einen booleschen Wert aufgelöst wird.	Boolesch	Nein	`true`	`true` | `false` | oder eine JSONPath-Variable, die in „Wahr“ oder „Falsch“ aufgelöst wird.

Feld	Beschreibung	Typ	Erforderlich	Standardwert	Beschränkungen
timeoutSeconds	Die maximale Zeit in Sekunden, für die der Schritt ausgeführt wird, bevor er fehlschlägt und erneut versucht wird, falls Wiederholungen erforderlich sind.	Ganzzahl	Nein	Hängt gegebenenfalls von der Standardeinstellung ab, die für die Schrittaktion definiert wurde.	Zwischen 1 und 86400 Sekunden (maximal 24 Stunden)

Beispiel

```
steps:
  - name: LaunchTestInstance
    action: LaunchInstance
    onFailure: Abort
    inputs:
      waitFor: "ssmAgent"

  - name: ApplyTestComponents
    action: ExecuteComponents
    onFailure: Abort
    inputs:
      instanceId.$: "$.stepOutputs.LaunchTestInstance.instanceId"

  - name: TerminateTestInstance
    action: TerminateInstance
    onFailure: Continue
```

```
    inputs:
      instanceId.$: "$.stepOutputs.LaunchTestInstance.instanceId"

  - name: WaitForActionAtEnd
    action: WaitForAction
    if:
      booleanEquals: true
      value: "$.parameters.waitForActionAtEnd"
```

Outputs

Definiert Ausgaben für den Workflow. Jede Ausgabe ist ein Schlüssel-Wert-Paar, das den Namen der Ausgabe und den Wert angibt. Sie können Ausgaben verwenden, um zur Laufzeit Daten zu exportieren, die nachfolgende Workflows verwenden können. Dieser Abschnitt ist optional.

Jede Ausgabe, die Sie definieren, enthält die folgenden Attribute.

Feld	Beschreibung	Typ	Erforderlich
Name	Der Name der Ausgabe. Der Name muss für alle Workflows, die Sie in Ihre Pipeline aufnehmen, eindeutig sein.	String	Ja
Wert	Der Wert für die Ausgabe. Der Wert der Zeichenfolge kann eine dynamische Variable sein, z. B. eine Ausgabedatei aus einer Schrittaktion. Weitere Informationen finden Sie unter Verwenden Sie	String	Ja

Feld	Beschreibung	Typ	Erforderlich
	dynamische Variablen in Ihrem Workflow-Dokument.		

Beispiel

Erstellen Sie eine Ausgabebild-ID für das Workflow-Dokument mit der Schrittausgabe aus dem `createProdImage` Schritt.

```
outputs:
  - name: 'outputImageId'
    value: '$.stepOutputs.createProdImage.imageId'
```

Sehen Sie sich die Workflow-Ausgabe im nächsten Workflow an.

```
$.workflowOutputs.outputImageId
```

Unterstützte Schrittaktionen für Ihr Workflow-Dokument

Dieser Abschnitt enthält Details zu den Schrittaktionen, die Image Builder unterstützt.

In diesem Abschnitt verwendete Begriffe

AMI

Amazon Machine Image

ARN

Amazon-Ressourcenname

Unterstützte Aktionen

- BootstrapInstanceForContainer
- CollectImageMetadata
- CollectImageScanFindings

- CreateImage
- ExecuteComponents
- LaunchInstance
- RunCommand
- RunSysPrep
- SanitizeInstance
- TerminateInstance
- WaitForAction

BootstrapInstanceForContainer

Diese Schrittaktion führt ein Dienstskript aus, um die Instanz mit Mindestanforderungen für die Ausführung von Container-Workflows zu booten. Image Builder verwendet die sendCommand in der Systems Manager Manager-API enthaltene, um dieses Skript auszuführen. Weitere Informationen finden Sie unter Befehl AWS Systems Manager ausführen.

> **Note**
>
> Das Bootstrap-Skript installiert die Pakete AWS CLI und Docker, die Voraussetzung dafür sind, dass Image Builder erfolgreich Docker-Container erstellen kann. Wenn Sie diese Schrittaktion nicht einbeziehen, kann der Image-Build fehlschlagen.

Standard-Timeout: 60 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID der Instanz, die	String	Ja		Dies muss die Ausgabein

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
	gebootet werden soll.				stanz-ID des Workflow-Schritts sein, mit dem die Instanz für diesen Workflow gestartet wurde.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
runCommandId	Die ID des Systems ManagersendCommand, der das Bootstrap-Skript auf der Instanz ausgeführt hat.	String
Status	Der vom Systems Manager zurückgegebene StatussendCommand.	String
output	Die Ausgabe wurde vom Systems Manager zurückgegebensendCommand.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: ContainerBootstrapStep
  action: BootstrapInstanceForContainer
  onFailure: Abort
```

```
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.ContainerBootstrapStep.status
```

CollectImageMetadata

Diese Schrittaktion ist nur für Build-Workflows gültig.

EC2 Image Builder führt den Agenten AWS Systems Manager (Systems Manager) auf den EC2-Instances aus, die er startet, um Ihr Image zu erstellen und zu testen. Image Builder sammelt zusätzliche Informationen über die Instanz, die während der Buildphase mit Systems Manager Inventory verwendet wurde. Zu diesen Informationen gehören der Name und die Version des Betriebssystems (OS) sowie die Liste der Pakete und ihrer jeweiligen Versionen, wie von Ihrem Betriebssystem gemeldet.

> **Note**
>
> Diese Schrittaktion funktioniert nur für Images, die AMIs erstellen.

Standard-Timeout: 30 Minuten

Rollback: Image Builder setzt alle Systems Manager Manager-Ressourcen zurück, die in diesem Schritt erstellt wurden.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die Build-Instanz, auf die die Metadaten einstellungen	String	Ja		Dies muss die Ausgabeinstanz-ID aus dem

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
	angewendet werden sollen.				Workflow-Schritt sein, der die Build-Instanz für diesen Workflow gestartet hat.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
osVersion	Der Name und die Version des Betriebssystems, die von der Build-Instanz erfasst wurden.	String
Assoziations-ID	Die Systems Manager Manager-Zuordnungs-ID, die für die Inventarerfassung verwendet wird.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: CollectMetadataStep
  action: CollectImageMetadata
  onFailure: Abort
  inputs:
      instanceId: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe der Schrittaktion im Workflow-Dokument.

```
$.stepOutputs.CollectMetadataStep.osVersion
```

CollectImageScanFindings

Wenn Amazon Inspector für Ihr Konto aktiviert ist und das Scannen von Bildern für Ihre Pipeline aktiviert ist, sammelt diese Schrittaktion die von Amazon Inspector für Ihre Test-Instance gemeldeten Bildscan-Ergebnisse. Diese Schrittaktion ist für Build-Workflows nicht verfügbar.

Standard-Timeout: 120 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID für die Instanz, auf der der Scan ausgeführt wurde.	String	Ja		Dies muss die Ausgabeinstanz-ID des Workflow-Schritts sein, mit dem die Instanz für diesen Workflow gestartet wurde.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
runCommandId	Die ID des Systems ManagersendCommand, der das Skript zum Sammeln von Ergebnissen ausgeführt hat.	String

Ausgabenname	Beschreibung	Typ
Status	Der vom Systems Manager zurückgegebene StatussendCommand.	String
output	Die Ausgabe wurde vom Systems Manager zurückgeg ebensendCommand.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: CollectFindingsStep
  action: CollectImageScanFindings
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.CollectFindingsStep.status
```

CreateImage

Diese Schrittaktion erstellt ein Image von einer laufenden Instance mit der Amazon EC2 `CreateImage` EC2-API. Während des Erstellungsvorgangs wartet die Schrittaktion nach Bedarf, um zu überprüfen, ob die Ressourcen den richtigen Status erreicht haben, bevor sie fortgesetzt wird.

Standard-Timeout: 720 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die Instanz, aus der das neue Bild erstellt werden soll.	String	Ja		Die Instanz für die angegebene Instanz-ID muss sich zu Beginn dieses Schritts in einem `running` Status befinden.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
imageId	Die AMI-ID des erstellten Images.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: CreateImageFromInstance
  action: CreateImage
  onFailure: Abort
  inputs:
      instanceId.$: "i-1234567890abcdef0"
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.CreateImageFromInstance.imageId
```

ExecuteComponents

Bei dieser Schrittaktion werden Komponenten ausgeführt, die im Rezept für das aktuell zu erstellende Bild angegeben sind. Build-Workflows führen Build-Komponenten auf der Build-Instanz aus. Testworkflows führen nur Testkomponenten auf der Testinstanz aus.

Image Builder verwendet die sendCommand in der Systems Manager API, um Komponenten auszuführen. Weitere Informationen finden Sie unter Befehl AWS Systems Manager ausführen.

Standard-Timeout: 720 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID für die Instanz, auf der die Komponenten ausgeführt werden sollen.	String	Ja		Dies muss die Ausgabeinstanz-ID des Workflow-Schritts sein, mit dem die Instanz für diesen Workflow gestartet wurde.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
runCommandId	Die ID des Systems ManagersendCommand, der	String

Ausgabenname	Beschreibung	Typ
	die Komponenten auf der Instanz ausgeführt hat.	
Status	Der vom Systems Manager zurückgegebene StatussendCommand.	String
output	Die Ausgabe wurde vom Systems Manager zurückgeg ebensendCommand.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: ExecComponentsStep
  action: ExecuteComponents
  onFailure: Abort
  inputs:
      instanceId: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe der Schrittaktion im Workflow-Dokument.

```
$.stepOutputs.ExecComponentsStep.status
```

LaunchInstance

Diese Schrittaktion startet eine Instanz in Ihrer AWS-Konto und wartet, bis der Systems Manager Manager-Agent auf der Instanz ausgeführt wird, bevor Sie mit dem nächsten Schritt fortfahren. Die Startaktion verwendet Einstellungen aus Ihren Rezept- und Infrastrukturkonfigurationsressourcen, die mit Ihrem Image verknüpft sind. Der Instance-Typ, der gestartet werden soll, stammt beispielsweise aus der Infrastrukturkonfiguration. Die Ausgabe ist die Instanz-ID der Instance, die sie gestartet hat.

Die `waitFor` Eingabe konfiguriert die Bedingung, die die Anforderung zum Abschluss des Schritts erfüllt.

Standard-Timeout: 60 Minuten

Rollback: Bei Build-Instances führt Rollback die Aktion aus, die Sie in Ihrer Infrastrukturkonfigurationsressource konfiguriert haben. Standardmäßig werden Build-Instances beendet, wenn die Image-Erstellung fehlschlägt. In der Infrastrukturkonfiguration gibt es jedoch eine Einstellung, nach der die Build-Instanz zur Fehlerbehebung beibehalten wird.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
Warte auf	Die Bedingung, auf die gewartet werden muss, bevor der Workflow-Schritt abgeschlossen und mit dem nächsten Schritt fortgefahren wird.	String	Ja		Image Builder unterstützt derzeit `ssmAgent`.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
instanceId	Die Instanz-ID der Instanz, die gestartet wurde.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: LaunchStep
  action: LaunchInstance
  onFailure: Abort
  inputs:
    waitFor: ssmAgent
```

Verwenden Sie die Ausgabe der Schrittaktion im Workflow-Dokument.

```
$.stepOutputs.LaunchStep.instanceId
```

RunCommand

Diese Schrittaktion führt ein Befehlsdokument für Ihren Workflow aus. Image Builder verwendet die sendCommand in der Systems Manager API, um es für Sie auszuführen. Weitere Informationen finden Sie unter Befehl AWS Systems Manager ausführen.

Standard-Timeout: 12 Stunden

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID der Instanz, auf der das Befehlsdokument ausgeführt werden soll.	String	Ja		Dies muss die Ausgabeinstanz-ID des Workflow-Schritts sein, mit dem die Instanz für diesen Workflow gestartet wurde.

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
documentName	Der Name des Systems Manager Manager-Befehlsdokuments, das ausgeführt werden soll.	String	Ja		
Parameter	Eine Liste von Schlüssel-Wert-Paaren für alle Parameter, die das Befehlsdokument benötigt.	<string>wörterbuch <string, list >	Bedingt		
Version des Dokuments	Die auszuführende Version des Befehlsdokuments.	String	Nein	$DEFAULT	

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
runCommandId	Die ID des Systems ManagersendCommand, der das Befehlsdokument auf der Instanz ausgeführt hat.	String

Ausgabenname	Beschreibung	Typ
Status	Der vom Systems Manager zurückgegebene StatussendCommand.	String
output	Die Ausgabe wurde vom Systems Manager zurückgeg ebensendCommand.	Liste von Zeichenfolgen

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: RunCommandDoc
  action: RunCommand
  onFailure: Abort
  inputs:
    documentName: SampleDocument
    parameters:
        osPlatform:
          - "linux"
    instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.RunCommandDoc.status
```

RunSysPrep

Diese Schrittaktion verwendet die sendCommand in der Systems Manager Manager-API, um das `AWSEC2-RunSysprep` Dokument für Windows-Instanzen auszuführen, bevor die Build-Instanz für den Snapshot heruntergefahren wird. Diese Aktionen folgen den AWS bewährten Methoden zum Härten und Reinigen des Images.

Standard-Timeout: 60 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID der Instanz, auf der das `AWSEC2-RunSysprep` Dokument ausgeführt werden soll.	String	Ja		Dies muss die Ausgabeinstanz-ID des Workflow-Schritts sein, der die Instanz für diesen Workflow gestartet hat.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
runCommandId	Die ID des Systems ManagersendCommand, der das `AWSEC2-RunSysprep` Dokument auf der Instanz ausgeführt hat.	String
Status	Der vom Systems Manager zurückgegebene StatussendCommand.	String
output	Die Ausgabe wurde vom Systems Manager zurückgegebensendCommand.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: RunSysprep
  action: RunSysPrep
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.RunSysprep.status
```

SanitizeInstance

Diese Schrittaktion führt das empfohlene Bereinigungsskript für Linux-Instances aus, bevor die Build-Instance für den Snapshot heruntergefahren wird. Das Bereinigungsskript trägt dazu bei, dass das endgültige Image den bewährten Sicherheitsmethoden entspricht und dass Build-Artefakte oder Einstellungen, die nicht auf Ihren Snapshot übertragen werden sollten, entfernt werden. Weitere Informationen zum Skript finden Sie unter. Bereinigung nach dem Build erforderlich Diese Schrittaktion gilt nicht für Container-Images.

Image Builder verwendet die sendCommand in der Systems Manager Manager-API enthaltene, um dieses Skript auszuführen. Weitere Informationen finden Sie unter Befehl AWS Systems Manager ausführen.

Standard-Timeout: 60 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID der Instanz, die bereinigt werden soll.	String	Ja		Dies muss die Ausgabeinstanz-ID des Workflow-Schritts sein, mit dem

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
					die Instanz für diesen Workflow gestartet wurde.

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
runCommandId	Die ID des Systems ManagersendCommand, der das Bereinigungsskript auf der Instanz ausgeführt hat.	String
Status	Der vom Systems Manager zurückgegebene StatussendCommand.	String
output	Die Ausgabe wurde vom Systems Manager zurückgegebensendCommand.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: SanitizeStep
  action: SanitizeInstance
  onFailure: Abort
  inputs:
      instanceId: $.stepOutputs.LaunchStep.instanceId
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.SanitizeStep.status
```

TerminateInstance

Diese Schrittaktion beendet die Instanz mit der Instanz-ID, die als Eingabe übergeben wurde.

Standard-Timeout: 30 Minuten

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
instanceId	Die ID der Instanz, die beendet werden soll.	String	Ja		

Ausgaben: Für diese Schrittaktion gibt es keine Ausgaben.

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: TerminateInstance
  action: TerminateInstance
  onFailure: Continue
  inputs:
      instanceId.$: i-1234567890abcdef0
```

WaitForAction

Diese Schrittaktion unterbricht den laufenden Workflow und wartet auf den Empfang einer externen Aktion von der Image Builder SendWorkflowStepAction Builder-API-Aktion. In diesem Schritt wird ein EventBridge Ereignis mit dem Detailtyp in Ihrem EventBridge Standard-Event-Bus veröffentlicht. `EC2 Image Builder Workflow Step Waiting` Der Schritt kann auch eine SNS-Benachrichtigung senden, wenn Sie einen SNS-Themen-ARN angeben.

Standard-Timeout: 3 Tage

Rollback: Für diese Schrittaktion gibt es kein Rollback.

Eingaben: Die folgende Tabelle enthält unterstützte Eingaben für diese Schrittaktion.

Name der Eingabe	Beschreibung	Typ	Erforderlich	Standard	Beschränkungen
snsTopicArn	Ein optionaler SNS-Thema-ARN, an den eine Benachrichtigung gesendet werden kann, wenn der Workflow-Schritt aussteht.	String	Nein		

Ausgaben: Die folgende Tabelle enthält Ausgaben für diese Schrittaktion.

Ausgabenname	Beschreibung	Typ
action	Die Aktion, die die SendWorkflowStepAction API-Aktion zurückgibt.	Zeichenfolge (`RESUME`oder`STOP`)
Grund	Der Grund für die zurückgegebene Aktion.	String

Beispiel

Geben Sie die Schrittaktion im Workflow-Dokument an.

```
- name: SendEventAndWait
  action: WaitForAction
  onFailure: Abort
  inputs:
    snsTopicArn: arn:aws:sns:us-west-2:111122223333:ExampleTopic
```

Verwenden Sie die Ausgabe des Schrittaktionswerts im Workflow-Dokument.

```
$.stepOutputs.SendEventAndWait.reason
```

Verwenden Sie dynamische Variablen in Ihrem Workflow-Dokument

Sie können dynamische Variablen in Ihren Workflow-Dokumenten verwenden, um Werte darzustellen, die zur Laufzeit Ihres Image-Erstellungsprozesses variieren. Dynamische Variablenwerte werden als JSONPath-Selektoren mit Strukturknoten dargestellt, die die Zielvariable eindeutig identifizieren.

JSONPath, dynamische Workflow-Variablenstruktur

```
$.<document structure>.[<step name>].<variable name>
```

Der erste Knoten nach dem Stamm ($) bezieht sich auf die Workflow-Dokumentstruktur, z. B. oder`stepOutputs`, im Fall von Image Builder Builder-Systemvariablen,`imageBuilder`. Die folgende Liste enthält unterstützte JSONPath-Workflow-Dokumentstrukturknoten.

Knoten für die Dokumentstruktur

- Parameter — Die Workflow-Parameter
- StepOutputs — Ausgaben aus einem Schritt im selben Workflow-Dokument
- WorkflowOutputs — Ausgaben aus einem Workflow-Dokument, das bereits ausgeführt wurde
- imagebuilder - Image Builder Builder-Systemvariablen

Die Knoten `parameters` und die `stepOutputs` Dokumentstruktur enthalten einen optionalen Knoten für den Schrittnamen. Dadurch wird sichergestellt, dass in allen Schritten eindeutige Variablennamen verwendet werden.

Der letzte Knoten im JSONPath ist der Name der Zielvariablen, z. B. `instanceId`

Jeder Schritt kann sich auf die Ausgabe aller Aktionen eines vorherigen Schritts mit diesen dynamischen JSONPath-Variablen beziehen. Dies wird auch als Verkettung oder Referenzierung bezeichnet. Um auf die Ausgabe einer Aktion aus einem vorherigen Schritt zu verweisen, können Sie die folgende dynamische Variable verwenden.

```
$.stepOutputs.step-name.output-name
```

Beispiel

```
- name: ApplyTestComponents
  action: ExecuteComponents
  onFailure: Abort
  inputs:
    instanceId.$: "$.stepOutputs.LaunchTestInstance.instanceId"
```

Verwenden Sie Image Builder Builder-Systemvariablen

Image Builder stellt die folgenden Systemvariablen bereit, die Sie in Ihrem Workflow-Dokument verwenden können:

Variablenname	Beschreibung	Typ	Beispielwert
cloudWatchLogGruppe	Der Name der CloudWatch Protokollgruppe für Ausgabeprotokolle. Format: `/aws/imagebuilder/<recipe-name>`	String	`/aws/imagebuilder/sampleImageRecipe`
cloudWatchLogStreamen	Der Name des CloudWatch Logs-Streams für Ausgabeprotokolle.	String	`1.0.0/1`
collectImageMetadata		Boolesch	`true` | `false`

Variablenname	Beschreibung	Typ	Beispielwert
	Die Einstellung, die Image Builder anweist, ob Instanzmetadaten erfasst werden sollen.		
collectImageScanErgebnisse	Der aktuelle Wert der Einstellung, die es Image Builder ermöglicht, Bildscanergebnisse zu sammeln.	Boolesch	`true` | `false`
imageBuildNumber	Die Build-Versionsnummer des Images.	Ganzzahl	*1*
imageId	Die AMI-ID des Basis-Images.	String	*ami-12345 67890abcdef1*
Name des Bildes	Der Name des Image.	String	*Beispielbild*
ImageType	Der Bildausgabetyp.	String	`AMI` | `Docker`
imageVersionNumber	Die Versionsnummer des Bildes.	String	*1.0.0*

Variablenname	Beschreibung	Typ	Beispielwert
instanceProfileName	Der Name der Instanzprofilrolle, die Image Builder zum Starten von Build- und Testinstanzen verwendet.	String	*SampleImageBuilderInstanceProfileRole*
platform	Die Betriebss ystemplattform des erstellten Images.	String	`Linux` | `Windows` | `MacOS`
S3-Protokolle	Ein JSON-Objekt, das die Konfiguration für die S3-Protokolle enthält, die Image Builder schreibt.	JSON-Objekt	*{'s3logs': {'s3':BucketName' sample-bucket', 's3KeyPrefix': 'ib-logs' }}*
securityGroups	Die Sicherhei tsgruppen-IDs, die für Build- und Test-Inst ances gelten.	Liste [Zeichenfolge]	*[sg-1234567890abcdef1, sg-11112222333344445]*
Quelle ImageARN	Der Amazon-Re ssourcenname (ARN) der Image Builder Builder-Image-Ressource, die der Workflow für Build- und Testphasen verwendet.	String	*arn:aws:imagebuilder: us-east-1 : 111122223333:image/sampleImage/1.0.0/1*

Variablenname	Beschreibung	Typ	Beispielwert
subnetId	Die ID des Subnetzes, in dem die Build- und Test-Instances gestartet werden sollen.	String	*subnet-1234567890abcdef1*
terminateInstanceOnFehlschlag	Der aktuelle Wert der Einstellung, die Image Builder anweist, die Instanz bei einem Fehler zu beenden oder sie zur Fehlerbehebung beizubehalten.	Boolesch	`true` | `false`
WorkflowPhase	Die aktuelle Phase, in der die Workflow-Ausführung ausgeführt wird.	String	`Build` | `Test`
workingDirectory	Der Pfad zum Arbeitsverzeichnis.	String	`/tmp`

Verwenden Sie bedingte Anweisungen in Ihren Workflow-Schritten

Bedingte Anweisungen beginnen mit dem Dokumentattribut „`if`Aussage“. Der ultimative Zweck der `if` Anweisung besteht darin, zu bestimmen, ob die Schrittaktion ausgeführt oder übersprungen werden soll. Wenn die `if` Anweisung zu aufgelöst wird`true`, wird die Schrittaktion ausgeführt. Wenn dies der Fall ist`false`, überspringt Image Builder die Schrittaktion und zeichnet den Schrittstatus `SKIPPED` im Protokoll auf.

Die `if` Anweisung unterstützt Verzweigungsanweisungen (`and`,`or`) und bedingte Modifikatoren (`not`). Sie unterstützt auch die folgenden Vergleichsoperatoren, die Wertvergleiche (gleich, kleiner als, größer als) auf der Grundlage der miteinander verglichenen Datentypen (Zeichenfolge oder Zahl) durchführen.

Unterstützte Vergleichsoperatoren

- `booleanEquals`
- `numberEquals`
- `numberGreaterThan`
- `numberGreaterThanEquals`
- `numberLessThan`
- `numberLessThanEquals`
- `stringEquals`

Regeln für Verzweigungsanweisungen und bedingte Modifikatoren

Die folgenden Regeln gelten für Verzweigungsanweisungen (`and`,`or`) und bedingte Modifikatoren (`not`).

- Verzweigungsanweisungen und bedingte Modifikatoren müssen eigenständig in einer Zeile erscheinen.
- Verzweigungsanweisungen und bedingte Modifikatoren müssen den Ebenenregeln entsprechen.
  - Auf der übergeordneten Ebene kann es nur eine Anweisung geben.
  - Jeder untergeordnete Zweig oder Modifikator beginnt auf einer neuen Ebene.

  Weitere Informationen zu Ebenen finden Sie unterVerschachtelte Ebenen.
- Jede Verzweigungsanweisung muss mindestens eine untergeordnete bedingte Anweisung haben, jedoch nicht mehr als zehn.
- Bedingte Modifikatoren funktionieren nur für eine untergeordnete bedingte Anweisung.

Verschachtelte Ebenen

Bedingte Anweisungen funktionieren auf mehreren Ebenen in einem eigenen Abschnitt. Beispielsweise erscheint das `if` Anweisungsattribut in Ihrem Workflow-Dokument auf derselben Ebene wie der Schrittname und die Aktion. Dies ist die Grundlage der bedingten Anweisung.

Sie können bis zu vier Ebenen bedingter Anweisungen angeben, aber nur eine Anweisung kann auf der übergeordneten Ebene erscheinen. Alle anderen Verzweigungsanweisungen, Bedingungsmodifikatoren oder Bedingungsoperatoren werden von dort aus eingerückt, und zwar ein Einzug pro Ebene.

Die folgende Übersicht zeigt die maximale Anzahl verschachtelter Ebenen für eine bedingte Anweisung.

```
base:
  parent:
    - child (level 2)
        - child (level 3)
            child (level 4)
```

`if` Attribut

Das `if` Attribut gibt die bedingte Anweisung als Dokumentattribut an. Das ist Stufe Null.

Ebene der Eltern

Dies ist die erste Ebene der Verschachtelung für bedingte Anweisungen. Auf dieser Ebene kann es nur eine Anweisung geben. Wenn Sie keine Verzweigungen oder Modifikatoren benötigen, kann dies ein bedingter Operator ohne untergeordnete Anweisungen sein. Diese Ebene verwendet keine Bindestrich-Notation, mit Ausnahme von Bedingungsoperatoren.

Stufen für Kinder

Die Stufen zwei bis vier gelten als Stufen für Kinder. Untergeordnete Anweisungen können Verzweigungsanweisungen, bedingte Modifikatoren oder bedingte Operatoren enthalten.

Beispiel: Verschachtelte Ebenen

Das folgende Beispiel zeigt die maximale Anzahl von Ebenen in einer bedingten Anweisung.

```
if:
  and:                            #first level
    - stringEquals: 'my_string'   #second level
      value: 'my_string'
    - and:                        #also second level
        - numberEquals: '1'       #third level
          value: 1
        - not:                    #also third level
```

```
            stringEquals: 'second_string'      #fourth level
            value: "diff_string"
```

Verschachtelungsregeln

- Jeder Zweig oder Modifikator auf der untergeordneten Ebene beginnt mit einer neuen Ebene.
- Jede Ebene ist eingerückt.
- Es können maximal vier Ebenen vorhanden sein, darunter eine Anweisung, ein Modifikator oder ein Operator auf der übergeordneten Ebene, und bis zu drei weitere Ebenen.

Beispiele

Diese Gruppe von Beispielen zeigt verschiedene Aspekte bedingter Aussagen.

Verzweigung: und

Die `and` Branching-Anweisung arbeitet mit einer Liste von Ausdrücken, die der Verzweigung untergeordnet sind. Alle Ausdrücke müssen als Ergebnis ausgewertet werden. `true` Image Builder wertet die Ausdrücke in der Reihenfolge aus, in der sie in der Liste erscheinen. Wenn ein Ausdruck zu ausgewertet wird`false`, wird die Verarbeitung gestoppt und die Verzweigung wird berücksichtigt. `false`

Im folgenden Beispiel wird als ausgewertet`true`, weil beide Ausdrücke als 0 ausgewertet werden. `true`

```
if:
  and:
    - stringEquals: 'test_string'
      value: 'test_string'
    - numberEquals: 1
      value: 1
```

Verzweigung: oder

Die `or` Branching-Anweisung arbeitet mit einer Liste von Ausdrücken, die der Verzweigung untergeordnet sind, von denen mindestens einer als Ergebnis ausgewertet werden muss. `true` Image Builder wertet die Ausdrücke in der Reihenfolge aus, in der sie in der Liste erscheinen. Wenn ein Ausdruck zu ausgewertet wird`true`, wird die Verarbeitung gestoppt und die Verzweigung wird berücksichtigt. `true`

Im folgenden Beispiel wird als ausgewertet`true`, obwohl der erste Ausdruck dies ist. `false`

```
if:
  or:
    - stringEquals: 'test_string'
      value: 'test_string_not_equal'
    - numberEquals: 1
      value: 1
```

Bedingter Modifikator: nicht

Der `not` bedingte Modifikator negiert die bedingten Anweisungen, die dem Zweig untergeordnet sind.

Im folgenden Beispiel wird ausgewertet, `true` wann der `not` Modifikator die bedingte Anweisung negiert. `stringEquals`

```
if:
  not:
    - stringEquals: 'test_string'
      value: 'test_string_not_equal'
```

Bedingte Anweisung: BooleanEquals

Der `booleanEquals` Vergleichsoperator vergleicht boolesche Werte und gibt true zurück, wenn die booleschen Werte exakt übereinstimmen.

Das folgende Beispiel bestimmt, ob es aktiviert ist. `collectImageScanFindings`

```
if:
  - booleanEquals: true
    value: '$.imagebuilder.collectImageScanFindings'
```

Bedingte Anweisung: StringEquals

Der `stringEquals` Vergleichsoperator vergleicht zwei Zeichenketten und gibt true zurück, wenn die Zeichenketten exakt übereinstimmen. Wenn einer der Werte keine Zeichenfolge ist, konvertiert Image Builder ihn vor dem Vergleich in eine Zeichenfolge.

Im folgenden Beispiel wird die Plattform-Systemvariable verglichen, um festzustellen, ob der Workflow auf einer Linux-Plattform ausgeführt wird.

```
if:
  - stringEquals: 'Linux'
    value: '$.imagebuilder.Platform'
```

Bedingte Aussage: NumberEquals

Der `numberEquals` Vergleichsoperator vergleicht zwei Zahlen und gibt true zurück, wenn die Zahlen gleich sind. Die zu vergleichenden Zahlen müssen eines der folgenden Formate haben.

- Ganzzahl
- Gleitkommazahl
- Eine Zeichenfolge, die dem folgenden Regex-Muster entspricht:. `^-?[0-9]+(\.)?[0-9]+$`

Die folgenden Beispielvergleiche werden alle als ausgewertet. `true`

```
if:
  # Value provider as a number
  numberEquals: 1
  value: '1'

  # Comparison value provided as a string
  numberEquals: '1'
  value: 1

  # Value provided as a string
  numberEquals: 1
  value: '1'

  # Floats are supported
  numberEquals: 5.0
  value: 5.0

  # Negative values are supported
  numberEquals: -1
  value: -1
```

Importieren und Exportieren von Images virtueller Maschinen (VM) mit EC2 Image Builder

Wenn Sie Ihre VM aus ihrer Virtualisierungsumgebung exportieren, erstellt dieser Prozess einen Satz von einer oder mehreren Festplattencontainerdateien, die als Snapshots der Umgebung, Einstellungen und Daten Ihrer VM dienen. Sie können diese Dateien verwenden, um Ihre VM zu importieren und sie als Basisimage für Ihre Image-Rezepte zu verwenden.

Image Builder unterstützt die folgenden Dateiformate für Ihre VM-Festplattencontainer:

- Öffnen Sie Virtualization Archive (OVA)
- Festplatte für virtuelle Maschinen (VMDK)
- Virtuelle Festplatte (VHD/VHDX)
- Raw

Beim Import werden die Festplatten verwendet, um ein Amazon Machine Image (AMI) und eine Image Builder Builder-Image-Ressource zu erstellen, von denen jede als Basis-Image für Ihr benutzerdefiniertes Image-Rezept dienen kann. Die VM-Festplatten müssen für den Import in S3-Buckets gespeichert werden. Alternativ können Sie aus einem vorhandenen EBS-Snapshot importieren.

In der Image Builder Builder-Konsole können Sie das Bild direkt importieren und dann das Ausgabebild oder AMI in Ihren Rezepten verwenden, oder Sie können Importparameter angeben, wenn Sie Ihr Rezept oder Ihre Rezeptversion erstellen. Weitere Informationen zum direkten Import finden Sie unterImportieren Sie eine VM (Konsole). Weitere Informationen zum Importieren als Teil Ihres Bildrezepts finden Sie unterKonfiguration für den VM-Import.

Eine VM in Image Builder importieren (AWS CLI)

Um eine VM von Festplatten in ein AMI zu importieren und eine Image Builder Builder-Image-Ressource zu erstellen, auf die Sie sofort verweisen können, gehen Sie wie folgt vor AWS CLI:

1. Initiieren Sie einen VM-Import mit dem import-image Befehl Amazon EC2 VM Import/Export in der. AWS CLI Notieren Sie sich die Aufgaben-ID, die in der Befehlsantwort zurückgegeben wird. Sie benötigen sie für den nächsten Schritt. Weitere Informationen finden Sie unter Importieren einer VM als Image mithilfe von VM Import/Export im VM Import/Export-Benutzerhandbuch.

2. Erstellen einer CLI-Eingabe-JSON-Datei

Um den in der verwendeten Image Builder import-vm-image Builder-Befehl zu optimieren AWS CLI, erstellen wir eine JSON-Datei, die die gesamte Importkonfiguration enthält, die wir an den Befehl übergeben möchten.

> **Note**
>
> Die Benennungskonvention für die Datenwerte in der JSON-Datei folgt dem Muster, das für die Aktionsanforderungsparameter der Image Builder Builder-API angegeben ist. Informationen zu den API-Befehlsanforderungsparametern finden Sie unter dem ImportVmImageBefehl in der EC2 Image Builder API-Referenz.
> Um die Datenwerte als Befehlszeilenparameter bereitzustellen, beziehen Sie sich auf die Parameternamen, die in der AWS CLI Befehlsreferenz angegeben sind. Auf den Image Builder import-vm-image Builder-Befehl als Optionen.

Im Folgenden finden Sie eine Zusammenfassung der Parameter, die wir in diesem Beispiel angeben:

- name (string, erforderlich) — Der Name für die Image Builder Builder-Bildressource, die als Ausgabe aus dem Import erstellt werden soll.
- semanticVersion <major>(string, erforderlich) — Die semantische Version für das Ausgabebild, die die Version im folgenden Format angibt, mit numerischen Werten an jeder Position, um eine bestimmte Version anzugeben:. <minor>. <patch>. z. B. `1.0.0`. Weitere Informationen zur semantischen Versionierung für Image Builder Builder-Ressourcen finden Sie unter. Semantische Versionsverwaltung
- description (string) — Die Beschreibung des Image-Rezepts.
- platform (string, erforderlich) — Die Betriebssystemplattform für die importierte VM.
- vmImportTaskId (string, required) — Das `ImportTaskId` (AWS CLI) aus dem Amazon EC2 EC2-VM-Importprozess. Image Builder überwacht den Importvorgang, um das von ihm erstellte AMI abzurufen und eine Image Builder Builder-Image-Ressource zu erstellen, die sofort in Rezepten verwendet werden kann.
- clientToken (string, required) — Eine eindeutige Kennung, bei der Groß- und Kleinschreibung berücksichtigt wird, die Sie angeben, um die Idempotenz der Anfrage sicherzustellen. Weitere Informationen finden Sie unter Ensuring Idempotenz in der Amazon EC2 API-Referenz.

- tags (String-Map) — Tags sind Schlüssel-Wert-Paare, die an die Importressourcen angehängt werden. Bis zu 50 Schlüssel-Wert-Paare sind zulässig.

Speichern Sie die Datei unter`import-vm-image.json`, um sie im Image Builder import-vm-image Builder-Befehl zu verwenden.

```
{
    "name": "example-request",
    "semanticVersion": "1.0.0",
    "description": "vm-import-test",
    "platform": "Linux",
    "vmImportTaskId": "import-ami-01ab234567890cd1e",
    "clientToken": "asz1231231234cs3z",
    "tags": {
     "Usage": "VMIE"
    }
}
```

3. Importiert das Bild

Führen Sie den import-vm-image Befehl mit der Datei aus, die Sie als Eingabe erstellt haben:

```
aws imagebuilder import-vm-image --cli-input-json file://import-vm-image.json
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

Verteilen Sie VM-Festplatten aus Ihrem Image build (AWS CLI)

Sie können die Verteilung unterstützter Dateien im VM-Festplattenformat an S3-Buckets in Zielregionen als Teil Ihres regulären Image-Erstellungsprozesses einrichten, indem Sie die Image

Builder Builder-Verteilungskonfigurationen in der AWS CLI verwenden. Weitere Informationen finden Sie unter Erstellen Sie Verteilungseinstellungen für Ausgabe-VM-Festplatten (AWS CLI).

EC2 Image Builder Builder-Ressourcen teilen

EC2 Image Builder ist in AWS Resource Access Manager (AWS RAM) integriert, sodass Sie bestimmte Ressourcen mit anderen AWS-Konto oder über AWS Organizations diese gemeinsam nutzen können. EC2 Image Builder Builder-Ressourcen, die gemeinsam genutzt werden können, sind:

- Komponenten
- Bilder
- Rezepte

Dieser Abschnitt enthält Informationen, die Ihnen helfen, diese EC2 Image Builder Builder-Ressourcen gemeinsam zu nutzen.

Inhalt des Abschnitts

Arbeiten mit gemeinsam genutzten Komponenten, Images und Rezepten in EC2 Image Builder

Die gemeinsame Nutzung von Komponenten, Bildern und Rezepten ermöglicht es Ressourcenbesitzern, Softwarekonfigurationen mit anderen AWS-Konten oder innerhalb einer AWS Organisation zu teilen. Sie können die gemeinsame Nutzung von Ressourcen zentral verwalten und eine Reihe von Konten definieren, mit denen die Konfiguration gemeinsam genutzt werden kann.

In diesem Modell teilt derjenige, dem AWS-Konto die Komponente, das Bild oder das Rezept gehört (Eigentümer), es mit anderen AWS-Konten (Verbrauchern). Verbraucher können ihren Image-Pipelines eine gemeinsam genutzte Komponente zuordnen, um automatisch Aktualisierungen der gemeinsam genutzten Komponente, des Images oder des Rezepts zu nutzen.

Ein Eigentümer einer Komponente, eines Bildes oder eines Rezepts kann diese Ressourcen mit folgenden Personen teilen:

- AWS-Konten Spezifisch innerhalb oder außerhalb seiner Organisation in AWS Organizations.
- Eine Organisationseinheit (OU) innerhalb ihrer Organisation in AWS Organizations.
- Seine gesamte Organisation in AWS Organizations.
- AWS Organizations oder Organisationseinheiten außerhalb ihrer Organisation.

Voraussetzungen für die gemeinsame Nutzung von Komponenten, Bildern und Rezepten

So geben Sie eine Image Builder Builder-Komponente, ein Image oder ein Rezept frei:

- Sie müssen die Komponente, das Bild oder das Rezept in Ihrem besitzen AWS-Konto. Sie können keine Ressourcen teilen, die mit Ihnen geteilt wurden.
- Der Schlüssel AWS Key Management Service (AWS KMS), der verschlüsselten Ressourcen zugeordnet ist, muss explizit mit den Zielkonten, Organisationen oder Organisationseinheiten geteilt werden.
- Um Ihre Image Builder Builder-Ressourcen für AWS Organizations und OUs, die sie verwenden, gemeinsam zu nutzen AWS RAM, müssen Sie die gemeinsame Nutzung aktivieren. Weitere Informationen finden Sie unter Freigabe für AWS Organizations aktivieren im AWS RAM - Benutzerhandbuch.

- Wenn Sie ein mit verschlüsseltes Image AWS KMS auf mehrere Konten in verschiedenen Regionen verteilen, müssen Sie in jeder Zielregion einen KMS-Schlüssel und einen Alias erstellen. Darüber hinaus benötigen die Personen, die Instances in diesen Regionen starten, Zugriff auf den KMS-Schlüssel, der in der Schlüsselrichtlinie angegeben ist.

Die folgenden Ressourcen, die Image Builder aus Ihrem Pipeline-Build erstellt, gelten nicht als Image Builder-Ressourcen, sondern als externe Ressourcen, die Image Builder in Ihrem Konto und an die AWS-Regionen Konten und Organisationen oder Organisationseinheiten (OUs) verteilt, die Sie in Ihrer Verteilungskonfiguration angeben.

- Amazon Machine Images (AMIs)
- Container-Images, die sich in Amazon ECR befinden

Weitere Informationen zu den Verteilungseinstellungen für Ihr AMI finden Sie unterAMI-Verteilungskonfigurationen erstellen und aktualisieren. Weitere Informationen zu den Verteilungseinstellungen für Ihr Container-Image in Amazon ECR finden Sie unterVerteilungseinstellungen für Container-Images erstellen und aktualisieren.

Weitere Informationen zur gemeinsamen Nutzung Ihres AMI mit AWS Organizations und OUs finden Sie unter Teilen eines AMI mit Organisationen oder OUs.

Zugehörige Services

AWS Resource Access Manager

Die gemeinsame Nutzung von Komponenten, Bildern und Rezepten ist in AWS Resource Access Manager (AWS RAM) integriert. AWS RAM ist ein Dienst, mit dem Sie Ihre AWS Ressourcen mit jedem beliebigen AWS Konto oder über dieses teilen können AWS Organizations. Mit können Sie Ressourcen AWS RAM, die Ihnen gehören, gemeinsam nutzen, indem Sie eine gemeinsame Nutzung erstellen. Eine gemeinsame Nutzung gibt die Ressourcen an, die gemeinsam genutzt werden sollen, und die Verbraucher, mit denen sie geteilt werden sollen. Bei Verbrauchern kann es sich um Einzelpersonen AWS-Konten, Organisationseinheiten oder eine gesamte Organisation handeln AWS Organizations.

Weitere Informationen zu AWS RAM finden Sie im AWS RAM Benutzerhandbuch.

Regionsübergreifendes Teilen

Geteilte Komponenten, Bilder und Rezepte können nur in einer bestimmten AWS Region geteilt werden. Wenn Sie diese Ressourcen gemeinsam nutzen, werden sie nicht regionsübergreifend repliziert.

Eine Komponente, ein Bild oder ein Rezept teilen

Um eine Image Builder Builder-Komponente, ein Image oder ein Rezept freizugeben, müssen Sie es zu einer Ressourcenfreigabe hinzufügen. Eine Ressourcenfreigabe ist eine AWS RAM Ressource, mit der Sie Ihre Ressourcen für mehrere AWS Konten gemeinsam nutzen können. Eine Ressourcenfreigabe gibt die Ressourcen an, die gemeinsam genutzt werden sollen, und die Verbraucher, mit denen sie geteilt werden. Um die Komponente, das Bild oder das Rezept zu einer neuen Ressourcenfreigabe hinzuzufügen, müssen Sie zunächst die Ressourcenfreigabe mithilfe der AWS RAM Konsole erstellen.

Wenn Sie Teil einer Organisation sind AWS Organizations und das Teilen innerhalb Ihrer Organisation aktiviert ist, erhalten Nutzer in Ihrer Organisation automatisch Zugriff auf die geteilte Komponente, das Bild oder das Rezept. Andernfalls erhalten Verbraucher eine Einladung zur Teilnahme am Resource Share und erhalten Zugriff auf die gemeinsam genutzte Ressource, nachdem sie die Einladung angenommen haben.

Für die gemeinsame Nutzung Ihrer Ressourcen stehen die folgenden Optionen zur Verfügung:.

Option 1: Erstellen Sie eine RAM-Ressourcenfreigabe

Wenn Sie eine RAM-Ressourcenfreigabe erstellen, können Sie eine Komponente, ein Bild oder ein Rezept, das Sie besitzen, in einem einzigen Schritt gemeinsam nutzen. Verwenden Sie eine der folgenden Methoden, um Ihre Ressourcenfreigabe zu erstellen:

- Konsole

  Informationen zum Erstellen Ihrer Ressourcenfreigabe mithilfe der AWS RAM Konsole finden Sie im AWS RAM Benutzerhandbuch unter Teilen von AWS Ressourcen, die Ihnen gehören.

- AWS CLI

  Um Ihre Resource Share über die AWS RAM Befehlszeilenschnittstelle zu erstellen, führen Sie den create-resource-share Befehl in der aus AWS CLI.

Option 2: Wenden Sie eine Ressourcenrichtlinie an und stufen Sie zu einer RAM-Ressourcenfreigabe herauf

Die zweite Option für die gemeinsame Nutzung Ihrer Ressourcen umfasst zwei Schritte, bei denen Befehle AWS CLI für beide ausgeführt werden. Im ersten Schritt werden Image Builder Builder-Befehle verwendet AWS CLI , um ressourcenbasierte Richtlinien auf die gemeinsam genutzte Ressource anzuwenden. Im zweiten Schritt wird die Ressource mithilfe des promote-resource-share-created-from-policy AWS RAM Befehls in zu einer RAM-Ressourcenfreigabe heraufgestuft, AWS CLI um sicherzustellen, dass die Ressource für alle Prinzipale sichtbar ist, mit denen Sie sie gemeinsam genutzt haben.

1. Wenden Sie die Ressourcenrichtlinie an

   Um die Ressourcenrichtlinie erfolgreich anwenden zu können, müssen Sie sicherstellen, dass das Konto, mit dem Sie Inhalte teilen, über Zugriffsberechtigungen für alle zugrunde liegenden Ressourcen verfügt.

   Wählen Sie für den entsprechenden Befehl die Registerkarte aus, die Ihrem Ressourcentyp entspricht.

   Image

   Sie können eine Ressourcenrichtlinie auf ein Bild anwenden, damit andere es als Basis-Image in ihren Rezepten verwenden können.

   Führen Sie den put-image-policy Image Builder Builder-Befehl in der aus AWS CLI, um die AWS Prinzipale zu identifizieren, mit denen das Image geteilt werden soll.

```
aws imagebuilder put-image-policy --image-arn arn:aws:imagebuilder:us-
west-2:123456789012:image/my-example-image/2019.12.03/1 --policy
 '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal":
 { "AWS": [ "123456789012" ] }, "Action": ["imagebuilder:GetImage",
 "imagebuilder:ListImages"], "Resource": [ "arn:aws:imagebuilder:us-
west-2:123456789012:image/my-example-image/2019.12.03/1" ] } ] }'
```

   Component

   Sie können eine Ressourcenrichtlinie auf eine Build- oder Testkomponente anwenden, um die kontenübergreifende gemeinsame Nutzung zu ermöglichen. Dieser Befehl gibt anderen Konten die Erlaubnis, Ihre Komponente in ihren Rezepten zu verwenden. Um die

Ressourcenrichtlinie erfolgreich anwenden zu können, müssen Sie sicherstellen, dass das Konto, mit dem Sie etwas teilen, berechtigt ist, auf alle Ressourcen zuzugreifen, auf die von der gemeinsam genutzten Komponente verwiesen wird, z. B. auf Dateien, die in privaten Repositorys gehostet werden.

Führen Sie den put-component-policy Image Builder Builder-Befehl in der aus AWS CLI, um die AWS Prinzipale zu identifizieren, mit denen die Komponente gemeinsam genutzt werden soll.

```
aws imagebuilder put-component-policy --component-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/my-example-component/2019.12.03/1 --policy
 '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal":
 { "AWS": [ "123456789012" ] }, "Action": [ "imagebuilder:GetComponent",
 "imagebuilder:ListComponents" ], "Resource": [ "arn:aws:imagebuilder:us-
west-2:123456789012:component/my-example-component/2019.12.03/1" ] } ] }'
```

Image recipe

Sie können eine Ressourcenrichtlinie auf ein Image-Rezept anwenden, um die kontoübergreifende gemeinsame Nutzung zu ermöglichen. Dieser Befehl gibt anderen Konten die Erlaubnis, dein Rezept zu verwenden, um Bilder in ihren Konten zu erstellen. Um die Ressourcenrichtlinie erfolgreich anwenden zu können, müssen Sie sicherstellen, dass das Konto, mit dem Sie Inhalte teilen, berechtigt ist, auf alle Ressourcen zuzugreifen, auf die das Rezept verweist, z. B. das Basis-Image oder ausgewählte Komponenten.

Führen Sie den put-image-recipe-policy Image Builder Builder-Befehl in der aus AWS CLI, um die AWS Prinzipale zu identifizieren, mit denen das Image geteilt werden soll.

```
aws imagebuilder put-image-recipe-policy --image-recipe-arn
 arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-
image-recipe/2019.12.03 --policy '{ "Version": "2012-10-17", "Statement":
 [ { "Effect": "Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":
 [ "imagebuilder:GetImageRecipe", "imagebuilder:ListImageRecipes" ], "Resource":
 [ "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-image-
recipe/2019.12.03" ] } ] }'
```

Container recipe

Sie können eine Ressourcenrichtlinie auf ein Container-Rezept anwenden, um die kontenübergreifende gemeinsame Nutzung zu ermöglichen. Dieser Befehl gibt anderen

Konten die Erlaubnis, Ihr Rezept zu verwenden, um Bilder in ihren Konten zu erstellen. Um die Ressourcenrichtlinie erfolgreich anwenden zu können, müssen Sie sicherstellen, dass das Konto, mit dem Sie Inhalte teilen, berechtigt ist, auf alle Ressourcen zuzugreifen, auf die das Rezept verweist, z. B. das Basis-Image oder ausgewählte Komponenten.

Führen Sie den put-container-recipe-policy Image Builder Builder-Befehl in der aus AWS CLI, um die AWS Prinzipale zu identifizieren, mit denen das Image geteilt werden soll.

```
aws imagebuilder put-container-recipe-policy --container-recipe-arn
 arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-example-
container-recipe/2021.12.03 --policy '{ "Version": "2012-10-17", "Statement":
 [ { "Effect": "Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":
 [ "imagebuilder:GetContainerRecipe", "imagebuilder:ListContainerRecipes" ],
 "Resource": [ "arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-
example-container-recipe/2021.12.03" ] } ] }'
```

> **Note**
>
> Um die richtigen Richtlinien für das Teilen und Aufheben der gemeinsamen Nutzung einer Ressource festzulegen, muss der Eigentümer der Ressource über entsprechende Berechtigungen verfügen`imagebuilder:put*`.

2. Als gemeinsam genutzte RAM-Ressource heraufstufen

   Um sicherzustellen, dass die Ressource für alle Prinzipale sichtbar ist, mit denen Sie sie gemeinsam genutzt haben, führen Sie den promote-resource-share-created-from-policy AWS RAM Befehl in der AWS CLI aus.

Aufheben der Freigabe einer gemeinsam genutzten Komponente, eines Images oder eines Rezepts

Um die gemeinsame Nutzung einer Komponente, eines Bilds oder einer Rezeptur, deren Eigentümer Sie sind, rückgängig zu machen, müssen Sie sie aus der Ressourcenfreigabe entfernen. Sie können dies mit der AWS Resource Access Manager Konsole oder dem AWS CLI tun.

> **Note**
>
> Um die Freigabe einer Komponente, eines Images oder eines Rezepts rückgängig zu machen, darf der Verbraucher keine Abhängigkeiten von ihnen haben. Der Verbraucher muss alle Abhängigkeiten von den gemeinsam genutzten Ressourcen entfernen, bevor der Eigentümer die gemeinsame Nutzung der Ressourcen rückgängig machen kann.

Um die gemeinsame Nutzung einer Komponente, eines Images oder eines Rezepts, dessen Eigentümer Sie sind, mithilfe der Konsole rückgängig zu machen AWS Resource Access Manager

Siehe Aktualisieren einer Ressourcenfreigabe im AWS RAM -Benutzerhandbuch.

Um die gemeinsame Nutzung einer Komponente, eines Bilds oder einer Rezeptur, deren Eigentümer Sie sind, rückgängig zu machen, verwenden Sie die AWS CLI

Verwenden Sie den disassociate-resource-share Befehl, um die gemeinsame Nutzung der Ressource zu beenden.

Identifizieren einer gemeinsam genutzten Komponente, eines gemeinsam genutzten Bildes oder Rezepts

Besitzer und Verbraucher können gemeinsam genutzte Komponenten, Bilder und Image-Rezepte mithilfe von Image Builder Builder-Befehlen in der identifizieren AWS CLI.

Identifizieren Sie eine gemeinsam genutzte Komponente

Führen Sie den Befehl list-components aus, um eine Liste der Komponenten zu erhalten, die Sie besitzen, und der Komponenten, die mit Ihnen gemeinsam genutzt werden. Der Befehl get-component zeigt die AWS-Konto ID des Eigentümers der Komponente an.

Identifizieren Sie ein geteiltes Bild

Führen Sie den Befehl list-images aus, um eine Liste der Bilder, die Sie besitzen, und der Bilder, die mit Ihnen geteilt wurden, abzurufen. Der Befehl get-image zeigt die AWS-Konto ID des Image-Besitzers an.

Identifizieren Sie ein gemeinsam genutztes Container-Image

Führen Sie den Befehl list-images aus, um eine Liste der Bilder, die Sie besitzen, und der Bilder, die mit Ihnen geteilt wurden, abzurufen. Der Befehl get-image zeigt die AWS-Konto ID des Image-Besitzers an.

Identifizieren Sie ein Rezept für ein geteiltes Image

Führen Sie den list-image-recipesBefehl aus, um eine Liste der Bildrezepte, die Sie besitzen, und der Bildrezepte, die mit Ihnen geteilt wurden, abzurufen. Der get-image-recipeBefehl zeigt die AWS-Konto ID des Besitzers des Bildrezepts an.

Identifizieren Sie ein Rezept für einen gemeinsam genutzten Container

Führen Sie den list-container-recipesBefehl aus, um eine Liste der Container-Rezepte, die Sie besitzen, und der Container-Rezepte, die mit Ihnen geteilt wurden, abzurufen. Der get-container-recipeBefehl zeigt die AWS-Konto ID des Besitzers des Container-Rezepts an.

Gemeinsam genutzte Komponenten-, Bild- und Rezeptberechtigungen

Berechtigungen für Besitzer

Besitzer können eine gemeinsam genutzte Komponente, ein Bild oder ein gemeinsames Bildrezept erst löschen, wenn sie nicht mehr gemeinsam genutzt werden. Ein Eigentümer kann die gemeinsame Nutzung dieser Ressourcen erst rückgängig machen, wenn keiner der Verbraucher von ihnen abhängig ist.

Berechtigungen für Konsumenten

Verbraucher können eine Komponente, ein Bild oder ein Bildrezept lesen, sie jedoch in keiner Weise ändern. Sie können diese Ressourcen nicht einsehen oder ändern, wenn sie anderen Benutzern oder dem Eigentümer der Ressource gehören. Verbraucher können gemeinsam genutzte Komponenten und Bilder in Image-Rezepten verwenden, um benutzerdefinierte Images zu erstellen. Verbraucher können Rezepte für gemeinsam genutzte Bilder verwenden, um ihre eigenen benutzerdefinierten Bilder zu erstellen.

Fakturierung und Messung

Die Nutzung von EC2 Image Builder ist kostenlos.

Ressourcenlimits

Gemeinsam genutzte Komponenten, Images und Image-Rezepte werden nur auf die entsprechenden Ressourcenlimits des Besitzers angerechnet. Die Ressourcenlimits der Verbraucher werden nicht durch die Ressourcen beeinflusst, die mit ihnen geteilt wurden.

Schlagwort: EC2 Image Builder Builder-Ressourcen

Das Markieren Ihrer Ressourcen kann nützlich sein, um Ressourcenkosten oder andere Kategorien zu filtern und zu verfolgen. Sie können den Zugriff auch anhand von Tags steuern. Weitere Informationen zur tagbasierten Autorisierung finden Sie unter Autorisierung auf Basis von Image Builder Builder-Tags

Image Builder unterstützt die folgenden dynamischen Tags:

- `- {{imagebuilder:buildDate}}`

  Gibt zum Zeitpunkt der Erstellung das Datum und die Uhrzeit der Erstellung an.

- `- {{imagebuilder:buildVersion}}`

  Wird in eine Build-Version aufgelöst, bei der es sich um eine Zahl handelt, die sich am Ende eines Image Builder Builder-Amazon-Ressourcennamens (ARN) befindet. `"arn:aws:imagebuilder:us-west-2:123456789012:component/myexample-component/2019.12.02/1"`Zeigt die Build-Version beispielsweise als`1`.

Damit Sie den Überblick über die von Ihnen verteilten Amazon Machine Images (AMIs) behalten, fügt Image Builder Ihren Ausgabe-AMIs automatisch die folgenden Tags hinzu.

- `"CreatedBy":"EC2 Image Builder"`
- `"Ec2ImageBuilderArn":"arn:aws:imagebuilder:`*`us-west-2`*`:`*`123456789012`*`:image/`*`simple-recipe-linux`*`/`*`1.0.0/10`*`"`. Dieses Tag enthält den ARN der Image Builder Builder-Image-Ressource, die zur Erstellung des AMI verwendet wurde.

Inhalt

- Listet alle Tags für eine bestimmte Ressource auf (AWS CLI)

Kennzeichnen Sie eine Ressource (AWS CLI)

Das folgende Beispiel zeigt, wie Sie einen imagebuilder CLI-Befehl verwenden, um eine Ressource in EC2 Image Builder hinzuzufügen und zu kennzeichnen. Sie müssen die Tags `resourceArn` und die Tags angeben, die darauf angewendet werden sollen.

Der `tag-resource.json` Inhalt des Beispiels lautet wie folgt:

```
{
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-example-pipeline",
    "tags": {
        "KeyName": "KeyValue"
    }
}
```

Führen Sie den folgenden Befehl aus, der auf die vorherige `tag-resource.json` Datei verweist.

```
aws imagebuilder tag-resource --cli-input-json file://tag-resource.json
```

Entferne die Markierung einer Ressource ()AWS CLI

Das folgende Beispiel zeigt, wie ein imagebuilder CLI-Befehl verwendet wird, um ein Tag aus einer Ressource zu entfernen. Sie müssen die Schlüssel `resourceArn` und die Schlüssel angeben, um das Tag zu entfernen.

Der `untag-resource.json` Inhalt des Beispiels lautet wie folgt:

```
{
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-example-pipeline",
    "tagKeys": [
        "KeyName"
    ]
}
```

Führen Sie den folgenden Befehl aus, der auf die vorherige `untag-resource.json` Datei verweist.

```
aws imagebuilder untag-resource --cli-input-json file://untag-resource.json
```

Listet alle Tags für eine bestimmte Ressource auf (AWS CLI)

Das folgende Beispiel zeigt, wie Sie einen imagebuilder CLI-Befehl verwenden, um alle Tags für eine bestimmte Ressource aufzulisten.

```
aws imagebuilder list-tags-for-resource --resource-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-pipeline/my-example-pipeline
```

EC2 Image Builder Builder-Ressourcen löschen

Ihre Image Builder Builder-Umgebung muss, genau wie Ihr Zuhause, regelmäßig gewartet werden, damit Sie das finden, was Sie benötigen, und Ihre Aufgaben erledigen können, ohne sich durch Unordnung zu wühlen. Stellen Sie sicher, dass Sie die temporären Ressourcen, die Sie zu Testzwecken erstellt haben, regelmäßig bereinigen. Andernfalls könnten Sie diese Ressourcen vergessen und sich später nicht mehr daran erinnern, wofür sie verwendet wurden. Bis dahin ist möglicherweise nicht klar, ob Sie sie sicher loswerden können.

Durch das Löschen von Ressourcen werden keine Amazon EC2 EC2-AMIs oder Amazon ECR-Container-Images gelöscht, die während des Image-Build-Prozesses erstellt wurden. Sie müssen diese separat bereinigen, indem Sie die entsprechenden Amazon EC2- oder Amazon ECR-Konsolenaktionen, APIs oder AWS CLI Befehle verwenden.

> **Tip**
>
> Um Abhängigkeitsfehler beim Löschen von Ressourcen zu vermeiden, stellen Sie sicher, dass Sie Ihre Ressourcen in der folgenden Reihenfolge löschen:
>
> 1. Image-Pipeline
> 2. Bildrezept
> 3. Alle verbleibenden Ressourcen

Löschen Sie Ressourcen mithilfe der AWS Management Console

Gehen Sie folgendermaßen vor, um eine Image-Pipeline und ihre Ressourcen zu löschen:

Löschen Sie die Pipeline

1. Um eine Liste der Build-Pipelines anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Image-Pipelines aus.
2. Aktivieren Sie das Kontrollkästchen neben Pipelinename, um die Pipeline auszuwählen, die Sie löschen möchten.
3. Wählen Sie oben im Bereich Image-Pipelines im Menü Aktionen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie den Text `Delete` in das Feld ein und wählen Sie „Löschen“.

Lösche das Rezept

1. Um eine Liste der unter Ihrem Konto erstellten Rezepte anzuzeigen, wählen Sie im Navigationsbereich Bildrezepte aus.
2. Aktivieren Sie das Kontrollkästchen neben Rezeptname, um das Rezept auszuwählen, das Sie löschen möchten.
3. Wählen Sie oben im Fenster Bildrezepte im Menü Aktionen die Option Rezept löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie den `Delete` Text in das Feld ein und wählen Sie „Löschen“.

Löschen Sie die Infrastrukturkonfiguration

1. Um eine Liste der unter Ihrem Konto erstellten Infrastrukturkonfigurationen anzuzeigen, wählen Sie im Navigationsbereich Infrastrukturkonfiguration aus.
2. Aktivieren Sie das Kontrollkästchen neben Konfigurationsname, um die Infrastrukturkonfiguration auszuwählen, die Sie löschen möchten.
3. Wählen Sie oben im Bereich Infrastrukturkonfigurationen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie den Text `Delete` in das Feld ein und wählen Sie Löschen.

Löschen Sie die Verteilungseinstellungen

1. Um eine Liste der unter Ihrem Konto erstellten Verteilungseinstellungen anzuzeigen, wählen Sie im Navigationsbereich Verteilungseinstellungen aus.

2. Aktivieren Sie das Kontrollkästchen neben dem Namen der Konfiguration, um die Verteilungseinstellungen auszuwählen, die Sie für dieses Tutorial erstellt haben.
3. Wählen Sie oben im Bereich Verteilungseinstellungen die Option Löschen aus.
4. Um den Löschvorgang zu bestätigen, geben Sie den Text `Delete` in das Feld ein und wählen Sie Löschen.

Ein Image löschen

1. Um eine Liste der unter Ihrem Konto erstellten Bilder zu sehen, wählen Sie im Navigationsbereich Bilder aus.
2. Wählen Sie die Image-Version für das Bild aus, das Sie entfernen möchten. Dadurch wird die Seite mit den Image-Build-Versionen geöffnet.
3. Aktivieren Sie das Kontrollkästchen neben der Version für jedes Image, das Sie löschen möchten. Sie können mehrere Image-Versionen gleichzeitig auswählen.
4. Wählen Sie oben im Bereich Image-Build-Versionen die Option Version löschen aus.
5. Um den Löschvorgang zu bestätigen, geben Sie den Text `Delete` in das Feld ein und wählen Sie Löschen.

Löschen Sie eine Image-Pipeline mit dem AWS CLI

Die folgenden Beispiele zeigen, wie Sie Image Builder Builder-Ressourcen mithilfe von löschen AWS CLI. Wie bereits erwähnt, müssen Ressourcen in der folgenden Reihenfolge gelöscht werden, um Abhängigkeitsfehler zu vermeiden:

1. Image-Pipeline
2. Bildrezept
3. Alle verbleibenden Ressourcen

Löschen Sie die Image-Pipeline (AWS CLI)

Das folgende Beispiel zeigt, wie eine Image-Pipeline gelöscht wird, indem ihr ARN angegeben wird.

```
aws imagebuilder delete-image-pipeline --image-pipeline-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-pipeline/my-example-pipeline
```

Rezept für das Löschen eines Bilds (AWS CLI)

Das folgende Beispiel zeigt, wie Sie ein Bildrezept löschen, indem Sie seinen ARN angeben.

```
aws imagebuilder delete-image-recipe --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-example-recipe/2019.12.03
```

Löschen einer Infrastrukturkonfiguration

Das folgende Beispiel zeigt, wie eine Infrastrukturkonfigurationsressource gelöscht wird, indem ihr ARN angegeben wird.

```
aws imagebuilder delete-infrastructure-configuration --infrastructure-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-
infrastructure-configuration
```

Löschen Sie die Verteilungseinstellungen

Das folgende Beispiel zeigt, wie eine Ressource für Verteilungseinstellungen gelöscht wird, indem ihr ARN angegeben wird.

```
aws imagebuilder delete-distribution-configuration --distribution-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-
distribution-configuration
```

Ein Image löschen

Das folgende Beispiel zeigt, wie eine Image-Build-Version gelöscht wird, indem ihr ARN angegeben wird.

```
aws imagebuilder delete-image --image-build-version-arn arn:aws:imagebuilder:us-
west-2:123456789012:image/my-example-image/2019.12.02/1
```

Löschen einer Komponente

Das folgende Beispiel zeigt, wie Sie mit einem imagebuilder CLI-Befehl eine Build-Version einer Komponente löschen, indem Sie ihren ARN angeben.

```
aws imagebuilder delete-component --component-build-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:component/my-example-
component/2019.12.02/1
```

Important

Stellen Sie sicher, dass es keine Rezepte gibt, die in irgendeiner Weise auf die Build-Version der Komponente verweisen, bevor Sie sie löschen. Andernfalls kann es zu Pipelineausfällen kommen.

EC2 Image Builder Builder-Pipelines mithilfe der Konsole verwalten

Image Builder Builder-Image-Pipelines bieten ein Automatisierungsframework für die Erstellung und Verwaltung benutzerdefinierter AMIs und Container-Images. Pipelines bieten die folgenden Funktionen:

- Stellen Sie das Basisimage, die Komponenten für den Aufbau und das Testen, die Infrastrukturkonfiguration und die Verteilungseinstellungen zusammen.
- Vereinfachen Sie die Planung automatisierter Wartungsprozesse mithilfe des Assistenten `Schedule builder` in der Konsole oder durch Eingabe von Cron-Ausdrücken für wiederkehrende Aktualisierungen Ihrer Images.
- Aktivieren Sie die Änderungserkennung für das Basis-Image und die Komponenten, um geplante Builds automatisch zu überspringen, wenn keine Änderungen vorgenommen wurden.
- Ermöglichen Sie regelbasierte Automatisierung über Amazon. EventBridge

> **Note**
>
> Weitere Informationen zur Verwendung der EventBridge API zum Anzeigen oder Ändern von Regeln finden Sie in der Amazon EventBridge API-Referenz. Weitere Informationen zur Verwendung von EventBridge events Befehlen AWS CLI zum Anzeigen oder Ändern von Regeln finden Sie in der AWS CLI Befehlsreferenz unter Ereignisse.

Inhalt

Pipeline-Details auflisten und anzeigen

In diesem Abschnitt werden die verschiedenen Möglichkeiten beschrieben, wie Sie Informationen finden und Details für Ihre EC2 Image Builder Builder-Image-Pipelines anzeigen können.

Einzelheiten zur Pipeline

- Image-Rohrleitungen auflisten ()AWS CLI
- Rufen Sie die Details der Image-Pipeline ab ()AWS CLI

Image-Rohrleitungen auflisten ()AWS CLI

Das folgende Beispiel zeigt, wie Sie den list-image-pipelines Befehl in verwenden, AWS CLI um alle Ihre Image-Pipelines aufzulisten.

```
aws imagebuilder list-image-pipelines
```

Rufen Sie die Details der Image-Pipeline ab ()AWS CLI

Das folgende Beispiel zeigt, wie Sie den get-image-pipeline Befehl in verwenden AWS CLI , um die Details zu einer Image-Pipeline über ihren ARN abzurufen.

```
aws imagebuilder get-image-pipeline --image-pipeline-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-pipeline/my-example-pipeline
```

AMI-Image-Pipelines erstellen und aktualisieren

Sie können AMI-Image-Pipelines über die Image Builder-Konsole, über die Image Builder Builder-API oder mit imagebuilder Befehlen in der AWS CLI einrichten, konfigurieren und verwalten. Sie können den Assistenten zur Erstellung einer Image-Pipeline-Konsole verwenden, der Sie durch die folgenden Schritte führt:

- Geben Sie Pipeline-Details wie Name, Beschreibung und Ressourcen-Tags an.
- Wählen Sie ein AMI-Image-Rezept aus, das ein Basis-Image aus verwalteten Schnellstart-Images oder Images enthält, die Sie erstellt haben oder die mit Ihnen geteilt wurden. Das Rezept enthält auch Komponenten, die die folgenden Aufgaben auf den EC2-Instances ausführen, die Image Builder zum Erstellen Ihres Images verwendet:

  - Software hinzufügen und entfernen
  - Passen Sie Einstellungen und Skripts an
  - Führen Sie ausgewählte Tests aus
- Geben Sie Workflows an, um die Image-Erstellung und die Testschritte zu konfigurieren, die in Ihrer Pipeline ausgeführt werden.
- Definieren Sie die Infrastrukturkonfiguration für Ihre Pipeline mit Standardeinstellungen oder Einstellungen, die Sie selbst konfigurieren. Die Konfiguration umfasst den Instanztyp und das key pair, das für Ihr Image verwendet werden soll, Sicherheits- und Netzwerkeinstellungen, Einstellungen für Protokollspeicherung und Fehlerbehebung sowie SNS-Benachrichtigungen.

  Dies ist ein optionaler Schritt. Image Builder verwendet Standardeinstellungen für Ihre Infrastrukturkonfiguration, wenn Sie die Konfiguration nicht selbst definieren.
- Definieren Sie Verteilungseinstellungen, um Ihre Bilder an AWS Zielregionen und Konten zu liefern. Sie können einen KMS-Schlüssel für die Verschlüsselung angeben, die AMI-Sharing- oder Lizenzkonfiguration konfigurieren oder eine Startvorlage für die von Ihnen verteilten AMIs konfigurieren.

  Dies ist ein optionaler Schritt. Wenn Sie die Konfiguration nicht selbst definieren, verwendet Image Builder die Standardbenennung für Ihr Ausgabe-AMI und verteilt das AMI an die Quellregion. Die Quellregion ist die Region, in der Sie die Pipeline ausführen.

Weitere Informationen und ein step-by-step Tutorial zur Verwendung des Konsolenassistenten zum Erstellen von Image-Pipelines mit Standardwerten, sofern vorhanden, finden Sie unterErstellen Sie eine Image-Pipeline mit dem EC2 Image Builder Builder-Konsolenassistenten.

Inhalt

Erstellen Sie eine AMI-Image-Pipeline (AWS CLI)

Sie können eine AMI-Image-Pipeline mit einer JSON-Datei erstellen, die Konfigurationsdetails als Eingabe für den create-image-pipeline Befehl in der enthält AWS CLI.

Wie oft Ihre Pipeline ein neues Image erstellt, um ausstehende Updates aus Ihrem Basis-Image und Ihren Komponenten zu integrieren, hängt davon ab`schedule`, welche Sie konfiguriert haben. A `schedule` hat die folgenden Attribute:

- `scheduleExpression`— Legt den Zeitplan fest, wann Ihre Pipeline ausgeführt wird, um das auszuwerten `pipelineExecutionStartCondition` und festzustellen, ob ein Build gestartet werden soll. Der Zeitplan ist mit Cron-Ausdrücken konfiguriert. Weitere Informationen zum Formatieren eines Cron-Ausdrucks in Image Builder finden Sie unterVerwenden Sie Cron-Ausdrücke in EC2 Image Builder.
- `pipelineExecutionStartCondition`— Legt fest, ob Ihre Pipeline den Build starten soll. Gültige Werte sind:
  - `EXPRESSION_MATCH_ONLY`— Ihre Pipeline erstellt jedes Mal ein neues Image, wenn der Cron-Ausdruck mit der aktuellen Uhrzeit übereinstimmt.
  - `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE`— Ihre Pipeline wird keinen neuen Image-Build starten, es sei denn, es stehen noch Änderungen an Ihrem Basis-Image oder Ihren Komponenten aus.

Wenn Sie den create-image-pipeline Befehl in der ausführen AWS CLI, sind viele der Konfigurationsressourcen optional. Für einige Ressourcen gelten jedoch bedingte Anforderungen, die davon abhängen, welchen Image-Typ die Pipeline erstellt. Die folgenden Ressourcen sind für AMI-Image-Pipelines erforderlich:

- Bildrezept ARN
- Konfiguration der Infrastruktur ARN

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den folgenden Schlüsseln sowie Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen`create-image-pipeline.json`:

   ```
   {
    "name": "MyWindows2019Pipeline",
    "description": "Builds Windows 2019 Images",
    "enhancedImageMetadataEnabled": true,
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-recipe/2020.12.03",
   ```

```
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/my-example-distribution-
configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 60
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * SUN *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "ENABLED"
}
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder create-image-pipeline --cli-input-json file://create-image-
pipeline.json
```

AMI-Image-Pipelines aktualisieren (Konsole)

Nachdem Sie eine Image Builder Builder-Image-Pipeline für Ihr AMI-Image erstellt haben, können Sie von der Image Builder Builder-Konsole aus Änderungen an der Infrastrukturkonfiguration und den Verteilungseinstellungen vornehmen.

Um eine Image-Pipeline mit einem neuen Image-Rezept zu aktualisieren, müssen Sie den verwenden AWS CLI. Weitere Informationen finden Sie unter AMI-Image-Pipelines aktualisieren ()AWS CLI in diesem Handbuch.

Wählen Sie eine vorhandene Image Builder Builder-Pipeline

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Um eine Liste der Image-Pipelines anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Image-Pipelines aus.

   > **Note**
   >
   > Die Liste der Image-Pipelines enthält einen Indikator für den Typ des Ausgabe-Images, das von der Pipeline erstellt wird — AMI oder Docker.

3. Um Details anzuzeigen oder eine Pipeline zu bearbeiten, wählen Sie den Link Pipeline-Name. Dadurch wird die Detailansicht für die Pipeline geöffnet.

   > **Note**
   >
   > Sie können auch das Kontrollkästchen neben dem Namen der Pipeline aktivieren und dann Detail anzeigen auswählen.

Details zur Pipeline

Die Seite mit den Pipeline-Details umfasst die folgenden Abschnitte:

Übersicht

Der Abschnitt oben auf der Seite fasst die wichtigsten Details der Pipeline zusammen, die sichtbar sind, wenn eine der Detailregisterkarten geöffnet ist. Die in diesem Abschnitt angezeigten Details können nur auf den jeweiligen Detailregisterkarten bearbeitet werden.

Detail-Registerkarten

- Ausgabebilder — Zeigt Ausgabebilder an, die von der Pipeline erzeugt wurden.
- Bildrezept — Zeigt Rezeptdetails an. Nachdem Sie ein Rezept erstellt haben, können Sie es nicht bearbeiten. Sie müssen eine neue Version des Rezepts auf der Seite Image-Rezepte in der Image

Builder Builder-Konsole oder mithilfe von Image Builder Builder-Befehlen in der erstellen AWS CLI. Weitere Informationen finden Sie unter Rezepte verwalten.

- Infrastrukturkonfiguration — Zeigt bearbeitbare Informationen für die Konfiguration Ihrer Build-Pipeline-Infrastruktur an.
- Verteilungseinstellungen — Zeigt bearbeitbare Informationen für die AMI-Verteilung an.
- EventBridge Regeln — Zeigt für den ausgewählten Event Bus EventBridge Regeln an, die auf die aktuelle Pipeline abzielen. Beinhaltet die Aktionen „Event-Bus erstellen“ und „Regel erstellen“, die mit der EventBridge Konsole verknüpft sind. Weitere Informationen zu dieser Registerkarte finden Sie unterVerwenden Sie Regeln EventBridge .

Bearbeiten Sie die Infrastrukturkonfiguration für Ihre Pipeline

Die Infrastrukturkonfiguration umfasst die folgenden Details, die Sie nach dem Erstellen der Pipeline bearbeiten können:

- Die Beschreibung für Ihre Infrastrukturkonfiguration.
- Die IAM-Rolle, die dem Instanzprofil zugeordnet werden soll.
- AWS Infrastruktur, einschließlich des Instanztyps und eines SNS-Themas für Benachrichtigungen.
- VPC, Subnetz und Sicherheitsgruppen.
- Einstellungen zur Fehlerbehebung, einschließlich „Instanz bei Ausfall beenden“, Schlüsselpaar für die Verbindung und optionaler S3-Bucket-Speicherort für Instanzprotokolle.

Gehen Sie wie folgt vor, um die Infrastrukturkonfiguration auf der Seite mit den Pipeline-Details zu bearbeiten:

1. Wählen Sie die Registerkarte Infrastrukturkonfiguration.
2. Wählen Sie in der oberen rechten Ecke des Bereichs mit den Konfigurationsdetails die Option Bearbeiten aus.
3. Wenn Sie bereit sind, die an Ihrer Infrastrukturkonfiguration vorgenommenen Aktualisierungen zu speichern, wählen Sie Änderungen speichern.

Bearbeiten Sie die Verteilungseinstellungen für Ihre Pipeline

Zu den Verteilungseinstellungen gehören die folgenden Details, die Sie nach dem Erstellen der Pipeline bearbeiten können:

- Die Beschreibung für Ihre Verteilungskonfiguration.
- Regionseinstellungen für die Regionen, in denen Sie Ihr Image vertreiben. Region 1 ist standardmäßig die Region, in der Sie die Pipeline erstellt haben. Mit der Schaltfläche „Region hinzufügen“ können Sie Regionen für den Vertrieb hinzufügen und alle Regionen außer Region 1 entfernen.

  Zu den Regionseinstellungen gehören:

  - Zielregion
  - Der Name des Ausgabe-AMIs
  - Startberechtigungen und Konten, mit denen sie geteilt werden können
  - Zugeordnete Lizenzen (Assoziierte Lizenzkonfigurationen)

  > **Note**
  >
  > License Manager Manager-Einstellungen werden nicht zwischen AWS Regionen repliziert, die in Ihrem Konto aktiviert sein müssen, z. B. zwischen den Regionen `ap-east-1` (Hongkong) und `me-south-1` (Bahrain).

Gehen Sie wie folgt vor, um Ihre Vertriebseinstellungen auf der Seite mit den Pipeline-Details zu bearbeiten:

1. Wählen Sie den Tab Verteilungseinstellungen.
2. Wählen Sie in der oberen rechten Ecke des Bereichs mit den Vertriebsdetails die Option Bearbeiten aus.
3. Wenn Sie bereit sind, Ihre Updates zu speichern, wählen Sie Änderungen speichern.

Bearbeiten Sie den Build-Zeitplan für Ihre Pipeline

Die Seite „Pipeline bearbeiten“ enthält die folgenden Details, die Sie nach dem Erstellen der Pipeline bearbeiten können:

- Die Beschreibung für Ihre Pipeline.
- Verbesserte Metadatensammlung. Diese Option ist standardmäßig aktiviert. Um sie zu deaktivieren, deaktivieren Sie das Kontrollkästchen Erweiterte Metadatensammlung aktivieren.

- Der Build-Zeitplan für Ihre Pipeline. Sie können Ihre Zeitplanoptionen und alle Einstellungen hier ändern.

Gehen Sie wie folgt vor, um Ihre Pipeline auf der Seite mit den Pipeline-Details zu bearbeiten:

1. Wählen Sie in der oberen rechten Ecke der Seite mit den Pipeline-Details Aktionen und dann Pipeline bearbeiten aus.
2. Wenn Sie bereit sind, Ihre Updates zu speichern, wählen Sie Änderungen speichern.

> **Note**
>
> Weitere Informationen zur Planung Ihres Builds mithilfe von Cron-Ausdrücken finden Sie unterVerwenden Sie Cron-Ausdrücke in EC2 Image Builder.

AMI-Image-Pipelines aktualisieren ()AWS CLI

Sie können eine AMI-Image-Pipeline aktualisieren, indem Sie eine JSON-Datei als Eingabe für den update-image-pipeline Befehl in der verwenden AWS CLI. Um die JSON-Datei zu konfigurieren, benötigen Sie Amazon Resource Names (ARNs), um auf die folgenden vorhandenen Ressourcen zu verweisen:

- Zu aktualisierende Image-Pipeline
- Bildrezept
- Konfiguration der Infrastruktur
- Distribution Settings (Einstellungen für die Verteilung)

Sie können eine AMI-Image-Pipeline mit dem AWS CLI folgenden update-image-pipeline Befehl aktualisieren:

> **Note**
>
> UpdateImagePipeline unterstützt keine selektiven Updates für die Pipeline. Sie müssen alle erforderlichen Eigenschaften in der Aktualisierungsanforderung angeben, nicht nur die Eigenschaften, die sich geändert haben.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den folgenden Schlüsseln sowie Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Name`create-component.json`:

   ```
   {
    "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-
   pipeline/my-example-pipeline",
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-
   example-recipe/2019.12.08",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
   west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
   configuration",
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
   west-2:123456789012:distribution-configuration/my-example-distribution-
   configuration",
    "imageTestsConfiguration": {
     "imageTestsEnabled": true,
     "timeoutMinutes": 120
    },
    "schedule": {
     "scheduleExpression": "cron(0 0 * * MON *)",
     "pipelineExecutionStartCondition":
    "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
    },
    "status": "DISABLED"
   }
   ```

   > **Note**
   >
   > - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
   > - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder update-image-pipeline --cli-input-json file://update-image-pipeline.json
```

Container-Image-Pipelines erstellen und aktualisieren

Sie können Container-Image-Pipelines mithilfe der Image Builder-Konsole, über die Image Builder Builder-API oder mit imagebuilder Befehlen in der AWS CLI einrichten, konfigurieren und verwalten. Der Assistent für die Konsole „Image-Pipeline erstellen“ stellt Startartefakte bereit und führt Sie durch die folgenden Schritte:

- Wählen Sie ein Basis-Image aus verwalteten Schnellstart-Images, Amazon ECR- oder Docker Hub-Repositorys
- Software hinzufügen und entfernen
- Passen Sie Einstellungen und Skripts an
- Führen Sie ausgewählte Tests aus
- Erstellen Sie ein Dockerfile mit vorkonfigurierten Build-Time-Variablen.
- Verteilen AWS Sie Bilder auf Regionen

Weitere Informationen und ein step-by-step Tutorial zur Verwendung des Konsolenassistenten zum Erstellen von Image-Pipelines finden Sie unterErstellen Sie eine Container-Image-Pipeline mit dem EC2 Image Builder Builder-Konsolenassistenten.

Inhalt

Erstellen Sie eine Container-Image-Pipeline (AWS CLI)

Sie können eine Container-Image-Pipeline erstellen, indem Sie eine JSON-Datei als Eingabe für den create-image-pipeline Befehl in der verwenden AWS CLI.

Wie oft Ihre Pipeline ein neues Image erstellt, um ausstehende Updates aus Ihrem Basis-Image und Ihren Komponenten zu integrieren, hängt davon ab`schedule`, welche Sie konfiguriert haben. A `schedule` hat die folgenden Attribute:

- `scheduleExpression`— Legt den Zeitplan fest, wann Ihre Pipeline ausgeführt wird, um das auszuwerten `pipelineExecutionStartCondition` und festzustellen, ob ein Build gestartet werden soll. Der Zeitplan ist mit Cron-Ausdrücken konfiguriert. Weitere Informationen zum Formatieren eines Cron-Ausdrucks in Image Builder finden Sie unterVerwenden Sie Cron-Ausdrücke in EC2 Image Builder.
- `pipelineExecutionStartCondition`— Legt fest, ob Ihre Pipeline den Build starten soll. Gültige Werte sind:
  - `EXPRESSION_MATCH_ONLY`— Ihre Pipeline erstellt jedes Mal ein neues Image, wenn der Cron-Ausdruck mit der aktuellen Uhrzeit übereinstimmt.
  - `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE`— Ihre Pipeline wird keinen neuen Image-Build starten, es sei denn, es stehen noch Änderungen an Ihrem Basis-Image oder Ihren Komponenten aus.

Wenn Sie den create-image-pipeline Befehl in der ausführen AWS CLI, sind viele der Konfigurationsressourcen optional. Für einige Ressourcen gelten jedoch bedingte Anforderungen, die davon abhängen, welchen Image-Typ die Pipeline erstellt. Die folgenden Ressourcen sind für Container-Image-Pipelines erforderlich:

- Container-Rezept ARN
- Konfiguration der Infrastruktur ARN

Wenn Sie bei der Ausführung des create-image-pipeline Befehls keine Ressource für die Verteilungskonfiguration angeben, wird das Ausgabebild im ECR-Repository gespeichert, das Sie in Ihrem Container-Rezept in der Region, in der Sie den Befehl ausführen, als Ziel-Repository angeben. Wenn Sie eine Ressource für die Verteilungskonfiguration für Ihre Pipeline einbeziehen, wird das Ziel-Repository verwendet, das Sie für die erste Region in der Verteilung angegeben haben.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den folgenden Schlüsseln sowie Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Namen`create-image-pipeline.json`:

```
{
 "name": "MyWindows2019Pipeline",
 "description": "Builds Windows 2019 Images",
 "enhancedImageMetadataEnabled": true,
 "containerRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:container-
recipe/my-example-recipe/2020.12.03",
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/my-example-distribution-
configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 60
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * SUN *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "ENABLED"
}
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder create-image-pipeline --cli-input-json file://create-image-
pipeline.json
```

Aktualisieren Sie eine Container-Image-Pipeline (Konsole)

Nachdem Sie eine Image Builder Builder-Container-Image-Pipeline für Ihr Docker-Image erstellt haben, können Sie in der Image Builder Builder-Konsole Änderungen an der Infrastrukturkonfiguration und den Verteilungseinstellungen vornehmen.

Um eine Container-Image-Pipeline mit einem neuen Container-Rezept zu aktualisieren, müssen Sie die AWS CLI verwenden. Weitere Informationen finden Sie unter Aktualisieren Sie die Pipelines für Container-Images ()AWS CLI in diesem Handbuch.

Wählen Sie eine vorhandene Image Builder Docker-Image-Pipeline

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Um eine Liste der Image-Pipelines anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Image-Pipelines aus.

   > **Note**
   >
   > Die Liste der Image-Pipelines enthält einen Indikator für den Typ des Ausgabe-Images, das von der Pipeline erstellt wird — AMI oder Docker.

3. Um Details anzuzeigen oder eine Pipeline zu bearbeiten, wählen Sie den Link Pipeline-Name. Dadurch wird die Detailansicht für die Pipeline geöffnet.

   > **Note**
   >
   > Sie können auch das Kontrollkästchen neben dem Namen der Pipeline aktivieren und dann Detail anzeigen auswählen.

Details zur Pipeline

Die Detailseite der EC2 Image Builder Builder-Pipeline umfasst die folgenden Abschnitte:

Übersicht

Der Abschnitt oben auf der Seite fasst die wichtigsten Details der Pipeline zusammen, die sichtbar sind, wenn eine der Detailregisterkarten geöffnet ist. Die in diesem Abschnitt angezeigten Details können nur auf den jeweiligen Detailregisterkarten bearbeitet werden.

Detail-Registerkarten

- Ausgabebilder — Zeigt Ausgabebilder an, die von der Pipeline erzeugt wurden.
- Container-Rezept — Zeigt Rezeptdetails an. Nachdem Sie ein Rezept erstellt haben, können Sie es nicht bearbeiten. Sie müssen auf der Seite Container-Rezepte eine neue Version des Rezepts erstellen. Weitere Informationen finden Sie unter Eine neue Version eines Container-Rezepts erstellen.
- Infrastrukturkonfiguration — Zeigt bearbeitbare Informationen für die Konfiguration Ihrer Build-Pipeline-Infrastruktur an.
- Verteilungseinstellungen — Zeigt bearbeitbare Informationen für die Docker-Image-Verteilung an.
- EventBridge Regeln — Zeigt für den ausgewählten Event Bus EventBridge Regeln an, die auf die aktuelle Pipeline abzielen. Beinhaltet die Aktionen „Event-Bus erstellen“ und „Regel erstellen“, die mit der EventBridge Konsole verknüpft sind. Weitere Informationen zu dieser Registerkarte finden Sie unterVerwenden Sie Regeln EventBridge .

Bearbeiten Sie die Infrastrukturkonfiguration für Ihre Pipeline

Die Infrastrukturkonfiguration umfasst die folgenden Details, die Sie nach dem Erstellen der Pipeline bearbeiten können:

- Die Beschreibung für Ihre Infrastrukturkonfiguration.
- Die IAM-Rolle, die dem Instanzprofil zugeordnet werden soll.
- AWS Infrastruktur, einschließlich des Instanztyps und eines SNS-Themas für Benachrichtigungen.
- VPC, Subnetz und Sicherheitsgruppen.
- Einstellungen zur Fehlerbehebung, einschließlich Instanz bei Ausfall beenden, Schlüsselpaar für die Verbindung und optionaler S3-Bucket-Speicherort für Instanzprotokolle.

Gehen Sie wie folgt vor, um die Infrastrukturkonfiguration auf der Seite mit den Pipeline-Details zu bearbeiten:

1. Wählen Sie die Registerkarte Infrastrukturkonfiguration.
2. Wählen Sie in der oberen rechten Ecke des Bereichs mit den Konfigurationsdetails die Option Bearbeiten aus.
3. Wenn Sie bereit sind, die an Ihrer Infrastrukturkonfiguration vorgenommenen Aktualisierungen zu speichern, wählen Sie Änderungen speichern.

Bearbeiten Sie die Verteilungseinstellungen für Ihre Pipeline

Zu den Verteilungseinstellungen gehören die folgenden Details, die Sie nach dem Erstellen der Pipeline bearbeiten können:

- Die Beschreibung für Ihre Verteilungseinstellungen.
- Regionseinstellungen für die Regionen, in denen Sie Ihr Bild vertreiben. Region 1 ist standardmäßig die Region, in der Sie die Pipeline erstellt haben. Mit der Schaltfläche „Region hinzufügen“ können Sie Regionen für den Vertrieb hinzufügen und alle Regionen außer Region 1 entfernen.

  Zu den Regionseinstellungen gehören:

  - Zielregion
  - Der Service ist standardmäßig auf „ECR“ eingestellt und kann nicht bearbeitet werden.
  - Repository-Name — der Name Ihres Ziel-Repositorys (ohne den Amazon ECR-Speicherort). Der Repository-Name mit dem Speicherort würde beispielsweise wie folgt aussehen:

    ```
    <account-id>.dkr.ecr.<region>.amazonaws.com/<repository-name>
    ```

    > **Note**
    >
    > Wenn Sie den Repository-Namen ändern, werden nur die Bilder, die nach der Namensänderung erstellt wurden, unter dem neuen Namen hinzugefügt. Alle vorherigen Bilder, die Ihre Pipeline erstellt hat, verbleiben in ihrem ursprünglichen Repository.

Gehen Sie wie folgt vor, um Ihre Verteilungseinstellungen auf der Seite mit den Pipeline-Details zu bearbeiten:

1. Wählen Sie den Tab Verteilungseinstellungen.
2. Wählen Sie in der oberen rechten Ecke des Bereichs mit den Vertriebsdetails die Option Bearbeiten aus.
3. Wenn Sie bereit sind, die Aktualisierungen, die Sie an Ihren Vertriebseinstellungen vorgenommen haben, zu speichern, wählen Sie Änderungen speichern.

Bearbeiten Sie den Build-Zeitplan für Ihre Pipeline

Die Seite „Pipeline bearbeiten“ enthält die folgenden Details, die Sie nach dem Erstellen der Pipeline bearbeiten können:

- Die Beschreibung für Ihre Pipeline.
- Verbesserte Metadatensammlung. Diese Option ist standardmäßig aktiviert. Um sie zu deaktivieren, deaktivieren Sie das Kontrollkästchen Erweiterte Metadatensammlung aktivieren.
- Der Build-Zeitplan für Ihre Pipeline. Sie können Ihre Zeitplanoptionen und alle Einstellungen in diesem Abschnitt ändern.

Gehen Sie wie folgt vor, um Ihre Pipeline auf der Seite mit den Pipeline-Details zu bearbeiten:

1. Wählen Sie in der oberen rechten Ecke der Seite mit den Pipeline-Details Aktionen und dann Pipeline bearbeiten aus.
2. Wenn Sie bereit sind, Ihre Updates zu speichern, wählen Sie Änderungen speichern.

> **Note**
>
> Weitere Informationen zur Planung Ihres Builds mithilfe von Cron-Ausdrücken finden Sie unterVerwenden Sie Cron-Ausdrücke in EC2 Image Builder.

Aktualisieren Sie die Pipelines für Container-Images ()AWS CLI

Sie können eine Container-Image-Pipeline aktualisieren, indem Sie eine JSON-Datei als Eingabe für den update-image-pipeline Befehl in der verwenden AWS CLI. Um die JSON-Datei zu konfigurieren, benötigen Sie Amazon Resource Names (ARNs), um auf die folgenden vorhandenen Ressourcen zu verweisen:

- Zu aktualisierende Image-Pipeline
- Rezept für Container
- Konfiguration der Infrastruktur
- Verteilungseinstellungen (sofern in der aktuellen Pipeline enthalten)

> **Note**
>
> Wenn die Ressource für die Verteilungseinstellungen enthalten ist, hat das ECR-Repository, das in den Verteilungseinstellungen für die Region, in der der Befehl ausgeführt wird (Region 1), als Ziel-Repository angegeben ist, Vorrang vor dem Ziel-Repository, das im Container-Rezept angegeben ist.

Gehen Sie wie folgt vor, um eine Container-Image-Pipeline mit dem update-image-pipeline folgenden Befehl zu aktualisieren: AWS CLI

> **Note**
>
> UpdateImagePipeline unterstützt keine selektiven Updates für die Pipeline. Sie müssen in der Aktualisierungsanforderung alle erforderlichen Eigenschaften angeben, nicht nur die Eigenschaften, die sich geändert haben.

1. Erstellen einer CLI-Eingabe-JSON-Datei

   Verwenden Sie Ihr bevorzugtes Dateibearbeitungstool, um eine JSON-Datei mit den folgenden Schlüsseln sowie Werten zu erstellen, die für Ihre Umgebung gültig sind. In diesem Beispiel wird eine Datei mit dem Name`create-component.json`:

```
{
 "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-
pipeline/my-example-pipeline",
 "containerRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:container-
recipe/my-example-recipe/2020.12.08",
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/my-example-distribution-
configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 120
 },
 "schedule": {
```

```
  "scheduleExpression": "cron(0 0 * * MON *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "DISABLED"
}
```

> **Note**
>
> - Sie müssen die`file://`-Notation am Anfang des JSON-Dateipfades.
> - Der Pfad für die JSON-Datei sollte der entsprechenden Konvention für das Basisbetriebssystem folgen, auf dem Sie den Befehl ausführen. Windows verwendet beispielsweise den umgekehrten Schrägstrich (\), um auf den Verzeichnispfad zu verweisen, und Linux verwendet den Schrägstrich (/).

2. Führen Sie den folgenden Befehl aus, indem Sie die Datei verwenden, die Sie als Eingabe erstellt haben.

```
aws imagebuilder update-image-pipeline --cli-input-json file://update-image-pipeline.json
```

Konfigurieren Sie Image-Workflows für Ihre EC2 Image Builder Builder-Pipeline

Mit Image-Workflows können Sie die Workflows, die Ihre Pipeline zum Erstellen und Testen von Images ausführt, an Ihre Bedürfnisse anpassen. Die Workflows, die Sie definieren, werden im Kontext des Image Builder Builder-Workflow-Frameworks ausgeführt. Weitere Informationen zu den Phasen, aus denen das Workflow-Framework besteht, finden Sie unterVerwaltung von Build- und Test-Workflows für EC2 Image Builder Builder-Images.

Workflow erstellen

Erstellen Sie Workflows, die während der `Build` Phase des Workflow-Frameworks ausgeführt werden. Sie können nur einen Build-Workflow für Ihre Pipeline angeben. Oder Sie können den Build komplett überspringen, um eine reine Test-Pipeline zu konfigurieren.

Test-Workflow

Test-Workflows werden während der `Test` Phase des Workflow-Frameworks ausgeführt. Sie können bis zu zehn Test-Workflows für Ihre Pipeline angeben. Sie können Tests auch komplett überspringen, wenn Sie nur möchten, dass Ihre Pipeline erstellt wird.

Definieren Sie Testgruppen für Test-Workflows

Test-Workflows werden innerhalb von Testgruppen definiert. Sie können bis zu zehn Test-Workflows für Ihre Pipeline ausführen. Sie entscheiden, ob Sie die Test-Workflows in einer bestimmten Reihenfolge oder so viele wie möglich gleichzeitig ausführen möchten. Wie sie ausgeführt werden, hängt davon ab, wie Sie Ihre Testgruppen definieren. Die folgenden Szenarien zeigen verschiedene Möglichkeiten, wie Sie Ihre Test-Workflows definieren können.

> **Note**
>
> Wenn Sie die Konsole zum Erstellen von Workflows verwenden, empfehlen wir, dass Sie sich Zeit nehmen, um zu planen, wie Sie Ihre Test-Workflows ausführen möchten, bevor Sie Ihre Testgruppen definieren. In der Konsole können Sie Testworkflows und -gruppen hinzufügen oder entfernen, aber Sie können sie nicht neu anordnen.

Szenario 1: Führen Sie jeweils einen Test-Workflow aus

Um alle Ihre Testworkflows nacheinander auszuführen, können Sie bis zu zehn Testgruppen konfigurieren, die jeweils einen einzigen Testworkflow enthalten. Testgruppen werden nacheinander in der Reihenfolge ausgeführt, in der Sie sie Ihrer Pipeline hinzufügen. Auf diese Weise können Sie sicherstellen, dass Ihre Testworkflows nacheinander in einer bestimmten Reihenfolge ausgeführt werden.

Szenario 2: Führen Sie mehrere Test-Workflows gleichzeitig aus

Wenn die Reihenfolge keine Rolle spielt und Sie so viele Test-Workflows wie möglich gleichzeitig ausführen möchten, können Sie eine einzelne Testgruppe konfigurieren und die maximale Anzahl von Test-Workflows darin platzieren. Image Builder startet bis zu fünf Test-Workflows gleichzeitig und startet weitere Test-Workflows, sobald andere abgeschlossen sind. Wenn Sie Ihre Test-Workflows so schnell wie möglich ausführen möchten, ist dies eine Möglichkeit, dies zu tun.

Szenario 3: Mischen und Kombinieren

Wenn Sie ein gemischtes Szenario haben, mit einigen Testworkflows, die gleichzeitig ausgeführt werden können, und anderen, die nacheinander ausgeführt werden sollten, können Sie Ihre Testgruppen so konfigurieren, dass dieses Ziel erreicht wird. Die einzige Beschränkung bei der Konfiguration Ihrer Testgruppen ist die maximale Anzahl von Testworkflows, die für Ihre Pipeline ausgeführt werden können

Workflow-Parameter in einer Image Builder Builder-Pipeline (Konsole) festlegen

Workflow-Parameter funktionieren bei Build-Workflows und Test-Workflows auf die gleiche Weise. Wenn Sie eine Pipeline erstellen oder aktualisieren, wählen Sie Build- und Test-Workflows aus, die Sie einbeziehen möchten. Wenn Sie im Workflow-Dokument Parameter für einen ausgewählten Workflow definiert haben, zeigt Image Builder sie im Bereich Parameter an. Bei Workflows, für die keine Parameter definiert sind, ist der Bereich ausgeblendet.

Jeder Parameter zeigt die folgenden Attribute an, die in Ihrem Workflow-Dokument definiert wurden:

- Name (nicht editierbar) — Der Name des Parameters.
- Typ (nicht editierbar) — Der Datentyp für den Parameterwert.
- Wert — Der Wert für den Parameter. Sie können den Parameterwert bearbeiten, um ihn für Ihre Pipeline festzulegen.

Geben Sie die IAM-Dienstrolle an, die Image Builder zum Ausführen von Workflow-Aktionen verwendet.

Zugriff auf Services

Um Image-Workflows auszuführen, benötigt Image Builder die Erlaubnis, Workflow-Aktionen auszuführen. Sie können die AWSServiceRoleForImageBuilderdienstbezogene Rolle angeben, oder Sie können Ihre eigene benutzerdefinierte Rolle für den Dienstzugriff wie folgt angeben.

- Konsole — Wählen Sie im Pipeline-Assistenten Schritt 3: Prozess zur Image-Erstellung definieren die serviceverknüpfte Rolle oder Ihre eigene benutzerdefinierte Rolle aus der IAM-Rollenliste im Bereich Dienstzugriff aus.
- Image Builder Builder-API — Geben Sie in der CreateImageAktionsanforderung die serviceverknüpfte Rolle oder Ihre eigene benutzerdefinierte Rolle als Wert für den `executionRole` Parameter an.

Weitere Informationen zum Erstellen einer Servicerolle finden Sie im AWS Identity and Access Management Benutzerhandbuch unter Erstellen einer Rolle zum Delegieren von Berechtigungen für einen AWS Dienst.

Führen Sie Ihre Image-Pipeline aus

Wenn Sie die manuelle Zeitplanoption für Ihre Pipeline ausgewählt haben, wird sie nur ausgeführt, wenn Sie den Build manuell starten. Wenn Sie sich für eine der automatischen Planungsoptionen entschieden haben, können Sie sie auch manuell zwischen regelmäßig geplanten Läufen ausführen. Wenn Sie beispielsweise eine Pipeline haben, die normalerweise einmal im Monat läuft, Sie aber zwei Wochen nach der vorherigen Ausführung ein Update für eine Ihrer Komponenten hinzufügen müssen, können Sie wählen, ob Ihre Pipeline manuell ausgeführt werden soll.

Console

Um Ihre Pipeline von der Seite mit den Pipeline-Details in der Image Builder Builder-Konsole aus auszuführen, wählen Sie im Menü Aktionen oben auf der Seite die Option Pipeline ausführen aus. Oben auf der Seite wird eine Statusmeldung angezeigt, die Sie darüber informiert, dass Ihre Pipeline gestartet wurde oder ob ein Fehler vorliegt.

1. Wählen Sie in der oberen linken Ecke der Seite mit den Pipeline-Details im Menü Aktionen die Option Pipeline ausführen aus.
2. Sie können den aktuellen Status Ihrer Pipeline auf der Registerkarte Ausgabebilder in der Spalte Status sehen.

AWS CLI

Das folgende Beispiel zeigt, wie Sie den start-image-pipeline-execution Befehl in verwenden AWS CLI , um eine Image-Pipeline manuell zu starten. Wenn Sie diesen Befehl ausführen, erstellt und verteilt die Pipeline ein neues Image.

```
aws imagebuilder start-image-pipeline-execution --image-pipeline-arn
 arn:aws:imagebuilder:us-west-2:111122223333:image-pipeline/my-example-pipeline
```

Informationen darüber, welche Ressourcen bei der Ausführung der Build-Pipeline erstellt werden, finden Sie unterRessourcen wurden erstellt.

Verwenden Sie Cron-Ausdrücke in EC2 Image Builder

Verwenden Sie Cron-Ausdrücke für EC2 Image Builder, um ein Zeitfenster einzurichten, in dem Ihr Image mit Updates aktualisiert wird, die für das Basis-Image und die Komponenten Ihrer Pipeline gelten. Das Zeitfenster für Ihre Pipeline-Aktualisierung beginnt mit der Zeit, die Sie im Cron-Ausdruck festgelegt haben. Sie können die Zeit in Ihrem Cron-Ausdruck auf die Minute genau festlegen. Ihr Pipeline-Build kann am oder nach der Startzeit ausgeführt werden.

Es kann manchmal einige Sekunden oder bis zu einer Minute dauern, bis Ihr Build ausgeführt wird.

> **Note**
>
> Cron-Ausdrücke verwenden standardmäßig die Zeitzone UTC (Universal Coordinated Time), oder Sie können die Zeitzone angeben. Weitere Informationen zur UTC-Zeit und zum Ermitteln des Offsets für Ihre Zeitzone finden Sie unter Abkürzungen für Zeitzonen — Weltweite Liste.

Unterstützte Werte für Cron-Ausdrücke in Image Builder

EC2 Image Builder verwendet ein Cron-Format, das aus sechs Pflichtfeldern besteht. Jedes Feld ist von den anderen durch ein Leerzeichen getrennt, ohne führende oder nachgestellte Leerzeichen:

```
<Minute> <Hour> <Day> <Month> <Day of the week> <Year>
```

Die folgende Tabelle zeigt die Werte, die für erforderliche Cron-Einträge unterstützt werden.

Unterstützte Werte für Cron-Ausdrücke

Feld	Werte	Platzhalter
Minute	`0-59`	`, - * /`
Stunde	`0-23`	`, - * /`
Tag	`1-31`	`, - * ? / L W`
Monat	`1-12` oder `jan-dec`	`, - * /`
Tag der Woche	`1-7` oder `sun-sat`	`, - * ? L #`

Feld	Werte	Platzhalter
Jahr	1970-2199	, - * /

Platzhalter

In der folgenden Tabelle wird beschrieben, wie Image Builder Platzhalter für Cron-Ausdrücke verwendet. Beachten Sie, dass es nach der von Ihnen angegebenen Zeit bis zu einer Minute dauern kann, bis der Build gestartet wird.

Unterstützte Platzhalter für Cron-Ausdrücke

Platzhalter	Beschreibung
,	Das Platzhalterzeichen , (Komma) schließt zusätzliche Werte ein. Im Feld Monat `jan,feb,mar` sind Januar, Februar und März enthalten.
-	Das Platzhalterzeichen - (Bindestrich) gibt einen Bereich an. Schließt im Feld Tag des Monats die Tage 1 bis 15 des angegebenen Monats ein. `1-15`
*	Der Platzhalter * (Sternchen) enthält alle gültigen Werte für das Feld.
?	Der Platzhalter ? (Fragezeichen) gibt an, dass der Feldwert von einer anderen Einstellung abhängt. Wenn bei den Feldern Day und D eines der ay-of-week Felder angegeben ist oder alle möglichen Werte (*) enthält, muss das andere Feld a ? sein. Sie können nicht beides angeben. Wenn Sie beispielsweise a 7 in das Feld Tag eingeben (führen Sie den Build am siebten Tag des Monats aus), muss die ay-of-week D-Position a enthalten?.

Platzhalter	Beschreibung
/	Das Platzhalterzeichen / (Schrägstrich) steht für schrittweise Steigerungen. Wenn Sie beispielsweise möchten, dass Ihr Build jeden zweiten Tag ausgeführt wird, geben Sie `*/2` in das Feld Tag ein.
L	Der Platzhalter L in einem der Tagesfelder gibt den letzten Tag an: 28-31 für den Tag des Monats, je nachdem, welcher Monat ist, oder Sonntag für den Wochentag.
W	Der Platzhalter W im ay-of-month Feld D gibt einen Wochentag an. Wenn Sie im ay-of-month Feld D eine Zahl vor dem eingeben, bedeutet das`W`, dass Sie den Wochentag als Ziel angeben möchten, der diesem Tag am nächsten liegt. Wenn Sie beispielsweise angeben`3W`, dass Ihr Build an dem Wochentag ausgeführt werden soll, der dem dritten Tag des Monats am nächsten liegt.
#	Das # (Hash) ist nur für das Feld Wochentag zulässig und muss von einer Zahl zwischen 1 und 5 gefolgt werden. Die Zahl gibt an, in welchen Wochen in einem bestimmten Monat der Build ausgeführt werden kann. Wenn Sie beispielsweise möchten, dass Ihr Build am zweiten Freitag eines jeden Monats ausgeführt wird, verwenden Sie das Feld `fri#2` für den Wochentag.

Einschränkungen

- Sie können die ay-of-week Felder D ay-of-month und D nicht in demselben Cron-Ausdruck angeben. Wenn Sie einen Wert oder * in einem dieser Felder angeben, müssen Sie ? in dem anderen Feld a verwenden.
- cron-Ausdrücke werden mit einer Ausführungsrate ab einer Minute unterstützt, kürzere Intervalle sind nicht möglich.

Beispiele für Cron-Ausdrücke in EC2 Image Builder

Cron-Ausdrücke werden für die Image Builder Builder-Konsole anders eingegeben als für die API oder CLI. Um Beispiele zu sehen, wählen Sie die für Sie zutreffende Registerkarte aus.

Image Builder console

Die folgenden Beispiele zeigen Cron-Ausdrücke, die Sie in die Konsole für Ihren Build-Zeitplan eingeben können. Die UTC-Zeit wird im 24-Stunden-Format angegeben.

Wird täglich um 10:00 Uhr (UTC) ausgeführt

```
0 10 * * ? *
```

Läuft täglich um 12:15 Uhr (UTC)

```
15 12 * * ? *
```

Läuft täglich um Mitternacht (UTC)

```
0 0 * * ? *
```

Läuft an jedem Werktagmorgen um 10:00 Uhr (UTC)

```
0 10 ? * 2-6 *
```

Laufen Sie an jedem Wochentag abends um 18 Uhr (UTC)

```
0 18 ? * mon-fri *
```

Läuft am ersten Tag jedes Monats um 8:00 Uhr (UTC)

```
0 8 1 * ? *
```

Läuft jeden zweiten Dienstag im Monat um 22:30 Uhr (UTC)

```
30 22 ? * tue#2 *
```

> **Tip**
>
> Wenn Sie nicht möchten, dass sich Ihr Pipeline-Job während der Ausführung auf den nächsten Tag erstreckt, stellen Sie sicher, dass Sie bei der Angabe der Startzeit die Zeit für Ihren Build berücksichtigen.

API/CLI

Die folgenden Beispiele zeigen Cron-Ausdrücke, die Sie mithilfe von CLI-Befehlen oder API-Anfragen für Ihren Build-Zeitplan eingeben können. Nur der Cron-Ausdruck wird angezeigt.

Wird täglich um 10:00 Uhr (UTC) ausgeführt

```
cron(0 10 * * ? *)
```

Läuft täglich um 12:15 Uhr (UTC)

```
cron(15 12 * * ? *)
```

Läuft täglich um Mitternacht (UTC)

```
cron(0 0 * * ? *)
```

Läuft an jedem Werktagmorgen um 10:00 Uhr (UTC)

```
cron(0 10 ? * 2-6 *)
```

Laufen Sie an jedem Wochentag abends um 18:00 Uhr (UTC)

```
cron(0 18 ? * mon-fri *)
```

Läuft am ersten Tag jedes Monats um 8:00 Uhr (UTC)

```
cron(0 8 1 * ? *)
```

Läuft jeden zweiten Dienstag im Monat um 22:30 Uhr (UTC)

```
cron(30 22 ? * tue#2 *)
```

> **Tip**
>
> Wenn Sie nicht möchten, dass sich Ihr Pipeline-Job während der Ausführung auf den nächsten Tag erstreckt, stellen Sie sicher, dass Sie bei der Angabe der Startzeit die Zeit für Ihren Build berücksichtigen.

Ausdrücke in EC2 Image Builder bewerten

Ein Rate-Ausdruck beginnt, wenn Sie eine Regel für ein geplantes Ereignis erstellen und mit dem definierten Zeitplan ausführen.

Rate-Ausdrücke bestehen aus zwei Pflichtfeldern. Die Felder werden durch ein Leerzeichen voneinander getrennt.

Syntax

```
rate(value unit)
```

Wert

Eine positive Zahl.

Einheit

Die Zeiteinheit. Für Werte von 1 werden verschiedene Einheiten benötigt, z. B. `minute`, ebenso für Werte über 1, z. B. `minutes`.

Zulässige Werte: Minute | Minuten | Stunde | Stunden | Tag | Tage

Einschränkungen

Wenn der Wert gleich 1 ist, muss die Einheit im Singular stehen. Wenn die Werte größer als 1 sind, muss die Einheit im Plural stehen. Beispielsweise sind `rate(1 hours)` und `rate(5 hour)` ungültige, `rate(1 hour)` und `rate(5 hours)` jedoch gültige Werte.

EventBridge Regeln mit Image Builder Builder-Pipelines verwenden

Veranstaltungen aus einer Vielzahl von Diensten AWS und Partnern werden nahezu in Echtzeit auf EventBridge Amazon-Eventbusse gestreamt. Sie können auch benutzerdefinierte Ereignisse

generieren und Ereignisse aus Ihren eigenen Anwendungen an EventBridge senden. Die Event-Busse verwenden Regeln, um zu bestimmen, wohin die Ereignisdaten weitergeleitet werden sollen.

Image Builder Builder-Pipelines sind als EventBridge Regelziele verfügbar. Das bedeutet, dass Sie eine Image Builder Builder-Pipeline auf der Grundlage von Regeln ausführen können, die Sie erstellen, um auf Ereignisse auf dem Bus oder nach einem Zeitplan zu reagieren.

Eine Zusammenfassung der vom System generierten Ereignisse, an die Image Builder sendet EventBridge, finden Sie unterVon Image Builder gesendete Ereignismeldungen.

> **Note**
>
> Event-Busse sind regionsspezifisch. Die Regel und das Ziel müssen sich in derselben Region befinden.

Inhalt

EventBridge Bedingungen

Dieser Abschnitt enthält eine Zusammenfassung der Begriffe, damit Sie besser verstehen, wie sie sich in Ihre Image Builder Builder-Pipelines EventBridge integrieren lassen.

Ereignis

Beschreibt eine Änderung in einer Umgebung, die sich auf eine oder mehrere Anwendungsressourcen auswirken kann. Die Umgebung kann eine AWS Umgebung, ein SaaS-Partnerdienst oder eine Anwendung oder eine Ihrer Anwendungen oder Dienste sein. Sie können auch geplante Ereignisse auf einer Zeitleiste einrichten.

Event Bus

Eine Pipeline, die Ereignisdaten von Anwendungen und Diensten empfängt.

Quelle

Der Dienst oder die Anwendung, die das Ereignis an den Ereignisbus gesendet hat.

Ziel

Eine Ressource oder ein Endpunkt, der EventBridge aufgerufen wird, wenn er einer Regel entspricht, und Daten vom Ereignis an das Ziel übermittelt.

Regel

Eine Regel ordnet eintreffende Ereignisse zu und leitet diese zur Verarbeitung an Ziele weiter. Eine einzelne Regel kann ein Ereignis an mehrere Ziele senden, die dann parallel ausgeführt werden können. Regeln basieren entweder auf einem Ereignismuster oder einem Zeitplan.

Muster

Ein Ereignismuster definiert die Ereignisstruktur und die Felder, denen eine Regel entspricht, um die Zielaktion auszulösen.

Plan

Mit Zeitplanregeln wird eine Aktion nach einem Zeitplan ausgeführt, z. B. das Ausführen einer Image Builder Builder-Pipeline zur vierteljährlichen Aktualisierung eines Images. Es gibt zwei Arten von Zeitplanausdrücken:

- Cron-Ausdrücke — Entsprechen bestimmten Planungskriterien mithilfe der Cron-Syntax, mit der einfache Kriterien beschrieben werden können, z. B. wöchentliche Ausführung an einem bestimmten Tag. Sie können auch komplexere Kriterien festlegen, z. B. vierteljährlich am fünften Tag des Monats, zwischen 2 Uhr und 4 Uhr morgens.
- Kursausdrücke — Geben Sie ein regelmäßiges Intervall an, in dem das Ziel aufgerufen wird, z. B. alle 12 Stunden.

EventBridge Regeln für Ihre Image Builder Builder-Pipeline anzeigen

Auf der Registerkarte EventBridge Regeln auf der Detailseite von Image Builder Image-Pipelines werden EventBridge Ereignisbusse angezeigt, auf die Ihr Konto Zugriff hat, sowie die Regeln für den ausgewählten Ereignisbus, die für die aktuelle Pipeline gelten. Diese Registerkarte ist auch direkt mit der EventBridge Konsole verknüpft, über die Sie neue Ressourcen erstellen können.

Aktionen, die mit der EventBridge Konsole verknüpft sind

- Event-Bus erstellen
- Regel erstellen

Weitere Informationen EventBridge finden Sie in den folgenden Themen im EventBridge Amazon-Benutzerhandbuch.

- Was ist Amazon EventBridge
- EventBridge Amazon-Eventbusse
- EventBridge Amazon-Veranstaltungen
- EventBridge Amazon-Regeln

Verwenden Sie EventBridge Regeln, um einen Pipeline-Build zu planen

In diesem Beispiel erstellen wir mithilfe eines Ratenausdrucks eine neue Zeitplanregel für den Standard-Event-Bus. Die Regel in diesem Beispiel generiert alle 90 Tage ein Ereignis im Eventbus. Das Ereignis initiiert einen Pipeline-Build, um das Image zu aktualisieren.

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Um eine Liste der Image-Pipelines anzuzeigen, die unter Ihrem Konto erstellt wurden, wählen Sie im Navigationsbereich Image-Pipelines aus.

   > **Note**
   >
   > Die Liste der Image-Pipelines enthält einen Indikator für den Typ des Ausgabe-Images, das von der Pipeline erstellt wird — AMI oder Docker.

3. Um Details anzuzeigen oder eine Pipeline zu bearbeiten, wählen Sie den Link Pipeline-Name. Dadurch wird die Detailansicht für die Pipeline geöffnet.

   > **Note**
   >
   > Sie können auch das Kontrollkästchen neben dem Namen der Pipeline aktivieren und dann Detail anzeigen auswählen.

4. Öffnen Sie die Registerkarte EventBridge Regeln.
5. Behalten Sie den Standard-Event-Bus bei, der im Event-Bus-Bereich vorausgewählt ist.
6. Wählen Sie Regel erstellen aus. Dadurch gelangen Sie zur Seite Regel erstellen in der EventBridge Amazon-Konsole.

7. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Der Regelname muss innerhalb des Event-Busses für die ausgewählte Region eindeutig sein.
8. Wählen Sie im Bereich „Muster definieren“ die Option „Zeitplan“. Dadurch wird der Bereich erweitert, sodass für jede Option die Option „Fester Tarif“ ausgewählt ist.
9. Geben Sie `90` in das erste Feld ein und wählen Sie in der Drop-down-Liste Tage aus.
10. Führen Sie im Bereich „Ziele auswählen“ die folgenden Aktionen aus:
    a. Wählen Sie `EC2 Image Builder` aus der Dropdownliste Ziel aus.
    b. Um die Regel auf eine Image Builder Builder-Pipeline anzuwenden, wählen Sie die Zielpipeline aus der Dropdownliste Image-Pipeline aus.
    c. EventBridge benötigt die Erlaubnis, einen Build für die ausgewählte Pipeline zu initiieren. Behalten Sie für dieses Beispiel die Standardoption Neue Rolle für diese spezifische Ressource erstellen bei.
    d. Wählen Sie Add target (Ziel hinzufügen) aus.
11. Wählen Sie Create (Erstellen) aus.

> **Note**
>
> Weitere Informationen zu Einstellungen für Preisausdrucksregeln, die in diesem Beispiel nicht behandelt werden, finden Sie unter Preisausdrücke im EventBridge Amazon-Benutzerhandbuch.

Integrieren Sie Produkte und Services in EC2 Image Builder

EC2 Image Builder lässt sich in andere AWS-Services Anwendungen integrieren, um Ihnen bei der Erstellung robuster, sicherer benutzerdefinierter Computer-Images zu helfen. AWS Marketplace

Produkte

In Image Builder-Rezepten können Image-Produkte aus AWS Marketplace und von Image Builder verwaltete Komponenten wie folgt integriert werden, um spezielle Build- und Testfunktionen bereitzustellen.

- AWS Marketplace Image-Produkte — Verwenden Sie ein Image-Produkt von AWS Marketplace als Basis-Image in Ihrem Rezept, um Unternehmensstandards wie CIS Hardening zu erfüllen. Wenn Sie in der Image Builder Builder-Konsole ein Rezept erstellen, können Sie aus Ihren vorhandenen Abonnements wählen oder nach einem bestimmten Produkt suchen AWS Marketplace. Wenn Sie ein Rezept über die Image Builder Builder-API, CLI oder das SDK erstellen, können Sie ein Image-Produkt mit dem Amazon Resource Name (ARN) angeben, das als Ihr Basis-Image verwendet werden soll.
- AWSTOE Komponenten — Komponenten, die Sie in Ihren Rezepten angeben, können Build- und Testaktionen ausführen, z. B. um Software zu installieren oder Konformitätsprüfungen durchzuführen. Einige Image-Produkte, die Sie abonnieren, enthalten AWS Marketplace möglicherweise eine Begleitkomponente, die Sie in Ihren Rezepten verwenden können. Die CIS-Hardened-Images enthalten eine passende AWSTOE Komponente, die Sie in Ihrem Rezept verwenden können, um die CIS-Benchmarks-Level-1-Richtlinien für Ihre Konfiguration durchzusetzen.

> **Note**
>
> Weitere Informationen zu Produkten im Zusammenhang mit der Einhaltung gesetzlicher Vorschriften finden Sie unter. Compliance-Produkte für Ihre Image Builder Builder-Images

Services

Image Builder lässt sich mit den folgenden Komponenten integrieren AWS-Services , um detaillierte Ereignismetriken, Protokollierung und Überwachung bereitzustellen. Diese Informationen helfen

Ihnen dabei, Ihre Aktivitäten zu verfolgen, Probleme bei der Image-Erstellung zu beheben und Automatisierungen auf der Grundlage von Ereignisbenachrichtigungen zu erstellen.

- AWS CloudTrail— Überwachen Sie Image Builder Builder-Ereignisse, an die gesendet werden CloudTrail. Weitere Informationen zu CloudTrail finden Sie unter Was ist AWS CloudTrail? im AWS CloudTrail Benutzerhandbuch.
- Amazon CloudWatch Logs — Überwachen Sie Ihre Image Builder Builder-Protokolldateien, speichern Sie sie und greifen Sie darauf zu. Optional können Sie Ihre Protokolle in einem S3-Bucket speichern. Weitere Informationen zu CloudWatch Logs finden Sie unter Was ist Amazon CloudWatch Logs? im Amazon CloudWatch Logs-Benutzerhandbuch.
- Amazon EventBridge — Stellen Sie eine Verbindung zu einem Stream von Echtzeit-Ereignisdaten aus Image Builder Builder-Aktivitäten in Ihrem Konto her. Weitere Informationen zu EventBridge finden Sie unter Was ist Amazon EventBridge? im EventBridge Amazon-Benutzerhandbuch.
- Amazon Inspector — Entdecken Sie Sicherheitslücken in Ihrer Software und Ihren Netzwerkeinstellungen mit automatischen Scans für die EC2-Testinstanz, die Image Builder startet, um ein neues Image zu erstellen. Image Builder speichert Ergebnisse für Ihre Ausgabe-Image-Ressource, sodass Sie sie nach dem Beenden Ihrer Testinstanz untersuchen und korrigieren können. Weitere Informationen zu Scans und Preisen finden Sie unter Was ist Amazon Inspector? im Amazon Inspector Inspector-Benutzerhandbuch.

  Amazon Inspector kann auch Ihre ECR-Repositorys scannen, wenn Sie das erweiterte Scannen konfigurieren. Weitere Informationen finden Sie unter Scannen von Amazon ECR-Container-Bildern im Amazon Inspector Inspector-Benutzerhandbuch.

  > **Note**
  >
  > Amazon Inspector ist eine kostenpflichtige Funktion.

- AWS Marketplace— Sehen Sie sich eine Liste Ihrer aktuellen AWS Marketplace Produktabonnements an und suchen Sie direkt in Image Builder nach Image-Produkten. Sie können auch ein Imageprodukt, das Sie abonniert haben, als Basisimage für ein Image Builder Builder-Rezept verwenden. Weitere Informationen zur Verwaltung von AWS Marketplace Abonnements finden Sie im AWS Marketplace Buyer Guide.
- Amazon Simple Notification Service (Amazon SNS) — Falls konfiguriert, veröffentlichen Sie detaillierte Nachrichten zu Ihrem Bildstatus in einem SNS-Thema, das Sie abonnieren. Weitere Informationen finden Sie unter Was ist Amazon SNS? im Entwicklerhandbuch für Amazon Simple Notification Service.

Themen zur Produkt- und Serviceintegration

AWS CloudTrail Integration in Image Builder

Dieser Dienst unterstützt AWS CloudTrail. CloudTrail ist ein Service, der AWS Anrufe für Sie aufzeichnet AWS-Konto und Protokolldateien an einen Amazon S3 S3-Bucket übermittelt. Anhand der von gesammelten Informationen können Sie feststellen CloudTrail, an welche Anfragen erfolgreich gestellt wurden AWS-Services, wer die Anfrage gestellt hat, wann sie gestellt wurde usw. Weitere Informationen zur CloudTrail Integration mit Image Builder finden Sie unterProtokollieren EC2 Image Builder Builder-API-Aufrufen mit AWS CloudTrail.

Weitere Informationen darüber CloudTrail, wie Sie die Funktion aktivieren und Ihre Protokolldateien finden, finden Sie im AWS CloudTrail Benutzerhandbuch.

Integration von Amazon CloudWatch Logs in Image Builder

CloudWatch Die Unterstützung von Protokollen ist standardmäßig aktiviert. Protokolle werden während des Build-Prozesses auf der Instance gespeichert und in CloudWatch Logs gestreamt. Die Instanzprotokolle werden vor der Image-Erstellung aus der Instanz entfernt.

Build-Logs werden in die folgende Image CloudWatch Builder-Logs-Gruppe und den folgenden Stream gestreamt:

LogGroup:

```
/aws/imagebuilder/ImageName
```

LogStream (x.x.x/x):

```
ImageVersion/ImageBuildVersion
```

Sie können das CloudWatch Log-Streaming deaktivieren, indem Sie die folgenden mit dem Instanzprofil verknüpften Berechtigungen entfernen.

```
"Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:CreateLogGroup",
            "logs:PutLogEvents"
        ],
        "Resource": "arn:aws:logs:*:*:log-group:/aws/imagebuilder/*"
    }
]
```

Für eine erweiterte Problembehandlung können Sie mit Run Command vordefinierte Befehle und Skripts AWS Systems Manager ausführen. Weitere Informationen finden Sie unter Problembehandlung bei EC2 Image Builder.

EventBridge Amazon-Integration in Image Builder

Amazon EventBridge ist ein serverloser Eventbus-Service, mit dem Sie Ihre Image Builder Builder-Anwendung mit verwandten Daten von anderen AWS-Services verbinden können. EventBridgeln ordnet eine Regel eingehende Ereignisse zu und sendet sie zur Verarbeitung an Ziele. Eine einzelne Regel kann ein Ereignis an mehrere Ziele senden, und diese Ereignisse laufen dann parallel ab.

Damit können Sie Systemereignisse wie Probleme mit EventBridge der Anwendungsverfügbarkeit oder Ressourcenänderungen automatisieren AWS-Services und automatisch darauf reagieren. Ereignisse von AWS-Services werden nahezu EventBridge in Echtzeit zugestellt. Sie können Regeln einrichten, die auf eingehende Ereignisse reagieren, um Aktionen einzuleiten. Beispielsweise das Senden eines Ereignisses an eine Lambda-Funktion, wenn sich der Status einer EC2-Instance von „Ausstehend“ in „Wird ausgeführt“ ändert. Diese werden Muster genannt. Informationen zum Erstellen einer Regel, die auf einem Ereignismuster basiert, finden Sie im EventBridge Amazon-Benutzerhandbuch unter EventBridge Amazon-Regeln erstellen, die auf Ereignisse reagieren.

Zu den Aktionen, die automatisch initiiert werden können, gehören:

- Rufen Sie eine Funktion auf AWS Lambda
- Aufrufen eines Amazon-EC2-Ausführungsbefehls
- Weitergabe des Ereignisses an Amazon Kinesis Data Streams
- Aktiviere eine AWS Step Functions Zustandsmaschine
- Ein Amazon SNS SNS-Thema oder eine Amazon SQS SQS-Warteschlange benachrichtigen

Sie können auch Planungsregeln für den Standardereignisbus einrichten, um in regelmäßigen Abständen eine Aktion auszuführen, z. B. das Ausführen einer Image Builder Builder-Pipeline zur vierteljährlichen Aktualisierung eines Images. Es gibt zwei Arten von Zeitplanausdrücken:

- Cron-Ausdrücke — Das folgende Beispiel für einen Cron-Ausdruck plant, dass eine Aufgabe jeden Tag um 12:00 Uhr UTC+0 ausgeführt wird:

  ```
  cron(0 12 * * ? *)
  ```

  Weitere Informationen zur Verwendung von Cron-Ausdrücken mit EventBridge finden Sie unter Cron-Ausdrücke im EventBridge Amazon-Benutzerhandbuch.
- Preisausdrücke — Das folgende Beispiel für einen Preisausdruck plant, dass eine Aufgabe alle 12 Stunden ausgeführt wird:

  ```
  rate(12 hour)
  ```

  Weitere Informationen zur Verwendung von Preisausdrücken mit EventBridge finden Sie unter Preisausdrücke im EventBridge Amazon-Benutzerhandbuch.

Weitere Informationen zur Integration von EventBridge Regeln in Image Builder Builder-Image-Pipelines finden Sie unter EventBridge Regeln mit Image Builder Builder-Pipelines verwenden.

Von Image Builder gesendete Ereignismeldungen

Image Builder sendet Ereignismeldungen, EventBridge wenn sich der Status der Image Builder Builder-Ressourcen erheblich ändert. Zum Beispiel, wenn sich der Status eines Images ändert. Die folgenden Beispiele zeigen typische JSON-Ereignisnachrichten, die Image Builder möglicherweise sendet.

EC2 Image Builder Image-Statusänderung

Image Builder sendet dieses Ereignis, wenn sich der Status einer Bildressource während der Image-Erstellung ändert. Wenn sich der Status des Images beispielsweise wie folgt von einem Status in einen anderen ändert:

- Von `building` bis `testing`
- Von `testing` bis `distribution`
- Von `testing` bis `failed`
- Von `integrating` bis `available`

```
{
    "version": "0",
    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "detail-type": "EC2 Image Builder Image State Change",
    "source": "aws.imagebuilder",
    "account": "111122223333",
    "time": "2024-01-18T17:50:56Z",
    "region": "us-west-2",
    "resources": ["arn:aws:imagebuilder:us-west-2:111122223333:image/
cmkencryptedworkflowtest-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222/1.0.0/1"],
    "detail": {
        "previous-state": {
            "status": "TESTING"
        },
        "state": {
            "status": "AVAILABLE"
        }
    }
}
```

EC2 Image Builder CVE erkannt

Wenn Sie die CVE-Erkennung für Ihr Bild aktiviert haben, sendet Image Builder nach Abschluss eines Bildscans eine Nachricht mit den Ergebnissen.

```
{
    "version": "0",
    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "detail-type": "EC2 Image Builder CVE Detected",
    "source": "aws.imagebuilder",
```

```
    "account": "111122223333",
    "time": "2023-03-01T16:59:09Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:imagebuilder:us-east-1:111122223333:image/test-image/1.0.0/1",
        "arn:aws:imagebuilder:us-east-1:111122223333:image-pipeline/test-pipeline"
    ],
    "detail": {
        "resource-id": "i-1234567890abcdef0",
        "finding-severity-counts": {
            "all": 0,
            "critical": 0,
            "high": 0,
            "medium": 0
        }
    }
}
```

EC2 Image Builder Builder-Workflow-Schritt „Warten“

Image Builder sendet eine Nachricht, wenn ein `WaitForAction` Workflow-Schritt angehalten wird, um auf den Abschluss einer asynchronen Aktion zu warten.

```
{
    "version": "0",
    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "detail-type": "EC2 Image Builder Workflow Step Waiting",
    "source": "aws.imagebuilder",
    "account": "111122223333",
    "time": "2024-01-18T16:54:44Z",
    "region": "us-west-2",
    "resources": ["arn:aws:imagebuilder:us-west-2:111122223333:image/
workflowstepwaitforactionwithvalidsnstopictest-a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222/1.0.0/1", "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/
build-workflow-a1b2c3d4-5678-90ab-cdef-EXAMPLE33333/1.0.0/1"],
    "detail": {
        "workflow-execution-id": "wf-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "workflow-step-execution-id": "step-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "workflow-step-name": "TestAutoSNSStop"
    }
}
```

Integration von Amazon Inspector in Image Builder

Wenn Sie Sicherheitsscans mit Amazon Inspector aktivieren, scannt Amazon Inspector kontinuierlich Computerbilder und laufende Instances in Ihrem Konto auf Sicherheitslücken in Betriebssystem und Programmiersprachen. Wenn aktiviert, erfolgt der Sicherheitsscan automatisch, und Image Builder kann eine Momentaufnahme der Ergebnisse Ihrer Testinstanz speichern, wenn Sie ein neues Image erstellen. Amazon Inspector ist ein kostenpflichtiger Service.

Wenn Amazon Inspector Sicherheitslücken in Ihrer Software oder Ihren Netzwerkeinstellungen entdeckt, ergreift Amazon Inspector die folgenden Maßnahmen:

- Informiert Sie darüber, dass ein Befund gefunden wurde.
- Bewertet den Schweregrad des Befundes. Bei der Bewertung des Schweregrads werden Sicherheitslücken kategorisiert, um Ihnen bei der Priorisierung Ihrer Ergebnisse zu helfen. Sie umfasst die folgenden Werte:
  - Ungeprüft
  - Informativ
  - Niedrig
  - Medium
  - Hoch
  - Kritisch
- Enthält Informationen zu den Ergebnissen und Links zu weiteren Ressourcen für weitere Informationen.
- Bietet Anleitungen zur Behebung der Probleme, die zu dem Ergebnis geführt haben.

Konfigurieren Sie Sicherheitsscans

Wenn Sie Amazon Inspector für Ihr Konto aktiviert haben, scannt Amazon Inspector automatisch die EC2-Instances, die Image Builder startet, um ein neues Image zu erstellen und zu testen. Diese Instances haben während des Build- und Testprozesses eine kurze Lebensdauer, und ihre Ergebnisse laufen normalerweise ab, sobald diese Instances heruntergefahren werden. Um Sie bei der Untersuchung und Behebung der Ergebnisse für Ihr neues Image zu unterstützen, kann Image Builder optional alle Ergebnisse, die Amazon Inspector während des Erstellungsprozesses auf Ihrer Test-Instance identifiziert hat, als Snapshot speichern.

Informationen zur Konfiguration von Sicherheitsscans für Ihre Pipeline finden Sie unterKonfigurieren Sie Sicherheitsscans für Image Builder Builder-Images in der AWS Management Console.

Überprüfen Sie die Sicherheitsergebnisse

In der Image Builder Builder-Konsole können Sie die Sicherheitsergebnisse für alle Ihre Image Builder Builder-Ressourcen an einem Ort einsehen. Sie können alle Ergebnisse auf der Seite mit den Sicherheitsergebnissen im Abschnitt Sicherheitsübersicht einsehen, oder Sie können Ihre Ergebnisse nach Sicherheitslücke, Image-Pipeline oder Image gruppieren. In der Konsole werden standardmäßig alle Sicherheitsergebnisse angezeigt. Im Übersichtsfenster für die Option Alle Sicherheitsergebnisse wird die Anzahl der Ergebnisse angezeigt, die Ihnen für jeden Schweregrad vorliegen. Weitere Informationen finden Sie unter Verwalten Sie die Sicherheitsergebnisse für Image Builder Builder-Images in der AWS Management Console.

Weitere Informationen zu den Ergebnissen von Sicherheitslücken in Amazon Inspector finden Sie unter Grundlegendes zu den Ergebnissen in Amazon Inspector im Amazon Inspector Inspector-Benutzerhandbuch.

AWS Marketplace Integration in Image Builder

AWS Marketplace ist ein kuratierter digitaler Katalog, in dem Sie Software, Daten und Dienste von Drittanbietern finden und abonnieren können, mit denen Sie Lösungen entwickeln können, die Ihren Geschäftsanforderungen entsprechen. AWS Marketplace bietet authentifizierten Käufern und registrierten Verkäufern Softwareangebote aus beliebten Kategorien wie Sicherheit, Netzwerk, Speicher, maschinelles Lernen und mehr.

Ein AWS Marketplace Verkäufer kann ein unabhängiger Softwareanbieter (ISV), ein Wiederverkäufer oder eine Einzelperson sein, die etwas zu bieten hat, das zu AWS Produkten und Dienstleistungen passt. Wenn der Verkäufer ein Produkt einreicht AWS Marketplace, legt er den Preis des Produkts sowie die Nutzungsbedingungen fest. Die Käufer erklären sich mit den für das Angebot festgelegten Preisen und Bedingungen einverstanden. Weitere Informationen AWS Marketplace dazu finden Sie unter Was ist AWS Marketplace?

> **Note**
>
> Anbieter von Datenprodukten müssen die Zulassungsvoraussetzungen für AWS Data Exchange erfüllen. Weitere Informationen finden Sie unter Bereitstellung von Datenprodukten für den AWS Data Exchange im AWS Data Exchange Exchange-Benutzerhandbuch.

AWS Marketplace Integrationsfunktionen

Image Builder lässt sich integrieren AWS Marketplace , um die folgenden Funktionen direkt von der Image Builder Builder-Konsole aus bereitzustellen:

- Suchen Sie nach Image-Produkten, die in verfügbar sind AWS Marketplace.
- Sehen Sie sich eine Liste Ihrer aktuellen AWS Marketplace Produktabonnements an.
- Verwenden Sie ein AWS Marketplace Image-Produkt als Basisimage für ein Image Builder Builder-Rezept.

Bei Produkten, die zugehörige AWS Task Orchestrator and Executor (AWSTOE) -Komponenten enthalten, können Sie in der Konsole sowie in der API, dem SDK und der CLI nach dem Product Owner filtern. Weitere Informationen finden Sie unter AWSTOE Komponenten auflisten.

Suchen Sie in der AWS Marketplace Image Builder Builder-Konsole nach Image-Produkten

Image Builder lässt sich integrieren AWS Marketplace , um Ihre Image-Produktabonnements direkt aus dem AWS MarketplaceBereich in der Image Builder Builder-Konsole anzuzeigen. Sie können auch auf der Seite AWS Marketplace Image-Produkte nach Image-Produkten suchen, ohne die Image Builder Builder-Konsole zu verlassen.

Gehen Sie folgendermaßen vor, um in der Image Builder Builder-Konsole nach einem AWS Marketplace Image-Produkt zu suchen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich im AWS MarketplaceAbschnitt Image-Produkte aus.
3. Auf der Seite Image-Produkte finden Sie eine Zusammenfassung der Image-Produkte, die Sie auf der Registerkarte Abonnements abonniert haben, oder Sie können auf der AWS MarketplaceRegisterkarte nach Image-Produkten suchen.

   Image Builder filtert Produkte vor und konzentriert sich AWS Marketplace auf Maschinenbilder, die Sie in Ihren Image Builder Builder-Rezepten verwenden können. Weitere Informationen zur AWS Marketplace Integration mit Image Builder erhalten Sie, wenn Sie die Registerkarte auswählen, die dem entspricht, was Sie sehen möchten.

AWS Marketplace

Diese Registerkarte enthält zwei Bereiche. Auf der linken Seite können Sie im Bereich Ergebnisse verfeinern Ihre Ergebnisse filtern, um die Produkte zu finden, die Sie abonnieren möchten. Auf der rechten Seite werden im Bereich Produkte suchen die Produkte angezeigt, die Ihren Filterkriterien entsprechen. Außerdem haben Sie die Möglichkeit, nach Produktnamen zu suchen.

Ergebnisse verfeinern

Die folgende Liste zeigt nur einige der Filter, die Sie auf Ihre Produktsuche anwenden können:

- Wählen Sie eine oder mehrere Produktkategorien aus, z. B. Infrastruktursoftware oder maschinelles Lernen.
- Wählen Sie die Betriebssysteme für Ihr Image-Produkt oder wählen Sie alle Produkte für eine bestimmte Betriebssystemplattform, zum Beispiel All Linux/Unix.
- Wählen Sie einen oder mehrere Herausgeber aus, deren verfügbare Produkte angezeigt werden sollen. Wählen Sie den Link Alle anzeigen aus, um alle Herausgeber anzuzeigen, deren Produkte den von Ihnen angewendeten Filtern entsprechen.

> **Note**
>
> Die Namen der Herausgeber sind nicht alphabetisch sortiert. Wenn Sie beispielsweise nach einem bestimmten Herausgeber suchen`Center for Internet Security`, können Sie einen Teil des Namens in das Suchfeld oben im Dialogfeld Alle Herausgeber eingeben. Sie sollten den Namen als Abkürzung buchstabieren, da dies `CIS` möglicherweise nicht zu den Ergebnissen führt, nach denen Sie suchen.
> Sie können die Namen der Herausgeber auch Seite für Seite durchsuchen.

Die Filterauswahl ist dynamisch. Jede Auswahl, die Sie treffen, wirkt sich auf Ihre Optionen für alle anderen Kategorien aus. Es sind Tausende von Produkten verfügbar. Je mehr Sie filtern können AWS Marketplace, desto wahrscheinlicher ist es, dass Sie das finden, was Sie suchen.

Produkte suchen

Um ein bestimmtes Produkt anhand des Namens zu finden, können Sie einen Teil des Namens in die Suchleiste oben in diesem Bereich eingeben. Jedes Produktergebnis enthält die folgenden Details:

- Der Produktname und das Logo. Beide sind mit der Produktdetailseite in verknüpft AWS Marketplace. Die Detailseite wird in einem neuen Tab in Ihrem Browser geöffnet. Von dort aus können Sie das Image-Produkt abonnieren, wenn Sie es in einem Image Builder Builder-Rezept verwenden möchten. Weitere Informationen finden Sie im AWS Marketplace Buyer Guide unter Produkte kaufen.

  Wenn Sie das Bildprodukt in abonnieren AWS Marketplace, wechseln Sie zurück zur Registerkarte Image Builder in Ihrem Browser und aktualisieren Sie Ihre Liste der abonnierten Bildprodukte, um es zu sehen.

  > **Note**
  >
  > Es kann einige Minuten dauern, bis Ihr neues Abonnement verfügbar ist.

- Der Name des Herausgebers. Dies ist mit der Detailseite des Herausgebers in verknüpft AWS Marketplace. Die Detailseite des Herausgebers wird in einem neuen Tab in Ihrem Browser geöffnet.
- Die Produktversion.
- Die Sternebewertung des Produkts und direkte Links zum Bewertungsbereich auf der Produktdetailseite finden Sie unter AWS Marketplace. Die Detailseite wird in einem neuen Tab in Ihrem Browser geöffnet.
- Die ersten Zeilen der Produktbeschreibung.

Direkt unter der Suchleiste können Sie sehen, wie viele Ergebnisse Ihre Suche erzielt hat und welche Teilmenge dieser Ergebnisse derzeit angezeigt wird. Mithilfe zusätzlicher Steuerelemente auf der rechten Seite des Bedienfelds können Sie Ihre Einstellungen für die Anzahl der Produkte, die gleichzeitig angezeigt werden, und die Sortierreihenfolge, die auf Ihre Ergebnisse angewendet werden soll, anpassen. Sie können auch die Seitennummerierung verwenden, um Ihre Ergebnisse durchzublättern.

Subscriptions

Auf dieser Registerkarte wird Ihnen eine Liste der Bildprodukte angezeigt, die Sie abonniert haben. AWS Marketplace Jedes abonnierte Produkt zeigt die folgenden Details:

- Der Produktname. Dies ist mit der Produktdetailseite in AWS Marketplace verknüpft. Die Produktdetailseite für Ihr abonniertes Produkt wird in einem neuen Tab in Ihrem Browser geöffnet.
- Der Name des Herausgebers. Dies ist mit der Detailseite des Herausgebers in verknüpft AWS Marketplace. Die Detailseite des Herausgebers wird in einem neuen Tab in Ihrem Browser geöffnet.
- Die Produktversion, die Sie abonniert haben.
- Wenn in Ihrem abonnierten Produkt eine zugehörige Komponente enthalten ist, zeigt Image Builder einen Link zu den AWSTOE Komponentendetails an.

Oben auf der Seite können Sie anhand des Namens nach einem bestimmten Produkt suchen, oder Sie können Ihre Ergebnisse mit den Steuerelementen für die Seitennummerierung durchblättern. Um ein abonniertes Produkt als Basisbild für ein neues Rezept zu verwenden, wählen Sie ein abonniertes Produkt aus und klicken Sie auf Neues Rezept erstellen. Image Builder wählt standardmäßig das erste Produkt in Ihrer Liste aus.

> **Note**
>
> Wenn Sie nach einem Produkt suchen, das Sie gerade abonniert haben, und es nicht in der Liste sehen, verwenden Sie die Schaltfläche „Aktualisieren“ oben auf der Registerkarte, um Ihre Ergebnisse zu aktualisieren. Es kann einige Minuten dauern, bis ein neues Abonnement in der Liste erscheint.

Verwenden Sie ein AWS Marketplace Image-Produkt in Image Builder Builder-Rezepten

In der Image Builder Builder-Konsole gibt es zwei Möglichkeiten, ein neues Image-Rezept auf der Grundlage eines Ihrer abonnierten Image-Produkte zu erstellen.

1. Sie können auf der Seite mit den Image-Produkten wie folgt beginnen:

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie im Navigationsbereich im AWS MarketplaceAbschnitt Image-Produkte aus.
3. Öffnen Sie den Tab Abonnements.
4. Wählen Sie das abonnierte Image-Produkt aus, das Sie als Basis-Image in Ihrem Rezept verwenden möchten.
5. Wählen Sie Neues Rezept erstellen. Dadurch wird die Seite „Rezept erstellen“ mit der Option „AWS Marketplace Bilder“ geöffnet und Ihr abonniertes Bildprodukt ist vorausgewählt.
6. Konfigurieren Sie die übrigen Einstellungen für Ihr Rezept wie gewohnt. Weitere Informationen zu Bildrezepten finden Sie unterErstellen Sie eine neue Version eines Bildrezepts.

2. Sie können auch die Seite „Rezept erstellen“ öffnen und ein AWS Marketplace Imageprodukt auswählen, das Sie als Basisbild verwenden möchten.

   1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
   2. Wählen Sie im Navigationsbereich im AWS MarketplaceAbschnitt Bildrezepte aus. Daraufhin wird Ihnen eine Liste der Bildrezepte angezeigt, die Sie erstellt haben.
   3. Wählen Sie Create image recipe (Image-Rezept erstellen) aus. Dadurch wird die Seite „Rezept erstellen“ geöffnet.
   4. Geben Sie den Namen und die Version Ihres Rezepts wie gewohnt in den Rezeptdetails ein.
   5. Wählen Sie im Bereich Basisbild die Option AWS Marketplace Bilder aus. Daraufhin wird Ihnen auf der Registerkarte Abonnements eine Liste der AWS Marketplace Image-Produkte angezeigt, die Sie abonniert haben. Sie können Ihr Basis-Image aus der Liste auswählen.

      Sie können auch AWS Marketplace direkt auf der AWS MarketplaceRegisterkarte nach anderen Bildprodukten suchen, die in verfügbar sind. Wählen Sie Produkte hinzufügen oder öffnen Sie den AWS MarketplaceTab direkt. Weitere Informationen zum Einstellen von Filtern und zur Suche in der AWS Marketplace finden Sie unterSuchen Sie in der AWS Marketplace Image Builder Builder-Konsole nach Image-Produkten.
   6. Geben Sie die restlichen Details wie gewohnt ein und wählen Sie „Rezept erstellen“.

> **Note**
>
> Wenn Ihr Image-Produktabonnement eine AWSTOE Build-Komponente enthält, können Sie diese aus der Liste Build-Komponenten auswählen. Wählen Sie `Third party managed` aus der Liste mit dem Komponenteneigentümertyp aus, um sie sich anzusehen. Wenn Ihr Produktabonnement eine AWSTOE Testkomponente beinhaltet, gehen Sie für die Liste der Testkomponenten genauso vor.

Amazon SNS SNS-Integration in Image Builder

Amazon Simple Notification Service (Amazon SNS) ist ein verwalteter Service, der die asynchrone Nachrichtenzustellung von Verlagen an Abonnenten (auch bekannt als Produzenten und Verbraucher) ermöglicht.

Sie können in Ihrer Infrastrukturkonfiguration ein SNS-Thema angeben. Wenn Sie ein Image erstellen oder eine Pipeline ausführen, kann Image Builder detaillierte Meldungen zu Ihrem Image-Status zu diesem Thema veröffentlichen. Wenn der Imagestatus einen der folgenden Zustände erreicht, veröffentlicht Image Builder eine Meldung:

- `AVAILABLE`
- `FAILED`

Ein Beispiel für eine SNS-Nachricht von Image Builder finden Sie unterFormat der SNS-Nachricht. Wenn Sie ein neues SNS-Thema erstellen möchten, finden Sie weitere Informationen unter Erste Schritte mit Amazon SNS im Amazon Simple Notification Service Developer Guide.

Themen rund um verschlüsseltes SNS

Wenn Ihr SNS-Thema verschlüsselt ist, müssen Sie der Image Builder Builder-Dienstrolle in der AWS KMS key Richtlinie die Erlaubnis erteilen, die folgenden Aktionen auszuführen:

- `kms:Decrypt`
- `kms:GenerateDataKey`

> **Note**
>
> Wenn Ihr SNS-Thema verschlüsselt ist, muss sich der Schlüssel, der dieses Thema verschlüsselt, in dem Konto befinden, auf dem der Image Builder Builder-Dienst ausgeführt wird. Image Builder kann keine Benachrichtigungen an SNS-Themen senden, die mit Schlüsseln anderer Konten verschlüsselt sind.

Beispiel für das Hinzufügen einer KMS-Schlüsselrichtlinie

Das folgende Beispiel zeigt den zusätzlichen Abschnitt, den Sie der KMS-Schlüsselrichtlinie hinzufügen. Verwenden Sie den Amazon-Ressourcennamen (ARN) für die mit dem IAM-Service verknüpfte Rolle, die Image Builder unter Ihrem Konto erstellt hat, als Sie ein Image Builder zum ersten Mal erstellt haben. Weitere Informationen zur serviceverknüpften Image Builder Builder-Rolle finden Sie unterVerwenden von serviceverknüpften Rollen für EC2 Image Builder.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::123456789012:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder"
    },
    "Action": [
      "kms:GenerateDataKey*",
      "kms:Decrypt"
    ],
    "Resource": "*"
  }]
}
```

Sie können eine der folgenden Methoden verwenden, um den ARN abzurufen.

AWS Management Console

Gehen Sie wie folgt vor, um den ARN für die serviceverknüpfte Rolle abzurufen AWS Management Console, die Image Builder unter Ihrem Konto erstellt hat:

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Roles aus.

3. Suchen Sie nach dem folgenden Rollennamen**`ImageBuilder`**, und wählen Sie ihn aus den Ergebnissen aus:`AWSServiceRoleForImageBuilder`. Daraufhin wird die Seite mit den Rollendetails angezeigt.
4. Um den ARN in Ihre Zwischenablage zu kopieren, wählen Sie das Symbol neben dem ARN-Namen.

AWS CLI

Verwenden Sie den IAM-Befehl get-role wie folgt, um den ARN für die dienstverknüpfte Rolle abzurufen AWS CLI, die Image Builder unter Ihrem Konto erstellt hat.

```
aws iam get-role --role-name AWSServiceRoleForImageBuilder
```

Teilweise Beispielausgabe:

```
{
    "Role": {
        "Path": "/aws-service-role/imagebuilder.amazonaws.com/",
        "RoleName": "AWSServiceRoleForImageBuilder",
        ...
        "Arn": "arn:aws:iam::123456789012:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder",
        ...
}
```

Format der SNS-Nachricht

Nachdem Image Builder eine Nachricht zu Ihrem Amazon SNS SNS-Thema veröffentlicht hat, können andere Dienste, die das Thema abonnieren, nach dem Nachrichtenformat filtern und feststellen, ob es die Kriterien für weitere Maßnahmen erfüllt. Beispielsweise kann eine Erfolgsmeldung eine Aufgabe zur Aktualisierung eines AWS Systems Manager Parameterspeichers oder zum Starten eines externen Compliance-Test-Workflows für das Ausgabe-AMI initiieren.

Das folgende Beispiel zeigt die JSON-Nutzlast für eine typische Nachricht, die Image Builder veröffentlicht, wenn ein Pipeline-Build vollständig ausgeführt wird, und erstellt ein Linux-Image.

```
{
  "versionlessArn": "arn:aws:imagebuilder:us-west-1:123456789012:image/example-linux-
image",
```

```
  "semver": 123794003928538027489912 4227,
  "arn": "arn:aws:imagebuilder:us-west-1:123456789012:image/example-linux-
image/1.0.0/3",
  "name": "example-linux-image",
  "version": "1.0.0",
  "type": "AMI",
  "buildVersion": 3,
  "state": {
    "status": "AVAILABLE"
  },
  "platform": "Linux",
  "imageRecipe": {
    "arn": "arn:aws:imagebuilder:us-west-1:123456789012:image-recipe/example-linux-
image/1.0.0",
    "name": "amjule-barebones-linux",
    "version": "1.0.0",
    "components": [
      {
        "componentArn": "arn:aws:imagebuilder:us-west-1:123456789012:component/update-
linux/1.0.2/1"
      }
    ],
    "platform": "Linux",
    "parentImage": "arn:aws:imagebuilder:us-west-1:987654321098:image/amazon-linux-2-
x86/2022.6.14/1",
    "blockDeviceMappings": [
      {
        "deviceName": "/dev/xvda",
        "ebs": {
          "encrypted": false,
          "deleteOnTermination": true,
          "volumeSize": 8,
          "volumeType": "gp2"
        }
      }
    ],
    "dateCreated": "Feb 24, 2021 12:31:54 AM",
    "tags": {
      "internalId": "1a234567-8901-2345-bcd6-ef7890123456",
      "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:image-recipe/example-
linux-image/1.0.0"
    },
    "workingDirectory": "/tmp",
    "accountId": "462045008730"
```

```
  },
  "sourcePipelineArn": "arn:aws:imagebuilder:us-west-1:123456789012:image-pipeline/
example-linux-pipeline",
  "infrastructureConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-1:123456789012:infrastructure-configuration/
example-linux-infra-config-uswest1",
    "name": "example-linux-infra-config-uswest1",
    "instanceProfileName": "example-linux-ib-baseline-admin",
    "tags": {
      "internalId": "234abc56-d789-0123-a4e5-6b789d012c34",
      "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:infrastructure-
configuration/example-linux-infra-config-uswest1"
    },
    "logging": {
      "s3Logs": {
        "s3BucketName": "12345-example-linux-testbucket-uswest1"
      }
    },
    "keyPair": "example-linux-key-pair-uswest1",
    "terminateInstanceOnFailure": true,
    "snsTopicArn": "arn:aws:sns:us-west-1:123456789012:example-linux-ibnotices-
uswest1",
    "dateCreated": "Feb 24, 2021 12:31:55 AM",
    "accountId": "123456789012"
  },
  "imageTestsConfigurationDocument": {
    "imageTestsEnabled": true,
    "timeoutMinutes": 720
  },
  "distributionConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-1:123456789012:distribution-configuration/
example-linux-distribution",
    "name": "example-linux-distribution",
    "dateCreated": "Feb 24, 2021 12:31:56 AM",
    "distributions": [
      {
        "region": "us-west-1",
        "amiDistributionConfiguration": {}
      }
    ],
    "tags": {
      "internalId": "345abc67-8910-12d3-4ef5-67a8b90c12de",
      "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:distribution-
configuration/example-linux-distribution"
```

```
    },
    "accountId": "123456789012"
  },
  "dateCreated": "Jul 28, 2022 1:13:45 AM",
  "outputResources": {
    "amis": [
      {
        "region": "us-west-1",
        "image": "ami-01a23bc4def5a6789",
        "name": "example-linux-image 2022-07-28T01-14-17.416Z",
        "accountId": "123456789012"
      }
    ]
  },
  "buildExecutionId": "ab0cd12e-34fa-5678-b901-2c3456d789e0",
  "testExecutionId": "6a7b8901-cdef-234a-56b7-8cd89ef01234",
  "distributionJobId": "1f234567-8abc-9d0e-1234-fa56b7c890de",
  "integrationJobId": "432109b8-afe7-6dc5-4321-0ba98f7654e3",
  "accountId": "123456789012",
  "osVersion": "Amazon Linux 2",
  "enhancedImageMetadataEnabled": true,
  "buildType": "USER_INITIATED",
  "tags": {
    "internalId": "901e234f-a567-89bc-0123-d4e567f89a01",
    "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:image/example-linux-
image/1.0.0/3"
  }
}
```

Das folgende Beispiel zeigt die JSON-Nutzlast für eine typische Nachricht, die Image Builder für einen Pipeline-Build-Fehler für ein Linux-Image veröffentlicht.

```
{
  "versionlessArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-
image",
  "semver": 123794003928538027489912423 1,
  "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-image/1.0.0/7",
  "name": "My Example Image",
  "version": "1.0.0",
  "type": "AMI",
  "buildVersion": 7,
  "state": {
    "status": "FAILED",
```

```
    "reason": "Image Failure reason."
  },
  "platform": "Linux",
  "imageRecipe": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-
image/1.0.0",
    "name": "My Example Image",
    "version": "1.0.0",
    "description": "Testing Image recipe",
    "components": [
      {
        "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/my-
example-image-component/1.0.0/1"
      }
    ],
    "platform": "Linux",
    "parentImage": "ami-0cd12345db678d90f",
    "dateCreated": "Jun 21, 2022 11:36:14 PM",
    "tags": {
      "internalId": "1a234567-8901-2345-bcd6-ef7890123456",
      "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-
example-image/1.0.0"
    },
    "accountId": "123456789012"
  },
  "sourcePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-
example-image-pipeline",
  "infrastructureConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/
my-example-infra-config",
    "name": "SNS topic Infra config",
    "description": "An example that will retain instances of failed builds",
    "instanceTypes": [
      "t2.micro"
    ],
    "instanceProfileName": "EC2InstanceProfileForImageBuilder",
    "tags": {
      "internalId": "234abc56-d789-0123-a4e5-6b789d012c34",
      "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-
configuration/my-example-infra-config"
    },
    "terminateInstanceOnFailure": true,
    "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:example-pipeline-notification-
topic",
```

```
    "dateCreated": "Jul 5, 2022 7:31:53 PM",
    "accountId": "123456789012"
  },
  "imageTestsConfigurationDocument": {
    "imageTestsEnabled": true,
    "timeoutMinutes": 720
  },
  "distributionConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-
example-distribution-config",
    "name": "New distribution config",
    "dateCreated": "Dec 3, 2021 9:24:22 PM",
    "distributions": [
      {
        "region": "us-west-2",
        "amiDistributionConfiguration": {},
        "fastLaunchConfigurations": [
          {
            "enabled": true,
            "snapshotConfiguration": {
              "targetResourceCount": 2
            },
            "maxParallelLaunches": 2,
            "launchTemplate": {
              "launchTemplateId": "lt-01234567890"
            },
            "accountId": "123456789012"
          }
        ]
      }
    ],
    "tags": {
      "internalId": "1fecd23a-4f56-7f89-01e2-345678abbe90",
      "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-
configuration/my-example-distribution-config"
    },
    "accountId": "123456789012"
  },
  "dateCreated": "Jul 5, 2022 7:40:15 PM",
  "outputResources": {
    "amis": []
  },
  "accountId": "123456789012",
  "enhancedImageMetadataEnabled": true,
```

```
  "buildType": "SCHEDULED",
  "tags": {
    "internalId": "456c78b9-0e12-3f45-afb6-7e89b0f1a23b",
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-
image/1.0.0/7"
  }
}
```

Compliance-Produkte für Ihre Image Builder Builder-Images

Angesichts der sich ständig weiterentwickelnden Sicherheitsstandards kann es eine Herausforderung sein, die Einhaltung von Vorschriften aufrechtzuerhalten und Ihr Unternehmen vor Cyberbedrohungen zu schützen. Um sicherzustellen, dass Ihre benutzerdefinierten Images konform sind und dies auch durch automatische Updates, wenn Herausgeber neue Versionen veröffentlichen, so bleibt, lässt sich Image Builder in AWS Marketplace Compliance-Produkte und AWSTOE -Komponenten integrieren.

Image Builder lässt sich in die folgenden Compliance-Produkte integrieren:

- Härtung von Benchmarks durch das Center for Internet Security (CIS)

  Sie können CIS Hardened Images und die zugehörigen CIS-Hardening-Komponenten verwenden, um benutzerdefinierte Images zu erstellen, die den neuesten CIS-Benchmarks-Level-1-Richtlinien entsprechen. CIS Hardened Images sind verfügbar in AWS Marketplace. Weitere Informationen zur Einrichtung und Verwendung von CIS Hardened Images und Hardening-Komponenten finden Sie in den Schnellstartanleitungen auf dem Supportportal der CIS-Website.

  > **Note**
  >
  > Wenn Sie ein CIS Hardened Image abonnieren, erhalten Sie auch Zugriff auf die zugehörige Build-Komponente, die ein Skript ausführt, um die CIS Benchmark Level 1-Richtlinien für Ihre Konfiguration durchzusetzen. Weitere Informationen finden Sie unter CIS-Härtungskomponenten.

- Technische Implementierungsleitfäden im Bereich Sicherheit (STIG)

  Aus Gründen der STIG-Konformität können Sie von Amazon verwaltete AWS Task Orchestrator and Executor (AWSTOE) STIG-Komponenten in Ihren Image Builder Builder-Rezepten verwenden. STIG-Komponenten scannen Ihre Build-Instance auf Fehlkonfigurationen und führen ein

Korrekturskript aus, um die gefundenen Probleme zu korrigieren. Wir können die STIG-Konformität für die Images, die Sie mit Image Builder erstellen, nicht garantieren. Sie müssen mit dem Compliance-Team Ihres Unternehmens zusammenarbeiten, um zu überprüfen, ob Ihr endgültiges Image konform ist. Eine vollständige Liste der AWSTOE STIG-Komponenten, die Sie in Ihren Image Builder Builder-Rezepten verwenden können, finden Sie unterVon Amazon verwaltete STIG-Härtungskomponenten für EC2 Image Builder.

Überwachen Sie Ereignisse und Protokolle in EC2 Image Builder

Um die Zuverlässigkeit, Verfügbarkeit und Leistung Ihrer EC2 Image Builder Builder-Pipelines aufrechtzuerhalten, ist es wichtig, Ereignisse und Protokolle zu überwachen. Ereignisse und Protokolle helfen Ihnen dabei, das Gesamtbild zu sehen und die Details zu untersuchen, wenn ein API-Aufruf fehlschlägt. Image Builder lässt sich in Dienste integrieren, die Benachrichtigungen senden und automatische Antworten auslösen können, wenn Ereignisse den von Ihnen konfigurierten Kriterien entsprechen.

In den folgenden Themen werden Überwachungstechniken beschrieben, die Sie mithilfe von Diensten verwenden können, die in Image Builder integriert sind.

Überwachen Sie Ereignisse und Protokolle

- Protokollieren EC2 Image Builder Builder-API-Aufrufen mit AWS CloudTrail

Protokollieren EC2 Image Builder Builder-API-Aufrufen mit AWS CloudTrail

EC2 Image Builder ist in einen Dienst integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen bereitstellt, die alle API-Aufrufe von einem Benutzer, einer Rolle oder einem AWS Dienst über die Image Builder Builder-API ausführen. CloudTrail erfasst Image Builder als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der Image Builder Builder-Konsole und Codeaufrufen für die Image Builder Builder-API-Operationen.

Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen an einen S3-Bucket aktivieren, einschließlich Ereignissen für Image Builder. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse trotzdem in der CloudTrail Konsole im Ereignisverlauf anzeigen. Anhand der von CloudTrail gesammelten Informationen können Sie die Anfrage an Image Builder, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details ermitteln.

Weitere Informationen CloudTrail dazu finden Sie im AWS CloudTrail Benutzerhandbuch.

Informationen zu Image Builder in CloudTrail

CloudTrail ist auf Ihrem aktiviert AWS-Konto , wenn Sie das Konto erstellen. Wenn eine Aktivität in Image Builder auftritt, wird diese Aktivität zusammen mit anderen CloudTrail AWS Dienstereignissen im Ereignisverlauf in einem Ereignis aufgezeichnet. Sie können aktuelle Ereignisse in Ihrem anzeigen, suchen und herunterladen AWS-Konto. Weitere Informationen finden Sie unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS-Konto, einschließlich Ereignissen für Image Builder, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen S3-Bucket. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser für alle AWS-Regionen-Regionen. Der Trail protokolliert Ereignisse aus allen Regionen der AWS Partition und übermittelt die Protokolldateien an den von Ihnen angegebenen S3-Bucket. Darüber hinaus können Sie andere konfigurieren, AWS-Services um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren. Weitere Informationen finden Sie hier:

- Übersicht zum Erstellen eines Trails.
- CloudTrail unterstützte Dienste und Integrationen.
- Konfiguration von Amazon SNS SNS-Benachrichtigungen für CloudTrail.
- Empfang von CloudTrail Protokolldateien aus mehreren Regionen.
- Empfangen von CloudTrail Protokolldateien von mehreren Konten.

CloudTrail protokolliert alle Image Builder-Aktionen, die in der EC2 Image Builder API-Referenz dokumentiert sind. Aufrufe der `StartImagePipelineExecution` Aktionen `CreateImagePipelineUpdateInfrastructureConfiguration`, und generieren beispielsweise Einträge in den CloudTrail Protokolldateien.

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 Anfrage von einem anderen AWS Dienst gestellt wurde.

Weitere Informationen zum Ermitteln, wer ein Ereignis angefordert hat, finden Sie im CloudTrail UserIdentity-Element.

Sicherheit in EC2 Image Builder

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud selbst und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS-Services in der AWS Cloud läuft. 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 EC2 Image Builder gelten, finden Sie unter AWS-Services Umfang nach AWS Compliance-Programm-Services .
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Service, den Sie nutzen. Sie sind auch für andere Faktoren verantwortlich, etwa für die 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 von Image Builder anwenden. In den folgenden Themen erfahren Sie, wie Sie Image Builder konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere verwenden können AWS-Services , die Ihnen helfen, Ihre Image Builder Builder-Ressourcen zu überwachen und zu sichern.

Themen

Datenschutz in EC2 Image Builder

Das AWS Modell der gilt für den Datenschutz in EC2 Image Builder. Wie in diesem Modell beschrieben, AWS ist verantwortlich für den Schutz der globalen Infrastruktur, auf der AWS Cloud alle Systeme laufen. Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Sie sind auch für die Sicherheitskonfiguration und die Verwaltungsaufgaben für die von Ihnen verwendeten AWS-Services verantwortlich. Weitere Informationen zum Datenschutz finden Sie unter Häufig gestellte Fragen zum Datenschutz. Informationen zum Datenschutz in Europa finden Sie im Blog-Beitrag AWS -Modell der geteilten Verantwortung und in der DSGVO im AWS -Sicherheitsblog.

Aus Datenschutzgründen empfehlen wir, dass Sie AWS-Konto Anmeldeinformationen schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einrichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem empfehlen wir, die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- Verwenden Sie SSL/TLS, um mit Ressourcen zu kommunizieren. AWS Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein. AWS CloudTrail
- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen darin enthaltenen Standardsicherheitskontrollen AWS-Services.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu schützen.
- Wenn Sie für den Zugriff AWS über eine Befehlszeilenschnittstelle oder eine API FIPS 140-2-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-2.

Wir empfehlen dringend, in Freitextfeldern, z. B. im Feld Name, keine vertraulichen oder sensiblen Informationen wie die E-Mail-Adressen Ihrer Kunden einzugeben. Dies gilt auch, wenn Sie mit Image Builder oder anderen Geräten arbeiten und die Konsole, die API oder AWS SDKs AWS-Services verwenden. AWS CLI Alle Daten, die Sie in Tags oder Freitextfelder eingeben, die für Namen verwendet werden, können für Abrechnungs- oder Diagnoseprotokolle verwendet werden. Wenn Sie eine URL für einen externen Server bereitstellen, empfehlen wir dringend, keine

Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL einzuschließen.

Verschlüsselung und Schlüsselverwaltung in EC2 Image Builder

Image Builder verschlüsselt Daten während der Übertragung und im Ruhezustand standardmäßig mit einem diensteigenen KMS-Schlüssel, mit Ausnahme der folgenden:

- Benutzerdefinierte Komponenten — Image Builder verschlüsselt benutzerdefinierte Komponenten mit Ihrem Standard-KMS-Schlüssel oder einem diensteigenen KMS-Schlüssel.
- Image-Workflows — Image Builder kann Ihre Image-Workflows mit einem vom Kunden verwalteten Schlüssel verschlüsseln, wenn Sie den Schlüssel bei der Workflow-Erstellung angeben. Image Builder übernimmt die Verschlüsselung und Entschlüsselung mit Ihrem Schlüssel, um die Workflows auszuführen, die Sie für Ihre Bilder konfiguriert haben.

Sie können Ihre eigenen Schlüssel über AWS KMS verwalten. Sie sind jedoch nicht berechtigt, den Image Builder-KMS-Schlüssel zu verwalten, der Image Builder gehört. Weitere Informationen zur Verwaltung Ihrer KMS-Schlüssel mit AWS Key Management Service finden Sie unter Erste Schritte im AWS Key Management Service Entwicklerhandbuch.

Verschlüsselungskontext

Um eine zusätzliche Integritäts- und Authentizitätsprüfung Ihrer verschlüsselten Daten durchzuführen, haben Sie die Möglichkeit, bei der Verschlüsselung der Daten einen Verschlüsselungskontext einzubeziehen. Wenn eine Ressource mit einem Verschlüsselungskontext verschlüsselt ist, wird der Kontext AWS KMS kryptografisch an den Chiffretext gebunden. Die Ressource kann nur entschlüsselt werden, wenn der Anforderer eine exakte Übereinstimmung mit dem Kontext unter Berücksichtigung der Groß- und Kleinschreibung angibt.

Die Richtlinienbeispiele in diesem Abschnitt verwenden einen Verschlüsselungskontext, der dem Amazon-Ressourcennamen (ARN) einer Image Builder Builder-Workflow-Ressource ähnelt.

Verschlüsseln Sie Image-Workflows mit einem vom Kunden verwalteten Schlüssel

Um eine zusätzliche Schutzebene hinzuzufügen, können Sie Ihre Image Builder Builder-Workflow-Ressourcen mit Ihrem eigenen, vom Kunden verwalteten Schlüssel verschlüsseln. Wenn Sie Ihren vom Kunden verwalteten Schlüssel zum Verschlüsseln der von Ihnen erstellten Image Builder-Workflows verwenden, müssen Sie in der Schlüsselrichtlinie Zugriff gewähren, damit Image Builder Ihren Schlüssel beim Verschlüsseln und Entschlüsseln von Workflow-Ressourcen verwenden kann.

Sie können den Zugriff jederzeit widerrufen. Image Builder hat jedoch keinen Zugriff auf Workflows, die bereits verschlüsselt sind, wenn Sie den Zugriff auf den Schlüssel widerrufen.

Das Verfahren, um Image Builder Zugriff auf die Verwendung Ihres vom Kunden verwalteten Schlüssels zu gewähren, besteht aus zwei Schritten:

Schritt 1: Wichtige Richtlinienberechtigungen für Image Builder Builder-Workflows hinzufügen

Damit Image Builder Workflow-Ressourcen bei der Erstellung oder Verwendung dieser Workflows ver- und entschlüsseln kann, müssen Sie in der KMS-Schlüsselrichtlinie Berechtigungen angeben.

Diese Beispielschlüsselrichtlinie gewährt Image Builder Builder-Pipelines Zugriff, um Workflow-Ressourcen während des Erstellungsprozesses zu verschlüsseln und Workflow-Ressourcen zu entschlüsseln, um sie zu verwenden. Die Richtlinie gewährt auch Schlüsseladministratoren Zugriff. Der Verschlüsselungskontext und die Ressourcenspezifikation verwenden einen Platzhalter, um alle Regionen abzudecken, in denen Sie über Workflow-Ressourcen verfügen.

Als Voraussetzung für die Verwendung von Image-Workflows haben Sie eine IAM-Workflow-Ausführungsrolle erstellt, die Image Builder die Erlaubnis erteilt, Workflow-Aktionen auszuführen. Der Principal für die erste Anweisung, der hier im Beispiel für eine wichtige Richtlinie gezeigt wird, muss Ihre IAM-Workflow-Ausführungsrolle angeben.

Weitere Informationen zu vom Kunden verwalteten Schlüsseln finden Sie unter Verwaltung des Zugriffs auf vom Kunden verwaltete Schlüssel im AWS Key Management Service Entwicklerhandbuch.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "Allow access to build images with encrypted workflow",
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/YourImageBuilderExecutionRole"
   },
   "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
   ],
   "Resource": "*",
   "Condition": {
    "StringLike": {
```

```
      "kms:EncryptionContext:aws:imagebuilder:arn":
 "arn:aws:imagebuilder:*:111122223333:workflow/*"
    }
   }
  },
  {
   "Sid": "Allow access for key administrators",
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:root"
   },
   "Action": [
    "kms:*"
   ],
   "Resource": "arn:aws:kms:*:111122223333:key/"
  }
 ]
}
```

Schritt 2: Gewähren Sie Schlüsselzugriff auf Ihre Workflow-Ausführungsrolle

Die IAM-Rolle, die Image Builder für die Ausführung Ihrer Workflows annimmt, benötigt die Erlaubnis, Ihren vom Kunden verwalteten Schlüssel zu verwenden. Ohne Zugriff auf Ihren Schlüssel kann Image Builder Ihre Workflow-Ressourcen nicht damit ver- oder entschlüsseln.

Bearbeiten Sie die Richtlinie für Ihre Workflow-Ausführungsrolle, um die folgende Richtlinienerklärung hinzuzufügen.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "Allow access to the workflow key",
   "Effect": "Allow",
   "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
   ],
   "Resource": "arn:aws:kms:us-west-2:111122223333:key/key_ID",
   "Condition": {
    "StringLike": {
     "kms:EncryptionContext:aws:imagebuilder:arn":
 "arn:aws:imagebuilder:*:111122223333:workflow/*"
```

```
    }
   }
  }
 ]
}
```

AWS CloudTrail Ereignisse für Image-Workflows

Die folgenden Beispiele zeigen typische AWS CloudTrail Einträge zum Verschlüsseln und Entschlüsseln von Image-Workflows, die mit einem vom Kunden verwalteten Schlüssel gespeichert werden.

Beispiel: GenerateDataKey

Dieses Beispiel zeigt, wie ein CloudTrail Ereignis aussehen könnte, wenn Image Builder die AWS KMS GenerateDataKey API-Aktion aus der Image Builder CreateWorkflow Builder-API-Aktion aufruft. Image Builder muss einen neuen Workflow verschlüsseln, bevor die Workflow-Ressource erstellt wird.

```
{
 "eventVersion": "1.08",
 "userIdentity": {
  "type": "AssumedRole",
  "principalId": "PRINCIPALID1234567890:workflow-role-name",
  "arn": "arn:aws:sts::111122223333:assumed-role/Admin/workflow-role-name",
  "accountId": "111122223333",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
  "sessionContext": {
   "sessionIssuer": {
    "type": "Role",
    "principalId": "PRINCIPALID1234567890",
    "arn": "arn:aws:iam::111122223333:role/Admin",
    "accountId": "111122223333",
    "userName": "Admin"
   },
   "webIdFederationData": {},
   "attributes": {
    "creationDate": "2023-11-21T20:29:31Z",
    "mfaAuthenticated": "false"
   }
  },
  "invokedBy": "imagebuilder.amazonaws.com"
 },
```

```
 "eventTime": "2023-11-21T20:31:03Z",
 "eventSource": "kms.amazonaws.com",
 "eventName": "GenerateDataKey",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "imagebuilder.amazonaws.com",
 "userAgent": "imagebuilder.amazonaws.com",
 "requestParameters": {
  "encryptionContext": {
   "aws:imagebuilder:arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/
sample-encrypted-workflow/1.0.0/*",
   "aws-crypto-public-key": "key value"
  },
  "keyId": "arn:aws:kms:us-west-2:111122223333:alias/ExampleKMSKey",
  "numberOfBytes": 32
 },
 "responseElements": null,
 "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
 "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
 "readOnly": true,
 "resources": [
  {
   "accountId": "111122223333",
   "type": "AWS::KMS::Key",
   "ARN": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLEzzzzz"
  }
 ],
 "eventType": "AwsApiCall",
 "managementEvent": true,
 "recipientAccountId": "111122223333",
 "eventCategory": "Management"
}
```

Beispiel: Entschlüsseln

Dieses Beispiel zeigt, wie ein CloudTrail Ereignis aussehen könnte, wenn Image Builder die AWS KMS Decrypt API-Aktion aus der Image Builder GetWorkflow Builder-API-Aktion aufruft. Image Builder Builder-Pipelines müssen eine Workflow-Ressource entschlüsseln, bevor sie sie verwenden können.

```
{
 "eventVersion": "1.08",
 "userIdentity": {
  "type": "AssumedRole",
```

```
  "principalId": "PRINCIPALID1234567890:workflow-role-name",
  "arn": "arn:aws:sts::111122223333:assumed-role/Admin/workflow-role-name",
  "accountId": "111122223333",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
  "sessionContext": {
   "sessionIssuer": {
    "type": "Role",
    "principalId": "PRINCIPALID1234567890",
    "arn": "arn:aws:iam::111122223333:role/Admin",
    "accountId": "111122223333",
    "userName": "Admin"
   },
   "webIdFederationData": {},
   "attributes": {
    "creationDate": "2023-11-21T20:29:31Z",
    "mfaAuthenticated": "false"
   }
  },
  "invokedBy": "imagebuilder.amazonaws.com"
 },
 "eventTime": "2023-11-21T20:34:25Z",
 "eventSource": "kms.amazonaws.com",
 "eventName": "Decrypt",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "imagebuilder.amazonaws.com",
 "userAgent": "imagebuilder.amazonaws.com",
 "requestParameters": {
  "keyId": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLEzzzzz",
  "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
  "encryptionContext": {
   "aws:imagebuilder:arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/
sample-encrypted-workflow/1.0.0/*",
   "aws-crypto-public-key": "ABC123def4567890abc12345678/90dE/F123abcDEF+4567890abc123D
+ef1=="
  }
 },
 "responseElements": null,
 "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEbbbbb",
 "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
 "readOnly": true,
 "resources": [
  {
   "accountId": "111122223333",
```

```
    "type": "AWS::KMS::Key",
    "ARN": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLEzzzzz"
   }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}
```

Datenspeicherung in EC2 Image Builder

Image Builder speichert keine Ihrer Protokolle im Service. Alle Protokolle werden auf Ihrer Amazon EC2 EC2-Instance gespeichert, die zum Erstellen des Images verwendet wird, oder in Ihren Systems Manager Manager-Automatisierungsprotokollen.

Datenschutz bei netzwerkinternem Datenverkehr in EC2 Image Builder

Verbindungen zwischen Image Builder und lokalen Standorten, zwischen AZs innerhalb einer AWS Region und zwischen AWS Regionen werden über HTTPS gesichert. Es gibt keine direkten Verbindungen zwischen Konten.

Identity and Access Management für EC2 Image Builder

Themen

Zielgruppe

Die Art und Weise, wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von der Arbeit ab, die Sie in Image Builder ausführen.

Dienstbenutzer — Wenn Sie den Image Builder Builder-Dienst für Ihre Arbeit verwenden, stellt Ihnen Ihr Administrator die erforderlichen Anmeldeinformationen und Berechtigungen zur Verfügung. Da Sie für Ihre Arbeit mehr Image Builder Builder-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. Wenn Sie in Image Builder nicht auf eine Funktion zugreifen können, finden Sie weitere Informationen unterProblembehandlung bei EC2 Image Builder Builder-Identität und -Zugriff.

Dienstadministrator — Wenn Sie in Ihrem Unternehmen für die Image Builder-Ressourcen verantwortlich sind, haben Sie wahrscheinlich vollen Zugriff auf Image Builder. Es ist Ihre Aufgabe, zu bestimmen, auf welche Image Builder Builder-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 mit Image Builder verwenden kann, finden Sie unterSo funktioniert EC2 Image Builder mit IAM.

IAM-Administrator — Wenn Sie ein IAM-Administrator sind, möchten Sie vielleicht mehr darüber erfahren, wie Sie Richtlinien schreiben können, um den Zugriff auf Image Builder zu verwalten. Beispiele für identitätsbasierte Image Builder Builder-Richtlinien, die Sie in IAM verwenden können, finden Sie unter. Identitätsbasierte Richtlinien von Image Builder

Authentifizierung mit Identitäten

Ausführliche Informationen zur Authentifizierung von Personen und Prozessen in Ihrem AWS-Konto finden Sie unter Identitäten im IAM-Benutzerhandbuch.

So funktioniert EC2 Image Builder mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf Image Builder zu verwalten, sollten Sie sich darüber informieren, welche IAM-Funktionen mit Image Builder verwendet werden können.

Einen allgemeinen Überblick darüber, wie Image Builder 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 Image Builder

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 Image Builder

Beispiele für identitätsbasierte Image Builder Builder-Richtlinien finden Sie unter. Identitätsbasierte Richtlinien von Image Builder

Ressourcenbasierte Richtlinien in Image Builder

Unterstützt ressourcenbasierte Richtlinien	Ja

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

Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Prinzipal in einer ressourcenbasierten Richtlinie angeben. Durch das Hinzufügen eines kontoübergreifenden Auftraggebers zu einer ressourcenbasierten Richtlinie ist nur die halbe Vertrauensbeziehung eingerichtet. Wenn sich der Prinzipal und die Ressource unterscheiden AWS-Konten, muss ein IAM-Administrator des vertrauenswürdigen Kontos auch der Prinzipalentität (Benutzer oder Rolle) die Berechtigung zum Zugriff auf die Ressource erteilen. Sie erteilen Berechtigungen, indem Sie der juristischen Stelle eine identitätsbasierte Richtlinie anfügen. Wenn jedoch eine ressourcenbasierte Richtlinie Zugriff auf einen Prinzipal in demselben Konto gewährt, ist keine zusätzliche identitätsbasierte Richtlinie erforderlich. Weitere Informationen finden Sie unter Wie sich IAM-Rollen von ressourcenbasierten Richtlinien unterscheiden im IAM-Benutzerhandbuch.

Richtlinienaktionen für Image Builder

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 Image Builder-Aktionen finden Sie unter Von EC2 Image Builder definierte Aktionen in der Service Authorization Reference.

Richtlinienaktionen in Image Builder verwenden das folgende Präfix vor der Aktion:

```
imagebuilder
```

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Kommata:

```
"Action": [
      "imagebuilder:action1",
      "imagebuilder:action2"
         ]
```

Beispiele für identitätsbasierte Image Builder Builder-Richtlinien finden Sie unter. Identitätsbasierte Richtlinien von Image Builder

Richtlinienressourcen für Image Builder

Unterstützt Richtlinienressourcen	Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was 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 Image Builder-Ressourcentypen und ihrer ARNs finden Sie unter Von EC2 Image Builder definierte Ressourcen in der Service Authorization Reference. Informationen darüber, mit welchen Aktionen Sie den ARN jeder Ressource angeben können, finden Sie unter Von EC2 Image Builder definierte Aktionen.

Beispiele für identitätsbasierte Image Builder Builder-Richtlinien finden Sie unter. Identitätsbasierte Richtlinien von Image Builder

Schlüssel für Richtlinienbedingungen für Image Builder

Unterstützt servicespezifische Richtlini enbedingungsschlüssel	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 `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 Image Builder-Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für EC2 Image Builder in der Service Authorization Reference. Informationen zu den Aktionen und Ressourcen, mit denen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter Von EC2 Image Builder definierte Aktionen.

Beispiele für identitätsbasierte Image Builder Builder-Richtlinien finden Sie unter. Identitätsbasierte Richtlinien von Image Builder

ACLs in Image Builder

Unterstützt ACLs	Nein

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.

ABAC mit Image Builder

Unterstützt ABAC (Tags in Richtlinien)	Teilweise

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen definiert werden. In AWS werden diese Attribute als Tags bezeichnet. Sie können Tags an IAM-Entitäten (Benutzer oder Rollen) und an viele AWS Ressourcen anhängen. Das Markieren von Entitäten und Ressourcen ist der erste Schritt von ABAC. Anschließend entwerfen Sie ABAC-Richtlinien, um Operationen zuzulassen, wenn das Tag des Prinzipals mit dem Tag der Ressource übereinstimmt, auf die sie zugreifen möchten.

ABAC ist in Umgebungen hilfreich, die schnell wachsen, und unterstützt Sie in Situationen, in denen die Richtlinienverwaltung mühsam wird.

Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, oder Bedingung `aws:TagKeys` verwenden.

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.

Temporäre Anmeldeinformationen mit Image Builder verwenden

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 Prinzipalberechtigungen für Image Builder

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 Image Builder

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 Image Builder Builder-Funktionalität beeinträchtigt werden. Bearbeiten Sie Servicerollen nur, wenn Image Builder Sie dazu anleitet.

Serviceverknüpfte Rollen für Image Builder

Unterstützt serviceverknüpfte Rollen	Nein

Eine serviceverknüpfte 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.

Einzelheiten zur dienstverknüpften Image Builder Builder-Rolle finden Sie unterVerwenden von serviceverknüpften Rollen für EC2 Image Builder.

Identitätsbasierte Richtlinien von Image Builder

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. Image Builder unterstützt bestimmte Aktionen, Ressourcen und Bedingungsschlüssel. Informationen zu allen Elementen, die Sie in einer JSON-

Richtlinie verwenden, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon EC2 Image Builder im IAM-Benutzerhandbuch.

Aktionen

Richtlinienaktionen in Image Builder verwenden das folgende Präfix vor der Aktion:`imagebuilder:`. Richtlinienanweisungen müssen entweder ein – `Action`oder ein `NotAction`-Element enthalten. Image Builder definiert eigene Aktionen, die Aufgaben beschreiben, die Sie mit diesem Dienst ausführen können.

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie wie folgt durch Kommata:

```
"Action": [
      "imagebuilder:action1",
      "imagebuilder:action2"
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `List` beginnen, einschließlich der folgenden Aktion:

```
"Action": "imagebuilder:List*"
```

Eine Liste der Image Builder Builder-Aktionen finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS-Services im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Ausführliche Informationen zur Verwaltung des Zugriffs AWS durch Erstellen von Richtlinien und deren Anhängen an IAM-Identitäten oder AWS Ressourcen finden Sie unter Richtlinien und Berechtigungen im IAM-Benutzerhandbuch.

Die IAM-Rolle, die Sie Ihrem Instanzprofil zuordnen, muss über Berechtigungen zum Ausführen der in Ihrem Image enthaltenen Build- und Testkomponenten verfügen. Die folgenden IAM-Rollenrichtlinien müssen der IAM-Rolle angehängt werden, die dem Instanzprofil zugeordnet ist:

- EC2InstanceProfileForImageBuilder
- EC2InstanceProfileForImageBuilderECRContainerBuilds
- AmazonSSMManagedInstanceCore

Ressourcen

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was 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": "*"
```

Die Image Builder Builder-Instance-Ressource hat den folgenden Amazon-Ressourcennamen (ARN).

```
arn:aws:imagebuilder:region:account-id:resource:resource-id
```

Weitere Informationen zum Format von ARNs finden Sie unter Amazon Resource Names (ARNs) und AWS Service Namespaces.

Um beispielsweise die `i-1234567890abcdef0` Instanz in Ihrer Anweisung anzugeben, verwenden Sie den folgenden ARN.

```
"Resource": "arn:aws:imagebuilder:us-east-1:123456789012:instance/i-1234567890abcdef0"
```

Um alle Instances anzugeben, die zu einem bestimmten Konto gehören, verwenden Sie den Platzhalter (*).

```
"Resource": "arn:aws:imagebuilder:us-east-1:123456789012:instance/*"
```

Einige Image Builder Builder-Aktionen, z. B. zum Erstellen von Ressourcen, können nicht für eine bestimmte Ressource ausgeführt werden. In diesen Fällen müssen Sie den Platzhalter (*) verwenden.

```
"Resource": "*"
```

Viele EC2 Image Builder API-Aktionen beinhalten mehrere Ressourcen. Um mehrere Ressourcen in einer einzigen Anweisung anzugeben, trennen Sie die ARNs durch Kommata voneinander.

```
"Resource": [
      "resource1",
      "resource2"
```

Bedingungsschlüssel

Image Builder stellt dienstspezifische Bedingungsschlüssel bereit und unterstützt die Verwendung einiger globaler Bedingungsschlüssel. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter AWS Globale Bedingungskontextschlüssel im IAM-Benutzerhandbuch. Die folgenden dienstspezifischen Bedingungsschlüssel werden bereitgestellt.

Imagebuilder: CreatedResourceTagKeys

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um den Zugriff nach dem Vorhandensein von Tag-Schlüsseln in der Anfrage zu filtern. Auf diese Weise können Sie die von Image Builder erstellten Ressourcen verwalten.

Verfügbarkeit — Dieser Schlüssel ist nur für die `UpdateInfrastructureConfiguration` APIs `CreateInfrastrucutreConfiguration` und verfügbar.

imagebuilder:/CreatedResourceTag<key>

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um den Zugriff nach den Tag-Schlüssel-Wert-Paaren zu filtern, die an die von Image Builder erstellte Ressource angehängt sind. Auf diese Weise können Sie Image Builder Builder-Ressourcen über definierte Tags verwalten.

Verfügbarkeit — Dieser Schlüssel ist nur für die `UpdateInfrastructureConfiguration` APIs `CreateInfrastrucutreConfiguration` und verfügbar.

ImageBuilder: EC2 MetadataHttpTokens

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um den Zugriff nach der in der Anfrage angegebenen HTTP-Token-Anforderung für EC2-Instance-Metadaten zu filtern.

Dieser Wert für diesen Schlüssel kann entweder `optional` oder `required` sein.

Verfügbarkeit — Dieser Schlüssel ist nur für die `UpdateInfrastructureConfiguration` APIs `CreateInfrastrucutreConfiguration` und verfügbar.

Imagebuilder: StatusTopicArn

Funktioniert mit Zeichenfolgenoperatoren.

Verwenden Sie diesen Schlüssel, um den Zugriff nach dem SNS-Themen-ARN in der Anfrage zu filtern, für die Terminalstatusbenachrichtigungen veröffentlicht werden.

Verfügbarkeit — Dieser Schlüssel ist nur für die `UpdateInfrastructureConfiguration` APIs `CreateInfrastrucutreConfiguration` und verfügbar.

Beispiele

Beispiele für identitätsbasierte Image Builder Builder-Richtlinien finden Sie unter. Identitätsbasierte Richtlinien von EC2 Image Builder

Ressourcenbasierte Richtlinien von Image Builder

Ressourcenbasierte Richtlinien geben an, welche Aktionen ein bestimmter Prinzipal auf der Image Builder Builder-Ressource ausführen kann und unter welchen Bedingungen. Image Builder unterstützt ressourcenbasierte Berechtigungsrichtlinien für Komponenten, Bilder und Image-Rezepte. Ressourcenbasierte Richtlinien ermöglichen die Erteilung von Nutzungsberechtigungen für andere -Konten pro Ressource. Sie können auch eine ressourcenbasierte Richtlinie verwenden, um einem AWS Dienst den Zugriff auf Ihre Komponenten, Bilder und Image-Rezepte zu ermöglichen.

Informationen zum Anhängen einer ressourcenbasierten Richtlinie an eine Komponente, ein Bild oder ein Image-Rezept finden Sie unter. EC2 Image Builder Builder-Ressourcen teilen

> **Note**
>
> Wenn Sie eine Ressourcenrichtlinie mit Image Builder aktualisieren, wird das Update in der RAM-Konsole angezeigt.

Autorisierung auf Basis von Image Builder Builder-Tags

Sie können Tags an Image Builder-Ressourcen anhängen oder Tags in einer Anfrage an Image Builder übergeben. Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `imagebuilder:ResourceTag/key-name`, `aws:RequestTag/key-name`, oder Bedingung `aws:TagKeys` verwenden. Weitere Informationen zum Taggen von Image Builder Builder-Ressourcen finden Sie unterKennzeichnen Sie eine Ressource (AWS CLI).

Image Builder IAM-Rollen

Eine IAM-Rolle ist eine Entität innerhalb von Ihnen AWS-Konto , die über bestimmte Berechtigungen verfügt.

Temporäre Anmeldeinformationen mit Image Builder verwenden

Sie können temporäre Anmeldeinformationen verwenden, um sich über einen Verbund anzumelden, eine IAM-Rolle anzunehmen oder eine kontenübergreifende Rolle anzunehmen. Sie erhalten temporäre Sicherheitsanmeldedaten, indem Sie AWS STS API-Operationen wie AssumeRoleoder aufrufen GetFederationToken.

Service-verknüpfte Rollen

Mit Diensten verknüpfte Rollen ermöglichen AWS-Services den Zugriff auf Ressourcen in anderen Diensten, um eine Aktion in Ihrem Namen auszuführen. Serviceverknüpfte Rollen werden in Ihrem IAM-Konto angezeigt und gehören zum Service. Ein Benutzer mit Administratorzugriff kann die Berechtigungen für dienstbezogene Rollen einsehen, aber nicht bearbeiten.

Image Builder unterstützt serviceverknüpfte Rollen. Informationen zum Erstellen oder Verwalten von dienstverknüpften Image Builder Builder-Rollen finden Sie unterVerwenden von serviceverknüpften Rollen für EC2 Image Builder.

Servicerollen

Dieses Feature ermöglicht einem Service das Annehmen einer Servicerolle in Ihrem Namen. Diese Rolle gewährt dem Service Zugriff auf Ressourcen in anderen Diensten, um eine Aktion in Ihrem Namen auszuführen. Servicerollen werden in Ihrem IAM-Konto angezeigt und gehören zum Konto. Das bedeutet, dass ein Benutzer mit Administratorzugriff die Berechtigungen für diese Rolle ändern kann. Dies kann jedoch die Funktionalität des Dienstes beeinträchtigen.

Identitätsbasierte Richtlinien von EC2 Image Builder

Themen

Bewährte Methoden für identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand Image Builder Builder-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder diese löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Erste Schritte mit AWS verwalteten Richtlinien und Umstellung auf Berechtigungen mit den geringsten Rechten — Um Ihren Benutzern und Workloads zunächst Berechtigungen zu gewähren, verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle 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 Image Builder Builder-Konsole

Um auf die EC2 Image Builder Builder-Konsole zugreifen zu können, benötigen Sie einen Mindestsatz an Berechtigungen. Mit diesen Berechtigungen können Sie Details zu den Image Builder Builder-Ressourcen in Ihrem auflisten und anzeigen AWS-Konto. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (IAM-Benutzer oder -Rollen) mit dieser Richtlinie.

Um sicherzustellen, dass Ihre IAM-Entitäten die Image Builder Builder-Konsole verwenden können, müssen Sie ihnen eine der folgenden AWS verwalteten Richtlinien zuordnen:

- AWSImageBuilderReadOnlyAccess-Richtlinie
- AWSImageBuilderFullAccess-Richtlinie

Weitere Informationen zu verwalteten Richtlinien von Image Builder finden Sie unterVerwenden verwalteter Richtlinien für EC2 Image Builder.

> **Important**
>
> Die AWSImageBuilderFullAccessRichtlinie ist erforderlich, um die mit dem Service verknüpfte Image Builder Builder-Rolle zu erstellen. Wenn Sie diese Richtlinie an eine IAM-Entität

anhängen, müssen Sie auch die folgende benutzerdefinierte Richtlinie anhängen und die Ressourcen angeben, die Sie verwenden möchten und die nicht `imagebuilder` im Ressourcennamen enthalten sind:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "sns topic arn"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetInstanceProfile"
            ],
            "Resource": "instance profile role arn"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "instance profile role arn",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": "bucket arn"
        }
    ]
}
```

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 Sie ausführen möchten.

Ressourcenbasierte Richtlinien für EC2 Image Builder

Informationen zum Erstellen einer Komponente finden Sie unter. Komponenten mit Image Builder verwalten

Beschränken des Zugriffs auf Image Builder Builder-Komponenten auf bestimmte IP-Adressen

Im folgenden Beispiel wird jedem Benutzer die Berechtigung erteilt, alle Image Builder Builder-Operationen an Komponenten auszuführen. Die Anfrage muss jedoch aus dem in der Bedingung angegebenen IP-Adressbereich stammen.

Die Bedingung in dieser Anweisung identifiziert den Bereich 54.240.143.* als zulässigen Bereich für Internetprotokoll 4-Adressen (IPv4-Adressen), mit einer Ausnahme: 54.240.143.188.

Der `Condition` Block verwendet die `NotIpAddress` Bedingungen `IpAddress` und und den `aws:SourceIp` Bedingungsschlüssel, bei dem es sich um einen AWS-weiten Bedingungsschlüssel handelt. Weitere Informationen zu diesen Bedingungsschlüsseln finden Sie unter Bedingungen in einer Richtlinie angeben. Die `aws:sourceIp` IPv4-Werte verwenden die CIDR-Standardnotation. Weitere Informationen finden Sie unter IP-Adressen-Bedingungsoperatoren im IAM-Benutzerhandbuch.

```
{
  "Version": "2012-10-17",
  "Id": "IBPolicyId1",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "imagebuilder.GetComponent:*",
      "Resource": "arn:aws:imagebuilder:::examplecomponent/*",
      "Condition": {
         "IpAddress": {"aws:SourceIp": "54.240.143.0/24"},
         "NotIpAddress": {"aws:SourceIp": "54.240.143.188/32"}
      }
```

```
    }
  ]
}
```

Verwenden verwalteter Richtlinien für EC2 Image Builder

Eine AWS verwaltete Richtlinie ist eine eigenständige Richtlinie, die von erstellt und verwaltet wird AWS. AWS Verwaltete Richtlinien dienen dazu, Berechtigungen für viele gängige Anwendungsfälle bereitzustellen, sodass Sie damit beginnen können, Benutzern, Gruppen und Rollen Berechtigungen zuzuweisen.

Beachten Sie, dass AWS verwaltete Richtlinien für Ihre speziellen Anwendungsfälle möglicherweise keine Berechtigungen mit den geringsten Rechten gewähren, da sie allen AWS Kunden zur Verfügung stehen. Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie kundenverwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind.

Sie können die in AWS verwalteten Richtlinien definierten Berechtigungen nicht ändern. Wenn die in einer AWS verwalteten Richtlinie definierten Berechtigungen AWS aktualisiert werden, wirkt sich das Update auf alle Prinzidentitäten (Benutzer, Gruppen und Rollen) aus, denen die Richtlinie zugeordnet ist. AWS aktualisiert eine AWS verwaltete Richtlinie höchstwahrscheinlich, wenn eine neue Richtlinie eingeführt AWS-Service wird oder neue API-Operationen für bestehende Dienste verfügbar werden.

Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien im IAM-Benutzerhandbuch.

AWSImageBuilderFullAccess-Richtlinie

Die AWSImageBuilderFullAccessRichtlinie gewährt vollen Zugriff auf Image Builder Builder-Ressourcen für die Rolle, der sie zugeordnet ist, sodass die Rolle Image Builder Builder-Ressourcen auflisten, beschreiben, erstellen, aktualisieren und löschen kann. Die Richtlinie gewährt außerdem gezielte Berechtigungen für verwandte Benutzer, AWS-Services die beispielsweise zur Überprüfung von Ressourcen oder zur Anzeige der aktuellen Ressourcen für das Konto in der erforderlich sind AWS Management Console.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen:

- Image Builder — Administratorzugriff wird gewährt, sodass die Rolle Image Builder Builder-Ressourcen auflisten, beschreiben, erstellen, aktualisieren und löschen kann.

- Amazon EC2 — Zugriff wird für Amazon EC2 gewährt. Beschreiben Sie Aktionen, die erforderlich sind, um das Vorhandensein von Ressourcen zu überprüfen oder Listen der Ressourcen abzurufen, die zu dem Konto gehören.
- IAM — Zugriff wird gewährt, um Instanzprofile abzurufen und zu verwenden, deren Name „imagebuilder“ enthält, um das Vorhandensein der mit dem Service verknüpften Image Builder Builder-Rolle über die `iam:GetRole` API-Aktion zu überprüfen und um die mit dem Service verknüpfte Image Builder Builder-Rolle zu erstellen.
- License Manager — Zugriff wird gewährt, um Lizenzkonfigurationen oder Lizenzen für eine Ressource aufzulisten.
- Amazon S3 — Zugriff wird auf Listen-Buckets gewährt, die zum Konto gehören, sowie auf Image Builder Builder-Buckets, deren Namen „imagebuilder“ enthalten.
- Amazon SNS — Amazon SNS werden Schreibberechtigungen erteilt, um die Inhaberschaft von Themen zu überprüfen, die „Imagebuilder“ enthalten.

Beispiel für eine Richtlinie

Im Folgenden finden Sie ein Beispiel für die AWSImageBuilderFullAccess Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "imagebuilder:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "sns:ListTopics"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
```

```
            "Resource": "arn:aws:sns:*:*:*imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "license-manager:ListLicenseConfigurations",
                "license-manager:ListLicenseSpecificationsForResource"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetInstanceProfile"
            ],
            "Resource": "arn:aws:iam::*:instance-profile/*imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListInstanceProfiles",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::*:instance-profile/*imagebuilder*",
                "arn:aws:iam::*:role/*imagebuilder*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:GetBucketLocation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": "arn:aws:s3::*:*imagebuilder*"
        },
        {
            "Action": "iam:CreateServiceLinkedRole",
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "imagebuilder.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "ec2:DescribeSnapshots",
                "ec2:DescribeVpcs",
                "ec2:DescribeRegions",
                "ec2:DescribeVolumes",
                "ec2:DescribeSubnets",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeLaunchTemplates"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

AWSImageBuilderReadOnlyAccess-Richtlinie

Die AWSImageBuilderReadOnlyAccessRichtlinie bietet schreibgeschützten Zugriff auf alle Image Builder Builder-Ressourcen. Über die `iam:GetRole` API-Aktion werden Berechtigungen erteilt, um zu überprüfen, ob die mit dem Image Builder Builder-Dienst verknüpfte Rolle vorhanden ist.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen:

- Image Builder — Zugriff wird für schreibgeschützten Zugriff auf Image Builder Builder-Ressourcen gewährt.
- IAM — Zugriff wird gewährt, um das Vorhandensein der mit dem Service verknüpften Image Builder Builder-Rolle über die `iam:GetRole` API-Aktion zu überprüfen.

Beispiel für eine Richtlinie

Im Folgenden finden Sie ein Beispiel für die AWSImageBuilderReadOnlyAccess Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "imagebuilder:Get*",
                "imagebuilder:List*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder"
        }
    ]
```

```
}
```

AWSServiceRoleForImageBuilder-Richtlinie

Die AWSServiceRoleForImageBuilderRichtlinie ermöglicht es Image Builder, in AWS-Services Ihrem Namen anzurufen.

Details zu Berechtigungen

Diese Richtlinie wird der mit dem Dienst verknüpften Image Builder Builder-Rolle zugewiesen, wenn die Rolle über Systems Manager erstellt wird. Informationen zu bestimmten erteilten Berechtigungen finden Sie im Richtlinienbeispiel in diesem Abschnitt. Weitere Informationen zur dienstverknüpften Image Builder Builder-Rolle finden Sie unterVerwenden von serviceverknüpften Rollen für EC2 Image Builder.

Die Richtlinie umfasst die folgenden Berechtigungen:

- CloudWatch Protokolle — Zugriff auf das Erstellen und Hochladen von CloudWatch Protokollen in jede Protokollgruppe, deren Name mit beginnt, wird gewährt`/aws/imagebuilder/`.
- Amazon EC2 — Image Builder erhält Zugriff, um Images zu erstellen und EC2-Instances in Ihrem Konto zu starten, wobei nach Bedarf zugehörige Snapshots, Volumes, Netzwerkschnittstellen, Subnetze, Sicherheitsgruppen, Lizenzkonfigurationen und Schlüsselpaare verwendet werden, sofern das Image, die Instance und die Volumes, die erstellt oder verwendet werden, mit oder gekennzeichnet sind. `CreatedBy: EC2 Image Builder` `CreatedBy: EC2 Fast Launch`

  Image Builder kann Informationen über Amazon EC2 EC2-Images, Instance-Attribute, Instance-Status, die für Ihr Konto verfügbaren Instance-Typen, Startvorlagen, Subnetze, Hosts und Tags auf Ihren Amazon EC2 EC2-Ressourcen abrufen.

  Image Builder kann die Bildeinstellungen aktualisieren, um das schnellere Starten von Windows-Instanzen in Ihrem Konto zu aktivieren oder zu deaktivieren, mit denen das Bild gekennzeichnet ist`CreatedBy: EC2 Image Builder`.

  Darüber hinaus kann Image Builder Instances, die in Ihrem Konto ausgeführt werden, starten, stoppen und beenden, Amazon EBS-Snapshots teilen, Images erstellen und aktualisieren und Vorlagen starten, bestehende Images deregistrieren, Tags hinzufügen und Images zwischen Konten replizieren, für die Sie über die Richtlinie Berechtigungen erteilt haben. Ec2ImageBuilderCrossAccountDistributionAccess Image Builder Builder-Tagging ist für all diese Aktionen erforderlich, wie zuvor beschrieben.

- Amazon ECR — Image Builder erhält Zugriff, um bei Bedarf ein Repository für Container-Image-Schwachstellenscans zu erstellen und die erstellten Ressourcen zu taggen, um den Umfang seiner Operationen einzuschränken. Image Builder erhält außerdem Zugriff auf das Löschen der Container-Images, die es für die Scans erstellt hat, nachdem es Schnappschüsse der Sicherheitsanfälligkeiten erstellt hat.
- EventBridge— Image Builder erhält Zugriff zum Erstellen und Verwalten von EventBridge Regeln.
- IAM — Image Builder erhält Zugriff, um jede Rolle in Ihrem Konto an Amazon EC2 und VM Import/Export weiterzugeben.
- Amazon Inspector — Image Builder erhält Zugriff, um festzustellen, wann Amazon Inspector Build-Instance-Scans abschließt, und um Ergebnisse für Images zu sammeln, die so konfiguriert sind, dass dies zulässig ist.
- AWS KMS— Amazon EBS wird Zugriff gewährt, um Amazon EBS-Volumes zu verschlüsseln, zu entschlüsseln oder erneut zu verschlüsseln. Dies ist wichtig, um sicherzustellen, dass verschlüsselte Volumes funktionieren, wenn Image Builder ein Image erstellt.
- License Manager — Image Builder erhält Zugriff, um die License Manager Manager-Spezifikationen über zu aktualisieren`license-manager:UpdateLicenseSpecificationsForResource`.
- Amazon SNS — Schreibberechtigungen werden für jedes Amazon SNS SNS-Thema in Ihrem Konto gewährt.
- Systems Manager — Image Builder erhält Zugriff, um Systems Manager Manager-Befehle und deren Aufrufe sowie Inventareinträge aufzulisten, Instanzinformationen und den Status der Automatisierungsausführung zu beschreiben und Details zum Befehlsaufruf abzurufen. Image Builder kann auch Automatisierungssignale senden und Automatisierungsausführungen für jede Ressource in Ihrem Konto beenden.

  Image Builder kann Ausführungsbefehle für jede Instanz ausgeben, die `"CreatedBy": "EC2 Image Builder"` für die folgenden Skriptdateien gekennzeichnet ist:`AWS-RunPowerShellScript`,`AWS-RunShellScript`, oder. `AWSEC2-RunSysprep` Image Builder kann in Ihrem Konto eine Systems Manager Manager-Automatisierungsausführung für Automatisierungsdokumente starten, deren Name mit beginnt`ImageBuilder`.

  Image Builder ist auch in der Lage, State Manager-Zuordnungen für jede Instanz in Ihrem Konto zu erstellen oder zu löschen, sofern das Zuordnungsdokument vorhanden ist`AWS-GatherSoftwareInventory`, und die mit dem Service verknüpfte Systems Manager Manager-Rolle in Ihrem Konto zu erstellen.

- AWS STS— Image Builder hat Zugriff darauf, EC2ImageBuilderDistributionCrossAccountRolevon Ihrem Konto benannte Rollen auf jedes Konto zu übertragen, sofern die Vertrauensrichtlinie für die Rolle dies zulässt. Dies wird für die kontoübergreifende Verteilung von Images verwendet.

Beispiel für eine Richtlinie

Im Folgenden finden Sie ein Beispiel für die AWSServiceRoleForImageBuilder Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "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:*:*:key-pair/*",
                "arn:aws:ec2:*:*:launch-template/*",
                "arn:aws:license-manager:*:*:license-configuration:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*:*:instance/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": [
                        "EC2 Image Builder",
                        "EC2 Fast Launch"
                    ]
                }
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn",
                        "vmie.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StopInstances",
                "ec2:StartInstances",
                "ec2:TerminateInstances"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CopyImage",
                "ec2:CreateImage",
                "ec2:CreateLaunchTemplate",
                "ec2:DeregisterImage",
                "ec2:DescribeImages",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeSubnets",
```

```
                "ec2:DescribeTags",
                "ec2:ModifyImageAttribute",
                "ec2:DescribeImportImageTasks",
                "ec2:DescribeExportImageTasks",
                "ec2:DescribeSnapshots",
                "ec2:DescribeHosts"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ModifySnapshotAttribute"
            ],
            "Resource": "arn:aws:ec2:*::snapshot/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": [
                        "RunInstances",
                        "CreateImage"
                    ],
                    "aws:RequestTag/CreatedBy": [
                        "EC2 Image Builder",
                        "EC2 Fast Launch"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
```

```
            ],
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*:*:export-image-task/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*:*:launch-template/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": [
                        "EC2 Image Builder",
                        "EC2 Fast Launch"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "license-manager:UpdateLicenseSpecificationsForResource"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:ListCommands",
                "ssm:ListCommandInvocations",
                "ssm:AddTagsToResource",
```

```
                "ssm:DescribeInstanceInformation",
                "ssm:GetAutomationExecution",
                "ssm:StopAutomationExecution",
                "ssm:ListInventoryEntries",
                "ssm:SendAutomationSignal",
                "ssm:DescribeInstanceAssociationsStatus",
                "ssm:DescribeAssociationExecutions",
                "ssm:GetCommandInvocation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ssm:SendCommand",
            "Resource": [
                "arn:aws:ssm:*:*:document/AWS-RunPowerShellScript",
                "arn:aws:ssm:*:*:document/AWS-RunShellScript",
                "arn:aws:ssm:*:*:document/AWSEC2-RunSysprep",
                "arn:aws:s3:::*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:instance/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ssm:resourceTag/CreatedBy": [
                        "EC2 Image Builder"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ssm:StartAutomationExecution",
            "Resource": "arn:aws:ssm:*:*:automation-definition/ImageBuilder*"
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "ssm:CreateAssociation",
                "ssm:DeleteAssociation"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:document/AWS-GatherSoftwareInventory",
                "arn:aws:ssm:*:*:association/*",
                "arn:aws:ec2:*:*:instance/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncryptFrom",
                "kms:ReEncryptTo",
                "kms:GenerateDataKeyWithoutPlaintext"
            ],
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "kms:EncryptionContextKeys": [
                        "aws:ebs:id"
                    ]
                },
                "StringLike": {
                    "kms:ViaService": [
                        "ec2.*.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:DescribeKey"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "kms:ViaService": [
                        "ec2.*.amazonaws.com"
                    ]
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "kms:CreateGrant",
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": true
                },
                "StringLike": {
                    "kms:ViaService": [
                        "ec2.*.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::*:role/
EC2ImageBuilderDistributionCrossAccountRole"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/imagebuilder/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateLaunchTemplateVersion",
                "ec2:DescribeLaunchTemplates",
                "ec2:ModifyLaunchTemplate",
                "ec2:DescribeLaunchTemplateVersions"
            ],
            "Resource": "*"
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "ec2:ExportImage"
            ],
            "Resource": "arn:aws:ec2:*::image/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ExportImage"
            ],
            "Resource": "arn:aws:ec2:*:*:export-image-task/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CancelExportTask"
            ],
            "Resource": "arn:aws:ec2:*:*:export-image-task/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "ssm.amazonaws.com",
                        "ec2fastlaunch.amazonaws.com"
                    ]
                }
            }
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "ec2:EnableFastLaunch"
            ],
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*:*:launch-template/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "inspector2:ListCoverage",
                "inspector2:ListFindings"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:CreateRepository"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:TagResource"
            ],
            "Resource": "arn:aws:ecr:*:*:repository/image-builder-*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "EC2 Image Builder"
                }
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchDeleteImage"
            ],
            "Resource": "arn:aws:ecr:*:*:repository/image-builder-*",
            "Condition": {
                "StringEquals": {
                    "ecr:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "events:DeleteRule",
                "events:DescribeRule",
                "events:PutRule",
                "events:PutTargets",
                "events:RemoveTargets"
            ],
            "Resource": [
                "arn:aws:events:*:*:rule/ImageBuilder-*"
            ]
        }
    ]
}
```

Ec2ImageBuilderCrossAccountDistributionAccess-Richtlinie

Die Ec2ImageBuilderCrossAccountDistributionAccessRichtlinie gewährt Image Builder die Erlaubnis, Bilder auf Konten in Zielregionen zu verteilen. Darüber hinaus kann Image Builder jedes Amazon EC2 EC2-Image im Konto beschreiben, kopieren und Tags darauf anwenden. Die Richtlinie gewährt auch die Möglichkeit, AMI-Berechtigungen über die `ec2:ModifyImageAttribute` API-Aktion zu ändern.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen:

- Amazon EC2 — Amazon EC2 erhält Zugriff, um Attribute für ein Bild zu beschreiben, zu kopieren und zu ändern und Tags für alle Amazon EC2 EC2-Images im Konto zu erstellen.

Beispiel für eine Richtlinie

Das Folgende ist ein Beispiel für die Ec2ImageBuilderCrossAccountDistributionAccess Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*::image/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "ec2:CopyImage",
                "ec2:ModifyImageAttribute"
            ],
            "Resource": "*"
        }
    ]
}
```

EC2ImageBuilderLifecycleExecutionPolicy-Richtlinie

Die EC2ImageBuilderLifecycleExecutionPolicyRichtlinie gewährt Image Builder die Erlaubnis, Aktionen wie das Verwerfen, Deaktivieren oder Löschen von Image Builder Builder-Image-Ressourcen und den ihnen zugrunde liegenden Ressourcen (AMIs, Snapshots) auszuführen, um automatisierte Regeln für Image-Lifecycle-Management-Aufgaben zu unterstützen.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen:

- Amazon EC2 — Amazon EC2 wird Zugriff gewährt, um die folgenden Aktionen für Amazon Machine Images (AMIs) in dem Konto auszuführen, das mit gekennzeichnet ist. `CreatedBy: EC2 Image Builder`
  - Aktiviert und deaktiviert ein AMI.
  - Aktiviert und deaktiviert die Image-Veralterung.
  - Beschreiben Sie ein AMI und melden Sie es ab.

  - Beschreiben und ändern Sie AMI-Image-Attribute.
  - Löschen Sie Volume-Snapshots, die dem AMI zugeordnet sind.
  - Ruft Tags für eine Ressource ab.
  - Hinzufügen oder Entfernen veralteter Tags aus einem AMI.
- Amazon ECR — Amazon ECR wird Zugriff gewährt, um die folgenden Batch-Aktionen an ECR-Repositorys mit dem Tag durchzuführen. `LifecycleExecutionAccess: EC2 Image Builder` Batch-Aktionen unterstützen automatisierte Lebenszyklusregeln für Container-Images.
  - `ecr:BatchGetImage`
  - `ecr:BatchDeleteImage`

  Der Zugriff auf Repository-Ebene für ECR-Repositorys, die mit gekennzeichnet sind, wird auf Repository-Ebene gewährt. `LifecycleExecutionAccess: EC2 Image Builder`
- AWS Ressourcengruppen — Image Builder erhält Zugriff, um Ressourcen auf der Grundlage von Tags abzurufen.
- EC2 Image Builder — Image Builder erhält Zugriff zum Löschen von Image Builder-Image-Ressourcen.

Beispiel für eine Richtlinie

Im Folgenden finden Sie ein Beispiel für die Richtlinie. EC2ImageBuilderLifecycleExecutionPolicy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Ec2ImagePermission",
            "Effect": "Allow",
            "Action": [
                "ec2:EnableImage",
                "ec2:DeregisterImage",
                "ec2:EnableImageDeprecation",
                "ec2:DescribeImageAttribute",
                "ec2:DisableImage",
                "ec2:DisableImageDeprecation"
            ],
            "Resource": "arn:aws:ec2:*::image/*",
            "Condition": {
                "StringEquals": {
```

```
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Sid": "EC2DeleteSnapshotPermission",
            "Effect": "Allow",
            "Action": "ec2:DeleteSnapshot",
            "Resource": "arn:aws:ec2:*::snapshot/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Sid": "EC2TagsPermission",
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteTags",
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*::image/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/DeprecatedBy": "EC2 Image Builder",
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "DeprecatedBy"
                }
            }
        },
        {
            "Sid": "ECRImagePermission",
            "Effect": "Allow",
            "Action": [
                "ecr:BatchGetImage",
                "ecr:BatchDeleteImage"
            ],
            "Resource": "arn:aws:ecr:*:*:repository/*",
```

```
            "Condition": {
                "StringEquals": {
                    "ecr:ResourceTag/LifecycleExecutionAccess": "EC2 Image Builder"
                }
            }
        },
        {
            "Sid": "ImageBuilderEC2TagServicePermission",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "tag:GetResources",
                "imagebuilder:DeleteImage"
            ],
            "Resource": "*"
        }
    ]
}
```

EC2InstanceProfileForImageBuilder-Richtlinie

Die EC2InstanceProfileForImageBuilderRichtlinie gewährt die Mindestberechtigungen, die für die Verwendung einer EC2-Instance mit Image Builder erforderlich sind. Dies schließt nicht die Berechtigungen ein, die für die Verwendung des Systems Manager Agent erforderlich sind.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen:

- CloudWatch Protokolle — Es wird Zugriff gewährt, um CloudWatch Protokolle zu erstellen und in jede Protokollgruppe hochzuladen, deren Name mit beginnt`/aws/imagebuilder/`.
- Image Builder — Zugriff wird gewährt, um jede Image Builder Builder-Komponente abzurufen.
- AWS KMS— Zugriff wird gewährt, um eine Image Builder Builder-Komponente zu entschlüsseln, wenn sie über AWS KMS verschlüsselt wurde.
- Amazon S3 — Zugriff wird gewährt, um Objekte abzurufen, die in einem Amazon S3 S3-Bucket gespeichert sind, dessen Name mit beginnt`ec2imagebuilder-`.

Beispiel für eine Richtlinie

Das Folgende ist ein Beispiel für die EC2InstanceProfileForImageBuilder Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "imagebuilder:GetComponent"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "kms:EncryptionContextKeys": "aws:imagebuilder:arn",
                    "aws:CalledVia": [
                        "imagebuilder.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::ec2imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/imagebuilder/*"
        }
    ]
}
```

EC2InstanceProfileForImageBuilderECRContainerBuilds-Richtlinie

Die EC2InstanceProfileForImageBuilderECRContainerBuildsRichtlinie gewährt die Mindestberechtigungen, die für eine EC2-Instance erforderlich sind, wenn mit Image Builder Docker-Images erstellt und die Images anschließend in einem Amazon ECR-Container-Repository registriert und gespeichert werden. Dies schließt nicht die Berechtigungen ein, die für die Verwendung des Systems Manager Agent erforderlich sind.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen:

- CloudWatch Protokolle — Es wird Zugriff gewährt, um CloudWatch Protokolle zu erstellen und in jede Protokollgruppe hochzuladen, deren Name mit beginnt`/aws/imagebuilder/`.
- Amazon ECR — Amazon ECR wird Zugriff gewährt, um ein Container-Image abzurufen, zu registrieren und zu speichern und ein Autorisierungstoken zu erhalten.
- Image Builder — Zugriff wird gewährt, um eine Image Builder Builder-Komponente oder ein Container-Rezept abzurufen.
- AWS KMS— Zugriff wird gewährt, um eine Image Builder Builder-Komponente oder ein Container-Rezept zu entschlüsseln, sofern es über AWS KMS verschlüsselt wurde.
- Amazon S3 — Zugriff wird gewährt, um Objekte abzurufen, die in einem Amazon S3 S3-Bucket gespeichert sind, dessen Name mit beginnt`ec2imagebuilder-`.

Beispiel für eine Richtlinie

Das Folgende ist ein Beispiel für die Richtlinie. EC2InstanceProfileForImageBuilderECRContainerBuilds

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "imagebuilder:GetComponent",
                "imagebuilder:GetContainerRecipe",
                "ecr:GetAuthorizationToken",
                "ecr:BatchGetImage",
                "ecr:InitiateLayerUpload",
```

```
                "ecr:UploadLayerPart",
                "ecr:CompleteLayerUpload",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:PutImage"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "kms:EncryptionContextKeys": "aws:imagebuilder:arn",
                    "aws:CalledVia": [
                        "imagebuilder.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::ec2imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/imagebuilder/*"
        }
    ]
}
```

Image Builder Builder-Updates für AWS verwaltete Richtlinien

Dieser Abschnitt enthält Informationen zu Aktualisierungen der AWS verwalteten Richtlinien für Image Builder, seit dieser Dienst begonnen hat, diese Änderungen zu verfolgen. Abonnieren Sie den RSS-Feed auf der Seite Image Builder Builder-Dokumentenverlauf, um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhalten.

Änderung	Beschreibung	Datum
EC2ImageBuilderLifecycleExecutionPolicy – Neue Richtlinie.	Image Builder hat die neue `EC2ImageBuilderLifecycleExecutionPolicy` Richtlinie hinzugefügt, die Berechtigungen für das Image-Lebenszyklusmanagement enthält.	17. November 2023
AWSServiceRoleForImageBuilder – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat die folgenden Änderungen an der Servicerolle vorgenommen, um macOS-Support bereitzustellen. • ec2 hinzugefügt: DescribeHosts aktiviert Image Builder, um die hostId abzufragen, um festzustellen, wann sie sich in einem gültigen Zustand befindet, um eine Instance zu starten. • Die API-Aktion ssm:GetCommandInvocation, wurde hinzugefügt, um die Methode zu verbessern, die Image Builder verwendet,	28. August 2023

Änderung	Beschreibung	Datum
	um Details zum Befehlsaufruf abzurufen.	

Änderung	Beschreibung	Datum
AWSServiceRoleForImageBuilder – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat die folgenden Änderungen an der Servicerolle vorgenommen, damit Image Builder Builder-Workflows Schwachstellen sowohl für AMI- als auch für ECR-Container-Image-Builds sammeln können. Die neuen Berechtigungen unterstützen die CVE-Erkennungs- und Berichtsfunktion. • inspector2: ListCoverage und inspector2: hinzugefügt, damit Image Builder feststellen kannListFindings , wann Amazon Inspector Testinstance-Scans abschließt, und Ergebnisse für Bilder sammeln kann, die so konfiguriert sind, dass dies zulässig ist. • ecr: hinzugefügtCreateRepository, mit der Anforderung, dass Image Builder das Repository mit `CreatedBy : EC2 Image Builder` (tag-on-create) kennzeichnen muss. Außerdem wurde ecr: TagResource (erforderlich für tag-on-create) mit derselben CreatedBy Tag-Einschränkung und einer	30. März 2023

Änderung	Beschreibung	Datum
	zusätzlichen Einschränkung hinzugefügt, bei der der Repository-Name beginnen muss. `image-builder-*` Die Namensbeschränkung verhindert die Eskalation von Rechten und verhinder t Änderungen an Repositor ys, die Image Builder nicht erstellt hat. • ecr: BatchDeleteImage für ECR-Repositorys hinzugefü gt, die mit gekennzeichnet sind. `CreatedBy: EC2 Image Builder` Diese Berechtigung erfordert, dass der Name des Repositorys mit beginnt. `image-bui lder-*` • Es wurden Ereignisb erechtigungen für Image Builder hinzugefügt, um von Amazon EventBrid ge verwaltete Regeln zu erstellen und zu verwalten, die `ImageBuilder-*` im Namen enthalten sind.	

Änderung	Beschreibung	Datum
AWSServiceRoleForImageBuilder – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat die folgenden Änderungen an der Servicerolle vorgenommen: • License Manager Manager-Lizenzen wurden als Ressource für den ec2: RunInstance -Aufruf hinzugefügt, damit Kunden Basis-Image-AMIs verwenden können, die mit einer Lizenzkonfiguration verknüpft sind.	22. März 2022
AWSServiceRoleForImageBuilder – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat die folgenden Änderungen an der Servicerolle vorgenommen: • Es wurden Berechtigungen für EnableFastLaunch EC2-API-Aktionen hinzugefügt, um das schnellere Starten für Windows-Instances zu aktivieren und zu deaktivieren. • Der Geltungsbereich für ec2 wurde stärker eingeschränkt: Bedingungen für CreateTags Aktionen und Ressourcen-Tags.	21. Februar 2022

Änderung	Beschreibung	Datum
AWSServiceRoleForImageBuilder – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat die folgenden Änderungen an der Servicerolle vorgenommen: • Es wurden Berechtigungen hinzugefügt, um den VMIE-Dienst aufzurufen, um eine VM zu importieren und daraus ein Basis-AMI zu erstellen. • Eingeschränkter Anwendungsbereich für ec2: Bedingungen für CreateTags Aktionen und Ressourcen-Tags.	20. November 2021
AWSServiceRoleForImageBuilder – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat neue Berechtigungen hinzugefügt, um Probleme zu beheben, bei denen mehrere Inventarzuordnungen dazu führen, dass der Image-Build hängen bleibt.	11. August 2021

Änderung	Beschreibung	Datum
AWSImageBuilderFullAccess – Aktualisierung auf eine bestehende Richtlinie	Image Builder hat die folgenden Änderungen an der Vollzugriffsrolle vorgenommen: • Es wurden Berechtigungen zum Zulassen hinzugefügt `ec2:DescribeInstanceTypeOfferings` . • Es wurden Aufrufberechtigungen hinzugefügt `ec2:DescribeInstanceTypeOfferings` , damit die Image Builder Builder-Konsole die Instanztypen, die im Konto verfügbar sind, genau wiedergeben kann.	13. April 2021
Image Builder hat mit der Nachverfolgung von Änderungen begonnen	Image Builder begann, Änderungen für seine AWS verwalteten Richtlinien nachzuverfolgen.	02. April 2021

Verwenden von serviceverknüpften Rollen für EC2 Image Builder

EC2 Image Builder verwendet AWS Identity and Access Management (IAM) serviceverknüpfte Rollen. Eine serviceverknüpfte Rolle ist ein einzigartiger Typ von IAM-Rolle, die direkt mit Image Builder verknüpft ist. Dienstbezogene Rollen sind von Image Builder vordefiniert und enthalten alle Berechtigungen, die der Dienst benötigt, um andere in AWS-Services Ihrem Namen aufzurufen.

Eine dienstbezogene Rolle macht die Einrichtung von Image Builder effizienter, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Image Builder definiert die

Berechtigungen seiner dienstbezogenen Rollen, und sofern nicht anders definiert, kann nur Image Builder seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Die Berechtigungsrichtlinie kann an keine andere IAM-Entität angefügt werden.

Informationen zu anderen Diensten, die dienstverknüpfte Rollen unterstützen, finden Sie unter That Work AWS-Services with IAM. Suchen Sie nach den Diensten, für die in der Spalte Serviceverknüpfte Rolle der Wert Ja steht. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Dienstbezogene Rollenberechtigungen für Image Builder

Image Builder verwendet die AWSServiceRoleForImageBuilderdienstverknüpfte Rolle, um EC2 Image Builder den Zugriff auf AWS Ressourcen in Ihrem Namen zu ermöglichen. Die serviceverknüpfte Rolle vertraut darauf, dass der Service imagebuilder.amazonaws.com die Rolle übernimmt.

Sie müssen diese serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie Ihr erstes Image Builder-Image in der AWS Management Console, der AWS CLI oder der AWS API erstellen, erstellt Image Builder die serviceverknüpfte Rolle für Sie.

Mit den folgenden Aktionen wird ein neues Image erstellt:

- Führen Sie den Pipeline-Assistenten in der Image Builder Builder-Konsole aus, um ein benutzerdefiniertes Image zu erstellen.
- Verwenden Sie eine der folgenden API-Aktionen oder den entsprechenden AWS CLI Befehl:
  - Die CreateImage API-Aktion (create-imagein der AWS CLI).
  - Die ImportVmImage API-Aktion (import-vm-imagein der AWS CLI).
  - Die StartImagePipelineExecution API-Aktion (start-image-pipeline-executionin der AWS CLI).

> **Important**
>
> Wenn die mit dem Dienst verknüpfte Rolle aus Ihrem Konto gelöscht wird, können Sie sie mit demselben Verfahren erneut erstellen. Wenn Sie Ihre erste EC2 Image Builder Builder-Ressource erstellen, erstellt Image Builder die serviceverknüpfte Rolle erneut für Sie.

Die Berechtigungen für die finden Sie auf der AWSServiceRoleForImageBuilderSeite. AWSServiceRoleForImageBuilder-Richtlinie Weitere Informationen zur Konfiguration von Berechtigungen für eine dienstverknüpfte Rolle finden Sie unter Berechtigungen für dienstverknüpfte Rollen im IAM-Benutzerhandbuch.

Entfernen einer mit dem Image Builder Builder-Dienst verknüpften Rolle aus Ihrem Konto

Sie können die IAM-Konsole, die oder die AWS API verwenden AWS CLI, um die serviceverknüpfte Rolle für Image Builder manuell aus Ihrem Konto zu entfernen. Bevor Sie dies tun, müssen Sie jedoch sicherstellen, dass keine Image Builder Builder-Ressourcen aktiviert sind, die darauf verweisen.

> **Note**
>
> Wenn der Image Builder Builder-Dienst die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt das Löschen möglicherweise fehl. Wenn dies passiert, warten Sie einige Minuten und versuchen Sie es erneut.

Von der **`AWSServiceRoleForImageBuilder`** Rolle verwendete Image Builder Builder-Ressourcen bereinigen

1. Stellen Sie vor dem Start sicher, dass keine Pipeline-Builds ausgeführt werden. Um einen laufenden Build abzubrechen, verwenden Sie den `cancel-image-creation` Befehl von AWS CLI.

   ```
   aws imagebuilder cancel-image-creation --image-build-version-
   arn arn:aws:imagebuilder:us-east-1:123456789012:image-pipeline/sample-pipeline
   ```

2. Ändern Sie alle Pipeline-Zeitpläne so, dass sie einen manuellen Erstellungsprozess verwenden, oder löschen Sie sie, wenn Sie sie nicht mehr verwenden möchten. Weitere Informationen zum Löschen von Ressourcen finden Sie unterEC2 Image Builder Builder-Ressourcen löschen.

Löschen Sie die dienstverknüpfte Rolle mithilfe von IAM

Sie können die IAM-Konsole, die oder die AWS API verwenden AWS CLI, um die `AWSServiceRoleForImageBuilder` Rolle aus Ihrem Konto zu löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für serviceverknüpfte EC2 Image Builder Builder-Rollen

Image Builder unterstützt die Verwendung von dienstbezogenen Rollen in allen AWS Regionen, in denen der Service verfügbar ist. Eine Liste der unterstützten AWS Regionen finden Sie unterAWS Regionen und Endpunkte.

Problembehandlung bei EC2 Image Builder Builder-Identität und -Zugriff

Themen

- Ich bin nicht berechtigt, eine Aktion in Image Builder auszuführen
- Ich bin nicht berechtigt, iam auszuführen: PassRole
- Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine Image Builder Builder-Ressourcen ermöglichen

Ich bin nicht berechtigt, eine Aktion in Image Builder auszuführen

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht zur Durchführung einer Aktion berechtigt sind, müssen Ihre Richtlinien aktualisiert werden, damit Sie die Aktion durchführen können.

Der folgende Beispielfehler tritt auf, wenn der IAM-Benutzer `mateojackson` versucht, über die Konsole Details zu einer fiktiven *`my-example-widget`*-Ressource anzuzeigen, jedoch nicht über `imagebuilder:`*`GetWidget`*-Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 imagebuilder:GetWidget on resource: my-example-widget
```

In diesem Fall muss die Richtlinie für den Benutzer `mateojackson` aktualisiert werden, damit er mit der `imagebuilder:`*`GetWidget`*-Aktion auf die *`my-example-widget`*-Ressource zugreifen kann.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich bin nicht berechtigt, iam auszuführen: PassRole

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht berechtigt sind, die `iam:PassRole` Aktion auszuführen, müssen Ihre Richtlinien aktualisiert werden, damit Sie eine Rolle an Image Builder übergeben 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 Image Builder 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 von mir den Zugriff AWS-Konto auf meine Image Builder Builder-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 Image Builder diese Funktionen unterstützt, finden Sie unterSo funktioniert EC2 Image Builder 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 für 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.

Konformitätsvalidierung für EC2 Image Builder

EC2 Image Builder fällt nicht in den Geltungsbereich von AWS Compliance-Programmen.

Eine Liste der AWS-Services einzelnen Compliance-Programme finden Sie unter AWS Services in Umfang nach Compliance-Programmen AWS-Services unter . Allgemeine Informationen finden Sie unter AWS Compliance-Programme AWS .

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte in AWS Artifact herunterladen Berichte in AWS Artifact .

Ihre Compliance-Verantwortung bei der Verwendung von Image Builder hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. AWS bietet die folgenden Ressourcen zur Unterstützung bei der Einhaltung von Vorschriften:

- Schnellstartanleitungen für Sicherheit und Compliance – In diesen Bereitstellungsleitfäden werden architektonische Überlegungen erörtert und Schritte für die Bereitstellung von sicherheits- und konformitätsorientierten Basisumgebungen auf AWS angegeben.
- AWS Ressourcen zur AWS von Vorschriften — Diese Sammlung von Arbeitsmappen und Leitfäden kann auf Ihre Branche und Ihren Standort zutreffen.
- Bewertung 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— Dieser AWS Service bietet einen umfassenden Überblick über Ihren Sicherheitsstatus, sodass Sie überprüfen können AWS , ob Sie die Sicherheitsstandards und Best Practices der Branche einhalten.

Sie können Compliance-Produkte von AWS Marketplace oder Komponenten von AWS Task Orchestrator and Executor (AWSTOE) in Ihre Image Builder Builder-Images integrieren, um sicherzustellen, dass Ihre Images konform sind. Weitere Informationen finden Sie unter Compliance-Produkte für Ihre Image Builder Builder-Images.

Resilienz in EC2 Image Builder

Die AWS globale Infrastruktur basiert auf AWS Regionen und Availability Zones. AWS Regionen bieten mehrere physisch getrennte und isolierte Availability Zones, die über Netzwerke mit niedriger Latenz, hohem Durchsatz und hoher Redundanz miteinander verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Zonen ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Mit dem EC2 Image Builder Builder-Service können Sie Images, die in einer Region erstellt wurden, mit anderen Regionen verteilen und ihnen so die Resilienz für AMIs in mehreren Regionen verleihen. Es gibt keinen Mechanismus zum „Sichern“ von Image-Pipelines, Rezepten oder Komponenten. Sie können die Rezept- und Komponentendokumente außerhalb des Image Builder Builder-Service speichern, z. B. in einem Amazon S3 S3-Bucket.

Der EC2 Image Builder kann nicht für High Availability (HA) konfiguriert werden. Sie können Images auf mehrere Regionen verteilen, um die Verfügbarkeit der Images zu erhöhen.

Weitere Informationen zu AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Infrastruktursicherheit in Image Builder

Das AWS globale Netzwerk bietet Sicherheitsfunktionen und steuert den Netzwerkzugriff für Dienste wie EC2 Image Builder. Weitere Informationen zur Infrastruktursicherheit, die diese Dienste AWS bieten, finden Sie im Abschnitt Infrastruktursicherheit im Whitepaper Einführung in die AWS Sicherheit.

Um Anfragen für Image Builder API-Aktionen über das AWS globale Netzwerk zu senden, muss Ihre Client-Software die folgenden Sicherheitsrichtlinien erfüllen:

- Um Anfragen für Image Builder Builder-API-Aktionen zu senden, muss die Clientsoftware eine unterstützte Version von Transport Layer Security (TLS) verwenden.

  > **Note**
  >
  > AWS stellt die Unterstützung für die TLS-Versionen 1.0 und 1.1 ein. Wir empfehlen dringend, dass Sie Ihre Client-Software so aktualisieren, dass sie TLS Version 1.2

oder höher verwendet, damit Sie weiterhin eine Verbindung herstellen können. Weitere Informationen finden Sie in diesem AWS Sicherheits-Blogbeitrag.

- Die Client-Software muss Cipher Suites mit Perfect Forward Secrecy (PFS) wie Ephemeral Diffie-Hellman (DHE) oder Elliptic Curve Ephemeral Diffie-Hellman (ECDHE) unterstützen. Die meisten aktuellen Systeme, wie Java 7 und höher, unterstützen diese Modi.
- Sie müssen Ihre API-Anfragen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signieren, der einem AWS Identity and Access Management (IAM-) Prinzipal zugeordnet ist. Oder Sie können das AWS Security Token Service(AWS STS) verwenden, um temporäre Sicherheitsanmeldedaten für Ihre Anfragen zu generieren.

Darüber hinaus müssen die EC2-Instances, die Image Builder zum Erstellen und Testen von Images verwendet, AWS Systems Manager Zugriff haben.

Patchverwaltung in EC2 Image Builder

EC2 Image Builder bietet die neuesten Versionen von Amazon Linux 2, Amazon Linux 2023, Red Hat Enterprise Linux (RHEL), CentOS, Ubuntu, SUSE Linux Enterprise Server und Windows 2012 R2 und spätere AMIs als verwaltete Image-Quellen. Sie behalten die Verantwortung für das Patchen des Amazon EC2-Systems gemäß dem Modell der gemeinsamen Verantwortung. Wenn die EC2-Instances in Ihrem Anwendungs-Workload einfach ausgetauscht werden können, ist es möglicherweise effizienter, das Basis-AMI zu aktualisieren und alle Rechenknoten, die auf diesem Image basieren, erneut bereitzustellen.

Im Folgenden finden Sie zwei Möglichkeiten, wie Sie Ihre Image Builder Builder-AMIs auf dem neuesten Stand halten können.

- AWS-bereitgestellte Patching-Komponenten — EC2 Image Builder bietet zwei Build-Komponenten `update-linux` und`update-windows`, die alle ausstehenden Betriebssystemupdates installieren. Diese Komponenten verwenden das `UpdateOS` Aktionsmodul. Weitere Informationen finden Sie unter OS aktualisieren. Die Komponenten können Ihren Image-Build-Pipelines hinzugefügt werden, indem Sie sie aus der Liste der AWS bereitgestellten Komponenten auswählen.
- Benutzerdefinierte Builder-Komponenten mit Patching-Vorgängen — Um Patches selektiv auf Betriebssystemen unterstützter AMIs zu installieren oder zu aktualisieren, können Sie eine Image Builder-Komponente erstellen, um die erforderlichen Patches zu installieren. Eine benutzerdefinierte Komponente kann Patches mithilfe von Shell-Skripts (Bash oder PowerShell) installieren oder das `UpdateOS` Aktionsmodul verwenden, um Patches für die Installation oder den

Ausschluss anzugeben. Weitere Informationen finden Sie unter Aktionsmodule, die vom AWSTOE Komponentenmanager unterstützt werden.

Komponente, die das `UpdateOS` Aktionsmodul verwendet (Linux und Windows)

```
schemaVersion: 1.0
phases:
  - name: build
 steps:
   - name: UpdateOS
  action: UpdateOS
```

Komponente, die Bash verwendet, um Yum-Updates zu installieren

```
schemaVersion: 1.0
phases:
  - name: build
 steps:
   - name: InstallYumUpdates
  action: ExecuteBash
  inputs:
    commands:
   - sudo yum update -y
```

Bewährte Sicherheitsmethoden für EC2 Image Builder

EC2 Image Builder bietet eine Reihe von Sicherheitsfunktionen, die Sie bei der Entwicklung und Implementierung Ihrer eigenen Sicherheitsrichtlinien berücksichtigen sollten. Die folgenden bewährten Methoden sind allgemeine Richtlinien und keine vollständige Sicherheitslösung. Da diese bewährten Methoden für Ihre Umgebung möglicherweise nicht angemessen oder ausreichend sind, sollten Sie sie als hilfreiche Überlegungen und nicht als bindend ansehen.

- Verwenden Sie in Image Builder Builder-Rezepten keine übermäßig freizügigen Sicherheitsgruppen.
- Teilen Sie keine Bilder mit Konten, denen Sie nicht vertrauen.
- Veröffentlichen Sie keine Bilder, die private oder sensible Daten enthalten.
- Wenden Sie bei der Image-Erstellung alle verfügbaren Windows- oder Linux-Sicherheitspatches an.

Wir empfehlen Ihnen dringend, Ihre Images zu testen, um den Sicherheitsstatus und die geltenden Sicherheitsstandards zu überprüfen. Lösungen wie Amazon Inspector können dabei helfen, den Sicherheits- und Compliance-Status von Bildern zu überprüfen.

IMDSv2 für Image Builder Builder-Pipelines

Wenn Ihre Image Builder-Pipeline ausgeführt wird, sendet sie HTTP-Anfragen zum Starten von EC2-Instances, die Image Builder zum Erstellen und Testen Ihres Images verwendet. Um die Version von IMDS zu konfigurieren, die Ihre Pipeline für die Startanfragen verwendet, legen Sie den `httpTokens` Parameter in den Metadateneinstellungen Ihrer Image Builder Builder-Infrastrukturkonfigurationsinstanz fest.

> **Note**
>
> Wir empfehlen, dass Sie alle EC2-Instances, die Image Builder aus einem Pipeline-Build startet, für die Verwendung von IMDSv2 konfigurieren, sodass für Anfragen zum Abrufen von Instance-Metadaten ein signierter Token-Header erforderlich ist.

Weitere Informationen zur Image Builder Builder-Infrastrukturkonfiguration finden Sie unterEC2 Image Builder Builder-Infrastrukturkonfiguration verwalten. Weitere Informationen zu EC2-Instance-Metadatenoptionen für Linux-Images finden Sie unter Konfiguration der Instance-Metadatenoptionen im Amazon EC2 EC2-Benutzerhandbuch für Linux-Instances. Informationen zu Windows-Images finden Sie unter Konfiguration der Optionen für Instance-Metadaten im Amazon EC2 EC2-Benutzerhandbuch für Windows-Instances.

Bereinigung nach dem Build erforderlich

Nachdem Image Builder alle Build-Schritte für Ihr benutzerdefiniertes Image abgeschlossen hat, bereitet Image Builder die Build-Instanz für Tests und Image-Erstellung vor. Bevor Sie die Build-Instanz herunterfahren, um den Snapshot zu erstellen, führt Image Builder die folgende Bereinigung durch, um die Sicherheit Ihres Images zu gewährleisten:

Linux

Die Image Builder Builder-Pipeline führt ein Bereinigungsskript aus, um sicherzustellen, dass das endgültige Image den bewährten Sicherheitsmethoden entspricht, und um alle Build-Artefakte oder Einstellungen zu entfernen, die nicht in Ihren Snapshot übertragen werden sollten. Sie können jedoch Abschnitte des Skripts überspringen oder die Benutzerdaten vollständig

überschreiben. Daher entsprechen die von Image Builder Builder-Pipelines erstellten Bilder nicht unbedingt bestimmten regulatorischen Kriterien.

Wenn die Pipeline ihre Build- und Testphasen abgeschlossen hat, führt Image Builder automatisch das folgende Bereinigungsskript aus, kurz bevor das Ausgabe-Image erstellt wird.

> **Important**
>
> Wenn Sie Benutzerdaten in Ihrem Rezept überschreiben, wird das Skript nicht ausgeführt. Stellen Sie in diesem Fall sicher, dass Sie einen Befehl in Ihre Benutzerdaten aufnehmen, der eine leere Datei mit dem Namen erstellt`perform_cleanup`. Image Builder erkennt diese Datei und führt das Bereinigungsskript aus, bevor das neue Image erstellt wird.

```
#!/bin/bash
if [[ ! -f {{workingDirectory}}/perform_cleanup ]]; then
    echo "Skipping cleanup"
    exit 0
else
    sudo rm -f {{workingDirectory}}/perform_cleanup
fi

function cleanup() {
    FILES=("$@")
    for FILE in "${FILES[@]}"; do
        if [[ -f "$FILE" ]]; then
            echo "Deleting $FILE";
            sudo shred -zuf $FILE;
        fi;
        if [[ -f $FILE ]]; then
            echo "Failed to delete '$FILE'. Failing."
            exit 1
        fi;
    done
};


# Clean up for cloud-init files
CLOUD_INIT_FILES=(
    "/etc/sudoers.d/90-cloud-init-users"
    "/etc/locale.conf"
    "/var/log/cloud-init.log"
```

```
    "/var/log/cloud-init-output.log"
)
if [[ -f {{workingDirectory}}/skip_cleanup_cloudinit_files ]]; then
    echo "Skipping cleanup of cloud init files"
else
    echo "Cleaning up cloud init files"
    cleanup "${CLOUD_INIT_FILES[@]}"
    if [[ $( sudo find /var/lib/cloud -type f | sudo wc -l ) -gt 0 ]]; then
        echo "Deleting files within /var/lib/cloud/*"
        sudo find /var/lib/cloud -type f -exec shred -zuf {} \;
    fi;

    if [[ $( sudo ls /var/lib/cloud | sudo wc -l ) -gt 0 ]]; then
        echo "Deleting /var/lib/cloud/*"
        sudo rm -rf /var/lib/cloud/* || true
    fi;
fi;


# Clean up for temporary instance files
INSTANCE_FILES=(
    "/etc/.updated"
    "/etc/aliases.db"
    "/etc/hostname"
    "/var/lib/misc/postfix.aliasesdb-stamp"
    "/var/lib/postfix/master.lock"
    "/var/spool/postfix/pid/master.pid"
    "/var/.updated"
    "/var/cache/yum/x86_64/2/.gpgkeyschecked.yum"
)
if [[ -f {{workingDirectory}}/skip_cleanup_instance_files ]]; then
    echo "Skipping cleanup of instance files"
else
    echo "Cleaning up instance files"
    cleanup "${INSTANCE_FILES[@]}"
fi;


# Clean up for ssh files
SSH_FILES=(
    "/etc/ssh/ssh_host_rsa_key"
    "/etc/ssh/ssh_host_rsa_key.pub"
    "/etc/ssh/ssh_host_ecdsa_key"
    "/etc/ssh/ssh_host_ecdsa_key.pub"
```

```
    "/etc/ssh/ssh_host_ed25519_key"
    "/etc/ssh/ssh_host_ed25519_key.pub"
    "/root/.ssh/authorized_keys"
)
if [[ -f {{workingDirectory}}/skip_cleanup_ssh_files ]]; then
    echo "Skipping cleanup of ssh files"
else
    echo "Cleaning up ssh files"
    cleanup "${SSH_FILES[@]}"
    USERS=$(ls /home/)
    for user in $USERS; do
        echo Deleting /home/"$user"/.ssh/authorized_keys;
        sudo find /home/"$user"/.ssh/authorized_keys -type f -exec shred -zuf {} \;
    done
    for user in $USERS; do
        if [[ -f /home/"$user"/.ssh/authorized_keys ]]; then
            echo Failed to delete /home/"$user"/.ssh/authorized_keys;
            exit 1
        fi;
    done;
fi;


# Clean up for instance log files
INSTANCE_LOG_FILES=(
    "/var/log/audit/audit.log"
    "/var/log/boot.log"
    "/var/log/dmesg"
    "/var/log/cron"
)
if [[ -f {{workingDirectory}}/skip_cleanup_instance_log_files ]]; then
    echo "Skipping cleanup of instance log files"
else
    echo "Cleaning up instance log files"
    cleanup "${INSTANCE_LOG_FILES[@]}"
fi;

# Clean up for TOE files
if [[ -f {{workingDirectory}}/skip_cleanup_toe_files ]]; then
    echo "Skipping cleanup of TOE files"
else
    echo "Cleaning TOE files"
    if [[ $( sudo find {{workingDirectory}}/TOE_* -type f | sudo wc -l) -gt 0 ]];
 then
```

```
        echo "Deleting files within {{workingDirectory}}/TOE_*"
        sudo find {{workingDirectory}}/TOE_* -type f -exec shred -zuf {} \;
    fi
    if [[ $( sudo find {{workingDirectory}}/TOE_* -type f | sudo wc -l) -gt 0 ]];
 then
        echo "Failed to delete {{workingDirectory}}/TOE_*"
        exit 1
    fi
    if [[ $( sudo find {{workingDirectory}}/TOE_* -type d | sudo wc -l) -gt 0 ]];
 then
        echo "Deleting {{workingDirectory}}/TOE_*"
        sudo rm -rf {{workingDirectory}}/TOE_*
    fi
    if [[ $( sudo find {{workingDirectory}}/TOE_* -type d | sudo wc -l) -gt 0 ]];
 then
        echo "Failed to delete {{workingDirectory}}/TOE_*"
        exit 1
    fi
fi

# Clean up for ssm log files
if [[ -f {{workingDirectory}}/skip_cleanup_ssm_log_files ]]; then
    echo "Skipping cleanup of ssm log files"
else
    echo "Cleaning up ssm log files"
    if [[ $( sudo find /var/log/amazon/ssm -type f | sudo wc -l) -gt 0 ]]; then
        echo "Deleting files within /var/log/amazon/ssm/*"
        sudo find /var/log/amazon/ssm -type f -exec shred -zuf {} \;
    fi
    if [[ $( sudo find /var/log/amazon/ssm -type f | sudo wc -l) -gt 0 ]]; then
        echo "Failed to delete /var/log/amazon/ssm"
        exit 1
    fi
    if [[ -d "/var/log/amazon/ssm" ]]; then
        echo "Deleting /var/log/amazon/ssm/*"
        sudo rm -rf /var/log/amazon/ssm
    fi
    if [[ -d "/var/log/amazon/ssm" ]]; then
        echo "Failed to delete /var/log/amazon/ssm"
        exit 1
    fi
fi
```

```
if [[ $( sudo find /var/log/sa/sa* -type f | sudo wc -l ) -gt 0 ]]; then
    echo "Deleting /var/log/sa/sa*"
    sudo shred -zuf /var/log/sa/sa*
fi
if [[ $( sudo find /var/log/sa/sa* -type f | sudo wc -l ) -gt 0 ]]; then
    echo "Failed to delete /var/log/sa/sa*"
    exit 1
fi

if [[ $( sudo find /var/lib/dhclient/dhclient*.lease -type f | sudo wc -l ) -gt
 0 ]]; then
        echo "Deleting /var/lib/dhclient/dhclient*.lease"
        sudo shred -zuf /var/lib/dhclient/dhclient*.lease
fi
if [[ $( sudo find /var/lib/dhclient/dhclient*.lease -type f | sudo wc -l ) -gt
 0 ]]; then
        echo "Failed to delete /var/lib/dhclient/dhclient*.lease"
        exit 1
fi

if [[ $( sudo find /var/tmp -type f | sudo wc -l) -gt 0 ]]; then
        echo "Deleting files within /var/tmp/*"
        sudo find /var/tmp -type f -exec shred -zuf {} \;
fi
if [[ $( sudo find /var/tmp -type f | sudo wc -l) -gt 0 ]]; then
        echo "Failed to delete /var/tmp"
        exit 1
fi
if [[ $( sudo ls /var/tmp | sudo wc -l ) -gt 0 ]]; then
        echo "Deleting /var/tmp/*"
        sudo rm -rf /var/tmp/*
fi

# Shredding is not guaranteed to work well on rolling logs

if [[ -f "/var/lib/rsyslog/imjournal.state" ]]; then
        echo "Deleting /var/lib/rsyslog/imjournal.state"
        sudo shred -zuf /var/lib/rsyslog/imjournal.state
        sudo rm -f /var/lib/rsyslog/imjournal.state
fi

if [[ $( sudo ls /var/log/journal/ | sudo wc -l ) -gt 0 ]]; then
        echo "Deleting /var/log/journal/*"
        sudo find /var/log/journal/ -type f -exec shred -zuf {} \;
```

```
        sudo rm -rf /var/log/journal/*
fi

sudo touch /etc/machine-id
```

Windows

Nachdem die Image Builder Builder-Pipeline Windows-Abbilder angepasst hat, wird das Microsoft Sysprep-Hilfsprogramm ausgeführt. Diese Aktionen folgen den AWS bewährten Methoden zum Härten und Reinigen des Images.

Überschreiben Sie das Linux-Bereinigungsskript

Image Builder erstellt Bilder, die standardmäßig sicher sind und unseren bewährten Sicherheitsmethoden entsprechen. In einigen fortgeschritteneren Anwendungsfällen müssen Sie jedoch möglicherweise einen oder mehrere Abschnitte des integrierten Bereinigungsskripts überspringen. Wenn Sie einen Teil der Bereinigung überspringen müssen, empfehlen wir Ihnen dringend, Ihr Ausgabe-AMI zu testen, um die Sicherheit Ihres Images zu gewährleisten.

Important

Das Überspringen von Abschnitten im Bereinigungsskript kann dazu führen, dass vertrauliche Informationen wie Kontodetails des Besitzers oder SSH-Schlüssel in das endgültige Image aufgenommen und in jedem Fall von diesem Image aus gestartet werden. Möglicherweise treten auch Probleme beim Starten in verschiedenen Availability Zones, Regionen oder Konten auf.

In der folgenden Tabelle werden die Abschnitte des Bereinigungsskripts, die in diesem Abschnitt gelöschten Dateien und die Dateinamen beschrieben, mit denen Sie einen Abschnitt kennzeichnen können, den Image Builder überspringen soll. Um einen bestimmten Abschnitt des Bereinigungsskripts zu überspringen, können Sie das Aktionsmodul der CreateFile Komponente oder einen Befehl in Ihren Benutzerdaten (falls überschrieben) verwenden, um eine leere Datei mit dem Namen zu erstellen, der in der Spalte Dateiname des Übersprungsabschnitts angegeben ist.

Note

Die Dateien, die Sie erstellen, um einen Abschnitt des Bereinigungsskripts zu überspringen, sollten keine Dateierweiterung enthalten. Wenn Sie beispielsweise den

`CLOUD_INIT_FILES` Abschnitt des Skripts überspringen möchten, aber eine Datei mit dem Namen erstellen`skip_cleanup_cloudinit_files.txt`, erkennt Image Builder die übersprungene Datei nicht.

Eingabe

Abschnitt aufräumen	Dateien wurden entfernt	Dateiname des Abschnitts überspringen

`CLOUD_INIT_FILES`	`/etc/sudoers.d/90-cloud-init-users`
	`/etc/locale.conf`
	`/var/log/cloud-init.log`
	`/var/log/cloud-init-output.log`
		`skip_cleanup_cloudinit_files`

`INSTANCE_FILES`	`/etc/.updated`
	`/etc/aliases.db`
	`/etc/hostname`
	`/var/lib/misc/postfix.aliasesdb-stamp`
	`/var/lib/postfix/master.lock`
	`/var/spool/postfix/pid/master.pid`
	`/var/.updated`
		`skip_cleanup_instance_files`

Abschnitt aufräumen	Dateien wurden entfernt	Dateiname des Abschnitts überspringen
	`/var/cache/yum/x86_64/2/.gpgkeyschecked.yum`	

`SSH_FILES`	`/etc/ssh/ssh_host_rsa_key`
	`/etc/ssh/ssh_host_rsa_key.pub`
	`/etc/ssh/ssh_host_ecdsa_key`
	`/etc/ssh/ssh_host_ecdsa_key.pub`
	`/etc/ssh/ssh_host_ed25519_key`
	`/etc/ssh/ssh_host_ed25519_key.pub`
	`/root/.ssh/authorized_keys`
	`/home/<all users>/.ssh/authorized_keys;`
		`skip_cleanup_ssh_files`

`INSTANCE_LOG_FILES`	`/var/log/audit/audit.log`
	`/var/log/boot.log`
	`/var/log/dmesg`
	`/var/log/cron`
		`skip_cleanup_instance_log_files`

Abschnitt aufräumen	Dateien wurden entfernt	Dateiname des Abschnitts überspringen
TOE_FILES	{{workingDirectory}}/TOE_*	skip_cleanup_toe_files
SSM_LOG_FILES	/var/log/amazon/ssm/*	skip_cleanup_ssm_log_files

Problembehandlung bei EC2 Image Builder

EC2 Image Builder lässt sich AWS-Services zur Überwachung und Fehlerbehebung integrieren, um Sie bei der Behebung von Problemen bei der Image-Erstellung zu unterstützen. Image Builder verfolgt und zeigt den Fortschritt für jeden Schritt im Imageerstellungsprozess an. Darüber hinaus kann Image Builder Protokolle an einen von Ihnen angegebenen Amazon S3 S3-Speicherort exportieren.

Für eine erweiterte Problembehandlung können Sie mit Run Command vordefinierte Befehle und Skripts AWS Systems Manager ausführen.

Inhalt

Problembehandlung bei Pipeline-Buil

Wenn ein Image Builder-Pipeline-Build fehlschlägt, gibt Image Builder eine Fehlermeldung zurück, die den Fehler beschreibt. Image Builder gibt auch a `workflow execution ID` in der Fehlermeldung zurück, wie in der folgenden Beispielausgabe:

```
Workflow Execution ID: wf-12345abc-6789-0123-abc4-567890123abc failed with reason: …
```

Image Builder organisiert und leitet die Aktionen zur Image-Erstellung über eine Reihe von Schritten, die für die Laufzeitphasen in seinem standardmäßigen Image-Erstellungsprozess definiert sind. Den Erstellungs- und Testphasen des Prozesses ist jeweils ein Workflow zugeordnet. Wenn Image Builder einen Workflow ausführt, um ein neues Image zu erstellen oder zu testen, generiert es eine Workflow-Metadatenressource, die die Laufzeitdetails verfolgt.

Container-Images verfügen über einen zusätzlichen Workflow, der während der Verteilung ausgeführt wird.

Recherchieren Sie in Ihrem Workflow nach Einzelheiten zu Ausfällen von Runtime-Instances

Um einen Laufzeitfehler für Ihren Workflow zu beheben, können Sie die GetWorkflowExecutionund ListWorkflowStepExecutionsAPI-Aktionen mit Ihrem aufrufen`workflow execution ID`.

Überprüfen Sie die Workflow-Laufzeitprotokolle

- CloudWatch Amazon-Protokolle

  Image Builder veröffentlicht detaillierte Workflow-Ausführungsprotokolle in der folgenden Image Builder CloudWatch Builder-Protokollgruppe und dem folgenden Stream:

  LogGroup:

  ```
  /aws/imagebuilder/ImageName
  ```

  LogStream (x.x.x/x):

  ```
  ImageVersion/ImageBuildVersion
  ```

  Mit CloudWatch Logs können Sie Protokolldaten mit Filtermustern durchsuchen. Weitere Informationen finden Sie unter Durchsuchen von Protokolldaten mithilfe von Filtermustern im Amazon CloudWatch Logs-Benutzerhandbuch.

- AWS CloudTrail

  Alle Build-Aktivitäten werden ebenfalls protokolliert CloudTrail , wenn sie in Ihrem Konto aktiviert sind. Sie können CloudTrail Ereignisse nach der Quelle filtern`imagebuilder.amazonaws.com`. Alternativ können Sie nach der Amazon EC2 EC2-Instance-ID suchen, die im Ausführungsprotokoll zurückgegeben wird, um weitere Details zur Pipeline-Ausführung zu erhalten.

- Amazon Simple Storage Service (S3)

  Wenn Sie in Ihrer Infrastrukturkonfiguration einen S3-Bucket-Namen und ein key prefix angegeben haben, folgt der Laufzeitprotokollpfad des Workflow-Schritts diesem Muster:

  ```
  S3://S3BucketName/KeyPrefix/ImageName/ImageVersion/ImageBuildVersion/
  WorkflowExecutionId/StepName
  ```

  Die Protokolle, die Sie an Ihren S3-Bucket senden, zeigen die Schritte und Fehlermeldungen für Aktivitäten auf der EC2-Instance während des Image-Build-Prozesses. Die Protokolle enthalten Protokollausgaben des Komponenten-Managers, die Definitionen der Komponenten, die ausgeführt wurden, und die detaillierte Ausgabe (in JSON) aller Schritte, die auf der Instance ausgeführt wurden. Wenn Sie auf ein Problem stoßen, sollten Sie diese Dateien zunächst überprüfen`application.log`, um die Ursache des Problems auf der Instanz zu diagnostizieren.

Standardmäßig fährt Image Builder die Amazon EC2 EC2-Build- oder Test-Instance herunter, die ausgeführt wird, wenn die Pipeline ausfällt. Sie können die Instance-Einstellungen für die Infrastrukturkonfigurationsressource ändern, die Ihre Pipeline verwendet, um Ihre Build- oder Test-Instance zur Fehlerbehebung beizubehalten.

Um die Instanzeinstellungen in der Konsole zu ändern, müssen Sie das Kontrollkästchen Instanz bei Ausfall beenden deaktivieren, das sich im Abschnitt Einstellungen zur Fehlerbehebung Ihrer Infrastrukturkonfigurationsressource befindet.

Sie können die Instanzeinstellungen auch mit dem update-infrastructure-configuration Befehl in der ändern AWS CLI. Legen Sie den `terminateInstanceOnFailure` Wert `false` in der JSON-Datei, auf die der Befehl mit dem `--cli-input-json` Parameter verweist, auf fest. Details hierzu finden Sie unter Aktualisieren Sie eine Infrastrukturkonfiguration.

Fehlerbehebungsszenarien

In diesem Abschnitt werden die folgenden detaillierten Problemlösungsszenarien aufgeführt:

- Zugriff verweigert — Statuscode 403
- Timeout beim Erstellen während der Überprüfung der Verfügbarkeit des Systems Manager Agents auf der Build-Instance
- Die sekundäre Windows-Festplatte ist beim Start offline
- Der Build schlägt mit dem CIS-gehärteten Basis-Image fehl
- AssertInventoryCollection schlägt fehl (Systems Manager Automation)

Um die Details eines Szenarios zu sehen, wählen Sie den Titel des Szenarios aus, um es zu erweitern. Sie können mehrere Titel gleichzeitig erweitern lassen.

Zugriff verweigert — Statuscode 403

Beschreibung

Der Pipeline-Build schlägt mit dem Statuscode "AccessDenied: Zugriff verweigert: 403“ fehl.

Ursache

Mögliche Gründe hierfür sind:

- Das Instanzprofil verfügt nicht über die erforderlichen Berechtigungen für den Zugriff auf APIs oder Komponentenressourcen.
- In der Instance-Profilrolle fehlen Berechtigungen, die für die Anmeldung bei Amazon S3 erforderlich sind. In den meisten Fällen tritt dies auf, wenn die Instance-Profilrolle keine PutObjectBerechtigungen für Ihre S3-Buckets hat.

Lösung

Je nach Ursache kann dieses Problem wie folgt behoben werden:

- Im Instanzprofil fehlen verwaltete Richtlinien — Fügen Sie die fehlenden Richtlinien zu Ihrer Instanzprofilrolle hinzu. Führen Sie dann die Pipeline erneut aus.
- Dem Instanzprofil fehlen Schreibberechtigungen für den S3-Bucket — Fügen Sie Ihrer Instanzprofilrolle eine Richtlinie hinzu, die PutObjectSchreibberechtigungen für Ihren S3-Bucket gewährt. Führen Sie dann die Pipeline erneut aus.

Timeout beim Erstellen während der Überprüfung der Verfügbarkeit des Systems Manager Agents auf der Build-Instance

Beschreibung

Der Pipeline-Build schlägt mit „status = 'TimedOut'“ und „failure message = 'Timeout für den Schritt bei der Überprüfung der Verfügbarkeit des Systems Manager Manager-Agenten auf der Zielinstanz (en) '“ fehl.

Ursache

Mögliche Gründe hierfür sind:

- Die Instanz, die gestartet wurde, um die Build-Operationen und Komponenten auszuführen, konnte nicht auf den Systems Manager Manager-Endpunkt zugreifen.
- Das Instanzprofil verfügt nicht über die erforderlichen Berechtigungen.

Lösung

Abhängig von der möglichen Ursache kann dieses Problem wie folgt behoben werden:

- Zugriffsproblem, privates Subnetz — Wenn Sie ein privates Subnetz einbauen, stellen Sie sicher, dass Sie PrivateLink Endpunkte für Systems Manager, Image Builder und, falls Sie protokollieren möchten, Amazon S3 S3/ eingerichtet haben. CloudWatch Weitere Informationen zum Einrichten von PrivateLink Endpunkten finden Sie unter Konzepte für VPC-Endgeräte ().AWS PrivateLink
- Fehlende Berechtigungen — Fügen Sie Ihrer mit dem IAM-Dienst verknüpften Rolle für Image Builder die folgenden verwalteten Richtlinien hinzu:
  - EC2 InstanceProfileForImageBuilder
  - EC2 ECR InstanceProfileForImageBuilder ContainerBuilds
  - Amazon SSM ManagedInstanceCore

  Weitere Informationen zur dienstverknüpften Image Builder Builder-Rolle finden Sie unterVerwenden von serviceverknüpften Rollen für EC2 Image Builder.

Die sekundäre Windows-Festplatte ist beim Start offline

Beschreibung

Wenn der Instance-Typ, der zum Erstellen eines Image Builder Builder-Windows-AMI verwendet wird, nicht mit dem Instance-Typ übereinstimmt, der zum Starten über das AMI verwendet wird, kann ein Problem auftreten, wenn Nicht-Root-Volumes beim Start offline sind. Dies ist in erster Linie der Fall, wenn die Build-Instance eine neuere Architektur als die Startinstanz verwendet.

Das folgende Beispiel zeigt, was passiert, wenn ein Image Builder Builder-AMI auf einem EC2-Nitro-Instance-Typ basiert und auf einer EC2-Xen-Instance gestartet wird:

Build-Instance-Typ: m5.large (Nitro)

Instanztyp starten: t2.medium (Xen)

```
PS C:\Users\Administrator> get-disk
Number  Friendly Name  Serial Number         Health Status  Operational Status  Total
 Size  Partition Style
------  -------------  -------------         -------------  ------------------
 ----------  ---------------
0       AWS PVDISK     vol0abc12d34e567f8a9  Healthy        Online                  30
 GB  MBR
1       AWS PVDISK     vol1bcd23e45f678a9b0  Healthy        Offline                  8
 GB  MBR
```

Ursache

Aufgrund der Windows-Standardeinstellungen werden neu entdeckte Festplatten nicht automatisch online geschaltet und formatiert. Wenn der Instanztyp auf EC2 geändert wird, behandelt Windows dies so, als würden neue Festplatten erkannt. Das liegt an der zugrunde liegenden Treiberänderung.

Lösung

Wir empfehlen, dass Sie beim Erstellen Ihres Windows-AMI, von dem aus Sie starten möchten, dasselbe System von Instance-Typen verwenden. Nehmen Sie in Ihrer Infrastrukturkonfiguration keine Instance-Typen auf, die auf unterschiedlichen Systemen basieren. Wenn einer der von Ihnen angegebenen Instance-Typen das Nitro-System verwendet, sollten alle das Nitro-System verwenden.

Weitere Informationen zu Instances, die auf dem Nitro-System basieren, finden Sie unter Instances, die auf dem Nitro-System basieren im Amazon EC2 EC2-Benutzerhandbuch für Windows-Instances.

Der Build schlägt mit dem CIS-gehärteten Basis-Image fehl

Beschreibung

Sie verwenden ein CIS-gehärtetes Basis-Image und der Build schlägt fehl.

Ursache

Wenn das `/tmp` Verzeichnis als klassifiziert ist`noexec`, kann dies dazu führen, dass Image Builder fehlschlägt.

Lösung

Wählen Sie im `workingDirectory` Feld des Image-Rezepts einen anderen Speicherort für Ihr Arbeitsverzeichnis aus. Weitere Informationen finden Sie in der Beschreibung des ImageRecipeDatentyps.

AssertInventoryCollection schlägt fehl (Systems Manager Automation)

Beschreibung

Systems Manager Automation zeigt einen Fehler im `AssertInventoryCollection` Automatisierungsschritt an.

Ursache

Möglicherweise haben Sie oder Ihre Organisation eine Systems Manager State Manager-Zuordnung erstellt, die Inventarinformationen für EC2-Instances sammelt. Wenn die erweiterte Erfassung von

Bildmetadaten für Ihre Image Builder-Pipeline aktiviert ist (dies ist die Standardeinstellung), versucht Image Builder, eine neue Inventarzuordnung für die Build-Instanz zu erstellen. Systems Manager lässt jedoch nicht mehrere Inventarzuordnungen für verwaltete Instanzen zu und verhindert eine neue Zuordnung, falls bereits eine vorhanden ist. Dadurch schlägt der Vorgang fehl und der Pipeline-Build schlägt fehl.

Lösung

Um dieses Problem zu beheben, deaktivieren Sie die erweiterte Erfassung von Bildmetadaten mit einer der folgenden Methoden:

- Aktualisieren Sie Ihre Image-Pipeline in der Konsole, um das Kontrollkästchen Erweiterte Metadatensammlung aktivieren zu deaktivieren. Speichern Sie Ihre Änderungen und führen Sie einen Pipeline-Build aus.

  Weitere Informationen zur Aktualisierung Ihrer AMI-Image-Pipeline mithilfe der EC2 Image Builder Builder-Konsole finden Sie unterAMI-Image-Pipelines aktualisieren (Konsole). Weitere Informationen zur Aktualisierung Ihrer Container-Image-Pipeline mithilfe der EC2 Image Builder Builder-Konsole finden Sie unterAktualisieren Sie eine Container-Image-Pipeline (Konsole).
- Sie können Ihre Image-Pipeline auch mit dem update-image-pipeline Befehl in der AWS CLI aktualisieren. Fügen Sie dazu die `EnhancedImageMetadataEnabled` Eigenschaft in Ihre JSON-Datei ein und setzen Sie sie auf`false`. Das folgende Beispiel zeigt die Eigenschaft, die auf gesetzt ist`false`.

```
{
    "name": "MyWindows2019Pipeline",
    "description": "Builds Windows 2019 Images",
    "enhancedImageMetadataEnabled": false,
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-recipe/2020.12.03",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-infrastructure-configuration",
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-distribution-configuration",
    "imageTestsConfiguration": {
        "imageTestsEnabled": true,
        "timeoutMinutes": 60
    },
    "schedule": {
```

```
        "scheduleExpression": "cron(0 0 * * SUN *)",
        "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
    },
    "status": "ENABLED"
}
```

Um zu verhindern, dass dies bei neuen Pipelines passiert, deaktivieren Sie das Kontrollkästchen Erweiterte Metadatensammlung aktivieren, wenn Sie mit der EC2 Image Builder Builder-Konsole eine neue Pipeline erstellen, oder setzen Sie den Wert der `EnhancedImageMetadataEnabled` Eigenschaft in Ihrer JSON-Datei auf, `false` wenn Sie Ihre Pipeline mit dem erstellen. AWS CLI

Benutzerhandbuch zur Dokumentenhistorie für EC2 Image Builder

In der folgenden Tabelle werden wichtige Änderungen an der Dokumentation nach Datum sortiert beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

- API-Version: 2023-12-12

Änderung	Beschreibung	Datum
STIG Q1-Aktualisierungen	Die Linux STIG-Versionen wurden aktualisiert und STIGS für die Veröffentlichung des ersten Quartals 2024 angewendet. Es gab keine Änderungen an den Windows-Versionen.	23. Februar 2024
Neue Funktion: Verwaltung des Bild-Workflows	Mit Image-Workflows haben Sie mehr Flexibilität, Transparenz und Kontrolle über den Image-Erstellungsprozess. Sie können die Build- und Testschritte für Ihre Workflows anpassen oder den Image Builder Builder-Standardworkflow verwenden.	12. Dezember 2023
STIG Q4-Updates	Die Linux STIG-Versionen wurden aktualisiert und STIGS für die Veröffentlichung im vierten Quartal 2023 angewendet. Es gab keine Änderungen an den Windows-Versionen. Außerdem	07. Dezember 2023

	wurden SCAP für Linux und Windows aktualisiert, um neue Komponenten-, Software- und Benchmark-Zahlen zu erhalten.	
Neue Funktion: Image-Lebenszyklusmanagement	Mithilfe von Richtlinien und Regeln für das Image-Lifecycle-Management können Sie Ihre Ressourcenverwaltungsstrategie definieren, um sicherzustellen, dass veraltete Images und die zugehörigen Ressourcen gekennzeichnet und entfernt werden.	17. November 2023
STIG-Aktualisierungen für das dritte Quartal	Die STIG-Versionen wurden aktualisiert und STIGS für die Veröffentlichung im dritten Quartal 2023 angewendet. Zusätzlich wurde die Meldung aktualisiert, um klarzustellen, dass Pakete von Drittanbietern mit sehr wenigen Ausnahmen nicht automatisch installiert werden. Alle übersprungenen STIGs werden protokolliert.	05. Oktober 2023
Neue STIG-Versionen	Die STIG-Versionen wurden aktualisiert und STIGS für die Veröffentlichung im zweiten Quartal 2023 angewendet.	3. Mai 2023

Neue STIG-Versionen	Die STIG-Versionen wurden aktualisiert und STIGS für die Veröffentlichung des ersten Quartals 2023 angewende t. Unterstützung für AL2023 hinzugefügt.	14. April 2023
Aktualisieren Sie die unterstützten Regionen für AWSTOE	AWSTOE Unterstützung für Folgendes wurde hinzugefügt AWS-Regionen: Asien-Pazifik (Hyderabad), Asien-Pazifik (Jakarta), Europa (Zürich), Europa (Spanien) und Naher Osten (VAE).	13. April 2023
AWSTOE Updates für Anwendungen herunterladen	Die Signatur für den AWSTOE Installationsdownload unter Windows wurde aktualisiert. Außerdem wurde TLS aktualisiert. Beachten Sie, dass für Anwendungsdownloads aus S3-Buckets jetzt TLS Version 1.2 oder höher erforderlich ist.	31. März 2023
Feature-Release: Verbesserte Build-Workflows	Laufzeitdetails für Image-Builds wurden auf der neuen Workflow-Registerkarte in den Image-Build-Versionsdetails hinzugefügt. Verbesserte Informationen zur Fehlerbehebung bei Builds.	30. März 2023

Neue Funktion: CVE-Erkennung und -Berichterstattung	Für Konten, die Amazon Inspector-Scans aktiviert haben, kann Image Builder während der Testphase des Erstellungsprozesses für neue Images, einschließlich in Amazon ECR gespeicherter Container-Images, die Common Vulnerability and Exposures (CVE) -Ergebnisse von Amazon Inspector erfassen. Image Builder erstellt eine Momentaufnahme der Ergebnisse, um die Detailanalyse zu unterstützen. Image Builder berichtet auch über die Anzahl der Ergebnisse, die nach Konto, Pipeline oder Bild gefiltert werden können, und bietet die Möglichkeit, Details aufzuschlüsseln.	30. März 2023
Versionsverlauf hinzugefügt	Versionsverlauf zu den Windows- und Linux-Abschnitten hinzugefügt.	17. Februar 2023
Neue STIG-Versionen	Die STIG-Versionen wurden aktualisiert und STIGS für die Veröffentlichung im vierten Quartal 2022 angewendet.	1. Februar 2023

Feature-Release: AWS Marketplace Integration und CIS-Härtung	Es wurde eine AWS Marketplace Integration hinzugefügt, um ein abonniertes Image einfach zu finden und als Grundlage für ein neues benutzerdefiniertes Image zu verwenden, einschließlich CIS-Hardened-Images und einer neuen CIS-Hardening-Komponente aus dem Center for Internet Security.	13. Januar 2023
CIS-Härtungskomponenten	Es wurden CIS-Härtungskomponenten hinzugefügt, die Eigentum von CIS sind und von CIS verwaltet werden.	13. Januar 2023
Neue STIG-Versionen	Einführung der Ubuntu-Unterstützung, Aktualisierung der STIG-Versionen und Anwendung von STIGS für die Veröffentlichung des zweiten Quartals 2022.	20. Juli 2022
Aktualisierung des Dokuments: Navigation zur Dokumentseite „YAML-Komponente erstellen“	Der Inhalt des Dokuments „YAML-Komponente erstellen“ wurde auf eine eigene Seite verschoben und andere Seiten aktualisiert, sodass sie darauf verweisen.	7. Juni 2022
Neue STIG-Versionen	Die STIG-Versionen wurden aktualisiert und STIGS für die Veröffentlichung des ersten Quartals 2022 angewendet.	25. April 2022

Aktionsmodul hinzugefügt ExecuteDocument	Dokumentation für das `ExecuteDocument` Aktionsmodul unter hinzugefügt`General execution` .	28. März 2022
Feature-Release: Support für schnelleres Starten von Windows AMI	Einstellungen für die Distributionskonfiguration wurden hinzugefügt, um einen schnelleren Start von Windows-AMIs zu unterstützen.	21. Februar 2022
Wartungsversion: AWSTOE Binär-Fingerabdruck aktualisieren	Der binäre Fingerabdruck für das Unterzeichnerzertifikat wurde aktualisiert. AWSTOE	18. Februar 2022
Featurerelease: Konfigurieren Sie die Eingabe für AWSTOE	Unterstützung für die Verwendung einer JSON-Konfigurationsdatei als Eingabe für den AWSTOE run Befehl hinzugefügt.	3. Februar 2022
Neue STIG-Versionen	Die STIG-Versionen wurden aktualisiert und STIGS für die Veröffentlichung im vierten Quartal 2021 angewendet. Außerdem wurde ein Abschnitt für neue SCAP Compliance Checker (SCC) -Komponenten hinzugefügt.	22. Dezember 2021

Funktionsveröffentlichung: VM-Import/Export (VMIE) - Integration	Unterstützung für den VM-Import über alle Kanäle (Konsole, API/CLI usw.) und für den VM-Export über API/CLI hinzugefügt. Der VM-Export ist derzeit nicht über die Image Builder Builder-Konsole verfügbar.	20. Dezember 2021
Feature-Release: AMI-Sharing für AWS Organizations und OUs	Die Distributionskonfiguration wurde aktualisiert, um Unterstützung für die gemeinsame Nutzung von Ausgangs-AMIs mit AWS Organizations und OUs hinzuzufügen.	24. November 2021
Aktualisierung des Dokuments: Stufen und Phasen der Aktualisierung der Komponenten	Erweiterter Inhalt für Komponentenstufen in Image Builder und deren Interaktion mit AWSTOE Komponentenphasen.	22. September 2021
Aktualisierung des Dokuments: CloudTrail Integrationsinhalte hinzufügen	Zusammenfassung der Überwachung und Inhalt CloudTrail zur Integration hinzugefügt.	17. September 2021
Neue STIG-Versionen	Die STIG-Versionen wurden aktualisiert und STIGS für die Veröffentlichung im dritten Quartal 2021 angewendet.	10 September 2021

Neue Funktion: EventBridge Amazon-Integration	EventBridge Unterstützung hinzugefügt, mit der Sie Image Builder mit Ereigniss en aus verwandten Bereichen verbinden und Ereigniss e auf der Grundlage von Regeln initiieren können AWS-Services, die in definiert sind EventBridge.	18. August 2021
Dokumentaktualisierung: Seiten neu anordnen AWSTOE	Die AWSTOE Seiten wurden aus Gründen der Übersicht lichkeit neu angeordnet.	11. August 2021
Feature-Release: Parametri sierte Komponenten und zusätzliche Instanzkonfigurati on	Unterstützung für die Angabe von Parametern zur Anpassung von Komponent en für Rezepte hinzugefü gt. Erweiterte Konfiguration der EC2-Instances, die zum Erstellen und Testen von Images verwendet werden, einschließlich der Möglichke it, Befehle anzugeben, die beim Start ausgeführt werden, und mehr Kontrolle über die Installation und Deinstall ation des Systems Manager Manager-Agenten.	7. Juli 2021
Neue STIG-Versionen	Die STIG-Versionen wurden aktualisiert und STIGS für die Veröffentlichung im zweiten Quartal 2021 angewendet.	30. Juni 2021

Verbesserung: Verbesserungen beim Tagging	Verbessertes Messaging rund um das Verschlagworten von Ressourcen.	25. Juni 2021
Funktionsveröffentlichung: Starten Sie die Vorlagenintegration	Unterstützung für die Verwendung von Amazon EC2 EC2-Startvorlagen für die AMI-Verteilung wurde in den Distributionseinstellungen hinzugefügt.	7. April 2021
Funktionsversion: Verbesserungen beim Erstellen von Containern	Unterstützung für die Konfiguration von Blockgerätezuordnungen und die Angabe von AMIs, die als Basis-Image für Container-Builds verwendet werden sollen, wurde hinzugefügt.	7. April 2021
Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewandte STIGS.	5. März 2021
Aktualisieren Sie die Cron-Ausdrücke	Die Cron-Verarbeitung von Image Builder wurde aktualisiert, um die Granularität von Cron-Ausdrücken auf die Minute genau zu erhöhen und eine standardmäßige Cron-Scheduling-Engine zu verwenden. Die Beispiele wurden mit dem neuen Format aktualisiert.	8. Februar 2021

Feature-Release: Container-Unterstützung	Unterstützung für die Erstellung von Docker-Container-Images mit Image Builder mit Registrierung und Speicherung der resultierenden Bilder in Amazon Elastic Container Registry (Amazon ECR) hinzugefügt. Die Inhalte wurden neu angeordnet, um neuen Funktionen Rechnung zu tragen und künftigem Wachstum Rechnung zu tragen.	17. Dezember 2020
Restrukturierte Cron-Dokumentation	Auf dieser Seite finden Sie nun weitere Informationen zur Funktionsweise von Cron mit Image Builder Builder-Pipeline-Builds sowie Informationen zur UTC-Zeit. Platzhalter, die für bestimmte Felder nicht zulässig sind, wurden entfernt. Zu den Beispielen gehören jetzt Ausdrucksbeispiele sowohl für die Konsole als auch für die CLI.	13. November 2020
Konsolenversion 2.0: Aktualisierung der Pipeline-Bearbeitung	Inhaltsänderungen in den Tutorials „Erste Schritte“ und „Pipelines erstellen“ sowie auf der Seite „Image-Pipelines verwalten“, um neue Konsolenfunktionen und Abläufe zu integrieren.	13. November 2020

Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewandte STIGS. Hinweis: Das Listenformat wurde geändert, sodass STIGs angezeigt werden, die standardmäßig angewendet werden.	15. Oktober 2020
Support für Looping-Konstrukte in AWSTOE	Erstellen Sie Schleifen konstrukte, um eine wiederholte Abfolge von Anweisungen in der Anwendung zu definieren. AWSTOE	29. Juli 2020
Support der lokalen Entwicklung von AWSTOE Komponenten	Entwickeln und testen Sie Image-Komponenten lokal mit der AWSTOE Anwendung.	28. Juli 2020
Verschlüsselte AMIs	EC2 Image Builder bietet Unterstützung für die verschlüsselte AMI-Verteilung.	1. Juli 2020
AutoScaling Verfall	Die Verwendung von zum Starten von Instances ist veraltet. AutoScaling	15. Juni 2020

Support für Konnektivität durch AWS PrivateLink	Sie können eine private Verbindung zwischen Ihrer VPC und EC2 Image Builder herstellen, indem Sie einen VPC-Schnittstellen-Endpunkt erstellen. Schnittstellenendpunkte basieren auf einer Technologie AWS PrivateLink, mit der Sie privat auf Image Builder Builder-APIs zugreifen können, ohne dass ein Internet-Gateway, ein NAT-Gerät, eine VPN-Verbindung oder eine AWS Direct Connect-Verbindung erforderlich ist. Instances in Ihrer VPC benötigen keine öffentlichen IP-Adressen, um mit Image Builder Builder-APIs zu kommunizieren. Der Datenverkehr zwischen Ihrer VPC und Image Builder verlässt das Amazon-Netzwerk nicht.	10. Juni 2020
Neue STIG-Versionen	Aktualisierte STIG-Versionen und angewandte STIGS.	23. Januar 2020
Fehlersuche	Allgemeine Problemlösungsszenarien hinzugefügt.	22. Januar 2020
STIG-Komponenten	Sie können STIG-konforme Bilder mit AWSTOE STIG-Komponenten erstellen.	22. Januar 2020

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.