aws

Leitfaden

Amazon EventBridge

Amazon EventBridge: Leitfaden

Table of Contents

Was ist Amazon EventBridge?

EventBridge ist ein Serverless-Service, der mithilfe von Ereignissen Anwendungskomponenten miteinander verbindet, sodass Sie leichter skalierbare, ereignisgesteuerte Anwendungen erstellen können. Bei der ereignisgesteuerten Architektur werden lose gekoppelte Softwaresysteme entwickelt, die zusammenarbeiten, indem sie Ereignisse senden und darauf reagieren. Eine ereignisgesteuerte Architektur kann Ihnen helfen, die Agilität zu erhöhen und zuverlässige, skalierbare Anwendungen zu entwickeln.

Verwenden Sie EventBridge, um Ereignisse aus Quellen wie selbst entwickelten Anwendungen, AWS-Services und Drittanbietersoftware an Konsumentenanwendungen in Ihrem Unternehmen weiterzuleiten. EventBridge bietet einfache und konsistente Methoden zum Erfassen, Filtern, Transformieren und Bereitstellen von Ereignissen, sodass Sie schnell Anwendungen erstellen können.

Das folgende Video stellt eine kurze Einführung in die Features von Amazon EventBridge bereit:

EventBridge bietet zwei Möglichkeiten zur Verarbeitung von Ereignissen: Event Buses und Pipes.

- Event Buses sind Router, die Ereignisse empfangen und sie an null oder mehr Ziele weiterleiten. Event Buses eignen sich hervorragend für die Weiterleitung von Ereignissen aus vielen Quellen an viele Ziele, wobei sie optional transformiert werden können, bevor sie an ein Ziel gesendet werden.

  Das folgende Video stellt einen allgemeinen Überblick über Event Buses bereit:

- Pipes EventBridge Pipes ist für Punkt-zu-Punkt-Integrationen vorgesehen. Jede Pipe empfängt Ereignisse von einer einzigen Quelle zur Verarbeitung und Übertragung an ein einzelnes Ziel. Pipes bieten auch Unterstützung für erweiterte Transformationen und die Anreicherung von Ereignissen vor der Übertragung an ein Ziel.

Pipes und Event Buses werden häufig zusammen eingesetzt. Ein häufiger Anwendungsfall ist die Erstellung einer Pipe mit einem Event Bus als Ziel. Die Pipe sendet Ereignisse an den Event Bus, der diese Ereignisse dann an mehrere Ziele weiterleitet. Sie könnten beispielsweise eine Pipe mit einem DynamoDB-Stream für eine Quelle und einem Event Bus als Ziel erstellen. Die Pipe empfängt Ereignisse aus dem DynamoDB-Stream und sendet sie an den Event Bus, der sie dann gemäß den Regeln, die Sie für den Event Bus angegeben haben, an mehrere Ziele weiterleitet.

EventBridge ist die Weiterentwicklung von Amazon CloudWatch Events.

EventBridge wurde früher als Amazon CloudWatch Events bezeichnet. Der Standard-Event-Bus und die Regeln, die Sie in CloudWatch Events erstellt haben, werden auch in der EventBridge-Konsole angezeigt. EventBridge verwendet dieselbe CloudWatch-Events-API, sodass Ihr Code, der die CloudWatch-Events-API verwendet, unverändert bleibt.

EventBridge baut auf den Funktionen von CloudWatch Events mit Features wie Partnerereignissen, der Schemaregistrierung und EventBridge Pipes auf. Neue Features, die zu EventBridge hinzugefügt wurden, werden nicht zu CloudWatch Events hinzugefügt. Weitere Informationen finden Sie unter ???.

Alle Features, die Sie von CloudWatch Events gewohnt sind, sind auch in EventBridge verfügbar, darunter:

- ???
- ???
- ???
- ???

Zu den EventBridge-Features, die auf den Funktionen von Ereignissen aufbauen und diese erweitern, gehören:

- ???
- ???
- ???
- ???

EventBridge Einrichtung und Voraussetzungen für Amazon

Um Amazon nutzen zu können EventBridge, benötigen Sie ein AWS Konto. Mit Ihrem Konto können Sie Dienste wie Amazon EC2 verwenden, um Ereignisse zu generieren, die Sie in der EventBridge Konsole sehen können. Sie können AWS Command Line Interface (AWS CLI) auch so installieren und konfigurieren, dass Ereignisse über eine Befehlszeilenschnittstelle angezeigt werden.

Themen

Melden Sie sich an für ein AWS-Konto

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

Um sich für eine anzumelden AWS-Konto

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

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

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

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

Erstellen Sie einen Benutzer mit Administratorzugriff

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

Sichern Sie Ihre Root-Benutzer des AWS-Kontos

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

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

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

Erstellen Sie einen Benutzer mit Administratorzugriff

1. Aktivieren Sie das IAM Identity Center.

   Anweisungen finden Sie unter Aktivieren AWS IAM Identity Center im AWS IAM Identity Center Benutzerhandbuch.
2. Gewähren Sie einem Benutzer in IAM Identity Center Administratorzugriff.

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

Melden Sie sich als Benutzer mit Administratorzugriff an

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

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

Weisen Sie weiteren Benutzern Zugriff zu

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

   Anweisungen finden Sie im Benutzerhandbuch unter Einen Berechtigungssatz erstellen.AWS IAM Identity Center
2. Weisen Sie Benutzer einer Gruppe zu und weisen Sie der Gruppe dann Single Sign-On-Zugriff zu.

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

Melden Sie sich bei der EventBridge Amazon-Konsole an

Um sich bei der EventBridge Amazon-Konsole anzumelden

- Melden Sie sich bei der an AWS Management Console und öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.

Kontoanmeldeinformationen

Sie können zwar Ihre Root-Benutzeranmeldedaten für den Zugriff verwenden EventBridge, wir empfehlen jedoch, stattdessen ein AWS Identity and Access Management (IAM-) Konto zu verwenden. Wenn Sie für den Zugriff ein IAM-Konto verwenden EventBridge, benötigen Sie die folgenden Berechtigungen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "events:*"
      ],
```

```
      "Effect": "Allow",
      "Resource": "arn:aws:events:*:*:*"
    },
    {
      "Action": [
        "iam:PassRole"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:PassedToService": "events.amazonaws.com"
        }
      }
    }
  ]
}
```

Weitere Informationen finden Sie unter Authentifizierung.

Richten Sie das ein AWS Command Line Interface

Sie können das verwenden AWS CLI , um EventBridge Operationen auszuführen.

Informationen zur Installation und Konfiguration von finden Sie unter Getting Setup with the AWS Command Line Interface im AWS Command Line Interface Benutzerhandbuch. AWS CLI

Regionale Endpunkte

Sie müssen die regionalen Standardendpunkte aktivieren, um sie verwenden zu können EventBridge. Weitere Informationen finden Sie unter Aktivierung und Deaktivierung AWS STS in einer AWS Region im IAM-Benutzerhandbuch.

Erste Schritte mit Amazon EventBridge

Die Grundlage von EventBridge ist die Erstellung von Regeln, die Ereignisse an ein Ziel weiterleiten. In diesem Abschnitt erstellen Sie eine Basisregel. Tutorials zu spezifischen Szenarien und Zielen finden Sie unter Amazon-EventBridge-Tutorials.

Eine Regel in Amazon erstellen EventBridge

Um eine Regel für Ereignisse zu erstellen, geben Sie eine Aktion an, die ausgeführt werden soll, EventBridge wenn ein Ereignis eintrifft, das dem Ereignismuster in der Regel entspricht. Wenn ein Ereignis übereinstimmt, wird das Ereignis an das angegebene Ziel EventBridge gesendet und die in der Regel definierte Aktion ausgelöst.

Wenn ein AWS Dienst in Ihrem AWS Konto ein Ereignis ausgibt, wird dieser immer an den Standard-Event-Bus für Ihr Konto weitergeleitet. Um eine Regel zu schreiben, die Ereignissen von AWS Diensten in Ihrem Konto entspricht, müssen Sie sie mit dem Standard-Event-Bus verknüpfen.

Um eine Regel für einen AWS Dienst zu erstellen

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen aus.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein.

   Eine Regel darf nicht denselben Namen wie eine andere Regel in derselben Region und auf demselben Event Bus haben.
5. Wählen Sie als Event bus (Event Bus) den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie AWS -Standard-Event-Bus aus. Wenn ein AWS -Service in Ihrem Konto ein Ereignis ausgibt, wird es stets an den Standard-Event-Bus Ihres Kontos weitergeleitet.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Weiter aus.
8. Als Event source (Ereignisquelle) wählen Sie AWS -Services aus.
9. (Optional) Wählen Sie für Beispielereignisse den Ereignistyp aus.

10. Gehen Sie für Ereignismuster wie folgt vor:

    - Wenn Sie eine Vorlage zum Erstellen Ihres Ereignismusters verwenden möchten, wählen Sie Ereignismusterformular und dann die gewünschten Einstellungen für Ereignisquelle sowie Ereignistyp aus. Wenn Sie „Alle Ereignisse“ als Ereignistyp wählen, entsprechen alle von diesem AWS Service ausgelösten Ereignisse der Regel.

      Um die Vorlage anzupassen, wählen Sie Custom pattern (JSON editor) (Benutzerdefiniertes Muster (JSON-Editor)) und nehmen Sie die erforderlichen Änderungen vor.
    - Wenn Sie ein benutzerdefiniertes Ereignismuster verwenden möchten, wählen Sie Custom pattern (JSON editor) (Benutzerdefiniertes Muster (JSON-Editor)) und erstellen Sie Ihr Ereignismuster.

11. Wählen Sie Weiter aus.
12. Bei Target types (Zieltypen) wählen Sie AWS -Service aus.
13. Wählen Sie unter Ziel auswählen den AWS Dienst aus, an den Sie Informationen senden möchten, wenn ein Ereignis EventBridge erkannt wird, das dem Ereignismuster entspricht.
14. Die angezeigten Felder variieren je nach ausgewähltem Service. Geben Sie nach Bedarf Informationen ein, die für diesen Zieltyp spezifisch sind.
15. Für viele Zieltypen sind EventBridge Berechtigungen erforderlich, um Ereignisse an das Ziel zu senden. In diesen Fällen EventBridge kann die IAM-Rolle erstellt werden, die für die Ausführung Ihrer Regel erforderlich ist. Führen Sie eine der folgenden Aktionen aus:

    - Um automatisch eine IAM-Rolle zu erstellen, wählen Sie Create a new role for this specific resource (Eine neue Rolle für diese spezifische Ressource erstellen).
    - Wenn Sie eine zuvor erstellte IAM-Rolle verwenden möchten, wählen Sie Vorhandene Rolle verwenden und dann die vorhandene Rolle aus der Dropdown-Liste aus.

16. (Optional) Gehen Sie unter Additional settings (Weitere Einstellungen) wie folgt vor:

    a. Geben Sie für Maximum age of event (Maximales Alter des Ereignisses) einen Wert zwischen einer Minute (00:01) und 24 Stunden (24:00) ein.

    b. Geben Sie für Wiederholungsversuche eine Zahl zwischen 0 und 185 ein.

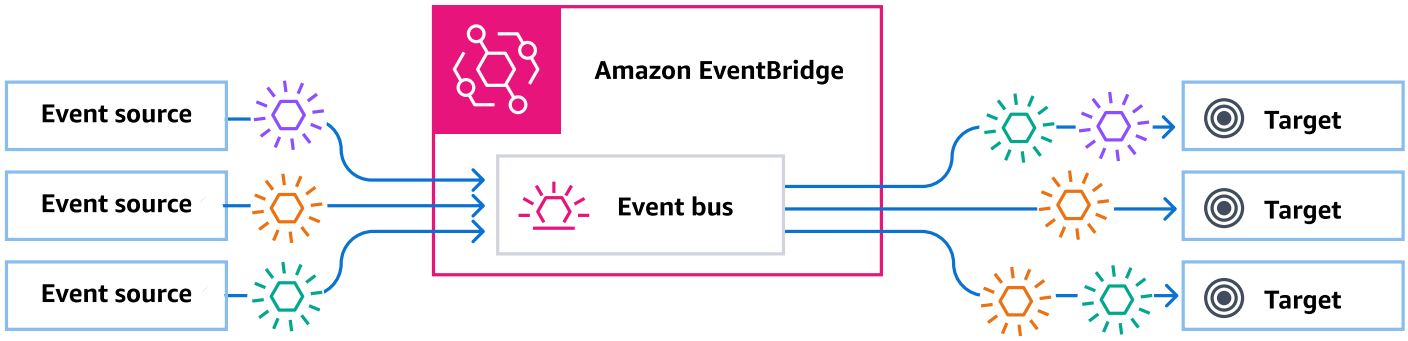
    c. Wählen Sie für Warteschlange für unzustellbare Briefe aus, ob Sie eine standardmäßige Amazon SQS SQS-Warteschlange als Warteschlange für unzustellbare Briefe verwenden möchten. EventBridge sendet Ereignisse, die dieser Regel entsprechen, an die Warteschlange für unzustellbare Briefe, wenn sie nicht erfolgreich an das Ziel zugestellt wurden. Führen Sie eine der folgenden Aktionen aus:

- Klicken Sie auf Keine, um keine Warteschlange für unzustellbare Nachrichten zu verwenden.
- Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange im aktuellen AWS - Konto als Warteschlange für unzustellbare Nachrichten und wählen Sie dann die Warteschlange aus der Dropdown-Liste aus.
- Wählen Sie Wählen Sie eine Amazon SQS SQS-Warteschlange in einem anderen AWS Konto als Warteschlange für unzustellbare Briefe aus und geben Sie dann den ARN der Warteschlange ein, die Sie verwenden möchten. Sie müssen der Warteschlange eine ressourcenbasierte Richtlinie hinzufügen, die das Senden von Nachrichten an die EventBridge Warteschlange ermöglicht. Weitere Informationen finden Sie unter Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten.

17. (Optional) Wählen Sie Add another target (Weiteres Ziel hinzufügen) aus, um ein weiteres Ziel für diese Regel hinzuzufügen.
18. Wählen Sie Weiter aus.
19. (Optional) Geben Sie ein oder mehrere Tags für die Regel ein. Weitere Informationen finden Sie unter Amazon- EventBridge Tags.
20. Wählen Sie Weiter.
21. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Amazon EventBridge Event Bus

Ein Event Bus ist ein Router, der Ereignisse empfängt und sie an null oder mehr Ziele weiterleitet. Event Buses eignen sich hervorragend für die Weiterleitung von Ereignissen aus vielen Quellen an viele Ziele, wobei sie optional transformiert werden können, bevor sie an ein Ziel gesendet werden.



Regeln, die dem Event Bus zugeordnet sind, werten die eintreffenden Ereignisse aus. Bei jeder Regel wird geprüft, ob ein Ereignis dem Muster der Regel entspricht. Wenn das Ereignis übereinstimmt, wird das Ereignis EventBridge gesendet

Sie verknüpfen eine Regel mit einem bestimmten Event Bus, sodass die Regel nur für Ereignisse gilt, die von diesem Event Bus empfangen werden.

> **Note**
>
> Sie können Ereignisse auch mithilfe von EventBridge Pipes verarbeiten. EventBridge Pipes ist für point-to-point Integrationen vorgesehen. Jede Pipe empfängt Ereignisse aus einer einzigen Quelle zur Verarbeitung und Übertragung an ein einziges Ziel. Pipes bieten auch Unterstützung für erweiterte Transformationen und die Anreicherung von Ereignissen vor der Übertragung an ein Ziel. Weitere Informationen finden Sie unter ???.

Themen

- Funktionsweise von Event Buses
- Amazon EventBridge Event Bus-Konzepte
- Einen EventBridge Amazon-Eventbus erstellen
- Aktualisierung eines EventBridge Amazon-Eventbusses

- Löschen eines EventBridge Amazon-Event-Busses
- Berechtigungen für Amazon EventBridge-Event-Buses
- Generieren einer AWS CloudFormation-Vorlage aus einem Amazon-EventBridge-Event-Bus

Funktionsweise von Event Buses

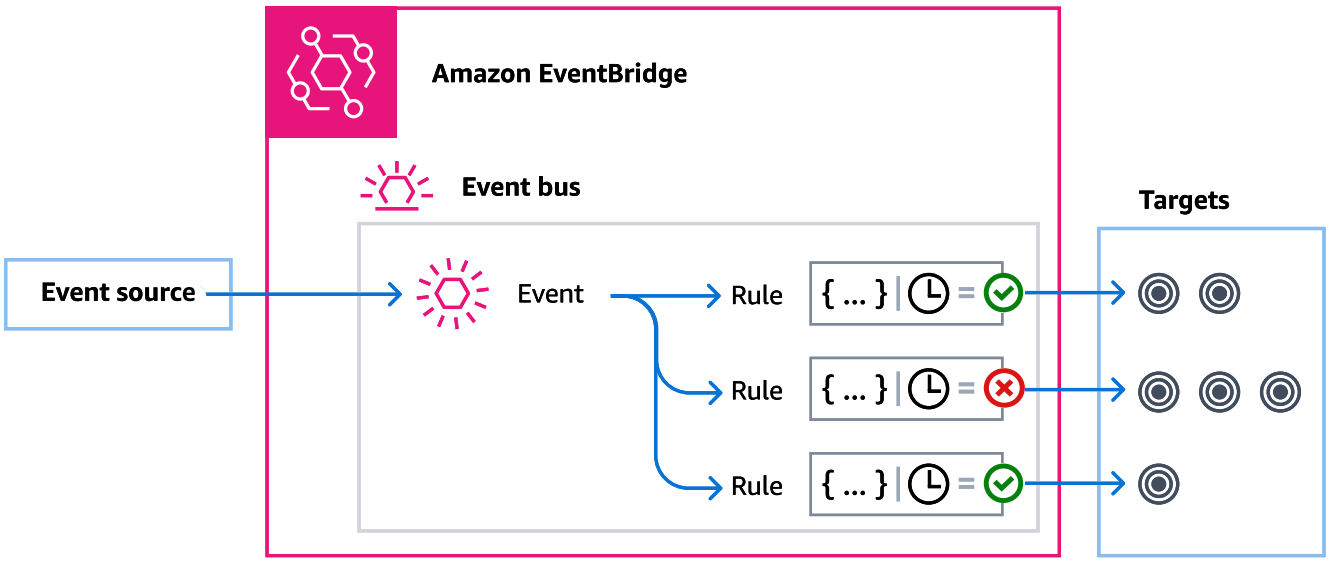
Mit Event Buses können Sie Ereignisse von mehreren Quellen an mehrere Ziele weiterleiten.

Grundsätzlich funktioniert das wie folgt:

1. Eine Ereignisquelle, bei der es sich um einen AWS Dienst, Ihre eigene benutzerdefinierte Anwendung oder einen SaaS-Anbieter handeln kann, sendet ein Ereignis an einen Ereignisbus.
2. EventBridge wertet das Ereignis dann anhand jeder Regel aus, die für diesen Ereignisbus definiert wurde.

   Für jedes Ereignis, das einer Regel entspricht, EventBridge wird das Ereignis dann an die für diese Regel angegebenen Ziele gesendet. Optional können Sie als Teil der Regel auch angeben, wie das Ereignis transformiert werden EventBridge soll, bevor es an die Ziele gesendet wird.

   Ein Ereignis kann mehreren Regeln entsprechen, und jede Regel kann bis zu fünf Ziele angeben. (Ein Ereignis entspricht möglicherweise keiner Regel. In diesem Fall wird EventBridge keine Aktion ausgeführt.)



Stellen Sie sich ein Beispiel vor, bei dem der EventBridge Standardereignisbus verwendet wird, der automatisch Ereignisse von AWS Diensten empfängt:

1. Sie erstellen eine Regel für den Standard-Event-Bus für das `EC2 Instance State-change Notification`-Ereignis:
   - Sie geben an, dass die Regel Ereignissen entspricht, zu denen eine Amazon-EC2-Instance ihren `state` zu `running` geändert hat.

     Dazu geben Sie JSON an, das die Attribute und Werte definiert, denen ein Ereignis entsprechen muss, um die Regel auszulösen. Dies wird als Ereignismuster bezeichnet.

     ```
     {
       "source": ["aws.ec2"],
       "detail-type": ["EC2 Instance State-change Notification"],
         "detail": {
           "state": ["running"]
       }
     }
     ```

   - Sie geben das Ziel der Regel als eine bestimmte Lambda-Funktion an.
2. Immer wenn eine Amazon-EC2-Instance den Status ändert, sendet Amazon EC2 (die Ereignisquelle) dieses Ereignis automatisch an den Standard-Event-Bus.
3. EventBridge wertet alle an den Standard-Event-Bus gesendeten Ereignisse anhand der von Ihnen erstellten Regel aus.

   Wenn das Ereignis Ihrer Regel entspricht (d. h. wenn es sich bei dem Ereignis um eine Amazon EC2 EC2-Instance handelt, zu der der Status geändert wurde`running`), wird das Ereignis an das angegebene Ziel EventBridge gesendet. In diesem Fall ist das die Lambda-Funktion.

Das folgende Video beschreibt, was Event Buses sind und was sie tun: Was sind Event Buses

Das folgende Video behandelt die verschiedenen Event Buses und wann sie eingesetzt werden sollten: Der Unterschied zwischen Event Buses

Amazon EventBridge Event Bus-Konzepte

Im Folgenden werden die Hauptkomponenten einer ereignisgesteuerten Architektur, die auf Event Buses basiert, genauer betrachtet.

Ereignisbusse

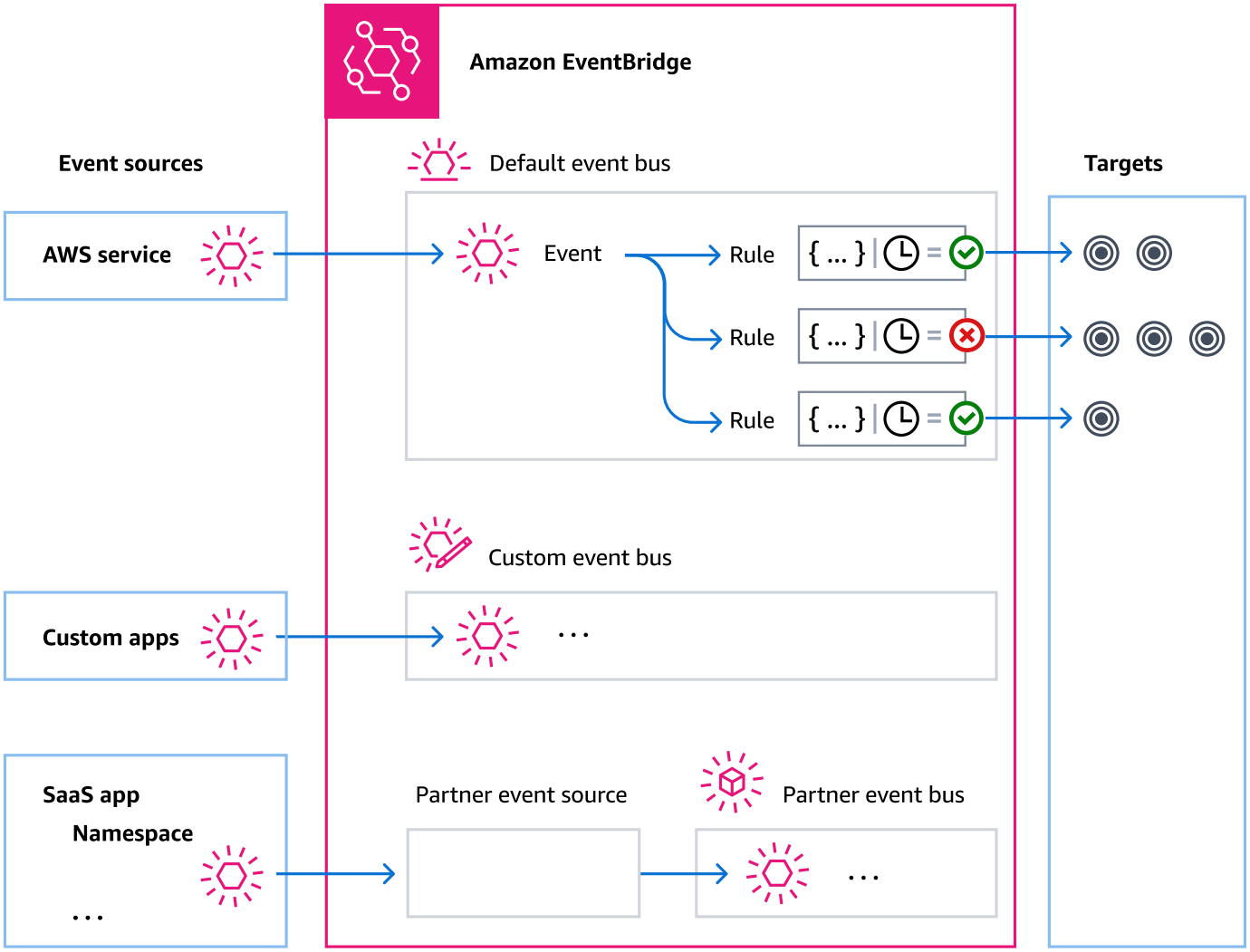
Ein Event Bus ist ein Router, der Ereignisse empfängt und sie an null oder mehr Ziele weiterleitet. Verwenden Sie einen Event Bus, wenn Sie Ereignisse aus vielen Quellen an viele Ziele weiterleiten müssen, wobei sie optional transformiert werden können, bevor sie an ein Ziel gesendet werden.

Ihr Konto enthält einen Standard-Event-Bus, der automatisch Ereignisse von AWS Services empfängt. Sie können auch:

- Erstellen Sie zusätzliche Event Buses, sogenannte benutzerdefinierte Event Buses, und geben Sie an, welche Ereignisse sie empfangen.
- Erstellen Sie Partner-Event-Buses, die Ereignisse von SaaS-Partnern empfangen.

Zu den häufigsten Anwendungsfällen für Event Buses gehören:

- Verwenden eines Event Bus als Broker zwischen verschiedenen Workloads, Services oder Systemen.
- Verwenden mehrerer Event Buses in Ihren Anwendungen, um den Ereignisdatenverkehr aufzuteilen. Zum Beispiel das Erstellen eines Bus zur Verarbeitung von Ereignissen, die personenbezogene Daten (PII) enthalten, und eines weiteren Bus für Ereignisse, bei denen dies nicht der Fall ist.
- Aggregieren von Ereignissen durch Senden von Ereignissen von mehreren Event Buses an einen zentralisierten Event Bus. Dieser zentralisierte Bus kann sich im selben Konto wie die anderen Buses, aber auch in einem anderen Konto oder in einer anderen Region befinden.



Ereignisse

Im einfachsten EventBridge Fall ist ein Ereignis ein JSON-Objekt, das an einen Event-Bus oder eine Pipe gesendet wird.

Im Kontext einer ereignisgesteuerten Architektur (EDA) stellt ein Ereignis häufig einen Indikator für eine Änderung einer Ressource oder Umgebung dar.

Weitere Informationen finden Sie unter ???.

Ereignisquellen

EventBridge kann Ereignisse aus Ereignisquellen empfangen, darunter:

- AWS Dienste
- Benutzerdefinierte Anwendungen
- Software-as-a-Service (SaaS)-Partner

Regeln

Eine Regel empfängt eintreffende Ereignisse und sendet diese je nach Bedarf zur Verarbeitung an Ziele. Sie können angeben, wie jede Regel ihr(e) Ziel(e) aufruft, und zwar auf der Grundlage von:

- Ein Ereignismuster, das einen oder mehrere Filter zum Abgleichen von Ereignissen enthält. Ereignismuster können Filter enthalten, die einen Abgleich auf Folgendes durchführen:
  - Ereignismetadaten – Daten über das Ereignis, z. B. die Ereignisquelle oder das Konto oder die Region, aus der das Ereignis stammt.
  - Ereignisdaten – Die Eigenschaften des Ereignisses selbst. Diese Eigenschaften variieren je nach Ereignis.
  - Ereignisinhalt – Die tatsächlichen Eigenschaftswerte der Ereignisdaten.
- Ein Zeitplan zum Aufrufen eines oder mehrerer Ziele in regelmäßigen Abständen.

  Sie können eine geplante Regel innerhalb EventBridge oder mithilfe des EventBridge Schedulers angeben.

  > **Note**
  >
  > EventBridge bietet Amazon EventBridge Scheduler, einen serverlosen Scheduler, mit dem Sie Aufgaben von einem zentralen, verwalteten Service aus erstellen, ausführen und verwalten können. EventBridge Scheduler ist hochgradig anpassbar und bietet eine verbesserte Skalierbarkeit im Vergleich zu EventBridge geplanten Regeln sowie ein breiteres Spektrum an API-Zieloperationen und -diensten. AWS
  > Wir empfehlen, dass Sie EventBridge Scheduler verwenden, um Ziele nach einem Zeitplan aufzurufen. Weitere Informationen finden Sie unter ???.

Jede Regel ist für einen bestimmten Event Bus definiert und gilt nur für Ereignisse in diesem Event Bus.

Eine einzelne Regel kann ein Ereignis an bis zu fünf Ziele senden.

Standardmäßig können Sie bis zu 300 Regeln pro Event Bus konfigurieren. Dieses Kontingent kann in der Service-Quotas-Konsole auf Tausende von Regeln erhöht werden. Da das Regellimit für jeden Bus gilt, können Sie in Ihrem Konto zusätzliche benutzerdefinierte Event Buses erstellen, wenn Sie noch mehr Regeln benötigen.

Sie können anpassen, wie Ereignisse in Ihrem Konto empfangen werden, indem Sie Event Buses mit unterschiedlichen Berechtigungen für verschiedene Services einrichten.

Um die Struktur oder das Datum eines Ereignisses anzupassen, bevor es an ein Ziel EventBridge übergeben wird, verwenden Sie den Eingangstransformator, um die Informationen zu bearbeiten, bevor sie an das Ziel weitergeleitet werden.

Weitere Informationen finden Sie unter ???.

Targets (Ziele)

Ein Ziel ist eine Ressource oder ein Endpunkt, an den ein Ereignis EventBridge gesendet wird, wenn das Ereignis dem für eine Regel definierten Ereignismuster entspricht.

Ein Ziel kann mehrere Ereignisse von mehreren Event Buses empfangen.

Weitere Informationen finden Sie unter ???.

Erweiterte Funktionen für Event Buses

EventBridge umfasst die folgenden Funktionen, die Sie bei der Entwicklung, Verwaltung und Verwendung von Event-Bussen unterstützen.

Verwenden von API-Zielen zur Aktivierung von REST-API-Aufrufen zwischen Services

EventBridge API-Ziele sind HTTP-Endpunkte, die Sie als Ziel einer Regel festlegen können, genauso wie Sie Ereignisdaten an einen AWS Dienst oder eine Ressource senden würden. Durch die Verwendung von API-Zielen können Sie API-Aufrufe verwenden, um Ereignisse zwischen AWS -Services, integrierten SaaS-Anwendungen und Ihren Anwendungen außerhalb von AWS weiterzuleiten. Wenn Sie ein API-Ziel erstellen, geben Sie eine dafür zu verwendende Verbindung an. Jede Verbindung enthält Details zum Autorisierungstyp und zu den Parametern, die zur Autorisierung mit dem API-Zielendpunkt verwendet werden.

Archivieren und Wiederholen von Ereignissen zur Unterstützung der Entwicklung und Notfallwiederherstellung

Sie können Ereignisse archivieren oder speichern und sie dann zu einem späteren Zeitpunkt aus dem Archiv wiederholen. Archivieren ist nützlich für:

- Testen einer Anwendung, weil Sie einen Ereignisspeicher zur Verfügung haben, den Sie verwenden können, anstatt auf neue Ereignisse warten zu müssen.
- Hydratisieren eines neuen Service, wenn er zum ersten Mal online ist.
- Erhöhen der Lebensdauer Ihrer ereignisgesteuerten Anwendungen.

Verwenden der Schemaregistrierung, um die Erstellung von Ereignismustern zu beschleunigen

Wenn Sie serverlose Anwendungen erstellen, die Folgendes verwenden EventBridge, kann es hilfreich sein, die Struktur typischer Ereignisse zu kennen, ohne das Ereignis selbst generieren zu müssen. Die Ereignisstruktur wird in Schemas beschrieben, die für alle Ereignisse verfügbar sind, die von AWS Services on generiert werden. EventBridge

Für Ereignisse, die nicht von AWS Diensten stammen, können Sie:

- Benutzerdefinierte Schemata erstellen oder hochladen
- Verwenden Sie Schema Discovery, um EventBridge automatisch Schemas für Ereignisse zu erstellen, die an den Event-Bus gesendet werden.

Sobald Sie ein Schema für ein Ereignis haben, können Sie Codebindungen für gängige Programmiersprachen herunterladen.

Verwalten der Ressourcen und des Zugriffs mit Richtlinien

Um AWS Ressourcen zu organisieren oder Kosten nachzuverfolgen EventBridge, können Sie AWS Ressourcen ein benutzerdefiniertes Label oder Tag zuweisen. Mithilfe von Tag-basierten Richtlinien können Sie steuern, was Ressourcen innerhalb von EventBridge Ressourcen tun dürfen und welche nicht.

EventBridge Unterstützt neben tagbasierten Richtlinien auch identitäts- und ressourcenbasierte Richtlinien zur Zugriffskontrolle. EventBridge Verwenden Sie identitätsbasierte Richtlinien, um die Berechtigungen einer Gruppe, Rolle oder eines Benutzers zu kontrollieren. Verwenden Sie ressourcenbasierte Richtlinien, um jeder Ressource spezifische Berechtigungen zu erteilen, z. B. einer Lambda-Funktion oder einem Amazon-SNS-Thema.

Einen EventBridge Amazon-Eventbus erstellen

Sie können einen benutzerdefinierten Event Bus erstellen, um Ereignisse aus Ihren Anwendungen zu empfangen. Ihre Anwendungen können auch Ereignisse an den Standard-Event-Bus senden. Wenn Sie einen Event Bus erstellen, können Sie eine ressourcenbasierte Richtlinie anhängen, um anderen Konten Berechtigungen zu gewähren. Dann können andere Konten Ereignisse an den Event Bus im aktuellen Konto senden.

Das folgende Video zeigt die Erstellung von Event Buses: Erstellen eines Event Bus

So erstellen Sie einen benutzerdefinierten Ereignisbus

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Event Buses (Ereignisbusse) aus.
3. Wählen Sie Create event bus (Ereignisbus erstellen) aus.
4. Geben Sie einen Namen für den neuen Ereignisbus ein.
5. Wählen Sie das KMS key für EventBridge die Verschlüsselung der auf dem Event-Bus gespeicherten Ereignisdaten aus.

> **Note**
>
> Archive und Schemaerkennung werden für Ereignisbusse, die mit a Kundenverwalteter Schlüssel verschlüsselt wurden, nicht unterstützt. Um Archive oder die Schemaerkennung auf einem Event-Bus zu aktivieren, wählen Sie die Verwendung von AWS-eigener Schlüssel. Weitere Informationen finden Sie unter ???.

- Wählen Sie Verwenden AWS-eigener Schlüssel für EventBridge , um die Daten mit einem AWS-eigener Schlüssel zu verschlüsseln.

  Dies AWS-eigener Schlüssel ist eine KMS key , die mehrere Konten EventBridge besitzt und verwaltet, sodass sie in mehreren AWS Konten verwendet werden können. Generell gilt: Sofern Sie nicht verpflichtet sind, den Verschlüsselungsschlüssel, der Ihre Ressourcen schützt, zu überprüfen oder zu kontrollieren, AWS-eigener Schlüssel ist an eine gute Wahl.

  Dies ist die Standardeinstellung.

- Wählen Sie Verwenden Kundenverwalteter Schlüssel für EventBridge , um die Daten mit dem zu verschlüsseln Kundenverwalteter Schlüssel , den Sie angeben oder erstellen.

  Kundenverwaltete Schlüssel befinden sich KMS keys in Ihrem AWS Konto, das Sie erstellen, besitzen und verwalten. Sie haben die volle Kontrolle über diese KMS keys.

  a. Geben Sie ein vorhandenes an Kundenverwalteter Schlüssel, oder wählen Sie Neues erstellen KMS key.

     EventBridge zeigt den Schlüsselstatus und alle Schlüsselaliase an, die dem angegebenen Kundenverwalteter Schlüssel Schlüssel zugeordnet wurden.

  b. Wählen Sie die Amazon SQS SQS-Warteschlange, die als Dead-Letter-Warteschlange (DLQ) für diesen Event-Bus verwendet werden soll, falls vorhanden.

     EventBridge sendet Ereignisse, die nicht erfolgreich verschlüsselt wurden, an den DLQ, sofern konfiguriert, sodass Sie sie später verarbeiten können.

6. Konfigurieren Sie optionale Event-Bus-Funktionen:

- Geben Sie eine ressourcenbasierte Richtlinie an, indem Sie einen der folgenden Schritte ausführen:
  - Geben Sie die Richtlinie ein, die die zu erteilenden Berechtigungen für den Event Bus beinhaltet. Sie können eine Richtlinie aus einer anderen Quelle einfügen oder den JSON-Code für die Richtlinie eingeben. Sie können eine der Beispielrichtlinien verwenden und sie für Ihre Umgebung ändern.
  - Um eine Vorlage für die Richtlinie zu verwenden, wählen Sie Vorlage laden aus. Ändern Sie die Richtlinie entsprechend Ihrer Umgebung und fügen Sie zusätzliche Aktionen hinzu, für deren Verwendung Sie den Prinzipal in der Richtlinie autorisieren.

  Weitere Informationen zum Erteilen von Berechtigungen für einen Event-Bus mithilfe ressourcenbasierter Richtlinien finden Sie unter ???

- Aktivieren Sie ein Archiv (optional)

  Sie können ein Archiv mit Ereignissen erstellen, sodass Sie sie zu einem späteren Zeitpunkt problemlos erneut abspielen können. Beispielsweise möchten Sie möglicherweise Ereignisse wiederholen, um Fehler zu beheben oder neue Funktionen in Ihrer Anwendung zu validieren. Weitere Informationen finden Sie unter ???.

  a. Wählen Sie unter Archive die Option Aktiviert aus.

b. Geben Sie einen Namen und eine Beschreibung für das Archiv an.

> **Note**
>
> Archive und Schemaerkennung werden für Ereignisbusse, die mit a verschlüsselt wurden, nicht unterstützt Kundenverwalteter Schlüssel. Um Archive oder die Schemaerkennung auf einem Event-Bus zu aktivieren, wählen Sie die Verwendung von AWS-eigener Schlüssel. Weitere Informationen finden Sie unter ???.

- Aktivieren Sie die Schemaerkennung (optional)

  Aktivieren Sie die Schemaerkennung, um Schemas EventBridge automatisch direkt aus Ereignissen abzuleiten, die auf diesem Ereignisbus ausgeführt werden. Weitere Informationen finden Sie unter ???.

  a. Wählen Sie unter Schemaerkennung die Option Aktiviert aus.

  > **Note**
  >
  > Archive und Schemaerkennung werden für Ereignisbusse, die mit a verschlüsselt wurden, nicht unterstützt Kundenverwalteter Schlüssel. Um Archive oder die Schemaerkennung auf einem Event-Bus zu aktivieren, wählen Sie die Verwendung von AWS-eigener Schlüssel. Weitere Informationen finden Sie unter ???.

- Geben Sie Tags an (optional)

  Ein Tag ist eine benutzerdefinierte Attributbezeichnung, die Sie einer AWS Ressource zuweisen. Verwenden Sie Tags, um Ihre AWS Ressourcen zu identifizieren und zu organisieren. Viele AWS Dienste unterstützen Tagging, sodass Sie Ressourcen aus verschiedenen Diensten dasselbe Tag zuweisen können, um anzuzeigen, dass die Ressourcen miteinander verknüpft sind. Weitere Informationen finden Sie unter ???.

  a. Wählen Sie unter Tags die Option Neuen Tag hinzufügen aus.

  b. Geben Sie einen Schlüssel und optional einen Wert für das neue Tag an.

7. Wählen Sie Create (Erstellen) aus.

Aktualisierung eines EventBridge Amazon-Eventbusses

Sie können die Konfiguration von Event-Bussen aktualisieren, nachdem Sie sie erstellt haben. Dazu gehört auch der Standard-Event-Bus, der automatisch in Ihrem Konto EventBridge erstellt wird.

Aktualisierung des für die Verschlüsselung KMS key verwendeten

> **Note**
>
> Archive und Schemaerkennung werden für Ereignisbusse, die mit a verschlüsselt wurden, nicht unterstützt Kundenverwalteter Schlüssel. Um Archive oder die Schemaerkennung auf einem Event-Bus zu aktivieren, wählen Sie die Verwendung von AWS-eigener Schlüssel. Weitere Informationen finden Sie unter ???.

Um den für die Verschlüsselung im Ruhezustand auf einem Event-Bus KMS key verwendeten Wert mithilfe der EventBridge Konsole zu ändern

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Event Buses (Ereignisbusse) aus.
3. Wählen Sie den Event-Bus aus, den Sie aktualisieren möchten.
4. Wählen Sie auf der Seite mit den Details zum Event-Bus die Registerkarte Verschlüsselung aus.
5. Wählen Sie die KMS key für EventBridge die Verschlüsselung der auf dem Event-Bus gespeicherten Ereignisdaten zu verwendende Form aus:
   - Wählen Sie Verwenden AWS-eigener Schlüssel für EventBridge , um die Daten mit einem zu verschlüsseln. AWS-eigener Schlüssel

     Dies AWS-eigener Schlüssel ist eine KMS key , die mehrere Konten EventBridge besitzt und für die Verwendung in mehreren AWS Konten verwaltet wird. Generell gilt: Sofern Sie nicht verpflichtet sind, den Verschlüsselungsschlüssel, der Ihre Ressourcen schützt, zu überprüfen oder zu kontrollieren, AWS-eigener Schlüssel ist an eine gute Wahl.

     Dies ist die Standardeinstellung.
   - Wählen Sie Verwenden Kundenverwalteter Schlüssel für EventBridge , um die Daten mit dem zu verschlüsseln Kundenverwalteter Schlüssel , den Sie angeben oder erstellen.

Kundenverwaltete Schlüssel befinden sich KMS keys in Ihrem AWS Konto, das Sie erstellen, besitzen und verwalten. Sie haben die volle Kontrolle über diese KMS keys.

a. Geben Sie ein vorhandenes an Kundenverwalteter Schlüssel, oder wählen Sie Neues erstellen KMS key.

   EventBridge zeigt den Schlüsselstatus und alle Schlüsselaliase an, die dem angegebenen Kundenverwalteter Schlüssel Schlüssel zugeordnet wurden.

b. Wählen Sie die Amazon SQS SQS-Warteschlange, die als Dead-Letter-Warteschlange (DLQ) für diesen Event-Bus verwendet werden soll, falls vorhanden.

   EventBridge sendet Ereignisse, die nicht erfolgreich verschlüsselt wurden, an den DLQ, sofern konfiguriert, sodass Sie sie später verarbeiten können.

Berechtigungen für einen Event-Bus werden aktualisiert

Sie können einem Event Bus zusätzliche Berechtigungen gewähren, indem Sie ihm eine ressourcenbasierte Richtlinie zuordnen. Ausführliche Anweisungen zum Aktualisieren der einem Event-Bus zugewiesenen Berechtigungen finden Sie unter Event-Bus-Berechtigungen verwalten.

Hinzufügen oder Entfernen von Archiven in Event-Bussen

Ein Archiv ermöglicht es Ihnen, Ereignisse aufzuzeichnen, sodass Sie sie zu einem späteren Zeitpunkt problemlos erneut abspielen können. Beispielsweise möchten Sie möglicherweise Ereignisse wiederholen, um Fehler zu beheben oder neue Funktionen in Ihrer Anwendung zu validieren. Weitere Informationen finden Sie unter EventBridge Archivieren und Wiedergeben.

> **Note**
>
> Archive und Schemaerkennung werden für Ereignisbusse, die mit a Kundenverwalteter Schlüssel verschlüsselt wurden, nicht unterstützt. Um Archive oder die Schemaerkennung auf einem Event-Bus zu aktivieren, wählen Sie die Verwendung von AWS-eigener Schlüssel. Weitere Informationen finden Sie unter ???.

Um mithilfe der EventBridge Konsole ein Archiv zu einem Event-Bus hinzuzufügen oder daraus zu entfernen

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Event Buses (Ereignisbusse) aus.
3. Wählen Sie den Event-Bus aus, den Sie aktualisieren möchten.
4. Wählen Sie auf der Detailseite des Event-Busses die Registerkarte Archiv aus.
5. Führen Sie eine der folgenden Aktionen aus:
   - Um ein Archiv hinzuzufügen:
     a. Wählen Sie Archiv erstellen.
     b. Geben Sie Attribute für das Archiv an.
     c. Wählen Sie Weiter aus.
     d. Wählen Sie das Ereignismuster, das auf Ereignisse für das Archiv angewendet werden soll.
     e. Wählen Sie Archiv erstellen.
   - Um ein Archiv zu löschen:
     a. Wählen Sie für das Tag, das Sie entfernen möchten, die Option Löschen.
     b. Geben Sie den Namen des Archivs ein und wählen Sie Löschen.

        Das Archiv wird dauerhaft gelöscht. Dieser Vorgang kann nicht rückgängig gemacht werden.

Um ein Archiv für einen Event-Bus zu erstellen oder zu löschen, verwenden Sie AWS CLI

- Um ein Archiv zu erstellen, verwenden Sie create-archive.

  Um ein Archiv dauerhaft zu löschen, verwenden Sie delete-archive.

Die Schemaerkennung in Event-Bussen starten oder beenden

Weitere Informationen zur Schemaerkennung finden Sie unter EventBridge Schemas.

> **Note**
>
> Archive und Schemaerkennung werden für Ereignisbusse, die mit a Kundenverwalteter Schlüssel verschlüsselt wurden, nicht unterstützt. Um Archive oder die Schemaerkennung auf einem Event-Bus zu aktivieren, wählen Sie die Verwendung von AWS-eigener Schlüssel. Weitere Informationen finden Sie unter ???.

Um die Schemaerkennung auf einem Event-Bus mithilfe der EventBridge Konsole zu starten oder zu beenden

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Event Buses (Ereignisbusse) aus.
3. Wählen Sie den Event-Bus aus, den Sie aktualisieren möchten.
4. Führen Sie eine der folgenden Aktionen aus:
   - Um die Schemaerkennung zu starten, wählen Sie Discovery starten.
   - Um die Schemaerkennung zu beenden, wählen Sie Discovery löschen.

Um die Schemaerkennung auf einem Event-Bus zu starten oder zu beenden, verwenden Sie AWS CLI

- Verwenden Sie create-discoverer, um die Schemaerkennung zu starten.

  Verwenden Sie delete-discoverer, um die Schemaerkennung zu beenden.

Hinzufügen oder Entfernen von Tags in Event-Bussen

Ein Tag ist eine benutzerdefinierte Attributbezeichnung, die Sie einer AWS Ressource AWS zuweisen oder zuweisen. Verwenden Sie Tags, um Ihre AWS Ressourcen zu identifizieren und zu organisieren. Weitere Informationen finden Sie unter EventBridge Tags.

So können Sie mithilfe der EventBridge Konsole Tags zu einem Event-Bus hinzufügen oder daraus entfernen

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Event Buses (Ereignisbusse) aus.

3. Wählen Sie den Event-Bus aus, den Sie aktualisieren möchten.
4. Wählen Sie auf der Seite mit den Details zum Event-Bus die Registerkarte „Tags“ und dann „Schlagworte verwalten“ aus.
5. Führen Sie eine der folgenden Aktionen aus:
   - So fügst du ein Tag hinzu:
     a. Wählen Sie Neues Tag hinzufügen aus.
     b. Geben Sie den Schlüssel und den Wert für das Tag an
     c. Wählen Sie Aktualisieren.
   - Um ein Tag zu entfernen:
     a. Wählen Sie für das Tag, das Sie entfernen möchten, die Option Entfernen aus.
     b. Wählen Sie Aktualisieren.

Um Stichwörter zu einem Event-Bus hinzuzufügen oder daraus zu entfernen, verwenden Sie AWS CLI

- Verwenden Sie Tag-Resource, um Tags hinzuzufügen.

  Verwenden Sie untag-resource, um Tags zu entfernen.

Den Standard-Event-Bus aktualisieren mit AWS CloudFormation

AWS CloudFormation ermöglicht es Ihnen, Ihre AWS Ressourcen konto- und regionsübergreifend auf zentralisierte und wiederholbare Weise zu konfigurieren und zu verwalten, indem die Infrastruktur als Code behandelt wird. CloudFormation ermöglicht dies, indem Sie Vorlagen erstellen können, die die Ressourcen definieren, die Sie bereitstellen und verwalten möchten.

Da EventBridge der Standardereignisbus Ihrem Konto automatisch zugewiesen wird, können Sie ihn nicht mithilfe einer CloudFormation Vorlage erstellen, wie Sie es normalerweise für jede Ressource tun würden, die Sie in einen CloudFormation Stapel aufnehmen möchten. Um den Standard-Event-Bus in einen CloudFormation Stack aufzunehmen, müssen Sie ihn zuerst in einen Stack importieren. Nachdem Sie den Standard-Event-Bus in einen Stack importiert haben, können Sie die Eigenschaften des Event-Busses nach Bedarf aktualisieren.

Um eine vorhandene Ressource in einen neuen oder vorhandenen CloudFormation Stack zu importieren, benötigen Sie die folgenden Informationen:

- Eine eindeutige Kennung für die zu importierende Ressource.

  Für Standard-Event-Busse ist der Bezeichner `Name` und dann der Identifier-Wert`default`.

- Eine Vorlage, die die aktuellen Eigenschaften der vorhandenen Ressource genau beschreibt.

  Der folgende Vorlagenausschnitt enthält eine `AWS::Events::EventBus` Ressource, die die aktuellen Eigenschaften eines Standard-Event-Busses beschreibt. In diesem Beispiel wurde der Eventbus so konfiguriert, dass er A Kundenverwalteter Schlüssel und DLQ für die Verschlüsselung im Ruhezustand verwendet.

  Außerdem sollte die `AWS::Events::EventBus` Ressource, die den Standard-Event-Bus beschreibt, den Sie importieren möchten, eine `DeletionPolicy` Eigenschaft enthalten, die auf `Retain` gesetzt ist.

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "Default event bus import example",
    "Resources": {
        "defaultEventBus": {
            "Type" : "AWS::Events::EventBus",
            "DeletionPolicy": "Retain",
            "Properties" : {
                "Name" : "default",
                "KmsKeyIdentifier" : "KmsKeyArn",
                "DeadLetterConfig" : {
                    "Arn" : "DLQ_ARN"
                }
            }
        }
    }
}
```

Weitere Informationen finden Sie im CloudFormation Benutzerhandbuch unter Integrieren vorhandener Ressourcen in die CloudFormation Verwaltung.

Löschen eines EventBridge Amazon-Event-Busses

Sie können einen benutzerdefinierten Event-Bus oder einen Partner-Event-Bus löschen. Sie können den Standard-Event-Bus nicht löschen. Durch das Löschen eines Event-Busses werden die mit diesem Event-Bus verknüpften Regeln gelöscht.

Um einen Event-Bus mit der EventBridge Konsole zu löschen

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Event Buses (Ereignisbusse) aus.
3. Wählen Sie den Event-Bus aus, den Sie löschen möchten.
4. Führen Sie eine der folgenden Aktionen aus:
   - Wählen Sie Löschen aus.
   - Wählen Sie den Namen des Event-Busses.

     Wählen Sie auf der Detailseite des Event-Busses die Option Löschen aus.

Berechtigungen für Amazon EventBridge-Event-Buses

Der standardmäßige Event Bus in Ihrem AWS-Konto erlaubt nur Ereignisse von einem Konto. Sie können einem Event Bus zusätzliche Berechtigungen gewähren, indem Sie ihm eine ressourcenbasierte Richtlinie zuordnen. Mit einer ressourcenbasierten Richtlinie können Sie `PutEvents`-, `PutRule`- und `PutTargets`-API-Aufrufe von einem anderen Konto zulassen. Sie können auch IAM-Bedingungen in der Richtlinie verwenden, um einer Organisation Berechtigungen zu erteilen, Tags anzuwenden oder Ereignisse nur nach denen einer bestimmten Regel oder eines bestimmten Kontos zu filtern. Sie können eine ressourcenbasierte Richtlinie für einen Event Bus bei der Erstellung oder danach festlegen.

EventBridge-APIs, die einen Event-Bus-`Name`-Parameter wie `PutRule`, `PutTargets`, `DeleteRule`, `RemoveTargets`, `DisableRule` und `EnableRule` akzeptieren, akzeptieren auch den Event-Bus-ARN. Verwenden Sie diese Parameter, um über die APIs auf konto- oder regionsübergreifende Event Buses zu verweisen. Sie können beispielsweise `PutRule` aufrufen, um eine Regel für einen Event Bus in einem anderen Konto zu erstellen, ohne eine Rolle übernehmen zu müssen.

Sie können die Beispielrichtlinien in diesem Thema einer IAM-Rolle zuordnen, um die Erlaubnis zu erteilen, Ereignisse an ein anderes Konto oder eine andere Region zu senden. Verwenden Sie IAM-Rollen, um Kontrollrichtlinien der Organisation und Grenzen dafür festzulegen, wer Ereignisse von

Ihrem Konto an andere Konten senden kann. Wir empfehlen, immer IAM-Rollen zu verwenden, wenn das Ziel einer Regel ein Event Bus ist. Sie können IAM-Rollen mithilfe von `PutTarget`-Aufrufen anhängen. Informationen zum Erstellen einer Regel zum Senden von Ereignissen an ein anderes Konto oder eine andere Region finden Sie unter EventBridge Amazon-Ereignisse zwischen AWS Konten senden und empfangen.

Themen

Verwalten von Event-Bus-Berechtigungen

Gehen Sie wie folgt vor, um die Berechtigungen eines vorhandenen Event Bus anzupassen. Informationen über die Verwendung von AWS CloudFormation zur Erstellung einer Event-Bus-Richtlinie finden Sie unter AWS::Events::EventBusPolicy.

So verwalten Sie Berechtigungen für einen vorhandenen Event Bus

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im linken Navigationsbereich die Option Event Buses aus.
3. Wählen Sie unter Name den Namen des Event Bus aus, für den Sie die Berechtigungen verwalten möchten.

   Wenn eine Ressourcenrichtlinie an den Event Bus angehängt ist, wird die Richtlinie angezeigt.
4. Wählen Sie Berechtigungen verwalten aus und führen Sie dann einen der folgenden Schritte aus:

- Geben Sie die Richtlinie ein, die die zu erteilenden Berechtigungen für den Event Bus beinhaltet. Sie können eine Richtlinie aus einer anderen Quelle einfügen oder den JSON-Code für die Richtlinie eingeben.
- Um eine Vorlage für die Richtlinie zu verwenden, wählen Sie Vorlage laden aus. Ändern Sie die Richtlinie entsprechend Ihrer Umgebung und fügen Sie zusätzliche Aktionen hinzu, für deren Verwendung Sie den Prinzipal in der Richtlinie autorisieren.

5. Wählen Sie Aktualisieren aus.

Die Vorlage enthält Beispiele mit Richtlinienanweisungen, die Sie an Ihr Konto und Ihre Umgebung anpassen können. Die Vorlage ist keine gültige Richtlinie. Sie können die Vorlage für Ihren Anwendungsfall ändern oder eine der Beispielrichtlinien kopieren und anpassen.

In der Vorlage werden Richtlinien geladen, die ein Beispiel dafür enthalten, wie einem Konto Berechtigungen zur Verwendung der `PutEvents`-Aktion erteilt werden, wie einer Organisation Berechtigungen erteilt werden und wie dem Konto Berechtigungen zur Verwaltung von Regeln im Konto erteilt werden. Sie können die Vorlage für Ihr spezielles Konto anpassen und dann die anderen Abschnitte aus der Vorlage löschen. Weitere Beispielrichtlinien finden Sie an späterer Stelle in diesem Thema.

Wenn Sie versuchen, die Berechtigungen für den Bus zu aktualisieren, die Richtlinie jedoch einen Fehler enthält, weist eine Fehlermeldung auf das spezifische Problem in der Richtlinie hin.

```
 ### Choose which sections to include in the policy to match your use case. ###
 ### Be sure to remove all lines that start with ###, including the ### at the end of
the line. ###

 ### The policy must include the following: ###

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

     ### To grant permissions for an account to use the PutEvents action, include the
following, otherwise delete this section: ###

     {

       "Sid": "AllowAccountToPutEvents",
       "Effect": "Allow",
```

```
        "Principal": {
          "AWS": "<ACCOUNT_ID>"
        },
        "Action": "events:PutEvents",
        "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/default"
      },

      ### Include the following section to grant permissions to all members of your AWS
Organizations to use the PutEvents action ###

      {
        "Sid": "AllowAllAccountsFromOrganizationToPutEvents",
        "Effect": "Allow",
        "Principal": "*",
        "Action": "events:PutEvents",
        "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/default",
        "Condition": {
          "StringEquals": {
            "aws:PrincipalOrgID": "o-yourOrgID"
          }
        }
      },

      ### Include the following section to grant permissions to the account to manage
the rules created in the account ###

      {
        "Sid": "AllowAccountToManageRulesTheyCreated",
        "Effect": "Allow",
        "Principal": {
          "AWS": "<ACCOUNT_ID>"
        },
        "Action": [
          "events:PutRule",
          "events:PutTargets",
          "events:DeleteRule",
          "events:RemoveTargets",
          "events:DisableRule",
          "events:EnableRule",
          "events:TagResource",
          "events:UntagResource",
          "events:DescribeRule",
          "events:ListTargetsByRule",
          "events:ListTagsForResource"],
```

```
        "Resource": "arn:aws:events:us-east-1:123456789012:rule/default",
        "Condition": {
          "StringEqualsIfExists": {
            "events:creatorAccount": "<ACCOUNT_ID>"
          }
        }
    }]
  }
```

Beispielrichtlinie: Senden von Ereignissen an den Standard-Bus in einem anderen Konto

Die folgende Beispielrichtlinie erteilt dem Konto 111122223333 die Berechtigung, Ereignisse im Standard-Event-Bus im Konto 123456789012 zu veröffentlichen.

```
{
   "Version": "2012-10-17",
   "Statement": [
       {
        "Sid": "sid1",
        "Effect": "Allow",
        "Principal": {"AWS":"arn:aws:iam::111112222333:root"},
        "Action": "events:PutEvents",
        "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/default"
        }
    ]
  }
```

Beispielrichtlinie: Senden von Ereignissen an einen benutzerdefinierten Bus in einem anderen Konto

Die folgende Beispielrichtlinie erteilt dem Konto 111122223333 die Berechtigung, Ereignisse im `central-event-bus` im Konto 123456789012 zu veröffentlichen, jedoch nur für Ereignisse, bei denen der Quellwert auf `com.exampleCorp.webStore` und der `detail-type` auf `newOrderCreated` gesetzt ist.

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

```
      "Sid": "WebStoreCrossAccountPublish",
      "Effect": "Allow",
      "Action": [
        "events:PutEvents"
      ],
      "Principal": {
        "AWS": "arn:aws:iam::111112222333:root"
      },
      "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/central-event-bus",
      "Condition": {
        "StringEquals": {
          "events:detail-type": "newOrderCreated",
          "events:source": "com.exampleCorp.webStore"
        }
      }
    }
  ]
}
```

Beispielrichtlinie: Senden von Ereignissen an einen Event Bus im selben Konto

Die folgende Beispielrichtlinie, die an einen Event Bus mit dem Namen `CustomBus1` angehängt ist, ermöglicht es dem Event Bus, Ereignisse aus demselben Konto und derselben Region zu empfangen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutEvents"
      ],
      "Resource": [
        "arn:aws:events:us-east-1:123456789:event-bus/CustomBus1"
      ]
    }
  ]
}
```

Beispielrichtlinie: Senden von Ereignissen an dasselbe Konto und Einschränken von Aktualisierungen

Die folgende Beispielrichtlinie erteilt dem Konto 123456789012 die Berechtigung, Regeln zu erstellen, zu löschen, zu aktualisieren, zu deaktivieren und zu aktivieren sowie Ziele hinzuzufügen oder zu entfernen. Sie schränkt diese Regeln ein, die auf Ereignisse mit einer Quelle von `com.exampleCorp.webStore` zutreffen, und verwendet das `"events:creatorAccount": "${aws:PrincipalAccount}"`, um sicherzustellen, dass nur das Konto 123456789012 diese Regeln und Ziele ändern kann, sobald sie erstellt wurden.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "InvoiceProcessingRuleCreation",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Action": [
        "events:PutRule",
        "events:DeleteRule",
        "events:DescribeRule",
        "events:DisableRule",
        "events:EnableRule",
        "events:PutTargets",
        "events:RemoveTargets"
      ],
      "Resource": "arn:aws:events:us-east-1:123456789012:rule/central-event-bus/*",
      "Condition": {
        "StringEqualsIfExists": {
          "events:creatorAccount": "${aws:PrincipalAccount}",
          "events:source": "com.exampleCorp.webStore"
        }
      }
    }
  ]
}
```

Beispielrichtlinie: Senden von Ereignissen nur aus einer bestimmten Regel an den Bus in einer anderen Region

Die folgende Beispielrichtlinie erteilt dem Konto 111122223333 die Berechtigung, Ereignisse, die einer Regel mit dem Namen `SendToUSE1AnotherAccount` in den Regionen Naher Osten (Bahrain) und USA West (Oregon) entsprechen, an einen Event Bus mit dem Namen `CrossRegionBus` in der Region USA Ost (Nord-Virginia) im Konto 123456789012 zu senden. Die Beispielrichtlinie wird dem Event Bus mit dem Namen `CrossRegionBus` im Konto 123456789012 hinzugefügt. Die Richtlinie lässt Ereignisse nur zu, wenn sie einer Regel entsprechen, die für den Event Bus im Konto 111122223333 angegeben ist. Die `Condition`-Anweisung beschränkt Ereignisse auf diejenigen, die den Regeln mit dem angegebenen Regel-ARN entsprechen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowSpecificRulesAsCrossRegionSource",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111112222333:root"
      },
      "Action": "events:PutEvents",
      "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/CrossRegionBus",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": [
            "arn:aws:events:us-west-2:111112222333:rule/CrossRegionBus/
SendToUSE1AnotherAccount",
            "arn:aws:events:me-south-1:111112222333:rule/CrossRegionBus/
SendToUSE1AnotherAccount"
          ]
        }
      }
    }
  ]
}
```

Beispielrichtlinie: Senden von Ereignissen nur aus einer bestimmten Region an eine andere Region

Die folgende Beispielrichtlinie erteilt dem Konto 111122223333 die Berechtigung, alle Ereignisse, die in den Regionen Naher Osten (Bahrain) und USA West (Oregon) generiert werden, an einen Event Bus mit dem Namen `CrossRegionBus` im Konto 123456789012 in der Region USA Ost (Nord-Virginia) zu senden. Das Konto 111122223333 ist nicht berechtigt, Ereignisse zu senden, die in einer anderen Region generiert wurden.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCrossRegionEventsFromUSWest2AndMESouth1",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111112222333:root"
      },
      "Action": "events:PutEvents",
      "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/CrossRegionBus",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": [
            "arn:aws:events:us-west-2:*:*",
            "arn:aws:events:me-south-1:*:*"
          ]
        }
      }
    }
  ]
}
```

Beispielrichtlinie: Verweigern des Sendens von Ereignissen aus bestimmten Regionen

Die folgende Beispielrichtlinie, die einem Event Bus mit dem Namen `CrossRegionBus` im Konto 123456789012 zugeordnet ist, erteilt dem Event Bus die Erlaubnis, Ereignisse vom Konto 111122223333 zu empfangen, jedoch keine Ereignisse, die in der Region USA West (Oregon) generiert wurden.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "1AllowAnyEventsFromAccount111112222333",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111112222333:root"
      },
      "Action": "events:PutEvents",
      "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/CrossRegionBus"
    },
    {
      "Sid": "2DenyAllCrossRegionUSWest2Events",
      "Effect": "Deny",
      "Principal": {
        "AWS": "*"
      },
      "Action": "events:PutEvents",
      "Resource": "arn:aws:events:us-east-1:123456789012:event-bus/CrossRegionBus",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": [
            "arn:aws:events:us-west-2:*:*"
          ]
        }
      }
    }
  ]
}
```

Generieren einer AWS CloudFormation-Vorlage aus einem Amazon-EventBridge-Event-Bus

Mit AWS CloudFormation können Sie Ihre AWS-Ressourcen über Konten und Regionen hinweg zentral und wiederholbar konfigurieren und verwalten, indem Sie die Infrastruktur als Code behandeln. CloudFormation bietet Ihnen zu diesem Zweck die Möglichkeit, Vorlagen zu erstellen, die die Ressourcen definieren, die Sie bereitstellen und verwalten möchten.

EventBridge ermöglicht es Ihnen, Vorlagen aus den vorhandenen Event Buses in Ihrem Konto zu generieren, um Ihnen den Einstieg in die Entwicklung von CloudFormation-Vorlagen zu erleichtern.

Darüber hinaus bietet EventBridge die Möglichkeit, die mit diesem Event Bus verknüpften Regeln in Ihre Vorlage aufzunehmen. Sie können diese Vorlagen dann als Grundlage zum Erstellen von Stacks von Ressourcen unter CloudFormation-Verwaltung verwenden.

Weitere Informationen zu CloudFormation finden Sie im AWS CloudFormation-Benutzerhandbuch..

> **Note**
>
> EventBridge enthält keine verwalteten Regeln in der generierten Vorlage.

Sie können eine Vorlage auch aus einer oder mehreren Regeln generieren, die in einem ausgewählten Event Bus enthalten sind.

So generieren Sie eine CloudFormation-Vorlage aus einem Event Bus

1. Öffnen Sie die Amazon-EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Event Buses (Ereignisbusse) aus.
3. Wählen Sie den Event Bus aus, aus dem Sie eine CloudFormation-Vorlage generieren möchten.
4. Wählen Sie im Menü Aktionen die Option CloudFormation-Vorlage und wählen Sie dann aus, in welchem Format EventBridge die Vorlage generieren soll: JSON oder YAML.

   EventBridge zeigt die Vorlage an, die im ausgewählten Format generiert wurde. Standardmäßig sind alle Regeln, die mit dem Event Bus verknüpft sind, in der Vorlage enthalten.

   - Um die Vorlage ohne Regeln zu generieren, deaktivieren Sie die Option Regeln in diesem Event Bus einbeziehen.
5. EventBridge bietet Ihnen die Möglichkeit, die Vorlagendatei herunterzuladen oder die Vorlage in die Zwischenablage zu kopieren.

   - Wählen Sie zum Herunterladen der Vorlagendatei Herunterladen aus.
   - Wählen Sie zum Kopieren der Vorlage in die Zwischenablage Kopieren aus.
6. Wählen Sie zum Beenden der Vorlage Abbrechen aus.

Sobald Sie Ihre AWS CloudFormation-Vorlage an Ihren Anwendungsfall angepasst haben, können Sie sie zum Erstellen von Stacks in CloudFormation verwenden.

Überlegungen zur Verwendung von CloudFormation-Vorlagen, die von Amazon EventBridge generiert wurden

Berücksichtigen Sie die folgenden Faktoren, wenn Sie eine CloudFormation-Vorlage verwenden, die Sie aus einem Event Bus generiert haben:

- EventBridge enthält keine Passwörter in der generierten Vorlage.

  Sie können die Vorlage so bearbeiten, dass sie Vorlagenparameter enthält, mit denen Benutzer Passwörter oder andere vertrauliche Informationen angeben können, wenn sie die Vorlage zum Erstellen oder Aktualisieren eines CloudFormation-Stacks verwenden.

  Darüber hinaus können Benutzer Secrets Manager verwenden, um ein Secret in der gewünschten Region zu erstellen und dann die generierte Vorlage so zu bearbeiten, dass dynamische Parameter eingesetzt werden.
- Die Ziele in der generierten Vorlage bleiben genau so, wie sie im ursprünglichen Event Bus angegeben wurden. Dies kann zu regionsübergreifenden Problemen führen, wenn Sie die Vorlage nicht entsprechend bearbeiten, bevor Sie sie zum Erstellen von Stacks in anderen Regionen verwenden.

  Darüber hinaus erstellt die generierte Vorlage die nachgelagerten Ziele nicht automatisch.

EventBridge Amazon-Veranstaltungen

Ein Ereignis weist auf eine Änderung in einer Umgebung hin, beispielsweise einer AWS -Umgebung, einem SaaS-Partnerservice oder einer SaaS-Partneranwendung oder einer Ihrer Anwendungen oder Services. Es folgen Beispiele für Ereignisse:

- Amazon EC2 generiert ein Ereignis, wenn sich der Status einer Instance von ausstehend zu ausgeführt ändert.
- Amazon EC2 Auto Scaling generiert Ereignisse, wenn Instances gestartet oder beendet werden.
- AWS CloudTrail veröffentlicht Ereignisse, wenn Sie API-Aufrufe tätigen.

Sie können auch geplante Ereignisse einrichten, die regelmäßig generiert werden.

Eine Liste der Services, die Ereignisse generieren, einschließlich Beispielereignissen der einzelnen Services, finden Sie unter Ereignisse im Zusammenhang mit Dienstleistungen AWS, und folgen Sie den Links in der Tabelle.

Ereignisse werden als JSON-Objekte dargestellt und sie haben alle eine ähnliche Struktur und dieselben Felder der obersten Ebene.

Der Inhalt des Feldes detail auf oberster Ebene unterscheidet sich, je nachdem, welcher Service das Ereignis generiert hat und um welches Ereignis es sich handelt. Die Kombination aus den Feldern source und detail-type dient zum Identifizieren der im Feld detail gefundenen Felder und Werte. Beispiele für Ereignisse, die von AWS Diensten generiert werden, finden Sie unterEreignisse im Zusammenhang mit Dienstleistungen AWS.

Themen

Das folgende Video erklärt die Grundlagen von Ereignissen: Was ist ein Ereignis?

Das folgende Video zeigt, wie Ereignisse entstehen EventBridge: Woher kommen Ereignisse

Referenz der Ereignisstruktur

Die folgenden Felder erscheinen in allen Ereignissen, die an einen Event-Bus übermittelt werden, und enthalten die Metadaten des Ereignisses:

```
{
  "???": "0",
  "???": "UUID",
  "???": "event name",
  "???": "event source",
  "???": "ARN",
  "???": "timestamp",
  "???": "region",
  "???": [
    "ARN"
  ],
  "???": {
    JSON object
  }
}
```

version

Die Standardeinstellung bei allen Ereignissen lautet 0 (Null).

id

Eine UUID der Version 4, die für jedes Ereignis generiert wurde. Sie können `id` verwenden, um Ereignisse zu verfolgen, während sie sich durch Regeln zu Zielen bewegen.

detail-type

Identifiziert in Kombination mit dem Feld source die Felder und Werte, die im Feld detail angezeigt werden.

Ereignisse, die von CloudTrail haben `AWS API Call via CloudTrail` als Wert für übermittelt `detail-type` werden.

Quelle

Identifiziert den Service, aus dem das Ereignis stammt. Alle Ereignisse, die von AWS -Services stammen, beginnen mit „aws“. Vom Kunden generierte Ereignisse können hier jeden Wert haben, solange sie nicht mit "aws" beginnen. Wir empfehlen, Reverse-Domänennamen-Zeichenfolgen im Namensstil von Java-Paketen zu verwenden.

Den richtigen Wert für `source` für einen AWS Dienst finden Sie in der Tabelle mit den Bedingungsschlüsseln. Wählen Sie einen Dienst aus der Liste aus und suchen Sie nach dem Dienstpräfix. Zum Beispiel CloudFront ist der `source` Wert für Amazon`aws.cloudfront`.

Konto

Die 12-stellige Zahl, die ein AWS Konto identifiziert.

time

Der Zeitstempel eines Ereignisses, der von dem Service, aus dem das Ereignis stammt, festgelegt werden kann. Wenn das Ereignis ein Zeitintervall umfasst, kann der Service die Startzeit berichten, sodass dieser Wert vor dem Zeitpunkt liegen kann, zu dem das Ereignis empfangen wird.

Region

Identifiziert die AWS Region, in der das Ereignis seinen Ursprung hat.

Ressourcen

Ein JSON-Array, das die ARNs enthält, die die am Ereignis beteiligten Ressourcen identifizieren. Der Service, der das Ereignis generiert, bestimmt, ob diese ARNs eingeschlossen werden. So enthalten Amazon EC2-Instance-Statusänderungen beispielsweise Amazon EC2-Instance-ARNs, und Auto Scaling-Ereignisse umfassen ARNs sowohl für Instances als auch Auto Scaling-Gruppen, jedoch schließen API-Aufrufe mit AWS CloudTrail keine Ressourcen-ARNs ein.

Detail

Ein JSON-Objekt, das Informationen zum Ereignis enthält. Der Service, der das Ereignis generiert, bestimmt den Inhalt dieses Feldes. Das kann `"{}"` sein.

AWS API-Aufrufereignisse enthalten Detailobjekte mit etwa 50 Feldern, die mehrere Ebenen tief verschachtelt sind.

> **Note**
>
> PutEventsakzeptiert Daten im JSON-Format. Für den Datentyp JSON-Zahl (Ganzzahl) gelten folgende Einschränkungen: ein Mindestwert von -9,223,372,036,854,775,808 und ein Höchstwert von 9,223,372,036,854,775,807.

Example Beispiel: Benachrichtigung über die Statusänderung für eine Amazon-EC2-Instance

Das folgende Ereignis in Amazon EventBridge weist darauf hin, dass eine Amazon EC2 EC2-Instance beendet wurde.

```
{
  "version": "0",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "EC2 Instance State-change Notification",
  "source": "aws.ec2",
  "account": "111122223333",
  "time": "2017-12-22T18:43:48Z",
  "region": "us-west-1",
  "resources": [
    "arn:aws:ec2:us-west-1:123456789012:instance/i-1234567890abcdef0"
  ],
  "detail": {
    "instance-id": " i-1234567890abcdef0",
    "state": "terminated"
  }
}
```

Mindestinformationen, die für ein gültiges benutzerdefiniertes Ereignis erforderlich sind

Wenn Sie benutzerdefinierte Ereignisse erstellen, müssen sie die folgenden Felder umfassen:

- `detail`
- `detail-type`
- `source`

```
{
```

```
  "detail-type": "event name",
  "source": "event source",
  "detail": {
  }
}
```

Hinzufügen von EventBridge Amazon-Ereignissen mit `PutEvents`

Die `PutEvents` Aktion sendet mehrere Ereignisse EventBridge in einer einzigen Anfrage an. Weitere Informationen finden Sie PutEventsin der Amazon EventBridge API-Referenz und put-events in der AWS CLI Befehlsreferenz.

Jede `PutEvents`-Anforderung kann eine begrenzte Anzahl von Einträgen unterstützen. Weitere Informationen finden Sie unter Amazon-EventBridge-Kontingente. Die `PutEvents`-Operation versucht, alle Einträge in der natürlichen Reihenfolge der Anforderung zu verarbeiten. EventBridge Weist jedem Ereignis nach dem Aufruf `PutEvents` eine eindeutige ID zu.

Themen

- Behandlung von Fehlern bei PutEvents
- Senden von Ereignissen mit dem AWS CLI
- Berechnung der Größe des EventBridge PutEvents Amazon-Eventeintrags

Der folgende Java-Beispielcode sendet zwei identische Ereignisse an EventBridge.

AWS SDK for Java Version 2.x

```
EventBridgeClient eventBridgeClient =
    EventBridgeClient.builder().build();

PutEventsRequestEntry requestEntry = PutEventsRequestEntry.builder()
    .resources("resource1", "resource2")
    .source("com.mycompany.myapp")
    .detailType("myDetailType")
    .detail("{ \"key1\": \"value1\", \"key2\": \"value2\" }")
    .build();

List <
PutEventsRequestEntry > requestEntries = new ArrayList <
PutEventsRequestEntry > ();
```

```
requestEntries.add(requestEntry);

PutEventsRequest eventsRequest = PutEventsRequest.builder()
    .entries(requestEntries)
    .build();

PutEventsResponse result = eventBridgeClient.putEvents(eventsRequest);

for (PutEventsResultEntry resultEntry: result.entries()) {
    if (resultEntry.eventId() != null) {
        System.out.println("Event Id: " + resultEntry.eventId());
    } else {
        System.out.println("PutEvents failed with Error Code: " +
 resultEntry.errorCode());
    }
}
```

AWS SDK for Java Version 1.0

```
EventBridgeClient eventBridgeClient =
    EventBridgeClient.builder().build();

PutEventsRequestEntry requestEntry = new PutEventsRequestEntry()
        .withTime(new Date())
        .withSource("com.mycompany.myapp")
        .withDetailType("myDetailType")
        .withResources("resource1", "resource2")
        .withDetail("{ \"key1\": \"value1\", \"key2\": \"value2\" }");

PutEventsRequest request = new PutEventsRequest()
        .withEntries(requestEntry, requestEntry);

PutEventsResult result = awsEventsClient.putEvents(request);

for (PutEventsResultEntry resultEntry : result.getEntries()) {
    if (resultEntry.getEventId() != null) {
        System.out.println("Event Id: " + resultEntry.getEventId());
    } else {
        System.out.println("Injection failed with Error Code: " +
 resultEntry.getErrorCode());
    }
}
```

Nachdem Sie diesen Code ausgeführt haben, enthält das `PutEvents`-Ergebnis ein Array von Antworteinträgen. Jeder Eintrag im Antwort-Array entspricht einem Eintrag im Anforderungs-Array in der Reihenfolge vom Anfang bis zum Ende der Anforderung und Antwort. Das Antwort-Array `Entries` enthält stets die gleiche Anzahl Einträge wie in der Anforderung.

Behandlung von Fehlern bei **`PutEvents`**

Wenn ein einzelner Eintrag in einer Anfrage fehlschlägt, EventBridge wird standardmäßig die Verarbeitung der restlichen Einträge in der Anforderung fortgesetzt. Ein Antwort-`Entries`-Array kann sowohl erfolgreiche als auch erfolglose Einträge enthalten. Sie müssen erfolglose Einträge erkennen und sie im nachfolgenden Aufruf aufnehmen.

Erfolgreiche Ergebniseinträge enthalten einen `Id`-Wert und erfolglose Ergebniseinträge enthalten `ErrorCode`- und `ErrorMessage`-Werte. `ErrorCode` beschreibt die Art des Fehlers. `ErrorMessage` enthält weitere Informationen über den Fehler. Im folgenden Beispiel gibt es drei Ergebniseinträge für eine `PutEvents`-Anforderung. Der zweite Eintrag ist erfolglos.

```
{
    "FailedEntryCount": 1,
    "Entries": [
        {
            "EventId": "11710aed-b79e-4468-a20b-bb3c0c3b4860"
        },
        {   "ErrorCode": "InternalFailure",
            "ErrorMessage": "Internal Service Failure"
        },
        {
            "EventId": "d804d26a-88db-4b66-9eaf-9a11c708ae82"
        }
    ]
}
```

> **Note**
>
> Wenn Sie früher `PutEvents` ein Ereignis in einem Event-Bus veröffentlichen, der nicht existiert, wird EventBridge beim Event-Matching keine entsprechende Regel gefunden und das Ereignis wird gelöscht. Es EventBridge wird zwar eine `200` Antwort gesendet, aber die Anfrage wird nicht fehlschlagen oder das Ereignis in den `FailedEntryCount` Wert der Anforderungsantwort einbeziehen.

Einträge, die erfolglos waren, können Sie in nachfolgenden `PutEvents`-Anforderungen aufnehmen. Wenn Sie herausfinden möchten, ob die Anforderung fehlerhafte Einträge enthält, überprüfen Sie zunächst den Parameter `FailedRecordCount` in `PutEventsResult`. Wenn er nicht Null ist, können Sie jeden `Entry`, der einen `ErrorCode` hat, der nicht Null ist, zu einer nachfolgenden Anforderung hinzufügen. Das folgende Beispiel zeigt einen einfachen Ausfall-Handler.

```
PutEventsRequestEntry requestEntry = new PutEventsRequestEntry()
        .withTime(new Date())
        .withSource("com.mycompany.myapp")
        .withDetailType("myDetailType")
        .withResources("resource1", "resource2")
        .withDetail("{ \"key1\": \"value1\", \"key2\": \"value2\" }");

List<PutEventsRequestEntry> putEventsRequestEntryList = new ArrayList<>();
for (int i = 0; i < 3; i++) {
    putEventsRequestEntryList.add(requestEntry);
}

PutEventsRequest putEventsRequest = new PutEventsRequest();
putEventsRequest.withEntries(putEventsRequestEntryList);
PutEventsResult putEventsResult = awsEventsClient.putEvents(putEventsRequest);

while (putEventsResult.getFailedEntryCount() > 0) {
    final List<PutEventsRequestEntry> failedEntriesList = new ArrayList<>();
    final List<PutEventsResultEntry> PutEventsResultEntryList =
 putEventsResult.getEntries();
    for (int i = 0; i < PutEventsResultEntryList.size(); i++) {
        final PutEventsRequestEntry putEventsRequestEntry =
 putEventsRequestEntryList.get(i);
        final PutEventsResultEntry putEventsResultEntry =
 PutEventsResultEntryList.get(i);
        if (putEventsResultEntry.getErrorCode() != null) {
            failedEntriesList.add(putEventsRequestEntry);
        }
    }
    putEventsRequestEntryList = failedEntriesList;
    putEventsRequest.setEntries(putEventsRequestEntryList);
    putEventsResult = awsEventsClient.putEvents(putEventsRequest);
    }
```

Senden von Ereignissen mit dem AWS CLI

Sie können das verwenden AWS CLI , um benutzerdefinierte Ereignisse an zu senden, EventBridge damit sie verarbeitet werden können. Im folgenden Beispiel wird ein benutzerdefiniertes Ereignis eingefügt EventBridge:

```
aws events put-events \
--entries '[{"Time": "2016-01-14T01:02:03Z", "Source": "com.mycompany.myapp",
 "Resources": ["resource1", "resource2"], "DetailType": "myDetailType", "Detail":
 "{ \"key1\": \"value1\", \"key2\": \"value2\" }"}]'
```

Sie können auch eine JSON-Datei erstellen, die benutzerdefinierte Ereignisse enthält.

```
[
  {
    "Time": "2016-01-14T01:02:03Z",
    "Source": "com.mycompany.myapp",
    "Resources": [
      "resource1",
      "resource2"
    ],
    "DetailType": "myDetailType",
    "Detail": "{ \"key1\": \"value1\", \"key2\": \"value2\" }"
  }
]
```

Geben Sie dann AWS CLI an der Befehlszeile Folgendes ein, um die Einträge aus dieser Datei zu lesen und Ereignisse zu senden:

```
aws events put-events --entries file://entries.json
```

Berechnung der Größe des EventBridge PutEvents Amazon-Eventeintrags

Mithilfe der `PutEvents` Aktion können Sie benutzerdefinierte Ereignisse an senden. EventBridge Sie können mehrere Ereigniseinträge aus Effizienzgründen im Stapel in einer Anforderung verarbeiten. Die Gesamtgröße des Eintrags muss weniger als 256 KB betragen. Sie können die Größe des Eintrags berechnen, bevor Sie die Ereignisse senden.

> **Note**
>
> Die Größenbeschränkung hängt vom Eintrag ab. Selbst wenn der Eintrag die Größenbeschränkung unterschreitet, EventBridge ist das Ereignis in aufgrund der erforderlichen Zeichen und Schlüssel der JSON-Darstellung des Ereignisses immer größer als die Eintragsgröße. Weitere Informationen finden Sie unter EventBridge Amazon-Veranstaltungen.

EventBridge berechnet die `PutEventsRequestEntry` Größe wie folgt:

- Falls angegeben, misst der Parameter `Time` 14 Bytes.
- Die Parameter `Source` und `DetailType` sind die Anzahl der Bytes für ihre in UTF-8 kodierte Form.
- Falls angegeben, ist der Parameter `Detail` die Anzahl von Bytes für die entsprechende UTF-8 kodierte Form.
- Falls angegeben, ist jeder Eintrag des Parameters `Resources` die Anzahl von Bytes für die entsprechende UTF-8 kodierte Form.

Das folgende Java-Code-Beispiel berechnet die Größe eines bestimmten `PutEventsRequestEntry`-Objekts.

```
int getSize(PutEventsRequestEntry entry) {
    int size = 0;
    if (entry.getTime() != null) {
        size += 14;
    }
    size += entry.getSource().getBytes(StandardCharsets.UTF_8).length;
    size += entry.getDetailType().getBytes(StandardCharsets.UTF_8).length;
    if (entry.getDetail() != null) {
```

```
        size += entry.getDetail().getBytes(StandardCharsets.UTF_8).length;
    }
    if (entry.getResources() != null) {
        for (String resource : entry.getResources()) {
            if (resource != null) {
                size += resource.getBytes(StandardCharsets.UTF_8).length;
            }
        }
    }
    return size;
}
```

> **Note**
>
> Wenn der Eintrag größer als 256 KB ist, empfehlen wir, das Ereignis in einen Amazon-S3-Bucket hochzuladen und die `Object URL` in den `PutEvents`-Eintrag aufzunehmen.

Ereignisse im Zusammenhang mit Dienstleistungen AWS

Viele AWS Dienste generieren Ereignisse, die EventBridge empfangen werden. Wenn ein AWS Dienst in Ihrem Konto ein Ereignis ausgibt, wird es an den Standard-Ereignisbus Ihres Kontos weitergeleitet.

Bereitstellung von AWS Ereignissen über Dienste

Jeder AWS Dienst, der Ereignisse generiert, sendet sie entweder nach bestem Bemühen oder EventBridge als Dauerzustellungsversuch an.

- Bei der Bereitstellung nach bestem Wissen versucht der Service, alle Ereignisse an zu senden EventBridge, aber in einigen seltenen Fällen kann es vorkommen, dass ein Ereignis nicht zugestellt wird.
- Dauerhafte Zustellung bedeutet, dass der Service erfolgreich versucht, Ereignisse EventBridge mindestens einmal zuzustellen.

  EventBridge akzeptiert alle gültigen Ereignisse unter normalen Bedingungen. In Fällen, in denen Ereignisse aufgrund einer EventBridge Betriebsunterbrechung nicht zugestellt werden können, werden sie später vom AWS Service für bis zu 24 Stunden erneut versucht.

Sobald ein Ereignis zugestellt wurde EventBridge, EventBridge gleicht es mit den Regeln ab und folgt dann der Wiederholungsrichtlinie sowie allen Warteschlangen für unzustellbare Nachrichten, die für das oder die Ereignisziele angegeben wurden.

Eine Liste der AWS Dienste, die Ereignisse generieren, finden Sie unter. ???

Zugreifen auf AWS Dienstereignisse über AWS CloudTrail

AWS CloudTrail ist ein Dienst, der Ereignisse wie AWS API-Aufrufe automatisch aufzeichnet. Sie können EventBridge Regeln erstellen, die die Informationen von verwenden CloudTrail. Weitere Informationen zu CloudTrail finden Sie unter Was ist AWS CloudTrail? .

Alle Ereignisse, die von geliefert werden`AWS API Call via CloudTrail`, CloudTrail haben den Wert für`detail-type`.

Um Ereignisse mit einem `detail-type` Wert von aufzuzeichnen`AWS API Call via CloudTrail`, ist ein CloudTrail Trail mit aktivierter Protokollierung erforderlich.

Bei Verwendung CloudTrail mit Amazon S3 müssen Sie die Protokollierung von Datenereignissen konfigurieren CloudTrail . Weitere Informationen finden Sie unter Aktivieren der CloudTrail Ereignisprotokollierung für S3-Buckets und -Objekte.

Einige Vorkommnisse in AWS Diensten können EventBridge sowohl vom Dienst selbst als auch von gemeldet werden. CloudTrail Beispielsweise generiert ein Amazon EC2 EC2-API-Aufruf, der eine Instance startet oder stoppt, sowohl EventBridge Ereignisse als auch Ereignisse durch CloudTrail.

CloudTrail unterstützt sowohl API-Aufrufer als auch Ressourcenbesitzer beim Empfangen von Ereignissen in ihren Amazon S3 S3-Buckets, indem sie Trails erstellen, und übermittelt Ereignisse an API-Aufrufer über. EventBridge Ressourcenbesitzer können zusätzlich zu API-Aufrufern kontoübergreifende API-Aufrufe über überwachen. EventBridge CloudTrailDie Integration mit EventBridge bietet eine bequeme Möglichkeit, automatisierte, regelbasierte Workflows als Reaktion auf Ereignisse einzurichten.

Sie können AWS Put*Events-API-Aufrufereignisse, die größer als 256 KB sind, nicht als Ereignismuster verwenden, da die maximale Größe aller Put*Events-Anfragen 256 KB beträgt. Weitere Informationen zu den API-Aufrufen, die Sie verwenden können, finden Sie unter CloudTrail Unterstützte Dienste und Integrationen.

Empfangen schreibgeschützter Verwaltungsereignisse von Diensten AWS

Sie können Regeln für Ihren standardmäßigen oder benutzerdefinierten Ereignisbus einrichten, um schreibgeschützte Verwaltungsereignisse von Diensten über zu empfangen. AWS CloudTrail Verwaltungsereignisse bieten Einblick in Verwaltungsvorgänge, die mit Ressourcen in Ihrem AWS Konto ausgeführt werden. Sie werden auch als Vorgänge auf Steuerebene bezeichnet. Weitere Informationen finden Sie unter Protokollieren von Verwaltungsereignissen im CloudTrail - Benutzerhandbuch.

Für jede Regel in den standardmäßigen oder benutzerdefinierten Event Buses können Sie den Regelstatus festlegen, um zu steuern, welche Ereignistypen empfangen werden sollen:

- Deaktivieren Sie die Regel, sodass EventBridge keine Ereignisse mit der Regel abgeglichen werden.
- Aktivieren Sie die Regel, sodass Ereignisse EventBridge mit der Regel abgeglichen werden, mit Ausnahme von schreibgeschützten AWS Verwaltungsereignissen, die über übermittelt werden. CloudTrail
- Aktivieren Sie die Regel, sodass alle Ereignisse EventBridge mit der Regel abgeglichen werden, einschließlich schreibgeschützter Verwaltungsereignisse, die über übermittelt werden. CloudTrail

Eventbusse von Partnern empfangen AWS keine Ereignisse.

Bei der Entscheidung, ob schreibgeschützte Verwaltungsereignisse empfangen werden sollen, sind einige Punkte zu beachten:

- Bestimmte Verwaltungsereignisse, die nur Lesezugriff habenDescribeKey, wie z. B. AWS Key Management Service GetKeyPolicy und oder IAM GetPolicy und GetRole Ereignisse, treten wesentlich häufiger auf als typische Änderungsereignisse.
- Möglicherweise erhalten Sie bereits schreibgeschützte Verwaltungsereignisse, wenn diese Ereignisse nicht mit Describe, Get oder List beginnen. Ereignisse aus den folgenden AWS STS APIs sind beispielsweise Änderungsereignisse, auch wenn sie mit dem Verb beginnen: Get
  - GetFederationToken
  - GetSessionToken

  Eine Liste schreibgeschützter Verwaltungsereignisse, die nicht der List BenennungskonventionDescribe, oder entsprechenGet, nach AWS Diensten geordnet, finden Sie unter. ???

So erstellen Sie eine Regel, die schreibgeschützte Verwaltungsereignisse mit der CLI empfängt AWS

- Verwenden Sie den Befehl `put-rule` zum Erstellen oder Aktualisieren der Regel mithilfe von Parametern, um:
  - Anzugeben, dass die Regel zum Standard-Event-Bus oder zu einem bestimmten benutzerdefinierten Event Bus gehört
  - Den Regelstatus als `ENABLED_WITH_ALL_CLOUDTRAIL_MANAGEMENT_EVENTS` festzulegen

```
aws events put-rule --name "ruleForManagementEvents" --event-bus-name
"default" --state "ENABLED_WITH_ALL_CLOUDTRAIL_MANAGEMENT_EVENTS"
```

> **Note**
>
> Die Aktivierung einer Regel für CloudWatch Verwaltungsereignisse wird nur über die AWS CLI und AWS CloudFormation Vorlagen unterstützt.

Example

Das folgende Beispiel veranschaulicht, wie Sie einen Abgleich mit bestimmten Ereignissen durchführen. Aus Gründen der Übersichtlichkeit und einfacheren Bearbeitung empfiehlt es sich, eine dedizierte Regel für den Abgleich bestimmter Ereignisse zu definieren.

In diesem Fall entspricht die dedizierte Regel dem `AssumeRole` Verwaltungsereignis von AWS Security Token Service.

```
{
    "source" : [ "aws.sts" ],
    "detail-type": ["AWS API Call via CloudTrail"],
    "detail" : {
        "eventName" : ["AssumeRole"]
    }
}
```

AWS Dienste, die Ereignisse generieren

Die folgende Tabelle zeigt AWS Dienste, die Ereignisse generieren. Wählen Sie den Namen des Dienstes aus, um weitere Informationen darüber zu erhalten, wie dieser Dienst und wie dieser Dienst EventBridge zusammenarbeiten.

Jeder AWS Dienst, der Ereignisse generiert, sendet sie entweder im Rahmen eines bestmöglichen Zustellungsversuchs oder EventBridge als dauerhafte Zustellungsversuche an. Weitere Informationen finden Sie unter ???.

Diese Tabelle enthält eine Darstellung der AWS Dienste, an die Ereignisse gesendet werden EventBridge, enthält jedoch nicht alle Dienste. Gehen Sie bei nicht aufgeführten Diensten, an die Ereignisse gesendet werden EventBridge, von einer bestmöglichen Bereitstellung aus.

Service	Versuchstyp
Alexa for Business	Bestmöglich
AWS Account Management	Bestmöglich
Amazon API Gateway	Bestmöglich
AWS AppConfig	Bestmöglich
Amazon AppFlow	Bestmöglich
Application Auto Scaling	Bestmöglich
AWS Cost Profiler für Anwendungen	Bestmöglich
AWS Application Migration Service	Bestmöglich
Amazon Athena	Bestmöglich
AWS Backup	Bestmöglich
AWS Batch	Dauerhaft
Amazon Braket	Dauerhaft
AWS Certificate Manager	Bestmöglich

Service	Versuchstyp
Amazon Chime	Bestmöglich
Amazon Cloud Directory	Bestmöglich
AWS CloudFormation	Dauerhaft
Amazon CloudFront	Bestmöglich
AWS CloudHSM	Bestmöglich
Amazon CloudSearch	Bestmöglich
AWS CloudShell	Bestmöglich
Ereignisse von AWS CloudTrail	Bestmöglich
Amazon CloudWatch	Dauerhaft
Einblicke in CloudWatch Amazon-An wendungen	Bestmöglich
Amazon CloudWatch Internetmonitor	Bestmöglich
CloudWatch Amazon-Protokolle	Bestmöglich
Amazon CloudWatch Synthetics	Bestmöglich
AWS CodeArtifact	Dauerhaft
AWS CodeBuild	Bestmöglich
AWS CodeCommit	Bestmöglich
AWS CodeDeploy	Bestmöglich
CodeGuru Amazon-Profiler	Bestmöglich
AWS CodePipeline	Bestmöglich
AWS CodeStar	Bestmöglich

Service	Versuchstyp
CodeConnections	Bestmöglich
Amazon Cognito Identity	Bestmöglich
Amazon-Cognito-Benutzerpools	Bestmöglich
Amazon Cognito Sync	Bestmöglich
AWS Config	Bestmöglich
Amazon Connect	Bestmöglich
Amazon Connect Voice ID	Bestmöglich
AWS Control Tower	Bestmöglich
AWS Database Migration Service	Bestmöglich
AWS Data Exchange	Bestmöglich
Amazon Data Lifecycle Manager	Bestmöglich
AWS Data Pipeline	Bestmöglich
AWS DataSync	Bestmöglich
AWS Device Farm	Bestmöglich
DevOpsAmazon-Guru	Bestmöglich
AWS Direct Connect	Bestmöglich
AWS Directory Service	Bestmöglich
Amazon-DynamoDB	Bestmöglich
AWS Elastic Beanstalk	Bestmöglich
Amazon Elastic Block Store	Bestmöglich

Service	Versuchstyp
Änderungen an einem Volume von Amazon Elastic Block Store	Bestmöglich
Amazon ElastiCache	Bestmöglich
Amazon Elastic Compute Cloud (Amazon EC2)	Bestmöglich
Amazon EC2 Auto Scaling	Bestmöglich
Amazon-EC2-Flotten	Bestmöglich
Unterbrechung einer Amazon-EC2-Spot-Instance	Bestmöglich
Amazon Elastic Container Registry	Bestmöglich
Amazon Elastic Container Service	Dauerhaft
AWS Elastic Disaster Recovery	Bestmöglich
Amazon Elastic File System	Bestmöglich
Amazon Elastic Kubernetes Service	Bestmöglich
Elastic Load Balancing	Bestmöglich
Amazon Elastic MapReduce	Bestmöglich
Amazon Elastic Transcoder	Bestmöglich
AWS Elemental MediaConnect	Bestmöglich
AWS Elemental MediaConvert	Dauerhaft
AWS Elemental MediaLive	Bestmöglich
AWS Elemental MediaPackage	Bestmöglich
AWS Elemental MediaStore	Dauerhaft

Service	Versuchstyp
Amazon EMR	Bestmöglich
Amazon EMR in EKS	Bestmöglich
Amazon EMR Serverless	Bestmöglich
EventBridge Geplante Regeln von Amazon	Dauerhaft
EventBridge Amazon-Schemas	Bestmöglich
AWS Fault Injection Service	Bestmöglich
Forecast	Bestmöglich
Amazon GameLift	Bestmöglich
AWS Glue	Bestmöglich
AWS Glue DataBrew	Bestmöglich
AWS Ground Station	Bestmöglich
Amazon GuardDuty	Bestmöglich
AWS Health	Bestmöglich
AWS HealthLake	Dauerhaft
AWS Identity and Access Management (ICH BIN)	Bestmöglich
IAM Access Analyzer	Bestmöglich
Amazon Inspector Classic	Bestmöglich
Amazon Inspector	Bestmöglich
AWS IoT	Bestmöglich
AWS IoT Analytics	Dauerhaft

Service	Versuchstyp
AWS IoT Greengrass V1	Bestmöglich
AWS IoT Greengrass V2	Bestmöglich
Amazon Interactive Video Service	Bestmöglich
Amazon Kinesis	Bestmöglich
Amazon Data Firehose	Bestmöglich
AWS Key Management Service CMK-Löschung	Dauerhaft
AWS Key Management Service CMK-Rotation	Bestmöglich
AWS Key Management Service Ablauf des importierten Schlüsselmaterials	Bestmöglich
AWS Lambda	Bestmöglich
Amazon Location Service	Dauerhaft
Amazon Machine Learning	Bestmöglich
Amazon Macie	Bestmöglich
Amazon Managed Blockchain	Bestmöglich
AWS Managed Services	Bestmöglich
AWS Management Console Melden Sie sich an	Bestmöglich
AWS Marketplace für Messgeräte	Bestmöglich
AWS Migration Hub	Bestmöglich
AWS Migration Hub Refactor Spaces	Bestmöglich
AWS Überwachung	Bestmöglich
AWS Network Manager	Bestmöglich

Service	Versuchstyp
OpenSearch Amazon-Dienst	Bestmöglich
AWS OpsWorks	Dauerhaft
AWS OpsWorks CM	Bestmöglich
AWS Organizations	Bestmöglich
Amazon Polly	Bestmöglich
AWS Private Certificate Authority	Bestmöglich
AWS Proton	Bestmöglich
Amazon QLDB	Dauerhaft
Amazon QuickSight	Bestmöglich
Amazon RDS	Bestmöglich
AWS Papierkorb	Bestmöglich
Amazon-Redshift	Dauerhaft
Amazon Redshift-Daten-API	Bestmöglich
Amazon Redshift Serverless	Bestmöglich
AWS Resource Access Manager	Bestmöglich
AWS Resource Groups	Bestmöglich
AWS Resource Groups Tagging API	Bestmöglich
Amazon Route 53	Bestmöglich
Amazon Route 53 Recovery-Bereitschaft	Bestmöglich
Amazon SageMaker	Bestmöglich

Service	Versuchstyp
Savings Plans	Bestmöglich
AWS Secrets Manager	Bestmöglich
AWS Security Hub	Dauerhaft
AWS Security Token Service	Bestmöglich
AWS Server Migration Service	Bestmöglich
AWS Service Catalog	Bestmöglich
AWS Signer	Dauerhaft
Amazon Simple Email Service	Bestmöglich
Amazon-Simple-Storage-Service (Amazon-S3)	Dauerhaft
Amazon S3 Glacier	Bestmöglich
Amazon S3 in Outposts	Bestmöglich
Amazon Simple Queue Service	Bestmöglich
Amazon Simple Notification Service	Bestmöglich
Amazon Simple Workflow Service	Bestmöglich
AWS Step Functions	Bestmöglich
AWS Storage Gateway	Dauerhaft
AWS Support	Bestmöglich
AWS Systems Manager	Bestmöglich
Amazon Transcribe	Bestmöglich
AWS Transfer Family	Bestmöglich

Service	Versuchstyp
AWS Transit Gateway	Bestmöglich
Amazon Translate	Dauerhaft
AWS Trusted Advisor	Bestmöglich
AWS WAF	Bestmöglich
AWS WAF Regional	Bestmöglich
AWS Well-Architected Tool	Bestmöglich
Amazon WorkDocs	Bestmöglich
Amazon WorkSpaces	Bestmöglich
AWS X-Ray	Bestmöglich

Von AWS Diensten generierte Managementereignisse

Im Allgemeinen beginnen APIs, die Verwaltungsereignisse (oder schreibgeschützte Ereignisse) generieren, mit den Verben `Describe`, `Get` oder `List`. In der folgenden Tabelle sind AWS Dienste und die von ihnen generierten Verwaltungsereignisse aufgeführt, die dieser Benennungskonvention nicht entsprechen. Weitere Informationen zu Verwaltungsereignissen finden Sie unter ???.

Verwaltungsereignisse, die nicht mit **`Describe`**, **`Get`** oder **`List`** beginnen

In der folgenden Tabelle sind AWS Dienste und die von ihnen generierten Verwaltungsereignisse aufgeführt, die nicht den typischen Namenskonventionen entsprechen`Describe`, nämlich mit`Get`, oder zu beginnen`List`.

Service	Ereignisname	Ereignistyp
Alexa for Business	ResolveRoom	API-Aufruf.
Alexa for Business	SearchAddressBooks	API-Aufruf.

Service	Ereignisname	Ereignistyp
Alexa for Business	SearchContacts	API-Aufruf.
Alexa for Business	SearchDevices	API-Aufruf.
Alexa for Business	SearchProfiles	API-Aufruf.
Alexa for Business	SearchRooms	API-Aufruf.
Alexa for Business	SearchSkillGroups	API-Aufruf.
Alexa for Business	SearchUsers	API-Aufruf.
IAM Access Analyzer	ValidatePolicy	API-Aufruf.
AWS AdSpace Saubere Räume	BatchGetSchema	API-Aufruf.
AWS Amplify UI-Builder	ExportComponents	API-Aufruf.
AWS Amplify UI-Builder	ExportForms	API-Aufruf.
AWS Amplify UI-Builder	ExportThemes	API-Aufruf.
OpenSearch Amazon-Dienst	BatchGetCollection	API-Aufruf.
Amazon API Gateway	ExportApi	API-Aufruf.
AWS AppConfig	ValidateConfiguration	API-Aufruf.
Amazon AppFlow	RetrieveConnectorData	API-Aufruf.
Einblicke in CloudWatch Amazon-Anwendungen	UpdateApplicationDashboardConfiguration	API-Aufruf.
Amazon Athena	BatchGetNamedQuery	API-Aufruf.
Amazon Athena	BatchGetPreparedStatement	API-Aufruf.
Amazon Athena	BatchGetQueryExecution	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon Athena	CheckQueryCompatibility	API-Aufruf.
Amazon Athena	ExportNotebook	API-Aufruf.
AWS Auto Scaling	AreScalableTargetsRegistered	API-Aufruf.
AWS Auto Scaling	Test	API-Aufruf.
AWS Marketplace	SearchAgreements	API-Aufruf.
AWS Backup	CreateLegalHold	API-Aufruf.
AWS Backup	ExportBackupPlanTemplate	API-Aufruf.
AWS Backup gateway	TestHypervisorConfiguration	API-Aufruf.
AWS Billing and Cost Management	AWSPaymentInstrumentGateway. Holen	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.DescribeMakePaymentPage	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.DescribePaymentsDashboard	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetAccountPreferences	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetAdvancePaySummary	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetAsoBulkDownload	Konsolenaktion

Service	Ereignisname	Ereignistyp
AWS Billing and Cost Management	AWSPaymentPortalService.GetBillingContactAddress	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetDocuments	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetEligiblePaymentInstruments	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetEntitiesByIds	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetFundingDocuments	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetKybcValidationStatus	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetOneTimePasswordStatus	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetPaymentHistory	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetPaymentProfileByArn	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetPaymentProfileCurrencies	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetPaymentProfiles	Konsolenaktion

Service	Ereignisname	Ereignistyp
AWS Billing and Cost Management	AWSPaymentPortalService.GetPaymentProfileServiceProviders	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetPaymentsDue	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetRemittanceInformation	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetTaxInvoiceMetadata	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetTermsAndConditionsForProgramGroup	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetTransactionsHistory	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetUnappliedFunds	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPortalService.GetUnpaidInvoices	Konsolenaktion
AWS Billing and Cost Management	AWSPaymentPreferenceGateway.Holen	Konsolenaktion
AWS Billing and Cost Management	CancelBulkDownload	Konsolenaktion
AWS Billing and Cost Management	DownloadCommercialInvoice	Konsolenaktion

Service	Ereignisname	Ereignistyp
AWS Billing and Cost Management	DownloadCsv	Konsolenaktion
AWS Billing and Cost Management	DownloadDoc	Konsolenaktion
AWS Billing and Cost Management	CSV herunterladen ForBillin gPeriod	Konsolenaktion
AWS Billing and Cost Management	DownloadPaymentHistory	Konsolenaktion
AWS Billing and Cost Management	DownloadRegistrati onDocument	Konsolenaktion
AWS Billing and Cost Management	DownloadTaxInvoice	Konsolenaktion
AWS Billing and Cost Management	FindBankRedirectPaymentInst ruments	Konsolenaktion
AWS Billing and Cost Management	Finden Sie CSV ForBillin gPeriod	Konsolenaktion
AWS Billing and Cost Management	ValidateReportDestination	Konsolenaktion
AWS Billing and Cost Management	VerifyChinaPaymentEligibility	Konsolenaktion
Amazon Braket	SearchCompilations	API-Aufruf.
Amazon Braket	SearchDevices	API-Aufruf.
Amazon Braket	SearchQuantumTasks	API-Aufruf.
Amazon Connect Cases	BatchGetField	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon Connect Cases	SearchCases	API-Aufruf.
Amazon Connect Cases	SearchRelatedItems	API-Aufruf.
Amazon Chime	RetrieveDataExports	API-Aufruf.
Amazon Chime	SearchChannels	API-Aufruf.
Amazon-Chime-SDK-Identität	DeleteProfile	Serviceereignis
Amazon-Chime-SDK-Identität	DeleteWorkTalkAccount	Serviceereignis
AWS Saubere Räume	BatchGetSchema	API-Aufruf.
Amazon Cloud Directory	BatchRead	API-Aufruf.
Amazon Cloud Directory	LookupPolicy	API-Aufruf.
AWS CloudFormation	DetectStackDrift	API-Aufruf.
AWS CloudFormation	DetectStackResourceDrift	API-Aufruf.
AWS CloudFormation	DetectStackSetDrift	API-Aufruf.
AWS CloudFormation	EstimateTemplateCost	API-Aufruf.
AWS CloudFormation	ValidateTemplate	API-Aufruf.
AWS CloudShell	RedeemCode	API-Aufruf.
AWS CloudTrail	LookupEvents	API-Aufruf.
AWS CodeArtifact	ReadFromRepository	API-Aufruf.
AWS CodeArtifact	SearchPackages	API-Aufruf.
AWS CodeArtifact	VerifyResourcesExistForTagris	API-Aufruf.
AWS CodeBuild	BatchGetBuildBatches	API-Aufruf.

Service	Ereignisname	Ereignistyp
AWS CodeBuild	BatchGetBuilds	API-Aufruf.
AWS CodeBuild	BatchGetProjects	API-Aufruf.
AWS CodeBuild	BatchGetReportGroups	API-Aufruf.
AWS CodeBuild	BatchGetReports	API-Aufruf.
AWS CodeBuild	BatchPutCodeCoverages	API-Aufruf.
AWS CodeBuild	BatchPutTestCases	API-Aufruf.
AWS CodeBuild	RequestBadge	Serviceereignis
AWS CodeCommit	BatchDescribeMergeConflicts	API-Aufruf.
AWS CodeCommit	BatchGetCommits	API-Aufruf.
AWS CodeCommit	BatchGetPullRequests	API-Aufruf.
AWS CodeCommit	BatchGetRepositories	API-Aufruf.
AWS CodeCommit	EvaluatePullRequestApprovalRules	API-Aufruf.
AWS CodeCommit	GitPull	API-Aufruf.
AWS CodeDeploy	BatchGetApplicationRevisions	API-Aufruf.
AWS CodeDeploy	BatchGetApplications	API-Aufruf.
AWS CodeDeploy	BatchGetDeploymentGroups	API-Aufruf.
AWS CodeDeploy	BatchGetDeploymentInstances	API-Aufruf.
AWS CodeDeploy	BatchGetDeployments	API-Aufruf.
AWS CodeDeploy	BatchGetDeploymentTargets	API-Aufruf.

Service	Ereignisname	Ereignistyp
AWS CodeDeploy	BatchGetOnPremises Instances	API-Aufruf.
CodeGuru Amazon-Profiler	BatchGetFrameMetricData	API-Aufruf.
CodeGuru Amazon-Profiler	SubmitFeedback	API-Aufruf.
AWS CodePipeline	PollForJobs	API-Aufruf.
AWS CodePipeline	PollForThirdPartyJobs	API-Aufruf.
CodeConnections	StartAppRegistrationHandsha ke	API-Aufruf.
CodeConnections	Fangen Sie an AuthHands hake	API-Aufruf.
CodeConnections	ValidateHostWebhook	API-Aufruf.
Amazon CodeWhisperer	CreateCodeScan	API-Aufruf.
Amazon CodeWhisperer	CreateProfile	API-Aufruf.
Amazon CodeWhisperer	CreateUploadUrl	API-Aufruf.
Amazon CodeWhisperer	GenerateRecommendations	API-Aufruf.
Amazon CodeWhisperer	UpdateProfile	API-Aufruf.
Amazon Cognito Identity	LookupDeveloperIdentity	API-Aufruf.
Amazon-Cognito-Benutzerpool s	AdminGetDevice	API-Aufruf.
Amazon-Cognito-Benutzerpool s	AdminGetUser	API-Aufruf.
Amazon-Cognito-Benutzerpool s	AdminListDevices	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon-Cognito-Benutzerpools	AdminListGroupsForUser	API-Aufruf.
Amazon-Cognito-Benutzerpools	AdminListUserAuthEvents	API-Aufruf.
Amazon-Cognito-Benutzerpools	Beta_Authorize_GET	Serviceereignis
Amazon-Cognito-Benutzerpools	Confirm_GET	Serviceereignis
Amazon-Cognito-Benutzerpools	ConfirmForgotPassword_HOLEN	Serviceereignis
Amazon-Cognito-Benutzerpools	Error_GET	Serviceereignis
Amazon-Cognito-Benutzerpools	ForgotPassword_HOLEN	Serviceereignis
Amazon-Cognito-Benutzerpools	IntrospectToken	API-Aufruf.
Amazon-Cognito-Benutzerpools	Login_Error_POST	Serviceereignis
Amazon-Cognito-Benutzerpools	Login_GET	Serviceereignis
Amazon-Cognito-Benutzerpools	Mfa_GET	Serviceereignis
Amazon-Cognito-Benutzerpools	MfaOption_HOLEN	Serviceereignis
Amazon-Cognito-Benutzerpools	ResetPassword_HOLEN	Serviceereignis

Service	Ereignisname	Ereignistyp
Amazon-Cognito-Benutzerpools	Signup_GET	Serviceereignis
Amazon-Cognito-Benutzerpools	UserInfo_HOLEN	Serviceereignis
Amazon-Cognito-Benutzerpools	UserInfo_POSTEN	Serviceereignis
Amazon Cognito Sync	BulkPublish	API-Aufruf.
Amazon Comprehend	BatchContainsPiiEntities	API-Aufruf.
Amazon Comprehend	BatchDetectDominantLanguage	API-Aufruf.
Amazon Comprehend	BatchDetectEntities	API-Aufruf.
Amazon Comprehend	BatchDetectKeyPhrases	API-Aufruf.
Amazon Comprehend	BatchDetectPiiEntities	API-Aufruf.
Amazon Comprehend	BatchDetectSentiment	API-Aufruf.
Amazon Comprehend	BatchDetectSyntax	API-Aufruf.
Amazon Comprehend	BatchDetectTargetedSentiment	API-Aufruf.
Amazon Comprehend	ClassifyDocument	API-Aufruf.
Amazon Comprehend	ContainsPiiEntities	API-Aufruf.
Amazon Comprehend	DetectDominantLanguage	API-Aufruf.
Amazon Comprehend	DetectEntities	API-Aufruf.
Amazon Comprehend	DetectKeyPhrases	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon Comprehend	DetectPiiEntities	API-Aufruf.
Amazon Comprehend	DetectSentiment	API-Aufruf.
Amazon Comprehend	DetectSyntax	API-Aufruf.
Amazon Comprehend	DetectTargetedSentiment	API-Aufruf.
Amazon Comprehend	DetectToxicContent	API-Aufruf.
AWS Compute Optimizer	ExportAutoScalingGroupRecommendations	API-Aufruf.
AWS Compute Optimizer	Exportiert EBS VolumeRecommendations	API-Aufruf.
AWS Compute Optimizer	Exportieren Sie EC InstanceRecommendations	API-Aufruf.
AWS Compute Optimizer	Exportiert ECS ServiceRecommendations	API-Aufruf.
AWS Compute Optimizer	ExportLambdaFunctionRecommendations	API-Aufruf.
AWS Compute Optimizer	Exportieren Sie RDS InstanceRecommendations	API-Aufruf.
AWS Config	BatchGetAggregateResourceConfig	API-Aufruf.
AWS Config	BatchGetResourceConfig	API-Aufruf.
AWS Config	SelectAggregateResourceConfig	API-Aufruf.
AWS Config	SelectResourceConfig	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon Connect	AdminGetEmergencyAccessToken	API-Aufruf.
Amazon Connect	SearchQueues	API-Aufruf.
Amazon Connect	SearchRoutingProfiles	API-Aufruf.
Amazon Connect	SearchSecurityProfiles	API-Aufruf.
Amazon Connect	SearchUsers	API-Aufruf.
AWS Glue DataBrew	SendProjectSessionAction	API-Aufruf.
AWS Data Pipeline	EvaluateExpression	API-Aufruf.
AWS Data Pipeline	QueryObjects	API-Aufruf.
AWS Data Pipeline	ValidatePipelineDefinition	API-Aufruf.
AWS DataSync	VerifyResourcesExistForTagris	API-Aufruf.
AWS DeepLens	BatchGetDevice	API-Aufruf.
AWS DeepLens	BatchGetModel	API-Aufruf.
AWS DeepLens	BatchGetProject	API-Aufruf.
AWS DeepLens	CreateDeviceCertificates	API-Aufruf.
AWS DeepRacer	AdminGetAccountConfig	API-Aufruf.
AWS DeepRacer	AdminListAssociatedUsers	API-Aufruf.
AWS DeepRacer	TestRewardFunction	API-Aufruf.
AWS DeepRacer	VerifyResourcesExistForTagris	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon Detective	BatchGetGraphMembe rDatasources	API-Aufruf.
Amazon Detective	BatchGetMembership Datasources	API-Aufruf.
Amazon Detective	SearchGraph	API-Aufruf.
DevOpsAmazon-Guru	SearchInsights	API-Aufruf.
DevOpsAmazon-Guru	SearchOrganizationInsights	API-Aufruf.
AWS Database Migration Service	BatchStartRecommendations	API-Aufruf.
AWS Database Migration Service	ModifyRecommendation	API-Aufruf.
AWS Database Migration Service	StartRecommendations	API-Aufruf.
AWS Database Migration Service	VerifyResourcesExistForTagr is	API-Aufruf.
AWS Directory Service	VerifyTrust	API-Aufruf.
Amazon Elastic Compute Cloud	ConfirmProductInstance	API-Aufruf.
Amazon Elastic Compute Cloud	ReportInstanceStatus	API-Aufruf.
Amazon Elastic Container Registry	BatchCheckLayerAvailability	API-Aufruf.
Amazon Elastic Container Registry	BatchGetImage	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon Elastic Container Registry	BatchGetImageReferrer	API-Aufruf.
Amazon Elastic Container Registry	BatchGetRepository ScanningConfiguration	API-Aufruf.
Amazon Elastic Container Registry	DryRunEvent	Serviceereignis
Amazon Elastic Container Registry	PolicyExecutionEvent	Serviceereignis
Amazon Elastic Container Registry Public	BatchCheckLayerAvailability	API-Aufruf.
Amazon Elastic Container Service	DiscoverPollEndpoint	API-Aufruf.
Amazon Elastic Container Service	FindSubfleetRoute	API-Aufruf.
Amazon Elastic Container Service	ValidateResources	API-Aufruf.
Amazon Elastic Container Service	VerifyTaskSetsExist	API-Aufruf.
Amazon Elastic Kubernetes Service	AccessKubernetesApi	API-Aufruf.
AWS Elastic Beanstalk	CheckDNSAvailability	API-Aufruf.
AWS Elastic Beanstalk	RequestEnvironmentInfo	API-Aufruf.
AWS Elastic Beanstalk	RetrieveEnvironmentInfo	API-Aufruf.
AWS Elastic Beanstalk	ValidateConfigurationSettings	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon Elastic File System	NewClientConnection	Serviceereignis
Amazon Elastic File System	UpdateClientConnection	Serviceereignis
Amazon Elastic Transcoder	ReadJob	API-Aufruf.
Amazon Elastic Transcoder	ReadPipeline	API-Aufruf.
Amazon Elastic Transcoder	ReadPreset	API-Aufruf.
Amazon EventBridge	TestEventPattern	API-Aufruf.
Amazon EventBridge	TestScheduleExpression	API-Aufruf.
Amazon FinSpace API	BatchListCatalogNodesByDataset	API-Aufruf.
Amazon FinSpace API	BatchListNodesByDataset	API-Aufruf.
Amazon FinSpace API	BatchValidateAccess	API-Aufruf.
Amazon FinSpace API	CreateAuditRecordsQuery	API-Aufruf.
Amazon FinSpace API	SearchDatasets	API-Aufruf.
Amazon FinSpace API	SearchDatasetsV	API-Aufruf.
Amazon FinSpace API	ValidateIdToken	API-Aufruf.
AWS Firewall Manager	DisassociateAdminAccount	API-Aufruf.
Amazon Forecast	InvokeForecastEndpoint	API-Aufruf.
Amazon Forecast	QueryFeature	API-Aufruf.
Amazon Forecast	QueryForecast	API-Aufruf.
Amazon Forecast	QueryWhatIfForecast	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon Forecast	VerifyResourcesExistForTagris	API-Aufruf.
Amazon Fraud Detector	BatchGetVariable	API-Aufruf.
Amazon Fraud Detector	VerifyResourcesExistForTagris	API-Aufruf.
FreeRTOS	VerifyEmailAddress	API-Aufruf.
Amazon GameLift	RequestUploadCredentials	API-Aufruf.
Amazon GameLift	ResolveAlias	API-Aufruf.
Amazon GameLift	SearchGameSessions	API-Aufruf.
Amazon GameLift	ValidateMatchmakingRuleSet	API-Aufruf.
Amazon GameSparks	ExportSnapshot	API-Aufruf.
Amazon Location Service	BatchGetDevicePosition	API-Aufruf.
Amazon Location Service	CalculateRoute	API-Aufruf.
Amazon Location Service	CalculateRouteMatrix	API-Aufruf.
Amazon Location Service	SearchPlaceIndexForPosition	API-Aufruf.
Amazon Location Service	SearchPlaceIndexForSuggestions	API-Aufruf.
Amazon Location Service	SearchPlaceIndexForText	API-Aufruf.
Amazon S3 Glacier	InitiateJob	API-Aufruf.
AWS Glue	BatchGetBlueprints	API-Aufruf.
AWS Glue	BatchGetColumnStatisticsForTable	API-Aufruf.

Service	Ereignisname	Ereignistyp
AWS Glue	BatchGetCrawlers	API-Aufruf.
AWS Glue	BatchGetCustomEntityTypes	API-Aufruf.
AWS Glue	BatchGetDataQualityResult	API-Aufruf.
AWS Glue	BatchGetDevEndpoints	API-Aufruf.
AWS Glue	BatchGetJobs	API-Aufruf.
AWS Glue	BatchGetMLTransform	API-Aufruf.
AWS Glue	BatchGetPartition	API-Aufruf.
AWS Glue	BatchGetTriggers	API-Aufruf.
AWS Glue	BatchGetWorkflows	API-Aufruf.
AWS Glue	QueryJobRuns	API-Aufruf.
AWS Glue	QueryJobRunsAggregated	API-Aufruf.
AWS Glue	QueryJobs	API-Aufruf.
AWS Glue	QuerySchemaVersion Metadata	API-Aufruf.
AWS Glue	SearchTables	API-Aufruf.
AWS HealthLake	ReadResource	API-Aufruf.
AWS HealthLake	SearchWithGet	API-Aufruf.
AWS HealthLake	SearchWithPost	API-Aufruf.
AWS Identity and Access Management	GenerateCredentialReport	API-Aufruf.
AWS Identity and Access Management	GenerateOrganizationsAccess Report	API-Aufruf.

Service	Ereignisname	Ereignistyp
AWS Identity and Access Management	GenerateServiceLastAccessedDetails	API-Aufruf.
AWS Identity and Access Management	SimulateCustomPolicy	API-Aufruf.
AWS Identity and Access Management	SimulatePrincipalPolicy	API-Aufruf.
AWS Identitätsspeicher	IsMemberInGroups	API-Aufruf.
AWS Identity Store Auth	BatchGetSession	API-Aufruf.
Amazon Inspector Classic	PreviewAgents	API-Aufruf.
Amazon Inspector Classic	BatchGetAccountStatus	API-Aufruf.
Amazon Inspector Classic	BatchGetFreeTrialInfo	API-Aufruf.
Amazon Inspector Classic	BatchGetMember	API-Aufruf.
AWS Invoicing	ValidateDocumentDeliveryS3LocationInfo	API-Aufruf.
AWS IoT	SearchIndex	API-Aufruf.
AWS IoT	TestAuthorization	API-Aufruf.
AWS IoT	TestInvokeAuthorizer	API-Aufruf.
AWS IoT	ValidateSecurityProfileBehaviors	API-Aufruf.
AWS IoT Analytics	SampleChannelData	API-Aufruf.
AWS IoT SiteWise	GatewaysVerifyResourcesExistForTagrisInternal	API-Aufruf.
AWS IoT Things Graph	SearchEntities	API-Aufruf.

Service	Ereignisname	Ereignistyp
AWS IoT Things Graph	SearchFlowExecutions	API-Aufruf.
AWS IoT Things Graph	SearchFlowTemplates	API-Aufruf.
AWS IoT Things Graph	SearchSystemInstances	API-Aufruf.
AWS IoT Things Graph	SearchSystemTemplates	API-Aufruf.
AWS IoT Things Graph	SearchThings	API-Aufruf.
AWS IoT TwinMaker	ExecuteQuery	API-Aufruf.
AWS IoT Wireless	CreateNetworkAnalyzerConfiguration	API-Aufruf.
AWS IoT Wireless	DeleteNetworkAnalyzerConfiguration	API-Aufruf.
AWS IoT Wireless	DeregisterWirelessDevice	API-Aufruf.
Amazon Interactive Video Service	BatchGetChannel	API-Aufruf.
Amazon Interactive Video Service	BatchGetStreamKey	API-Aufruf.
Amazon Kendra	BatchGetDocumentStatus	API-Aufruf.
Amazon Kendra	Abfrage	API-Aufruf.
Amazon Managed Service für Apache Flink	DiscoverInputSchema	API-Aufruf.
AWS Key Management Service	Decrypt	API-Aufruf.
AWS Key Management Service	Encrypt	API-Aufruf.

Service	Ereignisname	Ereignistyp
AWS Key Management Service	GenerateDataKey	API-Aufruf.
AWS Key Management Service	GenerateDataKeyPair	API-Aufruf.
AWS Key Management Service	GenerateDataKeyPairWithoutPlaintext	API-Aufruf.
AWS Key Management Service	GenerateDataKeyWithoutPlaintext	API-Aufruf.
AWS Key Management Service	GenerateMac	API-Aufruf.
AWS Key Management Service	GenerateRandom	API-Aufruf.
AWS Key Management Service	ReEncrypt	API-Aufruf.
AWS Key Management Service	Sign	API-Aufruf.
AWS Key Management Service	Verify	API-Aufruf.
AWS Key Management Service	VerifyMac	API-Aufruf.
AWS Lake Formation	SearchDatabasesByLF-Tags	API-Aufruf.
AWS Lake Formation	SearchTablesByLF-Tags	API-Aufruf.
AWS Lake Formation	StartQueryPlanning	API-Aufruf.
Amazon Lex	BatchCreateCustomVocabularyItem	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon Lex	BatchDeleteCustomVocabularyItem	API-Aufruf.
Amazon Lex	BatchUpdateCustomVocabularyItem	API-Aufruf.
Amazon Lex	DeleteCustomVocabulary	API-Aufruf.
Amazon Lex	SearchAssociatedTranscripts	API-Aufruf.
Amazon Lightsail	GUI erstellen SessionAccessDetails	API-Aufruf.
Amazon Lightsail	DownloadDefaultKeyPair	API-Aufruf.
Amazon Lightsail	IsVpcPeered	API-Aufruf.
CloudWatch Amazon-Protokolle	FilterLogEvents	API-Aufruf.
Amazon Macie	BatchGetCustomDataIdentifiers	API-Aufruf.
Amazon Macie	UpdateFindingsFilter	API-Aufruf.
AWS Elemental MediaConnect	ManagedDescribeFlow	API-Aufruf.
AWS Elemental MediaConnect	PrivateDescribeFlowMeta	API-Aufruf.
AWS Application Migration Service	OperationalDescribeJobLogItems	API-Aufruf.
AWS Application Migration Service	OperationalDescribeJobs	API-Aufruf.

Service	Ereignisname	Ereignistyp
AWS Application Migration Service	OperationalDescribeReplicationConfigurationTemplates	API-Aufruf.
AWS Application Migration Service	OperationalDescribeSourceServer	API-Aufruf.
AWS Application Migration Service	OperationalGetLaunchConfiguration	API-Aufruf.
AWS Application Migration Service	OperationalListSourceServers	API-Aufruf.
AWS Application Migration Service	VerifyClientRoleForMgn	API-Aufruf.
AWS HealthOmics	VerifyResourceExists	API-Aufruf.
AWS HealthOmics	VerifyResourcesExistForTagris	API-Aufruf.
Amazon Polly	SynthesizeLongSpeech	API-Aufruf.
Amazon Polly	SynthesizeSpeech	API-Aufruf.
Amazon Polly	SynthesizeSpeechGet	API-Aufruf.
AWS Service zur Bereitstellung verwalteter privater Netzwerke	Ping	API-Aufruf.
AWS Proton	DeleteEnvironmentTemplateVersion	API-Aufruf.
AWS Proton	DeleteServiceTemplateVersion	API-Aufruf.
Amazon QLDB	ShowCatalog	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon QuickSight	GenerateEmbedUrlFo rAnonymousUser	API-Aufruf.
Amazon QuickSight	GenerateEmbedUrlFo rRegisteredUser	API-Aufruf.
Amazon QuickSight	QueryDatabase	Serviceereignis
Amazon QuickSight	SearchAnalyses	API-Aufruf.
Amazon QuickSight	SearchDashboards	API-Aufruf.
Amazon QuickSight	SearchDataSets	API-Aufruf.
Amazon QuickSight	SearchDataSources	API-Aufruf.
Amazon QuickSight	SearchFolders	API-Aufruf.
Amazon QuickSight	SearchGroups	API-Aufruf.
Amazon QuickSight	SearchUsers	API-Aufruf.
Amazon Relational Database Service	DownloadCompleteDB LogFile	API-Aufruf.
Amazon Relational Database Service	DB herunterladen LogFilePo rtion	API-Aufruf.
Amazon Rekognition	CompareFaces	API-Aufruf.
Amazon Rekognition	DetectCustomLabels	API-Aufruf.
Amazon Rekognition	DetectFaces	API-Aufruf.
Amazon Rekognition	DetectLabels	API-Aufruf.
Amazon Rekognition	DetectModerationLabels	API-Aufruf.
Amazon Rekognition	DetectProtectiveEquipment	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon Rekognition	DetectText	API-Aufruf.
Amazon Rekognition	RecognizeCelebrities	API-Aufruf.
Amazon Rekognition	SearchFaces	API-Aufruf.
Amazon Rekognition	SearchFacesByImage	API-Aufruf.
Amazon Rekognition	SearchUsers	API-Aufruf.
Amazon Rekognition	SearchUsersByImage	API-Aufruf.
AWS Ressourcen Explorer	BatchGetView	API-Aufruf.
AWS Ressourcen Explorer	Suche	API-Aufruf.
AWS Resource Groups	SearchResources	API-Aufruf.
AWS Resource Groups	ValidateResourceSharing	API-Aufruf.
AWS RoboMaker	BatchDescribeSimulationJob	API-Aufruf.
Amazon Route 53	TestDNSAnswer	API-Aufruf.
Amazon Route 53-Domains	checkAvailabilities	API-Aufruf.
Amazon Route 53-Domains	CheckDomainAvailability	API-Aufruf.
Amazon Route 53-Domains	checkDomainTransferability	API-Aufruf.
Amazon Route 53-Domains	CheckDomainTransferability	API-Aufruf.
Amazon Route 53-Domains	isEmailReachable	API-Aufruf.
Amazon Route 53-Domains	searchDomains	API-Aufruf.
Amazon Route 53-Domains	sendVerificationMessage	API-Aufruf.
Amazon Route 53-Domains	ViewBilling	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon Route 53-Domains	viewBilling	API-Aufruf.
Amazon CloudWatch RUM	BatchGetRumMetricDefinitions	API-Aufruf.
Amazon Simple Storage Service	echo	API-Aufruf.
Amazon Simple Storage Service	GenerateInventory	Serviceereignis
Amazon SageMaker	BatchDescribeModelPackage	API-Aufruf.
Amazon SageMaker	DeleteModelCard	API-Aufruf.
Amazon SageMaker	QueryLineage	API-Aufruf.
Amazon SageMaker	RenderUiTemplate	API-Aufruf.
Amazon SageMaker	Suche	API-Aufruf.
EventBridge Amazon-Schemas	ExportSchema	API-Aufruf.
EventBridge Amazon-Schemas	SearchSchemas	API-Aufruf.
Amazon SimpleDB	DomainMetadata	API-Aufruf.
AWS Secrets Manager	ValidateResourcePolicy	API-Aufruf.
AWS Service Catalog	ScanProvisionedProducts	API-Aufruf.
AWS Service Catalog	SearchProducts	API-Aufruf.
AWS Service Catalog	SearchProductsAsAdmin	API-Aufruf.
AWS Service Catalog	SearchProvisionedProducts	API-Aufruf.
Amazon SES	BatchGetMetricData	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon SES	TestRenderEmailTemplate	API-Aufruf.
Amazon SES	TestRenderTemplate	API-Aufruf.
Amazon Simple Notification Service	CheckIfPhoneNumber IsOptedOut	API-Aufruf.
AWS SQL Workbench	BatchGetNotebookCell	API-Aufruf.
AWS SQL Workbench	ExportNotebook	API-Aufruf.
Amazon EC2 Systems Manager	ExecuteApi	API-Aufruf.
AWS Systems Manager Incident Manager	DeleteContactChannel	API-Aufruf.
AWS IAM Identity Center	IsMemberInGroup	API-Aufruf.
AWS IAM Identity Center	SearchGroups	API-Aufruf.
AWS IAM Identity Center	SearchUsers	API-Aufruf.
AWS STS	AssumeRole	API-Aufruf.
AWS STS	AssumeRoleWithSAML	API-Aufruf.
AWS STS	AssumeRoleWithWebIdentity	API-Aufruf.
AWS STS	DecodeAuthorizationMessage	API-Aufruf.
AWS Steuereinstellungen	BatchGetTaxExemptions	API-Aufruf.
AWS WAFV2	CheckCapacity	API-Aufruf.
AWS WAFV2	GenerateMobileSdkR eleaseUrl	API-Aufruf.
AWS Well-Architected Tool	ExportLens	API-Aufruf.

Service	Ereignisname	Ereignistyp
AWS Well-Architected Tool	TagResource	API-Aufruf.
AWS Well-Architected Tool	UntagResource	API-Aufruf.
AWS Well-Architected Tool	UpdateGlobalSettings	API-Aufruf.
Amazon Connect Wisdom	QueryAssistant	API-Aufruf.
Amazon Connect Wisdom	SearchContent	API-Aufruf.
Amazon Connect Wisdom	SearchSessions	API-Aufruf.
Amazon WorkDocs	AbortDocumentVersionUpload	API-Aufruf.
Amazon WorkDocs	AddUsersToGroup	API-Aufruf.
Amazon WorkDocs	BatchGetUsers	API-Aufruf.
Amazon WorkDocs	CheckAlias	API-Aufruf.
Amazon WorkDocs	CompleteDocumentVersionUpload	API-Aufruf.
Amazon WorkDocs	CreateAnnotation	API-Aufruf.
Amazon WorkDocs	CreateComment	API-Aufruf.
Amazon WorkDocs	CreateFeedbackRequest	API-Aufruf.
Amazon WorkDocs	CreateFolder	API-Aufruf.
Amazon WorkDocs	CreateGroup	API-Aufruf.
Amazon WorkDocs	CreateShare	API-Aufruf.
Amazon WorkDocs	CreateUser	API-Aufruf.
Amazon WorkDocs	DeleteAnnotation	API-Aufruf.
Amazon WorkDocs	DeleteComment	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon WorkDocs	DeleteDocument	API-Aufruf.
Amazon WorkDocs	DeleteFeedbackRequest	API-Aufruf.
Amazon WorkDocs	DeleteFolder	API-Aufruf.
Amazon WorkDocs	DeleteFolderContents	API-Aufruf.
Amazon WorkDocs	DeleteGroup	API-Aufruf.
Amazon WorkDocs	DeleteOrganizationShare	API-Aufruf.
Amazon WorkDocs	DeleteUser	API-Aufruf.
Amazon WorkDocs	DownloadDocumentVersion	API-Aufruf.
Amazon WorkDocs	DownloadDocumentVersionUnderlays	API-Aufruf.
Amazon WorkDocs	InitiateDocumentVersionUpload	API-Aufruf.
Amazon WorkDocs	LogoutUser	API-Aufruf.
Amazon WorkDocs	PaginatedOrganizationActivity	API-Aufruf.
Amazon WorkDocs	PublishAnnotations	API-Aufruf.
Amazon WorkDocs	PublishComments	API-Aufruf.
Amazon WorkDocs	RestoreDocument	API-Aufruf.
Amazon WorkDocs	RestoreFolder	API-Aufruf.
Amazon WorkDocs	SearchGroups	API-Aufruf.
Amazon WorkDocs	SearchOrganizationUsers	API-Aufruf.
Amazon WorkDocs	TransferUserResources	API-Aufruf.

Service	Ereignisname	Ereignistyp
Amazon WorkDocs	UpdateAnnotation	API-Aufruf.
Amazon WorkDocs	UpdateComment	API-Aufruf.
Amazon WorkDocs	UpdateDocument	API-Aufruf.
Amazon WorkDocs	UpdateDocumentVersion	API-Aufruf.
Amazon WorkDocs	UpdateFolder	API-Aufruf.
Amazon WorkDocs	UpdateGroup	API-Aufruf.
Amazon WorkDocs	UpdateOrganization	API-Aufruf.
Amazon WorkDocs	UpdateUser	API-Aufruf.
Amazon WorkMail	AssumeImpersonationRole	API-Aufruf.
Amazon WorkMail	QueryDnsRecords	API-Aufruf.
Amazon WorkMail	SearchMembers	API-Aufruf.
Amazon WorkMail	TestAvailabilityConfiguration	API-Aufruf.
Amazon WorkMail	TestInboundMailFlowRules	API-Aufruf.
Amazon WorkMail	TestOutboundMailFlowRules	API-Aufruf.

EventBridge Referenz zu den Einzelheiten der Ereignisse

EventBridge sendet selbst die folgenden Ereignisse aus. Diese Ereignisse werden wie bei jedem anderen AWS Dienst automatisch an den Standard-Event-Bus gesendet.

Definitionen der Metadatenfelder, die in allen Ereignissen enthalten sind, finden Sie unterthe section called “Referenz der Ereignisstruktur”.

Themen

- Geplantes Ereignis

- Schema wurde erstellt
- Schemaversion wurde erstellt

Geplantes Ereignis

Nachfolgend finden Sie die Detailfelder für die `Scheduled Event` Veranstaltung.

Die `detail-type` Felder `source` und sind enthalten, da sie spezifische Werte für EventBridge Ereignisse enthalten. Definitionen der anderen Metadatenfelder, die in allen Ereignissen enthalten sind, finden Sie unterthe section called “Referenz der Ereignisstruktur”.

```
{
  . . .,
  "detail-type": "Scheduled Event",
  "source": "aws.events",
  . . .,
  "detail": {}
}
```

`detail-type`

Identifiziert den Ereignistyp.

Für dieses Ereignis ist dieser Wert`Scheduled Event`.

Erforderlich: Ja

`source`

Identifiziert den Service, aus dem das Ereignis stammt. Für EventBridge Ereignisse ist dieser Wert`aws.events`.

Erforderlich: Ja

`detail`

Ein JSON-Objekt, das Informationen zum Ereignis enthält. Der Service, der das Ereignis generiert, bestimmt den Inhalt dieses Feldes.

Erforderlich: Ja

Dieses Objekt enthält keine Pflichtfelder für `Scheduled Event` Ereignisse.

Example Beispiel für ein geplantes Ereignis

```
{
  "version": "0",
  "id": "89d1a02d-5ec7-412e-82f5-13505f849b41",
  "detail-type": "Scheduled Event",
  "source": "aws.events",
  "account": "123456789012",
  "time": "2016-12-30T18:44:49Z",
  "region": "us-east-1",
  "resources": ["arn:aws:events:us-east-1:123456789012:rule/SampleRule"],
  "detail": {}
}
```

Schema wurde erstellt

Im Folgenden finden Sie die Detailfelder für die `Schema Created` Veranstaltung.

Wenn ein Schema erstellt wird, EventBridge sendet `Schema Created` sowohl ein als auch ein `Schema Version Created` Ereignis.

Die `detail-type` Felder `source` und sind enthalten, da sie spezifische Werte für EventBridge Ereignisse enthalten. Definitionen der anderen Metadatenfelder, die in allen Ereignissen enthalten sind, finden Sie unterthe section called “Referenz der Ereignisstruktur”.

```
{
  . . .,
  "detail-type": "Schema Created",
  "source": "aws.schemas",
  . . .,
  "detail": {
    "SchemaName" : "String",
    "SchemaType" : "String",
    "RegistryName" : "String",
    "CreationDate" : "DateTime",
    "Version" : "Number"
  }
}
```

`detail-type`

Identifiziert den Ereignistyp.

Für dieses Ereignis ist dieser Wert`Schema Created`.

Erforderlich: Ja

`source`

Identifiziert den Service, aus dem das Ereignis stammt. Für EventBridge Ereignisse ist dieser Wert`aws.schemas`.

Erforderlich: Ja

`detail`

Ein JSON-Objekt, das Informationen zum Ereignis enthält. Der Service, der das Ereignis generiert, bestimmt den Inhalt dieses Feldes.

Erforderlich: Ja

Für dieses Ereignis beinhalten diese Daten:

`SchemaName`

Der Name des Schemas.

Erforderlich: Ja

`SchemaType`

Der Typ des Schemas.

Zulässige Werte: `OpenApi3` | `JSONSchemaDraft4`

Erforderlich: Ja

`RegistryName`

Der Name der Registrierung, die das Schema enthält

Erforderlich: Ja

`CreationDate`

Das Datum, an dem das Schema erstellt wurde.

Erforderlich: Ja

`Version`

Die Version des Schemas.

Für `Schema Created` Ereignisse wird dieser Wert immer sein`1`.

Erforderlich: Ja

Example Beispiel für ein vom Schema erstelltes Ereignis

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "Schema Created",
  "source": "aws.schemas",
  "account": "123456789012",
  "time": "2019-05-31T21:49:54Z",
  "region": "us-east-1",
  "resources": ["arn:aws:schemas:us-east-1::schema/myRegistry/mySchema"],
  "detail": {
    "SchemaName": "mySchema",
    "SchemaType": "OpenApi3",
    "RegistryName": "myRegistry",
    "CreationDate": "2019-11-29T20:08:55Z",
    "Version": "1"
  }
}
```

Schemaversion wurde erstellt

Im Folgenden finden Sie die Detailfelder für die `Schema Version Created` Veranstaltung.

Wenn ein Schema erstellt wird, EventBridge sendet `Schema Created` sowohl ein als auch ein `Schema Version Created` Ereignis.

Die `detail-type` Felder `source` und sind enthalten, da sie spezifische Werte für EventBridge Ereignisse enthalten. Definitionen der anderen Metadatenfelder, die in allen Ereignissen enthalten sind, finden Sie unterthe section called “Referenz der Ereignisstruktur”.

```
{
  . . .,
  "detail-type": "Schema Version Created",
  "source": "aws.schemas",
  . . .,
  "detail": {
    "SchemaName" : "String",
```

```
    "SchemaType" : "String",
    "RegistryName" : "String",
    "CreationDate" : "DateTime",
    "Version" : "Number"
  }
}
```

`detail-type`

Identifiziert den Ereignistyp.

Für dieses Ereignis ist dieser Wert`Schema Version Created`.

Erforderlich: Ja

`source`

Identifiziert den Service, aus dem das Ereignis stammt. Für EventBridge Ereignisse ist dieser Wert`aws.schemas`.

Erforderlich: Ja

`detail`

Ein JSON-Objekt, das Informationen zum Ereignis enthält. Der Service, der das Ereignis generiert, bestimmt den Inhalt dieses Feldes.

Erforderlich: Ja

Für dieses Ereignis beinhalten diese Daten:

`SchemaName`

Der Name des Schemas.

Erforderlich: Ja

`SchemaType`

Der Typ des Schemas.

Zulässige Werte: `OpenApi3` | `JSONSchemaDraft4`

Erforderlich: Ja

`RegistryName`

Der Name der Registrierung, die das Schema enthält

Erforderlich: Ja

`CreationDate`

Das Datum, an dem die Schemaversion erstellt wurde.

Erforderlich: Ja

`Version`

Die Version des Schemas.

Erforderlich: Ja

Example Beispiel für ein Ereignis „Schemaversion erstellt“

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "Schema Version Created",
  "source": "aws.schemas",
  "account": "123456789012",
  "time": "2019-05-31T21:49:54Z",
  "region": "us-east-1",
  "resources": ["arn:aws:schemas:us-east-1::schema/myRegistry/mySchema"],
  "detail": {
    "SchemaName": "mySchema",
    "SchemaType": "OpenApi3",
    "RegistryName": "myRegistry",
    "CreationDate": "2019-11-29T20:08:55Z",
    "Version": "5"
  }
}
```

Empfangen von Ereignissen von einem SaaS-Partner mit Amazon EventBridge

Wenn Sie Ereignisse aus SaaS-Partneranwendungen und -services empfangen möchten, benötigen Sie eine Partnerereignisquelle vom Partner. Anschließend können Sie einen Partner-Event-Bus erstellen und ihn der Partnerereignisquelle zuordnen.

Das folgende Video behandelt SaaS-Integrationen mit EventBridge: Software-as-a-Service (SaaS) - Partnern

Themen

Unterstützte SaaS-Partnerintegrationen

EventBridge unterstützt die folgenden SaaS-Partnerintegrationen:

- Adobe
- Auth0
- Blitline
- BUIDLHub
- Buildkite
- CleverTap
- Datadog
- Epsagon
- Freshworks
- Genesys
- GS2
- Karte
- Kloudless
- Mackerel
- MongoDB
- New Relic
- OneLogin
- Opsgenie

- PagerDuty
- Payshield
- SaaSus Platform
- SailPoint
- Saviynt
- Segment
- Shopify
- SignalFx
- Site24x7
- Stax
- Stripe
- SugarCRM
- SugarCRM
- Symantec
- Thundra
- TriggerMesh
- Whispir
- Zendesk
- Amazon-Verkäuferpartner-API

Partnerereignisquellen sind in den folgenden Regionen verfügbar.

Code	Name
`us-east-1`	USA Ost (Nord-Virginia)
`us-east-2`	USA Ost (Ohio)
`us-west-1`	USA West (Nordkalifornien)
`us-west-2`	USA West (Oregon)
`ca-central-1`	Kanada (Zentral)

Code	Name
eu-central-1	Europa (Frankfurt)
eu-central-2	Europa (Zürich)
eu-west-1	Europa (Irland)
eu-west-2	Europe (London)
eu-west-3	Europe (Paris)
eu-north-1	Europe (Stockholm)
eu-south-1	Europa (Milan)
eu-south-2	Europa (Spain)
af-south-1	Afrika (Kapstadt)
ap-south-1	Asien-Pazifik (Mumbai)
ap-south-2	Asien-Pazifik (Hyderabad)
ap-east-1	Asien-Pazifik (Hongkong)
ap-northeast-1	Asien-Pazifik (Tokio)
ap-northeast-2	Asien-Pazifik (Seoul)
ap-northeast-3	Asien-Pazifik (Osaka)
ap-southeast-1	Asien-Pazifik (Singapur)
ap-southeast-2	Asien-Pazifik (Sydney)
ap-southeast-3	Asien-Pazifik (Jakarta)
ap-southeast-4	Asien-Pazifik (Melbourne)
cn-north-1	China (Peking)

Code	Name
cn-northwest-1	China (Ningxia)
me-central-1	Naher Osten (VAE)
me-south-1	Naher Osten (Bahrain)
sa-east-1	Südamerika (São Paulo)
il-central-1	Israel (Tel Aviv)

Amazon EventBridge für den Empfang von Ereignissen aus einer SaaS-Integration konfigurieren

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Partner event sources (Partnerereignisquellen) aus.
3. Suchen Sie den gewünschten Partner und wählen Sie dann für diesen Partner Einrichten aus.
4. Wenn Sie Ihre Konto-ID in die Zwischenablage kopieren möchten, wählen Sie Kopieren aus.
5. Wählen Sie im Navigationsbereich Partner event sources (Partnerereignisquellen) aus.
6. Rufen Sie die Website des Partners auf und befolgen Sie die Anweisungen zum Erstellen einer Partnerereignisquelle mithilfe Ihrer Konto-ID. Die Ereignisquelle, die Sie erstellen, ist nur für Ihr Konto verfügbar.
7. Kehren Sie zur EventBridge Konsole zurück und wählen Sie im Navigationsbereich Partnerereignisquellen aus.
8. Wählen Sie die Schaltfläche neben der Partnerereignisquelle aus und klicken Sie dann auf Mit Event Bus verknüpfen.

   Der Status der Ereignisquelle wird von `Pending` in `Active` geändert. Der Name des Event Bus wird entsprechend dem Namen der Partnerereignisquelle aktualisiert. Sie können jetzt mit dem Erstellen von Regeln beginnen, die Ereignissen aus dieser Partnerereignisquelle entsprechen. Weitere Informationen finden Sie unter Erstellen einer Regel, die SaaS-Partnerereignissen entspricht.

> **Note**
>
> Alle Ereignisse, die von einem Partner in einer Partnerereignisquelle veröffentlicht wurden, die nicht mit einem Event Bus verknüpft wurde, werden sofort gelöscht. Diese Ereignisse werden im Ruhezustand nicht dauerhaft gespeichert. EventBridge

Erstellen einer Regel, die SaaS-Partnerereignissen entspricht

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen aus.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein.

   Eine Regel darf nicht denselben Namen wie eine andere Regel in derselben Region und auf demselben Event Bus haben.
5. Wählen Sie als Event bus (Event Bus) den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie AWS -Standard-Event-Bus aus. Wenn ein AWS -Service in Ihrem Konto ein Ereignis ausgibt, wird es stets an den Standard-Event-Bus Ihres Kontos weitergeleitet.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Weiter aus.
8. Wählen Sie für Event source (Ereignisquelle) Other (Andere) aus.
9. (Optional) Wählen Sie für Beispielereignisse den Ereignistyp aus.
10. Geben Sie für Ereignismuster ein JSON-Ereignismuster ein.
11. Wählen Sie Weiter aus.
12. Bei Target types (Zieltypen) wählen Sie AWS -Service aus.
13. Wählen Sie unter Ziel auswählen den AWS Service aus, an den Sie Informationen senden möchten, wenn ein Ereignis EventBridge erkannt wird, das dem Ereignismuster entspricht.
14. Die angezeigten Felder variieren je nach ausgewähltem Service. Geben Sie nach Bedarf Informationen ein, die für diesen Zieltyp spezifisch sind.

15. Für viele Zieltypen sind EventBridge Berechtigungen erforderlich, um Ereignisse an das Ziel zu senden. In diesen Fällen EventBridge kann die IAM-Rolle erstellt werden, die für die Ausführung Ihrer Regel erforderlich ist. Führen Sie eine der folgenden Aktionen aus:

    - Um automatisch eine IAM-Rolle zu erstellen, wählen Sie Create a new role for this specific resource (Eine neue Rolle für diese spezifische Ressource erstellen).
    - Wenn Sie eine zuvor erstellte IAM-Rolle verwenden möchten, wählen Sie Vorhandene Rolle verwenden und dann die vorhandene Rolle aus der Dropdown-Liste aus.

16. (Optional) Gehen Sie unter Additional settings (Weitere Einstellungen) wie folgt vor:

    a. Geben Sie für Maximum age of event (Maximales Alter des Ereignisses) einen Wert zwischen einer Minute (00:01) und 24 Stunden (24:00) ein.

    b. Geben Sie für Wiederholungsversuche eine Zahl zwischen 0 und 185 ein.

    c. Wählen Sie für Warteschlange für unzustellbare Briefe aus, ob Sie eine standardmäßige Amazon SQS SQS-Warteschlange als Warteschlange für unzustellbare Briefe verwenden möchten. EventBridge sendet Ereignisse, die dieser Regel entsprechen, an die Warteschlange für unzustellbare Briefe, wenn sie nicht erfolgreich an das Ziel zugestellt wurden. Führen Sie eine der folgenden Aktionen aus:

        - Klicken Sie auf Keine, um keine Warteschlange für unzustellbare Nachrichten zu verwenden.
        - Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange im aktuellen AWS - Konto als Warteschlange für unzustellbare Nachrichten und wählen Sie dann die Warteschlange aus der Dropdown-Liste aus.
        - Wählen Sie Wählen Sie eine Amazon SQS SQS-Warteschlange in einem anderen AWS Konto als Warteschlange für unzustellbare Briefe aus und geben Sie dann den ARN der Warteschlange ein, die Sie verwenden möchten. Sie müssen der Warteschlange eine ressourcenbasierte Richtlinie hinzufügen, die das Senden von Nachrichten an die EventBridge Warteschlange ermöglicht. Weitere Informationen finden Sie unter Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten.

17. (Optional) Wählen Sie Add another target (Weiteres Ziel hinzufügen) aus, um ein weiteres Ziel für diese Regel hinzuzufügen.

18. Wählen Sie Weiter aus.

19. (Optional) Geben Sie ein oder mehrere Tags für die Regel ein. Weitere Informationen finden Sie unter Amazon- EventBridge Tags.

20. Wählen Sie Weiter.

21. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Empfangen von Ereignissen mithilfe von Funktions-URLs AWS Lambda

> **Note**
>
> Damit unsere Partner auf den Inbound Webhook zugreifen können, erstellen wir in Ihrem AWS Konto ein Open Lambda, das auf Lambda-Anwendungsebene gesichert ist, indem die vom Drittanbieter gesendete Authentifizierungssignatur überprüft wird. Bitte überprüfen Sie diese Konfiguration mit Ihrem Sicherheitsteam. Weitere Informationen finden Sie unter Sicherheits- und Authentifizierungsmodell für Lambda-Funktions-URLs.

Ihr EventBridge Amazon-Eventbus kann eine durch eine AWS CloudFormation Vorlage erstellte AWS Lambda Funktions-URL verwenden, um Ereignisse von unterstützten SaaS-Anbietern zu empfangen. Bei Funktions-URLs werden die Ereignisdaten an eine Lambda-Funktion gesendet. Die Funktion konvertiert diese Daten dann in ein Ereignis, das von einem Event-Bus aufgenommen EventBridge und zur Verarbeitung an einen Event-Bus gesendet werden kann. Sobald sich das Ereignis in einem Event Bus befindet, können Sie Regeln verwenden, um die Ereignisse zu filtern, alle konfigurierten Eingabetransformationen anzuwenden und es dann an das richtige Ziel weiterzuleiten.

> **Note**
>
> Das Erstellen von Lambda-Funktions-URLs erhöht Ihre monatlichen Kosten. Weitere Informationen finden Sie unter AWS Lambda Preise.

Um eine Verbindung herzustellen EventBridge, wählen Sie zunächst den SaaS-Anbieter aus, mit dem Sie eine Verbindung einrichten möchten. Anschließend geben Sie ein Signing Secret an, das Sie bei diesem Anbieter erstellt haben, und wählen den EventBridge Event-Bus aus, an den Ereignisse gesendet werden sollen. Schließlich verwenden Sie eine AWS CloudFormation Vorlage und erstellen die erforderlichen Ressourcen, um die Verbindung herzustellen.

Die folgenden SaaS-Anbieter stehen derzeit für die EventBridge Verwendung mit Lambda-Funktions-URLs zur Verfügung:

- GitHub
- Twilio

Themen

Einrichten einer Verbindung zu GitHub

Schritt 1: Erstellen Sie den Stack AWS CloudFormation

Verwenden Sie zunächst die EventBridge Amazon-Konsole, um einen CloudFormation Stack zu erstellen:

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Schnelle Starts aus.
3. Wählen Sie unter Eingehende Webhooks mit Lambda-FURLs die Option Erste Schritte aus.
4. Wählen Sie unter GitHub die Option Einrichten aus.
5. Wählen Sie unter Schritt 1: Auswählen eines Event Bus einen Event Bus aus der Dropdown-Liste aus. Dieser Event Bus empfängt Daten von der Lambda-Funktions-URL, die Sie für GitHub bereitstellen. Sie können auch einen Event Bus erstellen, indem Sie Neuer Event Bus auswählen.
6. Wählen Sie unter Schritt 2: Einrichtung mithilfe CloudFormation die Option Neuer GitHub Webhook aus.
7. Wählen Sie Ich bestätige, dass der von mir erstellte eingehende Webhook öffentlich zugänglich ist. und Bestätigen aus.
8. Geben Sie einen Namen für den Stack ein.
9. Vergewissern Sie sich, dass unter Parameter der richtige Event Bus aufgeführt ist, und geben Sie dann ein sicheres Token für das GitHubWebhookSecret an. Weitere Informationen zum Erstellen eines sicheren Tokens finden Sie unter Einrichten Ihres geheimen Tokens in der GitHub-Dokumentation.

10. Wählen Sie unter Funktionen und Transformationen jede der folgenden Optionen aus:
    - Ich erkenne an, dass AWS CloudFormation dadurch IAM-Ressourcen entstehen könnten.
    - Ich erkenne an, dass AWS CloudFormation dadurch möglicherweise IAM-Ressourcen mit benutzerdefinierten Namen erstellt werden.
    - Ich erkenne an, dass dafür AWS CloudFormation möglicherweise die folgenden Funktionen erforderlich sind: **`CAPABILITY_AUTO_EXPAND`**
11. Wählen Sie Stack erstellen aus.

Schritt 2: Erstellen eines GitHub-Webhooks

Als Nächstes erstellen Sie den Webhook in GitHub. Sie benötigen sowohl das sichere Token als auch die Lambda-Funktions-URL, die Sie in Schritt 2 erstellt haben, um diesen Schritt abzuschließen. Weitere Informationen finden Sie unter Erstellen von Webhooks in der GitHub-Dokumentation.

Einrichten einer Verbindung zu einem Twilio

Schritt 1: Suchen Ihres Twilio-Authentifizierungstokens

Um eine Verbindung zwischen Twilio und einzurichten EventBridge, richten Sie zunächst die Verbindung Twilio mit dem Authentifizierungstoken oder Secret für Ihr Twilio Konto ein. Weitere Informationen finden Sie unter Authentifizierungstoken und deren Änderung in der Twilio-Dokumentation.

Schritt 2: Erstellen Sie den Stack AWS CloudFormation

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Schnelle Starts aus.
3. Wählen Sie unter Eingehende Webhooks mit Lambda-FURLs die Option Erste Schritte aus.
4. Wählen Sie unter Twilio die Option Einrichten aus.
5. Wählen Sie unter Schritt 1: Auswählen eines Event Bus einen Event Bus aus der Dropdown-Liste aus. Dieser Event Bus empfängt Daten von der Lambda-Funktions-URL, die Sie für Twilio bereitstellen. Sie können auch einen Event Bus erstellen, indem Sie Neuer Event Bus auswählen.
6. Wählen Sie unter Schritt 2: Einrichtung mithilfe CloudFormation die Option Neuer Twilio Webhook aus.

7. Wählen Sie Ich bestätige, dass der von mir erstellte eingehende Webhook öffentlich zugänglich ist. und Bestätigen aus.
8. Geben Sie einen Namen für den Stack ein.
9. Vergewissern Sie sich, dass unter Parameter der richtige Event Bus aufgeführt ist, und geben Sie dann das TwilioWebhookSecret ein, das Sie in Schritt 1 erstellt haben.
10. Wählen Sie unter Funktionen und Transformationen jede der folgenden Optionen aus:
    - Ich erkenne an, dass AWS CloudFormation dadurch IAM-Ressourcen entstehen könnten.
    - Ich erkenne an, dass AWS CloudFormation dadurch möglicherweise IAM-Ressourcen mit benutzerdefinierten Namen erstellt werden.
    - Ich erkenne an, dass dafür AWS CloudFormation möglicherweise die folgende Fähigkeit erforderlich ist: CAPABILITY_AUTO_EXPAND
11. Wählen Sie Stack erstellen aus.

Schritt 3: Erstellen eines Twilio-Webhooks

Nachdem Sie die Lambda-Funktions-URL eingerichtet haben, müssen Sie sie an Twilio weitergeben, damit die Ereignisdaten gesendet werden können. Weitere Informationen finden Sie unter Konfigurieren Ihrer öffentlichen URL mit Twilio in der Twilio-Dokumentation.

Aktualisieren eines Webhook-Secrets oder Authentifizierungstokens

Aktualisieren eines GitHub-Secrets

> **Note**
>
> GitHub unterstützt nicht zwei Secrets gleichzeitig. Es kann zu Ressourcenausfällen kommen, wenn das GitHub Geheimnis und das Geheimnis im AWS CloudFormation Stack nicht synchron sind. GitHubNachrichten, die gesendet werden, obwohl die Geheimnisse nicht synchron sind, schlagen aufgrund falscher Signaturen fehl. Warten Sie, bis die GitHub und die CloudFormation Secrets synchron sind, und versuchen Sie es dann erneut.

1. Erstellen Sie ein neues GitHub-Secret. Weitere Informationen finden Sie unter Verschlüsselte Secrets in der GitHub-Dokumentation.
2. Öffnen Sie die AWS CloudFormation Konsole unter https://console.aws.amazon.com/cloudformation.

3. Wählen Sie im Navigationsbereich Stacks aus.
4. Wählen Sie den Stack für den Webhook aus, der das Secret enthält, das Sie aktualisieren möchten.
5. Wählen Sie Aktualisieren.
6. Vergewissern Sie sich, dass Aktuelle Vorlage verwenden ausgewählt ist, und klicken Sie auf Weiter.
7. Deaktivieren Sie unter GitHubWebhookSecretVorhandenen Wert verwenden, geben Sie das neue GitHub Geheimnis ein, das Sie in Schritt 1 erstellt haben, und wählen Sie Weiter aus.
8. Wählen Sie Weiter aus.
9. Wählen Sie Stack aktualisieren aus.

Es kann bis zu einer Stunde dauern, bis das Secret verbreitet ist. Wenn Sie diese Ausfallzeit reduzieren möchten, können Sie den Lambda-Ausführungskontext aktualisieren.

Aktualisieren eines Twilio-Secrets

> **Note**
>
> Twilio unterstützt nicht zwei Secrets gleichzeitig. Es kann zu Ressourcenausfällen kommen, wenn das Twilio Geheimnis und das Geheimnis im AWS CloudFormation Stapel nicht synchron sind. TwilioNachrichten, die gesendet werden, obwohl die Geheimnisse nicht synchron sind, schlagen aufgrund falscher Signaturen fehl. Warten Sie, bis die Twilio und CloudFormation Secrets synchron sind, und versuchen Sie es dann erneut.

1. Erstellen Sie ein neues Twilio-Secret. Weitere Informationen finden Sie unter Authentifizierungstoken und deren Änderung in der Twilio-Dokumentation.
2. Öffnen Sie die AWS CloudFormation Konsole unter https://console.aws.amazon.com/cloudformation.
3. Wählen Sie im Navigationsbereich Stacks aus.
4. Wählen Sie den Stack für den Webhook aus, der das Secret enthält, das Sie aktualisieren möchten.
5. Wählen Sie Aktualisieren.
6. Vergewissern Sie sich, dass Aktuelle Vorlage verwenden ausgewählt ist, und klicken Sie auf Weiter.

7. Deaktivieren Sie unter TwilioWebhookSecretVorhandenen Wert verwenden, geben Sie das neue Twilio Geheimnis ein, das Sie in Schritt 1 erstellt haben, und wählen Sie Weiter aus.
8. Wählen Sie Weiter aus.
9. Wählen Sie Stack aktualisieren aus.

Es kann bis zu einer Stunde dauern, bis das Secret verbreitet ist. Wenn Sie diese Ausfallzeit reduzieren möchten, können Sie den Lambda-Ausführungskontext aktualisieren.

Aktualisieren einer Lambda-Funktion

Die Lambda-Funktion, die vom CloudFormation Stack erstellt wird, erstellt den grundlegenden Webhook. Wenn Sie die Lambda-Funktion für einen bestimmten Anwendungsfall anpassen möchten, z. B. für die benutzerdefinierte Protokollierung, verwenden Sie die CloudFormation Konsole, um auf die Funktion zuzugreifen, und verwenden Sie dann die Lambda-Konsole, um den Lambda-Funktionscode zu aktualisieren.

Zugreifen auf die Lambda-Funktion

1. Öffnen Sie die AWS CloudFormation Konsole unter https://console.aws.amazon.com/cloudformation.
2. Wählen Sie im Navigationsbereich Stacks aus.
3. Wählen Sie den Stack für den Webhook aus, der die Lambda-Funktion enthält, die Sie aktualisieren möchten.
4. Wählen Sie die Registerkarte Ressourcen aus.
5. Wenn Sie die Lambda-Funktion in der Lambda-Konsole öffnen möchten, wählen Sie unter Physikalische ID die ID der Lambda-Funktion aus.

Nachdem Sie auf die Lambda-Funktion zugegriffen haben, verwenden Sie die Lambda-Konsole, um den Funktionscode zu aktualisieren.

Aktualisieren des Lambda-Funktionscodes

1. Wählen Sie unter Aktionen die Option Exportfunktion aus.
2. Wählen Sie Bereitstellungspaket herunterladen aus und speichern Sie die Datei auf Ihrem Computer.

3. Entpacken Sie die ZIP-Datei des Bereitstellungspakets, aktualisieren Sie die Datei `app.py` und komprimieren Sie das aktualisierte Bereitstellungspaket. Achten Sie dabei darauf, dass alle Dateien in der ursprünglichen ZIP-Datei enthalten sind.
4. Wählen Sie in der Lambda-Konsole die Registerkarte Code aus.
5. Wählen Sie unter Codequelle die Option Upload von aus.
6. Wählen Sie .zip-Datei und dann Hochladen.
   - Wählen Sie in der Dateiauswahl die aktualisierte Datei aus, wählen Sie Öffnen und dann Speichern.
7. Wählen Sie unter Aktionen die Option Neue Version veröffentlichen aus.

Verfügbare Ereignistypen

Die folgenden Ereignistypen werden derzeit von CloudFormation Event-Bussen unterstützt:

- GitHub— Alle Ereignistypen werden unterstützt.
- Twilio – Webhooks nach dem Ereignis werden unterstützt.

Kontingente, Fehlercodes und Wiederholen der Zustellung

Kontingente

Die Anzahl der eingehenden Anfragen an den Webhook wird durch die zugrunde liegenden AWS Dienste begrenzt. Die folgende Tabelle enthält die entsprechenden Kontingente.

Service	Kontingent
AWS Lambda	Standard: 10 gleichzeitige Ausführungen Weitere Informationen zu Kontingenten, einschließlich dem Anfordern von Kontingenterhöhungen, finden Sie unter Lambda-Kontingente.
AWS Secrets Manager	Standard: 5 000 Anforderungen pro Sekunde Weitere Informationen zu Kontingenten, einschließlich dem Anfordern von Kontingenterhöhungen, finden Sie unter AWS Secrets Manager -Kontingente.

Service	Kontingent
	Note Die Anzahl der Anforderungen pro Sekunde wird mithilfe des AWS Secrets Manager -Python-Caching-Clients minimiert.
Amazon EventBridge	256 KB maximale Eintragsgröße für PutEvents Aktionen. EventBridge setzt regionsspezifische Preiskontingente durch. Weitere Informationen finden Sie unter ???.

Fehlercodes

Jeder AWS Dienst gibt bestimmte Fehlercodes zurück, wenn Fehler auftreten. Die folgende Tabelle enthält die entsprechenden Fehlercodes.

Service	Fehlercode	Beschreibung
AWS Lambda	429 „“ TooManyRequestsException	Das Kontingent für gleichzeitige Ausführungen wurde überschritten.
AWS Secrets Manager	500 “Internal Server Error”	Das Kontingent für Anfoderungen pro Sekunde wurde überschritten.
Amazon EventBridge	500 “Internal Server Error”	Das Ratenkontingent für die Region wurde überschritten.

Erneute Zustellung von Ereignissen

Wenn Fehler auftreten, können Sie die Zustellung der betroffenen Ereignisse wiederholen. Jeder SaaS-Anbieter hat unterschiedliche Wiederholungsverfahren.

GitHub

Verwenden Sie die GitHub-Webhooks-API, um den Zustellstatus jedes Webhook-Aufrufs zu überprüfen und das Ereignis bei Bedarf erneut zuzustellen. Weitere Informationen finden Sie in der folgenden GitHub-Dokumentation:

- Organisation – Stellen Sie eine Zustellung für einen Organisations-Webhook erneut zu.
- Repository – Stellen Sie eine Zustellung für einen Repository-Webhook erneut zu.
- App – Stellen Sie eine Zustellung für einen App-Webhook erneut zu.

Twilio

Twilio-Benutzer können die Optionen für die Wiederholung von Ereignissen mithilfe von Verbindungsüberschreibungen anpassen. Weitere Informationen finden Sie unter Webhooks (HTTP-Callbacks): Verbindungsüberschreibungen in der Twilio-Dokumentation.

Empfangen von Ereignissen von Salesforce

Sie können Amazon EventBridge auf folgende Salesforce Weise verwenden, um Ereignisse zu empfangen:

- Indem Sie die Salesforce's Event Bus Relay-Funktion verwenden, um Ereignisse direkt auf einem EventBridge Partner-Eventbus zu empfangen.
- Indem Sie einen Flow in Amazon konfigurieren AppFlow, der Salesforce als Datenquelle verwendet wird. Amazon sendet AppFlow dann Salesforce Ereignisse EventBridge mithilfe eines Partner-Event-Busses an.

Sie können mithilfe von API-Zielen Ereignisinformationen an Salesforce senden. Sobald das Ereignis an Salesforce gesendet wurde, kann es von Flows oder Apex-Auslösern verarbeitet werden. Weitere Informationen zum Einrichten eines Salesforce-API-Ziels finden Sie unter ???.

Themen

- Empfangen von Ereignissen von Salesforce mithilfe von Event Bus Relay
- Empfangen von Ereignissen Salesforce über Amazon AppFlow

Empfangen von Ereignissen von Salesforce mithilfe von Event Bus Relay

Schritt 1: Richten Sie Salesforce Event Bus Relay und eine EventBridge Partner-Eventquelle ein

Wenn Sie eine Event-Relay-Konfiguration auf erstellenSalesforce, Salesforce wird eine Partner-Eventquelle mit dem Status „Ausstehend“ erstellt. EventBridge

So konfigurieren Sie Salesforce Event Bus Relay

1. Einrichten eines REST-API-Tools
2. (Optional) Definieren eines Plattformereignisses
3. Erstellen eines Kanals für ein benutzerdefiniertes Plattformereignis
4. Erstellen eines Kanalmitglieds zum Zuordnen des benutzerdefinierten Plattformereignisses
5. Erstellen einer benannten Anmeldeinformation
6. Erstellen einer Ereignis-Relay-Konfiguration

Schritt 2: Aktivieren Sie die Salesforce Partner-Eventquelle in der EventBridge Konsole und starten Sie das Event-Relay

1. Öffnen Sie in der EventBridge Konsole die Seite Partner-Eventquellen.
2. Wählen Sie die Salesforce-Partnerereignisquelle aus, die Sie in Schritt 1 erstellt haben.
3. Wählen Sie Mit Event Bus verknüpfen aus.
4. Überprüfen Sie den Namen des Partner-Event-Bus.
5. Wählen Sie Associate aus.
6. Starten das Ereignis-Relays

Nachdem Sie das Event Bus Relay eingerichtet und gestartet und die Partnerereignisquelle konfiguriert haben, können Sie eine EventBridge Regel erstellen, die auf Ereignisse reagiert, um die Daten zu filtern und an ein Ziel zu senden.

Empfangen von Ereignissen Salesforce über Amazon AppFlow

Amazon AppFlow kapselt Ereignisse aus einem Salesforce EventBridge Ereignisumschlag. Das folgende Beispiel zeigt ein Salesforce Ereignis, das von einem EventBridge Partner-Event-Bus empfangen wurde.

```
{
    "version": "0",
    "id": "5c42b99e-e005-43b3-c744-07990c50d2cc",
    "detail-type": "AccountChangeEvent",
    "source": "aws.partner/appflow.test/salesforce.com/364228160620/CustomSF-Source-
Final",
    "account": "000000000",
    "time": "2020-08-20T18:25:51Z",
    "region": "us-west-2",
    "resources": [],
    "detail": {
        "ChangeEventHeader": {
            "commitNumber": 248197218874,
            "commitUser": "0056g000003XW7AAAW",
            "sequenceNumber": 1,
            "entityName": "Account",
            "changeType": "UPDATE",
            "changedFields": [
                "LastModifiedDate",
```

```
                "Region__c"
            ],
            "changeOrigin": "com/salesforce/api/soap/49.0;client=SfdcInternalAPI/",
            "transactionKey": "000035af-b239-0581-9f14-461e4187de11",
            "commitTimestamp": 1597947935000,
            "recordIds": [
                "0016g00000MLhLeAAL"
            ]
        },
        "LastModifiedDate": "2020-08-20T18:25:35.000Z",
        "Region__c": "America"
    }
}
```

Schritt 1: Amazon AppFlow für die Verwendung Salesforce als Partner-Eventquelle konfigurieren

Um Ereignisse an zu senden EventBridge, müssen Sie zunächst Amazon so konfigurieren, AppFlow dass es Salesforce als Partnerereignisquelle verwendet wird.

1. Wählen Sie in der AppFlowAmazon-Konsole Create Flow aus.
2. Geben Sie im Abschnitt Flow-Details unter Flow-Name einen Namen für Ihren Flow ein.
3. (Optional) Geben Sie eine Beschreibung für den Flow ein und wählen Sie dann Weiter aus.
4. Wählen Sie unter Quelldetails die Option Salesforce im Dropdown-Menü Quellname und dann Verbinden aus, um eine neue Verbindung zu erstellen.
5. Wählen Sie im Dialogfeld Mit Salesforce verbinden entweder Produktion oder Sandbox für die Salesforce-Umgebung aus.
6. Geben Sie im Feld Verbindungsname einen eindeutigen Namen für die Verbindung ein und klicken Sie dann auf Weiter.
7. Führen Sie im Dialogfeld Salesforce folgende Schritte aus:
    a. Geben Sie Ihre Salesforce-Anmeldeinformationen ein, mit denen Sie sich bei Salesforce anmelden möchten.
    b. Wählen Sie Salesforce Ereignisse für die Datentypen aus, die Amazon verarbeiten AppFlow soll.
8. Wählen Sie im Drop-down-Menü Salesforce Ereignis auswählen den Ereignistyp aus, an den gesendet EventBridge werden soll.
9. Wählen Sie Amazon für ein Ziel aus EventBridge.
10. Wählen Sie Neue Partnerereignisquelle erstellen aus.

11. (Optional) Geben Sie ein eindeutiges Suffix für die Partnerereignisquelle an.
12. Wählen Sie Partnerereignisquelle generieren aus.
13. Wählen Sie einen Amazon-S3-Bucket aus, um Ereignisnutzlastdateien zu speichern, die größer als 256 KB sind.
14. Stellen Sie sicher, dass im Abschnitt Flow-Auslöser die Option Flow bei Ereignis ausführen ausgewählt ist. Diese Einstellung stellt sicher, dass der Flow ausgeführt wird, wenn ein neues Salesforce-Ereignis eintritt.
15. Wählen Sie Weiter aus.
16. Wählen Sie für die Feldzuordnung die Option Alle Felder direkt zuordnen aus. Alternativ können Sie die Felder, die für Sie von Interesse sind, aus der Liste Quellfeldname auswählen.

    Weitere Informationen zur Feldzuordnung finden Sie unter Zuordnen von Datenfeldern.
17. Wählen Sie Weiter aus.
18. (Optional) Konfigurieren Sie Filter für Datenfelder in Amazon AppFlow.
19. Wählen Sie Weiter aus.
20. Überprüfen Sie die Einstellungen und wählen Sie dann Flow erstellen aus.

Wenn der Ablauf konfiguriert ist, AppFlow erstellt Amazon eine neue Partner-Eventquelle, die Sie dann mit einem Partner-Event-Bus in Ihrem Konto verknüpfen müssen.

Schritt 2: Für EventBridge den Empfang von Salesforce Ereignissen konfigurieren

Stellen Sie sicher, dass der AppFlow Amazon-Flow, der durch Salesforce Ereignisse ausgelöst wird, die EventBridge als Ziel angegeben sind, konfiguriert ist, bevor Sie den Anweisungen in diesem Abschnitt folgen.

Um den Empfang EventBridge von Salesforce Ereignissen zu konfigurieren

1. Öffnen Sie in der EventBridge Konsole die Seite Partnerereignisquellen.
2. Wählen Sie die Salesforce-Partnerereignisquelle aus, die Sie in Schritt 1 erstellt haben.
3. Wählen Sie Mit Event Bus verknüpfen aus.
4. Überprüfen Sie den Namen des Partner-Event-Bus.
5. Wählen Sie Associate aus.
6. Öffnen Sie in der AppFlow Amazon-Konsole den von Ihnen erstellten Flow und wählen Sie Flow aktivieren.

7. Öffnen Sie die Seite Regeln in der EventBridge Konsole.
8. Wählen Sie Regel erstellen aus.
9. Geben Sie einen eindeutigen Namen für die Regel ein.
10. Wählen Sie die Option Ereignismuster im Abschnitt Muster definieren aus.
11. Wählen Sie für Event-Matching-Muster die Option Vordefiniertes Muster nach Service aus.
12. Wählen Sie im Bereich Serviceanbieter die Option Alle Ereignisse aus.
13. Wählen Sie für Event Bus auswählen die Option Benutzerdefinierter oder Partner-Event-Bus aus.
14. Wählen Sie den Event-Bus aus, den Sie mit der Eventquelle des AppFlow Amazon-Partners verknüpft haben.
15. Wählen Sie für Ausgewählte Ziele den AWS Service aus, der ausgeführt werden soll, wenn die Regel ausgeführt wird. Eine Regel kann bis zu fünf Ziele haben.
16. Wählen Sie Erstellen.

Der Zielservice empfängt alle für Ihr Konto konfigurierten Salesforce-Ereignisse. Wenn Sie die Ereignisse filtern oder einige Ereignisse an verschiedene Ziele senden möchten, können Sie eine inhaltsbasierte Filterung mit Ereignismustern verwenden.

> **Note**
>
> Bei Veranstaltungen, die größer als 256 KB sind, sendet Amazon AppFlow nicht die gesamte Veranstaltung an EventBridge. Stattdessen AppFlow fügt Amazon das Ereignis in einen S3-Bucket in Ihrem Konto ein und sendet dann ein Ereignis EventBridge mit einem Zeiger auf den Amazon S3 S3-Bucket an. Sie können den Zeiger verwenden, um das vollständige Ereignis aus dem Bucket abzurufen.

Debuggen der Ereigniszustellung

Probleme bei der Zustellung von Ereignissen können schwer zu identifizieren sein. Sie EventBridge bieten einige Möglichkeiten zum Debuggen und zur Behebung von Fehlern bei der Zustellung von Ereignissen.

Wie EventBridge versucht man erneut, Ereignisse zuzustellen

Manchmal wird ein Ereignis nicht erfolgreich an das in einer Regel angegebene Ziel zugestellt. Das kann zum Beispiel passieren:

- Wenn die Zielressource nicht verfügbar ist
- Aufgrund von Netzwerkbedingungen

Wenn ein Ereignis aufgrund von wiederherstellbaren Fehlern nicht erfolgreich an ein Ziel übermittelt werden kann, wird EventBridge erneut versucht, das Ereignis zu senden. Die Dauer der Versuche und die Anzahl der Wiederholungsversuche legen Sie in den Einstellungen der Wiederholungsrichtlinie für das Ziel fest. Standardmäßig wird EventBridge erneut versucht, das Ereignis 24 Stunden lang und bis zu 185 Mal zu senden, wobei ein exponentielles Back-Off und Jitter oder eine zufällige Verzögerung auftreten.

Wenn ein Ereignis nicht zugestellt wird, nachdem alle Wiederholungsversuche ausgeschöpft sind, wird das Ereignis gelöscht und es wird EventBridge nicht weiter verarbeitet.

Verwenden von Warteschlangen mit unzustellbaren Buchstaben zur Verarbeitung nicht zugestellter Ereignisse

Wenn Sie verhindern möchten, dass Ereignisse verloren gehen, nachdem sie nicht an ein Ziel zugestellt werden, können Sie eine Warteschlange für unzustellbare Nachrichten konfigurieren und alle fehlgeschlagenen Ereignisse zur späteren Verarbeitung an diese senden.

EventBridge DLQs sind standardmäßige Amazon SQS-Warteschlangen, in denen EventBridge Ereignisse gespeichert werden, die nicht erfolgreich an ein Ziel übermittelt werden konnten. Wenn Sie eine Regel erstellen und ein Ziel hinzufügen, können Sie wählen, ob Sie eine Warteschlange für unzustellbare Nachrichten verwenden möchten oder nicht. Wenn Sie eine Warteschlange für unzustellbare Nachrichten konfigurieren, können Sie alle Ereignisse beibehalten, die nicht erfolgreich zugestellt wurden. Anschließend können Sie das Problem lösen, das zur fehlgeschlagenen Ereigniszustellung geführt hat, und die Ereignisse zu einem späteren Zeitpunkt verarbeiten.

Wenn Sie eine DLQ für ein Ziel einer Regel konfigurieren, EventBridge sendet die Ereignisse mit fehlgeschlagenen Aufrufen an die ausgewählte Amazon SQS SQS-Warteschlange.

Ereignisfehler werden auf unterschiedliche Weise behandelt. Einige Ereignisse werden gelöscht oder ohne Wiederholungsversuche an eine Warteschlange für unzustellbare Nachrichten gesendet. Bei Fehlern, die auf fehlende Berechtigungen für ein Ziel oder auf eine nicht mehr vorhandene Zielressource zurückzuführen sind, schlagen alle Wiederholungsversuche fehl, bis eine Maßnahme zur Behebung des zugrundeliegenden Problems ergriffen wird. Anstatt es erneut zu versuchen, werden diese Ereignisse direkt an den DLQ EventBridge gesendet, falls Sie einen haben.

Wenn eine Ereigniszustellung fehlschlägt, EventBridge veröffentlicht es ein Ereignis in CloudWatch Amazon-Statistiken, das darauf hinweist, dass ein Ziel `invocation` fehlgeschlagen ist. Wenn Sie einen DLQ verwenden, werden zusätzliche Messwerte an CloudWatch einschließlich `InvocationsSentToDLQ` und `InvocationsFailedToBeSentToDLQ` gesendet.

Sie können auch DLQs für Ereignisbusse angeben, wenn Sie Ereignisse im AWS KMS Kundenverwaltete Schlüssel Ruhezustand verschlüsseln. Weitere Informationen finden Sie unter ???.

Jede Nachricht in Ihrer Warteschlange für unzustellbare Nachrichten enthält die folgenden benutzerdefinierten Attribute:

- `RULE_ARN`
- `TARGET_ARN`
- `ERROR_CODE`

  Im Folgenden finden Sie ein Beispiel für die Fehlercodes, die eine Warteschlange für unzustellbare Nachrichten zurückgeben kann:

  - `CONNECTION_FAILURE`
  - `CROSS_ACCOUNT_INGESTION_FAILED`
  - `CROSS_REGION_INGESTION_FAILED`
  - `ERROR_FROM_TARGET`
  - `EVENTS_IN_BATCH_REQUEST_REJECTED`
  - `EVENTS_IN_BATCH_REQUEST_REJECTED`
  - `FAILED_TO_ASSUME_ROLE`
  - `INTERNAL_ERROR`
  - `INVALID_JSON`
  - `INVALID_PARAMETER`
  - `NO_PERMISSIONS`
  - `NO_RESOURCE`
  - `RESOURCE_ALREADY_EXISTS`
  - `RESOURCE_LIMIT_EXCEEDED`
  - `RESOURCE_MODIFICATION_COLLISION`
  - `SDK_CLIENT_ERROR`
  - `THIRD_ACCOUNT_HOP_DETECTED`

  - THIRD_REGION_HOP_DETECTED
  - THROTTLING
  - TIMEOUT
  - TRANSIENT_ASSUME_ROLE
  - UNKNOWN
- ERROR_MESSAGE
- EXHAUSTED_RETRY_CONDITION

  Die folgenden Bedingungen können zurückgegeben werden:

  - MaximumRetryAttempts
  - MaximumEventAgeInSeconds
- RETRY_ATTEMPTS

Das folgende Video zeigt das Einrichten von Warteschlangen für unzustellbare Nachrichten: Verwenden von Warteschlangen für unzustellbare Nachrichten

Themen

- Überlegungen zum Verwenden einer Warteschlange für unzustellbare Nachrichten
- Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten
- So senden Sie Ereignisse aus einer Warteschlange für unzustellbare Nachrichten erneut

Überlegungen zum Verwenden einer Warteschlange für unzustellbare Nachrichten

Beachten Sie bei der Konfiguration eines DLQ für Folgendes. EventBridge

- Es werden nur Standardwarteschlangen unterstützt. Sie können keine FIFO-Warteschlange für eine DLQ in verwenden. EventBridge
- EventBridge schließt Ereignismetadaten und Nachrichtenattribute in die Nachricht ein, darunter: den Fehlercode, die Fehlermeldung, die Bedingung für erschöpfte Wiederholungen, den Regel-ARN, Wiederholungsversuche und den Ziel-ARN. Sie können diese Werte verwenden, um ein Ereignis und die Ursache des Fehlers zu identifizieren.
- Berechtigungen für Warteschlangen für unzustellbare Nachrichten im selben Konto:

  - Wenn Sie einer Regel mithilfe der Konsole ein Ziel hinzufügen und eine Amazon SQS SQS-Warteschlange in demselben Konto auswählen, wird eine ressourcenbasierte Richtlinie, die EventBridge Zugriff auf die Warteschlange gewährt, für Sie an die Warteschlange angehängt.
  - Wenn Sie den `PutTargets` Betrieb der EventBridge API verwenden, um ein Ziel für eine Regel hinzuzufügen oder zu aktualisieren, und Sie eine Amazon SQS SQS-Warteschlange in demselben Konto auswählen, müssen Sie der ausgewählten Warteschlange manuell Berechtigungen erteilen. Weitere Informationen hierzu finden Sie unter Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten.
- Berechtigungen für die Verwendung von Amazon SQS SQS-Warteschlangen von einem anderen AWS Konto aus.
  - Wenn Sie eine Regel über die Konsole erstellen, werden Warteschlangen aus anderen Konten nicht zur Auswahl angezeigt. Sie müssen den ARN für die Warteschlange in dem anderen Konto angeben und dann manuell eine ressourcenbasierte Richtlinie anhängen, um der Warteschlange eine Berechtigung zu gewähren. Weitere Informationen hierzu finden Sie unter Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten.
  - Wenn Sie mithilfe der API eine Regel erstellen, müssen Sie manuell eine ressourcenbasierte Richtlinie an die SQS-Warteschlange in einem anderen Konto anhängen, die als Warteschlange für unzustellbare Nachrichten verwendet wird. Weitere Informationen hierzu finden Sie unter Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten.
- Die von Ihnen verwendete Amazon-SQS-Warteschlange muss sich in derselben Region befinden, in der Sie die Regel erstellen.

Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten

Um Ereignisse erfolgreich an die Warteschlange weiterzuleiten, EventBridge müssen Sie über die entsprechende Genehmigung verfügen. Wenn Sie über die EventBridge Konsole eine DLQ angeben, werden die Berechtigungen automatisch hinzugefügt. Dies umfasst:

- Wenn Sie eine DLQ für ein Ziel einer Regel konfigurieren.
- Wenn Sie eine DLQ für einen Event-Bus konfigurieren, für den Sie angegeben haben, dass ein EventBridge verwendet wird, um Ereignisse im AWS KMS Kundenverwalteter Schlüssel Ruhezustand zu verschlüsseln.

  Weitere Informationen finden Sie unter ???.

Wenn Sie mithilfe der API einen DLQ angeben oder eine Warteschlange verwenden, die sich in einem anderen AWS Konto befindet, müssen Sie manuell eine ressourcenbasierte Richtlinie erstellen, die die erforderlichen Berechtigungen gewährt, und sie dann an die Warteschlange anhängen.

Beispiel für Ziel-Warteschlangenberechtigungen mit unerlaubten Buchstaben

Die folgende ressourcenbasierte Richtlinie zeigt, wie Sie die erforderlichen Berechtigungen für EventBridge das Senden von Ereignisnachrichten an eine Amazon SQS SQS-Warteschlange gewähren. Das Richtlinienbeispiel erteilt dem EventBridge Service die Erlaubnis, den `SendMessage` Vorgang zum Senden von Nachrichten an eine Warteschlange mit dem Namen "DLQ" MyEvent zu verwenden. Die Warteschlange muss sich in der Region US-West-2 im AWS Konto 123456789012 befinden. Die `Condition` Anweisung erlaubt nur Anfragen, die von einer Regel mit dem Namen "MyTestRule" stammen, die in der Region us-west-2 im AWS Konto 123456789012 erstellt wurde.

```
{
  "Sid": "Dead-letter queue permissions",
  "Effect": "Allow",
  "Principal": {
     "Service": "events.amazonaws.com"
  },
  "Action": "sqs:SendMessage",
  "Resource": "arn:aws:sqs:us-west-2:123456789012:MyEventDLQ",
  "Condition": {
    "ArnEquals": {
      "aws:SourceArn": "arn:aws:events:us-west-2:123456789012:rule/MyTestRule"
    }
  }
}
```

Beispiel für Berechtigungen für eine Warteschlange mit unverschlüsselten Buchstaben im Event-Bus

Die folgende ressourcenbasierte Richtlinie zeigt, wie die erforderlichen Berechtigungen erteilt werden, wenn ein DLQ für einen Eventbus angegeben wird. `aws:SourceArn`Gibt in diesem Fall den ARN des Event-Busses an, der die Ereignisse an den DLQ sendet. Auch in diesem Beispiel muss sich die Warteschlange in derselben Region wie der Event-Bus befinden.

```
{
  "Sid": "Dead-letter queue permissions",
  "Effect": "Allow",
  "Principal": {
```

```
      "Service": "events.amazonaws.com"
    },
    "Action": "sqs:SendMessage",
    "Resource": "arn:aws:sqs:region:account-id:queue-name",
    "Condition": {
      "ArnEquals": {
        "aws:SourceArn": "arn:aws:events:region:account-id:event-bus/event-bus-arn"
      }
    }
}
```

Wenn Sie die Richtlinie an die Warteschlange anhängen möchten, verwenden Sie die Amazon-SQS-Konsole, öffnen Sie die Warteschlange, wählen Sie dann die Zugriffsrichtlinie aus und bearbeiten Sie die Richtlinie. Sie können auch die AWS CLI verwenden. Weitere Informationen hierzu finden Sie unter Amazon-SQS-Berechtigungen.

So senden Sie Ereignisse aus einer Warteschlange für unzustellbare Nachrichten erneut

Sie können Nachrichten auf zwei Arten aus einer Warteschlange für unzustellbare Nachrichten verschieben:

- Vermeiden des Schreibens von Amazon-SQS-Verbraucherlogik – Legen Sie Ihre Warteschlange für unzustellbare Nachrichten als Ereignisquelle für die Lambda-Funktion fest, um Ihre Warteschlange für unzustellbare Nachrichten zu leeren.
- Amazon SQS SQS-Verbraucherlogik schreiben — Verwenden Sie die Amazon SQS SQS-API, AWS das SDK oder AWS CLI schreiben Sie eine benutzerdefinierte Verbraucherlogik für das Abfragen, Verarbeiten und Löschen der Nachrichten in der DLQ.

EventBridge Amazon-Ereignismuster

Ereignismuster haben dieselbe Struktur wie die Ereignisse, mit denen sie übereinstimmen. Regeln verwenden Ereignismuster, um Ereignisse auszuwählen und sie an Ziele zu senden. Ein Ereignismuster stimmt entweder mit einem Ereignis überein oder nicht.

> **Important**
>
> In ist es möglich EventBridge, Regeln zu erstellen, die zu higher-than-expected Gebühren und Drosselungen führen können. Sie können beispielsweise versehentlich eine Regel erstellen, die zu einer Endlosschleife führt, bei der eine Regel rekursiv ohne Ende ausgelöst wird. Angenommen, Sie haben eine Regel erstellt, um zu erkennen, dass sich ACLs in einem Amazon-S3-Bucket geändert haben, und Software auszulösen, um sie in den gewünschten Status zu ändern. Ist die Regel nicht sorgfältig geschrieben, löst die anschließende Änderung der ACLs die Regel erneut aus, wodurch eine Endlosschleife entsteht.
> Anleitungen zum Schreiben präziser Regeln und Ereignismuster zur Minimierung solcher unerwarteter Ergebnisse finden Sie unter ??? und ???.

Das folgende Video befasst sich mit den Grundlagen von Ereignismustern: So filtern Sie Ereignisse

Themen

- Erstellen von Ereignismustern
- Beispiele für Ereignisse und Ereignismuster
- Übereinstimmende Nullwerte und leere Zeichenketten in EventBridge Amazon-Ereignismustern
- Arrays in EventBridge Amazon-Ereignismustern
- Inhaltsfilterung in EventBridge Amazon-Ereignismustern
- Testen eines Ereignismusters mithilfe der EventBridge Sandbox
- Bewährte Methoden bei der Definition von EventBridge Amazon-Eventmustern

Das folgende Ereignis zeigt ein einfaches AWS Ereignis aus Amazon EC2.

```
{
```

```
  "version": "0",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "EC2 Instance State-change Notification",
  "source": "aws.ec2",
  "account": "111122223333",
  "time": "2017-12-22T18:43:48Z",
  "region": "us-west-1",
  "resources": [
    "arn:aws:ec2:us-west-1:123456789012:instance/i-1234567890abcdef0"
  ],
  "detail": {
    "instance-id": "i-1234567890abcdef0",
    "state": "terminated"
  }
}
```

Das folgende Ereignismuster verarbeitet alle `instance-termination`-Ereignisse von Amazon EC2.

```
{
  "source": ["aws.ec2"],
  "detail-type": ["EC2 Instance State-change Notification"],
  "detail": {
    "state": ["terminated"]
  }
}
```

Erstellen von Ereignismustern

Wenn Sie ein Ereignismuster erstellen möchten, geben Sie die Felder eines Ereignisses an, denen das Ereignismuster entsprechen soll. Geben Sie nur die Felder an, die Sie für den Abgleich verwenden. Das vorherige Beispiel für ein Ereignismuster liefert nur Werte für drei Felder: die Felder der obersten Ebene `"source"` und und `"detail-type"` das `"state"` Feld innerhalb des `"detail"` Objektfeldes. EventBridge ignoriert alle anderen Felder des Ereignisses, wenn die Regel angewendet wird.

Damit ein Ereignismuster mit einem Ereignis übereinstimmt, muss das Ereignis alle im Ereignismuster aufgeführten Feldnamen enthalten. Die Feldnamen müssen im Ereignis auch mit derselben verschachtelten Struktur angezeigt werden.

Wenn Sie Ereignismuster für den Abgleich von Ereignissen schreiben, können Sie die `TestEventPattern`-API oder den CLI-Befehl `test-event-pattern` verwenden, um zu testen, ob Ihr Muster den korrekten Ereignissen entspricht. Weitere Informationen finden Sie unter TestEventPattern.

Abgleichen von Ereigniswerten

In einem Ereignismuster befindet sich der abzugleichende Wert in einem JSON-Array, das von eckigen Klammern („[“, „]“) umgeben ist, sodass Sie mehrere Werte angeben können. Um beispielsweise Ereignisse aus Amazon EC2 oder abzugleichen AWS Fargate, könnten Sie das folgende Muster verwenden, das Ereignisse abgleicht, bei denen der Wert für das `"source"` Feld entweder `"aws.ec2"` oder `"aws.fargate"` ist.

```
{
    "source": ["aws.ec2", "aws.fargate"]
}
```

Überlegungen zur Erstellung von Ereignismustern

Im Folgenden finden Sie einige Dinge, die Sie beim Erstellen Ihrer Ereignismuster berücksichtigen sollten:

- EventBridge ignoriert die Felder im Ereignis, die nicht im Ereignismuster enthalten sind. Das hat zur Folge, dass es einen `"*": "*"`-Platzhalter bei Feldern gibt, die nicht im Ereignismuster vorkommen.
- Für die von Ereignismustern abgeglichenen Werte gelten JSON-Regeln. Sie können in Anführungszeichen (") gesetzte Zeichenfolgen sowie Zahlen und die Schlüsselwörter `true`, `false` und `null` verwenden.
- EventBridge Verwendet für Zeichenketten den exakten character-by-character Abgleich ohne Umschaltung der Groß- und Kleinschreibung oder andere Normalisierung von Zeichenketten.
- EventBridge Verwendet für Zahlen die Zeichenkettendarstellung. 300, 300,0 und 3,0e2 werden z. B. nicht gleich behandelt.
- Wenn mehrere Muster für dasselbe JSON-Feld angegeben sind, wird EventBridge nur das letzte verwendet.
- Beachten Sie, dass beim EventBridge Kompilieren von Ereignismustern zur Verwendung ein Punkt (.) als Verbindungszeichen verwendet wird.

Das bedeutet, EventBridge dass die folgenden Ereignismuster als identisch behandelt werden:

```
## has no dots in keys
{ "detail" : { "state": { "status": [ "running" ] } } }

## has dots in keys
{ "detail" : { "state.status": [ "running" ] } }
```

Und dass beide Ereignismuster den folgenden beiden Ereignissen entsprechen:

```
## has no dots in keys
{ "detail" : { "state": { "status": "running" } } }

## has dots in keys
{ "detail" : { "state.status": "running"  } }
```

> **Note**
>
> Dies beschreibt EventBridge das aktuelle Verhalten und man sollte sich nicht darauf verlassen, dass es sich nicht ändert.

- Ereignismuster, die doppelte Felder enthalten, sind ungültig. Wenn ein Muster doppelte Felder enthält, wird EventBridge nur der endgültige Feldwert berücksichtigt.

  Beispielsweise stimmen die folgenden Ereignismuster mit demselben Ereignis überein:

```
## has duplicate keys
{
  "source": ["aws.s3"],
  "source": ["aws.sns"],
  "detail-type": ["AWS API Call via CloudTrail"],
  "detail":  {
      "eventSource": ["s3.amazonaws.com"],
      "eventSource": ["sns.amazonaws.com"]
  }
}

## has unique keys
{
  "source": ["aws.sns"],
```

```
  "detail-type": ["AWS API Call via CloudTrail"],
  "detail": { "eventSource": ["sns.amazonaws.com"] }
}
```

Und EventBridge behandelt die folgenden beiden Ereignisse als identisch:

```
## has duplicate keys
{
  "source": ["aws.s3"],
  "source": ["aws.sns"],
  "detail-type": ["AWS API Call via CloudTrail"],
  "detail":  [
    {
      "eventSource": ["s3.amazonaws.com"],
      "eventSource": ["sns.amazonaws.com"]
    }
  ]
}

## has unique keys
{
  "source": ["aws.sns"],
  "detail-type": ["AWS API Call via CloudTrail"],
  "detail": [
    { "eventSource": ["sns.amazonaws.com"] }
  ]
}
```

> **Note**
>
> Dies beschreibt EventBridge das aktuelle Verhalten und man sollte sich nicht darauf verlassen, dass es sich nicht ändert.

Vergleichsoperationen zur Verwendung in Ereignismustern

Im Folgenden finden Sie eine Zusammenfassung aller Vergleichsoperatoren, die in verfügbar sind EventBridge.

Vergleichsoperatoren funktionieren nur in Blattknoten, mit Ausnahme von `$or` und `anything-but`.

Vergleich	Beispiel	Regelsyntax
And	Location is "New York" and Day is "Monday"	`"Location": [ "New York" ], "Day": ["Monday"]`
Alles außer	State ist ein beliebiger Wert außer „initialisieren“.	`"state": [ { "anything-but": "initializing" } ]`
Alles andere als (beginnt mit)	Die Region liegt nicht in den USA.	`"Region": [ { "anything-but": {"prefix": "us-" } } ]`
Alles andere als (endet mit)	FileName endet nicht mit der Erweiterung.png.	`"FileName": [ { "anything-but": { "suffix": ".png" } } ]`
Alles außer (Groß- und Kleinschreibung ignorieren)	State ist ein beliebiger Wert außer „initialisieren“ oder einer anderen Variante der Groß-/Kleinschreibung, z. B. „INITIALIZING“.	`"state": : [{ "anything-but": { "equals-ignore-case": "initializing" }}]}`
Alles, nur nicht die Verwendung eines Platzhalters	FileName ist kein Dateipfad, der Folgendes beinhaltet. `/lib/`	`"FilePath" : [{ "anything-but": { "wildcard": "*/lib/*" }}]`
Beginnt mit	Die Region befindet sich in den USA.	`"Region": [ {"prefix": "us-" } ]`
Beginnt mit (Groß- und Kleinschreibung ignorieren)	Der Dienstname beginnt mit den Buchstaben „eventb“, unabhängig von der Groß- und Kleinschreibung.	`{"service" : [{ "prefix": { "equals-ignore-case": "eventb" }}]}`
Leer	LastName ist leer.	`"LastName": [""]`
Gleichheitszeichen	Name is "Alice"	`"Name": [ "Alice" ]`

Vergleich	Beispiel	Regelsyntax
Gleich (Groß-/Kleinschreibung ignorieren)	Name is "Alice"	`"Name": [ { "equals-ignore-case": "alice" } ]`
Endet mit	FileName endet mit der Erweiterung.png	`"FileName": [ { "suffix": ".png" } ]`
Endet mit (Groß- und Kleinschreibung ignorieren)	Der Dienstname endet mit den Buchstaben „tbridge“ oder einer anderen Variante der Groß-/Kleinschreibung, z. B. „TBRIDGE“.	`{"service" : [{ "suffix": { "equals-ignore-case": "tBridge" }}]}`
Vorhanden	ProductName existiert	`"ProductName": [ { "exists": true } ]`
Nicht vorhanden	ProductName existiert nicht	`"ProductName": [ { "exists": false } ]`
Nicht	Weather is anything but "Raining"	`"Weather": [ { "anything-but": [ "Raining" ] } ]`
Null	UserID is null	`"UserID": [ null ]`
Numerisch (ist gleich)	Price is 100	`"Price": [ { "numeric": [ "=", 100 ] } ]`
Numerisch (Bereich)	Price is more than 10, and less than or equal to 20	`"Price": [ { "numeric": [ ">", 10, "<=", 20 ] } ]`
Oder	PaymentType ist „Kredit“ oder „Lastschrift“	`"PaymentType": [ "Credit", "Debit"]`
Oder (mehrere Felder)	Location is "New York", or Day is "Monday".	`"$or": [ { "Location": [ "New York" ] }, { "Day": [ "Monday" ] } ]`

Vergleich	Beispiel	Regelsyntax
Platzhalter	Jede Datei mit der Erweiterung .png, die sich im Ordner „dir“ befindet	`"FileName": [ { "wildcard": "dir/*.png" } ]`

Beispiele für Ereignisse und Ereignismuster

Sie können alle JSON-Datentypen und -Werte verwenden, um Ereignisse abzugleichen. Die folgenden Beispiele zeigen Ereignisse und die Ereignismuster, die ihnen entsprechen.

Feldabgleich

Sie können anhand des Werts eines Felds einen Abgleich vornehmen. Betrachten Sie das folgende Ereignis von Amazon EC2 Auto Scaling.

```
{
  "version": "0",
  "id": "3e3c153a-8339-4e30-8c35-687ebef853fe",
  "detail-type": "EC2 Instance Launch Successful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2015-11-11T21:31:47Z",
  "region": "us-east-1",
  "resources": [],
  "detail": {
    "eventVersion": "",
    "responseElements": null
  }
}
```

Für das vorhergehende Ereignis können Sie das entsprechende Feld `"responseElements"` verwenden.

```
{
  "source": ["aws.autoscaling"],
  "detail-type": ["EC2 Instance Launch Successful"],
  "detail": {
   "responseElements": [null]
  }
}
```

Wertabgleich

Betrachten Sie das folgende Ereignis von Amazon Macie, das gekürzt wurde.

```
{
  "version": "0",
  "id": "0948ba87-d3b8-c6d4-f2da-732a1example",
  "detail-type": "Macie Finding",
  "source": "aws.macie",
  "account": "123456789012",
  "time": "2021-04-29T23:12:15Z",
  "region":"us-east-1",
  "resources": [

  ],
  "detail": {
    "schemaVersion": "1.0",
    "id": "64b917aa-3843-014c-91d8-937ffexample",
    "accountId": "123456789012",
    "partition": "aws",
    "region": "us-east-1",
    "type": "Policy:IAMUser/S3BucketEncryptionDisabled",
    "title": "Encryption is disabled for the S3 bucket",
    "description": "Encryption is disabled for the Amazon S3 bucket. The data in the
 bucket isn’t encrypted
        using server-side encryption.",
    "severity": {
        "score": 1,
        "description": "Low"
    },
    "createdAt": "2021-04-29T15:46:02Z",
    "updatedAt": "2021-04-29T23:12:15Z",
    "count": 2,
.
.
.
```

Das folgende Ereignismuster entspricht jedem Ereignis mit einem Schweregrad von 1 und einer Anzahl von 2.

```
{
  "source": ["aws.macie"],
  "detail-type": ["Macie Finding"],
```

```
  "detail": {
    "severity": {
      "score": [1]
    },
    "count":[2]
  }
}
```

Übereinstimmende Nullwerte und leere Zeichenketten in EventBridge Amazon-Ereignismustern

> **Important**
>
> In ist es möglich EventBridge, Regeln zu erstellen, die zu higher-than-expected Gebühren und Drosselungen führen können. Sie können beispielsweise versehentlich eine Regel erstellen, die zu einer Endlosschleife führt, bei der eine Regel rekursiv ohne Ende ausgelöst wird. Angenommen, Sie haben eine Regel erstellt, um zu erkennen, dass sich ACLs in einem Amazon-S3-Bucket geändert haben, und Software auszulösen, um sie in den gewünschten Status zu ändern. Ist die Regel nicht sorgfältig geschrieben, löst die anschließende Änderung der ACLs die Regel erneut aus, wodurch eine Endlosschleife entsteht.
> Anleitungen zum Schreiben präziser Regeln und Ereignismuster zur Minimierung solcher unerwarteter Ergebnisse finden Sie unter ??? und ???.

Sie können ein Ereignismuster erstellen, das einem Feld in einem Ereignis entspricht, das einen Nullwert oder eine leere Zeichenfolge hat. Betrachten Sie das folgende -Beispielereignis:

Informieren Sie sich über bewährte Methoden zur Vermeidung unerwarteter Gebühren und Drosselung.

```
{
  "version": "0",
  "id": "3e3c153a-8339-4e30-8c35-687ebef853fe",
  "detail-type": "EC2 Instance Launch Successful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2015-11-11T21:31:47Z",
  "region": "us-east-1",
  "resources": [
   ],
  "detail": {
    "eventVersion": "",
    "responseElements": null
   }
}
```

Wenn Sie Ereignisse abgleichen möchten, bei denen der Wert von `eventVersion` eine leere Zeichenfolge ist, verwenden Sie das folgende Ereignismuster, das dem vorhergehenden Ereignis entspricht.

```
{
  "detail": {
     "eventVersion": [""]
  }
}
```

Wenn Sie Ereignisse abgleichen möchten, bei denen der Wert von `responseElements` null ist, verwenden Sie das folgende Ereignismuster, das dem vorhergehenden Ereignis entspricht.

```
{
  "detail": {
     "responseElements": [null]
  }
}
```

> **Note**
>
> Nullwerte und leere Zeichenfolgen sind beim Musterabgleich nicht austauschbar. Ein Ereignismuster, das mit leeren Zeichenfolgen übereinstimmt, entspricht nicht den Werten von `null`.

Arrays in EventBridge Amazon-Ereignismustern

Der Wert jedes Felds in einem Ereignismuster ist ein Array, das einen oder mehrere Werte enthält. Ein Ereignismuster entspricht dem Ereignis, wenn einer der Werte im Array mit dem Wert im Ereignis übereinstimmt. Ist der Wert im Ereignis ein Array, dann stimmt das Ereignismuster überein, wenn die Schnittmenge aus dem Ereignismuster-Array und dem Ereignis-Array nicht leer ist.

> **Important**
>
> In ist es möglich EventBridge, Regeln zu erstellen, die zu higher-than-expected Gebühren und Drosselungen führen können. Sie können beispielsweise versehentlich eine Regel erstellen, die zu einer Endlosschleife führt, bei der eine Regel rekursiv ohne Ende ausgelöst wird. Angenommen, Sie haben eine Regel erstellt, um zu erkennen, dass sich ACLs in einem Amazon-S3-Bucket geändert haben, und Software auszulösen, um sie in den gewünschten Status zu ändern. Ist die Regel nicht sorgfältig geschrieben, löst die anschließende Änderung der ACLs die Regel erneut aus, wodurch eine Endlosschleife entsteht.
> Anleitungen zum Schreiben präziser Regeln und Ereignismuster zur Minimierung solcher unerwarteter Ergebnisse finden Sie unter ??? und ???.

Betrachten Sie beispielsweise ein Ereignismuster, das das folgende Feld enthält.

```
"resources": [
   "arn:aws:ec2:us-east-1:123456789012:instance/i-b188560f",
   "arn:aws:ec2:us-east-1:111122223333:instance/i-b188560f",
   "arn:aws:ec2:us-east-1:444455556666:instance/i-b188560f",
]
```

Das vorhergehende Ereignismuster stimmt mit einem Ereignis überein, das das folgende Feld enthält, da das erste Element im Ereignismuster-Array dem zweiten Element im Ereignis-Array entspricht.

```
"resources": [
   "arn:aws:autoscaling:us-east-1:123456789012:autoScalingGroup:eb56d16b-bbf0-401d-
b893-d5978ed4a025:autoScalingGroupName/ASGTerminate",
   "arn:aws:ec2:us-east-1:123456789012:instance/i-b188560f"
]
```

Inhaltsfilterung in EventBridge Amazon-Ereignismustern

Amazon EventBridge unterstützt die deklarative Inhaltsfilterung mithilfe von Ereignismustern. Bei der Inhaltsfilterung können Sie komplexe Ereignismuster schreiben, die nur unter sehr spezifischen Bedingungen Ereignissen entsprechen. Sie können beispielsweise ein Ereignismuster erstellen, das einem Ereignis entspricht, wenn:

- Ein Feld des Ereignisses innerhalb eines bestimmten numerischen Bereichs liegt
- Das Ereignis von einer bestimmten IP-Adresse stammt
- Ein bestimmtes Feld im Ereignis-JSON nicht vorhanden ist

> **Important**
>
> In ist es möglich EventBridge, Regeln zu erstellen, die zu higher-than-expected Gebühren und Drosselungen führen können. Sie können beispielsweise versehentlich eine Regel erstellen, die zu einer Endlosschleife führt, bei der eine Regel rekursiv ohne Ende ausgelöst wird. Angenommen, Sie haben eine Regel erstellt, um zu erkennen, dass sich ACLs in einem Amazon-S3-Bucket geändert haben, und Software auszulösen, um sie in den gewünschten Status zu ändern. Ist die Regel nicht sorgfältig geschrieben, löst die anschließende Änderung der ACLs die Regel erneut aus, wodurch eine Endlosschleife entsteht.
> Anleitungen zum Schreiben präziser Regeln und Ereignismuster zur Minimierung solcher unerwarteter Ergebnisse finden Sie unter ??? und ???.

Filtertypen

- Übereinstimmung mit einem Präfix
- Suffix-Abgleich
- „Alles außer“-Abgleich
- Numerischer Abgleich
- Abgleich von IP-Adressen
- „Vorhanden“-Abgleich
- E quals-ignore-case entspricht
- Abgleich mithilfe von Platzhaltern
- Komplexes Beispiel mit mehrfachem Abgleich

- Komplexes Beispiel mit $or-Abgleich

Übereinstimmung mit einem Präfix

Sie können ein Ereignis abhängig vom Präfix eines Werts in der Ereignisquelle abgleichen. Sie können den Präfix-Abgleich für Zeichenfolgenwerte verwenden.

Beispielsweise würde das folgende Ereignismuster mit jedem Ereignis übereinstimmen, bei dem das Feld `"time"` mit `"2017-10-02"` wie `"time": "2017-10-02T18:43:48Z"` beginnt.

```
{
  "time": [ { "prefix": "2017-10-02" } ]
}
```

Präfixabgleich ohne Berücksichtigung der Groß- und Kleinschreibung

Sie können einen Präfixwert auch unabhängig von der Groß- und Kleinschreibung der Zeichen, mit denen ein Wert beginnt, abgleichen, indem Sie ihn `equals-ignore-case` in Verbindung mit verwenden `prefix`.

Das folgende Ereignismuster würde beispielsweise auf jedes Ereignis zutreffen, bei dem das `service` Feld mit der Zeichenfolge begann`EventB`, aber auch `EVENTB` auf jede andere Groß-/Kleinschreibung dieser Zeichen. `eventb`

```
{
  "detail": {"service" : [{ "prefix": { "equals-ignore-case": "EventB" }}]}
}
```

Suffix-Abgleich

Sie können ein Ereignis abhängig vom Suffix eines Werts in der Ereignisquelle abgleichen. Sie können den Suffix-Abgleich für Zeichenfolgenwerte verwenden.

Beispielsweise würde das folgende Ereignismuster mit jedem Ereignis übereinstimmen, bei dem das Feld `"FileName"` mit der Dateierweiterung `.png` endet.

```
{
  "FileName": [ { "suffix": ".png" } ]
}
```

Suffixabgleich ohne Berücksichtigung der Groß- und Kleinschreibung

Sie können einen Suffixwert auch unabhängig von der Groß- und Kleinschreibung der Zeichen, mit denen ein Wert endet, abgleichen, und zwar in Verbindung mit `equals-ignore-case` `suffix.`

Das folgende Ereignismuster würde beispielsweise jedem Ereignis entsprechen, bei dem das `FileName` Feld mit der Zeichenfolge endet`.png`, aber auch `.PNG` mit jeder anderen Groß-/Kleinschreibung dieser Zeichen.

```
{
  "detail": {"FileName" : [{ "suffix": { "equals-ignore-case": ".png" }}]}
}
```

„Alles außer“-Abgleich

Alles außer der Übereinstimmung entspricht allem, was nicht in der Regel angegeben ist.

Sie können den „Alles andere als“-Abgleich mit Zeichenfolgen und numerischen Werten verwenden, einschließlich Listen, die nur Zeichenfolgen oder nur Zahlen enthalten.

Im Folgenden Ereignismuster ist ein „Alles andere als“-Abgleich mit Zeichenfolgen und Zahlen dargestellt.

```
{
  "detail": {
    "state": [ { "anything-but": "initializing" } ]
  }
}

{
  "detail": {
    "x-limit": [ { "anything-but": 123 } ]
  }
}
```

Im Folgenden Ereignismuster ist ein „Alles andere als“-Abgleich mit einer Liste von Zeichenfolgen dargestellt.

```
{
  "detail": {
    "state": [ { "anything-but": [ "stopped", "overloaded" ] } ]
```

```
  }
}
```

Im Folgenden Ereignismuster ist ein „Alles andere als“-Abgleich mit einer Liste von Zahlen dargestellt.

```
{
  "detail": {
    "x-limit": [ { "anything-but": [ 100, 200, 300 ] } ]
  }
}
```

Alles andere als übereinstimmend, wobei die Groß- und Kleinschreibung ignoriert wird

Sie können es auch zusammen mit verwenden `equals-ignore-case`anything-but, um Zeichenkettenwerte unabhängig von der Groß- und Kleinschreibung abzugleichen.

Das folgende Ereignismuster entspricht `state` Feldern, die nicht die Zeichenfolge „initializing“, „INITIALIZING“, „Initializing“ oder eine andere Groß- und Kleinschreibung dieser Zeichen enthalten.

```
{
  "detail": {"state" : [{ "anything-but": { "equals-ignore-case": "initializing" }}]}
}
```

Sie können es auch `equals-ignore-case` in Verbindung mit verwendenanything-but, um einen Vergleich mit einer Werteliste vorzunehmen:

```
{
  "detail": {"state" : [{ "anything-but": { "equals-ignore-case": ["initializing",
 "stopped"] }}]}
}
```

Alles andere als übereinstimmende Präfixe

Sie können es `prefix` in Verbindung mit verwendenanything-but, um Zeichenkettenwerte abzugleichen, die nicht mit dem angegebenen Wert beginnen. Dazu gehören Einzelwerte oder eine Liste von Werten.

Das folgende Ereignismuster zeigt alles andere als eine Übereinstimmung, die mit jedem Ereignis übereinstimmt, für das das Präfix nicht `"init"` im Feld steht. `"state"`

```
{
  "detail": {
    "state": [ { "anything-but": { "prefix": "init" } } ]
  }
}
```

Das folgende Ereignismuster zeigt, dass alles andere als eine Übereinstimmung mit einer Liste von Präfixwerten verwendet wird. Dieses Ereignismuster entspricht jedem Ereignis, das weder das Präfix noch das Feld `"init"` enthält. `"stop"` `"state"`

```
{
"detail": {
  "state" : [{ "anything-but": { "prefix": ["init", "stop"] } } ] }
  }
}
```

Alles andere als übereinstimmende Suffixe

Sie können es `suffix` in Verbindung mit verwenden`anything-but`, um Zeichenkettenwerte abzugleichen, die nicht mit dem angegebenen Wert enden. Dazu gehören Einzelwerte oder eine Liste von Werten.

Das folgende Ereignismuster entspricht allen Werten für das `FileName` Feld, die nicht mit enden`.txt`.

```
{
  "detail": {
    "FileName": [ { "anything-but": { "suffix": ".txt" } } ]
  }
}
```

Das folgende Ereignismuster zeigt, dass alles andere als eine Übereinstimmung mit einer Liste von Suffixwerten verwendet wird. Dieses Ereignismuster entspricht allen Werten für das `FileName` Feld, die nicht mit einem oder enden. `.txt` `.rtf`

```
{
  "detail": {
    "FileName": [ { "anything-but": { "suffix": [".txt", ".rtf"] } } ]
  }
}
```

Alles andere als ein Abgleich mithilfe von Platzhaltern

Sie können das Platzhalterzeichen (*) in den von Ihnen angegebenen Werten für alles andere als für einen Abgleich verwenden. Dazu gehören Einzelwerte oder eine Liste von Werten.

Das folgende Ereignismuster entspricht allen Werten für das `FileName` Feld, die nicht enthalten`/lib/`.

```
{
"detail": {
  "FilePath" : [{ "anything-but": { "wildcard": "*/lib/*" }}]
  }
}
```

Das folgende Ereignismuster zeigt alles andere als übereinstimmende Werte mit einer Liste von Werten, einschließlich Platzhaltern. Dieses Ereignismuster entspricht allen Werten für das `FileName` Feld, die weder oder enthalten. `/lib/` `/bin/`

```
{
"detail": {
  "FilePath" : [{ "anything-but": { "wildcard": ["*/lib/*", "*/bin/*"] }}]
  }
}
```

Weitere Informationen finden Sie unter ???.

Numerischer Abgleich

Der numerische Abgleich funktioniert mit Werten, bei denen es sich um JSON-Zahlen handelt. Sie ist beschränkt auf Werte von -5.0e9 bis einschließlich +5.0e9, auf 15 Stellen genau oder sechs Stellen rechts vom Dezimalzeichen.

Im Folgenden wird der numerische Abgleich für ein Ereignismuster gezeigt, das nur Ereignissen entspricht, die für alle Felder zutreffend sind.

```
{
  "detail": {
    "c-count": [ { "numeric": [ ">", 0, "<=", 5 ] } ],
    "d-count": [ { "numeric": [ "<", 10 ] } ],
    "x-limit": [ { "numeric": [ "=", 3.018e2 ] } ]
  }
```

```
}
```

Abgleich von IP-Adressen

Sie können den Abgleich der IP-Adressen für IPv4- und IPv6-Adressen verwenden. Das folgende Ereignismuster zeigt den Abgleich der IP-Adressen, die mit 10.0.0 beginnen und mit einer Zahl zwischen 0 und 255 enden.

```
{
  "detail": {
    "sourceIPAddress": [ { "cidr": "10.0.0.0/24" } ]
  }
}
```

„Vorhanden“-Abgleich

Bei dem „Vorhanden“-Abgleich kommt es auf das Vorhandensein oder Fehlen eines Feldes in den JSON-Daten des Ereignisses an.

Der „Vorhanden“-Abgleich funktioniert nur auf Blattknoten. Auf Zwischenknoten funktioniert sie nicht.

Das folgende Ereignismuster entspricht jedem Ereignis, das über ein `detail.state`-Feld verfügt.

```
{
  "detail": {
    "state": [ { "exists": true  } ]
  }
}
```

Das vorhergehende Ereignismuster stimmt mit dem folgenden Ereignis überein.

```
{
  "version": "0",
  "id": "7bf73129-1428-4cd3-a780-95db273d1602",
  "detail-type": "EC2 Instance State-change Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2015-11-11T21:29:54Z",
  "region": "us-east-1",
  "resources": ["arn:aws:ec2:us-east-1:123456789012:instance/i-abcd1111"],
  "detail": {
```

```
    "instance-id": "i-abcd1111",
    "state": "pending"
  }
}
```

Das vorhergehende Ereignismuster stimmt NICHT mit dem folgenden Ereignis überein, da es kein `detail.state`-Feld hat.

```
{
  "detail-type": [ "EC2 Instance State-change Notification" ],
  "resources": [ "arn:aws:ec2:us-east-1:123456789012:instance/i-02ebd4584a2ebd341" ],
  "detail": {
    "c-count" : {
       "c1" : 100
    }
  }
}
```

E quals-ignore-case entspricht

Der quals-ignore-caseE-Abgleich funktioniert bei Zeichenkettenwerten unabhängig von der Groß- und Kleinschreibung.

Das folgende Ereignismuster entspricht jedem Ereignis, dessen `detail-type`-Feld der angegebenen Zeichenfolge entspricht, unabhängig von der Groß- und Kleinschreibung.

```
{
  "detail-type": [ { "equals-ignore-case": "ec2 instance state-change notification" } ]
}
```

Das vorhergehende Ereignismuster stimmt mit dem folgenden Ereignis überein.

```
{
  "detail-type": [ "EC2 Instance State-change Notification" ],
  "resources": [ "arn:aws:ec2:us-east-1:123456789012:instance/i-02ebd4584a2ebd341" ],
  "detail": {
    "c-count" : {
       "c1" : 100
    }
  }
}
```

Abgleich mithilfe von Platzhaltern

Sie können das Platzhalterzeichen (*) verwenden, um Zeichenfolgenwerte in Ereignismustern abzugleichen.

> **Note**
>
> Derzeit wird das Platzhalterzeichen nur in Event-Bus-Regeln unterstützt.

Überlegungen zur Verwendung von Platzhaltern in Ihren Ereignismustern:

- Sie können eine beliebige Anzahl von Platzhalterzeichen in einem bestimmten Zeichenfolgenwert angeben. Aufeinanderfolgende Platzhalterzeichen werden jedoch nicht unterstützt.
- EventBridge unterstützt die Verwendung des Backslash-Zeichens (\) zur Angabe der Literalzeichen * und\ in Platzhalterfiltern:
  - Die Zeichenfolge * steht für das Literalzeichen *
  - Die Zeichenfolge \\ steht für das Literalzeichen \

  Die Verwendung des umgekehrten Schrägstrichs, um andere Zeichen durch ein Escape-Zeichen zu schützen, wird nicht unterstützt.

Platzhalter und Komplexität von Ereignismustern

Es gibt eine Grenze, wie komplex eine Regel sein kann, die Platzhalter verwendet. Wenn eine Regel zu komplex ist, wird `InvalidEventPatternException` beim Versuch, die Regel zu erstellen, ein EventBridge zurückgegeben. Wenn Ihre Regel einen solchen Fehler generiert, sollten Sie die folgenden Anleitungen verwenden, um die Komplexität des Ereignismusters zu reduzieren:

- Reduzieren der Anzahl der verwendeten Platzhalterzeichen

  Verwenden Sie Platzhalterzeichen nur dann, wenn Sie tatsächlich einen Abgleich mit mehreren möglichen Werten durchführen müssen. Stellen Sie sich zum Beispiel das folgende Ereignismuster vor, bei dem Sie einen Abgleich mit Event Buses in derselben Region durchführen möchten:

```
{
"EventBusArn": [ { "wildcard": "*:*:*:*:*:event-bus/*" } ]
}
```

Im obigen Fall hängen viele Abschnitte des ARN direkt von der Region ab, in der sich Ihre Event Buses befinden. Wenn Sie also die Region `us-east-1` verwenden, könnte das folgende Beispiel ein weniger komplexes Muster sein, das immer noch den gewünschten Werten entspricht:

```
{
"EventBusArn": [ { "wildcard": "arn:aws:events:us-east-1:*:event-bus/*" } ]
}
```

- Reduzieren sich wiederholender Zeichenfolgen, die nach einem Platzhalterzeichen vorkommen

  Wenn dieselbe Zeichenfolge nach der Verwendung eines Platzhalters mehrfach vorkommt, erhöht sich die Komplexität der Verarbeitung des Ereignismusters. Formulieren Sie Ihr Ereignismuster neu, um wiederholte Sequenzen zu minimieren. Betrachten Sie beispielsweise das folgende Beispiel, das mit der `doc.txt`-Dateinamendatei für jeden Benutzer übereinstimmt:

  ```
  {
  "FileName": [ { "wildcard": "/Users/*/dir/dir/dir/dir/dir/doc.txt" } ]
  }
  ```

  Wenn Sie wüssten, dass die Datei `doc.txt` nur im angegebenen Pfad vorkommt, könnten Sie die wiederholte Zeichenfolge auf diese Weise reduzieren:

  ```
  {
  "FileName": [ { "wildcard": "/Users/*/doc.txt" } ]
  }
  ```

Komplexes Beispiel mit mehrfachem Abgleich

Sie können mehrere Abgleichsregeln zu einem komplexeren Ereignismuster kombinieren. Das folgende Ereignismuster kombiniert beispielsweise `anything-but` und `numeric`.

```
{
  "time": [ { "prefix": "2017-10-02" } ],
  "detail": {
    "state": [ { "anything-but": "initializing" } ],
    "c-count": [ { "numeric": [ ">", 0, "<=", 5 ] } ],
    "d-count": [ { "numeric": [ "<", 10 ] } ],
    "x-limit": [ { "anything-but": [ 100, 200, 300 ] } ]
  }
```

```
}
```

> **Note**
>
> Wenn Sie beim Erstellen von Ereignismustern einen Schlüssel mehrfach angeben, wird die letzte Referenz zur Auswertung von Ereignissen verwendet. Zum Beispiel für das folgende Muster:
>
> ```
> {
>   "detail": {
>     "location": [ { "prefix": "us-" } ],
>     "location": [ { "anything-but": "us-east" } ]
>   }
> }
> ```
>
> Wird bei der Auswertung des `location` nur `{ "anything-but": "us-east" }` berücksichtigt

Komplexes Beispiel mit **`$or`**-Abgleich

Sie können auch komplexe Ereignismuster erstellen, mit denen überprüft wird, ob beliebige Feldwerte in mehreren Feldern übereinstimmen. Verwenden Sie `$or`, um ein Ereignismuster zu erstellen, das überprüft, ob beliebige Werte für mehrere Felder übereinstimmen.

Beachten Sie, dass Sie andere Filtertypen, wie z. B. den numerischen Abgleich und Arrays, in den Musterabgleich für einzelne Felder in Ihrem `$or`-Konstrukt einbeziehen können.

Das folgende Ereignismuster stimmt überein, wenn eine der folgenden Bedingungen erfüllt ist:

- Das Feld `c-count` ist größer als 0 oder kleiner als oder gleich 5.
- Das Feld `d-count` ist kleiner als 10.
- Das Feld `x-limit` entspricht 3.018e2.

```
{
  "detail": {
    "$or": [
      { "c-count": [ { "numeric": [ ">", 0, "<=", 5 ] } ] },
      { "d-count": [ { "numeric": [ "<", 10 ] } ] },
```

```
      { "x-limit": [ { "numeric": [ "=", 3.018e2 ] } ] }
    ]
  }
}
```

> **Note**
>
> APIs, die ein Ereignismuster akzeptieren (wie `PutRule`, `CreateArchive`, `UpdateArchive` und `TestEventPattern`) lösen eine `InvalidEventPatternException` aus, wenn die Verwendung von `$or` mehr als 1000 Regelkombinationen ergibt.
> Wenn Sie die Anzahl der Regelkombinationen in einem Ereignismuster ermitteln möchten, multiplizieren Sie die Gesamtzahl der Argumente aus jedem `$or`-Array im Ereignismuster. Das obige Muster enthält beispielsweise ein einzelnes `$or`-Array mit drei Argumenten, sodass die Gesamtzahl der Regelkombinationen ebenfalls drei beträgt. Wenn Sie ein weiteres `$or`-Array mit zwei Argumenten hinzufügen würden, wäre die Gesamtzahl der Regelkombinationen dann sechs.

Testen eines Ereignismusters mithilfe der EventBridge Sandbox

Regeln verwenden Ereignismuster, um Ereignisse auszuwählen und sie an Ziele zu senden. Ereignismuster haben dieselbe Struktur wie die Ereignisse, mit denen sie übereinstimmen. Ein Ereignismuster stimmt entweder mit einem Ereignis überein oder nicht.

Das Definieren eines Ereignismusters erfolgt in der Regel im Rahmen der Erstellung einer neuen Regel oder der Bearbeitung einer vorhandenen Regel. Mithilfe der Sandbox in können Sie EventBridge jedoch schnell ein Ereignismuster definieren und anhand eines Beispielereignisses überprüfen, ob das Muster den gewünschten Ereignissen entspricht, ohne eine Regel erstellen oder bearbeiten zu müssen. Sobald Sie Ihr Ereignismuster getestet haben, EventBridge bieten Sie Ihnen die Möglichkeit, eine neue Regel zu erstellen, die dieses Ereignismuster direkt in der Sandbox verwendet.

Weitere Informationen zu Ereignismustern finden Sie unter ???.

> **Important**
>
> In ist es möglich EventBridge, Regeln zu erstellen, die zu higher-than-expected Gebühren und Drosselungen führen können. Sie können beispielsweise versehentlich eine Regel erstellen, die zu einer Endlosschleife führt, bei der eine Regel rekursiv ohne Ende ausgelöst

wird. Angenommen, Sie haben eine Regel erstellt, um zu erkennen, dass sich ACLs in einem Amazon-S3-Bucket geändert haben, und Software auszulösen, um sie in den gewünschten Status zu ändern. Ist die Regel nicht sorgfältig geschrieben, löst die anschließende Änderung der ACLs die Regel erneut aus, wodurch eine Endlosschleife entsteht.
Anleitungen zum Schreiben präziser Regeln und Ereignismuster zur Minimierung solcher unerwarteter Ergebnisse finden Sie unter ??? und ???.

Um ein Ereignismuster mithilfe der Sandbox zu testen EventBridge

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Entwicklerressourcen und dann Sandbox aus. Wählen Sie auf der Seite Sandbox die Registerkarte Ereignismuster.
3. Wählen Sie als Eventquelle AWS Events oder EventBridge Partnerevents aus.
4. Wählen Sie im Abschnitt Beispielereignisse einen Beispiel-Ereignistyp aus, anhand dessen Sie das Ereignismuster testen möchten.

   Die folgenden Beispiel-Ereignistypen sind verfügbar:

   - AWS Ereignisse — Wählen Sie unter „Unterstützte Ereignisse“ aus AWS-Services.
   - EventBridge Partnerereignisse — Wählen Sie aus Ereignissen aus EventBridge, die von unterstützenden Drittanbieterdiensten wie Salesforce gesendet werden.
   - Mein eigenes eingeben – Geben Sie Ihr eigenes Ereignis als JSON-Text ein.

     Sie können auch ein Ereignis AWS oder ein Partnerereignis als Ausgangspunkt für die Erstellung Ihres eigenen benutzerdefinierten Ereignisses verwenden.

     1. Wählen Sie AWS Veranstaltungen oder EventBridge Partnerveranstaltungen aus.
     2. Verwenden Sie das Dropdown-Menü Beispielereignisse, um das Ereignis auszuwählen, das Sie als Ausgangspunkt für Ihr benutzerdefiniertes Ereignis verwenden möchten.

        EventBridge zeigt das Beispielereignis an.
     3. Wählen Sie Kopieren aus.
     4. Wählen Sie Mein eigenes eingeben für Ereignistyp aus.
     5. Löschen Sie die Beispiel-Eventstruktur im JSON-Bearbeitungsbereich und fügen Sie das Ereignis AWS oder das Partnerereignis an seiner Stelle ein.
     6. Bearbeiten Sie das Ereignis-JSON, um Ihr eigenes Beispielereignis zu erstellen.

5. Wählen Sie eine Erstellungsmethode aus. Sie können ein Ereignismuster anhand eines EventBridge Schemas oder einer Vorlage erstellen, oder Sie können ein benutzerdefiniertes Ereignismuster erstellen.

   Existing schema

   Gehen Sie wie folgt vor, um ein vorhandenes EventBridge Schema zum Erstellen des Ereignismusters zu verwenden:

   1. Wählen Sie im Abschnitt Erstellungsmethode für Methode die Option Schema verwenden aus.
   2. Wählen Sie im Abschnitt Ereignismuster für Schematyp die Option Schema aus der Schemaregistrierung auswählen aus.
   3. Wählen Sie für Schemaregistrierung das Dropdown-Feld aus und geben Sie den Namen einer Schemaregistrierung ein, z. B. `aws.events`. Sie können auch eine Option aus der angezeigten Dropdown-Liste auswählen.
   4. Wählen Sie für Schema das Dropdown-Feld aus und geben Sie den Namen des zu verwendenden Schemas ein. z. B. `aws.s3@0bjectDeleted`. Sie können auch eine Option aus der angezeigten Dropdown-Liste auswählen.
   5. Klicken Sie im Bereich Modelle neben einem beliebigen Attribut auf die Schaltfläche Bearbeiten, um dessen Eigenschaften zu öffnen. Stellen Sie die Felder Beziehung und Wert nach Bedarf ein und wählen Sie dann Einrichten aus, um das Attribut zu speichern.

      > **Note**
      >
      > Informationen zur Definition eines Attributs erhalten Sie, wenn Sie das Infosymbol neben dem Namen des Attributs auswählen. Eine Referenz zum Einrichten von Attributeigenschaften in Ihrem Ereignis finden Sie im Abschnitt Notiz des Dialogfelds mit den Attributeigenschaften.
      > Zum Löschen der Eigenschaften eines Attributs klicken Sie auf die Schaltfläche Bearbeiten für dieses Attribut und anschließend auf Löschen.

   6. Wählen Sie Ereignismuster in JSON generieren aus, um Ihr Ereignismuster als JSON-Text zu generieren und zu validieren.
   7. Zum Testen des Beispielereignisses anhand Ihres Testmusters wählen Sie Testmuster aus.

EventBridge zeigt ein Meldungsfeld an, in dem angegeben wird, ob Ihr Beispielereignis mit dem Ereignismuster übereinstimmt.

Sie können auch eine der folgenden Optionen wählen:

- Kopieren – Kopiert das Ereignismuster in die Zwischenablage Ihres Geräts.
- Verschönern – Erleichtert das Lesen des JSON-Texts durch Hinzufügen von Zeilenumbrüchen, Tabulatoren und Leerzeichen.

Custom schema

Gehen Sie wie folgt vor, um ein benutzerdefiniertes Schema zu erstellen und es in ein Ereignismuster zu konvertieren:

1. Wählen Sie im Abschnitt Erstellungsmethode für Methode die Option Schema verwenden aus.
2. Wählen Sie im Abschnitt Ereignismuster für Schematyp die Option Schema eingeben aus.
3. Geben Sie Ihr Schema in das Textfeld ein. Sie müssen das Schema als gültigen JSON-Text formatieren.
4. Klicken Sie im Bereich Modelle neben einem beliebigen Attribut auf die Schaltfläche Bearbeiten, um dessen Eigenschaften zu öffnen. Stellen Sie die Felder Beziehung und Wert nach Bedarf ein und wählen Sie dann Einrichten aus, um das Attribut zu speichern.

   > **Note**
   >
   > Informationen zur Definition eines Attributs erhalten Sie, wenn Sie das Infosymbol neben dem Namen des Attributs auswählen. Eine Referenz zum Einrichten von Attributeigenschaften in Ihrem Ereignis finden Sie im Abschnitt Notiz des Dialogfelds mit den Attributeigenschaften.
   > Zum Löschen der Eigenschaften eines Attributs klicken Sie auf die Schaltfläche Bearbeiten für dieses Attribut und anschließend auf Löschen.

5. Wählen Sie Ereignismuster in JSON generieren aus, um Ihr Ereignismuster als JSON-Text zu generieren und zu validieren.
6. Zum Testen des Beispielereignisses anhand Ihres Testmusters wählen Sie Testmuster aus.

EventBridge zeigt ein Meldungsfeld an, in dem angegeben wird, ob Ihr Beispielereignis dem Ereignismuster entspricht.

Sie können auch eine der folgenden Optionen wählen:

- Kopieren – Kopiert das Ereignismuster in die Zwischenablage Ihres Geräts.
- Verschönern – Erleichtert das Lesen des JSON-Texts durch Hinzufügen von Zeilenumbrüchen, Tabulatoren und Leerzeichen.

Event pattern

Gehen Sie wie folgt vor, um ein benutzerdefiniertes Ereignismuster im JSON-Format zu schreiben:

1. Wählen Sie im Abschnitt Erstellungsmethode für Methode die Option Benutzerdefiniertes Muster (JSON-Editor) aus.
2. Geben Sie für Ereignismuster Ihr benutzerdefiniertes Ereignismuster in JSON-formatiertem Text ein.
3. Zum Testen des Beispielereignisses anhand Ihres Testmusters wählen Sie Testmuster aus.

   EventBridge zeigt ein Meldungsfeld an, in dem angegeben wird, ob Ihr Beispielereignis dem Ereignismuster entspricht.

   Sie können auch eine der folgenden Optionen wählen:

   - Kopieren – Kopiert das Ereignismuster in die Zwischenablage Ihres Geräts.
   - Verschönern – Erleichtert das Lesen des JSON-Texts durch Hinzufügen von Zeilenumbrüchen, Tabulatoren und Leerzeichen.
   - Ereignismusterformular – Öffnet das Ereignismuster in Pattern Builder. Wenn das Muster in Pattern Builder nicht unverändert gerendert werden kann, werden Sie EventBridge gewarnt, bevor Pattern Builder geöffnet wird.

6. (Optional) Wenn Sie eine Regel mit diesem Ereignismuster erstellen und die Regel einem bestimmten Event Bus zuweisen möchten, wählen Sie Regel mit Muster erstellen aus.

   EventBridge führt Sie zu Schritt 1 von Regel erstellen, in dem Sie eine Regel erstellen und sie dem Event-Bus Ihrer Wahl zuweisen können.

Beachten Sie, dass Schritt 2 – Ereignismuster erstellen die Informationen zum Ereignismuster enthält, die Sie bereits angegeben haben und die Sie akzeptieren oder aktualisieren können.

Weitere Informationen zum Erstellen von Regeln finden Sie unter ???.

Bewährte Methoden bei der Definition von EventBridge Amazon-Eventmustern

Im Folgenden finden Sie einige bewährte Methoden, die Sie bei der Definition von Ereignismustern in Ihren Event-Bus-Regeln berücksichtigen sollten.

Vermeiden des Schreibens von Endlosschleifen

In ist es möglich EventBridge, Regeln zu erstellen, die zu Endlosschleifen führen, bei denen eine Regel wiederholt ausgelöst wird. Eine Regel kann beispielsweise erkennen, dass sich ACLs in einem S3-Bucket geändert haben, und Software auslösen, die sie in den gewünschten Zustand ändern. Ist die Regel nicht sorgfältig geschrieben, löst die anschließende Änderung der ACLs die Regel erneut aus, wodurch eine Endlosschleife entsteht.

Zur Vermeidung dieser Probleme sollten Sie die Ereignismuster für Ihre Regeln so präzise wie möglich schreiben, sodass sie nur den Ereignissen entsprechen, die Sie tatsächlich an das Ziel senden möchten. Im obigen Beispiel würden Sie ein Ereignismuster erstellen, das den Ereignissen entspricht, sodass die ausgelösten Aktionen nicht dieselbe Regel erneut auslösen. Erstellen Sie beispielsweise in Ihrer Regel ein Ereignismuster, das Ereignissen nur dann enspricht, wenn festgestellt wird, dass sich ACLs in einem fehlerhaften Zustand befinden, und nicht nach jeder Änderung. Weitere Informationen finden Sie unter ??? und ???.

Eine Endlosschleife kann schnell höhere Gebühren als erwartet verursachen. Dies kann auch zu Drosselung und verzögerter Ereigniszustellung führen. Sie können die Obergrenze Ihrer Aufrufraten überwachen, um vor unerwarteten Volumenspitzen gewarnt zu werden.

Verwenden Sie die Budgetierung, um Sie zu warnen, wenn die Gebühren das von Ihnen angegebene Limit überschreiten. Weitere Informationen finden Sie unter Verwalten der Kosten mit Budgets.

Möglichst präzises Gestalten von Ereignismustern

Je genauer Ihr Ereignismuster ist, desto wahrscheinlicher ist es, dass es nur mit den tatsächlich gewünschten Ereignissen übereinstimmt und unerwartete Übereinstimmungen vermieden werden,

wenn neue Ereignisse zu einer Ereignisquelle hinzugefügt oder bestehende Ereignisse mit neuen Eigenschaften aktualisiert werden.

Ereignismuster können Filter enthalten, die einen Abgleich auf Folgendes durchführen:

- Ereignismetadaten über das Ereignis, wie `source`, `detail-type`, `account` oder `region`.
- Ereignisdaten, d. h. die Felder innerhalb des `detail`-Objekts.
- Der Inhalt des Ereignisses oder die tatsächlichen Werte der Felder innerhalb des `detail`-Objekts.

Die meisten Muster sind einfach, z. B. nur die Angabe von `source`- und `detail-type`-Filtern. Zu den EventBridge Mustern gehört jedoch die Flexibilität, nach jedem Schlüssel oder Wert des Ereignisses zu filtern. Darüber hinaus können Sie Inhaltsfilter wie `prefix`- und `suffix`-Filter anwenden, um die Präzision Ihrer Muster zu verbessern. Weitere Informationen finden Sie unter ???.

Geben Sie die Ereignisquelle und den Detailtyp als Filter an.

Sie können das Generieren von Endlosschleifen und das Abgleichen unerwünschter Ereignisse reduzieren, indem Sie Ihre Ereignismuster mithilfe der `source`- und `detail-type`-Metadatenfelder präzisieren.

Wenn Sie bestimmte Werte in zwei oder mehr Feldern abgleichen müssen, verwenden Sie den `$or`-Vergleichsoperator, anstatt alle möglichen Werte in einem einzigen Werte-Array aufzulisten.

Für Ereignisse, die per übermittelt werden AWS CloudTrail, empfehlen wir, das `eventName` Feld als Filter zu verwenden.

Das folgende Beispiel für ein Ereignismuster entspricht `CreateQueue` oder `SetQueueAttributes` aus dem Amazon Simple Queue Service Service `CreateKey` oder `DisableKeyRotation` Ereignissen aus dem AWS Key Management Service Service.

```
{
  "detail-type": ["AWS API Call via CloudTrail"],
  "$or": [{
      "source": [
        "aws.sqs"
        ],
      "detail": {
        "eventName": [
          "CreateQueue",
          "SetQueueAttributes"
```

```
        ]
      }
    },
    {
      "source": [
        "aws.kms"
        ],
      "detail": {
        "eventName": [
          "CreateKey",
          "DisableKeyRotation"
        ]
      }
    }
  ]
}
```

Angeben von Konto und Region als Filter

Wenn Sie `region`- und `account`-Felder in Ihr Ereignismuster aufnehmen, können Sie den konto- oder regionsübergreifenden Ereignisabgleich einschränken.

Angeben von Inhaltsfiltern

Die inhaltsbasierte Filterung kann dazu beitragen, die Genauigkeit von Ereignismustern zu verbessern und gleichzeitig die Länge des Ereignismusters auf ein Minimum zu beschränken. Beispielsweise kann ein Abgleich auf der Grundlage eines numerischen Bereichs hilfreich sein, anstatt alle möglichen numerischen Werte aufzulisten.

Weitere Informationen finden Sie unter ???.

Definieren Sie Ihre Ereignismuster so, dass sie Aktualisierungen der Ereignisquellen berücksichtigen

Bei der Erstellung von Ereignismustern sollten Sie berücksichtigen, dass sich Ereignisschemata und Ereignis-Domains im Laufe der Zeit weiterentwickeln und erweitern können. Auch hier hilft es Ihnen, Ihre Ereignismuster so präzise wie möglich zu gestalten, um unerwartete Übereinstimmungen zu vermeiden, wenn sich die Ereignisquelle ändert oder erweitert.

Nehmen wir zum Beispiel an, Sie führen einen Abgleich mit Ereignissen aus einem neuen Mikroservice durch, der Ereignisse im Zusammenhang mit Zahlungen veröffentlicht. Anfänglich

verwendet der Service die Domain `acme.payments` und veröffentlicht ein einzelnes Ereignis, `Payment accepted`:

```
{
  "detail-type": "Payment accepted",
  "source": "acme.payments",
  "detail": {
    "type": "credit",
    "amount": "100",
    "date": "2023-06-10",
    "currency": "USD"
    }
  }
}
```

An dieser Stelle könnten Sie ein einfaches Ereignismuster erstellen, das „Zahlung akzeptiert“-Ereignissen entspricht:

```
{ “source” : “acme.payments” }
```

Nehmen wir jedoch an, dass der Service später ein neues Ereignis für abgelehnte Zahlungen einführt:

```
{
  "detail-type": "Payment rejected",
  "source": "acme.payments",
  "detail": {
  }
}
```

In diesem Fall wird das einfache Ereignismuster, das Sie erstellt haben, nun mit beiden `Payment rejected`- und `Payment accepted`-Ereignissen abgeglichen. EventBridge leitet beide Ereignistypen zur Verarbeitung an das angegebene Ziel weiter, was zu Verarbeitungsfehlern und zusätzlichen Verarbeitungskosten führen kann.

Wenn Sie Ihr Ereignismuster nur auf `Payment accepted`-Ereignisse beschränken möchten, sollten Sie mindestens beide `source` und `detail-type` angeben:

```
{
  "detail-type": "Payment accepted",
```

```
  "source": "acme.payments"
  }
}
```

Sie können in Ihrem Ereignismuster auch Konto und Region angeben, um weiter einzuschränken, wann konto- oder regionsübergreifende Ereignisse dieser Regel entsprechen.

```
{
  "account": "012345678910",
  "source": "acme.payments",
  "region": "AWS-Region",
  "detail-type": "Payment accepted"
}
```

Überprüfen von Ereignismustern

Wenn Sie sicherstellen möchten, dass die Regeln mit den gewünschten Ereignissen übereinstimmen, empfehlen wir Ihnen dringend, Ihre Ereignismuster zu überprüfen. Sie können Ihre Ereignismuster mithilfe der EventBridge Konsole oder der API überprüfen:

- In der EventBridge Konsole können Sie Ereignismuster als Teil der Regelerstellung oder separat mithilfe der Sandbox erstellen und testen.
- Mithilfe der Aktion können Sie Ihre Ereignismuster programmgesteuert testen. TestEventPattern

Amazon- EventBridge Regeln

Sie geben an, was mit den Ereignissen EventBridge macht, die an jeden Event Bus übermittelt werden. Dazu erstellen Sie Regeln . Eine Regel gibt an, welche Ereignisse an welche Ziele für die Verarbeitung gesendet werden sollen. Eine einzelne Regel kann ein Ereignis an mehrere Ziele senden, die dann parallel ausgeführt werden.

Sie können zwei Arten von Regeln erstellen:

- Regeln, die mit Ereignisdaten übereinstimmen

  Sie können Regeln erstellen, die anhand von Ereignisdatenkriterien mit eingehenden Ereignissen übereinstimmen (ein Ereignismuster genannt). Ein Ereignismuster definiert die Ereignisstruktur und die Felder, denen eine Regel entspricht. Wenn ein Ereignis den im Ereignismuster definierten Kriterien entspricht, EventBridge sendet es an das/die von Ihnen angegebenen Ziel(e).

  Weitere Informationen finden Sie unter ???.
- Regeln, die nach einem Zeitplan ausgeführt werden

  Sie können auch Regeln erstellen, die Ereignisse in bestimmten Intervallen an die angegebenen Ziele senden. Um beispielsweise regelmäßig eine - Lambda Funktion auszuführen, können Sie eine Regel erstellen, die nach einem Zeitplan ausgeführt wird.

> **Note**
>
> EventBridge bietet Amazon EventBridge Scheduler, einen Serverless-Scheduler, mit dem Sie Aufgaben von einem zentralen, verwalteten Service aus erstellen, ausführen und verwalten können. EventBridge Scheduler ist hochgradig anpassbar und bietet eine verbesserte Skalierbarkeit gegenüber EventBridge geplanten Regeln mit einer breiteren Palette von Ziel-API-Operationen und - AWS Services.
> Wir empfehlen Ihnen, EventBridge Scheduler zu verwenden, um Ziele nach einem Zeitplan aufzurufen. Weitere Informationen finden Sie unter ???.

Das folgende Video befasst sich mit den Grundlagen von Regeln: Was sind Regeln?

Von Amazon EventBridge verwaltete Regeln

Zusätzlich zu den von Ihnen erstellten Regeln können AWS Services EventBridge Regeln in Ihrem AWS Konto erstellen und verwalten, die für bestimmte Funktionen in diesen Services benötigt werden. Diese werden als verwaltete Regeln bezeichnet.

Wenn ein Service eine verwaltete Regel erstellt, kann er auch eine -IAM Richtlinie erstellen, die diesem Service die Berechtigung zum Erstellen der Regel erteilt. Auf diese Weise erstellte IAM-Richtlinien sind eng mit Berechtigungen auf Ressourcenebene verbunden, um die Erstellung nur der notwendigen Regeln zu ermöglichen.

Sie können verwaltete Regeln mit der Option Löschen erzwingen löschen. Sie sollten sie jedoch nur löschen, wenn Sie sicher sind, dass der andere Service die Regel nicht mehr benötigt. Andernfalls führt das Löschen einer verwalteten Regel dazu, dass die Funktionen, die von ihr abhängen, nicht mehr funktionieren.

Erstellen von Amazon-EventBridge-Regeln, die auf Ereignisse reagieren

Wenn Sie Maßnahmen für Ereignisse ergreifen möchten, die von Amazon EventBridge empfangen wurden, können Sie Regeln erstellen. Wenn ein Ereignis mit dem Ereignismuster übereinstimmt, das in der Regel definiert wurde, sendet EventBridge das Ereignis an das angegebene Ziel und löst die in der Regel definierte Aktion aus.

Das folgende Video zeigt, wie verschiedene Arten von Regeln erstellt werden und wie man sie testet: Weitere Informationen zu Regeln

Führen Sie das folgende Verfahren aus, um eine Amazon-EventBridge-Regel zu erstellen, die auf Ereignisse antwortet.

Erstellen einer Regel, die auf Ereignisse reagiert

In den folgenden Schritten erfahren Sie, wie Sie eine Regel erstellen, die EventBridge verwendet, um Ereignisse abzugleichen, wenn sie an den angegebenen Event Bus gesendet werden.

Schritte

- Definieren der Regel
- Erstellen des Ereignismusters
- Auswählen von Zielen
- Konfigurieren von Tags und Überprüfen von Regeln

Definieren der Regel

Geben Sie zunächst einen Namen und eine Beschreibung für die Regel ein, um sie zu identifizieren. Sie müssen auch den Event Bus definieren, in dem die Regel nach Ereignissen sucht, die einem Ereignismuster entsprechen.

So definieren Sie die Regeldetails

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen.

4. Geben Sie für die Regel einen Namen und optional eine Beschreibung ein.

   Eine Regel darf nicht denselben Namen wie eine andere Regel in derselben AWS-Region und auf demselben Event Bus haben.

5. Wählen Sie für Event Bus den Event Bus aus, der dieser Regel zugeordnet werden soll. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie AWS-Standard-Event-Bus aus. Wenn ein AWS-Service in Ihrem Konto ein Ereignis ausgibt, wird es stets an den Standard-Event-Bus Ihres Kontos weitergeleitet.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Next (Weiter).

Erstellen des Ereignismusters

Als Nächstes erstellen Sie das Ereignismuster. Geben Sie dazu die Ereignisquelle an, wählen Sie die Grundlage für das Ereignismuster aus und definieren Sie die Attribute und Werte, anhand derer abgeglichen werden soll. Sie können das Ereignismuster auch in JSON generieren und es anhand eines Beispielereignisses testen.

So erstellen Sie das Ereignismuster

1. Wählen Sie für Event source (Ereignisquelle) AWS events or EventBridge partner events (-Ereignisse oder EventBridge-Partnerereignisse).
2. (Optional) Wählen Sie im Abschnitt Beispielereignisse einen Beispiel-Ereignistyp aus, anhand dessen Sie das Ereignismuster testen möchten.

   Die folgenden Beispiel-Ereignistypen sind verfügbar:

   - AWS-Ereignisse – Wählen Sie aus Ereignissen aus, die von unterstützten AWS-Services ausgegeben wurden.
   - EventBridge-Partnerereignisse – Wählen Sie aus Ereignissen aus, die von Drittanbieterservices ausgegeben wurden, die EventBridge unterstützen, wie Salesforce.
   - Mein eigenes eingeben – Geben Sie Ihr eigenes Ereignis als JSON-Text ein.

     Sie können auch ein AWS- oder Partnerereignis als Ausgangspunkt für die Erstellung Ihres eigenen benutzerdefinierten Ereignisses verwenden.

     1. Wählen Sie AWS-Ereignisse oder EventBridge-Partnerereignisse aus.

2. Verwenden Sie das Dropdown-Menü Beispielereignisse, um das Ereignis auszuwählen, das Sie als Ausgangspunkt für Ihr benutzerdefiniertes Ereignis verwenden möchten.

   EventBridge zeigt das Beispielereignis an.
3. Wählen Sie Kopieren aus.
4. Wählen Sie Mein eigenes eingeben für Ereignistyp aus.
5. Löschen Sie die Beispiel-Ereignisstruktur im JSON-Bearbeitungsbereich und fügen Sie das AWS- oder Partnerereignis an deren Stelle ein.
6. Bearbeiten Sie das Ereignis-JSON, um Ihr eigenes Beispielereignis zu erstellen.

3. Wählen Sie eine Erstellungsmethode aus. Sie können ein Ereignismuster aus einem EventBridge-Schema oder einer EventBridge-Vorlage erstellen, oder Sie können ein benutzerdefiniertes Ereignismuster erstellen.

   Existing schema

   Gehen Sie wie folgt vor, um ein vorhandenes EventBridge-Schema zum Erstellen des Ereignismusters zu verwenden:

   1. Wählen Sie im Abschnitt Erstellungsmethode für Methode die Option Schema verwenden aus.
   2. Wählen Sie im Abschnitt Ereignismuster für Schematyp die Option Schema aus der Schemaregistrierung auswählen aus.
   3. Wählen Sie für Schemaregistrierung das Dropdown-Feld aus und geben Sie den Namen einer Schemaregistrierung ein, z. B. `aws.events`. Sie können auch eine Option aus der angezeigten Dropdown-Liste auswählen.
   4. Wählen Sie für Schema das Dropdown-Feld aus und geben Sie den Namen des zu verwendenden Schemas ein. Zum Beispiel `aws.s3@ObjectDeleted`. Sie können auch eine Option aus der angezeigten Dropdown-Liste auswählen.
   5. Klicken Sie im Bereich Modelle neben einem beliebigen Attribut auf die Schaltfläche Bearbeiten, um dessen Eigenschaften zu öffnen. Stellen Sie die Felder Beziehung und Wert nach Bedarf ein und wählen Sie dann Einrichten aus, um das Attribut zu speichern.

      > **Note**
      >
      > Informationen zur Definition eines Attributs erhalten Sie, wenn Sie das Infosymbol neben dem Namen des Attributs auswählen. Eine Referenz zum Einrichten

von Attributeigenschaften in Ihrem Ereignis finden Sie im Abschnitt Notiz des Dialogfelds mit den Attributeigenschaften.
Zum Löschen der Eigenschaften eines Attributs klicken Sie auf die Schaltfläche Bearbeiten für dieses Attribut und anschließend auf Löschen.

6. Wählen Sie Ereignismuster in JSON generieren aus, um Ihr Ereignismuster als JSON-Text zu generieren und zu validieren.
7. (Optional) Zum Testen des Beispielereignisses anhand Ihres Testmusters wählen Sie Testmuster aus.

   EventBridge zeigt ein Meldungsfeld mit dem Hinweis an, ob Ihr Beispielereignis dem Ereignismuster entspricht.

   Sie können auch eine der folgenden Optionen wählen:

   - Kopieren – Kopiert das Ereignismuster in die Zwischenablage Ihres Geräts.
   - Verschönern – Erleichtert das Lesen des JSON-Texts durch Hinzufügen von Zeilenumbrüchen, Tabulatoren und Leerzeichen.

Custom schema

Gehen Sie wie folgt vor, um ein benutzerdefiniertes Schema zu erstellen und es in ein Ereignismuster zu konvertieren:

1. Wählen Sie im Abschnitt Erstellungsmethode für Methode die Option Schema verwenden aus.
2. Wählen Sie im Abschnitt Ereignismuster für Schematyp die Option Schema eingeben aus.
3. Geben Sie Ihr Schema in das Textfeld ein. Sie müssen das Schema als gültigen JSON-Text formatieren.
4. Klicken Sie im Bereich Modelle neben einem beliebigen Attribut auf die Schaltfläche Bearbeiten, um dessen Eigenschaften zu öffnen. Stellen Sie die Felder Beziehung und Wert nach Bedarf ein und wählen Sie dann Einrichten aus, um das Attribut zu speichern.

> **Note**
>
> Informationen zur Definition eines Attributs erhalten Sie, wenn Sie das Infosymbol neben dem Namen des Attributs auswählen. Eine Referenz zum Einrichten

von Attributeigenschaften in Ihrem Ereignis finden Sie im Abschnitt Notiz des Dialogfelds mit den Attributeigenschaften.
Zum Löschen der Eigenschaften eines Attributs klicken Sie auf die Schaltfläche Bearbeiten für dieses Attribut und anschließend auf Löschen.

5. Wählen Sie Ereignismuster in JSON generieren aus, um Ihr Ereignismuster als JSON-Text zu generieren und zu validieren.
6. (Optional) Zum Testen des Beispielereignisses anhand Ihres Testmusters wählen Sie Testmuster aus.

   EventBridge zeigt ein Meldungsfeld mit dem Hinweis an, ob Ihr Beispielereignis dem Ereignismuster entspricht.

   Sie können auch eine der folgenden Optionen wählen:

   - Kopieren – Kopiert das Ereignismuster in die Zwischenablage Ihres Geräts.
   - Verschönern – Erleichtert das Lesen des JSON-Texts durch Hinzufügen von Zeilenumbrüchen, Tabulatoren und Leerzeichen.

Event pattern

Gehen Sie wie folgt vor, um ein benutzerdefiniertes Ereignismuster im JSON-Format zu schreiben:

1. Wählen Sie im Abschnitt Erstellungsmethode für Methode die Option Benutzerdefiniertes Muster (JSON-Editor) aus.
2. Geben Sie für Ereignismuster Ihr benutzerdefiniertes Ereignismuster in JSON-formatiertem Text ein.
3. (Optional) Zum Testen des Beispielereignisses anhand Ihres Testmusters wählen Sie Testmuster aus.

   EventBridge zeigt ein Meldungsfeld mit dem Hinweis an, ob Ihr Beispielereignis dem Ereignismuster entspricht.

   Sie können auch eine der folgenden Optionen wählen:

   - Kopieren – Kopiert das Ereignismuster in die Zwischenablage Ihres Geräts.
   - Verschönern – Erleichtert das Lesen des JSON-Texts durch Hinzufügen von Zeilenumbrüchen, Tabulatoren und Leerzeichen.

- Ereignismusterformular – Öffnet das Ereignismuster in Pattern Builder. Wenn das Muster in Pattern Builder nicht unverändert gerendert werden kann, warnt EventBridge Sie, bevor es Pattern Builder öffnet.

4. Wählen Sie Next (Weiter).

Auswählen von Zielen

Wählen Sie ein oder mehrere Ziele aus, um Ereignisse zu empfangen, die dem angegebenen Muster entsprechen. Zu den Zielen können ein EventBridge-Event-Bus, EventBridge-API-Ziele, einschließlich SaaS-Partner wie Salesforce, oder andere AWS-Service gehören.

So wählen Sie Ziele aus

1. Wählen Sie für Zieltyp einen der folgenden Zieltypen aus:

   Event bus

   Zum Auswählen eines EventBridge-Event-Bus klicken Sie auf EventBridge-Event-Bus und gehen dann wie folgt vor:

   - So verwenden Sie einen Event Bus in derselben AWS-Region wie diese Regel:
     1. Wählen Sie Event Bus in demselben Konto und derselben Region aus.
     2. Wählen Sie für Event Bus für das Ziel das Dropdown-Feld aus und geben Sie den Namen des Event Bus ein. Sie können den Event Bus auch aus der Dropdown-Liste auswählen.

        Weitere Informationen finden Sie unter ???.
   - So verwenden Sie einen Event Bus in einer anderen AWS-Region oder einem anderen Konto als diese Regel:
     1. Wählen Sie Event Bus in einem anderen Konto oder einer anderen Region aus.
     2. Geben Sie für Event Bus als Ziel den ARN des Event Bus ein, den Sie verwenden möchten.

        Weitere Informationen finden Sie unter:

        - ???
        - ???

API destination

Zum Verwenden eines EventBridge-API-Ziels wählen Sie EventBridge-API-Ziel aus und führen Sie dann einen der folgenden Schritte aus:

- Zum Verwenden eines vorhandenen API-Ziels wählen Sie Vorhandenes API-Ziel verwenden aus. Wählen Sie dann ein API-Ziel aus der Dropdown-Liste aus.
- Zum Erstellen eines neuen API-Ziels wählen Sie Neues API-Ziel erstellen aus. Geben Sie anschließend die folgenden Details für das Ziel an:
  - Name – Geben Sie einen Namen für das Ziel ein.

    Namen müssen innerhalb Ihres AWS-Konto-Kontos eindeutig sein. Namen können bis zu 64 Zeichen lang sein. Gültige Zeichen sind A-Z, a-z, 0-9 und . _ - (Bindestrich).
  - (Optional) Beschreibung – Geben Sie eine Beschreibung für das Ziel ein.

    Beschreibungen können bis zu 512 Zeichen lang sein.
  - API-Zielendpunkt – Der URL-Endpunkt für das Ziel.

    Die Endpunkt-URL muss mit **`https`** beginnen. Sie können das **`*`** als Pfadparameterplatzhalter angeben. Sie können Pfadparameter über das `HttpParameters`-Attribut des Ziels festlegen.
  - HTTP-Methode – Wählen Sie die HTTP-Methode aus, die beim Aufrufen des Endpunkts verwendet wird.
  - (Optional) Aufrufratenlimit pro Sekunde – Geben Sie die maximale Anzahl von Aufrufen ein, die pro Sekunde für dieses Ziel akzeptiert werden.

    Dieser Wert muss größer als null sein. Standardmäßig ist dieser Wert auf 300 festgelegt.
  - Verbindung – Wählen Sie, ob Sie eine neue oder eine vorhandene Verbindung verwenden möchten:
    - Zum Verwenden einer vorhandenen Verbindung wählen Sie Vorhandene Verbindung verwenden und die Verbindung aus der Dropdown-Liste aus.
    - Zum Erstellen einer neuen Verbindung für dieses Ziel wählen Sie Neue Verbindung erstellen aus und definieren dann den Namen, den Zieltyp und den Autorisierungstyp der Verbindung. Sie können auch eine optionale Beschreibung für diese Verbindung hinzufügen.

Weitere Informationen finden Sie unter ???.

AWS-Service

Zum Verwenden eines AWS-Service wählen Sie AWS-Service aus und gehen dann wie folgt vor:

1. Wählen Sie für Ziel auswählen einen AWS-Service aus, der als Ziel verwendet werden soll. Geben Sie die angeforderten Informationen für den ausgewählten Service ein.

> **Note**
>
> Die angezeigten Felder variieren je nach ausgewähltem Service. Weitere Informationen zu verfügbaren Zielen finden Sie unter In der EventBridge Konsole verfügbare Ziele.

2. Für viele Zieltypen benötigt EventBridge Berechtigungen zum Senden von Ereignissen an das Ziel. In diesen Fällen kann EventBridge die IAM-Rolle erstellen, die zum Ausführen Ihrer Regel erforderlich ist.

   Führen Sie für Ausführungsrolle einen der folgenden Schritte aus:

   - Gehen Sie wie folgt vor, um eine neue Ausführungsrolle für diese Regel zu erstellen:
     a. Wählen Sie Eine neue Rolle für diese spezifische Ressource erstellen aus.
     b. Geben Sie entweder einen Namen für diese Ausführungsrolle ein, oder verwenden Sie den von EventBridge generierten Namen.
   - So verwenden Sie eine vorhandene Ausführungsrolle für diese Regel:
     a. Wählen Sie Vorhandene Rolle verwenden aus.
     b. Geben Sie den Namen der zu verwendenden Ausführungsrolle ein oder wählen Sie ihn aus der Dropdown-Liste aus.

3. (Optional) Geben Sie für Zusätzliche Einstellungen eine der optionalen Einstellungen an, die für Ihren Zieltyp verfügbar sind:

   Event bus

   (Optional) Wählen Sie für Warteschlange für unzustellbare Nachrichten aus, ob Sie eine standardmäßige Amazon-SQS-Warteschlange als Warteschlange für unzustellbare

Nachrichten verwenden möchten. EventBridge sendet Ereignisse, die dieser Regel entsprechen, an die Warteschlange für Dead-Letter, wenn sie nicht erfolgreich an das Ziel übermittelt wurden. Führen Sie eine der folgenden Aktionen aus:

- Klicken Sie auf Keine, um keine Warteschlange für unzustellbare Nachrichten zu verwenden.
- Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange im aktuellen AWS-Konto als Warteschlange für unzustellbare Nachrichten und wählen Sie dann die Warteschlange aus der Dropdown-Liste aus.
- Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange in einem anderen AWS-Konto als Warteschlange für unzustellbare Nachrichten und geben Sie dann den ARN der Warteschlange ein, die verwendet werden soll. Sie müssen eine ressourcenbasierte Richtlinie an die Warteschlange anhängen, die EventBridge Berechtigung zum Senden von Nachrichten an die Warteschlange erteilt.

  Weitere Informationen finden Sie unter Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten.

API destination

1. (Optional) Wählen Sie für „Zieleingabe konfigurieren“ aus, wie Sie den an das Ziel gesendeten Text für passende Ereignisse anpassen möchten. Wählen Sie eine der folgenden Optionen aus:
   - Übereinstimmende Ereignisse – EventBridge sendet das gesamte ursprüngliche Quellereignis an das Ziel. Dies ist die Standardeinstellung.
   - Teil der übereinstimmenden Ereignisse – EventBridge sendet nur den angegebenen Teil des ursprünglichen Quellereignisses an das Ziel.

     Geben Sie unter Den Teil des übereinstimmenden Ereignisses angeben einen JSON-Pfad an, der den Teil des Ereignisses definiert, den EventBridge an das Ziel senden soll.
   - Konstante (JSON-Text) – EventBridge sendet nur den angegebenen JSON-Text an das Ziel. Es wird kein Teil des ursprünglichen Quellereignisses gesendet.

     Geben Sie unter Die Konstante in JSON angeben den JSON-Text an, den EventBridge anstelle des Ereignisses an das Ziel senden soll.

- Eingabe-Transformator – Konfigurieren Sie einen Eingabe-Transformator, um den Text anzupassen, den EventBridge an das Ziel senden soll. Weitere Informationen finden Sie unter ???.

  a. Wählen Sie Eingabe-Transformator konfigurieren aus.

  b. Konfigurieren Sie den Eingabe-Transformator wie unter ??? beschrieben.

2. (Optional) Geben Sie unter Wiederholungsrichtlinie an, wie EventBridge das Senden eines Ereignisses an ein Ziel wiederholen soll, nachdem ein Fehler aufgetreten ist.

   - Maximales Alter des Ereignisses – Geben Sie die maximale Zeitspanne (in Stunden, Minuten und Sekunden) ein, für die EventBridge unverarbeitete Ereignisse beibehält. Die Standardeinstellung ist 24 Stunden.
   - Wiederholungsversuche – Geben Sie ein, wie oft EventBridge das Senden an ein Ziel maximal wiederholen soll, nachdem ein Fehler aufgetreten ist. Die Standardeinstellung ist 185 Mal.

3. (Optional) Wählen Sie für Warteschlange für unzustellbare Nachrichten aus, ob Sie eine standardmäßige Amazon-SQS-Warteschlange als Warteschlange für unzustellbare Nachrichten verwenden möchten. EventBridge sendet Ereignisse, die dieser Regel entsprechen, an die Warteschlange für Dead-Letter, wenn sie nicht erfolgreich an das Ziel übermittelt wurden. Führen Sie eine der folgenden Aktionen aus:

   - Klicken Sie auf Keine, um keine Warteschlange für unzustellbare Nachrichten zu verwenden.
   - Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange im aktuellen AWS-Konto als Warteschlange für unzustellbare Nachrichten und wählen Sie dann die Warteschlange aus der Dropdown-Liste aus.
   - Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange in einem anderen AWS-Konto als Warteschlange für unzustellbare Nachrichten und geben Sie dann den ARN der Warteschlange ein, die verwendet werden soll. Sie müssen eine ressourcenbasierte Richtlinie an die Warteschlange anhängen, die EventBridge Berechtigung zum Senden von Nachrichten an die Warteschlange erteilt.

     Weitere Informationen finden Sie unter Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten.

AWS service

Beachten Sie, dass EventBridge möglicherweise nicht alle der folgenden Felder für einen bestimmten AWS-Service anzeigt.

1. (Optional) Wählen Sie für „Zieleingabe konfigurieren“ aus, wie Sie den an das Ziel gesendeten Text für passende Ereignisse anpassen möchten. Wählen Sie eine der folgenden Optionen aus:
   - Übereinstimmende Ereignisse – EventBridge sendet das gesamte ursprüngliche Quellereignis an das Ziel. Dies ist die Standardeinstellung.
   - Teil der übereinstimmenden Ereignisse – EventBridge sendet nur den angegebenen Teil des ursprünglichen Quellereignisses an das Ziel.

     Geben Sie unter Den Teil des übereinstimmenden Ereignisses angeben einen JSON-Pfad an, der den Teil des Ereignisses definiert, den EventBridge an das Ziel senden soll.
   - Konstante (JSON-Text) – EventBridge sendet nur den angegebenen JSON-Text an das Ziel. Es wird kein Teil des ursprünglichen Quellereignisses gesendet.

     Geben Sie unter Die Konstante in JSON angeben den JSON-Text an, den EventBridge anstelle des Ereignisses an das Ziel senden soll.
   - Eingabe-Transformator – Konfigurieren Sie einen Eingabe-Transformator, um den Text anzupassen, den EventBridge an das Ziel senden soll. Weitere Informationen finden Sie unter ???.

     a. Wählen Sie Eingabe-Transformator konfigurieren aus.

     b. Konfigurieren Sie den Eingabe-Transformator wie unter ??? beschrieben.
2. (Optional) Geben Sie unter Wiederholungsrichtlinie an, wie EventBridge das Senden eines Ereignisses an ein Ziel wiederholen soll, nachdem ein Fehler aufgetreten ist.
   - Maximales Alter des Ereignisses – Geben Sie die maximale Zeitspanne (in Stunden, Minuten und Sekunden) ein, für die EventBridge unverarbeitete Ereignisse beibehält. Die Standardeinstellung ist 24 Stunden.
   - Wiederholungsversuche – Geben Sie ein, wie oft EventBridge das Senden an ein Ziel maximal wiederholen soll, nachdem ein Fehler aufgetreten ist. Die Standardeinstellung ist 185 Mal.

3. (Optional) Wählen Sie für Warteschlange für unzustellbare Nachrichten aus, ob Sie eine standardmäßige Amazon-SQS-Warteschlange als Warteschlange für unzustellbare Nachrichten verwenden möchten. EventBridge sendet Ereignisse, die dieser Regel entsprechen, an die Warteschlange für Dead-Letter, wenn sie nicht erfolgreich an das Ziel übermittelt wurden. Führen Sie eine der folgenden Aktionen aus:
   - Klicken Sie auf Keine, um keine Warteschlange für unzustellbare Nachrichten zu verwenden.
   - Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange im aktuellen AWS-Konto als Warteschlange für unzustellbare Nachrichten und wählen Sie dann die Warteschlange aus der Dropdown-Liste aus.
   - Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange in einem anderen AWS-Konto als Warteschlange für unzustellbare Nachrichten und geben Sie dann den ARN der Warteschlange ein, die verwendet werden soll. Sie müssen eine ressourcenbasierte Richtlinie an die Warteschlange anhängen, die EventBridge Berechtigung zum Senden von Nachrichten an die Warteschlange erteilt.

     Weitere Informationen finden Sie unter Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten.
4. (Optional) Wählen Sie Add another target (Weiteres Ziel hinzufügen) aus, um ein weiteres Ziel für diese Regel hinzuzufügen.
5. Wählen Sie Next (Weiter).

Beachten Sie, dass EventBridge möglicherweise nicht alle der folgenden Felder für einen bestimmten AWS-Service anzeigt.

Konfigurieren von Tags und Überprüfen von Regeln

Geben Sie abschließend alle gewünschten Tags für die Regel ein, überprüfen und erstellen Sie dann die Regel.

So konfigurieren Sie Tags und überprüfen und erstellen die Regel

1. (Optional) Geben Sie ein oder mehrere Tags für die Regel ein. Weitere Informationen finden Sie unter Amazon- EventBridge Tags.
2. Wählen Sie Next (Weiter).

3. Überprüfen Sie die Details der neuen Regel. Um Änderungen an einem Abschnitt vorzunehmen, wählen Sie neben diesem Abschnitt die Schaltfläche Bearbeiten aus.

   Wenn Sie mit den Regeldetails zufrieden sind, wählen Sie Regel erstellen aus.

Verwenden von Amazon EventBridge Scheduler mit Amazon EventBridge

Amazon EventBridge Scheduler ist ein Serverless-Scheduler, mit dem Sie Aufgaben von einem zentralen, verwalteten Service aus erstellen, ausführen und verwalten können. Mit EventBridge Scheduler können Sie mithilfe von Cron- und Rate-Ausdrücken Zeitpläne für wiederkehrende Muster erstellen oder einmalige Aufrufe konfigurieren. Sie können flexible Zeitfenster für die Zustellung einrichten, Wiederholungslimits definieren und die maximale Aufbewahrungszeit für fehlgeschlagene API-Aufrufe festlegen.

EventBridge Scheduler ist hochgradig anpassbar und bietet eine verbesserte Skalierbarkeit gegenüber den geplanten EventBridge-Regeln mit einer breiteren Palette von API-Zieloperationen und AWS-Services. Wir empfehlen, dass Sie EventBridge Scheduler verwenden, um Ziele anhand eines Zeitplans aufzurufen.

Themen

Einrichten der Ausführungsrolle

Wenn Sie einen neuen Zeitplan erstellen, muss EventBridge Scheduler über die Berechtigung verfügen, seinen Ziel-API-Vorgang in Ihrem Namen aufzurufen. Sie gewähren EventBridge Scheduler diese Berechtigungen mithilfe einer Ausführungsrolle. Die Berechtigungsrichtlinie, die Sie der Ausführungsrolle Ihres Zeitplans hinzufügen, definiert die erforderlichen Berechtigungen. Diese Berechtigungen hängen von der Ziel-API ab, die EventBridge Scheduler aufrufen soll.

Wenn Sie die EventBridge-Scheduler-Konsole zum Erstellen eines Zeitplans verwenden, wie im folgenden Verfahren, richtet EventBridge Scheduler automatisch eine Ausführungsrolle basierend auf Ihrem ausgewählten Ziel ein. Wenn Sie einen Zeitplan mit einem der EventBridge-Scheduler-SDKs (AWS CLI oder AWS CloudFormation) erstellen möchten, müssen Sie über eine vorhandene Ausführungsrolle verfügen, die die Berechtigungen gewährt, die EventBridge Scheduler zum Aufrufen eines Ziels benötigt. Weitere Informationen zum manuellen Einrichten einer Ausführungsrolle für Ihren Zeitplan finden Sie unter Einrichten einer Ausführungsrolle im Benutzerhandbuch für EventBridge Scheduler.

Erstellen eines Zeitplans

So erstellen Sie einen Zeitplan mithilfe der Konsole

1. Öffnen Sie die Amazon-EventBridge-Scheduler-Konsole unter https://console.aws.amazon.com/scheduler/home.
2. Wählen Sie auf der Seite Zeitpläne die Option Zeitplan erstellen aus.
3. Gehen Sie auf der Seite Zeitplandetails angeben im Abschnitt Zeitplanname und -beschreibung wie folgt vor:

   a. Geben Sie unter Zeitplanname einen Namen für Ihren Zeitplan ein. Zum Beispiel **`MyTestSchedule`**.

   b. (Optional) Geben Sie unter Beschreibung eine Beschreibung für Ihren Zeitplan ein. Zum Beispiel **`My first schedule`**.

   c. Wählen Sie für Zeitplangruppe eine Zeitplangruppe aus der Dropdown-Liste aus. Wenn Sie noch keine Gruppe haben, wählen Sie Standard. Um eine Zeitplangruppe zu erstellen, wählen Sie Eigenen Zeitplan erstellen.

   Sie verwenden Zeitplangruppen, um Tags zu Zeitplangruppen hinzuzufügen.

4. • Wählen Sie Ihre Zeitplanoptionen.

Vorkommen	Vorgehensweise	
Einmaliger Zeitplan Ein einmaliger Zeitplan ruft ein Ziel nur einmal zu dem von Ihnen angegebenen Datum und der angegeben en Uhrzeit auf.	Gehen Sie für Datum und Uhrzeit wie folgt vor: • Geben Sie ein gültiges Datum im `YYYY/MM/DD` -Format ein. • Geben Sie einen Zeitstempel im 24-Stunden-Format (`hh:mm`) ein. • Wählen Sie unter Zeitzone die Zeitzone aus.	

Vorkommen	Vorgehensweise	
Wiederkehrender Zeitplan Ein wiederkehrender Zeitplan ruft ein Ziel mit einer Rate auf, die Sie mit einem cron-Ausdruck oder einem Rate-Ausdruck angeben.	a. Gehen Sie bei Zeitplantyp wie folgt vor: • Um den Zeitplan mithilfe eines Cron-Ausdrucks zu definieren, wählen Sie Cron-basierter Zeitplan und geben Sie den Cron-Ausdruck ein. • Um den Zeitplan mithilfe eines Rate-Ausdrucks zu definieren, wählen Sie Rate-Ausdruck ein. Weitere Informationen zu Cron- und Rate-Ausdrücken finden Sie unter Zeitplantypen im EventBridge Scheduler im Benutzerhandbuch zu Amazon EventBridge Scheduler. b. Wählen Sie für Flexibles Zeitfenster die Option Aus, um die Option zu deaktivieren, oder wählen Sie eines der vordefinierten Zeitfenster aus. Wenn Sie beispielsweise 15 Minuten auswählen und einen wiederkehrenden	

Vorkommen	Vorgehensweise	
	Zeitplan festlegen, der sein Ziel einmal pro Stunde aufruft, wird der Zeitplan innerhalb von 15 Minuten nach Beginn jeder Stunde ausgeführt.	

5. (Optional) Wenn Sie im vorherigen Schritt Wiederkehrender Zeitplan ausgewählt haben, gehen Sie im Abschnitt Zeitrahmen wie folgt vor:

   a. Wählen Sie unter Zeitzone eine Zeitzone aus.

   b. Geben Sie für Startdatum und -uhrzeit ein gültiges Datum im `YYYY/MM/DD`-Format ein und geben Sie dann einen Zeitstempel im 24-Stunden-Format (`hh:mm`) an.

   c. Geben Sie für Enddatum und -uhrzeit ein gültiges Datum im `YYYY/MM/DD`-Format ein und geben Sie dann einen Zeitstempel im 24-Stunden-Format (`hh:mm`) an.

6. Wählen Sie Next (Weiter).

7. Wählen Sie auf der Seite Ziel auswählen den AWS-API-Vorgang aus, den EventBridge Scheduler aufruft:

   a. Wählen Sie für Ziel-API die Option Vorlagenziele aus.

   b. Wählen Sie Amazon EventBridge PutEvents aus.

   c. Geben Sie unter PutEvents Folgendes an:

      - Wählen Sie für EventBridge-Event-Bus den Event Bus aus dem Dropdown-Menü aus. Zum Beispiel **`default`**.

        Sie können einen neuen Event Bus auch in der EventBridge-Konsole erstellen, indem Sie Neuen Event Bus erstellen auswählen.

      - Geben Sie für Detailtyp den Detailtyp der Ereignisse ein, die Sie abgleichen möchten. Zum Beispiel **`Object Created`**.

      - Geben Sie für Quelle den Namen des Service ein, der die Quelle der Ereignisse ist.

        Geben Sie für AWS-Serviceereignisse das Servicepräfix als Quelle an. Verwenden Sie nicht das Präfix `aws.`. Geben Sie beispielsweise für Amazon-S3-Ereignisse **`s3`** ein.

Informationen zum Ermitteln des Präfixes eines Service finden Sie in Die Tabelle der Bedingungsschlüssel in der Service-Authorization-Referenz. Weitere Informationen zu Quell- und Detailtyp-Ereigniswerten finden Sie unter ???.

- (Optional): Geben Sie für Detail ein Ereignismuster ein, um die Ereignisse, die EventBridge Scheduler an EventBridge sendet, weiter zu filtern.

  Weitere Informationen finden Sie unter ???.

8. Wählen Sie Next (Weiter).
9. Führen Sie auf der Seite Settings (Einstellungen) die folgenden Schritte aus:

   a. Um den Zeitplan zu aktivieren, schalten Sie unter Zeitplanstatus die Option Zeitplan aktivieren ein.

   b. Um eine Wiederholungsrichtlinie für Ihren Zeitplan zu konfigurieren, gehen Sie unter Wiederholungsrichtlinie und Warteschlange für unzustellbare Nachrichten (DLQ) wie folgt vor:

      - Aktivieren Sie die Option Wiederholen.
      - Geben Sie unter Maximales Alter des Ereignisses die maximale(n) Stunde(n) und Minute(n) ein, die EventBridge Scheduler ein unverarbeitetes Ereignis aufbewahren muss.
      - Die Höchstdauer beträgt 24 Stunden.
      - Geben Sie unter Maximale Wiederholungsversuche an, wie oft EventBridge Scheduler den Zeitplan maximal wiederholen soll, wenn das Ziel einen Fehler zurückgibt.

        Der Maximalwert beträgt 185 Wiederholungen.

      Bei Wiederholungsrichtlinien führt EventBridge Scheduler den Zeitplan erneut aus, wenn ein Zeitplan sein Ziel nicht aufrufen kann. Falls konfiguriert, müssen Sie die maximale Aufbewahrungszeit und Wiederholungsversuche für den Zeitplan festlegen.

   c. Wählen Sie aus, wo EventBridge Scheduler nicht zugestellte Ereignisse speichert.

Option für Warteschlange für unzustellbare Nachrichten (DLQ)	Vorgehensweise	
Nicht speichern	Wählen Sie None.	
Speichert das Ereignis im selben AWS-Konto, in dem Sie den Zeitplan erstellen	a. Wählen Sie Eine Amazon-SQS-Warteschlange in meinem AWS-Konto als DLQ auswählen. b. Wählen Sie den Amazon-Ressourcennamen (ARN) der Amazon SQS-Warteschlange.	
Speichert das Ereignis in einem anderen AWS-Konto als dem, in dem Sie den Zeitplan erstellen	a. Wählen Sie Angeben einer Amazon-SQS-Warteschlange in anderen AWS-Konten als DLQ. b. Geben Sie den Amazon-Ressourcennamen (ARN) der Amazon-SQS-Warteschlange ein.	

d. Um einen kundenverwalteten Schlüssel zur Verschlüsselung Ihrer Zieleingabe zu verwenden, wählen Sie unter Verschlüsselung die Option Verschlüsselungseinstellungen anpassen (erweitert).

   Wenn Sie diese Option wählen, geben Sie einen vorhandenen CMK-ARN ein oder wählen Sie Erstellen eines AWS KMS key, um zur AWS KMS-Konsole zu navigieren. Weitere Informationen darüber, wie EventBridge Scheduler Ihre Daten im Ruhezustand verschlüsselt, finden Sie unter Verschlüsselung im Ruhezustand im Benutzerhandbuch zu Amazon EventBridge Scheduler.

e. Damit EventBridge Scheduler eine neue Ausführungsrolle für Sie erstellt, wählen Sie Neue Rolle für diesen Zeitplan erstellen. Geben Sie dann einen Namen für Rollenname ein. Wenn Sie diese Option wählen, fügt EventBridge Scheduler der Rolle die erforderlichen Berechtigungen für Ihr Vorlagenziel hinzu.

10. Wählen Sie Next (Weiter).
11. Überprüfen Sie auf der Seite Zeitplan überprüfen und erstellen die Details Ihres Zeitplans. Wählen Sie in jedem Abschnitt Bearbeiten aus, um zu diesem Schritt zurückzukehren und seine Details zu bearbeiten.
12. Wählen Sie Zeitplan erstellen.

    Auf der Seite Zeitpläne können Sie eine Liste Ihrer neuen und vorhandenen Zeitpläne anzeigen. Überprüfen Sie in der Spalte Status, ob Ihr neuer Zeitplan aktiviert ist.

Zugehörige Ressourcen

Weitere Informationen zum EventBridge Scheduler finden Sie hier:

- EventBridge-Scheduler-Benutzerhandbuch
- EventBridge-Scheduler-API-Referenz
- EventBridge Scheduler – Preisgestaltung

Erstellen einer Amazon-EventBridge-Regel, die nach einem Zeitplan ausgeführt wird

Eine Regel kann als Antwort auf ein Ereignis oder in bestimmten Zeitintervallen ausgeführt werden. Um beispielsweise eine AWS Lambda-Funktion regelmäßig auszuführen, können Sie eine Regel erstellen, die nach einem Zeitplan ausgeführt wird.

> **Note**
>
> EventBridge bietet Amazon EventBridge Scheduler, wobei es sich um einen Serverless-Scheduler handelt, mit dem Sie Aufgaben von einem zentralen, verwalteten Service aus erstellen, ausführen und verwalten können. EventBridge Scheduler ist hochgradig anpassbar und bietet eine verbesserte Skalierbarkeit gegenüber den geplanten EventBridge-Regeln mit einer breiteren Palette von API-Zieloperationen und AWS-Services.

Wir empfehlen, dass Sie EventBridge Scheduler verwenden, um Ziele anhand eines Zeitplans aufzurufen. Weitere Informationen finden Sie unter ???.

Sie können in EventBridge zwei Arten von geplanten Regeln erstellen:

- Regeln, die regelmäßig ausgeführt werden

  EventBridge führt diese Regeln in regelmäßigen Abständen aus, z. B. alle 20 Minuten.

  Zum Festlegen des Zeitintervalls für eine geplante Regel definieren Sie einen Rate-Ausdruck.
- Regeln, die zu bestimmten Zeiten ausgeführt werden

  EventBridge führt diese Regeln zu bestimmten Zeiten und Daten aus, z. B. um 8:00 Uhr. PST am ersten Montag jedes Monats

  Zur Angabe einer Uhrzeit und eines Datums, an dem eine geplante Regel ausgeführt wird, definieren Sie einen Cron-Ausdruck.

Rate-Ausdrücke sind einfacher zu definieren, während Cron-Ausdrücke eine detaillierte Steuerung des Zeitplans ermöglichen. Mit einem Cron-Ausdruck können Sie zum Beispiel eine Regel definieren, die zu bestimmten Uhrzeiten an bestimmten Tagen einer Woche oder eines Monats ausgeführt wird. Mit Rate-Ausdrücken dagegen wird die Regel in regelmäßigen Intervallen ausgeführt, zum Beispiel stündlich oder täglich.

Alle geplanten Ereignisse verwenden die Zeitzone UTC+0, und die Mindestanforderungen an die Genauigkeit der Zeitpläne beträgt eine Minute.

Note

EventBridge stellt in Zeitplanausdrücken keine Präzision der zweiten Ebene bereit. Die feinste Zeitauflösung bei Verwendung eines Cron-Ausdrucks ist eine Minute. Aufgrund der verteilten Natur von EventBridge und den Zielservices kann es zwischen dem Zeitpunkt der Auslösung der geplanten Regel und dem Zeitpunkt, zu dem der Zielservice die Zielressource ausführt, zu einer Verzögerung von einigen Sekunden kommen.

Das folgende Video gibt einen Überblick über die Planung von Aufgaben: Erstellen geplanter Aufgaben mit EventBridge

Themen

- Erstellen einer Regel, die nach einem Zeitplan ausgeführt wird
- Referenz für Cron-Ausdrücke
- Referenz für Rate-Ausdrücke

Erstellen einer Regel, die nach einem Zeitplan ausgeführt wird

In den folgenden Schritten erfahren Sie, wie Sie eine EventBridge-Regel erstellen, die nach einem regelmäßigen Zeitplan ausgeführt wird.

> **Note**
>
> Sie können geplante Regeln nur mit dem Standard-Event-Bus erstellen.

Schritte

- Definieren der Regel
- Definieren des Zeitplans
- Auswählen von Zielen
- Konfigurieren von Tags und Überprüfen von Regeln

Definieren der Regel

Geben Sie zunächst einen Namen und eine Beschreibung für die Regel ein, um sie zu identifizieren.

So definieren Sie die Regeldetails

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen.
4. Geben Sie für die Regel einen Namen und optional eine Beschreibung ein.

Eine Regel darf nicht denselben Namen wie eine andere Regel in derselben AWS-Region und auf demselben Event Bus haben.

5. Wählen Sie für Event Bus den Standard-Event-Bus aus. Sie können geplante Regeln nur mit dem Standard-Event-Bus erstellen.
6. Damit die Regel sofort nach ihrer Erstellung wirksam wird, stellen Sie sicher, dass die Option Regel für den ausgewählten Event Bus aktivieren aktiviert ist.
7. Wählen Sie unter Rule type (Regeltyp) die Option Schedule (Zeitplan) aus.

   An dieser Stelle können Sie wählen, ob Sie mit der Erstellung einer Regel fortfahren möchten, die nach einem Zeitplan ausgeführt wird, oder Amazon EventBridge Scheduler verwenden möchten.
8. Wählen Sie aus, wie Sie fortfahren möchten:

   - Verwenden von EventBridge Scheduler, um Ihren Zeitplan zu erstellen

     > **Note**
     >
     > EventBridge Scheduler ist ein Serverless-Scheduler, mit dem Sie Aufgaben von einem zentralen, verwalteten Service aus erstellen, ausführen und verwalten können. Er bietet Funktionen zur einmaligen und wiederkehrenden Terminplanung, unabhängig von Event Buses und Regeln. EventBridge Scheduler ist hochgradig anpassbar und bietet eine verbesserte Skalierbarkeit gegenüber den geplanten EventBridge-Regeln mit einer breiteren Palette von API-Zieloperationen und AWS-Services.
     > Wir empfehlen, dass Sie EventBridge Scheduler verwenden, um Ziele anhand eines Zeitplans aufzurufen. Weitere Informationen finden Sie unter Was ist Amazon EventBridge Scheduler? im Benutzerhandbuch von Amazon EventBridge Scheduler.

     1. Wählen Sie In EventBridge Scheduler fortfahren aus.

        EventBridge öffnet die EventBridge-Scheduler-Konsole auf der Seite Zeitplan erstellen.
     2. Erstellen Sie den Zeitplan in der EventBridge-Scheduler-Konsole.
   - Verwenden Sie weiterhin EventBridge, um eine geplante Regel für den Standard-Event-Bus zu erstellen.

     1. Wählen Sie Mit dem Erstellen einer Regel fortfahren aus.

Definieren des Zeitplans

Als Nächstes definieren Sie das Zeitplanmuster.

So definieren Sie das Zeitplanmuster

1. Wählen Sie für Zeitplanmuster aus, ob der Zeitplan zu einer bestimmten Zeit oder in regelmäßigen Intervallen ausgeführt werden soll:

   Specific time

   1. Wählen Sie Ein detaillierter Zeitplan, der zu einer bestimmten Zeit ausgeführt wird, z. B. um 8:00 Uhr aus. PST am ersten Montag jedes Monats
   2. Geben Sie für Cron-Ausdruck Felder an, um den Cron-Ausdruck zu definieren, den EventBridge verwenden sollte, um zu bestimmen, wann diese geplante Regel ausgeführt werden soll.

      Nachdem Sie alle Felder angegeben haben, zeigt EventBridge die nächsten zehn Daten an, an denen EventBridge diese geplante Regel ausführen wird. Sie können wählen, ob diese Daten in UTC oder in der lokalen Zeitzone angezeigt werden sollen.

      Weitere Informationen zum Erstellen eines Cron-Ausdrucks finden Sie unter ???.

   Regular rate

   1. Wählen Sie Ein Zeitplan, der regelmäßig ausgeführt wird, z. B. alle 10 Minuten aus.
   2. Geben Sie für Rate-Ausdruck die Felder Wert und Einheit an, um das Zeitintervall zu definieren, mit dem EventBridge diese geplante Regel ausführen soll.

      Weitere Informationen zum Erstellen eines Rate-Ausdrucks finden Sie unter ???.
2. Wählen Sie Next (Weiter).

Auswählen von Zielen

Wählen Sie ein oder mehrere Ziele aus, um Ereignisse zu empfangen, die dem angegebenen Muster entsprechen. Zu den Zielen können ein EventBridge-Event-Bus, EventBridge-API-Ziele, einschließlich SaaS-Partner wie Salesforce, oder andere AWS-Service gehören.

So wählen Sie Ziele aus

1. Wählen Sie für Zieltyp einen der folgenden Zieltypen aus:

   Event bus

   Zum Auswählen eines EventBridge-Event-Bus klicken Sie auf EventBridge-Event-Bus und gehen dann wie folgt vor:

   - So verwenden Sie einen Event Bus in derselben AWS-Region wie diese Regel:
     1. Wählen Sie Event Bus in demselben Konto und derselben Region aus.
     2. Wählen Sie für Event Bus für das Ziel das Dropdown-Feld aus und geben Sie den Namen des Event Bus ein. Sie können den Event Bus auch aus der Dropdown-Liste auswählen.

        Weitere Informationen finden Sie unter ???.
   - So verwenden Sie einen Event Bus in einer anderen AWS-Region oder einem anderen Konto als diese Regel:
     1. Wählen Sie Event Bus in einem anderen Konto oder einer anderen Region aus.
     2. Geben Sie für Event Bus als Ziel den ARN des Event Bus ein, den Sie verwenden möchten.

        Weitere Informationen finden Sie unter:

        - ???
        - ???

   API destination

   Zum Verwenden eines EventBridge-API-Ziels wählen Sie EventBridge-API-Ziel aus und führen Sie dann einen der folgenden Schritte aus:

   - Zum Verwenden eines vorhandenen API-Ziels wählen Sie Vorhandenes API-Ziel verwenden aus. Wählen Sie dann ein API-Ziel aus der Dropdown-Liste aus.
   - Zum Erstellen eines neuen API-Ziels wählen Sie Neues API-Ziel erstellen aus. Geben Sie anschließend die folgenden Details für das Ziel an:
     - Name – Geben Sie einen Namen für das Ziel ein.

Namen müssen innerhalb Ihres AWS-Konto-Kontos eindeutig sein. Namen können bis zu 64 Zeichen lang sein. Gültige Zeichen sind A-Z, a-z, 0-9 und . _ - (Bindestrich).

- (Optional) Beschreibung – Geben Sie eine Beschreibung für das Ziel ein.

  Beschreibungen können bis zu 512 Zeichen lang sein.

- API-Zielendpunkt – Der URL-Endpunkt für das Ziel.

  Die Endpunkt-URL muss mit **`https`** beginnen. Sie können das **`*`** als Pfadparameterplatzhalter angeben. Sie können Pfadparameter über das `HttpParameters`-Attribut des Ziels festlegen.

- HTTP-Methode – Wählen Sie die HTTP-Methode aus, die beim Aufrufen des Endpunkts verwendet wird.
- (Optional) Aufrufratenlimit pro Sekunde – Geben Sie die maximale Anzahl von Aufrufen ein, die pro Sekunde für dieses Ziel akzeptiert werden.

  Dieser Wert muss größer als null sein. Standardmäßig ist dieser Wert auf 300 festgelegt.

- Verbindung – Wählen Sie, ob Sie eine neue oder eine vorhandene Verbindung verwenden möchten:
  - Zum Verwenden einer vorhandenen Verbindung wählen Sie Vorhandene Verbindung verwenden und die Verbindung aus der Dropdown-Liste aus.
  - Zum Erstellen einer neuen Verbindung für dieses Ziel wählen Sie Neue Verbindung erstellen aus und definieren dann den Namen, den Zieltyp und den Autorisierungstyp der Verbindung. Sie können auch eine optionale Beschreibung für diese Verbindung hinzufügen.

Weitere Informationen finden Sie unter ???.

AWS-Service

Zum Verwenden eines AWS-Service wählen Sie AWS-Service aus und gehen dann wie folgt vor:

1. Wählen Sie für Ziel auswählen einen AWS-Service aus, der als Ziel verwendet werden soll. Geben Sie die angeforderten Informationen für den ausgewählten Service ein.

> **Note**
>
> Die angezeigten Felder variieren je nach ausgewähltem Service. Weitere Informationen zu verfügbaren Zielen finden Sie unter In der EventBridge Konsole verfügbare Ziele.

2. Für viele Zieltypen benötigt EventBridge Berechtigungen zum Senden von Ereignissen an das Ziel. In diesen Fällen kann EventBridge die IAM-Rolle erstellen, die zum Ausführen Ihrer Regel erforderlich ist.

   Führen Sie für Ausführungsrolle einen der folgenden Schritte aus:

   - Gehen Sie wie folgt vor, um eine neue Ausführungsrolle für diese Regel zu erstellen:

     a. Wählen Sie Eine neue Rolle für diese spezifische Ressource erstellen aus.

     b. Geben Sie entweder einen Namen für diese Ausführungsrolle ein, oder verwenden Sie den von EventBridge generierten Namen.

   - So verwenden Sie eine vorhandene Ausführungsrolle für diese Regel:

     a. Wählen Sie Vorhandene Rolle verwenden aus.

     b. Geben Sie den Namen der zu verwendenden Ausführungsrolle ein oder wählen Sie ihn aus der Dropdown-Liste aus.

3. (Optional) Geben Sie für Zusätzliche Einstellungen eine der optionalen Einstellungen an, die für Ihren Zieltyp verfügbar sind:

   Event bus

   (Optional) Wählen Sie für Warteschlange für unzustellbare Nachrichten aus, ob Sie eine standardmäßige Amazon-SQS-Warteschlange als Warteschlange für unzustellbare Nachrichten verwenden möchten. EventBridge sendet Ereignisse, die dieser Regel entsprechen, an die Warteschlange für Dead-Letter, wenn sie nicht erfolgreich an das Ziel übermittelt wurden. Führen Sie eine der folgenden Aktionen aus:

   - Klicken Sie auf Keine, um keine Warteschlange für unzustellbare Nachrichten zu verwenden.
   - Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange im aktuellen AWS-Konto als Warteschlange für unzustellbare Nachrichten und wählen Sie dann die Warteschlange aus der Dropdown-Liste aus.

- Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange in einem anderen AWS-Konto als Warteschlange für unzustellbare Nachrichten und geben Sie dann den ARN der Warteschlange ein, die verwendet werden soll. Sie müssen eine ressourcenbasierte Richtlinie an die Warteschlange anhängen, die EventBridge Berechtigung zum Senden von Nachrichten an die Warteschlange erteilt.

  Weitere Informationen finden Sie unter Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten.

API destination

1. (Optional) Wählen Sie für „Zieleingabe konfigurieren“ aus, wie Sie den an das Ziel gesendeten Text für passende Ereignisse anpassen möchten. Wählen Sie eine der folgenden Optionen aus:
   - Übereinstimmende Ereignisse – EventBridge sendet das gesamte ursprüngliche Quellereignis an das Ziel. Dies ist die Standardeinstellung.
   - Teil der übereinstimmenden Ereignisse – EventBridge sendet nur den angegebenen Teil des ursprünglichen Quellereignisses an das Ziel.

     Geben Sie unter Den Teil des übereinstimmenden Ereignisses angeben einen JSON-Pfad an, der den Teil des Ereignisses definiert, den EventBridge an das Ziel senden soll.
   - Konstante (JSON-Text) – EventBridge sendet nur den angegebenen JSON-Text an das Ziel. Es wird kein Teil des ursprünglichen Quellereignisses gesendet.

     Geben Sie unter Die Konstante in JSON angeben den JSON-Text an, den EventBridge anstelle des Ereignisses an das Ziel senden soll.
   - Eingabe-Transformator – Konfigurieren Sie einen Eingabe-Transformator, um den Text anzupassen, den EventBridge an das Ziel senden soll. Weitere Informationen finden Sie unter ???.

     a. Wählen Sie Eingabe-Transformator konfigurieren aus.

     b. Konfigurieren Sie den Eingabe-Transformator wie unter ??? beschrieben.
2. (Optional) Geben Sie unter Wiederholungsrichtlinie an, wie EventBridge das Senden eines Ereignisses an ein Ziel wiederholen soll, nachdem ein Fehler aufgetreten ist.
   - Maximales Alter des Ereignisses – Geben Sie die maximale Zeitspanne (in Stunden, Minuten und Sekunden) ein, für die EventBridge unverarbeitete Ereignisse beibehält. Die Standardeinstellung ist 24 Stunden.

- Wiederholungsversuche – Geben Sie ein, wie oft EventBridge das Senden an ein Ziel maximal wiederholen soll, nachdem ein Fehler aufgetreten ist. Die Standardeinstellung ist 185 Mal.

3. (Optional) Wählen Sie für Warteschlange für unzustellbare Nachrichten aus, ob Sie eine standardmäßige Amazon-SQS-Warteschlange als Warteschlange für unzustellbare Nachrichten verwenden möchten. EventBridge sendet Ereignisse, die dieser Regel entsprechen, an die Warteschlange für Dead-Letter, wenn sie nicht erfolgreich an das Ziel übermittelt wurden. Führen Sie eine der folgenden Aktionen aus:
   - Klicken Sie auf Keine, um keine Warteschlange für unzustellbare Nachrichten zu verwenden.
   - Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange im aktuellen AWS-Konto als Warteschlange für unzustellbare Nachrichten und wählen Sie dann die Warteschlange aus der Dropdown-Liste aus.
   - Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange in einem anderen AWS-Konto als Warteschlange für unzustellbare Nachrichten und geben Sie dann den ARN der Warteschlange ein, die verwendet werden soll. Sie müssen eine ressourcenbasierte Richtlinie an die Warteschlange anhängen, die EventBridge Berechtigung zum Senden von Nachrichten an die Warteschlange erteilt.

     Weitere Informationen finden Sie unter Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten.

AWS service

Beachten Sie, dass EventBridge möglicherweise nicht alle der folgenden Felder für einen bestimmten AWS-Service anzeigt.

1. (Optional) Wählen Sie für „Zieleingabe konfigurieren“ aus, wie Sie den an das Ziel gesendeten Text für passende Ereignisse anpassen möchten. Wählen Sie eine der folgenden Optionen aus:
   - Übereinstimmende Ereignisse – EventBridge sendet das gesamte ursprüngliche Quellereignis an das Ziel. Dies ist die Standardeinstellung.
   - Teil der übereinstimmenden Ereignisse – EventBridge sendet nur den angegebenen Teil des ursprünglichen Quellereignisses an das Ziel.

Geben Sie unter Den Teil des übereinstimmenden Ereignisses angeben einen JSON-Pfad an, der den Teil des Ereignisses definiert, den EventBridge an das Ziel senden soll.

- Konstante (JSON-Text) – EventBridge sendet nur den angegebenen JSON-Text an das Ziel. Es wird kein Teil des ursprünglichen Quellereignisses gesendet.

  Geben Sie unter Die Konstante in JSON angeben den JSON-Text an, den EventBridge anstelle des Ereignisses an das Ziel senden soll.

- Eingabe-Transformator – Konfigurieren Sie einen Eingabe-Transformator, um den Text anzupassen, den EventBridge an das Ziel senden soll. Weitere Informationen finden Sie unter ???.

  a. Wählen Sie Eingabe-Transformator konfigurieren aus.

  b. Konfigurieren Sie den Eingabe-Transformator wie unter ??? beschrieben.

2. (Optional) Geben Sie unter Wiederholungsrichtlinie an, wie EventBridge das Senden eines Ereignisses an ein Ziel wiederholen soll, nachdem ein Fehler aufgetreten ist.

   - Maximales Alter des Ereignisses – Geben Sie die maximale Zeitspanne (in Stunden, Minuten und Sekunden) ein, für die EventBridge unverarbeitete Ereignisse beibehält. Die Standardeinstellung ist 24 Stunden.
   - Wiederholungsversuche – Geben Sie ein, wie oft EventBridge das Senden an ein Ziel maximal wiederholen soll, nachdem ein Fehler aufgetreten ist. Die Standardeinstellung ist 185 Mal.

3. (Optional) Wählen Sie für Warteschlange für unzustellbare Nachrichten aus, ob Sie eine standardmäßige Amazon-SQS-Warteschlange als Warteschlange für unzustellbare Nachrichten verwenden möchten. EventBridge sendet Ereignisse, die dieser Regel entsprechen, an die Warteschlange für Dead-Letter, wenn sie nicht erfolgreich an das Ziel übermittelt wurden. Führen Sie eine der folgenden Aktionen aus:

   - Klicken Sie auf Keine, um keine Warteschlange für unzustellbare Nachrichten zu verwenden.
   - Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange im aktuellen AWS-Konto als Warteschlange für unzustellbare Nachrichten und wählen Sie dann die Warteschlange aus der Dropdown-Liste aus.
   - Klicken Sie auf Wählen Sie eine Amazon SQS Warteschlange in einem anderen AWS-Konto als Warteschlange für unzustellbare Nachrichten und geben Sie dann den ARN der Warteschlange ein, die verwendet werden soll. Sie müssen eine ressourcenbasierte

Richtlinie an die Warteschlange anhängen, die EventBridge Berechtigung zum Senden von Nachrichten an die Warteschlange erteilt.

Weitere Informationen finden Sie unter Erteilen von Berechtigungen für die Warteschlange für unzustellbare Nachrichten.

4. (Optional) Wählen Sie Add another target (Weiteres Ziel hinzufügen) aus, um ein weiteres Ziel für diese Regel hinzuzufügen.
5. Wählen Sie Next (Weiter).

Konfigurieren von Tags und Überprüfen von Regeln

Geben Sie abschließend alle gewünschten Tags für die Regel ein, überprüfen und erstellen Sie dann die Regel.

So konfigurieren Sie Tags und überprüfen und erstellen die Regel

1. (Optional) Geben Sie ein oder mehrere Tags für die Regel ein. Weitere Informationen finden Sie unter Amazon- EventBridge Tags.
2. Wählen Sie Next (Weiter).
3. Überprüfen Sie die Details der neuen Regel. Um Änderungen an einem Abschnitt vorzunehmen, wählen Sie neben diesem Abschnitt die Schaltfläche Bearbeiten aus.

   Wenn Sie mit den Regeldetails zufrieden sind, wählen Sie Regel erstellen aus.

Referenz für Cron-Ausdrücke

Cron-Ausdrücke verfügen über sechs Pflichtfelder, die durch Leerzeichen voneinander getrennt sind.

Syntax

```
cron(fields)
```

Feld	Werte	Platzhalter
Minuten	0-59	, - * /
Stunden	0-23	, - * /

Feld	Werte	Platzhalter
Tag des Monats	1-31	, - * ? / L W
Monat	1-12 oder JAN-DEC	, - * /
Wochentag	1-7 oder SUN-SAT	, - * ? / L #
Jahr	1970-2199	, - * /

Platzhalter

- Das Platzhalterzeichen , (Komma) schließt zusätzliche Werte ein. Im Feld Monat steht JAN, FEB, MAR für Januar, Februar und März.
- Das Platzhalterzeichen - (Bindestrich) gibt einen Bereich an. Im Feld Tag steht 1-15 für die Tage 1 bis 15 des angegebenen Monats.
- Das Platzhalterzeichen * (Sternchen) steht für alle Werte im Feld. Im Feld für die Stundenangaben steht * für alle Stunden. Sie können kein Sternchen (*) gleichzeitig in den beiden Feldern Tag des Monats und Wochentag verwenden. Wenn Sie es in einem der Felder eingeben, müssen Sie im anderen Feld ein ? verwenden.
- Das Platzhalterzeichen / (Schrägstrich) steht für schrittweise Steigerungen. Im Feld "Minuten" können Sie 1/10 eingeben, um einen Bereich von je 10 Minuten beginnend mit der ersten Minute der Stunde anzugeben (z. B. die 11., 21. und 31. Minute usw.).
- Das Platzhalterzeichen ? (Fragezeichen) steht für einen beliebigen Wert. Im Feld Tag des Monats könnten Sie 7 eingeben, und wenn ein beliebiger Wochentag akzeptabel wäre, könnten Sie im Feld Wochentag ? eingeben.
- Das Platzhalterzeichen L in den Feldern für den Tag des Monats oder für den Wochentag gibt den letzten Tag des Monats oder der Woche an.
- Das Platzhalterzeichen **`W`** im Feld "Tag des Monats" gibt einen Wochentag an. Im Feld für den Tag des Monats gibt **`3W`** den Wochentag an, der dem dritten Tag des Monats am nächsten ist.
- Mit dem #-Platzhalter im Feld für den Wochentag wird eine bestimmte Instance des angegebenen Wochentags innerhalb eines Monats angegeben. Beispiel: 3#2 steht für den zweiten Dienstag des Monats: Die 3 bezieht sich auf Dienstag, da dies der dritte Tag jeder Woche ist, und die 2 bezieht sich auf den zweiten Tag dieses Typs innerhalb des Monats.

> **Note**
>
> Wenn Sie ein '#' -Zeichen verwenden, können Sie nur einen Ausdruck im Wochentag-Feld definieren. Beispiel, `"3#1,6#3"` ist ungültig, da es als zwei Ausdrücke interpretiert wird.

Einschränkungen

- Es ist nicht möglich, die Felder für den Tag des Monats und den Wochentag im gleichen Cron-Ausdruck anzugeben. Wenn Sie in einem der Felder einen Wert oder ein * (Sternchen) angeben, müssen Sie ein ? (Fragezeichen) im anderen verwenden.
- Cron-Ausdrücke, die zu schnelleren Häufigkeiten als mit 1 Minute führen, werden nicht unterstützt.

Beispiele

Sie können die folgenden Beispiel-Cron-Zeichenfolgen beim Erstellen einer Regel mit Zeitplan verwenden.

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

Minuten	Stunden	Tag des Monats	Monat	Wochentag	Jahr	Bedeutung
						18:00 Uhr (UTC+0)
0	8	1	*	?	*	Ausführung jeden 1. Tag des Monats um 08:00 Uhr (UTC+0)
0/15	*	*	*	?	*	Ausführung alle 15 Minuten
0/10	*	?	*	MO-FR	*	Ausführung alle 10 Minuten von Montag bis Freitag
0/5	8-17	?	*	MO-FR	*	Ausführung alle 5 Minuten von Montag bis Freitag zwischen 08:00 Uhr und 17:55 Uhr (UTC+0)

Minuten	Stunden	Tag des Monats	Monat	Wochentag	Jahr	Bedeutung
0/30	20-2	?	*	MO-FR	*	Ausführung alle 30 Minuten von Montag bis Freitag zwischen 22:00 Uhr am Starttag und 02:00 Uhr am Folgetag (UTC) Ausführung von 00:00 Uhr bis 02:00 Uhr am Montagmorgen (UTC).

Im folgenden Beispiel wird eine Regel erstellt, die jeden Tag um 12:00 Uhr UTC+0 ausgeführt wird.

```
aws events put-rule --schedule-expression "cron(0 12 * * ? *)" --name MyRule1
```

Im folgenden Beispiel wird eine Regel erstellt, die jeden Tag um 14:05 Uhr und 14:35 Uhr UTC+0 ausgeführt wird.

```
aws events put-rule --schedule-expression "cron(5,35 14 * * ? *)" --name MyRule2
```

Im folgenden Beispiel wird eine Regel erstellt, die in den Jahren 2019 bis 2022 an jedem letzten Freitag des Monats um 10:15 Uhr UTC+0 ausgeführt wird.

```
aws events put-rule --schedule-expression "cron(15 10 ? * 6L 2019-2022)" --name MyRule3
```

Referenz für Rate-Ausdrücke

Ein Rate-Ausdruck beginnt, wenn Sie die Regel für geplante Ereignisse erstellen, und wird dann nach einem definierten Zeitplan ausgeführt.

Rate-Ausdrücke verfügen über zwei Pflichtfelder, die durch Leerzeichen voneinander getrennt sind.

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, dann muss die Einheit im Singular stehen. Wenn der Wert größer als 1 ist, muss die Einheit im Plural sein. Beispiel: rate(1 Stunden) und rate(5 Stunde) sind nicht gültig, aber rate(1 Stunde) und rate(5 Stunden) sind gültig.

Beispiele

Die folgenden Beispiele zeigen, wie Sie Rate-Ausdrücke mit dem AWS CLI-Befehl `put-rule` verwenden. Das erste Beispiel löst die Regel einmal jede Minute aus, das zweite Beispiel einmal alle fünf Minuten, das dritte Beispiel einmal jede Stunde und das letzte Beispiel einmal jeden Tag.

```
aws events put-rule --schedule-expression "rate(1 minute)" --name MyRule2
```

```
aws events put-rule --schedule-expression "rate(5 minutes)" --name MyRule3
```

```
aws events put-rule --schedule-expression "rate(1 hour)" --name MyRule4
```

```
aws events put-rule --schedule-expression "rate(1 day)" --name MyRule5
```

Deaktivieren oder Löschen einer Amazon-EventBridge-Regel

Um zu verhindern, dass eine Regel Ereignisse verarbeitet oder nach einem Zeitplan ausgeführt wird, können Sie die Regel löschen oder deaktivieren. In den folgenden Schritten erfahren Sie, wie Sie eine EventBridge-Regel löschen oder deaktivieren.

So löschen oder deaktivieren Sie eine Regel

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.

   Wählen Sie in Event bus (Ereignisbus) den Ereignisbus aus, der der Regel zugeordnet ist.
3. Führen Sie eine der folgenden Aktionen aus:

   a. Wählen Sie zum Löschen einer Regel die Schaltfläche neben der Regel und dann Actions, Delete und Delete aus.

      Wenn es sich bei der Regel um eine verwaltete Regel handelt, geben Sie den Namen der Regel ein, um zu bestätigen, dass es sich um eine verwaltete Regel handelt und dass das Löschen dieser Regel die Funktionalität des Service, der die Regel erstellt hat, beeinträchtigen kann. Um fortzufahren, geben Sie den Namen der Regel ein und wählen Force delete (Löschen erzwingen) aus.

   b. Wenn Sie eine Regel vorübergehend deaktivieren möchten, wählen Sie die Schaltfläche neben der Regel und dann Actions (Aktionen) und Disable (Deaktivieren) aus.

      Sie können eine verwaltete Regel nicht deaktivieren.

Bewährte Methoden bei der Definition von Amazon-EventBridge-Regeln

Im Folgenden finden Sie einige bewährte Methoden, die Sie bei der Erstellung von Regeln für Ihre Event Buses berücksichtigen sollten.

Festlegen eines einzelnen Ziels für jede Regel

Sie können zwar bis zu fünf Ziele für eine bestimmte Regel angeben, die Verwaltung von Regeln ist jedoch einfacher, wenn Sie für jede Regel ein einzelnes Ziel angeben. Wenn mehrere Ziele dieselbe Gruppe von Ereignissen empfangen müssen, empfehlen wir, die Regel zu duplizieren, um dieselben

Ereignisse an verschiedene Ziele zu übermitteln. Diese Kapselung vereinfacht die Verwaltung von Regeln: Wenn die Anforderungen der Ereignisziele im Laufe der Zeit voneinander abweichen, können Sie jede Regel und ihr Ereignismuster unabhängig von den anderen aktualisieren.

Festlegen von Regelberechtigungen

Sie können ereignisverbrauchenden Anwendungskomponenten oder -services ermöglichen, die Verwaltung ihrer eigenen Regeln selbst zu steuern. Ein gängiger Architekturansatz von Kunden besteht darin, diese Anwendungskomponenten oder -services mithilfe separater AWS-Konten zu isolieren. Um den Ablauf von Ereignissen von einem Konto zum anderen zu ermöglichen, müssen Sie für einen Event Bus eine Regel erstellen, die Ereignisse an einen Event Bus in einem anderen Konto weiterleitet. Sie können ereignisverbrauchenden Teams oder Services die Kontrolle über die Verwaltung ihrer eigenen Regeln geben. Dazu geben Sie mithilfe von Ressourcenrichtlinien die entsprechenden Berechtigungen für ihre Konten an. Dies funktioniert konto- und regionsübergreifend.

Weitere Informationen finden Sie unter ???.

Beispiele für Ressourcenrichtlinien finden Sie unter Multi-account design patterns with Amazon EventBridge auf GitHub.

Überwachen der Regelleistung

Überwachen Sie Ihre Regeln, um sicherzustellen, dass sie erwartungsgemäß funktionieren:

- Wenn Sie die Metrik `TriggeredRules` auf fehlende Datenpunkte oder Anomalien überwachen, können Sie Unstimmigkeiten bei einem Publisher erkennen, der eine grundlegende Änderung vorgenommen hat. Weitere Informationen finden Sie unter ???.
- Ein Alarm bei Anomalien oder der maximal erwarteten Anzahl kann auch dabei helfen, zu erkennen, ob eine Regel mit neuen Ereignissen übereinstimmt. Dies kann passieren, wenn Ereignis-Publisher, einschließlich AWS-Services und SaaS-Partner, neue Ereignisse beim Aktivieren neuer Anwendungsfälle und Features einführen. Wenn diese neuen Ereignisse unerwartet auftreten und dazu führen, dass das Volumen die Verarbeitungsrate des nachgelagerten Ziels übersteigt, können sie zu einem Ereignis-Backlog führen.

  Eine solche Verarbeitung unerwarteter Ereignisse kann auch zu unerwünschten Abrechnungsgebühren führen.

  Es kann auch zu einer Drosselung der Regeln führen, wenn das Konto sein Servicekontingent für aggregierte Zielaufrufe pro Sekunde überschreitet. EventBridge versucht weiterhin, Ereignisse zu

übermitteln, denen gedrosselte Regeln entsprechen, und wiederholt es bis zu 24 Stunden oder wie in der benutzerdefinierten Wiederholungsrichtlinie des Ziels beschrieben. Mithilfe der Metrik `ThrottledRules` können Sie gedrosselte Regeln erkennen und bei diesen einen Alarm auslösen.

- In Anwendungsfällen mit niedriger Latenz können Sie die Latenz auch mithilfe von `IngestionToInvocationStartLatency` überwachen, was einen Hinweis auf den Zustand Ihres Event Bus gibt. Längere Zeiträume mit hoher Latenz von mehr als 30 Sekunden können auf eine Serviceunterbrechung oder Regeldrosselung hinweisen.

Verwenden von Amazon EventBridge und AWS Serverless Application Model-Vorlagen

Sie können Regeln manuell in der EventBridge-Konsole erstellen und testen, was den Entwicklungsprozess bei der Verfeinerung von Ereignismustern unterstützen kann. Sobald Sie jedoch bereit sind, Ihre Anwendung bereitzustellen, ist es einfacher, ein Framework wie AWS SAM zu verwenden, um beispielsweise all Ihre Serverless-Ressourcen konsistent zu starten.

Wir verwenden diese Beispielanwendung, um zu untersuchen, wie Sie AWS SAM-Vorlagen verwenden können, um EventBridge-Ressourcen zu erstellen. Die Datei template.yaml in diesem Beispiel ist eine AWS SAM-Vorlage, die vier AWS Lambda-Funktionen definiert und zwei verschiedene Möglichkeiten zur Integration der Lambda-Funktionen in EventBridge zeigt.

Eine exemplarische Vorgehensweise für diese Beispielanwendung finden Sie unter ???.

Für die Verwendung von EventBridge und AWS SAM-Vorlagen gibt es zwei Ansätze. Für einfache Integrationen, bei denen eine Lambda-Funktion durch eine Regel aufgerufen wird, ist der Ansatz Kombinierte Vorlage empfohlen. Wenn Sie über eine komplexe Weiterleitungslogik verfügen oder eine Verbindung zu Ressourcen außerhalb Ihrer AWS SAM-Vorlage herstellen, ist der Ansatz Getrennte Vorlage die bessere Wahl.

Ansätze:

- Kombinierte Vorlage
- Getrennte Vorlage

Kombinierte Vorlage

Der erste Ansatz verwendet die `Events`-Eigenschaft, um die EventBridge-Regel zu konfigurieren. Der folgende Beispielcode definiert ein Ereignis, das Ihre Lambda-Funktion aufruft.

> **Note**
>
> In diesem Beispiel wird die Regel automatisch für den standardmäßigen Event Bus erstellt, der in jedem AWS-Konto vorhanden ist. Um die Regel einem benutzerdefinierten Event Bus zuzuordnen, können Sie den `EventBusName` zur Vorlage hinzufügen.

```
atmConsumerCase3Fn:
  Type: AWS::Serverless::Function
  Properties:
    CodeUri: atmConsumer/
    Handler: handler.case3Handler
    Runtime: nodejs12.x
    Events:
      Trigger:
        Type: CloudWatchEvent
        Properties:
          Pattern:
            source:
              - custom.myATMapp
            detail-type:
              - transaction
            detail:
              result:
                - "anything-but": "approved"
```

Dieser YAML-Code entspricht einem Ereignismuster in der EventBridge-Konsole. In YAML müssen Sie nur das Ereignismuster definieren und AWS SAM erstellt automatisch eine IAM-Rolle mit den erforderlichen Berechtigungen.

Getrennte Vorlage

Beim zweiten Ansatz zur Definition einer EventBridge-Konfiguration in AWS SAM sind die Ressourcen in der Vorlage klarer voneinander getrennt.

1. Zuerst definieren Sie die Lambda-Funktion:

   ```
   atmConsumerCase1Fn:
     Type: AWS::Serverless::Function
     Properties:
       CodeUri: atmConsumer/
       Handler: handler.case1Handler
       Runtime: nodejs12.x
   ```

2. Als Nächstes definieren Sie die Regel mithilfe einer `AWS::Events::Rule`-Ressource. Die Eigenschaften definieren das Ereignismuster und können auch Ziele angeben. Sie können mehrere Ziele explizit definieren.

```
EventRuleCase1:
  Type: AWS::Events::Rule
  Properties:
    Description: "Approved transactions"
    EventPattern:
      source:
        - "custom.myATMapp"
      detail-type:
        - transaction
      detail:
        result:
          - "approved"
    State: "ENABLED"
    Targets:
      -
        Arn:
          Fn::GetAtt:
            - "atmConsumerCase1Fn"
            - "Arn"
        Id: "atmConsumerTarget1"
```

3. Definieren Sie abschließend eine `AWS::Lambda::Permission`-Ressource, die EventBridge die Berechtigung erteilt, das Ziel aufzurufen.

```
PermissionForEventsToInvokeLambda:
  Type: AWS::Lambda::Permission
  Properties:
    FunctionName:
      Ref: "atmConsumerCase1Fn"
    Action: "lambda:InvokeFunction"
    Principal: "events.amazonaws.com"
    SourceArn:
      Fn::GetAtt:
        - "EventRuleCase1"
        - "Arn"
```

Generieren einer AWS CloudFormation-Vorlage aus Amazon-EventBridge-Regeln

Mit AWS CloudFormation können Sie Ihre AWS-Ressourcen über Konten und Regionen hinweg zentral und wiederholbar konfigurieren und verwalten, indem Sie die Infrastruktur als Code behandeln. CloudFormation bietet Ihnen zu diesem Zweck die Möglichkeit, Vorlagen zu erstellen, die die Ressourcen definieren, die Sie bereitstellen und verwalten möchten.

EventBridge ermöglicht es Ihnen, Vorlagen aus den vorhandenen Regeln in Ihrem Konto zu generieren, um Ihnen den Einstieg in die Entwicklung von CloudFormation-Vorlagen zu erleichtern. Sie können eine einzelne Regel oder mehrere Regeln auswählen, die in die Vorlage aufgenommen werden sollen. Sie können diese Vorlagen dann als Grundlage zum Erstellen von Stacks von Ressourcen unter CloudFormation-Verwaltung verwenden.

Weitere Informationen zu CloudFormation finden Sie im AWS CloudFormation-Benutzerhandbuch..

> **Note**
>
> EventBridge enthält keine verwalteten Regeln in der generierten Vorlage.

Sie können auch eine Vorlage aus einem vorhandenen Event Bus generieren, einschließlich der Regeln, die der Event Bus enthält.

So generieren Sie eine AWS CloudFormation-Vorlage aus einer oder mehreren Regeln

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie unter Event Bus auswählen den Event Bus aus, der die Regeln enthält, die Sie in die Vorlage aufnehmen möchten.
4. Wählen Sie unter Regeln die Regeln aus, die Sie in die generierte AWS CloudFormation-Vorlage aufnehmen möchten.

   Bei einer einzelnen Regel können Sie auch den Namen der Regel auswählen, um die Seite mit den Details zu der Regel anzuzeigen.
5. Wählen Sie die Option CloudFormation-Vorlage und dann, in welchem Format EventBridge die Vorlage generieren soll: JSON oder YAML.

EventBridge zeigt die Vorlage an, die im ausgewählten Format generiert wurde.

6. EventBridge bietet Ihnen die Möglichkeit, die Vorlagendatei herunterzuladen oder die Vorlage in die Zwischenablage zu kopieren.

   - Wählen Sie zum Herunterladen der Vorlagendatei Herunterladen aus.
   - Wählen Sie zum Kopieren der Vorlage in die Zwischenablage Kopieren aus.

7. Wählen Sie zum Beenden der Vorlage Abbrechen aus.

Sobald Sie Ihre AWS CloudFormation-Vorlage an Ihren Anwendungsfall angepasst haben, können Sie sie verwenden, um in AWS CloudFormation Stacks zu erstellen.

Überlegungen zur Verwendung von CloudFormation-Vorlagen, die von Amazon EventBridge generiert wurden

Berücksichtigen Sie die folgenden Faktoren, wenn Sie eine CloudFormation-Vorlage verwenden, die Sie von EventBridge aus generiert haben:

- EventBridge enthält keine Passwörter in der generierten Vorlage.

  Sie können die Vorlage so bearbeiten, dass sie Vorlagenparameter enthält, mit denen Benutzer Passwörter oder andere vertrauliche Informationen angeben können, wenn sie die Vorlage zum Erstellen oder Aktualisieren eines CloudFormation-Stacks verwenden.

  Darüber hinaus können Benutzer Secrets Manager verwenden, um ein Secret in der gewünschten Region zu erstellen und dann die generierte Vorlage so zu bearbeiten, dass dynamische Parameter eingesetzt werden.
- Die Ziele in der generierten Vorlage bleiben genau so, wie sie im ursprünglichen Event Bus angegeben wurden. Dies kann zu regionsübergreifenden Problemen führen, wenn Sie die Vorlage nicht entsprechend bearbeiten, bevor Sie sie zum Erstellen von Stacks in anderen Regionen verwenden.

  Darüber hinaus erstellt die generierte Vorlage die nachgelagerten Ziele nicht automatisch.

EventBridge Amazon-Ziele

Ein Ziel ist eine Ressource oder ein Endpunkt, EventBridge an den ein Ereignis gesendet wird, wenn das Ereignis dem für eine Regel definierten Ereignismuster entspricht. Die Regel verarbeitet die Daten des Ereignisses und sendet die relevanten Informationen an das Ziel. Um Ereignisdaten an ein Ziel zu senden, EventBridge ist eine Zugriffsberechtigung für die Zielressource erforderlich. Sie können für jede Regel bis zu fünf Ziele definieren.

Wenn Sie einer Regel Ziele hinzufügen und diese kurz darauf ausgeführt wird, werden neue oder aktualisierte Ziele möglicherweise nicht sofort aufgerufen. Warten Sie einen Augenblick, bis die Änderungen wirksam werden.

Das folgende Video behandelt die Grundlagen von Zielen: Was ist ein Ziel?

In der EventBridge Konsole verfügbare Ziele

Sie können die folgenden Ziele für Ereignisse in der EventBridge Konsole konfigurieren:

- API-Ziel
- API Gateway
- AWS AppSync;
- Stapelauftrag-Warteschlange
- CloudWatch Gruppe protokollieren
- CodeBuild Projekt
- CodePipeline
- `CreateSnapshot`-API-Aufruf von Amazon EBS
- EC2 Image Builder
- `RebootInstances`-API-Aufruf von EC2
- `StopInstances`-API-Aufruf von EC2
- `TerminateInstances`-API-Aufruf von EC2
- ECS-Aufgabe
- Event Bus in einem anderen Konto oder einer anderen Region

- Event Bus auf demselben Konto und derselben Region
- Firehose-Bereitstellungsdat
- Glue-Workflow
- Incident-Manager-Reaktionsplan
- Vorlage für die Inspector-Beurteilung
- Kinesis-Stream
- Lambda-Funktion (ASYNC)
- API-Abfragen für Amazon-Redshift-Cluster-Daten
- API-Abfragen für Amazon-Redshift-Serverless-Arbeitsgruppendaten
- SageMaker Pipeline
- Amazon SNS-Thema

  EventBridge unterstützt keine Amazon SNS FIFO-Themen (first in, first out).
- Amazon-SQS-Warteschlange
- Step-Functions-Zustandsautomat (ASYNC)
- Systems Manager Automation
- Systems Manager OpsItem
- Aufrufen von Systems Manager Run Command

Zielparameter

Einige Ziele senden die Informationen in der Event-Payload nicht an das Ziel, sondern behandeln das Ereignis als Auslöser für den Aufruf einer bestimmten API. EventBridge verwendet die Target-Parameter, um zu bestimmen, was mit diesem Ziel passiert. Diese umfassen u. a. folgende:

- API-Ziele (Die an ein API-Ziel gesendeten Daten müssen der Struktur der API entsprechen. Sie müssen das `InputTransformer`-Objekt verwenden, um sicherzustellen, dass die Daten korrekt strukturiert sind. Wenn Sie die ursprüngliche Ereignisnutzlast einbeziehen möchten, verweisen Sie darauf in der `InputTransformer`.)
- API Gateway (Die an API Gateway gesendeten Daten müssen der Struktur der API entsprechen. Sie müssen das `InputTransformer`-Objekt verwenden, um sicherzustellen, dass die Daten korrekt strukturiert sind. Wenn Sie die ursprüngliche Ereignisnutzlast einbeziehen möchten, verweisen Sie darauf in der `InputTransformer`.)

- Amazon EC2 Image Builder
- `RedshiftDataParameters` (API-Cluster für Amazon-Redshift-Daten)
- `SageMakerPipelineParameters`(Pipelines zur Erstellung von SageMaker Amazon-Runtime-Modellen)

> **Note**
>
> EventBridge unterstützt nicht die gesamte JSON-Pfad-Syntax und wertet sie zur Laufzeit aus. Die unterstützte Syntax umfasst:
>
> - Punktnotation (zum Beispiel `$.detail`)
> - Bindestriche
> - Unterstriche
> - Alphanumerische Zeichen
> - Array-Indizes
> - Platzhalter (*)

Dynamische Pfadparameter

Einige Zielparameter unterstützen die optionale dynamische JSON-Pfadsyntax. Diese Syntax ermöglicht es Ihnen, JSON-Pfade anstelle von statischen Werten anzugeben (z. B. `$.detail.state`). Der gesamte Wert muss ein JSON-Pfad sein, nicht nur ein Teil davon. Zum Beispiel kann `RedshiftParameters.Sql` `$.detail.state` sein, aber es kann nicht `"SELECT * FROM $.detail.state"` sein. Diese Pfade werden zur Laufzeit dynamisch durch Daten aus der Ereignisnutzlast selbst am angegebenen Pfad ersetzt. Dynamische Pfadparameter können nicht auf neue oder transformierte Werte verweisen, die sich aus der Eingabetransformation ergeben. Die unterstützte Syntax für JSON-Pfade mit dynamischen Parametern ist dieselbe wie bei der Transformation von Eingaben. Weitere Informationen finden Sie unter ???.

Die dynamische Syntax kann für alle Zeichenfolgen- und Nicht-Enum-Felder dieser Parameter verwendet werden:

- `EcsParameters`
- `HttpParameters` (außer `HeaderParameters`-Schlüssel)
- `RedshiftDataParameters`

- `SageMakerPipelineParameters`

Berechtigungen

Um API-Aufrufe für die Ressourcen zu tätigen, die Ihnen gehören, EventBridge ist eine entsprechende Genehmigung erforderlich. Für AWS Lambda und Amazon SNS SNS-Ressourcen EventBridge verwendet ressourcenbasierte Richtlinien. Für EC2-Instances, Kinesis-Datenstreams und Step Functions Functions-Zustandsmaschinen werden IAM-Rollen EventBridge verwendet, die Sie im Parameter in angeben. `RoleARN` `PutTargets` Sie können einen API-Gateway-Endpunkt mit konfigurierter IAM-Autorisierung aufrufen, aber die Rolle ist optional, wenn Sie die Autorisierung nicht konfiguriert haben. Weitere Informationen finden Sie unter Amazon EventBridge und AWS Identity and Access Management.

Wenn sich ein anderes Konto in derselben Region befindet und Ihnen die Berechtigung erteilt hat, dann können Sie Ereignisse an dieses Konto senden. Weitere Informationen finden Sie unter EventBridge Amazon-Ereignisse zwischen AWS Konten senden und empfangen.

Wenn Ihr Ziel verschlüsselt ist, müssen Sie den folgenden Abschnitt in Ihre KMS-Schlüsselrichtlinie aufnehmen.

```
{
    "Sid": "Allow EventBridge to use the key",
    "Effect": "Allow",
    "Principal": {
        "Service": "events.amazonaws.com"
    },
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
    ],
    "Resource": "*"
}
```

EventBridge Besonderheiten des Ziels

AWS Batch Job-Warteschlangen

Bestimmte Parameter AWS Batch `submitJob` können über BatchParameterskonfiguriert werden.

Andere können in der Ereignisnutzlast angegeben werden. Wenn die Nutzdaten des Ereignisses (weitergeleitet oder über InputTransformers) die folgenden Schlüssel enthalten, werden sie den `submitJob` Anforderungsparametern zugeordnet:

- `ContainerOverrides: containerOverrides`

  > **Note**
  >
  > Dazu gehören nur Befehl, Umgebung, Speicher und vcpus

- `DependsOn: dependsOn`

  > **Note**
  >
  > Dazu gehört nur jobId

- `Parameters: parameters`

CloudWatch Gruppe „Protokolle“

Wenn Sie eine nicht InputTransformermit einem CloudWatch Logs-Ziel verwenden, wird die Ereignisnutzlast als Protokollnachricht und die Quelle des Ereignisses als Zeitstempel verwendet. Wenn Sie eine verwenden InputTransformer, muss die Vorlage wie folgt aussehen:

```
{"timestamp":<timestamp>,"message":<message>}
```

EventBridge fasst die an einen Log-Stream gesendeten Einträge stapelweise zusammen und EventBridge kann daher je nach Traffic ein oder mehrere Ereignisse in einen Log-Stream übertragen.

CodeBuild Projekt

Wenn Sie InputTransformersdas Eingabeereignis so gestalten, dass es der CodeBuild StartBuildRequestStruktur entspricht, werden die Parameter 1:1 zugeordnet und an übergeben. `codeBuild.StartBuild`

Amazon-ECS-Aufgabe

Wenn Sie InputTransformersdas Eingabeereignis so gestalten, dass es der Amazon RunTask TaskOverrideECS-Struktur entspricht, werden die Parameter 1 zu 1 zugeordnet und an übergeben. `ecs.RunTask`

Incident-Manager-Antwortplan

Wenn das entsprechende Ereignis von CloudWatch Alarms stammt, werden die Details zur Änderung des Alarmstatus in die Auslösedetails des StartIncidentRequest Anrufs an Incident Manager übernommen.

Konfigurieren von Zielen

Erfahren Sie, wie Sie Einstellungen für EventBridge Ziele konfigurieren.

Ziele:

- API-Ziele
- EventBridge Amazon-Ziele für Amazon API Gateway
- AWS AppSync Ziele für Amazon EventBridge
- Verbindungen für HTTP-Endpunktziele
- EventBridge Amazon-Ereignisse zwischen AWS Konten senden und empfangen
- Senden und Empfangen von EventBridge Amazon-Events zwischen AWS Regionen
- Senden und Empfangen von EventBridge Amazon-Events zwischen Eventbussen desselben Kontos und derselben Region

API-Ziele

EventBridge Amazon-API-Ziele sind HTTP-Endpunkte, die Sie als Ziel einer Regel aufrufen können, ähnlich wie Sie einen AWS Service oder eine Ressource als Ziel aufrufen. Mithilfe von API-Zielen können Sie Ereignisse mithilfe von API-Aufrufen zwischen AWS Diensten, integrierten Software-as-a-Service (SaaS) -Anwendungen und Ihren Anwendungen außerhalb AWS von weiterleiten. Wenn Sie ein API-Ziel als Ziel einer Regel angeben, EventBridge ruft es den HTTP-Endpunkt für jedes Ereignis auf, das dem in der Regel angegebenen Ereignismuster entspricht, und übermittelt dann die Ereignisinformationen zusammen mit der Anfrage. Mit EventBridge können Sie jede HTTP-Methode außer CONNECT und TRACE für die Anfrage verwenden. Die am häufigsten verwendeten HTTP-Methoden sind PUT und POST. Sie können auch Eingabe-Transformatoren verwenden, um das Ereignis an die Parameter bestimmter HTTP-Endpunktparameter anzupassen. Weitere Informationen finden Sie unter Transformation Amazon EventBridge Amazon-Eingaben.

> **Note**
>
> API-Ziele unterstützen keine privaten Ziele, wie z. B. Schnittstellen-VPC-Endpunkte, einschließlich privater HTTPS-APIs in Virtual Private Clouds (VPC), die private Netzwerk- und Application Load Balancer- und Schnittstellen-VPC-Endpunkte verwenden.
> Weitere Informationen finden Sie unter ???.

> **Important**
>
> EventBridge Anfragen an einen API-Zielendpunkt müssen ein maximales Timeout für die Client-Ausführung von 5 Sekunden haben. Wenn die Antwort des Zielendpunkts länger als 5 Sekunden dauert, kommt es bei der Anfrage zu einem EventBridge Timeout. EventBridge wiederholt Anfragen mit Timeout bis zu den Höchstwerten, die in Ihrer Wiederholungsrichtlinie konfiguriert sind. Standardmäßig sind die Höchstwerte 24 Stunden und 185 Mal. Sobald die maximale Anzahl an Wiederholungen erreicht ist, werden Ereignisse an Ihre Warteschlange für unzustellbare Nachrichten gesendet. Andernfalls wird das Ereignis gelöscht.

Das folgende Video zeigt die Verwendung des API-Ziels: Verwenden von API-Zielen

In diesem Thema:

- Erstellen eines API-Ziels
- Erstellen von Regeln, die Ereignisse an ein API-Ziel senden
- Serviceverknüpfte Rolle für API-Ziele
- Header in Anforderungen an API-Ziele
- Fehlercodes für das API-Ziel
- Wie wirkt sich die Aufrufrate auf die Ereigniszustellung aus
- CloudEvents Ereignisse werden an API-Ziele gesendet
- API-Zielpartner

Erstellen eines API-Ziels

Jedes API-Ziel erfordert eine Verbindung. Eine Verbindung gibt den Autorisierungstyp und die Anmeldeinformationen an, die zur Autorisierung mit dem API-Zielendpunkt verwendet werden. Sie können eine vorhandene Verbindung auswählen oder gleichzeitig mit der Erstellung des API-Ziels eine Verbindung erstellen. Weitere Informationen finden Sie unter ???.

Um ein API-Ziel mit der Konsole zu erstellen EventBridge

1. Melden Sie sich AWS mit einem Konto an, das über Berechtigungen zum Verwalten EventBridge und Öffnen der EventBridgeKonsole verfügt.
2. Wählen Sie im linken Navigationsbereich API-Ziele aus.
3. Scrollen Sie nach unten zur Tabelle API-Ziele und wählen Sie dann API-Ziel erstellen aus.
4. Geben Sie auf der Seite API-Ziel erstellen einen Namen für das API-Ziel ein. Sie können bis zu 64 Groß- oder Kleinbuchstaben, Zahlen, Punkte (.), Bindestriche (-) oder Unterstriche (_) verwenden.

   Der Name muss für das Konto in der aktuellen Region eindeutig sein.
5. Geben Sie eine Beschreibung für das API-Ziel ein.
6. Geben Sie einen API-Zielendpunkt für das API-Ziel ein. Der API-Zielendpunkt ist ein HTTP-Aufrufendpunktziel für Ereignisse. Die Autorisierungsinformationen, die Sie in der für dieses API-Ziel verwendeten Verbindung angeben, werden zur Autorisierung im Hinblick auf diesen Endpunkt verwendet. Die URL muss HTTPS verwenden.
7. Geben Sie die HTTP-Methode ein, die für die Verbindung zum API-Zielendpunkt verwendet werden soll.

8. (Optional) Geben Sie für das Feld Begrenzung der Aufrufrate pro Sekunde die maximale Anzahl von Aufrufen pro Sekunde ein, die an den API-Zielendpunkt gesendet werden.

   Das von Ihnen festgelegte Ratenlimit kann sich darauf auswirken, wie EventBridge Ereignisse übertragen werden. Weitere Informationen finden Sie unter Wie wirkt sich die Aufrufrate auf die Ereigniszustellung aus.
9. Gehen Sie für Verbindung gemäß einer der folgenden Vorgehensweisen vor:
   - Wählen Sie Vorhandene Verbindung verwenden und dann die Verbindung aus, die für dieses API-Ziel verwendet werden soll.
   - Wählen Sie Neue Verbindung erstellen aus und geben Sie dann die Details für die zu erstellende Verbindung ein. Weitere Informationen finden Sie unter Verbindungen.
10. Wählen Sie Erstellen.

Erstellen von Regeln, die Ereignisse an ein API-Ziel senden

Nachdem Sie ein API-Ziel erstellt haben, können Sie es als Ziel einer Regel auswählen. Wenn Sie ein API-Ziel als Ziel verwenden möchten, müssen Sie eine IAM-Rolle mit den richtigen Berechtigungen bereitstellen. Weitere Informationen finden Sie unter ???.

Die Auswahl eines API-Ziels als Ziel ist Teil der Erstellung der Regel.

So erstellen Sie mithilfe der Konsole eine Regel, die Ereignisse an ein API-Ziel sendet

1. Befolgen Sie die Schritte im Verfahren ???.
2. Wenn Sie in dem ??? Schritt aufgefordert werden, ein API-Ziel als Zieltyp auszuwählen, gehen Sie wie folgt vor:

   a. Wählen Sie das EventBridge API-Ziel aus.

   b. Führen Sie eine der folgenden Aktionen aus:

      - Wählen Sie Bestehendes API-Ziel verwenden und wählen Sie ein vorhandenes API-Ziel aus
      - Wählen Sie Neues API-Ziel erstellen und geben Sie die erforderlichen Einstellungen an, um Ihr neues API-Ziel zu definieren.

        Weitere Informationen zur Angabe der erforderlichen Einstellungen finden Sie unter???.

c. (Optional): Um Header-Parameter für das Ereignis anzugeben, wählen Sie unter Header-Parameter die Option Header-Parameter hinzufügen aus.

   Geben Sie als Nächstes den Schlüssel und den Wert für den Header-Parameter an.

d. (Optional): Um Abfragezeichenfolgenparameter für das Ereignis anzugeben, wählen Sie unter Parameter für Abfragezeichenfolge die Option Abfragezeichenfolge-Parameter hinzufügen aus.

   Geben Sie als Nächstes den Schlüssel und den Wert für den Abfragezeichenfolgenparameter an.

3. Schließen Sie die Erstellung der Regel gemäß den Verfahrensschritten ab.

Serviceverknüpfte Rolle für API-Ziele

Wenn Sie eine Verbindung für ein API-Ziel erstellen, AWS ServiceRoleForAmazonEventBridgeApiDestinationswird Ihrem Konto eine dienstbezogene Rolle mit dem Namen hinzugefügt. EventBridge verwendet die dienstverknüpfte Rolle, um ein Geheimnis in Secrets Manager zu erstellen und zu speichern. Um der dienstbezogenen Rolle die erforderlichen Berechtigungen zu gewähren, hängt die EventBridge AmazonEventBridgeApiDestinationsServiceRolePolicyRichtlinie an die Rolle an. Die Richtlinie beschränkt die gewährten Berechtigungen auf diejenigen, die für die Interaktion der Rolle mit dem Secret für die Verbindung erforderlich sind. Es sind keine weiteren Berechtigungen enthalten und die Rolle kann nur mit den Verbindungen in Ihrem Konto interagieren, um das Secret zu verwalten.

Die folgende Richtlinie ist die `AmazonEventBridgeApiDestinationsServiceRolePolicy`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:CreateSecret",
                "secretsmanager:UpdateSecret",
                "secretsmanager:DescribeSecret",
                "secretsmanager:DeleteSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue"
            ],
```

```
            "Resource": "arn:aws:secretsmanager:*:*:secret:events!connection/*"
        }
    ]
}
```

Weitere Informationen zu serviceverknüpften Rollen finden Sie unter Verwenden serviceverknüpfter Rollen in der IAM-Dokumentation.

Die `AmazonEventBridgeApiDestinationsServiceRolePolicy` dienstbezogene Rolle wird in den folgenden Regionen unterstützt: AWS

- USA Ost (Nord-Virginia)
- USA Ost (Ohio)
- USA West (Nordkalifornien)
- USA West (Oregon)
- Africa (Cape Town)
- Asia Pacific (Hong Kong)
- Asia Pacific (Mumbai)
- Asia Pacific (Osaka)
- Asia Pacific (Seoul)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Tokio)
- Canada (Central)
- Europe (Frankfurt)
- Europa (Irland)
- Europa (London)
- Europa (Milan)
- Europe (Paris)
- Europa (Stockholm)
- Südamerika (São Paulo)
- China (Ningxia)
- China (Peking)

Header in Anforderungen an API-Ziele

Im folgenden Abschnitt wird beschrieben, wie HTTP-Header in Anfragen an API-Ziele EventBridge behandelt werden.

Header, die in Anforderungen an API-Ziele enthalten sind

EventBridge Enthält zusätzlich zu den Autorisierungsheadern, die für die Verbindung definiert sind, die für ein API-Ziel verwendet wird, die folgenden Header in jeder Anfrage.

Header-Schlüssel	Header-Wert
Benutzer-Agent	Amazonas//EventBridgeApiDestinations
Content-Type	Wenn kein benutzerdefinierter Content-Type-Wert angegeben ist, EventBridge enthält dieser Wert den folgenden Standardwert als Content-Type: application/json; charset=utf-8
Bereich	bytes=0-1048575
Accept-Encoding	gzip,deflate
Verbindung	close
Content-Length	Ein Entity-Header, der die Größe des entity-body in Byte angibt, der an den Empfänger gesendet wird
Host	Ein Anforderungsheader, der den Host und die Portnummer des Servers angibt, an den die Anfrage gesendet wird

Header, die in Anforderungen an API-Ziele nicht überschrieben werden können

EventBridge erlaubt es Ihnen nicht, die folgenden Header zu überschreiben:

- Benutzer-Agent

- Bereich

Headers EventBridge werden aus Anfragen an API-Ziele entfernt

EventBridge entfernt die folgenden Header für alle API-Zielanfragen:

- A-IM
- Accept-Charset
- Accept-Datetime
- Accept-Encoding
- Cache-Control
- Verbindung
- Content-Encoding
- Content-Length
- Inhalt-MD5
- Datum
- Expect
- Forwarded
- Aus
- Host
- HTTP2-Settings
- If-Match
- If-Modified-Since
- If-None-Match
- If-Range
- If-Unmodified-Since
- Max-Forwards
- Ursprung
- Pragma
- Proxy-Authorization
- Bereich
- Referer

- TE
- Trailer
- Transfer-Encoding
- Benutzer-Agent
- Upgrade
- Via
- Warnung

Fehlercodes für das API-Ziel

Wenn EventBridge versucht wird, ein Ereignis an ein API-Ziel zu senden, und ein Fehler auftritt, EventBridge geht Folgendes vor:

- Ereignisse, die den Fehlercodes 409, 429 und 5xx zugeordnet sind, werden wiederholt.
- Ereignisse, die den Fehlercodes 1xx, 2xx, 3xx und 4xx (außer 429) zugeordnet sind, werden nicht wiederholt.

EventBridge API-Ziele lesen den Standard-HTTP-Antwort-Header`Retry-After`, um herauszufinden, wie lange gewartet werden muss, bevor eine Folgeanfrage gestellt wird. EventBridge wählt den konservativeren Wert zwischen der definierten Wiederholungsrichtlinie und dem `Retry-After` Header. Wenn der `Retry-After` Wert negativ ist, wird der erneute Zustellungsversuch für dieses Ereignis EventBridge beendet.

Wie wirkt sich die Aufrufrate auf die Ereigniszustellung aus

Wenn Sie die Aufrufrate pro Sekunde auf einen Wert festlegen, der deutlich unter der Anzahl der generierten Aufrufe liegt, werden Ereignisse möglicherweise nicht innerhalb der 24-stündigen Wiederholungszeit für Ereignisse zugestellt. Wenn Sie beispielsweise die Aufrufrate auf 10 Aufrufe pro Sekunde festlegen, aber Tausende von Ereignissen pro Sekunde generiert werden, entsteht schnell ein Backlog an zuzustellenden Ereignissen, der 24 Stunden überschreitet. Wenn Sie sicherstellen möchten, dass keine Ereignisse verloren gehen, richten Sie eine Warteschlange für unzustellbare Nachrichten ein, an die Ereignisse mit fehlgeschlagenen Aufrufen gesendet werden, sodass Sie die Ereignisse zu einem späteren Zeitpunkt verarbeiten können. Weitere Informationen finden Sie unter Verwenden von Warteschlangen mit unzustellbaren Buchstaben zur Verarbeitung nicht zugestellter Ereignisse.

CloudEvents Ereignisse werden an API-Ziele gesendet

CloudEvents ist eine herstellerneutrale Spezifikation für die Formatierung von Ereignissen mit dem Ziel, Interoperabilität zwischen Diensten, Plattformen und Systemen zu gewährleisten. Sie können sie verwenden EventBridge , um AWS Dienstereignisse so umzuwandeln CloudEvents , dass sie an ein Ziel, z. B. ein API-Ziel, gesendet werden.

> **Note**
>
> Das folgende Verfahren erklärt, wie Quellereignisse in den strukturierten CloudEvents Modus umgewandelt werden. In der CloudEvents Spezifikation ist eine Nachricht im strukturierten Modus eine Nachricht, bei der das gesamte Ereignis (Attribute und Daten) in der Nutzlast des Ereignisses kodiert ist.

Weitere Informationen zur Spezifikation finden Sie unter cloudevents.io CloudEvents .

Um AWS Ereignisse mithilfe der Konsole in das CloudEvents Format umzuwandeln

Um Ereignisse in das CloudEvents Format vor der Übermittlung an ein Ziel umzuwandeln, erstellen Sie zunächst eine Event-Bus-Regel. Im Rahmen der Definition der Regel verwenden Sie einen Eingangstransformator für Transformationsereignisse, bevor sie an das von Ihnen angegebene Ziel gesendet werden. EventBridge

1. Befolgen Sie die Schritte im Verfahren ???.
2. Wenn Sie in dem ??? Schritt aufgefordert werden, ein API-Ziel als Zieltyp auszuwählen, gehen Sie wie folgt vor:

   a. Wählen Sie das EventBridge API-Ziel aus.
   b. Führen Sie eine der folgenden Aktionen aus:

      - Wählen Sie Bestehendes API-Ziel verwenden und wählen Sie ein vorhandenes API-Ziel aus
      - Wählen Sie Neues API-Ziel erstellen und geben Sie die erforderlichen Einstellungen an, um Ihr neues API-Ziel zu definieren.

        Weitere Informationen zur Angabe der erforderlichen Einstellungen finden Sie unter???.
   c. Geben Sie die erforderlichen Content-Type-Header-Parameter für die Ereignisse an CloudEvents :

- Wählen Sie unter Header-Parameter die Option Header-Parameter hinzufügen aus.
- Geben Sie als Schlüssel an`Content-Type`.

  Geben Sie als Wert an`application/cloudevents+json; charset=UTF-8`.

3. Geben Sie eine Ausführungsrolle für Ihr Ziel an.
4. Definieren Sie einen Eingangstransformator, um die Quellereignisdaten in das folgende CloudEvents Format umzuwandeln:

   a. Wählen Sie unter Zusätzliche Einstellungen für Zieleingabe konfigurieren die Option Eingangstransformator aus.

      Wählen Sie dann Eingangstransformator konfigurieren.

   b. Geben Sie unter Zieleingangstransformator den Eingabepfad an.

      Im Eingabepfad unten ist das Regionsattribut ein benutzerdefiniertes Erweiterungsattribut des CloudEvents Formats. Daher ist es für die Einhaltung der CloudEvents Spezifikation nicht erforderlich.

      CloudEvents ermöglicht es Ihnen, Erweiterungsattribute zu verwenden und zu erstellen, die nicht in der Kernspezifikation definiert sind. Weitere Informationen, einschließlich einer Liste bekannter Erweiterungsattribute, finden Sie unter CloudEvents Erweiterungsattribute in der CloudEvents Spezifikationsdokumentation von GitHub.

      ```
      {
        "detail": "$.detail",
        "detail-type": "$.detail-type",
        "id": "$.id",
        "region": "$.region",
        "source": "$.source",
        "time": "$.time"
      }
      ```

   c. Geben Sie unter Vorlage die Vorlage ein, um die Quellereignisdaten in das CloudEvents Format umzuwandeln.

      In der Vorlage unten `region` ist dies nicht unbedingt erforderlich, da das `region` Attribut im Eingabepfad ein Erweiterungsattribut der CloudEvents Spezifikation ist.

      ```
      {
      ```

```
  "specversion":"1.0",
  "id":<id>,
  "source":<source>,
  "type":<detail-type>,
  "time":<time>,
  "region":<region>,
  "data":<detail>
}
```

5. Schließen Sie die Erstellung der Regel gemäß den Verfahrensschritten ab.

API-Zielpartner

Verwenden Sie die von den folgenden AWS Partnern bereitgestellten Informationen, um ein API-Ziel und eine Verbindung für ihren Dienst oder ihre Anwendung zu konfigurieren.

Cisco Cloud-Observability

Endpunkt-URL des API-Zielaufrufs:

`https://tenantName.observe.appdynamics.com/rest/awsevents/aws-eventbridge-integration/endpoint`

Unterstützte Autorisierungstypen:

OAuth-Client-Anmeldeinformationen

OAuth-Token werden aktualisiert, wenn eine 401- oder 407-Antwort zurückgegeben wird

Zusätzliche Autorisierungsparameter erforderlich:

Cisco AppDynamics Client-ID und Client Secret

OAuth-Endpunkt:

`https://tenantName.observe.appdynamics.com/auth/tenantId/default/oauth2/token`

Die folgenden Parameter für das OAuth-Schlüssel/Wert-Paar:

Typ	Schlüssel	Wert
Feld „Körper“	Gewährungsart	client_credentials

Typ	Schlüssel	Wert
Header	Content-Type	Anwendung/x-www-form-urlencoded; Zeichensatz=UTF-8

AppDynamics Cisco-Dokumentation:

AWS Erfassung von Ereignissen

Häufig verwendete API-Operationen:

Nicht zutreffend

Zusätzliche Informationen:

Wenn Sie Cisco AppDynamics aus dem Drop-down-Menü Partnerziele auswählen, werden die erforderlichen OAuth-Informationen vorab ausgefüllt, einschließlich der Schlüssel/Wert-Paare für Header und Body, die für API-Aufrufe erforderlich sind.

Weitere Informationen finden Sie in der Cisco-Dokumentation unter AWS Erfassung von Ereignissen. AppDynamics

Konfluent

Endpunkt-URL des API-Zielaufrufs:

In der Regel das folgende Format:

```
https://random-id.region.aws.confluent.cloud:443/kafka/v3/
clusters/cluster-id/topics/topic-name/records
```

Weitere Informationen finden Sie unter Suchen der REST-Endpunktadresse und Cluster-ID in der Confluent-Dokumentation.

Unterstützte Autorisierungstypen:

Basic

Zusätzliche Autorisierungsparameter erforderlich:

Nicht zutreffend

Confluent-Dokumentation:

Aufzeichnungen erstellen

Confluent REST-Proxy für Apache Kafka

Häufig verwendete API-Operationen:

`POST`

Zusätzliche Informationen:

Um die Ereignisdaten in eine Nachricht umzuwandeln, die der Endpunkt verarbeiten kann, erstellen Sie einen Zieleingangstransformator.

- Um einen Datensatz ohne Angabe eines Kafka-Partitionierungsschlüssels zu generieren, verwenden Sie die folgende Vorlage für Ihren Eingangstransformator. Es ist kein Eingabepfad erforderlich.

```
{
  "value":{
      "type":"JSON",
      "data":aws.events.event.json
  },
}
```

- Um einen Datensatz mit einem Ereignisdatenfeld als Kafka-Partitionierungsschlüssel zu generieren, folgen Sie dem nachfolgenden Beispiel für den Eingabepfad und die Vorlage. Dieses Beispiel definiert den Eingabepfad für das `orderId` Feld und gibt dieses Feld dann als Partitionsschlüssel an.

  Definieren Sie zunächst den Eingabepfad für das Ereignisdatenfeld:

```
{
  "orderId":"$.detail.orderId"
}
```

  Verwenden Sie dann die Vorlage für den Eingangstransformator, um das Datenfeld als Partitionsschlüssel anzugeben:

```
{
  "value":{
    "type":"JSON",
```

```
    "data":aws.events.event.json
  },
  "key":{
    "data":"<orderId>",
    "type":"STRING"
  }
}
```

Coralogix

Endpunkt-URL des API-Zielaufrufs

Eine vollständige Liste der Endpunkte finden Sie in der Coralogix-API-Referenz.

Unterstützte Autorisierungstypen

API-Schlüssel

Zusätzliche Autorisierungsparameter erforderlich

Header `"x-amz-event-bridge-access-key"`, der Wert ist der Coralogix-API-Schlüssel

Coralogix-Dokumentation

EventBridgeAmazon-Authentifizierung

Häufig verwendete API-Operationen

USA: https://ingress.coralogix.us/aws/event-bridge

Singapur: https://ingress.coralogixsg.com/aws/event-bridge

Irland: https://ingress.coralogix.com/aws/event-bridge

Stockholm: https://ingress.eu2.coralogix.com/aws/event-bridge

Indien: https://ingress.coralogix.in/aws/event-bridge

Zusätzliche Informationen

Die Ereignisse werden als Protokolleinträge mit `applicationName=[AWS Account]` und `subsystemName=[event.source]` gespeichert.

Datadog

Endpunkt-URL des API-Zielaufrufs

Eine vollständige Liste der Endpunkte finden Sie in der Datadog-API-Referenz.

Unterstützte Autorisierungstypen

API-Schlüssel

Zusätzliche Autorisierungsparameter erforderlich

None

Datadog-Dokumentation

Authentifizierung

Häufig verwendete API-Operationen

POST https://api.datadoghq.com/api/v1/events

POST https://http-intake.logs.datadoghq.com/v1/input

Zusätzliche Informationen

Endpunkt-URLs unterscheiden sich je nach Standort Ihrer Datadog-Organisation. Die richtige URL für Ihre Organisation finden Sie in der Dokumentation.

Freshworks

Endpunkt-URL des API-Zielaufrufs

Eine Liste der Endpunkte finden Sie unter https://developers.freshworks.com/documentation/.

Unterstützte Autorisierungstypen

Basic, API Key

Zusätzliche Autorisierungsparameter erforderlich

Nicht zutreffend

Freshworks-Dokumentation

Authentifizierung

Häufig verwendete API-Operationen

https://developers.freshdesk.com/api/#create_ticket

https://developers.freshdesk.com/api/#update_ticket

https://developer.freshsales.io/api/#create_lead

https://developer.freshsales.io/api/#update_lead

Zusätzliche Informationen

None

MongoDB

Endpunkt-URL des API-Zielaufrufs

https://data.mongodb-api.com/app/*App-ID*/endpoint/

Unterstützte Autorisierungstypen

API-Schlüssel

E-Mail/Passwort

Benutzerdefinierte JWT-Authentifizierung

Zusätzliche Autorisierungsparameter erforderlich

None

MongoDB-Dokumentation

Atlas-Daten-API

Endpunkte

Benutzerdefinierte HTTPS-Endpunkte

Authentifizierung

Häufig verwendete API-Operationen

None

Zusätzliche Informationen

None

New Relic

Endpunkt-URL des API-Zielaufrufs

Weitere Informationen finden Sie unter Unsere Rechenzentren in der EU- und USA-Region.

Ereignisse

USA– https://insights-collector.newrelic.com/v1/accounts/*YOUR_NEW_RELIC_ACCOUNT_ID*/events

EU– https://insights-collector.eu01.nr-data.net/v1/accounts/*YOUR_NEW_RELIC_ACCOUNT_ID*/events

Metriken

USA– https://metric-api.newrelic.com/metric/v1

EU– https://metric-api.eu.newrelic.com/metric/v1

Protokolle

USA– https://log-api.newrelic.com/log/v1

EU– https://log-api.eu.newrelic.com/log/v1

Ablaufverfolgungen

USA– https://trace-api.newrelic.com/trace/v1

EU– https://trace-api.eu.newrelic.com/trace/v1

Unterstützte Autorisierungstypen

API-Schlüssel

New Relic-Dokumentation

Metrische API

Ereignis-API

Protokoll-API

- Ablaufverfolgungs-API

Häufig verwendete API-Operationen

- Metrische API
- Ereignis-API
- Protokoll-API
- Ablaufverfolgungs-API

Zusätzliche Informationen

- Metrische API-Limits
- Ereignis-API-Limits
- Protokoll-API-Limits
- Ablaufverfolgungs-API-Limits

Operata

Endpunkt-URL des API-Zielaufrufs:

- https://api.operata.io/v2/aws/events/contact-record

Unterstützte Autorisierungstypen:

- Basic

Zusätzliche Autorisierungsparameter erforderlich:

- None

Operata-Dokumentation:

- Wie erstelle, betrachte, ändere und widerrufe ich API-Token?
- AWS Operata-Integration mit Amazon EventBridge Scheduler Pipes

Häufig verwendete API-Operationen:

- `POST https://api.operata.io/v2/aws/events/contact-record`

Zusätzliche Informationen:

- Der `username` ist die Operata-Gruppen-ID und das Passwort ist Ihr API-Token.

Salesforce

Endpunkt-URL des API-Zielaufrufs

Objekt — myDomainName https://.my.salesforce.com/services/data/ versionNumber /sobjects*//* SobjectEndpoint*

Benutzerdefinierte Plattformereignisse — *https://myDomainName.my.salesforce.com/services/data/versionNumber /sobjects/ /* customPlatformEndpoint*

Eine vollständige Liste der Endpunkte finden Sie in der Salesforce-API-Referenz.

Unterstützte Autorisierungstypen

OAuth-Client-Anmeldeinformationen

OAUTH-Token werden aktualisiert, wenn eine 401- oder 407-Antwort zurückgegeben wird.

Zusätzliche Autorisierungsparameter erforderlich

Client-ID und Client-Secret für die Salesforce-verbundene App

Einer der folgenden Autorisierungsendpunkte:

- Produktion — https *MyDomainName*://.my.salesforce.com. /services/oauth2/token
- Sandbox ohne erweiterte Domänen — https://-- .my. salesforce.com/services /oauth2/token *MyDomainName SandboxName*
- Sandbox mit erweiterten Domänen — https://-- .sandbox.my.salesforce.com/services/oauth2/token ***MyDomainName SandboxName***

Das folgende Schlüssel/Wert-Paar:

Key (Schlüssel)	Value (Wert)
Gewährungsart	client_credentials

Salesforce-Dokumentation

REST-API-Entwicklerhandbuch

Häufig verwendete API-Operationen

Arbeiten mit Objektmetadaten

Arbeiten mit Datensätzen

Zusätzliche Informationen

Ein Tutorial, in dem erklärt wird, wie Sie mit der EventBridge Konsole eine Verbindung zu Salesforce einem API-Ziel und eine Regel zum Weiterleiten von Informationen herstellenSalesforce, finden Sie unter. ???

Slack

Endpunkt-URL des API-Zielaufrufs

Eine Liste von Endpunkten und anderen Ressourcen finden Sie unter Verwenden der Slack-Web-API.

Unterstützte Autorisierungstypen

OAuth 2.0

OAUTH-Token werden aktualisiert, wenn eine 401- oder 407-Antwort zurückgegeben wird.

Wenn Sie eine Slack-Anwendung erstellen und sie in Ihrem Workspace installieren, wird in Ihrem Namen ein OAuth-Bearer-Token erstellt, das für die Authentifizierung von Aufrufen Ihrer API-Zielverbindung verwendet wird.

Zusätzliche Autorisierungsparameter erforderlich

Nicht zutreffend

Slack-Dokumentation

Grundlegende App-Einrichtung

Installation mit OAuth

Abrufen von Nachrichten

Senden von Nachrichten

Senden von Nachrichten mit eingehenden Webhooks

Häufig verwendete API-Operationen

https://slack.com/api/chat.postMessage

Zusätzliche Informationen

Bei der Konfiguration Ihrer EventBridge Regel sollten Sie zwei Konfigurationen hervorheben:

- Fügen Sie einen Header-Parameter hinzu, der den Inhaltstyp als „application/json;charset=utf-8“ definiert.
- Verwenden Sie einen Eingabe-Transformator, um das Eingabeereignis der erwarteten Ausgabe für die Slack-API zuzuordnen. Stellen Sie also sicher, dass die an die Slack-API gesendete Nutzlast „Kanal“- und „Text“-Schlüssel-Wert-Paare enthält.

Shopify

Endpunkt-URL des API-Zielaufrufs

Eine Liste der Endpunkte und anderer Ressourcen und Methoden finden Sie unter Endpunkte und Anforderungen.

Unterstützte Autorisierungstypen

OAuth, API Key

> **Note**
>
> OAUTH-Token werden aktualisiert, wenn eine 401- oder 407-Antwort zurückgegeben wird.

Zusätzliche Autorisierungsparameter erforderlich

Nicht zutreffend

Shopify-Dokumentation

Übersicht über Authentifizierung und Autorisierung

Häufig verwendete API-Operationen

POST – /admin/api/2022-01/products.json

GET – admin/api/2022-01/products/{product_id}.json

PUT – admin/api/2022-01/products/{product_id}.json

DELETE – admin/api/2022-01/products/{product_id}.json

Zusätzliche Informationen

Erstellen einer App

Amazon EventBridge Webhook-Lieferung

Zugriffstoken für benutzerdefinierte Apps im Shopify-Admin

Produkt

Shopify-Admin-API

Splunk

Endpunkt-URL des API-Zielaufrufs

https://*SPLUNK_HEC_ENDPOINT*:*optional_port*/services/collector/raw

Unterstützte Autorisierungstypen

Basic, API Key

Zusätzliche Autorisierungsparameter erforderlich

None

Splunk-Dokumentation

Für beide Autorisierungstypen benötigen Sie eine HEC-Token-ID. Weitere Informationen finden Sie unter Einrichten und Verwenden des HTTP Event Collectors im Splunk Web.

Häufig verwendete API-Operationen

POST https://*SPLUNK_HEC_ENDPOINT*:*optional_port*/services/collector/raw

Zusätzliche Informationen

API-Schlüssel — Bei der Konfiguration des Endpunkts für EventBridge lautet der API-Schlüsselname „Authorization“ und der Wert ist die Splunk HEC-Token-ID.

Basic (Benutzername/Passwort) — Bei der Konfiguration des Endpunkts für EventBridge lautet der Benutzername „Splunk“ und das Passwort ist die Splunk HEC-Token-ID.

Sumo Logic

Endpunkt-URL des API-Zielaufrufs

Die Endpunkt-URLs für HTTP-Protokoll- und Metrikquellen sind für jeden Benutzer unterschiedlich. Weitere Informationen finden Sie unter HTTP-Protokoll- und Metrikquellen.

Unterstützte Autorisierungstypen

Sumo Logic erfordert keine Authentifizierung für ihre HTTP-Quellen, da ein eindeutiger Schlüssel in die URL integriert wird. Aus diesem Grund sollten Sie sicherstellen, dass diese URL geheim gehalten wird.

Wenn Sie das EventBridge API-Ziel konfigurieren, ist ein Autorisierungstyp erforderlich. Zum Erfüllen dieser Anforderung wählen Sie API Key aus und geben ihm den Schlüsselnamen „dummy-key“ und den Schlüsselwert „dummy-value“.

Zusätzliche Autorisierungsparameter erforderlich

Nicht zutreffend

Sumo Logic-Dokumentation

Sumo Logichat bereits gehostete Quellen zur Erfassung von Protokollen und Messdaten von vielen AWS Diensten erstellt. Sie können die Informationen auf ihrer Website verwenden, um mit diesen Quellen zu arbeiten. Weitere Informationen finden Sie unter Amazon Web Services.

Wenn Sie benutzerdefinierte Ereignisse aus einer Anwendung generieren und diese entweder Sumo Logic als Protokolle oder als Metriken an diese senden möchten, verwenden Sie EventBridge API-Ziele und Sumo Logic HTTP-Protokoll- und Metrikquellendpunkte.

- Informationen zur Registrierung und Erstellung einer kostenlosen Sumo Logic-Instance finden Sie unter Starten Sie noch heute Ihre kostenlose Testversion.
- Weitere Informationen zur Verwendung von Sumo Logic finden Sie unter HTTP-Protokoll- und Metrikquelle.

Häufig verwendete API-Operationen

POST https://endpoint4.collection.us2.sumologic.com/receiver/v1/http/*UNIQUE_ID_PER_COLLECTOR*

Zusätzliche Informationen

None

TriggerMesh

Endpunkt-URL des API-Zielaufrufs

Verwenden Sie die Informationen im Thema Ereignisquelle für HTTP, um die Endpunkt-URL zu formulieren. Eine Endpunkt-URL enthält den Namen der Ereignisquelle und den Benutzer-Namespace im folgenden Format:

https://*source-name*.*user-namespace*.cloud.triggermesh.io

Nehmen Sie die Basic-Autorisierungsparameter in die Anforderung an den Endpunkt auf.

Unterstützte Autorisierungstypen

Basic

Zusätzliche Autorisierungsparameter erforderlich

None

TriggerMesh-Dokumentation

Ereignisquelle für HTTP

Häufig verwendete API-Operationen

Nicht zutreffend

Zusätzliche Informationen

None

Zendesk

Endpunkt-URL des API-Zielaufrufs

https://developer.zendesk.com/rest_api/docs/support/tickets

Unterstützte Autorisierungstypen

Basic, API Key

Zusätzliche Autorisierungsparameter erforderlich

None

Zendesk-Dokumentation

Sicherheit und Authentifizierung

Häufig verwendete API-Operationen

POST https://*your_Zendesk_subdomain*/api/v2/tickets

Zusätzliche Informationen

API-Anfragen werden auf EventBridge Ihre Zendesk-API-Limits angerechnet. Informationen zu den Zendesk-Limits für Ihren Plan finden Sie unter Nutzungslimits.

Wenn Sie Ihr Konto und Ihre Daten besser schützen möchten, empfehlen wir die Verwendung eines API-Schlüssels anstelle der grundlegenden Authentifizierung mit Anmeldeinformationen.

EventBridge Amazon-Ziele für Amazon API Gateway

Amazon API Gateway können Sie benutzen, um APIs zu erstellen, veröffentlichen, warten und überwachen. Amazon EventBridge unterstützt das Senden von Ereignissen an einen API-Gateway-Endpunkt. Wenn Sie einen API-Gateway-Endpunkt als Ziel angeben, wird jedes an das Ziel gesendete Ereignis einer an den Endpunkt gesendeten Anforderung zugeordnet.

> **Important**
>
> EventBridge unterstützt die Verwendung von API Gateway Edge-optimierten und regionalen Endpunkten als Ziele. Private Endpunkte werden derzeit nicht unterstützt. Weitere Informationen zu Endpunkten finden Sie unter https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-api-endpoint-types.html.

Sie können ein API-Gateway-Ziel für die folgenden Anwendungsfälle verwenden:

- Um eine vom Kunden angegebene API aufzurufen, die in API Gateway gehostet wird und auf AWS Ereignissen von Drittanbietern basiert.
- Wenn Sie einen Endpunkt regelmäßig nach einem Zeitplan aufrufen möchten.

Die EventBridge JSON-Ereignisinformationen werden als Hauptteil der HTTP-Anfrage an Ihren Endpunkt gesendet. Sie können die anderen Anforderungsattribute im `HttpParameters`-Feld des Ziels wie folgt angeben:

- `PathParameterValues` listet beispielsweise die Werte auf, die sequentiell beliebigen Pfadvariablen in Ihrem Endpunkt-ARN entsprechen, z. B. `"arn:aws:execute-api:us-east-1:112233445566:myapi/dev/POST/pets/*/*"`.
- `QueryStringParameters`stellt die Parameter der Abfragezeichenfolge dar, die an den aufgerufenen Endpunkt EventBridge angehängt werden.
- `HeaderParameters` definiert HTTP-Header, die der Anforderung hinzugefügt werden sollen.

> **Note**
>
> Aus Sicherheitsgründen sind die folgenden HTTP-Header-Schlüssel nicht zulässig:
>
> - Alles, was mit dem Präfix `X-Amz` oder `X-Amzn` versehen ist
> - `Authorization`
> - `Connection`
> - `Content-Encoding`
> - `Content-Length`
> - `Host`
> - `Max-Forwards`
> - `TE`
> - `Transfer-Encoding`
> - `Trailer`
> - `Upgrade`
> - `Via`
> - `WWW-Authenticate`
> - `X-Forwarded-For`

Dynamische Parameter

Wenn Sie ein API-Gateway-Ziel aufrufen, können Sie dynamisch Daten zu Ereignissen hinzufügen, die an das Ziel gesendet werden. Weitere Informationen finden Sie unter the section called “Zielparameter”.

Aufrufwiederholungen

EventBridge Wiederholt, wie bei allen Zielen, einige fehlgeschlagene Aufrufe. Für API Gateway werden Antworten, EventBridge die mit einem 5xx- oder 429-HTTP-Statuscode gesendet wurden, bis zu 24 Stunden lang wiederholt, wobei exponentielles Back-Off und Jitter auftreten. Veröffentlicht danach eine EventBridge `FailedInvocations` Metrik in Amazon CloudWatch. EventBridge wiederholt keine anderen 4xx-HTTP-Fehler.

Zeitüberschreitung

EventBridge Regel API Gateway Gateway-Anfragen müssen ein maximales Client-Ausführungstimeout von 5 Sekunden haben. Wenn die Antwort von API Gateway länger als 5 Sekunden dauert, wird das EventBridge Zeitlimit für die Anfrage überschritten und es wird erneut versucht.

EventBridge Pipes API Gateway Gateway-Anfragen haben ein maximales Timeout von 29 Sekunden, das API-Gateway-Maximum.

AWS AppSync Ziele für Amazon EventBridge

AWS AppSync ermöglicht es Entwicklern, ihre Anwendungen und Dienste mit sicheren, serverlosen und leistungsstarken GraphQL- und Pub/Sub-APIs mit Daten und Ereignissen zu verbinden.
Mit AWS AppSync können Sie Datenaktualisierungen in Echtzeit für Ihre Anwendungen mit GraphQL-Mutationen veröffentlichen. EventBridge unterstützt das Aufrufen einer gültigen GraphQL-Mutationsoperation für übereinstimmende Ereignisse. Wenn Sie eine AWS AppSync API-Mutation als Ziel angeben, AWS AppSync verarbeitet das Ereignis über einen Mutationsvorgang, der dann Abonnements auslösen kann, die mit der Mutation verknüpft sind.

> **Note**
>
> EventBridge unterstützt AWS AppSync öffentliche GraphQL-APIs. EventBridge unterstützt derzeit keine AWS AppSync privaten APIs.

Sie können ein AWS AppSync GraphQL-API-Ziel für die folgenden Anwendungsfälle verwenden:

- Um Ereignisdaten in Ihre konfigurierten Datenquellen zu übertragen, zu transformieren und zu speichern.
- Um Benachrichtigungen in Echtzeit an verbundene Anwendungs-Clients zu senden.

> **Note**
>
> AWS AppSync Ziele unterstützen nur das Aufrufen von AWS AppSync GraphQL-APIs mit dem `AWS_IAM`Autorisierungstyp.

Weitere Informationen zu AWS AppSync GraphQL-APIs finden Sie unter GraphQL und AWS AppSync Architektur im AWS AppSync Developer Guide.

Um mithilfe der Konsole ein AWS AppSync Ziel für eine EventBridge Regel anzugeben

1. Erstellen oder bearbeiten Sie die Regel.
2. Geben Sie unter Ziel das Ziel an, indem Sie AWS -Service und dann AWS AppSync auswählen.
3. Geben Sie die zu analysierende und auszuführende Mutationsoperation sowie den Auswahlsatz an.

   - Wählen Sie die AWS AppSync API und dann die aufzurufende GraphQL-API-Mutation aus.
   - Wählen Sie unter Parameter und Auswahlsatz konfigurieren aus, ob Sie einen Auswahlsatz mithilfe der Schlüssel-Wert-Zuordnung oder eines Eingabetransformators erstellen möchten.

     Key-value mapping

     So verwenden Sie die Schlüssel-Wert-Zuordnung zum Erstellen Ihres Auswahlsatzes:

     - Geben Sie Variablen für die API-Parameter an. Jede Variable kann entweder ein statischer Wert oder ein dynamischer JSON-Pfadausdruck für die Ereignisnutzlast sein.
     - Wählen Sie unter Auswahlsatz die Variablen aus, die Sie in die Antwort aufnehmen möchten.

     Input transformer

     So verwenden Sie einen Eingabetransformator zum Erstellen Ihres Auswahlsatzes:

     - Geben Sie einen Eingabepfad an, der die zu verwendenden Variablen definiert.
     - Geben Sie eine Eingabevorlage an, um die Informationen zu definieren und zu formatieren, die an das Ziel übergeben werden sollen.

     Weitere Informationen finden Sie unter ???.
4. Wählen Sie für Ausführungsrolle aus, ob Sie eine neue Rolle erstellen oder eine vorhandene Rolle verwenden möchten.

5. Schließen Sie die Erstellung oder Bearbeitung der Regel ab.

Beispiel: AWS AppSync Ziele für Amazon EventBridge

Im folgenden Beispiel erfahren Sie, wie Sie ein AWS AppSync Ziel für eine EventBridge Regel angeben, einschließlich der Definition einer Eingabetransformation zur Formatierung von Ereignissen für die Übermittlung.

Angenommen, Sie haben eine AWS AppSync GraphQL-API`Ec2EventAPI`, definiert durch das folgende Schema:

```
type Event {
    id: ID!
    statusCode: String
    instanceId: String
}

type Mutation {
    pushEvent(id: ID!, statusCode: String!, instanceId: String): Event
}

type Query {
    listEvents: [Event]
}

type Subscription {
    subscribeToEvent(id: ID, statusCode: String, instanceId: String): Event
        @aws_subscribe(mutations: ["pushEvent"])
}
```

Anwendungs-Clients, die diese API verwenden, können das Abonnement `subscribeToEvent` abonnieren, das durch die `pushEvent`-Mutation ausgelöst wird.

Sie können eine EventBridge Regel mit einem Ziel erstellen, das Ereignisse über die `pushEvent` Mutation an die AppSync API sendet. Wenn die Mutation aufgerufen wird, erhält jeder abonnierte Client das Ereignis.

Um diese API als Ziel für eine EventBridge Regel anzugeben, gehen Sie wie folgt vor:

1. Legen Sie als Amazon-Ressourcenname (ARN) des Regelziels den GraphQL-Endpunkt-ARN der `Ec2EventAPI`-API fest.

2. Geben Sie die Mutations-GraphQL-Operation als Zielparameter an:

```
mutation CreatePushEvent($id: ID!, $statusCode: String, $instanceId: String) {
  pushEvent(id: $input, statusCode: $statusCode, instanceId: $instanceId) {
    id
    statusCode
    instanceId
  }
}
```

   Ihr Mutationsauswahlsatz muss alle Felder enthalten, die Sie in Ihrem GraphQL-Abonnement abonnieren möchten.

3. Konfigurieren Sie einen Eingabetransformator, um anzugeben, wie Daten aus übereinstimmenden Ereignissen in Ihrer Operation verwendet werden.

   Angenommen, Sie haben das Beispielereignis "`EC2 Instance Launch Successful`" ausgewählt:

```
{
  "version": "0",
  "id": "3e3c153a-8339-4e30-8c35-687ebef853fe",
  "detail-type": "EC2 Instance Launch Successful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2015-11-11T21:31:47Z",
  "region": "us-east-1",
  "resources": ["arn:aws:autoscaling:us-
east-1:123456789012:autoScalingGroup:eb56d16b-bbf0-401d-b893-
d5978ed4a025:autoScalingGroupName/sampleLuanchSucASG", "arn:aws:ec2:us-
east-1:123456789012:instance/i-b188560f"],
  "detail": {
    "StatusCode": "InProgress",
    "AutoScalingGroupName": "sampleLuanchSucASG",
    "ActivityId": "9cabb81f-42de-417d-8aa7-ce16bf026590",
    "Details": {
      "Availability Zone": "us-east-1b",
      "Subnet ID": "subnet-95bfcebe"
    },
    "RequestId": "9cabb81f-42de-417d-8aa7-ce16bf026590",
    "EndTime": "2015-11-11T21:31:47.208Z",
    "EC2InstanceId": "i-b188560f",
    "StartTime": "2015-11-11T21:31:13.671Z",
```

```
    "Cause": "At 2015-11-11T21:31:10Z a user request created an AutoScalingGroup
 changing the desired capacity from 0 to 1.  At 2015-11-11T21:31:11Z an instance was
 started in response to a difference between desired and actual capacity, increasing
 the capacity from 0 to 1."
  }
}
```

Sie können die folgenden Variablen für die Verwendung in Ihrer Vorlage definieren, indem Sie den Eingabepfad des Zieleingabetransformators verwenden:

```
{
  "id": "$.id",
  "statusCode": "$.detail.StatusCode",
  "EC2InstanceId": "$.detail.EC2InstanceId"
}
```

Verfassen Sie die Vorlage für den Eingangstransformator, EventBridge um die Variablen zu definieren, die an die AWS AppSync Mutationsoperation übergeben werden. Die Vorlage muss als JSON ausgewertet werden. Ausgehend von unserem Eingabepfad können Sie die folgende Vorlage erstellen:

```
{
  "id": <id>,
  "statusCode": <statusCode>,
  "instanceId": <EC2InstanceId>
}
```

Verbindungen für HTTP-Endpunktziele

Eine Verbindung definiert die Autorisierungsmethode und die Anmeldeinformationen EventBridge , die für die Verbindung mit einem bestimmten HTTP-Endpunkt verwendet werden sollen. Wenn Sie die Autorisierungseinstellungen konfigurieren und eine Verbindung herstellen, wird ein Geheimnis erstellt, in AWS Secrets Manager dem die Autorisierungsinformationen sicher gespeichert werden. Sie können auch zusätzliche Parameter hinzufügen, die in die Verbindung aufgenommen werden sollen, je nach Ihrem HTTP-Endpunktziel.

Verwenden Sie Verbindungen mit:

- API-Ziele

Wenn Sie ein API-Ziel erstellen, geben Sie eine dafür zu verwendende Verbindung an. Sie können eine bestehende Verbindung aus Ihrem Konto auswählen oder eine Verbindung erstellen, wenn Sie ein API-Ziel erstellen.

Autorisierungsmethoden für Verbindungen

EventBridge Verbindungen unterstützen die folgenden Autorisierungsmethoden:

- Basic
- API-Schlüssel

  EventBridge Füllt für die Basic- und API-Schlüssel-Autorisierung die erforderlichen Autorisierungsheader für Sie aus.
- OAuth

  Tauscht bei der OAuth-Autorisierung EventBridge auch Ihre Client-ID und Ihr Secret gegen ein Zugriffstoken aus und verwaltet es dann sicher.

  OAUTH-Token werden aktualisiert, wenn eine 401- oder 407-Antwort zurückgegeben wird.

Wenn Sie eine Verbindung erstellen, können Sie auch die Header-, Text- und Abfrageparameter angeben, die für die Autorisierung mit einem Endpunkt erforderlich sind. Sie können dieselbe Verbindung für mehr als einen HTTP-Endpunkt verwenden, wenn die Autorisierung für den Endpunkt dieselbe ist.

Wenn Sie eine Verbindung herstellen und Autorisierungsparameter hinzufügen, EventBridge erstellt ein Geheimnis in AWS Secrets Manager. Die Kosten für die Speicherung und den Zugriff auf das Secrets-Manager-Secret sind in der Gebühr für die Verwendung eines API-Ziels enthalten. Weitere Informationen zu bewährten Methoden für die Verwendung von Geheimnissen mit API-Zielen finden Sie AWS::Events::ApiDestinationim CloudFormation Benutzerhandbuch.

> **Note**
>
> Wenn Sie eine Verbindung erfolgreich erstellen oder aktualisieren möchten, müssen Sie ein Konto verwenden, das über die Berechtigung zur Verwendung von Secrets Manager verfügt. Die erforderliche Berechtigung ist in der AmazonEventBridgeFullAccess Richtlinie enthalten.

Dieselbe Berechtigung wird der serviceverknüpften Rolle gewährt, die in Ihrem Konto für die Verbindung erstellt wurde.

Verbindungen für HTTP-Endpunktziele erstellen

Um mithilfe der Konsole eine Verbindung zur Verwendung mit HTTP-Endpunkten herzustellen EventBridge

1. Melden Sie sich AWS mit einem Konto an, das über Berechtigungen zum Verwalten EventBridge und Öffnen der EventBridge Konsole verfügt.
2. Wählen Sie im linken Navigationsbereich API-Ziele aus.
3. Scrollen Sie nach unten zur Tabelle API-Ziele und wählen Sie dann die Registerkarte Verbindungen aus.
4. Wählen Sie Create Connection (Verbindung erstellen) aus.
5. Geben Sie auf der Seite Verbindung erstellen einen Verbindungsnamen für die Verbindung ein.
6. Geben Sie eine Beschreibung für die Verbindung ein.
7. Wählen Sie für Autorisierungstyp den Autorisierungstyp aus, der verwendet werden soll, um Verbindungen zu dem HTTP-Endpunkt zu autorisieren, der für das API-Ziel angegeben ist, das diese Verbindung verwendet. Führen Sie eine der folgenden Aktionen aus:
   - Wählen Sie Basic (Benutzername/Passwort) aus und geben Sie dann den Benutzernamen und das Passwort ein, die für die Autorisierung mit dem HTTP-Endpunkt verwendet werden sollen.
   - Wählen Sie OAuth-Client-Anmeldeinformationen aus und geben Sie dann den Autorisierungsendpunkt, die HTTP-Methode, die Client-ID und das Client-Secret ein, die für die Autorisierung mit dem Endpunkt verwendet werden sollen.

     Fügen Sie unter OAuth-Http-Parameter alle zusätzlichen Parameter hinzu, die Sie für die Autorisierung mit dem Autorisierungsendpunkt einbeziehen möchten. Wählen Sie einen Parameter aus der Dropdown-Liste aus und geben Sie dann einen Schlüssel und einen Wert ein. Wenn Sie einen zusätzlichen Parameter hinzufügen möchten, wählen Sie Parameter hinzufügen aus.

     Fügen Sie unter Aufruf-Http-Parameter alle zusätzlichen Parameter hinzu, die Sie in die Autorisierungsanforderung einbeziehen möchten. Wenn Sie einen Parameter hinzufügen möchten, wählen Sie einen Parameter aus der Dropdown-Liste aus und geben Sie dann

einen Schlüssel und einen Wert ein. Wenn Sie einen zusätzlichen Parameter hinzufügen möchten, wählen Sie Parameter hinzufügen aus.

- Wählen Sie API-Schlüssel aus und geben Sie dann den API-Schlüsselnamen und den zugehörigen Wert ein, der für die API-Key-Autorisierung verwendet werden soll.

  Fügen Sie unter Aufruf-Http-Parameter alle zusätzlichen Parameter hinzu, die Sie in die Autorisierungsanforderung einbeziehen möchten. Wenn Sie einen Parameter hinzufügen möchten, wählen Sie einen Parameter aus der Dropdown-Liste aus und geben Sie dann einen Schlüssel und einen Wert ein. Wenn Sie einen zusätzlichen Parameter hinzufügen möchten, wählen Sie Parameter hinzufügen aus.

8. Wählen Sie Erstellen.

Verbindungen mit der EventBridge Konsole bearbeiten

Sie können bestehende Verbindungen bearbeiten.

Um eine Verbindung mit der EventBridge Konsole zu bearbeiten

1. Melden Sie sich AWS mit einem Konto an, das berechtigt ist, die EventBridge Konsole zu verwalten EventBridge und zu öffnen.
2. Wählen Sie im linken Navigationsbereich API-Ziele aus.
3. Scrollen Sie nach unten zur Tabelle API-Ziele und wählen Sie dann die Registerkarte Verbindungen aus.
4. Wählen Sie in der Tabelle Verbindungen die Verbindung aus, die Sie bearbeiten möchten.
5. Klicken Sie auf der Seite Verbindungsdetails auf Bearbeiten.
6. Aktualisieren Sie die Werte für die Verbindung und wählen Sie dann Aktualisieren aus.

Autorisierung von Verbindungen über die Konsole aufheben EventBridge

Wenn Sie die Autorisierung einer Verbindung aufheben, werden alle Autorisierungsparameter entfernt. Durch das Entfernen von Autorisierungsparametern wird das Secret aus der Verbindung entfernt, sodass Sie es wiederverwenden können, ohne eine neue Verbindung erstellen zu müssen.

> **Note**
>
> Sie müssen alle HTTP-Endpunkte, die die deautorisierte Verbindung verwenden, so aktualisieren, dass sie eine andere Verbindung verwenden, um Anfragen erfolgreich an den HTTP-Endpunkt zu senden.

So heben Sie die Autorisierung einer Verbindung auf

1. Melden Sie sich AWS mit einem Konto an, das über Berechtigungen zum Verwalten EventBridge und Öffnen der EventBridge Konsole verfügt.
2. Wählen Sie im linken Navigationsbereich API-Ziele aus.
3. Scrollen Sie nach unten zur Tabelle API-Ziele und wählen Sie dann die Registerkarte Verbindungen aus.
4. Wählen Sie in der Tabelle Verbindungen die Verbindung aus.
5. Klicken Sie auf der Seite Verbindungsdetails auf Autorisierung aufheben.
6. Geben Sie im Dialogfeld Autorisierung der Verbindung aufheben? den Namen der Verbindung ein und wählen Sie dann Autorisierung aufheben aus.

Der Status der Verbindung ändert sich in Aufheben der Autorisierung, bis der Vorgang abgeschlossen ist. Dann ändert sich der Status in Autorisierung wurde aufgehoben. Jetzt können Sie die Verbindung bearbeiten, um neue Autorisierungsparameter hinzuzufügen.

EventBridge Amazon-Ereignisse zwischen AWS Konten senden und empfangen

Sie können in AWS Konten konfigurieren EventBridge , dass Ereignisse zwischen Eventbussen gesendet und empfangen werden. Wenn Sie das Senden oder Empfangen von Ereignissen zwischen Konten konfigurieren EventBridge , können Sie angeben, welche AWS Konten Ereignisse an den Event-Bus in Ihrem Konto senden oder von diesem empfangen können. Sie können auch Ereignisse aufgrund bestimmter Regeln, die mit dem Event Bus verknüpft sind, oder Ereignisse aus bestimmten Quellen zulassen oder verweigern. Weitere Informationen finden Sie unter Vereinfachung des kontoübergreifenden Zugriffs mit Amazon-Ressourcenrichtlinien EventBridge

> **Note**
>
> Wenn Sie diese Option verwenden AWS Organizations, können Sie eine Organisation angeben und Zugriff auf alle Konten in dieser Organisation gewähren. Darüber hinaus müssen dem sendenden Event Bus IAM-Rollen zugewiesen sein, wenn Ereignisse an ein anderes Konto gesendet werden. Weitere Informationen finden Sie unter Was ist AWS Organizations? im AWS Organizations -Benutzerhandbuch.

> **Note**
>
> Wenn Sie einen Incident-Manager-Antwortplan als Ziel verwenden, sind alle Antwortpläne, die mit Ihrem Konto geteilt wurden, standardmäßig verfügbar.

Sie können Ereignisse zwischen Eventbussen auf AWS Konten innerhalb derselben Region in allen Regionen und zwischen Konten in verschiedenen Regionen senden und empfangen, sofern es sich bei der Zielregion um eine unterstützte regionsübergreifende Zielregion handelt.

Die Konfiguration für das Senden von Ereignissen EventBridge an oder das Empfangen von Ereignissen an einen Event-Bus in einem anderen Konto umfasst die folgenden Schritte:

- Bearbeiten Sie auf dem Empfängerkonto die Berechtigungen für einen Event-Bus, sodass bestimmte AWS Konten, eine Organisation oder alle AWS Konten Ereignisse an das Empfängerkonto senden können.
- Richten Sie im Sender-Konto eine oder mehrere Regeln ein, die den Standardereignisbus des Empfängers als Ziel besitzen.

  Wenn das Absenderkonto die Berechtigungen zum Senden von Ereignissen von einer AWS Organisation erbt, muss das Absenderkonto auch über eine IAM-Rolle mit Richtlinien verfügen, die es ihm ermöglichen, Ereignisse an das Empfängerkonto zu senden. Wenn Sie die Regel verwenden, die AWS Management Console auf den Event-Bus im Empfängerkonto abzielt, wird die Rolle automatisch erstellt. Wenn Sie die verwenden AWS CLI, müssen Sie die Rolle manuell erstellen.
- Richten Sie auf dem Empfänger-Konto eine oder mehrere Regeln ein, die mit Ereignissen übereinstimmen, die vom Senderkonto kommen.

Ereignisse, die von einem Konto zu einem anderen gesendet werden, werden dem sendenden Konto als benutzerdefinierte Ereignisse in Rechnung gestellt. Das empfangende Konto wird nicht belastet. Weitere Informationen finden Sie unter EventBridge Amazon-Preise.

Falls ein Empfänger-Konto eine Regel festlegt, dass Ereignisse, die von einem Sender-Konto empfangen wurden, zu einem dritten Konto gesendet werden, werden diese Ereignisse jedoch nicht an das dritte Konto weitergeleitet.

Wenn Sie drei Event-Busse in demselben Konto haben und für den ersten Event-Bus eine Regel einrichten, nach der Events vom zweiten Event-Bus an einen dritten Event-Bus weitergeleitet werden, werden diese Events nicht an den dritten Event-Bus gesendet.

Das folgende Video behandelt die Weiterleitung von Ereignissen zwischen Konten: Weiterleiten von Ereignissen an Busse in anderen AWS Konten

Erteilen Sie Berechtigungen, um Ereignisse von anderen AWS Konten zuzulassen

Wenn Sie Ereignisse aus anderen Konten oder Organisationen erhalten möchten, müssen Sie zuerst die Berechtigungen für den Standard-Event-Bus bearbeiten, in dem Sie Ereignisse empfangen möchten. Der Standard-Event-Bus akzeptiert Ereignisse von AWS Diensten, anderen autorisierten AWS Konten und `PutEvents` Aufrufen. Die Berechtigungen für einen Event Bus werden mithilfe einer ressourcenbasierten Richtlinie, die an den Event Bus angehängt ist, gewährt oder verweigert. In der Richtlinie können Sie anderen AWS Konten mithilfe der Konto-ID oder einer AWS Organisation mithilfe der Organisations-ID Berechtigungen gewähren. Weitere Informationen zu Event-Bus-Berechtigungen, einschließlich Beispielrichtlinien, finden Sie unter Berechtigungen für Amazon EventBridge-Event-Buses.

> **Note**
>
> EventBridge erfordert jetzt, dass alle neuen kontoübergreifenden Event-Bus-Ziele IAM-Rollen hinzufügen. Dies gilt nur für Event-Bus-Ziele, die nach dem 2. März 2023 erstellt wurden. Anwendungen, die vor diesem Datum ohne eine IAM-Rolle erstellt wurden, sind nicht betroffen. Wir empfehlen jedoch, IAM-Rollen hinzuzufügen, um Benutzern Zugriff auf Ressourcen in einem anderen Konto zu gewähren. Auf diese Weise wird sichergestellt, dass Organisationsgrenzen mithilfe von Service-Kontrollrichtlinien (SCPs) angewendet werden, um zu bestimmen, wer Ereignisse aus Konten in Ihrer Organisation senden und empfangen kann.

> **Important**
>
> Wenn Sie sich dafür entscheiden, Ereignisse von allen AWS Konten zu empfangen, achten Sie darauf, Regeln zu erstellen, die nur den Ereignissen entsprechen, die Sie von anderen erhalten. Um sicherere Regeln zu erstellen, achten Sie darauf, dass das Ereignismuster für jede Regel ein `Account`-Feld mit den Konto-IDs eines oder mehrerer Konten enthält, von denen Ereignisse empfangen werden sollen. Regeln mit einem Ereignismuster, das das Feld „Account-(Konto)“ enthält, stimmen nicht mit Ereignissen überein, die von Konten gesendet wurden, die nicht im Feld `Account` aufgelistet werden. Weitere Informationen finden Sie unter EventBridge Amazon-Veranstaltungen.

Regeln für Ereignisse zwischen AWS Konten

Wenn Ihr Konto so eingerichtet ist, dass Ereignisse von Eventbussen anderer AWS Konten empfangen werden, können Sie Regeln schreiben, die diesen Ereignissen entsprechen. Legen Sie das Ereignismuster der Regel so fest, dass es mit den Ereignissen, die Sie aus Event Buses im anderen Konto empfangen, übereinstimmt.

Wenn Sie `account` nicht im Ereignismuster einer Regel angeben, lösen alle neuen und vorhandenen Regeln Ihres Kontos, die mit Ereignissen, die Sie aus Event Buses in anderen Konten erhalten, übereinstimmen, auf Grundlage dieser Ereignisse aus. Wenn Sie Ereignisse aus Event Buses in einem anderen Konto empfangen und Sie möchten, dass eine Regel von diesem Ereignismuster nur dann ausgelöst wird, wenn es von Ihrem eigenen Konto generiert wurde, dann müssen Sie dem Ereignismuster der Regel `account` hinzufügen und Ihre eigene Konto-ID angeben.

Wenn du dein AWS Konto so einrichtest, dass Events von Eventbussen in allen AWS Konten akzeptiert werden, empfehlen wir dir dringend, alle EventBridge Regeln in deinem Konto `account` zu ergänzen. Dadurch wird verhindert, dass Regeln in Ihrem Konto bei Ereignissen von unbekannten AWS Konten ausgelöst werden. Wenn Sie in der Regel das Feld `account` festlegen, können Sie Konto-IDs von mehr als einem AWS -Konto in dem Feld angeben.

Damit eine Regel bei einem passenden Ereignis von beliebigen Eventbussen in einem AWS Konto ausgelöst wird, für das Sie Berechtigungen erteilt haben, geben Sie im `account` Feld der Regel kein Sternchen (*) an. Da "*" niemals im Feld `account` eines Ereignisses erscheint, würden in diesem Fall keine Ereignisse gefunden werden. Lassen Sie stattdessen das Feld `account` in der Regel weg.

Regeln erstellen, die Ereignisse zwischen AWS Konten senden

Das Angeben eines Event Bus in einem anderen Konto als Ziel ist Teil der Regelerstellung.

Um eine Regel zu erstellen, die Ereignisse mithilfe der Konsole an ein anderes AWS Konto sendet

1. Befolgen Sie die Schritte im Verfahren ???.
2. Wenn Sie im ???-Schritt aufgefordert werden, einen Zieltyp auszuwählen, gehen Sie wie folgt vor:
   a. Wählen Sie den EventBridge Event-Bus aus.
   b. Wählen Sie Event Bus in einem anderen Konto oder einer anderen Region aus.
   c. Geben Sie für Event Bus als Ziel den ARN des Event Bus ein, den Sie verwenden möchten.
3. Schließen Sie die Erstellung der Regel ab, indem Sie die Verfahrensschritte befolgen.

Senden und Empfangen von EventBridge Amazon-Events zwischen AWS Regionen

Sie können das Senden und Empfangen von Ereignissen zwischen AWS Regionen konfigurieren EventBridge . Sie können auch Ereignisse aus bestimmten Regionen aufgrund bestimmter Regeln, die mit dem Event Bus verknüpft sind, oder Ereignisse aus bestimmten Quellen zulassen oder verweigern. Weitere Informationen finden Sie unter Einführung von regionsübergreifendem Event-Routing mit Amazon EventBridge

Die folgenden Regionen werden als Zielregionen unterstützt:

- USA Ost (Nord-Virginia)
- USA Ost (Ohio)
- USA West (Nordkalifornien)
- USA West (Oregon)
- Africa (Cape Town)
- Asia Pacific (Hongkong)
- Asien-Pazifik (Tokio)
- Asien-Pazifik (Seoul)
- Asien-Pazifik (Osaka)

- Asien-Pazifik (Mumbai)
- Asien-Pazifik (Hyderabad)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Jakarta)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Melbourne)
- Kanada (Zentral)
- Kanada West (Calgary)
- Europa (Frankfurt)
- Europa (Spain)
- Europa (Zürich)
- Europa (Stockholm)
- Europa (Milan)
- Europa (Irland)
- Europe (London)
- Europa (Paris)
- Israel (Tel Aviv)
- Naher Osten (VAE)
- Naher Osten (Bahrain)
- Südamerika (São Paulo)

Das folgende Video behandelt das Routing von Ereignissen zwischen Regionen mithilfe von https://console.aws.amazon.com/events/, AWS CloudFormation, und AWS Serverless Application Model: Regionsübergreifendes Event-Routing

Regeln erstellen, die Ereignisse an eine andere AWS Region senden

Die Angabe eines Event-Busses in einer anderen AWS Region als Ziel ist Teil der Regelerstellung.

So erstellen Sie mithilfe der Konsole eine Regel, die Ereignisse an ein anderes AWS Konto sendet

1. Befolgen Sie die Schritte im Verfahren ???.

2. Wenn Sie im ???-Schritt aufgefordert werden, einen Zieltyp auszuwählen, gehen Sie wie folgt vor:

   a. Wählen Sie den EventBridge Event-Bus aus.

   b. Wählen Sie Event Bus in einem anderen Konto oder einer anderen Region aus.

   c. Geben Sie für Event Bus als Ziel den ARN des Event Bus ein, den Sie verwenden möchten.

3. Schließen Sie die Erstellung der Regel ab, indem Sie die Verfahrensschritte befolgen.

Senden und Empfangen von EventBridge Amazon-Events zwischen Eventbussen desselben Kontos und derselben Region

Sie können so konfigurieren EventBridge , dass Ereignisse zwischen Eventbussen desselben AWS Kontos und derselben Region gesendet und empfangen werden.

Wenn Sie das Senden oder Empfangen von Ereignissen zwischen Event-Bussen konfigurieren EventBridge , verwenden Sie IAM-Rollen auf dem Sender-Event-Bus, um dem Sender-Event-Bus die Berechtigung zu erteilen, Ereignisse an den Empfänger-Event-Bus zu senden. Sie verwenden ressourcenbasierte Richtlinien im Empfänger-Event-Bus, um dem Empfänger-Event-Bus die Berechtigung zu gewähren, Ereignisse aus dem Sender-Event-Bus zu empfangen. Sie können auch Ereignisse aus bestimmten Event Buses aufgrund bestimmter Regeln, die mit dem Event Bus verknüpft sind, oder Ereignisse aus bestimmten Quellen zulassen oder verweigern. Weitere Informationen zu Event-Bus-Berechtigungen, einschließlich Beispielrichtlinien, finden Sie unter Berechtigungen für Amazon EventBridge-Event-Buses.

Die Konfiguration für das Senden von Ereignissen EventBridge an oder den Empfang von Ereignissen zwischen Eventbussen in Ihrem Konto umfasst die folgenden Schritte:

- Wenn Sie eine vorhandene IAM-Rolle verwenden möchten, müssen Sie entweder dem Sender-Event-Bus Berechtigungen für den Empfänger-Event-Bus oder dem Empfänger-Event-Bus Berechtigungen für den Sender-Event-Bus gewähren.
- Richten Sie im Sender-Event-Bus eine oder mehrere Regeln ein, die den Empfänger-Event-Bus als Ziel haben, und erstellen Sie eine IAM-Rolle. Ein Beispiel für die Richtlinie, die der Rolle zugeordnet werden sollte, finden Sie unter ???.
- Bearbeiten Sie im Empfänger-Event-Bus die Berechtigungen, sodass Ereignisse aus dem anderen Event Bus weitergeleitet werden können.

- Richten Sie im Empfänger-Ereignis eine oder mehrere Regeln ein, die mit Ereignissen übereinstimmen, die vom Sender-Event-Bus kommen.

> **Note**
>
> EventBridge kann Ereignisse, die von einem Sender-Event-Bus empfangen wurden, nicht an einen dritten Event-Bus weiterleiten.

Ereignisse, die von einem Event Bus an einen anderen gesendet werden, werden als benutzerdefinierte Ereignisse berechnet. Weitere Informationen finden Sie unter Amazon EventBridge – Preise.

Regeln erstellen, die Ereignisse an einen anderen Event-Bus im selben AWS Konto und in derselben Region senden

Wenn Sie Ereignisse an einen anderen Event Bus senden möchten, erstellen Sie eine Regel mit einem Event Bus als Ziel. Die Angabe eines Event-Busses in demselben AWS Konto und derselben Region als Ziel ist Teil der Regelerstellung.

Um mithilfe der Konsole eine Regel zu erstellen, die Ereignisse an einen anderen Event-Bus im selben AWS Konto und in derselben Region sendet

1. Befolgen Sie die Schritte im Verfahren ???.
2. Wenn Sie im ???-Schritt aufgefordert werden, einen Zieltyp auszuwählen, gehen Sie wie folgt vor:
   a. Wählen Sie den EventBridge Event-Bus aus.
   b. Wählen Sie Eventbus für dasselbe AWS Konto und dieselbe Region aus.
   c. Wählen Sie für Event Bus als Ziel einen Event Bus aus der Dropdown-Liste aus.
3. Schließen Sie die Erstellung der Regel ab, indem Sie die Verfahrensschritte befolgen.

Transformation Amazon EventBridge Amazon-Eingaben

Sie können den Text eines Ereignisses anpassen, bevor EventBridge die Informationen an das Ziel einer Regel weitergegeben werden. Mithilfe des Eingabe-Transformators in der Konsole oder der API definieren Sie Variablen, die den JSON-Pfad verwenden, um auf Werte in der ursprünglichen Ereignisquelle zu verweisen. Das transformierte Ereignis wird anstelle des ursprünglichen Ereignisses an ein Ziel gesendet. Dynamische Pfadparameter müssen jedoch auf das ursprüngliche Ereignis verweisen, nicht auf das transformierte Ereignis. Sie können bis zu 100 Variablen definieren und dabei jeder einen Wert aus der Eingabe zuweisen. Anschließend können Sie diese Variablen in der Eingabevorlage als <*variable-name*> verwenden.

Ein Tutorial zur Verwendung des Eingabe-Transformators finden Sie unter ???.

> **Note**
>
> EventBridge unterstützt nicht die gesamte JSON-Pfad-Syntax und wertet sie zur Laufzeit aus. Die unterstützte Syntax umfasst:
>
> - Punktnotation (zum Beispiel `$.detail`)
> - Bindestriche
> - Unterstriche
> - Alphanumerische Zeichen
> - Array-Indizes
> - Platzhalter (*)

In diesem Thema:

Vordefinierte Variablen

Es gibt vordefinierte Variablen, die Sie verwenden können, ohne einen JSON-Pfad zu definieren. Diese Variablen sind reserviert und Sie können keine Variablen mit diesen Namen erstellen:

- `aws.events.rule-arn`— Der Amazon-Ressourcenname (ARN) der EventBridge Regel.
- `aws.events.rule-name`— Der Name der EventBridge Regel.
- `aws.events.event.ingestion-time`— Die Uhrzeit, zu der die Veranstaltung bei eingegangen ist EventBridge. Dies ist ein ISO-8601-Zeitstempel. Diese Variable wird von generiert EventBridge und kann nicht überschrieben werden.
- `aws.events.event` – Die ursprüngliche Ereignisnutzlast als JSON (ohne das `detail`-Feld) Kann nur als Wert für ein JSON-Feld verwendet werden, da dessen Inhalt nicht durch Escape-Zeichen geschützt ist
- `aws.events.event.json` – Die gesamte ursprüngliche Ereignisnutzlast als JSON (mit dem `detail`-Feld) Kann nur als Wert für ein JSON-Feld verwendet werden, da dessen Inhalt nicht durch Escape-Zeichen geschützt ist

Beispiele für die Eingabetransformation

Im Folgenden sehen Sie ein Amazon-EC2-Beispielereignis.

```
{
  "version": "0",
  "id": "7bf73129-1428-4cd3-a780-95db273d1602",
  "detail-type": "EC2 Instance State-change Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2015-11-11T21:29:54Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2:us-east-1:123456789012:instance/i-abcd1111"
  ],
  "detail": {
    "instance-id": "i-0123456789",
    "state": "RUNNING"
  }
}
```

Wenn Sie eine Regel in der Konsole definieren, wählen Sie die Option Input Transformer unter Configure input (Eingabe konfigurieren) aus. Diese Option zeigt zwei Textfelder an: eines für den Input Path (Eingabepfad) und eines für die Input Template (Eingabevorlage).

Der Eingabepfad wird verwendet, um Variablen zu definieren. Verwenden Sie den JSON-Pfad, um auf Elemente in Ihrem Ereignis zu verweisen und diese Werte in Variablen zu speichern. Sie könnten beispielsweise einen Eingabepfad erstellen, um auf Werte in dem Beispielereignis zu verweisen, indem Sie Folgendes in das erste Textfeld eingeben. Sie können auch Klammern und Indizes verwenden, um Elemente aus Arrays abzurufen.

> **Note**
>
> EventBridge ersetzt Eingangstransformatoren zur Laufzeit, um eine gültige JSON-Ausgabe sicherzustellen. Setzen Sie aus diesem Grund Variablen, die auf JSON-Pfadparameter verweisen, in Anführungszeichen, Variablen, die sich auf JSON-Objekte oder -Arrays beziehen, jedoch nicht in Anführungszeichen.

```
{
  "timestamp" : "$.time",
  "instance" : "$.detail.instance-id",
  "state" : "$.detail.state",
  "resource" : "$.resources[0]"
}
```

Damit werden vier Variablen definiert, `<timestamp>`, `<instance>`, `<state>` und `<resource>`. Sie können auf diese Variablen beim Erstellen Ihrer Eingabevorlage verweisen.

Die Eingabevorlage ist eine Vorlage für die Informationen, die Sie an Ihr Ziel übergeben möchten. Sie können eine Vorlage erstellen, die entweder eine Zeichenfolge oder JSON an das Ziel übergibt. Unter Verwendung des vorherigen Ereignisses und Eingabepfads wird das Ereignis in den folgenden Beispielen für Eingabevorlagen in die Beispielausgabe transformiert, bevor es an ein Ziel weitergeleitet wird.

Beschreibung	Vorlage	Output
Einfache Zeichenfolge	`"instance <instance> is in <state>"`	`"instance i-0123456789 is in RUNNING"`

Beschreibung	Vorlage	Output
Zeichenfolge mit Anführungszeichen, die durch Escape-Zeichen geschützt sind	`"instance \"<instance> \" is in <state>"`	`"instance \"i-01234 56789\" is in RUNNING"` Beachten Sie, dass dies das Verhalten in der EventBrid ge Konsole ist. In der AWS CLI werden die Schrägstr iche durch Escape-Zeichen geschützt. Das Ergebnis lautet `"instance "i-012345 6789" is in RUNNING"` .
Einfache JSON	`{ "instance" : <instance>, "state": <state> }`	`{ "instance" : "i-0123456789", "state": "RUNNING" }`
JSON mit Zeichenfolgen und Variablen	`{ "instance" : <instance >, "state": "<state>", "instanceStatus": "instance \"<instance> \" is in <state>" }`	`{ "instance" : "i-012345 6789", "state": "RUNNING", "instanceStatus": "instance \"i-01234 56789\" is in RUNNING" }`

Beschreibung	Vorlage	Output
JSON mit einer Mischung aus Variablen und statischen Informationen	`{ "instance" : <instance>, "state": [ 9, <state>, true ], "Transformed" : "Yes" }`	`{ "instance" : "i-0123456789", "state": [ 9, "RUNNING", true ], "Transformed" : "Yes" }`
Einbeziehen von reservierten Variablen in JSON	`{ "instance" : <instance>, "state": <state>, "ruleArn" : <aws.events.rule-arn>, "ruleName" : <aws.events.rule-name>, "originalEvent" : <aws.events.event.json> }`	`{ "instance" : "i-0123456789", "state": "RUNNING", "ruleArn" : "arn:aws:events:us-east-2:123456789012:rule/example", "ruleName" : "example", "originalEvent" : { ... // commented for brevity } }`
Einbeziehen von reservierten Variablen in eine Zeichenfolge	`"<aws.events.rule-name> triggered"`	`"example triggered"`
CloudWatch Amazon-Protokollgruppe	`{ "timestamp" : <timestamp>, "message": "instance \"<instance>\" is in <state>" }`	`{ "timestamp" : 2015-11-11T21:29:54Z, "message": "instance "i-0123456789" is in RUNNING }`

Transformieren von Eingaben mithilfe der EventBridge API

Informationen zur Verwendung der EventBridge API zur Transformation von Eingaben finden Sie unter Verwenden von Input Transformer, um Daten aus einem Ereignis zu extrahieren und diese Daten in das Ziel einzugeben.

Transformieren von Eingaben mithilfe von AWS CloudFormation

Hinweise zur Verwendung AWS CloudFormation zum Transformieren von Eingaben finden Sie unter AWS::Events::Rule InputTransformer.

Häufige Probleme beim Transformieren von Eingaben

Dies sind einige häufig auftretende Probleme bei der Transformation von Eingaben in EventBridge:

- Für Zeichenfolgen sind Anführungszeichen erforderlich.
- Beim Erstellen des JSON-Pfads für Ihre Vorlage erfolgt keine Validierung.
- Wenn Sie eine Variable angeben, die einem JSON-Pfad entspricht, der im Ereignis nicht vorhanden ist, wird diese Variable nicht erstellt und nicht in der Ausgabe angezeigt.
- JSON-Eigenschaften wie `aws.events.event.json` können nur als Wert eines JSON-Felds verwendet werden, nicht inline in anderen Zeichenfolgen.
- EventBridge maskiert Werte, die mit dem Eingabepfad extrahiert wurden, nicht, wenn die Eingabevorlage für ein Ziel gefüllt wird.
- Wenn ein JSON-Pfad auf ein JSON-Objekt oder ein JSON-Array verweist, die Variable jedoch in einer Zeichenfolge referenziert wird, werden alle internen Anführungszeichen EventBridge entfernt, um sicherzustellen, dass eine gültige Zeichenfolge vorliegt. Bei einer Variablen, auf die `<detail>` verwiesen wird`$.detail`, <detail>würde „Detail is“ beispielsweise dazu führen, dass Anführungszeichen aus dem Objekt EventBridge entfernt werden.

  Wenn Sie also ein JSON-Objekt ausgeben möchten, das auf einer einzelnen JSON-Pfadvariablen basiert, müssen Sie es als Schlüssel platzieren. In diesem Beispiel `{"detail": <detail>}`.
- Für Variablen, die Zeichenfolgen darstellen, sind keine Anführungszeichen erforderlich. Sie sind zulässig, fügen aber während der Transformation EventBridge automatisch Anführungszeichen zu Zeichenkettenvariablenwerten hinzu, um sicherzustellen, dass es sich bei der Transformationsausgabe um ein gültiges JSON-Format handelt. EventBridge fügt Variablen, die JSON-Objekte oder -Arrays darstellen, keine Anführungszeichen hinzu. Fügen Sie für Variablen, die JSON-Objekte oder -Arrays darstellen, keine Anführungszeichen hinzu.

Die folgende Eingabevorlage enthält beispielsweise Variablen, die sowohl Zeichenfolgen als auch JSON-Objekte darstellen:

```
{
  "ruleArn" : <aws.events.rule-arn>,
  "ruleName" : <aws.events.rule-name>,
  "originalEvent" : <aws.events.event.json>
}
```

Das Ergebnis ist gültiges JSON mit den richtigen Anführungszeichen:

```
{
  "ruleArn" : "arn:aws:events:us-east-2:123456789012:rule/example",
  "ruleName" : "example",
  "originalEvent" : {
    ... // commented for brevity
  }
}
```

- Bei (Nicht-JSON-) Textausgabe als mehrzeilige Zeichenfolgen setzen Sie jede einzelne Zeile in Ihrer Eingabevorlage in doppelte Anführungszeichen.

  Wenn Sie beispielsweise Amazon Inspector Finding-Ereignisse mit dem folgenden Ereignismuster abgleichen würden:

  ```
  {
    "detail": {
      "severity": ["HIGH"],
      "status": ["ACTIVE"]
    },
    "detail-type": ["Inspector2 Finding"],
    "source": ["inspector2"]
  }
  ```

  Und mit dem folgenden Eingabepfad:

  ```
  {
    "account": "$.detail.awsAccountId",
    "ami": "$.detail.resources[0].details.awsEc2Instance.imageId",
    "arn": "$.detail.findingArn",
  ```

```
  "description": "$.detail.description",
  "instance": "$.detail.resources[0].id",
  "platform": "$.detail.resources[0].details.awsEc2Instance.platform",
  "region": "$.detail.resources[0].region",
  "severity": "$.detail.severity",
  "time": "$.time",
  "title": "$.detail.title",
  "type": "$.detail.type"
}
```

Sie könnten die folgende Eingabevorlage verwenden, um eine mehrzeilige Zeichenkettenausgabe zu generieren:

```
"<severity> severity finding <title>"
"Description: <description>"
"ARN: \"<arn>\""
"Type: <type>"
"AWS Account: <account>"
"Region: <region>"
"EC2 Instance: <instance>"
"Platform: <platform>"
"AMI: <ami>"
```

Konfigurieren eines Eingabe-Transformators als Teil der Erstellung einer Regel

Im Rahmen der Erstellung einer Regel können Sie einen Eingangstransformator angeben, der verwendet werden EventBridge soll, um übereinstimmende Ereignisse zu verarbeiten, bevor diese Ereignisse an das angegebene Ziel gesendet werden. Sie können Eingangstransformatoren für Ziele konfigurieren, bei denen es sich um AWS Dienste oder API-Ziele handelt.

So erstellen Sie einen Zieleingabe-Transformator als Teil einer Regel

1. Befolgen Sie die Schritte zum Erstellen einer Regel, wie unter ??? beschrieben.
2. Erweitern Sie in Schritt 3 – Ziel(e) auswählen die Option Zusätzliche Einstellungen.
3. Wählen Sie für Zieleingabe konfigurieren die Option Eingabe-Transformator im Dropdown-Menü aus.

   Klicken Sie auf Eingabe-Transformator konfigurieren.

EventBridge zeigt das Dialogfeld Eingangstransformator konfigurieren an.

4. Wählen Sie im Abschnitt Beispielereignis einen Beispiel-Ereignistyp aus, anhand dessen Sie das Ereignismuster testen möchten. Sie können ein AWS Ereignis oder ein Partnerereignis auswählen oder ein eigenes benutzerdefiniertes Ereignis eingeben.

   AWS events

   Wählen Sie aus Ereignissen aus, die von unterstützten AWS-Services ausgegeben wurden.

   1. Wählen Sie AWS -Ereignisse aus.
   2. Wählen Sie unter Beispielereignisse die gewünschte AWS Veranstaltung aus. Die Veranstaltungen werden nach AWS Service organisiert.

      Wenn Sie ein Ereignis auswählen, wird das Beispielereignis EventBridge aufgefüllt.

      Wenn Sie beispielsweise S3 Object Created wählen, EventBridge wird ein Beispielereignis vom Typ S3 Object Created angezeigt.
   3. (Optional) Sie können auch Kopieren auswählen, um das Beispielereignis in die Zwischenablage Ihres Geräts zu kopieren.

   Partner events

   Wählen Sie Ereignisse aus, die von EventBridge unterstützenden Drittanbieterdiensten wie Salesforce ausgelöst werden.

   1. Wählen Sie EventBridge Partnerereignisse aus.
   2. Wählen Sie unter Beispielereignisse das gewünschte Partnerereignis aus. Die Ereignisse sind nach Partner organisiert.

      Wenn Sie ein Ereignis auswählen, wird das Beispielereignis EventBridge aufgefüllt.
   3. (Optional) Sie können auch Kopieren auswählen, um das Beispielereignis in die Zwischenablage Ihres Geräts zu kopieren.

   Enter your own

   Geben Sie Ihr eigenes Ereignis als JSON-Text ein.

1. Wählen Mein eigenes eingeben aus.
2. EventBridge füllt das Beispielereignis mit einer Vorlage mit den erforderlichen Ereignisattributen auf.
3. Bearbeiten Sie das Beispielereignis nach Bedarf und fügen Sie es hinzu. Das Beispielereignis muss gültiges JSON sein.
4. (Optional) Sie können auch eine der folgenden Optionen wählen:
   - Kopieren – Kopiert das Beispielereignis in die Zwischenablage Ihres Geräts.
   - Verschönern – Erleichtert das Lesen des JSON-Texts durch Hinzufügen von Zeilenumbrüchen, Tabulatoren und Leerzeichen.

5. (Optional) Erweitern Sie den Abschnitt Beispiel für Eingabepfade, Vorlagen und Ausgaben, um Beispiele für Folgendes zu sehen:

   - So werden JSON-Pfade verwendet, um Variablen zu definieren, die Ereignisdaten darstellen
   - So können diese Variablen in einer Eingabe-Transformator-Vorlage verwendet werden
   - Die resultierende Ausgabe, die EventBridge an das Ziel gesendet wird

   Ausführlichere Beispiele für Eingabetransformationen finden Sie unter ???.

6. Definieren Sie im Abschnitt Zieleingabe-Transformator alle Variablen, die Sie in der Eingabevorlage verwenden möchten.

   Variablen verwenden einen JSON-Pfad, um auf Werte in der ursprünglichen Ereignisquelle zu verweisen. Sie können dann in der Eingabevorlage auf diese Variablen verweisen, um Daten aus dem ursprünglichen Quellereignis in das transformierte Ereignis einzubeziehen, das EventBridge an das Ziel weitergegeben wird. Sie können bis zu 100 Variablen definieren. Der Eingabe-Transformator muss gültiges JSON sein.

   Nehmen wir beispielsweise an, Sie hätten das AWS Ereignis S3 Object Created als Beispielereignis für diesen Eingangstransformator ausgewählt. Sie könnten dann die folgenden Variablen zur Verwendung in Ihrer Vorlage definieren:

```
{
  "requester": "$.detail.requester",
  "key": "$.detail.object.key",
  "bucket": "$.detail.bucket.name"
}
```

(Optional) Sie können auch Kopieren auswählen, um den Eingabe-Transformator in die Zwischenablage Ihres Geräts zu kopieren.

7. Verfassen Sie im Abschnitt Vorlage die Vorlage, anhand derer Sie bestimmen möchten, was EventBridge an das Ziel übergeben wird.

   Sie können JSON, Zeichenfolgen, statische Informationen, von Ihnen definierte Variablen sowie reservierte Variablen verwenden. Ausführlichere Beispiele für Eingabetransformationen finden Sie unter ???.

   Angenommen, Sie haben die Variablen im vorherigen Beispiel definiert. Sie könnten dann die folgende Vorlage erstellen, die auf diese Variablen sowie auf reservierte Variablen und statische Informationen verweist.

   ```
   {
       "message": "<requester> has created the object \"<key>\" in the bucket
    \"<bucket>\"",
       "RuleName": <aws.events.rule-name>,
       "ruleArn" : <aws.events.rule-arn>,
       "Transformed": "Yes"
   }
   ```

   (Optional) Sie können auch Kopieren auswählen, um die Vorlage in die Zwischenablage Ihres Geräts zu kopieren.

8. Wenn Sie Ihre Vorlage testen möchten, wählen Sie Ausgabe erzeugen aus.

   EventBridge verarbeitet das Beispielereignis auf der Grundlage der Eingabevorlage und zeigt die transformierte Ausgabe an, die unter Ausgabe generiert wurde. Dies sind die Informationen, EventBridge die anstelle des ursprünglichen Quellereignisses an das Ziel weitergegeben werden.

   Die erzeugte Ausgabe für die oben beschriebene Beispiel-Eingabevorlage wäre die folgende:

   ```
   {
       "message": "123456789012 has created the object "example-key" in the bucket
    "example-bucket"",
       "RuleName": rule-name,
       "ruleArn" : arn:aws:events:us-east-1:123456789012:rule/rule-name,
       "Transformed": "Yes"
   ```

```
}
```

(Optional) Sie können auch Kopieren auswählen, um die erzeugte Ausgabe in die Zwischenablage Ihres Geräts zu kopieren.

9. Wählen Sie Bestätigen aus.
10. Befolgen Sie die restlichen Schritte zum Erstellen einer Regel, wie unter ??? beschrieben.

Testen eines Zieleingangstransformators mit der EventBridge Sandbox

Sie können Eingangstransformatoren verwenden, um den Text eines Ereignisses anzupassen, EventBridge bevor die Informationen an das Ziel einer Regel weitergegeben werden.

Die Konfiguration eines Eingabe-Transformators ist in der Regel Teil eines größeren Prozesses, bei dem während der Erstellung einer neuen Regel oder der Bearbeitung einer vorhandenen ein Ziel angegeben wird. Mithilfe der Sandbox in können Sie EventBridge jedoch schnell einen Eingangstransformator konfigurieren und anhand eines Beispielereignisses bestätigen, dass Sie die gewünschte Ausgabe erhalten, ohne eine Regel erstellen oder bearbeiten zu müssen.

Weitere Informationen zu Eingabetransformationen finden Sie unter ???.

So testen Sie einen Zieleingabe-Transformator

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie unter Entwicklerressourcen die Option Sandbox und auf der Sandbox-Seite die Registerkarte Zieleingabe-Transformator aus.
3. Wählen Sie im Abschnitt Beispielereignis einen Beispiel-Ereignistyp aus, anhand dessen Sie das Ereignismuster testen möchten. Sie können eine AWS Veranstaltung oder eine Partnerveranstaltung auswählen oder Ihre eigene benutzerdefinierte Veranstaltung eingeben.

   AWS events

   Wählen Sie aus Ereignissen aus, die von unterstützten AWS-Services ausgegeben wurden.

   1. Wählen Sie AWS -Ereignisse aus.
   2. Wählen Sie unter Beispielereignisse die gewünschte AWS Veranstaltung aus. Die Veranstaltungen werden nach AWS Service organisiert.

      Wenn Sie ein Ereignis auswählen, wird das Beispielereignis EventBridge aufgefüllt.

Wenn Sie beispielsweise S3 Object Created wählen, EventBridge wird ein Beispielereignis vom Typ S3 Object Created angezeigt.

3. (Optional) Sie können auch Kopieren auswählen, um das Beispielereignis in die Zwischenablage Ihres Geräts zu kopieren.

Partner events

Wählen Sie Ereignisse aus, die von EventBridge unterstützenden Drittanbieterdiensten wie Salesforce ausgelöst werden.

1. Wählen Sie EventBridge Partnerereignisse aus.
2. Wählen Sie unter Beispielereignisse das gewünschte Partnerereignis aus. Die Ereignisse sind nach Partner organisiert.

   Wenn Sie ein Ereignis auswählen, wird das Beispielereignis EventBridge aufgefüllt.
3. (Optional) Sie können auch Kopieren auswählen, um das Beispielereignis in die Zwischenablage Ihres Geräts zu kopieren.

Enter your own

Geben Sie Ihr eigenes Ereignis als JSON-Text ein.

1. Wählen Mein eigenes eingeben aus.
2. EventBridge füllt das Beispielereignis mit einer Vorlage mit den erforderlichen Ereignisattributen auf.
3. Bearbeiten Sie das Beispielereignis nach Bedarf und fügen Sie es hinzu. Das Beispielereignis muss gültiges JSON sein.
4. (Optional) Sie können auch eine der folgenden Optionen wählen:
   - Kopieren – Kopiert das Beispielereignis in die Zwischenablage Ihres Geräts.
   - Verschönern – Erleichtert das Lesen des JSON-Texts durch Hinzufügen von Zeilenumbrüchen, Tabulatoren und Leerzeichen.

4. (Optional) Erweitern Sie den Abschnitt Beispiel für Eingabepfade, Vorlagen und Ausgaben, um Beispiele für Folgendes zu sehen:

- So werden JSON-Pfade verwendet, um Variablen zu definieren, die Ereignisdaten darstellen

- So können diese Variablen in einer Eingabe-Transformator-Vorlage verwendet werden
- Die resultierende Ausgabe, die EventBridge an das Ziel gesendet wird

Ausführlichere Beispiele für Eingabetransformationen finden Sie unter ???.

5. Definieren Sie im Abschnitt Zieleingabe-Transformator alle Variablen, die Sie in der Eingabevorlage verwenden möchten.

   Variablen verwenden einen JSON-Pfad, um auf Werte in der ursprünglichen Ereignisquelle zu verweisen. Sie können dann in der Eingabevorlage auf diese Variablen verweisen, um Daten aus dem ursprünglichen Quellereignis in das transformierte Ereignis einzubeziehen, das EventBridge an das Ziel weitergegeben wird. Sie können bis zu 100 Variablen definieren. Der Eingabe-Transformator muss gültiges JSON sein.

   Nehmen wir beispielsweise an, Sie hätten das AWS Ereignis S3 Object Created als Beispielereignis für diesen Eingangstransformator ausgewählt. Sie könnten dann die folgenden Variablen zur Verwendung in Ihrer Vorlage definieren:

   ```
   {
     "requester": "$.detail.requester",
     "key": "$.detail.object.key",
     "bucket": "$.detail.bucket.name"
   }
   ```

   (Optional) Sie können auch Kopieren auswählen, um den Eingabe-Transformator in die Zwischenablage Ihres Geräts zu kopieren.

6. Verfassen Sie im Abschnitt Vorlage die Vorlage, anhand derer Sie bestimmen möchten, was EventBridge an das Ziel übergeben wird.

   Sie können JSON, Zeichenfolgen, statische Informationen, von Ihnen definierte Variablen sowie reservierte Variablen verwenden. Ausführlichere Beispiele für Eingabetransformationen finden Sie unter ???.

   Angenommen, Sie haben die Variablen im vorherigen Beispiel definiert. Sie könnten dann die folgende Vorlage erstellen, die auf diese Variablen sowie auf reservierte Variablen und statische Informationen verweist.

   ```
   {
   ```

```
    "message": "<requester> has created the object \"<key>\" in the bucket
 \"<bucket>\"",
    "RuleName": <aws.events.rule-name>,
    "ruleArn" : <aws.events.rule-arn>,
    "Transformed": "Yes"
}
```

(Optional) Sie können auch Kopieren auswählen, um die Vorlage in die Zwischenablage Ihres Geräts zu kopieren.

7. Wenn Sie Ihre Vorlage testen möchten, wählen Sie Ausgabe erzeugen aus.

   EventBridge verarbeitet das Beispielereignis auf der Grundlage der Eingabevorlage und zeigt die transformierte Ausgabe an, die unter Ausgabe generiert wurde. Dies sind die Informationen, EventBridge die anstelle des ursprünglichen Quellereignisses an das Ziel weitergegeben werden.

   Die erzeugte Ausgabe für die oben beschriebene Beispiel-Eingabevorlage wäre die folgende:

```
{
    "message": "123456789012 has created the object "example-key" in the bucket
 "example-bucket"",
    "RuleName": rule-name,
    "ruleArn" : arn:aws:events:us-east-1:123456789012:rule/rule-name,
    "Transformed": "Yes"
}
```

   (Optional) Sie können auch Kopieren auswählen, um die erzeugte Ausgabe in die Zwischenablage Ihres Geräts zu kopieren.

Archivieren und Wiederholen in Amazon EventBridge

In EventBridge können Sie ein Archiv mit Ereignissen erstellen, sodass Sie sie zu einem späteren Zeitpunkt problemlos wiederholen können. Beispielsweise möchten Sie möglicherweise Ereignisse wiederholen, um Fehler zu beheben oder neue Funktionen in Ihrer Anwendung zu validieren.

> **Note**
>
> Es kann zu einer Verzögerung zwischen der Veröffentlichung eines Ereignisses in einem Event Bus und dem Eingang des Ereignisses im Archiv kommen. Wir empfehlen, die Wiederholung archivierter Ereignisse um 10 Minuten zu verschieben, um sicherzustellen, dass alle Ereignisse wiederholt werden.

Das folgende Video zeigt die Verwendung von Archivieren und Wiederholen: Erstellen von Archiven und Wiederholungen

Themen

- Archivierung Amazon EventBridge Amazon-Ereignissen
- Wiederholen archivierter Amazon-EventBridge-Ereignisse

Archivierung Amazon EventBridge Amazon-Ereignissen

Wenn Sie ein Archiv in erstellen EventBridge, können Sie bestimmen, welche Ereignisse an das Archiv gesendet werden, indem Sie ein Ereignismuster angeben. EventBridge sendet Ereignisse, die dem Ereignismuster entsprechen, an das Archiv. Sie legen auch den Aufbewahrungszeitraum fest, um Ereignisse im Archiv zu speichern, bevor sie verworfen werden.

EventBridge Verschlüsselt standardmäßig Ereignisdaten in einem Archiv mithilfe des 256-Bit-Advanced Encryption Standard (AES-256) unter einem AWS eigenen CMK, wodurch Ihre Daten vor unbefugtem Zugriff geschützt werden.

> **Note**
>
> Die `SizeBytes` Werte `EventCount` und Werte des `DescribeArchive`Vorgangs haben einen Abgleichszeitraum von 24 Stunden. Daher werden kürzlich abgelaufene oder neu archivierte Ereignisse möglicherweise nicht sofort in diesen Werten berücksichtigt.

So erstellen Sie ein Archiv für alle Ereignisse

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im linken Navigationsbereich die Option Archive aus.
3. Wählen Sie Archiv erstellen.
4. Geben Sie unter Archivdetails einen Namen für das Archiv ein. Der Name muss für das Konto in der ausgewählten Region eindeutig sein.

   Sie können den Namen nach der Erstellung des Archivs nicht mehr ändern.
5. (Optional) Geben Sie eine Beschreibung für das Archiv ein.
6. Wählen Sie bei Quelle den Event Bus aus, der die Ereignisse ausgibt, die an das Archiv gesendet werden sollen.
7. Führen Sie bei Aufbewahrungszeitraum einen der folgenden Schritte aus:
   - Wählen Sie Unbegrenzt, um die Ereignisse im Archiv beizubehalten und niemals zu löschen.
   - Geben Sie die Anzahl der Tage für die Beibehaltung der Ereignisse ein. EventBridge Löscht die Ereignisse nach der angegebenen Anzahl von Tagen aus dem Archiv.
8. Wählen Sie Weiter aus.
9. Wählen Sie unter Ereignismuster die Option Keine Ereignisfilterung aus.

10. Wählen Sie Archiv erstellen.

So erstellen Sie ein Archiv mit einem Ereignismuster

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im linken Navigationsbereich die Option Archive aus.
3. Wählen Sie Archiv erstellen.
4. Geben Sie unter Archivdetails einen Namen für das Archiv ein. Der Name muss für das Konto in der ausgewählten Region eindeutig sein.

   Sie können den Namen nach der Erstellung des Archivs nicht mehr ändern.
5. (Optional) Geben Sie eine Beschreibung für das Archiv ein.
6. Wählen Sie bei Quelle den Event Bus aus, der die Ereignisse ausgibt, die an das Archiv gesendet werden sollen.
7. Führen Sie bei Aufbewahrungszeitraum einen der folgenden Schritte aus:
   - Wählen Sie Unbegrenzt*, um die Ereignisse im Archiv beizubehalten und niemals zu löschen.
   - Geben Sie die Anzahl der Tage für die Beibehaltung der Ereignisse ein. EventBridge Löscht die Ereignisse nach der angegebenen Anzahl von Tagen aus dem Archiv.
8. Wählen Sie Weiter aus.
9. Wählen Sie unter Ereignismuster die Option Filtern von Ereignissen nach Ereignismusterübereinstimmung aus.
10. Führen Sie eine der folgenden Aktionen aus:
    - Wählen Sie Pattern Builder und dann den Dienstanbieter aus. Wenn Sie AWS auswählen, wählen Sie auch den AWS -Servicenamen und den Ereignistyp aus, die im Muster verwendet werden sollen.
    - Wählen Sie JSON-Editor aus, um ein Muster manuell zu erstellen. Sie können das Muster auch aus einer Regel kopieren und dann in den JSON-Editor einfügen.
11. Wählen Sie Archiv erstellen.

Um zu überprüfen, ob Ereignisse erfolgreich an das Archiv gesendet wurden, können Sie mithilfe der `DescribeArchive` EventBridge API überprüfen, ob das der Anzahl der Ereignisse im Archiv `EventCount` entspricht. Wenn der Wert 0 ist, gibt es keine Ereignisse im Archiv.

Wiederholen archivierter Amazon-EventBridge-Ereignisse

Nachdem Sie ein Archiv erstellt haben, können Sie Ereignisse aus dem Archiv wiederholen. Wenn Sie beispielsweise eine Anwendung mit zusätzlichen Funktionen aktualisieren, können Sie Verlaufsereignisse wiederholen, um sicherzustellen, dass die Ereignisse erneut verarbeitet werden, damit die Anwendung konsistent bleibt. Sie können ein Archiv auch verwenden, um Ereignisse für neue Funktionen zu wiederholen. Wenn Sie Ereignisse wiederholen, können Sie angeben, aus welchem Archiv Ereignisse wiederholt werden sollen, die Start- und Endzeit für die Wiederholung des Ereignisses, den Event Bus oder eine oder mehrere Regeln, zu denen die Ereignisse wiederholt werden sollen.

Ereignisse werden nicht unbedingt in derselben Reihenfolge wiederholt, in der sie dem Archiv hinzugefügt wurden. Bei einer Wiederholung werden Ereignisse so verarbeitet, dass sie auf der Grundlage der Uhrzeit des Ereignisses wiederholt werden, und sie werden in Intervallen von einer Minute wiederholt. Wenn Sie eine Startzeit und eine Endzeit für ein Ereignis angeben, die einen Zeitraum von 20 Minuten abdecken, werden die Ereignisse der ersten Minute dieses 20-Minuten-Bereichs zuerst wiederholt. Dann werden die Ereignisse der zweiten Minute wiederholt. Sie können die `DescribeReplay`-Operation der EventBridge-API verwenden, um den Fortschritt einer Wiederholung zu ermitteln. `EventLastReplayedTime` gibt den Zeitstempel des letzten wiederholten Ereignisses zurück.

Ereignisse werden basierend auf, aber getrennt von, dem Limit für `PutEvents`-Transaktionen pro Sekunde für das AWS-Konto wiederholt. Sie können eine Erhöhung des Limits für PutEvents beantragen. Weitere Informationen finden Sie unter Amazon-EventBridge-Kontingente.

> **Note**
>
> Sie können maximal 10 aktive gleichzeitige Wiederholungen pro Konto und AWS-Region festlegen.

So starten Sie eine Ereigniswiederholung

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im linken Navigationsbereich Wiederholungen.
3. Wählen Sie Neue Wiederholung starten.
4. Geben Sie für die Wiederholung einen Namen und optional eine Beschreibung ein.

5. Wählen Sie für Quelle das Archiv aus, aus dem Ereignisse wiederholt werden sollen.
6. Für das Ziel können Sie Ereignisse nur zu demselben Event Bus wiederholen, der die Ereignisse ausgegeben hat.
7. Führen Sie für Regeln angeben einen der folgenden Schritte aus:
   - Wählen Sie Alle Regeln, um Ereignisse zu allen Regeln zu wiederholen.
   - Wählen Sie Regeln angeben und anschließend die Regel oder Regeln aus, zu denen die Ereignisse wiederholt werden sollen.
8. Geben Sie unter Zeitrahmen der Wiederholung das Datum, die Uhrzeit und die Zeitzone für die Startzeit und die Endzeit an. Nur Ereignisse, die zwischen der Startzeit und der Endzeit aufgetreten sind, werden wiederholt.
9. Wählen Sie Start replay „Wiederholung starten“.

Wenn die Ereignisse aus dem Archivierten wiederholt werden, weist die Wiedergabe den Status Abgeschlossen auf.

Wenn Sie eine Wiederholung starten und sie dann unterbrechen möchten, können Sie sie abbrechen, sofern der Status Wird gestartet oder Wird ausgeführt lautet.

So brechen Sie eine Wiederholung ab

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im linken Navigationsbereich Wiederholungen.
3. Wählen Sie die Wiederholung aus, die Sie abbrechen möchten.
4. Klicken Sie auf Abbrechen.

EventBridge Amazon-Pfeifen

Amazon EventBridge Pipes verbindet Quellen mit Zielen. Pipes sind für point-to-point Integrationen zwischen unterstützten Quellen und Zielen vorgesehen und unterstützen erweiterte Transformationen und Anreicherungen. Es reduzieren den Bedarf an Fachwissen und Integrationscode bei der Entwicklung ereignisgesteuerter Architekturen und fördern die Konsistenz zwischen den Anwendungen Ihres Unternehmens. Zum Einrichten einer Pipe wählen Sie die Quelle aus, fügen Sie optionale Filterung hinzu, definieren Sie die optionale Anreicherung und wählen Sie das Ziel für die Ereignisdaten.

> **Note**
>
> Sie können Ereignisse auch mithilfe von Event Buses weiterleiten. Event-Busse eignen sich hervorragend für die many-to-many Weiterleitung von Ereignissen zwischen ereignisgesteuerten Diensten. Weitere Informationen finden Sie unter ???.

Wie funktionieren Pipes EventBridge

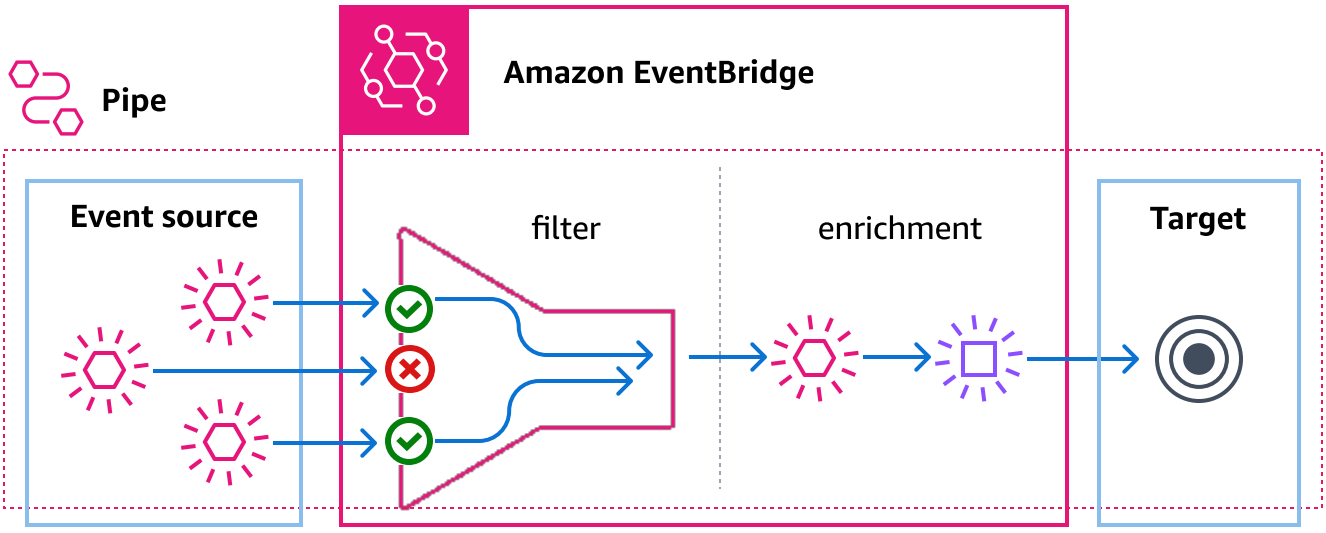
Auf hoher Ebene funktioniert EventBridge Pipes wie folgt:

1. Sie erstellen eine Pipe in Ihrem Konto. Dies umfasst:
   - Geben Sie eine der unterstützten Ereignisquellen an, von denen die Pipe Ereignisse empfangen soll.
   - Optional können Sie einen Filter so konfigurieren, dass die Pipe nur eine Teilmenge der Ereignisse verarbeitet, die sie von der Quelle empfängt.
   - Optional können Sie einen Anreicherungsschritt konfigurieren, der die Ereignisdaten optimiert, bevor sie an das Ziel gesendet werden.
   - Geben Sie eines der unterstützten Ziele an, an das die Pipe Ereignisse senden soll.
2. Die Ereignisquelle beginnt mit dem Senden von Ereignissen an die Pipe und die Pipe verarbeitet das Ereignis, bevor es an das Ziel gesendet wird.
   - Wenn Sie einen Filter konfiguriert haben, wertet die Pipe das Ereignis aus und sendet es nur dann an das Ziel, wenn es diesem Filter entspricht.

     Ihnen werden nur die Ereignisse in Rechnung gestellt, die dem Filter entsprechen.

- Wenn Sie eine Anreicherung konfiguriert haben, führt die Pipe diese Anreicherung für das Ereignis durch, bevor es an das Ziel gesendet wird.

  Wenn die Ereignisse gestapelt werden, behält die Anreicherung die Reihenfolge der Ereignisse im Stapel bei.



Eine Pipe könnte beispielsweise verwendet werden, um ein E-Commerce-System zu erstellen. Angenommen, Sie haben eine API, die Kundeninformationen wie Lieferadressen enthält.

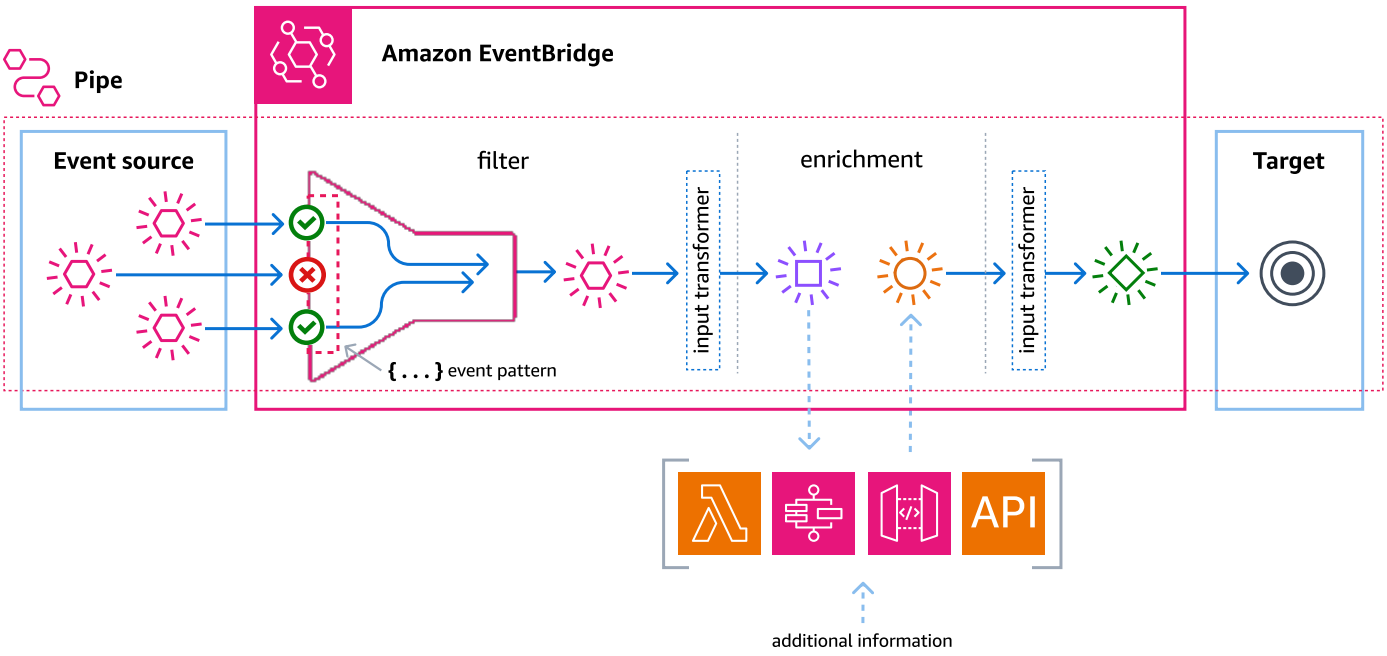
1. Dann erstellen Sie eine Pipe mit Folgendem:
   - Einer Amazon-SQS-Nachrichtenwarteschlange für „Bestellung erhalten“ als Ereignisquelle
   - Eine EventBridge API-Destination als Erweiterung
   - Eine AWS Step Functions Zustandsmaschine als Ziel
2. Wenn dann eine Amazon-SQS-Nachricht für „Bestellung erhalten“ in der Warteschlange erscheint, wird sie an die Pipe gesendet.
3. Die Pipe sendet diese Daten dann an die EventBridge API Destination Enrichment, die die Kundeninformationen für diese Bestellung zurückgibt.
4. Schließlich sendet die Pipe die angereicherten Daten an die AWS Step Functions Zustandsmaschine, die die Bestellung verarbeitet.

EventBridge Konzepte von Pipes

Hier sehen Sie sich die grundlegenden Komponenten von EventBridge Pipes genauer an.

Pipe

Eine Pipe leitet Ereignisse von einer einzelnen Quelle an ein einzelnes Ziel weiter. Die Pipe bietet auch die Möglichkeit, nach bestimmten Ereignissen zu filtern und die Ereignisdaten anzureichern, bevor sie an das Ziel gesendet werden.



Quelle

EventBridge Pipes empfängt Ereignisdaten aus einer Vielzahl von Quellen, wendet optionale Filter und Anreicherungen auf diese Daten an und sendet sie an ein Ziel. Wenn eine Quelle die Reihenfolge der an Pipes gesendeten Ereignisse erzwingt, wird diese Reihenfolge während des gesamten Prozesses bis zum Ziel beibehalten.

Weitere Informationen zu Quellen finden Sie unter ???.

Filter

Eine Pipe kann die Ereignisse einer bestimmten Quelle filtern und anschließend nur eine Teilmenge dieser Ereignisse verarbeiten. Zum Konfigurieren der Filterung für eine Pipe definieren Sie ein

Ereignismuster, anhand dessen die Pipe bestimmt, welche Ereignisse an das Ziel gesendet werden sollen.

Ihnen werden nur die Ereignisse in Rechnung gestellt, die dem Filter entsprechen.

Weitere Informationen finden Sie unter ???.

Anreicherung

Mit dem Anreicherungsschritt von EventBridge Pipes können Sie die Daten aus der Quelle verbessern, bevor Sie sie an das Ziel senden. Beispielsweise erhalten Sie möglicherweise Ereignisse für Ticket erstellt, die nicht die vollständigen Ticketdaten enthalten. Mithilfe der Anreicherung können Sie eine Lambda-Funktion veranlassen, die `get-ticket`-API für die vollständigen Ticketdetails aufzurufen. Die Pipe kann diese Informationen dann an ein Ziel senden.

Weitere Informationen zur Anreicherung von Ereignisdaten finden Sie unter ???.

Ziel

Nachdem die Ereignisdaten gefiltert und angereichert wurden, können Sie angeben, wie sie per Pipe an ein bestimmtes Ziel gesendet werden sollen, z. B. an einen Amazon Kinesis Kinesis-Stream oder eine CloudWatch Amazon-Protokollgruppe. Eine Liste der verfügbaren Ziele finden Sie unter ???.

Sie können die Daten transformieren, nachdem sie optimiert wurden und bevor sie über die Pipe an das Ziel gesendet werden. Weitere Informationen finden Sie unter ???.

Mehrere Pipes, jede mit einer anderen Quelle, können Ereignisse an dasselbe Ziel senden.

Sie können Pipes und Event Buses auch zusammen verwenden, um Ereignisse an mehrere Ziele zu senden. Ein häufiger Anwendungsfall ist die Erstellung einer Pipe mit einem Event Bus als Ziel. Die Pipe sendet Ereignisse an den Event Bus, der diese Ereignisse dann an mehrere Ziele weiterleitet. Sie könnten beispielsweise eine Pipe mit einem DynamoDB-Stream für eine Quelle und einem Event Bus als Ziel erstellen. Die Pipe empfängt Ereignisse aus dem DynamoDB-Stream und sendet sie an den Event Bus, der sie dann gemäß den Regeln, die Sie für den Event Bus angegeben haben, an mehrere Ziele weiterleitet.

Berechtigungen für Amazon EventBridge Pipes

Beim Einrichten einer Pipe können Sie eine vorhandene Ausführungsrolle verwenden oder EventBridge eine für Sie mit den erforderlichen Berechtigungen erstellen lassen. Die Berechtigungen,

die EventBridge Pipes benötigt, variieren je nach Quelltyp und sind unten aufgeführt. Wenn Sie Ihre eigene Ausführungsrolle einrichten, müssen Sie diese Berechtigungen selbst hinzufügen.

> **Note**
>
> Wenn Sie sich nicht sicher sind, welche genauen Berechtigungen für den Zugriff auf die Quelle erforderlich sind, verwenden Sie die EventBridge-Pipes-Konsole, um eine neue Rolle zu erstellen, und überprüfen Sie dann die in der Richtlinie aufgeführten Aktionen.

Themen

Berechtigungen für DynamoDB-Ausführungsrollen

Für DynamoDB Streams benötigt EventBridge Pipes die folgenden Berechtigungen zum Verwalten von Ressourcen, die zu Ihrem DynamoDB-Datenstrom gehören.

- `dynamodb:DescribeStream`
- `dynamodb:GetRecords`
- `dynamodb:GetShardIterator`
- `dynamodb:ListStreams`

Um Datensätze über fehlgeschlagene Stapel an die Pipe-Warteschlange für unzustellbare Nachrichten zu senden, benötigt Ihre Pipe-Ausführungsrolle die folgende Berechtigung:

- `sqs:SendMessage`

Berechtigungen für Kinesis-Ausführungsrollen

Für Kinesis benötigt EventBridge Pipes die folgenden Berechtigungen zum Verwalten von Ressourcen, die zu Ihrem Kinesis-Datenstrom gehören.

- kinesis:DescribeStream
- kinesis:DescribeStreamSummary
- kinesis:GetRecords
- kinesis:GetShardIterator
- kinesis:ListShards
- kinesis:ListStreams
- kinesis:SubscribeToShard

Um Datensätze über fehlgeschlagene Stapel an die Pipe-Warteschlange für unzustellbare Nachrichten zu senden, benötigt Ihre Pipe-Ausführungsrolle die folgende Berechtigung:

- sqs:SendMessage

Berechtigungen für Amazon-MQ-Ausführungsrollen

Für Amazon MQ benötigt EventBridge Pipes die folgenden Berechtigungen zum Verwalten von Ressourcen, die zu Ihrem Amazon-MQ-Message-Broker gehören.

- mq:DescribeBroker
- secretsmanager:GetSecretValue
- ec2:CreateNetworkInterface
- ec2:DeleteNetworkInterface
- ec2:DescribeNetworkInterfaces
- ec2:DescribeSecurityGroups
- ec2:DescribeSubnets
- ec2:DescribeVpcs
- logs:CreateLogGroup
- logs:CreateLogStream
- logs:PutLogEvents

Berechtigungen für Amazon-MSK-Ausführungsrollen

Für Amazon MSK benötigt EventBridge die folgenden Berechtigungen zum Verwalten von Ressourcen, die zu Ihrem Amazon-MSK-Thema gehören.

> **Note**
>
> Wenn Sie die rollenbasierte IAM-Authentifizierung verwenden, benötigt Ihre Ausführungsrolle zusätzlich zu den unten aufgeführten Berechtigungen die in ??? aufgeführten.

- `kafka:DescribeClusterV2`
- `kafka:GetBootstrapBrokers`
- `ec2:CreateNetworkInterface`
- `ec2:DescribeNetworkInterfaces`
- `ec2:DescribeVpcs`
- `ec2:DeleteNetworkInterface`
- `ec2:DescribeSubnets`
- `ec2:DescribeSecurityGroups`
- `logs:CreateLogGroup`
- `logs:CreateLogStream`
- `logs:PutLogEvents`

Berechtigungen für selbstverwaltete Apache-Kafka-Ausführungsrollen

Für selbstverwaltetes Apache Kafka benötigt EventBridge die folgenden Berechtigungen zum Verwalten von Ressourcen, die zu Ihrem selbstverwalteten Apache-Kafka-Stream gehören.

Erforderliche Berechtigungen

Um Protokolle in einer Protokollgruppe in Amazon CloudWatch Logs zu erstellen und zu speichern, muss Ihre Pipe die folgenden Berechtigungen in ihrer Ausführungsrolle haben:

- `logs:CreateLogGroup`
- `logs:CreateLogStream`

- logs:PutLogEvents

Optionale Berechtigungen

Ihre Pipe benötigt möglicherweise auch Berechtigungen, um:

- Beschreiben Ihres Secrets-Manager-Secrets
- Zugriff auf Ihre AWS Key Management Service (AWS KMS) vom Kunden verwalteten Schlüssel
- Zugriff auf Ihre Amazon VPC

Secrets Manager und AWS KMS-Berechtigungen

Abhängig von der Art der Zugriffssteuerung, die Sie für Ihre Apache-Kafka-Broker konfigurieren, benötigt Ihre Pipe möglicherweise die Berechtigung, auf Ihr Secrets-Manager-Secret zuzugreifen oder Ihren vom Kunden verwalteten AWS KMS-Schlüssel zu entschlüsseln. Um auf diese Ressourcen zuzugreifen, muss die Ausführungsrolle Ihrer Funktion die folgenden Berechtigungen besitzen:

- secretsmanager:GetSecretValue
- kms:Decrypt

VPC-Berechtigungen

Wenn nur Benutzer innerhalb einer VPC auf Ihren selbstverwalteten Apache-Kafka-Cluster zugreifen können, muss Ihre Pipe die Berechtigung zum Zugriff auf Ihre Amazon-VPC-Ressourcen haben. Zu diesen Ressourcen gehören Ihre VPC, Subnetze, Sicherheitsgruppen und Netzwerkschnittstellen. Um auf diese Ressourcen zuzugreifen, muss die Ausführungsrolle Ihrer Pipe die folgenden Berechtigungen besitzen:

- ec2:CreateNetworkInterface
- ec2:DescribeNetworkInterfaces
- ec2:DescribeVpcs
- ec2:DeleteNetworkInterface
- ec2:DescribeSubnets
- ec2:DescribeSecurityGroups

Berechtigungen für Amazon-SQS-Ausführungsrollen

Für Amazon SQS benötigt EventBridge die folgenden Berechtigungen zum Verwalten von Ressourcen, die zu Ihrer Amazon-SQS-Warteschlange gehören.

- `sqs:ReceiveMessage`
- `sqs:DeleteMessage`
- `sqs:GetQueueAttributes`

Berechtigungen für Anreicherungen und Ziele

Um API-Aufrufe für die Ressourcen auszuführen, die Sie besitzen, muss EventBridge Pipes über die entsprechenden Berechtigungen verfügen. EventBridge Pipes verwendet die IAM-Rolle, die Sie für die Pipe angeben, für Anreicherungs- und Zielaufrufe mithilfe des IAM-Prinzipals `pipes.amazonaws.com`.

Eine EventBridge Amazon-Pipe erstellen

EventBridge Pipes ermöglicht Ihnen, point-to-point Integrationen zwischen Quellen und Zielen zu erstellen, einschließlich erweiterter Event-Transformationen und Anreicherungen. Um eine EventBridge Pipe zu erstellen, führen Sie die folgenden Schritte aus:

1. ???
2. ???
3. ???
4. ???
5. ???

Informationen zum Erstellen einer Pipe mit der AWS CLI finden Sie unter create-pipe in der AWS CLI-Befehlsreferenz.

Angeben einer Quelle

Geben Sie zunächst die Quelle an, von der die Pipe Ereignisse empfangen soll.

So geben Sie eine Pipe-Quelle mithilfe der Konsole an

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Pipes aus.
3. Wählen Sie Pipe erstellen aus.
4. Geben Sie einen Namen für die Pipe ein.
5. (Optional) Geben Sie eine Beschreibung für die Pipe ein.
6. Wählen Sie auf der Registerkarte Pipe erstellen unter Quelle den Quellentyp aus, den Sie für diese Pipe angeben möchten, und konfigurieren Sie die Quelle.

   Konfigurationseigenschaften unterscheiden sich je nach ausgewähltem Quellentyp:

   Confluent

   Um mithilfe der Konsole einen Confluent Cloud-Stream als Quelle zu konfigurieren

   1. Wählen Sie als Quelle Confluent Cloud aus.
   2. Geben Sie für Bootstrap-Server die `host:port`-Paaradressen Ihrer Broker ein.
   3. Geben Sie für Themenname den Namen des Themas ein, aus dem die Pipe lesen wird.
   4. (Optional) Wählen Sie für VPC die VPC aus, die Sie möchten. Wählen Sie dann für VPC-Subnetze die gewünschten Subnetze aus. Wählen Sie für VPC-Sicherheitsgruppen die Sicherheitsgruppen aus.
   5. Aktivieren Sie unter Authentifizierung — optional die Option Authentifizierung verwenden und gehen Sie wie folgt vor:

      a. Wählen Sie für Authentifizierungsmethode den Authentifizierungstyp aus.

      b. Wählen Sie für Geheimschlüssel den Geheimschlüssel aus.

      Weitere Informationen finden Sie unter Authenticate to Confluent Cloud-Ressourcen in der Confluent-Dokumentation.
   6. (Optional) Gehen Sie unter Weitere Einstellungen – optional wie folgt vor:

      a. Wählen Sie für Startposition eine der folgenden Optionen:

         - Neueste – Starten Sie das Lesen des Streams mit dem neuesten Datensatz im Shard.
         - Horizont trimmen – Starten Sie das Lesen des Streams mit dem letzten nicht getrimmten Datensatz im Shard. Dies ist der älteste Datensatz im Shard.

b. Geben Sie für Stapelgröße – optional eine maximale Anzahl von Datensätzen für jeden Stapel ein. Der Standardwert lautet 100.

c. Geben Sie für Stapelfenster – optional eine maximale Anzahl von Sekunden für das Sammeln von Datensätzen ein, bevor Sie fortfahren.

DynamoDB

1. Wählen Sie für Quelle DynamoDB aus.
2. Wählen Sie für DynamoDB-Stream den Stream aus, den Sie als Quelle verwenden möchten.
3. Wählen Sie für Startposition eine der folgenden Optionen:
   - Neueste – Starten Sie das Lesen des Streams mit dem neuesten Datensatz im Shard.
   - Horizont trimmen – Starten Sie das Lesen des Streams mit dem letzten nicht getrimmten Datensatz im Shard. Dies ist der älteste Datensatz im Shard.
4. (Optional) Gehen Sie unter Weitere Einstellungen – optional wie folgt vor:
   a. Geben Sie für Stapelgröße – optional eine maximale Anzahl von Datensätzen für jeden Stapel ein. Der Standardwert lautet 100.
   b. Geben Sie für Stapelfenster – optional eine maximale Anzahl von Sekunden für das Sammeln von Datensätzen ein, bevor Sie fortfahren.
   c. Geben Sie für Gleichzeitige Stapel pro Shard – optional die Anzahl der Stapel aus demselben Shard ein, die gleichzeitig gelesen werden können.
   d. Wählen Sie für Bei teilweisem Stapelelementfehler Folgendes aus:
      - AUTOMATIC_BISECT – Halbieren Sie jeden Stapel und wiederholen Sie jede Hälfte, bis alle Datensätze verarbeitet sind oder eine fehlgeschlagene Nachricht im Stapel übrig ist.

> **Note**
>
> Wenn Sie AUTOMATIC_BISECT nicht wählen, können Sie bestimmte fehlgeschlagene Datensätze zurückgeben und nur diese werden erneut versucht.

Kinesis

So konfigurieren Sie eine Kinesis-Quelle mithilfe der Konsole

1. Wählen Sie für Quelle Kinesis aus.
2. Wählen Sie für Kinesis-Stream den Stream aus, den Sie als Quelle verwenden möchten.
3. Wählen Sie für Startposition eine der folgenden Optionen:
   - Neueste – Starten Sie das Lesen des Streams mit dem neuesten Datensatz im Shard.
   - Horizont trimmen – Starten Sie das Lesen des Streams mit dem letzten nicht getrimmten Datensatz im Shard. Dies ist der älteste Datensatz im Shard.
   - Am Zeitstempel – Starten Sie ab einem bestimmten Zeitpunkt mit dem Lesen des Streams. Geben Sie unter Zeitstempel ein Datum und eine Uhrzeit im Format JJJJ/MM/TT und hh:mm:ss ein.
4. (Optional) Gehen Sie unter Weitere Einstellungen – optional wie folgt vor:
   a. Geben Sie für Stapelgröße – optional eine maximale Anzahl von Datensätzen für jeden Stapel ein. Der Standardwert lautet 100.
   b. (Optional) Geben Sie für Stapelfenster – optional eine maximale Anzahl von Sekunden für das Sammeln von Datensätzen ein, bevor fortgefahren wird.
   c. Geben Sie für Gleichzeitige Stapel pro Shard – optional die Anzahl der Stapel aus demselben Shard ein, die gleichzeitig gelesen werden können.
   d. Wählen Sie für Bei teilweisem Stapelelementfehler Folgendes aus:
      - AUTOMATIC_BISECT – Halbieren Sie jeden Stapel und wiederholen Sie jede Hälfte, bis alle Datensätze verarbeitet sind oder eine fehlgeschlagene Nachricht im Stapel übrig ist.

> **Note**
>
> Wenn Sie AUTOMATIC_BISECT nicht wählen, können Sie bestimmte fehlgeschlagene Datensätze zurückgeben und nur diese werden erneut versucht.

Amazon MQ

So konfigurieren Sie eine Amazon-MQ-Quelle mithilfe der Konsole

1. Wählen Sie für Quelle Amazon MQ aus.
2. Wählen Sie für Amazon-MQ-Broker den Stream aus, den Sie als Quelle verwenden möchten.
3. Geben Sie für Warteschlangenname den Namen der Warteschlange ein, aus der die Pipe lesen wird.
4. Wählen Sie für Authentifizierungsmethode BASIC_AUTH aus.
5. Wählen Sie für Geheimschlüssel den Geheimschlüssel aus.
6. (Optional) Gehen Sie unter Weitere Einstellungen – optional wie folgt vor:
   a. Geben Sie für Stapelgröße – optional eine maximale Anzahl von Nachrichten für jeden Stapel ein. Der Standardwert lautet 100.
   b. Geben Sie für Stapelfenster – optional eine maximale Anzahl von Sekunden für das Sammeln von Datensätzen ein, bevor Sie fortfahren.

Amazon MSK

So konfigurieren Sie eine Amazon-MSK-Quelle mithilfe der Konsole

1. Wählen Sie für Quelle Amazon MSK aus.
2. Wählen Sie für Amazon-MSK-Cluster den Cluster aus, den Sie verwenden möchten.
3. Geben Sie für Themenname den Namen des Themas ein, aus dem die Pipe lesen wird.
4. (Optional) Geben Sie für Konsumentengruppen-ID – optional die ID der Konsumentengruppe ein, der die Pipe beitreten soll.
5. (Optional) Aktivieren Sie für Authentifizierung – optional die Option Authentifizierung verwenden und gehen Sie wie folgt vor:
   a. Wählen Sie für Authentifizierungsmethode den gewünschten Typ aus.
   b. Wählen Sie für Geheimschlüssel den Geheimschlüssel aus.
6. (Optional) Gehen Sie unter Weitere Einstellungen – optional wie folgt vor:
   a. Geben Sie für Stapelgröße – optional eine maximale Anzahl von Datensätzen für jeden Stapel ein. Der Standardwert lautet 100.

b. Geben Sie für Stapelfenster – optional eine maximale Anzahl von Sekunden für das Sammeln von Datensätzen ein, bevor Sie fortfahren.

c. Wählen Sie für Startposition eine der folgenden Optionen:

- Neueste – Starten Sie das Lesen des Themas mit dem neuesten Datensatz im Shard.
- Horizont trimmen – Starten Sie das Lesen des Themas mit dem letzten nicht getrimmten Datensatz im Shard. Dies ist der älteste Datensatz im Shard.

> **Note**
>
> Horizont trimmen ist dasselbe wie Frühestens für Apache Kafka.

Self managed Apache Kafka

So konfigurieren Sie eine selbstverwaltete Apache-Kafka-Quelle mithilfe der Konsole

1. Wählen Sie für Quelle Selbstverwaltetes Apache Kafka aus.
2. Geben Sie für Bootstrap-Server die `host:port`-Paaradressen Ihrer Broker ein.
3. Geben Sie für Themenname den Namen des Themas ein, aus dem die Pipe lesen wird.
4. (Optional) Wählen Sie für VPC die VPC aus, die Sie möchten. Wählen Sie dann für VPC-Subnetze die gewünschten Subnetze aus. Wählen Sie für VPC-Sicherheitsgruppen die Sicherheitsgruppen aus.
5. (Optional) Aktivieren Sie für Authentifizierung – optional die Option Authentifizierung verwenden und gehen Sie wie folgt vor:
   a. Wählen Sie für Authentifizierungsmethode den Authentifizierungstyp aus.
   b. Wählen Sie für Geheimschlüssel den Geheimschlüssel aus.
6. (Optional) Gehen Sie unter Weitere Einstellungen – optional wie folgt vor:
   a. Wählen Sie für Startposition eine der folgenden Optionen:
      - Neueste – Starten Sie das Lesen des Streams mit dem neuesten Datensatz im Shard.
      - Horizont trimmen – Starten Sie das Lesen des Streams mit dem letzten nicht getrimmten Datensatz im Shard. Dies ist der älteste Datensatz im Shard.
   b. Geben Sie für Stapelgröße – optional eine maximale Anzahl von Datensätzen für jeden Stapel ein. Der Standardwert lautet 100.

c. Geben Sie für Stapelfenster – optional eine maximale Anzahl von Sekunden für das Sammeln von Datensätzen ein, bevor Sie fortfahren.

Amazon SQS

So konfigurieren Sie eine Amazon-SQS-Quelle mithilfe der Konsole

1. Wählen Sie für Quelle SQS aus.
2. Wählen Sie für SQS-Warteschlange die Warteschlange aus, die Sie verwenden möchten.
3. (Optional) Gehen Sie unter Weitere Einstellungen – optional wie folgt vor:
   a. Geben Sie für Stapelgröße – optional eine maximale Anzahl von Datensätzen für jeden Stapel ein. Der Standardwert lautet 100.
   b. Geben Sie für Stapelfenster – optional eine maximale Anzahl von Sekunden für das Sammeln von Datensätzen ein, bevor Sie fortfahren.

Konfigurieren der Ereignisfilterung (optional)

Sie können Ihrer Pipe Filter hinzufügen, sodass Sie nur eine Teilmenge der Ereignisse von Ihrer Quelle zum Ziel senden.

So konfigurieren Sie die Filterung mithilfe der Konsole

1. Wählen Sie Filtern aus.
2. Sie finden unter Beispielereignis – optional ein Beispielereignis, mit dem Sie Ihr Ereignismuster erstellen können, oder Sie können Ihr eigenes Ereignis eingeben, indem Sie Eigenes eingeben wählen.
3. Geben Sie unter Ereignismuster das Ereignismuster ein, das Sie zum Filtern der Ereignisse verwenden möchten. Weitere Informationen zur Erstellung von Filtern finden Sie unter. ???

   Im Folgenden finden Sie ein Beispiel für ein Ereignismuster, bei dem nur Ereignisse mit dem Wert Seattle im Feld Stadt gesendet werden.

```
{
  "data": {
    "City": ["Seattle"]
  }
}
```

Jetzt, da die Ereignisse gefiltert werden, können Sie eine optionale Anreicherung und ein Ziel für die Pipe hinzufügen.

Definieren der Ereignisanreicherung (optional)

Sie können die Ereignisdaten zur Anreicherung an eine Lambda-Funktion, eine AWS Step Functions Zustandsmaschine, ein Amazon API Gateway oder ein API-Ziel senden.

So wählen Sie die Anreicherung aus

1. Wählen Sie Anreicherung aus.
2. Wählen Sie unter Details für Service den Service und die zugehörigen Einstellungen aus, die Sie für die Anreicherung verwenden möchten.

Sie können die Daten auch transformieren, bevor Sie sie zur Optimierung senden.

(Optional) So definieren Sie den Eingabe-Transformator

1. Wählen Sie Eingabe-Transformator für die Anreicherung – optional.
2. Wählen Sie für Beispielereignisse/Ereignisnutzlast den Typ des Beispielereignisses aus.
3. Geben Sie für Transformator die Transformator-Syntax ein, z. B. `"Event happened at <$.detail.field>."`, wobei `<$.detail.field>` ein Verweis auf ein Feld aus dem Beispielereignis ist. Sie können auch auf ein Feld aus dem Beispielereignis doppelklicken, um es dem Transformator hinzuzufügen.
4. Stellen Sie für Ausgabe sicher, dass die Ausgabe Ihren Vorstellungen entspricht.

Nachdem die Daten nun gefiltert und optimiert wurden, müssen Sie ein Ziel definieren, an das die Ereignisdaten gesendet werden sollen.

Konfigurieren eines Ziels

So konfigurieren Sie ein Ziel

1. Wählen Sie Target aus.
2. Wählen Sie unter Details für Zielservice das Ziel aus. Welche Felder angezeigt werden, hängt vom ausgewählten Ziel ab. Geben Sie nach Bedarf Informationen ein, die für diesen Zieltyp spezifisch sind.

Sie können die Daten auch transformieren, bevor Sie sie an das Ziel senden.

(Optional) So definieren Sie den Eingabe-Transformator

1. Wählen Sie Eingabe-Transformator für das Ziel – optional aus.
2. Wählen Sie für Beispielereignisse/Ereignisnutzlast den Typ des Beispielereignisses aus.
3. Geben Sie für Transformator die Transformator-Syntax ein, z. B. `"Event happened at <$.detail.field>."`, wobei `<$.detail.field>` ein Verweis auf ein Feld aus dem Beispielereignis ist. Sie können auch auf ein Feld aus dem Beispielereignis doppelklicken, um es dem Transformator hinzuzufügen.
4. Stellen Sie für Ausgabe sicher, dass die Ausgabe Ihren Vorstellungen entspricht.

Nachdem die Pipe konfiguriert ist, stellen Sie sicher, dass ihre Einstellungen korrekt konfiguriert sind.

Konfigurieren der Pipe-Einstellungen

Eine Pipe ist standardmäßig aktiv, aber Sie können sie deaktivieren. Sie können auch die Berechtigungen der Pipe angeben, die Pipe-Protokollierung einrichten und Tags hinzufügen.

So konfigurieren Sie die Pipe-Einstellungen

1. Wählen Sie die Registerkarte Pipe-Einstellungen aus.
2. Standardmäßig sind neu erstellte Pipes aktiv, sobald sie erstellt wurden. Wenn Sie eine inaktive Pipe erstellen möchten, deaktivieren Sie unter Aktivierung für Pipe aktivieren die Option Aktiv.
3. Führen Sie unter Berechtigungen für Ausführungsrolle einen der folgenden Schritte aus:

    a. Um eine neue Ausführungsrolle für diese Pipe EventBridge erstellen zu lassen, wählen Sie Create a new role for this specific resource. Sie können unter Rollenname optional den Rollennamen bearbeiten.

    b. Wählen Sie Vorhandene Rolle verwenden aus, wenn Sie eine vorhandene Ausführungsrolle verwenden möchten. Wählen Sie unter Rollenname die Rolle aus.
4. (Optional) Wenn Sie einen DynamoDB Stream Kinesis oder als Pipe-Quelle angegeben haben, können Sie eine Wiederholungsrichtlinie und eine Dead-Letter-Queue (DLQ) konfigurieren.

    Gehen Sie für Wiederholungsrichtlinie und Warteschlange für unzustellbare Nachrichten – optional wie folgt vor:

    Gehen Sie unter Wiederholungsrichtlinie wie folgt vor:

a. Wenn Sie Wiederholungsrichtlinien aktivieren möchten, aktivieren Sie Wiederholen. Standardmäßig ist für neu erstellte Pipes keine Wiederholungsrichtlinie aktiviert.

b. Geben Sie für Maximum age of event (Maximales Alter des Ereignisses) einen Wert zwischen einer Minute (00:01) und 24 Stunden (24:00) ein.

c. Geben Sie für Wiederholungsversuche eine Zahl zwischen 0 und 185 ein.

d. Wenn Sie eine Warteschlange für unzustellbare Nachrichten verwenden möchten, aktivieren Sie Warteschlange für unzustellbare Nachrichten, wählen Sie die Methode Ihrer Wahl und wählen Sie die Warteschlange oder das Thema aus, das Sie verwenden möchten. Standardmäßig verwenden neu erstellte Pipes keine Warteschlangen für unzustellbare Nachrichten.

5. (Optional) Sie können unter Protokolle – optional festlegen, wie EventBridge Pipes Protokollierungsinformationen an unterstützte Services sendet, einschließlich der Konfiguration dieser Protokolle.

   Weitere Informationen zur Protokollierung von Pipe-Datensätzen finden Sie unter ???.

   CloudWatch logs ist standardmäßig als Protokollziel ausgewählt, ebenso wie die Protokollebene. `ERROR` Daher erstellt EventBridge Pipes standardmäßig eine neue CloudWatch Protokollgruppe, an die Protokolldatensätze gesendet werden, die den `ERROR` Detaillierungsgrad enthalten.

   Gehen Sie wie folgt vor, damit EventBridge Pipes Protokolldatensätze an eines der unterstützten Protokollziele sendet:

   a. Wählen Sie unter Protokolle – optional die Ziele aus, an die Protokolldatensätze gesendet werden sollen.

   b. Wählen Sie unter Protokollebene die Informationsebene aus, die in EventBridge die Protokolldatensätze aufgenommen werden soll. Die `ERROR`-Protokollebene ist standardmäßig ausgewählt.

      Weitere Informationen finden Sie unter ???.

   c. Wählen Sie Ausführungsdaten einbeziehen aus, wenn Sie Informationen EventBridge zur Ereignisnutzlast sowie Informationen zu Serviceanfragen und -antworten in die Protokolldatensätze aufnehmen möchten.

      Weitere Informationen finden Sie unter ???.

   d. Konfigurieren Sie jedes ausgewählte Protokollziel:

Gehen Sie bei CloudWatch Logs Protokollen unter CloudWatch Protokolle wie folgt vor:

- Wählen Sie für CloudWatch Protokollgruppe aus, ob eine neue Protokollgruppe EventBridge erstellt werden soll, oder Sie können eine vorhandene Protokollgruppe auswählen oder den ARN einer vorhandenen Protokollgruppe angeben.
- Bearbeiten Sie für neue Protokollgruppen den Namen der Protokollgruppe nach Bedarf.

CloudWatch logs ist standardmäßig ausgewählt.

Wählen Sie für Firehose Stream-Logs unter Firehose Stream-Protokoll den Firehose Stream aus.

Gehen Sie bei Amazon S3 Protokollen unter S3-Logs wie folgt vor:

- Geben Sie den Namen des Buckets ein, der als Protokollziel verwendet werden soll.
- Geben Sie die AWS Konto-ID des Bucket-Besitzers ein.
- Geben Sie einen beliebigen Präfixtext ein, der verwendet werden soll, wenn EventBridge S3-Objekte erstellt.

  Weitere Informationen finden Sie unter Organisieren von Objekten mit Präfixen im Amazon Simple Storage Service -Benutzerhandbuch.
- Wählen Sie aus, wie Sie S3-Protokolldatensätze formatieren möchten EventBridge :
  - `json`: JSON
  - `plain`: Klartext
  - `w3c`: Erweitertes W3C-Format für Protokollierungsdateien

6. (Optional) Wählen Sie unter Tags – optional die Option Neues Tag hinzufügen und geben Sie ein oder mehrere Tags für die Regel ein. Weitere Informationen finden Sie unter ???.
7. Wählen Sie Pipe erstellen aus.

Validieren von Konfigurationsparameter

EventBridge Validiert nach der Erstellung einer Pipe die folgenden Konfigurationsparameter:

- IAM-Rolle — Da die Quelle einer Pipe nach der Erstellung der Pipe nicht geändert werden kann, wird EventBridge überprüft, ob die angegebene IAM-Rolle auf die Quelle zugreifen kann.

> **Note**
>
> EventBridge führt nicht dieselbe Überprüfung für Anreicherungen oder Ziele durch, da diese nach der Erstellung der Pipe aktualisiert werden können.

- Batching — EventBridge überprüft, ob die Batchgröße der Quelle die maximale Batchgröße des Ziels nicht überschreitet. Ist dies der Fall, ist eine geringere Batchgröße EventBridge erforderlich. Wenn ein Ziel die Batchverarbeitung nicht unterstützt, können Sie außerdem die Batching-Funktion EventBridge für die Quelle nicht konfigurieren.
- Anreicherungen — EventBridge überprüft, ob die Batchgröße für API Gateway- und API-Zielanreicherungen 1 ist, da nur Batchgrößen von 1 unterstützt werden.

Starten oder Stoppen einer Pipe

Standardmäßig weist eine Pipe den Status `Running` auf und verarbeitet Ereignisse, wenn sie erstellt wird.

Wenn Sie eine Pipe mit Amazon-SQS-, Kinesis- oder DynamoDB-Quellen erstellen, kann die Pipe-Erstellung in der Regel ein oder zwei Minuten dauern.

Wenn Sie eine Pipe mit Amazon-MSK-, selbstverwalteten Apache-Kafka- oder Amazon-MQ-Quellen erstellen, kann die Erstellung von Pipes bis zu zehn Minuten dauern.

So erstellen Sie eine Pipe ohne Verarbeitung von Ereignissen mithilfe der Konsole

- Schalten Sie die Einstellung Pipe aktivieren aus.

So erstellen Sie eine Pipe ohne programmgesteuerte Verarbeitung von Ereignissen

- Setzen Sie in Ihrem API-Aufruf den Wert `DesiredState` auf `Stopped`.

So starten oder stoppen Sie eine vorhandene Pipe mithilfe der Konsole

- Aktivieren oder deaktivieren Sie auf der Registerkarte Pipes-Einstellungen unter Aktivierung für Pipe aktivieren die Option Aktiv.

So starten oder stoppen Sie eine vorhandene Pipe programmgesteuert

- Setzen Sie den `DesiredState`-Parameter in Ihrem API-Aufruf entweder auf `RUNNING` oder `STOPPED`.

Zwischen dem Zeitpunkt, an dem eine Pipe den Status `STOPPED` aufweist und dem Zeitpunkt, an dem sie keine Ereignisse mehr verarbeitet, kann es zu einer Verzögerung kommen:

- Bei Amazon-SQS- und Stream-Quellen beträgt diese Verzögerung in der Regel weniger als zwei Minuten.
- Bei Amazon-MQ- und Apache-Kafka-Quellen kann diese Verzögerung bis zu fünfzehn Minuten betragen.

Amazon EventBridge Pipes-Quellen

EventBridge Pipes empfängt Ereignisdaten aus einer Vielzahl von Quellen, wendet optionale Filter und Anreicherungen auf diese Daten an und sendet sie an ein Ziel.

Wenn eine Quelle die Reihenfolge der an EventBridge Pipes gesendeten Ereignisse erzwingt, wird diese Reihenfolge während des gesamten Prozesses bis zum Ziel beibehalten.

Die folgenden AWS Dienste können als Quellen für EventBridge Pipes angegeben werden:

- Amazon-DynamoDB-Stream
- Amazon-Kinesis-Stream
- Amazon-MQ-Broker
- Amazon-MSK-Stream
- Amazon-SQS-Warteschlange
- Apache Kafka-Stream

  Wenn Sie einen Apache Kafka-Stream als Pipe-Quelle angeben, können Sie einen Apache Kafka-Stream angeben, den Sie selbst verwalten, oder einen Stream, der von einem Drittanbieter verwaltet wird, wie zum Beispiel:

  - Confluent Cloud
  - CloudKarafka
  - Redpanda

Amazon-DynamoDB-Stream als Quelle

Sie können EventBridge Pipes verwenden, um Datensätze in einem DynamoDB-Stream zu empfangen. Sie können diese Datensätze dann optional filtern oder optimieren, bevor Sie sie zur Verarbeitung an ein Ziel senden. Es gibt spezifische Einstellungen für Amazon DynamoDB Streams, die Sie beim Einrichten der Pipe auswählen können. EventBridge Pipes behält die Reihenfolge der Datensätze aus dem Datenstrom bei, wenn diese Daten an das Ziel gesendet werden.

Important

Das Deaktivieren eines DynamoDB-Streams, der die Quelle einer Pipe ist, führt dazu, dass diese Pipe unbrauchbar wird, selbst wenn Sie den Stream dann erneut aktivieren. Dies passiert aus folgenden Gründen:

- Sie können eine Pipe, deren Quelle deaktiviert ist, nicht stoppen, starten oder aktualisieren.
- Sie können eine Pipe nach der Erstellung nicht mit einer neuen Quelle aktualisieren. Wenn Sie einen DynamoDB-Stream erneut aktivieren, wird diesem Stream ein neuer Amazon-Ressourcenname (ARN) zugewiesen und er ist nicht mehr mit Ihrer Pipe verknüpft.

Wenn Sie den DynamoDB-Stream erneut aktivieren, müssen Sie anschließend eine neue Pipe mit dem neuen ARN des Streams erstellen.

Beispielereignis

Das folgende Beispielereignis zeigt die Informationen, die von der Pipe empfangen werden. Sie können dieses Ereignis verwenden, um Ihre Ereignismuster zu erstellen und zu filtern oder um die Eingabetransformation zu definieren. Nicht alle Felder können gefiltert werden. Weitere Informationen darüber, welche Felder Sie filtern können, finden Sie unter ???.

```
[
  {
    "eventID": "1",
    "eventVersion": "1.0",
    "dynamodb": {
      "Keys": {
        "Id": {
          "N": "101"
        }
```

```
      },
      "NewImage": {
        "Message": {
          "S": "New item!"
        },
        "Id": {
          "N": "101"
        }
      },
      "StreamViewType": "NEW_AND_OLD_IMAGES",
      "SequenceNumber": "111",
      "SizeBytes": 26
    },
    "awsRegion": "us-west-2",
    "eventName": "INSERT",
    "eventSourceARN": "arn:aws:dynamodb:us-east-1:111122223333:table/EventSourceTable",
    "eventSource": "aws:dynamodb"
  },
  {
    "eventID": "2",
    "eventVersion": "1.0",
    "dynamodb": {
      "OldImage": {
        "Message": {
          "S": "New item!"
        },
        "Id": {
          "N": "101"
        }
      },
      "SequenceNumber": "222",
      "Keys": {
        "Id": {
          "N": "101"
        }
      },
      "SizeBytes": 59,
      "NewImage": {
        "Message": {
          "S": "This item has changed"
        },
        "Id": {
          "N": "101"
        }
```

```
      },
      "StreamViewType": "NEW_AND_OLD_IMAGES"
    },
    "awsRegion": "us-west-2",
    "eventName": "MODIFY",
    "eventSourceARN": "arn:aws:dynamodb:us-east-1:111122223333:table/EventSourceTable",
    "eventSource": "aws:dynamodb"
  }
]
```

Abfragen und Stapeln von Streams

EventBridge fragt Shards in Ihrem DynamoDB-Stream nach Datensätzen bei einer Basisrate von viermal pro Sekunde ab. Sind Datensätze verfügbar, verarbeitet EventBridge das Ereignis und wartet auf das Ergebnis. Ist die Verarbeitung erfolgreich, setzt EventBridge die Abrufe fort, bis es weitere Datensätze erhält.

Standardmäßig ruft EventBridge Ihre Pipe auf, sobald Datensätze verfügbar sind. Wenn der Stapel, den EventBridge aus der Quelle liest, nur einen Datensatz enthält, wird nur ein Ereignis verarbeitet. Um zu vermeiden, dass eine kleine Anzahl von Datensätzen verarbeitet wird, können Sie der Pipe mitteilen, Datensätze bis zu fünf Minuten zu puffern, indem Sie ein Stapelverarbeitungsfenster konfigurieren. Bevor die Ereignisse verarbeitet werden, liest EventBridge so lange Datensätze aus der Quelle, bis es einen vollständigen Stapel erfasst hat, das Stapelverarbeitungsfenster abläuft oder der Stapel die Nutzlastgrenze von 6 MB erreicht.

Sie können die Parallelität auch erhöhen, indem Sie mehrere Batches aus jedem Shard parallel verarbeiten. EventBridge kann bis zu 10 Stapel in jedem Shard gleichzeitig verarbeiten. Wenn Sie die Anzahl der gleichzeitigen Stapel pro Shard erhöhen, stellt EventBridge weiterhin eine Verarbeitung in der Reihenfolge auf Partitionsschlüsselebene sicher.

Konfigurieren Sie die `ParallelizationFactor`-Einstellung, um einen Shard eines Kinesis- oder DynamoDB-Datenstroms mit mehr als einer Pipe-Ausführung gleichzeitig zu verarbeiten. Sie können die Anzahl der gleichzeitigen Stapel angeben, die EventBridge von einem Shard über einen Parallelisierungsfaktor von 1 (Standard) bis 10 abfragt. Wenn `ParallelizationFactor` beispielsweise auf 2 gesetzt ist, können Sie maximal 200 gleichzeitige EventBridge-Pipe-Ausführungen haben, um 100 Kinesis-Daten-Shards zu verarbeiten. Dies hilft, den Verarbeitungsdurchsatz hochzuskalieren, wenn das Datenvolumen flüchtig ist und `IteratorAge` hoch ist. Beachten Sie, dass der Parallelisierungsfaktor nicht funktioniert, wenn Sie die Kinesis-Aggregation verwenden.

Abfrage und Startposition des Streams

Beachten Sie, dass die Stream-Quellenabfrage bei der Pipe-Erstellung und -Aktualisierung letztendlich konsistent ist.

- Bei der Pipe-Erstellung kann es mehrere Minuten dauern, bis mit der Abfrage von Ereignissen aus dem Stream begonnen wird.
- Bei Pipe-Aktualisierungen der Quellenabfragekonfiguration kann es mehrere Minuten dauern, bis die Abfrage von Ereignissen aus dem Stream gestoppt und neu gestartet wird.

Dies bedeutet, dass, wenn Sie `LATEST` als Startposition für den Stream angeben, die Pipe möglicherweise Ereignisse übersehen könnte, die bei der Pipe-Erstellung oder -Aktualisierung gesendet werden. Um sicherzustellen, dass keine Ereignisse übersehen werden, geben Sie die Stream-Startposition als `TRIM_HORIZON` an.

Melden von Batch-Elementen

Wenn EventBridge Streaming-Daten aus einer Quelle konsumiert und verarbeitet, werden standardmäßig Checkpoints auf die höchste Sequenznummer eines Stapels überprüft, aber nur, wenn der Stapel ein voller Erfolg ist. Um zu vermeiden, dass erfolgreich verarbeitete Nachrichten in einem fehlgeschlagenen Stapel erneut verarbeitet werden, können Sie die Anreicherung oder das Ziel so konfigurieren, dass ein Objekt zurückgegeben wird, das angibt, welche Nachrichten erfolgreich waren und welche fehlgeschlagen sind. Dies wird als partielle Batch-Antwort bezeichnet.

Weitere Informationen finden Sie unter ???.

Erfolgs- und Misserfolgsbedingungen

Wenn Sie eines der folgenden Elemente zurückgeben, behandelt EventBridge einen Stapel als vollständigen Erfolg:

- Eine leere `batchItemFailure`-Liste
- Eine ungültige `batchItemFailure`-Liste
- Ein leeres `EventResponse`
- Ein ungültiges `EventResponse`

Wenn Sie eines der folgenden Elemente zurückgeben, behandelt EventBridge einen Stapel als vollständigen Misserfolg:

- Eine leere Zeichenfolge `itemIdentifier`
- Ein ungültiges `itemIdentifier`
- Ein `itemIdentifier` mit einem falschen Schlüsselnamen

EventBridge wiederholt Fehler basierend auf Ihrer Wiederholungsstrategie.

Amazon-Kinesis-Stream als Quelle

Sie können EventBridge Pipes verwenden, um Datensätze in einem Kinesis-Datenstrom zu empfangen. Sie können diese Datensätze dann optional filtern oder optimieren, bevor Sie sie zur Verarbeitung an eines der verfügbaren Ziele senden. Es gibt spezifische Einstellungen für Kinesis, die Sie beim Einrichten der Pipe auswählen können. EventBridge Pipes behält die Reihenfolge der Datensätze aus dem Datenstrom bei, wenn diese Daten an das Ziel gesendet werden.

Ein Kinesis-Daten-Stream ist eine Gruppe von Shards. Jeder Shard enthält eine Sequenz von Datensätzen. Ein Konsument ist eine Anwendung, die die Daten aus einem Kinesis-Daten-Stream verarbeitet. Sie können ein EventBridge Pipe zu einem Konsumenten mit gemeinsam genutztem Durchsatz (Standard-Iterator) oder zu einem Konsumenten mit dediziertem Durchsatz mit erweitertem Rundsenden zuweisen.

Bei Standard-Iteratoren verwendet EventBridge das HTTP-Protokoll, um jeden Shard in Ihrem Kinesis-Stream nach Datensätzen abzufragen. Die Pipe teilt den Lesedurchsatz mit anderen Konsumenten des Shards.

Um die Latenz zu minimieren und den Lesedurchsatz zu maximieren, können Sie einen Daten-Stream-Konsumenten mit erweitertem Rundsenden erstellen. Stream-Konsumenten erhalten Sie eine dedizierte Verbindung für jeden Shard, der keine Auswirkungen auf andere Anwendungen hat, die aus dem Stream lesen. Der dedizierte Durchsatz ist hilfreich, wenn viele Anwendungen die gleichen Daten lesen oder wenn ein Stream mit großen Datensätzen verarbeitet wird. Kinesis überträgt Datensätze über HTTP/2 nach EventBridge. Weitere Informationen zu Kinesis-Datenströmen finden Sie unter Lesen von Daten aus Amazon Kinesis Data Streams.

Beispielereignis

Das folgende Beispielereignis zeigt die Informationen, die von der Pipe empfangen werden. Sie können dieses Ereignis verwenden, um Ihre Ereignismuster zu erstellen und zu filtern oder um die Eingabetransformation zu definieren. Nicht alle Felder können gefiltert werden. Weitere Informationen darüber, welche Felder Sie filtern können, finden Sie unter ???.

```
[
  {
    "kinesisSchemaVersion": "1.0",
    "partitionKey": "1",
    "sequenceNumber": "49590338271490256608559692538361571095921575989136588898",
    "data": "SGVsbG8sIHRoaXMgaXMgYSB0ZXN0Lg==",
    "approximateArrivalTimestamp": 1545084650.987
    "eventSource": "aws:kinesis",
    "eventVersion": "1.0",
    "eventID":
 "shardId-000000000006:49590338271490256608559692538361571095921575989136588898",
    "eventName": "aws:kinesis:record",
    "invokeIdentityArn": "arn:aws:iam::123456789012:role/lambda-role",
    "awsRegion": "us-east-2",
    "eventSourceARN": "arn:aws:kinesis:us-east-2:123456789012:stream/lambda-stream"
  },
  {
    "kinesisSchemaVersion": "1.0",
    "partitionKey": "1",
    "sequenceNumber": "49590338271490256608559692540925702759324208523137515618",
    "data": "VGhpcyBpcyBvbmx5IGEgdGVzdC4=",
    "approximateArrivalTimestamp": 1545084711.166
    "eventSource": "aws:kinesis",
    "eventVersion": "1.0",
    "eventID":
 "shardId-000000000006:49590338271490256608559692540925702759324208523137515618",
    "eventName": "aws:kinesis:record",
    "invokeIdentityArn": "arn:aws:iam::123456789012:role/lambda-role",
    "awsRegion": "us-east-2",
    "eventSourceARN": "arn:aws:kinesis:us-east-2:123456789012:stream/lambda-stream"
  }
]
```

Abfragen und Stapeln von Streams

EventBridge fragt Shards in Ihrem Kinesis-Stream nach Datensätzen bei einer Basisrate von viermal pro Sekunde ab. Sind Datensätze verfügbar, verarbeitet EventBridge das Ereignis und wartet auf das Ergebnis. Ist die Verarbeitung erfolgreich, setzt EventBridge die Abrufe fort, bis es weitere Datensätze erhält.

Standardmäßig ruft EventBridge Ihre Pipe auf, sobald Datensätze verfügbar sind. Wenn der Stapel, den EventBridge aus der Quelle liest, nur einen Datensatz enthält, wird nur ein Ereignis verarbeitet.

Um zu vermeiden, dass eine kleine Anzahl von Datensätzen verarbeitet wird, können Sie der Pipe mitteilen, Datensätze bis zu fünf Minuten zu puffern, indem Sie ein Stapelverarbeitungsfenster konfigurieren. Bevor die Ereignisse verarbeitet werden, liest EventBridge so lange Datensätze aus der Quelle, bis es einen vollständigen Stapel erfasst hat, das Stapelverarbeitungsfenster abläuft oder der Stapel die Nutzlastgrenze von 6 MB erreicht.

Sie können die Parallelität auch erhöhen, indem Sie mehrere Batches aus jedem Shard parallel verarbeiten. EventBridge kann bis zu 10 Stapel in jedem Shard gleichzeitig verarbeiten. Wenn Sie die Anzahl der gleichzeitigen Stapel pro Shard erhöhen, stellt EventBridge weiterhin eine Verarbeitung in der Reihenfolge auf Partitionsschlüsselebene sicher.

Konfigurieren Sie die `ParallelizationFactor`-Einstellung, um einen Shard eines Kinesis- oder DynamoDB-Datenstroms mit mehr als einer Pipe-Ausführung gleichzeitig zu verarbeiten. Sie können die Anzahl der gleichzeitigen Stapel angeben, die EventBridge von einem Shard über einen Parallelisierungsfaktor von 1 (Standard) bis 10 abfragt. Wenn `ParallelizationFactor` beispielsweise auf 2 gesetzt ist, können Sie maximal 200 gleichzeitige EventBridge-Pipe-Ausführungen haben, um 100 Kinesis-Daten-Shards zu verarbeiten. Dies hilft, den Verarbeitungsdurchsatz hochzuskalieren, wenn das Datenvolumen flüchtig ist und `IteratorAge` hoch ist. Beachten Sie, dass der Parallelisierungsfaktor nicht funktioniert, wenn Sie die Kinesis-Aggregation verwenden.

Abfrage und Startposition des Streams

Beachten Sie, dass die Stream-Quellenabfrage bei der Pipe-Erstellung und -Aktualisierung letztendlich konsistent ist.

- Bei der Pipe-Erstellung kann es mehrere Minuten dauern, bis mit der Abfrage von Ereignissen aus dem Stream begonnen wird.
- Bei Pipe-Aktualisierungen der Quellenabfragekonfiguration kann es mehrere Minuten dauern, bis die Abfrage von Ereignissen aus dem Stream gestoppt und neu gestartet wird.

Dies bedeutet, dass, wenn Sie `LATEST` als Startposition für den Stream angeben, die Pipe möglicherweise Ereignisse übersehen könnte, die bei der Pipe-Erstellung oder -Aktualisierung gesendet werden. Um sicherzustellen, dass keine Ereignisse übersehen werden, geben Sie die Startposition des Streams als `TRIM_HORIZON` oder `AT_TIMESTAMP` an.

Melden von Batch-Elementen

Wenn EventBridge Streaming-Daten aus einer Quelle konsumiert und verarbeitet, werden standardmäßig Checkpoints auf die höchste Sequenznummer eines Stapels überprüft, aber nur, wenn der Stapel ein voller Erfolg ist. Um zu vermeiden, dass erfolgreich verarbeitete Nachrichten in einem fehlgeschlagenen Stapel erneut verarbeitet werden, können Sie die Anreicherung oder das Ziel so konfigurieren, dass ein Objekt zurückgegeben wird, das angibt, welche Nachrichten erfolgreich waren und welche fehlgeschlagen sind. Dies wird als partielle Batch-Antwort bezeichnet.

Weitere Informationen finden Sie unter ???.

Erfolgs- und Misserfolgsbedingungen

Wenn Sie eines der folgenden Elemente zurückgeben, behandelt EventBridge einen Stapel als vollständigen Erfolg:

- Eine leere `batchItemFailure`-Liste
- Eine ungültige `batchItemFailure`-Liste
- Ein leeres `EventResponse`
- Ein ungültiges `EventResponse`

Wenn Sie eines der folgenden Elemente zurückgeben, behandelt EventBridge einen Stapel als vollständigen Misserfolg:

- Eine leere Zeichenfolge `itemIdentifier`
- Ein ungültiges `itemIdentifier`
- Ein `itemIdentifier` mit einem falschen Schlüsselnamen

EventBridge wiederholt Fehler basierend auf Ihrer Wiederholungsstrategie.

Amazon-MQ-Message-Broker als Quelle

Sie können EventBridge Pipes verwenden, um Datensätze von einem Amazon MQ MQ-Nachrichtenbroker zu empfangen. Sie können diese Datensätze dann optional filtern oder optimieren, bevor Sie sie zur Verarbeitung an eines der verfügbaren Ziele senden. Es gibt spezielle Amazon MQ MQ-Einstellungen, die Sie beim Einrichten einer Pipe auswählen können. EventBridge Pipes behält die Reihenfolge der Datensätze des Message Brokers bei, wenn diese Daten an das Ziel gesendet werden.

Amazon MQ ist ein verwalteter Message-Broker-Service für Apache ActiveMQ und RabbitMQ. Mit einem Message Broker können Software-Anwendungen und -Komponenten mithilfe unterschiedlicher Programmiersprachen, Betriebssysteme und formeller Messaging-Protokolle entweder mit Themen oder Warteschlangen als Ereignisziele kommunizieren.

Amazon MQ kann auch Amazon Elastic Compute Cloud (Amazon EC2)-Instances in Ihrem Namen verwalten, indem es ActiveMQ- oder RabbitMQ-Broker installiert. Nachdem ein Broker installiert wurde, stellt er Ihren Instances unterschiedliche Netzwerktopologien und andere Infrastrukturanforderungen zur Verfügung.

Das Amazon-MQ-Quelle hat die folgenden Konfigurationseinschränkungen:

- Kontenübergreifende Verarbeitung — unterstützt EventBridge keine kontenübergreifende Verarbeitung. Sie können nicht verwenden EventBridge , um Datensätze von einem Amazon MQ MQ-Nachrichtenbroker zu verarbeiten, der sich in einem anderen AWS Konto befindet.
- Authentifizierung — Für ActiveMQ wird nur SimpleAuthenticationPluginActiveMQ unterstützt. Für RabbitMQ wird nur der PLAIN-Authentifizierungsmechanismus unterstützt. Zur Verwaltung von Anmeldeinformationen verwenden Sie AWS Secrets Manager. Weitere Informationen zur ActiveMQ-Authentifizierung finden Sie unter Integrieren von ActiveMQ-Brokern mit LDAP im Amazon-MQ-Entwicklerhandbuch.
- Verbindungskontingent – Broker haben eine maximale Anzahl zulässiger Verbindungen für jedes Wire-Level-Protokoll. Dieses Kontingent basiert auf dem Instance-Typ des Brokers. Weitere Informationen finden Sie im Abschnitt Broker in *Kontingente in Amazon MQ* im Amazon-MQ-Entwicklerhandbuch.
- Konnektivität – Sie können Broker in einer öffentlichen oder privaten Virtual Private Cloud (VPC) erstellen. Bei privaten VPCs benötigt Ihre Pipe Zugriff auf die VPC, um Nachrichten zu empfangen.
- Ereignisziele – Es werden nur Warteschlangenziele unterstützt. Sie können jedoch ein virtuelles Thema verwenden, das sich sowohl intern als Thema als auch extern als Warteschlange verhält, wenn es mit Ihren Pipes interagiert. Weitere Informationen finden Sie unter Virtuelle Ziele auf der Apache-ActiveMQ-Website und Virtuelle Hosts auf der RabbitMQ-Website.
- Netzwerktopologie – Für ActiveMQ wird nur ein Single-Instance- oder Standby-Broker für die Pipe unterstützt. Für RabbitMQ wird nur eine Single-Instance-Broker- oder Cluster-Bereitstellung für jede Pipe unterstützt. Single-Instance-Broker benötigen einen Failover-Endpunkt. Weitere Informationen zu diesen Broker-Bereitstellungsmodi finden Sie unter Aktive MQ-Broker-Architektur und Broker-Architektur von Rabbit MQ im Amazon-MQ-Entwicklerhandbuch.
- Protokolle – Die unterstützten Protokolle hängen von der Amazon-MQ-Integration ab, die Sie verwenden.

- EventBridge Verwendet für ActiveMQ-Integrationen das OpenWire /Java Message Service (JMS) -Protokoll, um Nachrichten zu verarbeiten. Der Nachrichtenverbrauch wird in keinem anderen Protokoll unterstützt. EventBridge unterstützt nur die BytesMessageOperationen TextMessageund innerhalb des JMS-Protokolls. Weitere Informationen zum OpenWire Protokoll finden Sie OpenWireauf der Apache ActiveMQ-Website.
- EventBridge Verwendet bei RabbitMQ-Integrationen das AMQP 0-9-1-Protokoll, um Nachrichten zu verarbeiten. Es werden keine anderen Protokolle unterstützt, um Nachrichten zu verbrauchen. Weitere Informationen zur Implementierung des AMQP 0-9-1-Protokolls durch RabbitMQ finden Sie im Kompletten AMQ-0-9-1-Referenzhandbuch auf der RabbitMQ-Website.

EventBridge unterstützt automatisch die neuesten Versionen von ActiveMQ und RabbitMQ, die Amazon MQ unterstützt. Die neuesten unterstützten Versionen finden Sie in den Amazon-MQ-Versionshinweisen im Amazon-MQ-Entwicklerhandbuch.

> **Note**
>
> Amazon MQ hat standardmäßig ein wöchentliches Wartungsfenster für Broker. Während dieses Zeitfensters sind Broker nicht verfügbar. Bei Brokern ohne Standby-Modus werden Nachrichten EventBridge erst verarbeitet, wenn das Fenster endet.

Beispielereignisse

Das folgende Beispielereignis zeigt die Informationen, die von der Pipe empfangen werden. Sie können dieses Ereignis verwenden, um Ihre Ereignismuster zu erstellen und zu filtern oder um die Eingabetransformation zu definieren. Nicht alle Felder können gefiltert werden. Weitere Informationen darüber, welche Felder Sie filtern können, finden Sie unter ???.

ActiveMQ

```
[
  {
    "eventSource": "aws:amq",
    "eventSourceArn": "arn:aws:mq:us-
west-2:112556298976:broker:test:b-9bcfa592-423a-4942-879d-eb284b418fc8",
    "messageID": "ID:b-9bcfa592-423a-4942-879d-eb284b418fc8-1.mq.us-
west-2.amazonaws.com-37557-1234520418293-4:1:1:1:1",
    "messageType": "jms/text-message",
    "data": "QUJDOkFBQUE=",
```

```
    "connectionId": "myJMSCoID",
    "redelivered": false,
    "destination": {
      "physicalname": "testQueue"
    },
    "timestamp": 1598827811958,
    "brokerInTime": 1598827811958,
    "brokerOutTime": 1598827811959
  },
  {
    "eventSource": "aws:amq",
    "eventSourceArn": "arn:aws:mq:us-
west-2:112556298976:broker:test:b-9bcfa592-423a-4942-879d-eb284b418fc8",
    "messageID": "ID:b-9bcfa592-423a-4942-879d-eb284b418fc8-1.mq.us-
west-2.amazonaws.com-37557-1234520418293-4:1:1:1:1",
    "messageType": "jms/bytes-message",
    "data": "3DTOOW7crj51prgVLQaGQ82S48k=",
    "connectionId": "myJMSCoID1",
    "persistent": false,
    "destination": {
      "physicalname": "testQueue"
    },
    "timestamp": 1598827811958,
    "brokerInTime": 1598827811958,
    "brokerOutTime": 1598827811959
  }
]
```

RabbitMQ

```
[
  {
    "eventSource": "aws:rmq",
    "eventSourceArn": "arn:aws:mq:us-
west-2:111122223333:broker:pizzaBroker:b-9bcfa592-423a-4942-879d-eb284b418fc8",
    "eventSourceKey": "pizzaQueue::/",
    "basicProperties": {
      "contentType": "text/plain",
      "contentEncoding": null,
      "headers": {
        "header1": {
          "bytes": [
            118,
```

```
            97,
            108,
            117,
            101,
            49
          ]
        },
        "header2": {
          "bytes": [
            118,
            97,
            108,
            117,
            101,
            50
          ]
        },
        "numberInHeader": 10
      },
      "deliveryMode": 1,
      "priority": 34,
      "correlationId": null,
      "replyTo": null,
      "expiration": "60000",
      "messageId": null,
      "timestamp": "Jan 1, 1970, 12:33:41 AM",
      "type": null,
      "userId": "AIDACKCEVSQ6C2EXAMPLE",
      "appId": null,
      "clusterId": null,
      "bodySize": 80
    },
    "redelivered": false,
    "data": "eyJ0aW1lb3V0IjowLCJkYXRhIjoiQ1pybWYwR3c4T3Y0YnFMUXhENEUifQ=="
  }
]
```

Verbrauchergruppe

Um mit Amazon MQ zu interagieren, EventBridge erstellt eine Verbrauchergruppe, die von Ihren Amazon MQ-Brokern lesen kann. Die Verbrauchergruppe wird mit derselben ID wie die Pipe-UUID erstellt.

Bei Amazon MQ MQ-Quellen EventBridge werden Datensätze gebündelt und in einer einzigen Payload an Ihre Funktion gesendet. Um das Verhalten zu steuern, können Sie das Batch-Fenster und die Batch-Größe konfigurieren. EventBridge ruft Nachrichten ab, bis eine der folgenden Situationen eintritt:

- Die verarbeiteten Datensätze erreichen die maximale Nutzlastgröße von 6 MB.
- Das Stapelverarbeitungsfenster läuft ab.
- Die Anzahl der Datensätze hat die volle Stapelgröße erreicht.

EventBridge konvertiert Ihren Stapel in eine einzelne Nutzlast und ruft dann Ihre Funktion auf. Nachrichten werden nicht behalten oder deserialisiert. Stattdessen ruft die Verbrauchergruppe sie als ein BLOB von Bytes ab. Anschließend werden sie in eine JSON-Nutzlast base64-kodiert. Wenn die Pipe für eine der Nachrichten in einem Batch einen Fehler zurückgibt, EventBridge wiederholt sie den gesamten Nachrichtenstapel, bis die Verarbeitung erfolgreich ist oder die Nachrichten ablaufen.

Netzwerkkonfiguration

Standardmäßig werden Amazon-MQ-Broker erstellt, wobei das `PubliclyAccessible` Flag auf „false“ gesetzt ist. Nur wenn `PubliclyAccessible` auf „true“ gesetzt ist, erhält der Broker eine öffentliche IP-Adresse. Für den vollständigen Zugriff auf Ihre Pipe muss Ihr Broker entweder einen öffentlichen Endpunkt verwenden oder Zugriff auf die VPC gewähren.

Wenn Ihr Amazon MQ-Broker nicht öffentlich zugänglich ist, EventBridge muss er Zugriff auf die Amazon Virtual Private Cloud (Amazon VPC) -Ressourcen haben, die Ihrem Broker zugeordnet sind.

- Um auf die VPC Ihrer Amazon MQ-Broker zuzugreifen, EventBridge können Sie den ausgehenden Internetzugang für die Subnetze Ihrer Quelle verwenden. Für öffentliche Subnetze muss es sich um ein verwaltetes NAT-Gateway handeln. Für private Subnetze kann es sich um ein NAT-Gateway oder Ihr eigenes NAT handeln. Stellen Sie sicher, dass das NAT über eine öffentliche IP-Adresse verfügt und sich mit dem Internet verbinden kann.
- EventBridge Pipes unterstützt auch die Übertragung von Ereignissen durch AWS PrivateLink, sodass Sie Ereignisse von einer Ereignisquelle, die sich in an Amazon Virtual Private Cloud (Amazon VPC) befindet, an ein Pipes-Ziel senden können, ohne das öffentliche Internet zu nutzen. Sie können Pipes für Abfragen von Amazon Managed Streaming for Apache Kafka (Amazon MSK), selbstverwaltetem Apache Kafka und Amazon MQ Quellen verwenden, die sich in einem privaten Subnetz befinden, ohne dass Sie ein Internet-Gateway einrichten, Firewallregeln konfigurieren oder Proxyserver einrichten müssen.

Informationen zum Einrichten eines VPC-Endpunkts finden Sie unter Erstellen eines VPC-Endpunkts im AWS PrivateLink Benutzerhandbuch. Wählen Sie als Dienstnamen aus. `com.amazonaws.region.pipes-data`

Konfigurieren Sie Ihre Amazon-VPC-Sicherheitsgruppen (mindestens) mit den folgenden Regeln:

- Regeln für eingehenden Datenverkehr — Lassen Sie den gesamten Datenverkehr auf dem Amazon MQ-Broker-Port für die für Ihre Quelle angegebenen Sicherheitsgruppen zu.
- Ausgehende Regeln - Erlauben Sie allen Datenverkehr auf Port 443 für alle Ziele. Lassen Sie den gesamten Verkehr auf dem Amazon MQ-Broker-Port für die für Ihre Quelle angegebenen Sicherheitsgruppen zu.

  Zu den Broker-Ports gehören:

  - 9092 für Klartext
  - 9094 für TLS
  - 9096 für SASL
  - 9098 für IAM

> **Note**
>
> Ihre Amazon-VPC-Konfiguration ist über die Amazon-MQ-API erkennbar. Sie müssen sie während der Einrichtung nicht konfigurieren.

Thema von Amazon Managed Streaming for Apache Kafka als Quelle

Sie können EventBridge Pipes verwenden, um Datensätze von einem Amazon Managed Streaming for Apache Kafka (Amazon MSK) -Thema zu empfangen. Sie können diese Datensätze optional filtern oder optimieren, bevor Sie sie zur Verarbeitung an eines der verfügbaren Ziele senden. Es gibt Amazon MSK-spezifische Einstellungen, die Sie beim Einrichten einer Pipe auswählen können. EventBridge Pipes behält die Reihenfolge der Datensätze des Message Brokers bei, wenn diese Daten an das Ziel gesendet werden.

Amazon MSK ist ein vollständig verwalteter Service, mit dem Sie Anwendungen erstellen und ausführen können, die Apache Kafka zur Verarbeitung von Streaming-Daten nutzen. Amazon MSK vereinfacht die Einrichtung, Skalierung und Verwaltung von Clustern, auf denen Apache Kafka

ausgeführt wird. Mit Amazon MSK können Sie Ihre Anwendung für mehrere Availability Zones und für die Sicherheit mit AWS Identity and Access Management (IAM) konfigurieren. Amazon MSK unterstützt mehrere Open-Source-Versionen von Kafka.

Amazon MSK als Quelle funktioniert ähnlich wie die Verwendung von Amazon Simple Queue Service (Amazon SQS) oder Amazon Kinesis. EventBridgefragt intern nach neuen Nachrichten von der Quelle ab und ruft dann synchron das Ziel auf. EventBridge liest die Nachrichten stapelweise und stellt sie Ihrer Funktion als Event-Payload zur Verfügung. Die maximale Batchgröße ist konfigurierbar. (Der Standardwert beträgt 100 Nachrichten.)

EventBridge Unterstützt bei Quellen, die auf Apache Kafka basieren, die Verarbeitung von Steuerungsparametern wie Stapelverarbeitungsfenstern und Batchgröße.

EventBridge liest die Nachrichten sequentiell für jede Partition. Nach der EventBridge Verarbeitung jedes Batches werden die Offsets der Nachrichten in diesem Batch festgeschrieben. Wenn das Ziel der Pipe für eine der Nachrichten in einem Stapel einen Fehler zurückgibt, EventBridge wird der gesamte Nachrichtenstapel wiederholt, bis die Verarbeitung erfolgreich ist oder die Nachrichten ablaufen.

EventBridge sendet den Nachrichtenstapel für das Ereignis, wenn es das Ziel aufruft. Die Ereignisnutzlast enthält ein Array von Meldungen. Jedes Array-Element enthält Details zum Amazon-MSK-Thema und zur Partitions-ID sowie einen Zeitstempel und eine base64-codierte Nachricht.

Beispielereignisse

Das folgende Beispielereignis zeigt die Informationen, die von der Pipe empfangen werden. Sie können dieses Ereignis verwenden, um Ihre Ereignismuster zu erstellen und zu filtern oder um die Eingabetransformation zu definieren. Nicht alle Felder können gefiltert werden. Weitere Informationen darüber, welche Felder Sie filtern können, finden Sie unter ???.

```
[
  {
    "eventSource": "aws:kafka",
    "eventSourceArn": "arn:aws:kafka:sa-east-1:123456789012:cluster/
vpc-2priv-2pub/751d2973-a626-431c-9d4e-d7975eb44dd7-2",
    "eventSourceKey": "mytopic-0",
    "topic": "mytopic",
    "partition": "0",
    "offset": 15,
    "timestamp": 1545084650987,
    "timestampType": "CREATE_TIME",
```

```
    "key":"abcDEFghiJKLmnoPQRstuVWXyz1234==",
    "value":"SGVsbG8sIHRoaXMgaXMgYSB0ZXN0Lg==",
    "headers": [
      {
        "headerKey": [
          104,
          101,
          97,
          100,
          101,
          114,
          86,
          97,
          108,
          117,
          101
        ]
      }
    ]
  }
]
```

Abfrage und Startposition des Streams

Beachten Sie, dass die Stream-Quellenabfrage bei der Pipe-Erstellung und -Aktualisierung letztendlich konsistent ist.

- Bei der Pipe-Erstellung kann es mehrere Minuten dauern, bis mit der Abfrage von Ereignissen aus dem Stream begonnen wird.
- Bei Pipe-Aktualisierungen der Quellenabfragekonfiguration kann es mehrere Minuten dauern, bis die Abfrage von Ereignissen aus dem Stream gestoppt und neu gestartet wird.

Dies bedeutet, dass, wenn Sie `LATEST` als Startposition für den Stream angeben, die Pipe möglicherweise Ereignisse übersehen könnte, die bei der Pipe-Erstellung oder -Aktualisierung gesendet werden. Um sicherzustellen, dass keine Ereignisse übersehen werden, geben Sie die Stream-Startposition als `TRIM_HORIZON` an.

MSK-Cluster-Authentifizierung

EventBridge benötigt die Erlaubnis, auf den Amazon MSK-Cluster zuzugreifen, Datensätze abzurufen und andere Aufgaben auszuführen. Amazon MSK unterstützt mehrere Optionen zur

Steuerung des Client-Zugriffs auf den MSK-Cluster. Weitere Informationen darüber, welche Authentifizierungsmethode wann verwendet wird, finden Sie unter ???.

Cluster-Zugriffsoptionen

- Nicht authentifizierter Zugriff
- SASL/SCRAM-Authentifizierung
- Auf IAM-Rolle basierende Authentifizierung
- Gegenseitige TLS-Authentifizierung
- Konfigurieren des mTLS-Secrets
- Wie wählt man EventBridge einen Bootstrap-Broker

Nicht authentifizierter Zugriff

Wir empfehlen, für die Entwicklung nur nicht authentifizierten Zugriff zu verwenden. Ein nicht authentifizierter Zugriff funktioniert nur, wenn die rollenbasierte IAM-Authentifizierung für den Cluster deaktiviert ist.

SASL/SCRAM-Authentifizierung

Amazon MSK unterstützt die Authentifizierung mit Transport Layer Security (TLS)-Verschlüsselung von Simple Authentication and Security Layer/Salted Challenge Response Authentication Mechanism (SASL/SCRAM). EventBridge Um eine Verbindung zum Cluster herzustellen, speichern Sie die Authentifizierungsdaten (Anmeldedaten) geheim. AWS Secrets Manager

Weitere Informationen zur Verwendung von Secrets Manager finden Sie unter Benutzername und Passwortauthentifizierung mit AWS Secrets Manager im Entwicklerhandbuch für Amazon Managed Streaming for Apache Kafka.

Amazon MSK unterstützt die SASL/PLAIN-Authentifizierung nicht.

Auf IAM-Rolle basierende Authentifizierung

Sie können IAM verwenden, um die Identität von Clients zu authentifizieren, die sich mit dem MSK-Cluster verbinden. Wenn die IAM-Authentifizierung auf Ihrem MSK-Cluster aktiv ist und Sie kein Geheimnis für die Authentifizierung angeben, EventBridge wird standardmäßig die IAM-Authentifizierung verwendet. Verwenden Sie die IAM-Konsole oder API, um Richtlinien zu erstellen und zu implementieren, die auf IAM-Benutzern oder -Rollen basieren. Weitere Informationen finden Sie unter IAM-Zugriffskontrolle im Entwicklerhandbuch für Amazon Managed Streaming for Apache Kafka.

Damit Sie eine Verbindung EventBridge zum MSK-Cluster herstellen, Datensätze lesen und andere erforderliche Aktionen ausführen können, fügen Sie der Ausführungsrolle Ihrer Pipes die folgenden Berechtigungen hinzu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:Connect",
                "kafka-cluster:DescribeGroup",
                "kafka-cluster:AlterGroup",
                "kafka-cluster:DescribeTopic",
                "kafka-cluster:ReadData",
                "kafka-cluster:DescribeClusterDynamicConfiguration"
            ],
            "Resource": [
                "arn:aws:kafka:region:account-id:cluster/cluster-name/cluster-uuid",
                "arn:aws:kafka:region:account-id:topic/cluster-name/cluster-uuid/topic-
name",
                "arn:aws:kafka:region:account-id:group/cluster-name/cluster-
uuid/consumer-group-id"
            ]
        }
    ]
}
```

Sie können diese Berechtigungen für einen bestimmten Cluster, ein bestimmtes Thema und eine bestimmte Gruppe einteilen. Weitere Informationen finden Sie unter Amazon-MSK-Kafka-Aktionen im Entwicklerhandbuch für Amazon Managed Streaming for Apache Kafka.

Gegenseitige TLS-Authentifizierung

Gegenseitige TLS (mTLS) bietet eine bidirektionale Authentifizierung zwischen Client und Server. Der Client sendet ein Zertifikat an den Server, damit der Server den Client überprüfen kann, und der Server sendet ein Zertifikat an den Client, damit der Client den Server überprüfen kann.

Fungiert für Amazon EventBridge MSK als Client. Sie konfigurieren ein Client-Zertifikat (als Secret in Secrets Manager), um sich EventBridge bei den Brokern in Ihrem MSK-Cluster zu authentifizieren. Das Clientzertifikat muss von einer Zertifizierungsstelle (CA) im Trust Store des Servers signiert sein. Der MSK-Cluster sendet ein Serverzertifikat an, mit dem EventBridge die Broker authentifiziert

werden. EventBridge Das Serverzertifikat muss von einer Zertifizierungsstelle signiert sein, die sich im AWS Trust Store befindet.

Amazon MSK unterstützt keine selbstsignierten Serverzertifikate, da alle Broker in Amazon MSK öffentliche Zertifikate verwenden, die von Amazon Trust Services-Zertifizierungsstellen signiert wurden, die standardmäßig EventBridge vertrauen.

Weitere Informationen über mTLS für Amazon MSK finden Sie unter Gegenseitige TLS-Authentifizierung im Entwicklerhandbuch für Amazon Managed Streaming for Apache Kafka.

Konfigurieren des mTLS-Secrets

Das Secret CLIENT_CERTIFICATE_TLS_AUTH erfordert ein Zertifikatfeld und ein Feld für einen privaten Schlüssel. Für einen verschlüsselten privaten Schlüssel erfordert das Secret ein Passwort für den privaten Schlüssel. Sowohl das Zertifikat als auch der private Schlüssel müssen im PEM-Format vorliegen.

> **Note**
>
> EventBridge unterstützt die Verschlüsselungsalgorithmen mit privaten Schlüsseln von PBES1 (aber nicht von PBES2).

Das Zertifikatfeld muss eine Liste von Zertifikaten enthalten, beginnend mit dem Client-Zertifikat, gefolgt von etwaigen Zwischenzertifikaten und endend mit dem Root-Zertifikat. Jedes Zertifikat muss in einer neuen Zeile mit der folgenden Struktur beginnen:

```
-----BEGIN CERTIFICATE-----
        <certificate contents>
-----END CERTIFICATE-----
```

Secrets Manager unterstützt Secrets von bis zu 65 536 Bytes, was genügend Platz für lange Zertifikatsketten bietet.

Der private Schlüssel muss im Format PKCS #8 mit folgender Struktur vorliegen:

```
-----BEGIN PRIVATE KEY-----
         <private key contents>
-----END PRIVATE KEY-----
```

Verwenden Sie für einen verschlüsselten privaten Schlüssel die folgende Struktur:

```
-----BEGIN ENCRYPTED PRIVATE KEY-----
          <private key contents>
-----END ENCRYPTED PRIVATE KEY-----
```

Im folgenden Beispiel sehen Sie den Inhalt eines Secrets für mTLS-Authentifizierung mit einem verschlüsselten privaten Schlüssel. Fügen Sie für einen verschlüsselten privaten Schlüssel das Passwort für den privaten Schlüssel in das Secret ein.

```
{
 "privateKeyPassword": "testpassword",
 "certificate": "-----BEGIN CERTIFICATE-----
MIIE5DCCAsygAwIBAgIRAPJdwaFaNRrytHBto0j5BA0wDQYJKoZIhvcNAQELBQAw
...
j0Lh4/+1HfgyE2KlmII36dg4IMzNjAFEBZiCRoPimO40s1cRqtFHXoal0QQbIlxk
cmUuiAii9R0=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIFgjCCA2qgAwIBAgIQdjNZd6uFf9hbNC5RdfmHrzANBgkqhkiG9w0BAQsFADBb
...
rQoiowbbk5wXCheYSANQIfTZ6weQTgiCHCCbuuMKNVS95FkXm0vqVD/YpXKwA/no
c8PH3PSoAaRwMMgOSA2ALJvbRz8mpg==
-----END CERTIFICATE-----",
 "privateKey": "-----BEGIN ENCRYPTED PRIVATE KEY-----
MIIFKzBVBgkqhkiG9w0BBQ0wSDAnBgkqhkiG9w0BBQwwGgQUiAFcK5hT/X7Kjmgp
...
QrSekqF+kWzmB6nAfSzgO9IaoAaytLvNgGTckWeUkWn/V0Ck+LdGUXzAC4RxZnoQ
zp2mwJn2NYB7AZ7+imp0azDZb+8YG2aUCiyqb6PnnA==
-----END ENCRYPTED PRIVATE KEY-----"
}
```

Wie wählt man EventBridge einen Bootstrap-Broker

EventBridge wählt einen Bootstrap-Broker auf der Grundlage der in Ihrem Cluster verfügbaren Authentifizierungsmethoden aus und legt fest, ob Sie ein Geheimnis für die Authentifizierung angeben. Wenn Sie ein Geheimnis für mTLS oder SASL/SCRAM angeben, wählt EventBridge automatisch diese Authentifizierungsmethode aus. Wenn Sie kein Geheimnis angeben, EventBridge wählt die stärkste Authentifizierungsmethode, die auf Ihrem Cluster aktiv ist. Im Folgenden finden Sie die Reihenfolge der Priorität, in der ein Broker EventBridge ausgewählt wird, von der stärksten zur schwächsten Authentifizierung:

- mTLS (Geheimnis für mTLS bereitgestellt)
- SASL/SCRAM (Geheimnis für SASL/SCRAM bereitgestellt)
- SASL IAM (kein Secret angegeben und IAM-Authentifizierung ist aktiv)
- Nicht authentifiziertes TLS (kein Secret angegeben und IAM-Authentifizierung ist nicht aktiv)
- Klartext (kein Secret angegeben und sowohl IAM-Authentifizierung als auch nicht authentifiziertes TLS sind nicht aktiv)

> **Note**
>
> Wenn keine Verbindung mit dem sichersten Brokertyp hergestellt werden EventBridge kann, wird nicht versucht, eine Verbindung zu einem anderen (schwächeren) Brokertyp herzustellen. Wenn Sie einen schwächeren Brokertyp wählen möchten EventBridge , deaktivieren Sie alle stärkeren Authentifizierungsmethoden in Ihrem Cluster.

Netzwerkkonfiguration

EventBridge muss Zugriff auf die Amazon Virtual Private Cloud (Amazon VPC) -Ressourcen haben, die mit Ihrem Amazon MSK-Cluster verknüpft sind.

- Um auf die VPC Ihres Amazon MSK-Clusters zuzugreifen, EventBridge können Sie den ausgehenden Internetzugang für die Subnetze Ihrer Quelle verwenden. Für öffentliche Subnetze muss es sich um ein verwaltetes NAT-Gateway handeln. Für private Subnetze kann es sich um ein NAT-Gateway oder Ihr eigenes NAT handeln. Stellen Sie sicher, dass das NAT über eine öffentliche IP-Adresse verfügt und sich mit dem Internet verbinden kann.
- EventBridge Pipes unterstützt auch die Übertragung von Ereignissen über AWS PrivateLink, sodass Sie Ereignisse von einer Ereignisquelle, die sich in an Amazon Virtual Private Cloud (Amazon VPC) befindet, an ein Pipes-Ziel senden können, ohne das öffentliche Internet zu nutzen. Sie können Pipes für Abfragen von Amazon Managed Streaming for Apache Kafka (Amazon MSK), selbstverwaltetem Apache Kafka und Amazon MQ Quellen verwenden, die sich in einem privaten Subnetz befinden, ohne dass Sie ein Internet-Gateway einrichten, Firewallregeln konfigurieren oder Proxyserver einrichten müssen.

  Informationen zum Einrichten eines VPC-Endpunkts finden Sie unter Erstellen eines VPC-Endpunkts im AWS PrivateLink Benutzerhandbuch. Wählen Sie als Dienstnamen aus. `com.amazonaws.region.pipes-data`

Konfigurieren Sie Ihre Amazon-VPC-Sicherheitsgruppen (mindestens) mit den folgenden Regeln:

- Regeln für eingehenden Datenverkehr — Lassen Sie den gesamten Verkehr auf dem Amazon MSK-Broker-Port für die für Ihre Quelle angegebenen Sicherheitsgruppen zu.
- Ausgehende Regeln - Erlauben Sie allen Datenverkehr auf Port 443 für alle Ziele. Lassen Sie den gesamten Verkehr auf dem Amazon MSK-Broker-Port für die für Ihre Quelle angegebenen Sicherheitsgruppen zu.

  Zu den Broker-Ports gehören:

  - 9092 für Klartext
  - 9094 für TLS
  - 9096 für SASL
  - 9098 für IAM

> **Note**
>
> Ihre Amazon-VPC-Konfiguration ist über die Amazon MSK API erkennbar. Sie müssen sie während der Einrichtung nicht konfigurieren.

Anpassbare Konsumentengruppen-ID

Wenn Sie Apache Kafka als Quelle einrichten, können Sie eine Konsumentengruppen-ID angeben. Diese Konsumentengruppen-ID ist eine vorhandene Kennung für die Apache-Kafka-Konsumentengruppe, der Ihre Pipe beitreten soll. Sie können diese Funktion verwenden, um alle laufenden Einstellungen für die Verarbeitung von Apache Kafka-Datensätzen von anderen Benutzern zu migrieren. EventBridge

Wenn Sie eine Konsumentengruppen-ID angeben und sie innerhalb dieser Konsumentengruppe weitere aktive Poller gibt, verteilt Apache Kafka Nachrichten an alle Konsumenten. Mit anderen Worten, empfängt EventBridge nicht alle Nachrichten zum Apache Kafka-Thema. Wenn Sie alle Nachrichten im Thema bearbeiten EventBridge möchten, schalten Sie alle anderen Poller in dieser Nutzergruppe aus.

Wenn Sie außerdem eine Nutzergruppen-ID angeben und Apache Kafka eine gültige bestehende Nutzergruppe mit derselben ID findet, EventBridge ignoriert Apache Kafka den `StartingPosition` Parameter für Ihre Pipe. EventBridge Beginnt stattdessen mit der Verarbeitung von Datensätzen

entsprechend dem festgeschriebenen Offset der Nutzergruppe. Wenn Sie eine Nutzergruppen-ID angeben und Apache Kafka keine bestehende Nutzergruppe finden kann, EventBridge konfiguriert Apache Kafka Ihre Quelle mit der angegebenen. `StartingPosition`

Die Konsumentengruppen-ID, die Sie angeben, muss unter all Ihren Apache-Kafka-Ereignisquellen eindeutig sein. Nachdem Sie eine Pipe mit der angegebenen Konsumentengruppen-ID erstellt haben, können Sie diesen Wert nicht aktualisieren.

Auto Scaling der Amazon-MSK-Quelle

Wenn Sie zum ersten Mal eine Amazon MSK-Quelle erstellen, EventBridge weist sie einen Consumer zu, um alle Partitionen im Apache Kafka-Thema zu verarbeiten. Jeder Verbraucher hat mehrere Prozessoren, die parallel laufen, um erhöhte Workloads zu bewältigen. Darüber hinaus EventBridge wird die Anzahl der Verbraucher je nach Arbeitslast automatisch nach oben oder unten skaliert. Um die Nachrichtenreihenfolge in jeder Partition beizubehalten, ist die maximale Anzahl von Verbrauchern pro Partition im Thema ein Verbraucher pro Partition.

In Intervallen von einer Minute EventBridge wird die Consumer-Offset-Verzögerung aller Partitionen im Thema ausgewertet. Wenn die Verzögerung zu hoch ist, empfängt die Partition Nachrichten schneller, als sie verarbeiten EventBridge kann. EventBridge Fügt bei Bedarf Benutzer zum Thema hinzu oder entfernt sie aus dem Thema. Der Skalierungsprozess zum Hinzufügen oder Entfernen von Verbrauchern erfolgt innerhalb von drei Minuten nach der Bewertung.

Wenn Ihr Ziel überlastet ist, wird die Anzahl der Verbraucher EventBridge reduziert. Diese Aktion reduziert den Workload für die Pipe, indem die Anzahl der Nachrichten reduziert wird, die Verbraucher abrufen und an die Pipe senden können.

Apache Kafka streamt als Quelle

Apache Kafka ist eine Open-Source-Event-Streaming-Plattform, die Workloads wie Datenpipelines und Streaming-Analysen unterstützt. Sie können Amazon Managed Streaming for Apache Kafka (Amazon MSK) oder einen selbstverwalteten Apache Kafka-Cluster verwenden. In der AWS Terminologie bezieht sich ein selbstverwalteter Cluster auf jeden Apache Kafka-Cluster, der nicht von gehostet wird. AWS Dazu gehören sowohl Cluster, die Sie selbst verwalten, als auch solche, die von einem Drittanbieter gehostet werden, z. B. Confluent CloudCloudKarafka, oder Redpanda.

Weitere Informationen zu anderen AWS Hosting-Optionen für Ihren Cluster finden Sie unter Best Practices for Running Apache Kafka AWS on im AWS Big Data-Blog.

Apache Kafka als Quelle funktioniert ähnlich wie die Verwendung von Amazon Simple Queue Service (Amazon SQS) oder Amazon Kinesis. EventBridgefragt intern nach neuen Nachrichten von der Quelle ab und ruft dann synchron das Ziel auf. EventBridge liest die Nachrichten stapelweise und stellt sie Ihrer Funktion als Event-Payload zur Verfügung. Die maximale Batchgröße ist konfigurierbar. (Der Standardwert beträgt 100 Nachrichten.)

EventBridge Unterstützt bei Quellen, die auf Apache Kafka basieren, die Verarbeitung von Steuerungsparametern wie Stapelverarbeitungsfenstern und Batchgröße.

EventBridge sendet den Nachrichtenstapel im Event-Parameter, wenn es Ihre Pipe aufruft. Die Ereignisnutzlast enthält ein Array von Meldungen. Jedes Array-Element enthält Details zum Apache-Kafka-Thema und zur Apache-Kafka-Partitions-ID, zusammen mit einem Zeitstempel und einer base64-codierten Nachricht.

Beispielereignisse

Das folgende Beispielereignis zeigt die Informationen, die von der Pipe empfangen werden. Sie können dieses Ereignis verwenden, um Ihre Ereignismuster zu erstellen und zu filtern oder um die Eingabetransformation zu definieren. Nicht alle Felder können gefiltert werden. Weitere Informationen darüber, welche Felder Sie filtern können, finden Sie unter ???.

```
[
  {
    "eventSource": "SelfManagedKafka",
    "bootstrapServers": "b-2.demo-cluster-1.a1bcde.c1.kafka.us-
east-1.amazonaws.com:9092,b-1.demo-cluster-1.a1bcde.c1.kafka.us-
east-1.amazonaws.com:9092",
    "eventSourceKey": "mytopic-0",
    "topic": "mytopic",
    "partition": 0,
    "offset": 15,
    "timestamp": 1545084650987,
    "timestampType": "CREATE_TIME",
    "key":"abcDEFghiJKLmnoPQRstuVWXyz1234==",
    "value":"SGVsbG8sIHRoaXMgaXMgYSB0ZXN0Lg==",
    "headers": [
      {
        "headerKey": [
          104,
          101,
          97,
          100,
```

```
          101,
          114,
          86,
          97,
          108,
          117,
          101
        ]
      }
    ]
  }
]
```

Authentifizierung für Apache-Kafka-Cluster

EventBridge Pipes unterstützt mehrere Methoden zur Authentifizierung bei Ihrem selbst verwalteten Apache Kafka-Cluster. Stellen Sie sicher, dass Sie den Apache-Kafka-Cluster für die Verwendung einer der folgenden unterstützten Authentifizierungsmethoden konfigurieren. Weitere Informationen zur Sicherheit von Apache Kafka finden Sie unter Sicherheit in der Apache-Kafka-Dokumentation.

VPC-Zugriff

Wenn Sie eine selbstverwaltete Apache Kafka-Umgebung verwenden, in der nur Apache Kafka-Benutzer in Ihrer VPC Zugriff auf Ihre Apache Kafka-Broker haben, müssen Sie die Amazon Virtual Private Cloud (Amazon VPC) in der Apache Kafka-Quelle konfigurieren.

SASL/SCRAM-Authentifizierung

EventBridge Pipes unterstützt Simple Authentication und Security Layer/Salted Challenge Response Authentication Mechanism (SASL/SCRAM) -Authentifizierung mit Transport Layer Security (TLS) -Verschlüsselung. EventBridge Pipes sendet die verschlüsselten Anmeldeinformationen zur Authentifizierung beim Cluster. Weitere Informationen zur IAM-Authentifizierung finden Sie unter RFC 5802.

EventBridge Pipes unterstützt die SASL/PLAIN-Authentifizierung mit TLS-Verschlüsselung. Bei der SASL/PLAIN-Authentifizierung sendet EventBridge Pipes Anmeldeinformationen als Klartext (unverschlüsselt) an den Server.

Für die SASL-Authentifizierung speichern Sie die Anmeldeinformationen als Secret in AWS Secrets Manager.

Gegenseitige TLS-Authentifizierung

Gegenseitige TLS (mTLS) bietet eine bidirektionale Authentifizierung zwischen Client und Server. Der Client sendet ein Zertifikat an den Server, damit der Server den Client überprüfen kann, und der Server sendet ein Zertifikat an den Client, damit der Client den Server überprüfen kann.

Im selbstverwalteten Apache Kafka fungiert EventBridge Pipes als Client. Sie konfigurieren ein Client-Zertifikat (als Secret in Secrets Manager), um EventBridge Pipes bei Ihren Apache Kafka-Brokern zu authentifizieren. Das Clientzertifikat muss von einer Zertifizierungsstelle (CA) im Trust Store des Servers signiert sein.

Der Apache Kafka-Cluster sendet ein Serverzertifikat an Pipes, um die Apache Kafka-Broker mit EventBridge Pipes zu authentifizieren. EventBridge Das Serverzertifikat kann ein öffentliches CA-Zertifikat oder ein privates CA-Zertifikat/selbstsigniertes Zertifikat sein. Das öffentliche CA-Zertifikat muss von einer CA signiert werden, die sich im EventBridge Pipes Trust Store befindet. Für ein privates CA/selbstsigniertes Zertifikat konfigurieren Sie das Root-CA-Zertifikat des Servers (als Secret in Secrets Manager). EventBridge Pipes verwendet das Stammzertifikat, um die Apache Kafka-Broker zu verifizieren.

Weitere Informationen über mTLS finden Sie unter Vorstellung von gegenseitiger TLS-Authentifizierung für Amazon MSK als Quelle.

Konfigurieren des Client-Zertifikat-Secrets

Das Secret CLIENT_CERTIFICATE_TLS_AUTH erfordert ein Zertifikatfeld und ein Feld für einen privaten Schlüssel. Für einen verschlüsselten privaten Schlüssel erfordert das Secret ein Passwort für den privaten Schlüssel. Sowohl das Zertifikat als auch der private Schlüssel müssen im PEM-Format vorliegen.

> **Note**
>
> EventBridge Pipes unterstützt die Verschlüsselungsalgorithmen mit privaten Schlüsseln von PBES1 (aber nicht von PBES2).

Das Zertifikatfeld muss eine Liste von Zertifikaten enthalten, beginnend mit dem Client-Zertifikat, gefolgt von etwaigen Zwischenzertifikaten und endend mit dem Root-Zertifikat. Jedes Zertifikat muss in einer neuen Zeile mit der folgenden Struktur beginnen:

```
-----BEGIN CERTIFICATE-----
```

```
        <certificate contents>
-----END CERTIFICATE-----
```

Secrets Manager unterstützt Secrets von bis zu 65 536 Bytes, was genügend Platz für lange Zertifikatsketten bietet.

Der private Schlüssel muss im Format PKCS #8 mit folgender Struktur vorliegen:

```
-----BEGIN PRIVATE KEY-----
         <private key contents>
-----END PRIVATE KEY-----
```

Verwenden Sie für einen verschlüsselten privaten Schlüssel die folgende Struktur:

```
-----BEGIN ENCRYPTED PRIVATE KEY-----
          <private key contents>
-----END ENCRYPTED PRIVATE KEY-----
```

Im folgenden Beispiel sehen Sie den Inhalt eines Secrets für mTLS-Authentifizierung mit einem verschlüsselten privaten Schlüssel. Fügen Sie für einen verschlüsselten privaten Schlüssel das Passwort für den privaten Schlüssel in das Secret ein.

```
{
 "privateKeyPassword": "testpassword",
 "certificate": "-----BEGIN CERTIFICATE-----
MIIE5DCCAsygAwIBAgIRAPJdwaFaNRrytHBto0j5BA0wDQYJKoZIhvcNAQELBQAw
...
j0Lh4/+1HfgyE2KlmII36dg4IMzNjAFEBZiCRoPimO40s1cRqtFHXoal0QQbIlxk
cmUuiAii9R0=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIFgjCCA2qgAwIBAgIQdjNZd6uFf9hbNC5RdfmHrzANBgkqhkiG9w0BAQsFADBb
...
rQoiowbbk5wXCheYSANQIfTZ6weQTgiCHCCbuuMKNVS95FkXm0vqVD/YpXKwA/no
c8PH3PSoAaRwMMgOSA2ALJvbRz8mpg==
-----END CERTIFICATE-----",
 "privateKey": "-----BEGIN ENCRYPTED PRIVATE KEY-----
MIIFKzBVBgkqhkiG9w0BBQ0wSDAnBgkqhkiG9w0BBQwwGgQUiAFcK5hT/X7Kjmgp
...
QrSekqF+kWzmB6nAfSzgO9IaoAaytLvNgGTckWeUkWn/V0Ck+LdGUXzAC4RxZnoQ
zp2mwJn2NYB7AZ7+imp0azDZb+8YG2aUCiyqb6PnnA==
```

```
-----END ENCRYPTED PRIVATE KEY-----"
}
```

Konfigurieren des Secrets des Server-Root-CA-Zertifikats

Sie erstellen dieses Secret, wenn Ihre Apache-Kafka-Broker TLS-Verschlüsselung mit Zertifikaten verwenden, die von einer privaten Zertifizierungsstelle signiert wurden. Sie können die TLS-Verschlüsselung zur VPC-, SASL/SCRAM-, SASL/PLAIN- oder mTLS-Authentifizierung verwenden.

Das Secret des Server-Root-CA-Zertifikats erfordert ein Feld, das das Root-CA-Zertifikat des Apache-Kafka-Brokers im PEM-Format enthält. Das folgende Beispiel zeigt die Struktur des Secrets.

```
{
    "certificate": "-----BEGIN CERTIFICATE-----
  MIID7zCCAtegAwIBAgIBADANBgkqhkiG9w0BAQsFADCBmDELMAkGA1UEBhMCVVMx
  EDAOBgNVBAgTB0FyaXpvbmExEzARBgNVBAcTClNjb3R0c2RhbGUxJTAjBgNVBAoT
  HFN0YXJmaWVsZCBUZWNobm9sb2dpZXMsIEluYy4xOzA5BgNVBAMTMlN0YXJmaWVs
  ZCBTZXJ2aWNlcyBSb290IENlcnRpZmljYXRlIEF1dG...
  -----END CERTIFICATE-----"
```

Netzwerkkonfiguration

Wenn Sie eine selbstverwaltete Apache Kafka-Umgebung verwenden, die private VPC-Konnektivität verwendet, EventBridge müssen Sie Zugriff auf die Amazon Virtual Private Cloud (Amazon VPC) -Ressourcen haben, die Ihren Apache Kafka-Brokern zugeordnet sind.

- Um auf die VPC Ihres Apache Kafka-Clusters zuzugreifen, EventBridge können Sie den ausgehenden Internetzugang für die Subnetze Ihrer Quelle verwenden. Für öffentliche Subnetze muss es sich um ein verwaltetes NAT-Gateway handeln. Für private Subnetze kann es sich um ein NAT-Gateway oder Ihr eigenes NAT handeln. Stellen Sie sicher, dass das NAT über eine öffentliche IP-Adresse verfügt und sich mit dem Internet verbinden kann.
- EventBridge Pipes unterstützt auch die Übertragung von Ereignissen durch AWS PrivateLink, sodass Sie Ereignisse von einer Ereignisquelle, die sich in an Amazon Virtual Private Cloud (Amazon VPC) befindet, an ein Pipes-Ziel senden können, ohne das öffentliche Internet zu durchqueren. Sie können Pipes für Abfragen von Amazon Managed Streaming for Apache Kafka (Amazon MSK), selbstverwaltetem Apache Kafka und Amazon MQ Quellen verwenden, die sich in einem privaten Subnetz befinden, ohne dass Sie ein Internet-Gateway einrichten, Firewallregeln konfigurieren oder Proxyserver einrichten müssen.

Informationen zum Einrichten eines VPC-Endpunkts finden Sie unter Erstellen eines VPC-Endpunkts im AWS PrivateLink Benutzerhandbuch. Wählen Sie als Dienstnamen aus. `com.amazonaws.region.pipes-data`

Konfigurieren Sie Ihre Amazon-VPC-Sicherheitsgruppen (mindestens) mit den folgenden Regeln:

- Regeln für eingehenden Datenverkehr — Erlauben Sie den gesamten Datenverkehr auf dem Apache Kafka-Broker-Port für die für Ihre Quelle angegebenen Sicherheitsgruppen.
- Ausgehende Regeln - Erlauben Sie allen Datenverkehr auf Port 443 für alle Ziele. Erlauben Sie den gesamten Verkehr auf dem Apache Kafka-Broker-Port für die für Ihre Quelle angegebenen Sicherheitsgruppen.

  Zu den Broker-Ports gehören:

  - 9092 für Klartext
  - 9094 für TLS
  - 9096 für SASL
  - 9098 für IAM

Automatische Skalierung für Verbraucher mit Apache Kafka-Quellen

Wenn Sie zum ersten Mal eine Apache Kafka-Quelle erstellen, EventBridge weist sie einem Verbraucher die Verarbeitung aller Partitionen im Kafka-Thema zu. Jeder Verbraucher hat mehrere Prozessoren, die parallel laufen, um erhöhte Workloads zu bewältigen. Darüber hinaus EventBridge wird die Anzahl der Verbraucher je nach Arbeitslast automatisch nach oben oder unten skaliert. Um die Nachrichtenreihenfolge in jeder Partition beizubehalten, ist die maximale Anzahl von Verbrauchern pro Partition im Thema ein Verbraucher pro Partition.

In Intervallen von einer Minute EventBridge wird die Consumer-Offset-Verzögerung aller Partitionen im Thema ausgewertet. Wenn die Verzögerung zu hoch ist, empfängt die Partition Nachrichten schneller, als sie verarbeiten EventBridge kann. EventBridge Fügt bei Bedarf Benutzer zum Thema hinzu oder entfernt sie aus dem Thema. Der Skalierungsprozess zum Hinzufügen oder Entfernen von Verbrauchern erfolgt innerhalb von drei Minuten nach der Bewertung.

Wenn Ihr Ziel überlastet ist, wird die Anzahl der Verbraucher EventBridge reduziert. Diese Aktion reduziert die Workload für die Funktion, indem die Anzahl der Nachrichten reduziert wird, die Verbraucher abrufen und an die Funktion senden können.

Amazon Simple Queue Service als Quelle

Sie können EventBridge Pipes verwenden, um Datensätze aus einer Amazon SQS-Warteschlange zu empfangen. Sie können diese Datensätze dann optional filtern oder optimieren, bevor Sie sie zur Verarbeitung an ein verfügbares Ziel senden.

Sie können eine Pipe verwenden, um Nachrichten in einer Amazon Simple Queue Service (Amazon SQS)-Warteschlange zu verarbeiten. EventBridge Pipes unterstützen Standardwarteschlangen und First-In, First-Out (FIFO)-Warteschlangen. Mit Amazon SQS können Sie Aufgaben von einer Komponente Ihrer Anwendung auslagern, indem Sie sie an eine Warteschlange senden und asynchron verarbeiten.

EventBridge fragt die Warteschlange ab und ruft Ihre Pipe synchron mit einem Ereignis auf, das Warteschlangennachrichten enthält. EventBridge liest Nachrichten in Batches und ruft Ihre Pipe für jeden Batch einmal auf. Wenn Ihre Pipe einen Batch erfolgreich verarbeitet, EventBridge löscht seine Nachrichten aus der Warteschlange.

Standardmäßig EventBridge sammelt bis zu 10 Nachrichten gleichzeitig in Ihrer Warteschlange und sendet diesen Batch an Ihre Pipe. Um zu vermeiden, dass die Pipe mit einer kleinen Anzahl von Datensätzen aufgerufen wird, können Sie der Ereignisquelle mitteilen, Datensätze bis zu fünf Minuten zu puffern, indem Sie ein Stapelfenster konfigurieren. Bevor Sie die Pipe aufrufen, fragt EventBridge weiterhin Nachrichten aus der Amazon SQS-Standardwarteschlange ab, bis eines der folgenden Ereignisse eintritt:

- Das Stapelfenster läuft ab.
- Das Größenkontingent für die Nutzlast des Aufrufs ist erreicht.
- Die konfigurierte maximale Stapelgröße ist erreicht.

> **Note**
>
> Wenn Sie ein Batch-Fenster verwenden und Ihre Amazon SQS-Warteschlange wenig Datenverkehr enthält, EventBridge kann bis zu 20 Sekunden warten, bevor Sie Ihre Pipe aufrufen. Dies gilt auch, wenn Sie ein Stapelfenster auf weniger als 20 Sekunden festlegen. Bei FIFO-Warteschlangen enthalten Datensätze zusätzliche Attribute, die mit der Deduplizierung und Sequenzierung zusammenhängen.

Wenn einen Batch EventBridge liest, verbleiben die Nachrichten in der Warteschlange, werden aber für die Dauer des Sichtbarkeits-Timeouts der Warteschlange ausgeblendet. Wenn Ihre Pipe den Batch erfolgreich verarbeitet, EventBridge löscht die Nachrichten aus der Warteschlange. Wenn die Pipe während der Verarbeitung eines Stapels auf einen Fehler stößt, werden standardmäßig alle Nachrichten in diesem Stapel wieder in der Warteschlange sichtbar. Deshalb muss der Pipe-Code in der Lage sein, dieselbe Nachricht mehrmals ohne unbeabsichtigte Begleiterscheinungen zu verarbeiten. Sie können dieses Verhalten bei der erneuten Verarbeitung ändern, indem Sie Stapelelementfehler in die Pipe-Antwort einbeziehen. Das folgende Beispiel zeigt ein Ereignis für einen Batch von zwei Meldungen.

Beispielereignisse

Das folgende Beispielereignis zeigt die Informationen, die von der Pipe empfangen werden. Sie können dieses Ereignis verwenden, um Ihre Ereignismuster zu erstellen und zu filtern oder um die Eingabetransformation zu definieren. Nicht alle Felder können gefiltert werden. Weitere Informationen darüber, welche Felder Sie filtern können, finden Sie unter ???.

Standardwarteschlangen


```
[
  {
    "messageId": "059f36b4-87a3-44ab-83d2-661975830a7d",
    "receiptHandle": "AQEBwJnKyrHigUMZj6rYigCgxlaS3SLy0a...",
    "body": "Test message.",
    "attributes": {
      "ApproximateReceiveCount": "1",
      "SentTimestamp": "1545082649183",
      "SenderId": "AIDAIENQZJOLO23YVJ4VO",
      "ApproximateFirstReceiveTimestamp": "1545082649185"
    },
    "messageAttributes": {},
    "md5OfBody": "e4e68fb7bd0e697a0ae8f1bb342846b3",
    "eventSource": "aws:sqs",
    "eventSourceARN": "arn:aws:sqs:us-east-2:123456789012:my-queue",
    "awsRegion": "us-east-2"
  },
  {
    "messageId": "2e1424d4-f796-459a-8184-9c92662be6da",
    "receiptHandle": "AQEBzWwaftRI0KuVm4tP+/7q1rGgNqicHq...",
    "body": "Test message.",
    "attributes": {
      "ApproximateReceiveCount": "1",
```

```
      "SentTimestamp": "1545082650636",
      "SenderId": "AIDAIENQZJOLO23YVJ4VO",
      "ApproximateFirstReceiveTimestamp": "1545082650649"
    },
    "messageAttributes": {},
    "md5OfBody": "e4e68fb7bd0e697a0ae8f1bb342846b3",
    "eventSource": "aws:sqs",
    "eventSourceARN": "arn:aws:sqs:us-east-2:123456789012:my-queue",
    "awsRegion": "us-east-2"
  }
]
```

FIFO-Warteschlange

```
[
  {
    "messageId": "11d6ee51-4cc7-4302-9e22-7cd8afdaadf5",
    "receiptHandle": "AQEBBX8nesZEXmkhsmZeyIE8iQAMig7qw...",
    "body": "Test message.",
    "attributes": {
      "ApproximateReceiveCount": "1",
      "SentTimestamp": "1573251510774",
      "SequenceNumber": "18849496460467696128",
      "MessageGroupId": "1",
      "SenderId": "AIDAIO23YVJENQZJOL4VO",
      "MessageDeduplicationId": "1",
      "ApproximateFirstReceiveTimestamp": "1573251510774"
    },
    "messageAttributes": {},
    "md5OfBody": "e4e68fb7bd0e697a0ae8f1bb342846b3",
    "eventSource": "aws:sqs",
    "eventSourceARN": "arn:aws:sqs:us-east-2:123456789012:fifo.fifo",
    "awsRegion": "us-east-2"
  }
]
```

Skalierung und Verarbeitung

Bei Standardwarteschlangen EventBridge verwendet Langabfragen, um eine Warteschlange abzufragen, bis sie aktiv wird. Wenn Nachrichten verfügbar sind, EventBridge liest bis zu fünf Stapel und sendet sie an Ihre Pipe. Wenn weiterhin Nachrichten verfügbar sind, EventBridge erhöht die

Anzahl der Prozesse, die Stapel lesen, um bis zu 300 weitere Instances pro Minute. Die maximale Anzahl von Stapeln, die eine Pipe gleichzeitig verarbeiten kann, ist 1 000.

Bei FIFO-Warteschlangen EventBridge sendet Nachrichten in der Reihenfolge, in der sie empfangen werden, an Ihre Pipe. Wenn Sie eine Nachricht an eine FIFO-Warteschlange senden, geben Sie eine Nachrichtengruppen-ID an. Amazon SQS erleichtert die Zustellung von Nachrichten in derselben Gruppe an EventBridge, der Reihe nach. EventBridge sortiert die empfangenen Nachrichten in Gruppen und sendet jeweils nur einen Stapel für eine Gruppe. Wenn Ihre Pipe einen Fehler zurückgibt, versucht die Pipe alle Wiederholungsversuche für die betroffenen Nachrichten, bevor zusätzliche Nachrichten von derselben Gruppe EventBridge erhält.

Konfigurieren einer Warteschlange für die Verwendung mit EventBridge Pipes

Erstellen Sie eine Amazon-SQS-Warteschlange als Quelle für die Pipe. Konfigurieren Sie dann die Warteschlange so, dass Ihre Pipe Zeit hat, jeden Stapel von Ereignissen zu verarbeiten – und es erneut versuchen EventBridge kann, wenn Drosselungsfehler beim Hochskalieren auftreten.

Um der Pipe Zeit zum Verarbeiten der einzelnen Stapel von Datensätzen einzuräumen, legen Sie die Zeitbeschränkung für die Sichtbarkeit der Quellwarteschlange auf mindestens den sechsfachen Wert der kombinierten Laufzeit der Pipe-Anreicherung und der Zielkomponenten fest. Die zusätzliche Zeit ermöglicht es , es erneut EventBridge zu versuchen, wenn Ihre Pipe während der Verarbeitung eines vorherigen Batches gedrosselt wird.

Wenn die Pipe eine Nachricht mehrmals nicht verarbeiten kann, kann Amazon SQS sie an eine Warteschlange für unzustellbare Nachrichten senden. Wenn Ihre Pipe einen Fehler zurückgibt, EventBridge behält sie in der Warteschlange bei. Wenn die Zeitbeschränkung für die Sichtbarkeit auftritt, empfängt EventBridge die Nachricht erneut. Um Nachrichten nach einer Anzahl von Empfangsvorgängen an eine zweite Warteschlange zu senden, konfigurieren Sie eine Warteschlange für den Posteingang in der Quellwarteschlange.

> **Note**
>
> Achten Sie darauf, die Warteschlange für unzustellbare Nachrichten für die Quellwarteschlange und nicht für die Pipe zu konfigurieren. Die Warteschlange für unzustellbare Nachrichten, die Sie für eine Pipe konfigurieren, wird für die Warteschlange asynchroner Aufrufe der Pipe und nicht für Quellwarteschlangen verwendet.

Wenn die Pipe einen Fehler zurückgibt oder nicht aufgerufen werden kann, da ihre maximale Gleichzeitigkeit erreicht wurde, kann die Verarbeitung mit zusätzlichen Versuchen erfolgreich sein. Damit Nachrichten mehr Chancen zur Verarbeitung haben, bevor sie an die Warteschlange für unzustellbare Nachrichten gesendet werden, stellen Sie `maxReceiveCount` für die Redrive-Richtlinie der Quellwarteschlange auf mindestens 5 ein.

Melden von Batch-Elementen

Wenn Streaming-Daten aus einer -Quelle EventBridge verwendet und verarbeitet, werden standardmäßig Checkpoints auf die höchste Sequenznummer eines Batches überprüft, jedoch nur, wenn der Batch ein voller Erfolg ist. Um zu vermeiden, dass erfolgreich verarbeitete Nachrichten in einem fehlgeschlagenen Stapel erneut verarbeitet werden, können Sie die Anreicherung oder das Ziel so konfigurieren, dass ein Objekt zurückgegeben wird, das angibt, welche Nachrichten erfolgreich waren und welche fehlgeschlagen sind. Dies wird als partielle Batch-Antwort bezeichnet.

Weitere Informationen finden Sie unter ???.

Erfolgs- und Misserfolgsbedingungen

Wenn Sie eine der folgenden Optionen zurückgeben, EventBridge führt einen Batch als vollständigen Erfolg aus:

- Eine leere `batchItemFailure`-Liste
- Eine ungültige `batchItemFailure`-Liste
- Ein leeres `EventResponse`
- Ein ungültiges `EventResponse`

Wenn Sie eine der folgenden Optionen zurückgeben, EventBridge führt einen Batch als vollständigen Fehler aus:

- Eine leere Zeichenfolge `itemIdentifier`
- Ein ungültiges `itemIdentifier`
- Ein `itemIdentifier` mit einem falschen Schlüsselnamen

EventBridge wiederholt Fehler basierend auf Ihrer Wiederholungsstrategie.

Filterung durch Amazon EventBridge Pipes

Mit EventBridge Pipes können Sie die Ereignisse einer bestimmten Quelle filtern und nur eine Teilmenge davon verarbeiten. Diese Filterung funktioniert genauso wie die Filterung auf einer EventBridge Eventbus- oder Lambda-Ereignisquellenzuordnung, indem sie Ereignismuster verwendet. Weitere Informationen zu Ereignismustern finden Sie unter ???.

Ein Filterkriterienobjekt `FilterCriteria` ist eine Struktur, die aus einer Liste von Filtern (`Filters`) besteht. Jeder Filter ist eine Struktur, die ein Filtermuster (`Pattern`) definiert. Ein `Pattern` ist eine Zeichenfolgendarstellung einer JSON-Filterregel. Ein `FilterCriteria`-Objekt sieht z. B. folgendermaßen aus:

```
{
  "Filters": [
    {"Pattern": "{ \"Metadata1\": [ rule1 ], \"data\": { \"Data1\": [ rule2 ] }}"
    }
  ]
}
```

Zur Verdeutlichung sehen Sie hier den Wert des Filter-`Pattern` in reinem JSON:

```
{
  "Metadata1": [ pattern1 ],
  "data": {"Data1": [ pattern2 ]}
}
```

Die Hauptbestandteile eines `FilterCriteria`-Objekts sind Metadateneigenschaften und Dateneigenschaften.

- Metadateneigenschaften sind die Felder des Ereignisobjekts. Im Beispiel bezieht sich `FilterCriteria.Metadata1` auf eine Metadateneigenschaft.
- Dateneigenschaften sind die Felder des Ereignistexts. Im Beispiel bezieht sich `FilterCriteria.Data1` auf eine Dateneigenschaft.

Nehmen wir zum Beispiel an, Ihr Kinesis-Stream enthält ein Ereignis wie das folgende:

```
{
  "kinesisSchemaVersion": "1.0",
```

```
  "partitionKey": "1",
  "sequenceNumber": "49590338271490256608559692538361571095921575989136588898",
  "data": {"City": "Seattle",
    "State": "WA",
    "Temperature": "46",
    "Month": "December"
  },
  "approximateArrivalTimestamp": 1545084650.987
}
```

Wenn das Ereignis durch Ihre Pipe fließt, sieht es wie folgt aus, wobei das `data`-Feld base64-kodiert ist:

```
{
  "kinesisSchemaVersion": "1.0",
  "partitionKey": "1",
  "sequenceNumber": "49590338271490256608559692538361571095921575989136588898",
  "data": "SGVsbG8sIHRoaXMgaXMgYSB0ZXN0Lg==",
  "approximateArrivalTimestamp": 1545084650.987,
  "eventSource": "aws:kinesis",
  "eventVersion": "1.0",
  "eventID":
 "shardId-000000000006:49590338271490256608559692538361571095921575989136588898",
  "eventName": "aws:kinesis:record",
  "invokeIdentityArn": "arn:aws:iam::123456789012:role/lambda-role",
  "awsRegion": "us-east-2",
  "eventSourceARN": "arn:aws:kinesis:us-east-2:123456789012:stream/lambda-stream"
}
```

Bei den Metadateneigenschaften des Kinesis-Ereignisses handelt es sich um beliebige Felder außerhalb des `data`-Objekts, z. B. `partitionKey` oder `sequenceNumber`.

Bei den Dateneigenschaften des Kinesis-Ereignisses handelt es sich um die Felder innerhalb des `data`-Objekts, z. B. `City` oder `Temperature`.

Wenn Sie filtern, um dieses Ereignis abzugleichen, können Sie Filter für die dekodierten Felder verwenden. Um beispielsweise nach `partitionKey` und `City` zu filtern, würden Sie den folgenden Filter verwenden:

```
{
  "partitionKey": [
```

```
    "1"
  ],
  "data": {
    "City": [
      "Seattle"
    ]
  }
}
```

Wenn Sie Ereignisfilter erstellen, können EventBridge Pipes auf Ereignisinhalte zugreifen. Dieser Inhalt ist entweder JSON-maskiert, wie das Amazon-SQS-Feld `body`, oder base64-kodiert, wie das Kinesis-Feld `data`. Wenn es sich bei Ihren Daten um gültiges JSON handelt, können Ihre Eingabevorlagen oder JSON-Pfade für Zielparameter direkt auf den Inhalt verweisen. Wenn es sich bei einer Kinesis-Ereignisquelle beispielsweise um ein gültiges JSON handelt, können Sie mithilfe von `<$.data.someKey>` auf eine Variable verweisen.

Bei der Erstellung von Ereignismustern können Sie nach den von der Quell-API gesendeten Feldern filtern und nicht nach Feldern, die durch die Abfrageoperation hinzugefügt wurden. Die folgenden Felder können nicht in Ereignismustern verwendet werden:

- `awsRegion`
- `eventSource`
- `eventSourceARN`
- `eventVersion`
- `eventID`
- `eventName`
- `invokeIdentityArn`
- `eventSourceKey`

Nachrichten- und Datenfelder

Jede EventBridge Pipe-Quelle enthält ein Feld, das die Kernbotschaft oder die Kerndaten enthält. Wir bezeichnen diese als Nachrichtenfelder oder Datenfelder. Diese Felder sind besonders, weil sie JSON-maskiert oder base64-kodiert sein können, aber wenn sie gültiges JSON sind, können sie mit JSON-Mustern gefiltert werden, als ob der Text nicht maskiert wäre. Der Inhalt dieser Felder kann auch problemlos in Eingabe-Transformatoren verwendet werden.

Ordnungsgemäßes Filtern von Amazon-SQS-Nachrichten

Wenn eine Amazon SQS SQS-Nachricht Ihre Filterkriterien nicht erfüllt, EventBridge wird die Nachricht automatisch aus der Warteschlange entfernt. Sie müssen diese Nachrichten in Amazon SQS nicht manuell löschen.

Für Amazon SQS kann der Nachrichten-`body` eine beliebige Zeichenfolge sein. Dies kann jedoch problematisch sein, wenn ihre `FilterCriteria` erwarten, dass `body` ein gültiges JSON-Format hat. Umgekehrt gilt dasselbe – wenn der `body` der eingehenden Nachricht ein gültiges JSON-Format aufweist, die Filterkriterien jedoch erwarten, dass `body` eine einfache Zeichenfolge ist, führt dies zu unbeabsichtigtem Verhalten.

Stellen Sie zur Vermeidung dieses Problems sicher, dass das Format von `body` in Ihren `FilterCriteria` dem erwarteten Format von `body` in Nachrichten entspricht, die Sie aus Ihrer Warteschlange erhalten. Vor dem Filtern Ihrer Nachrichten EventBridge werden automatisch das Format der eingehenden Nachricht `body` und Ihr Filtermuster für ausgewertet. `body` Wenn es eine Nichtübereinstimmung gibt, wird die EventBridge Nachricht gelöscht. In der folgenden Tabelle ist diese Auswertung zusammengefasst:

body-Format der eingehenden Nachricht	**body**-Format des Filtermusters	Resultierende Aktion
Einfache Zeichenfolge	Einfache Zeichenfolge	EventBridge filtert auf der Grundlage Ihrer Filterkriterien.
Einfache Zeichenfolge	Kein Filtermuster für Dateneigenschaften	EventBridge filtert (nur für die anderen Metadateneigenschaften) basierend auf Ihren Filterkriterien.
Einfache Zeichenfolge	Gültiges JSON	EventBridge löscht die Nachricht.
Gültiges JSON	Einfache Zeichenfolge	EventBridge löscht die Nachricht.
Gültiges JSON	Kein Filtermuster für Dateneigenschaften	EventBridge filtert (nur für die anderen Metadateneigenscha

body-Format der eingehenden Nachricht	**body**-Format des Filtermus ters	Resultierende Aktion
		ften) auf der Grundlage Ihrer Filterkriterien.
Gültiges JSON	Gültiges JSON	EventBridge filtert basierend auf Ihren Filterkriterien.

Wenn Sie dies nicht `body` als Teil Ihrer angeben`FilterCriteria`, EventBridge überspringt Sie diese Prüfung.

Ordnungsgemäßes Filtern von Kinesis- und DynamoDB-Nachrichten

Nachdem Ihre Filterkriterien einen Kinesis- oder DynamoDB-Datensatz verarbeitet haben, geht der Streams-Iterator über diesen Datensatz hinaus. Wenn der Datensatz Ihre Filterkriterien nicht erfüllt, müssen Sie den Datensatz nicht manuell aus Ihrer Ereignisquelle löschen. Nach Ablauf der Aufbewahrungsfrist löschen Kinesis und DynamoDB diese alten Datensätze automatisch. Wenn Sie möchten, dass Datensätze früher gelöscht werden, lesen Sie Ändern des Zeitraums der Datenaufbewahrung.

Um Ereignisse aus Stream-Ereignisquellen ordnungsgemäß zu filtern, müssen sowohl das Datenfeld als auch Ihre Filterkriterien für das Datenfeld ein gültiges JSON-Format haben. (Bei Kinesis ist das Datenfeld `data`. Bei DynamoDB ist das Datenfeld `dynamodb`.) Wenn eines der Felder kein gültiges JSON-Format hat, wird die EventBridge Nachricht gelöscht oder eine Ausnahme ausgelöst. In der folgenden Tabelle ist das Verhalten zusammengefasst:

Format der eingehenden Daten (**`data`** oder **`dynamodb`**)	Filtermusterformat für Dateneigenschaften	Resultierende Aktion
Gültiges JSON	Gültiges JSON	EventBridge filtert basierend auf Ihren Filterkriterien.
Gültiges JSON	Kein Filtermuster für Dateneigenschaften	EventBridge filtert (nur für die anderen Metadateneigenscha ften) basierend auf Ihren Filterkriterien.

Format der eingehenden Daten (**`data`** oder **`dynamodb`**)	Filtermusterformat für Dateneigenschaften	Resultierende Aktion
Gültiges JSON	Kein JSON	EventBridge löst zum Zeitpunkt der Pipe oder Aktualisierung eine Ausnahme aus. Das Filtermuster für Dateneigenschaften muss ein gültiges JSON-Format haben.
Kein JSON	Gültiges JSON	EventBridge löscht den Datensatz.
Kein JSON	Kein Filtermuster für Dateneigenschaften	EventBridge filtert (nur für die anderen Metadateneigenschaften) auf der Grundlage Ihrer Filterkriterien.
Kein JSON	Kein JSON	EventBridge löst bei der Erstellung oder Aktualisierung der Pipe eine Ausnahme aus. Das Filtermuster für Dateneigenschaften muss ein gültiges JSON-Format haben.

Ordnungsgemäßes Filtern der Nachrichten von Amazon Managed Streaming for Apache Kafka, von selbstverwaltetem Apache Kafka und von Amazon MQ

Für Amazon-MQ-Quellen lautet das Nachrichtenfeld `data`. Für Apache-Kafka-Quellen (Amazon MSK und selbstverwaltetes Apache Kafka) gibt es zwei Nachrichtenfelder: `key` und `value`.

EventBridge löscht Nachrichten, die nicht allen im Filter enthaltenen Feldern entsprechen. Übergibt bei Apache Kafka nach erfolgreichem Aufruf der Funktion Offsets für übereinstimmende und nicht übereinstimmende Nachrichten. EventBridge EventBridge Bestätigt für Amazon MQ übereinstimmende Nachrichten nach erfolgreichem Aufruf der Funktion und bestätigt nicht zugeordnete Nachrichten, wenn sie gefiltert werden.

Apache-Kafka- und Amazon-MQ-Nachrichten müssen UTF-8-kodierte Zeichenfolgen sein (entweder einfache Zeichenfolgen oder im JSON-Format). Das liegt daran, dass Apache Kafka- und Amazon MQ-Byte-Arrays vor der Anwendung von Filterkriterien in UTF-8 EventBridge dekodiert werden. Wenn Ihre Nachrichten eine andere Kodierung verwenden, z. B. UTF-16 oder ASCII, oder wenn das Nachrichtenformat nicht dem Format entspricht, verarbeitet es nur Metadatenfilter. `FilterCriteria` EventBridge In der folgenden Tabelle ist das Verhalten zusammengefasst:

Format eingehender Nachrichten (**`data`** oder **`key`** und **`value`**)	Filtermusterformat für Nachrichteneigenschaften	Resultierende Aktion
Einfache Zeichenfolge	Einfache Zeichenfolge	EventBridge filtert auf der Grundlage Ihrer Filterkriterien.
Einfache Zeichenfolge	Kein Filtermuster für Dateneigenschaften	EventBridge filtert (nur für die anderen Metadateneigenschaften) basierend auf Ihren Filterkriterien.
Einfache Zeichenfolge	Gültiges JSON	EventBridge filtert (nur für die anderen Metadateneigenschaften) auf der Grundlage Ihrer Filterkriterien.
Gültiges JSON	Einfache Zeichenfolge	EventBridge filtert (nur für die anderen Metadateneigenschaften) auf der Grundlage Ihrer Filterkriterien.
Gültiges JSON	Kein Filtermuster für Dateneigenschaften	EventBridge filtert (nur für die anderen Metadateneigenschaften) auf der Grundlage Ihrer Filterkriterien.
Gültiges JSON	Gültiges JSON	EventBridge filtert basierend auf Ihren Filterkriterien.

Format eingehender Nachrichten (**`data`** oder **`key`** und **`value`**)	Filtermusterformat für Nachrichteneigenschaften	Resultierende Aktion
Nicht UTF-8-kodierte Zeichenfolge	JSON, einfache Zeichenfolge oder kein Muster	EventBridge filtert (nur für die anderen Metadateneigenschaften) basierend auf Ihren Filterkriterien.

Unterschiede zwischen Lambda ESM und Pipes EventBridge

Beim Filtern von Ereignissen funktionieren Lambda ESM und EventBridge Pipes im Allgemeinen auf die gleiche Weise. Der Hauptunterschied besteht darin, dass das `eventSourceKey`-Feld in ESM-Nutzlasten nicht vorhanden ist.

Anreicherung von Ereignissen in Amazon EventBridge Pipes

Mit dem Anreicherungsschritt von EventBridge Pipes können Sie die Daten aus der Quelle optimieren, bevor Sie sie an das Ziel senden. Beispielsweise erhalten Sie möglicherweise Ereignisse für Ticket erstellt, die nicht die vollständigen Ticketdaten enthalten. Mithilfe der Anreicherung können Sie eine Lambda-Funktion veranlassen, die `get-ticket`-API für die vollständigen Ticketdetails aufzurufen. Pipes können diese Informationen dann an ein Ziel senden.

Sie können die folgenden Anreicherungen konfigurieren, wenn Sie eine Pipe in EventBridge einrichten:

- API-Ziel
- Amazon API Gateway
- Lambda-Funktion
- Step Functions Zustandsautomat

> **Note**
>
> EventBridge Pipes unterstützt nur Express-Workflows als Anreicherungen.

EventBridge ruft Anreicherungen synchron auf, da es auf eine Antwort von der Anreicherung warten muss, bevor es das Ziel aufruft.

Anreicherungsantworten sind auf eine Maximalgröße von 6 MB begrenzt.

Sie können die Daten, die Sie von der Quelle erhalten, auch transformieren, bevor Sie sie zur Optimierung senden. Weitere Informationen finden Sie unter ???.

Filtern von Ereignissen mithilfe der Anreicherung

EventBridge Pipes leitet die Anreicherungsantworten direkt an das konfigurierte Ziel weiter. Dazu gehören Array-Antworten für Ziele, die Stapel unterstützen. Weitere Informationen zum Stapelverhalten finden Sie unter ???. Sie können Ihre Anreicherung auch als Filter verwenden und weniger Ereignisse weiterleiten, als von der Quelle empfangen wurden. Wenn Sie das Ziel nicht aufrufen möchten, geben Sie eine leere Antwort zurück, z. B. `""`, `{}` oder `[]`.

> **Note**
>
> Wenn Sie das Ziel mit einer leeren Nutzlast aufrufen möchten, geben Sie ein Array mit leerem JSON (`[{}]`) zurück.

Aufrufen von Anreicherungen

EventBridge ruft Anreicherungen synchron auf (der Aufruftyp ist auf `REQUEST_RESPONSE` eingestellt), da es auf eine Antwort von der Anreicherung warten muss, bevor es das Ziel aufruft.

> **Note**
>
> Für Step-Functions-Zustandsmaschinen unterstützt EventBridge nur Express-Workflows als Anreicherungen, da sie synchron aufgerufen werden können.

Ziele von Amazon EventBridge Pipes

Sie können Daten in Ihrer Pipe an ein bestimmtes Ziel senden. Sie können die folgenden Ziele konfigurieren, wenn Sie eine Pipe in einrichten EventBridge:

- API-Ziel

- API Gateway
- Stapelauftrag-Warteschlange
- CloudWatch Gruppe protokollieren
- ECS-Aufgabe
- Event Bus auf demselben Konto und derselben Region
- Firehose-Bereitstellungsdat
- Vorlage für die Inspector-Beurteilung
- Kinesis-Stream
- Lambda-Funktion (SYNC oder ASYNC)
- API-Abfragen für Redshift-Cluster-Daten
- SageMaker Pipeline
- Amazon-SNS-Thema (SNS-FIFO-Themen werden nicht unterstützt)
- Amazon-SQS-Warteschlange
- Step-Functions-Zustandsautomat
  - Express-Workflows (SYNC oder ASYNC)
  - Standard-Workflows (ASYNC)
- Timestream für LiveAnalytics Tabelle

Zielparameter

Einige Zieldienste senden die Nutzdaten des Ereignisses nicht an das Ziel, sondern behandeln das Ereignis als Auslöser für den Aufruf einer bestimmten API. EventBridge verwendet die `PipeTargetParameters`, um anzugeben, welche Informationen an diese API gesendet werden. Diese umfassen u. a. folgende:

- API-Ziele (Die an ein API-Ziel gesendeten Daten müssen der Struktur der API entsprechen. Sie müssen das `InputTemplate`-Objekt verwenden, um sicherzustellen, dass die Daten korrekt strukturiert sind. Wenn Sie die ursprüngliche Ereignisnutzlast einbeziehen möchten, verweisen Sie darauf in der `InputTemplate`.)
- API Gateway (Die an API Gateway gesendeten Daten müssen der Struktur der API entsprechen. Sie müssen das `InputTemplate`-Objekt verwenden, um sicherzustellen, dass die Daten korrekt strukturiert sind. Wenn Sie die ursprüngliche Ereignisnutzlast einbeziehen möchten, verweisen Sie darauf in der `InputTemplate`.)

- `PipeTargetRedshiftDataParameters` (API-Cluster für Amazon-Redshift-Daten)
- `PipeTargetSageMakerPipelineParameters`(Pipelines zur Erstellung von SageMaker Amazon-Runtime-Modellen)
- `PipeTargetBatchJobParameters` (AWS Batch)

> **Note**
>
> EventBridge unterstützt nicht die gesamte JSON-Pfad-Syntax und wertet sie zur Laufzeit aus. Die unterstützte Syntax umfasst:
>
> - Punktnotation (zum Beispiel `$.detail`)
> - Bindestriche
> - Unterstriche
> - Alphanumerische Zeichen
> - Array-Indizes
> - Platzhalter (*)

Dynamische Pfadparameter

EventBridge Pipes-Zielparameter unterstützen die optionale dynamische JSON-Pfadsyntax. Sie können diese Syntax verwenden, um JSON-Pfade anstelle von statischen Werten anzugeben (z. B. `$.detail.state`). Der gesamte Wert muss ein JSON-Pfad sein, nicht nur ein Teil davon. Zum Beispiel kann `RedshiftParameters.Sql` `$.detail.state` sein, aber es kann nicht `"SELECT * FROM $.detail.state"` sein. Diese Pfade werden zur Laufzeit dynamisch durch Daten aus der Ereignisnutzlast selbst am angegebenen Pfad ersetzt. Dynamische Pfadparameter können nicht auf neue oder transformierte Werte verweisen, die sich aus der Eingabetransformation ergeben. Die unterstützte Syntax für JSON-Pfade mit dynamischen Parametern ist dieselbe wie bei der Transformation von Eingaben. Weitere Informationen finden Sie unter ???.

Die dynamische Syntax kann für alle Zeichenketten- und Nicht-Enum-Felder aller EventBridge Pipes-Anreicherungs- und Zielparameter verwendet werden, mit Ausnahme von:

- `PipeTargetCloudWatchLogsParameters.LogStreamName`
- `PipeTargetEventBridgeEventBusParameters.EndpointId`
- `PipeEnrichmentHttpParameters.HeaderParameters`

- `PipeTargetHttpParameters.HeaderParameters`

Um beispielsweise das Ziel `PartitionKey` eines Pipe-Kinesis-Ziels auf einen benutzerdefinierten Schlüssel aus Ihrem Quell-Event festzulegen, legen Sie den KinesisTargetParameter fest. PartitionKeyauf:

- "`$.data.`*`someKey`*" für eine Kinesis-Quelle
- "`$.body.`*`someKey`*" für eine Amazon-SQS-Quelle

Wenn es sich bei der Ereignisnutzlast um eine gültige JSON-Zeichenfolge handelt`{"`*`someKey`*`":"`*`someValue`*`"}`, wird der Wert aus dem JSON-Pfad EventBridge extrahiert und als Zielparameter verwendet. In diesem Beispiel EventBridge würde die Kinesis `PartitionKey` auf "*`SomeValue`*" gesetzt.

Berechtigungen

EventBridge Pipes benötigt die entsprechenden Berechtigungen, um API-Aufrufe für die Ressourcen durchzuführen, die Ihnen gehören. EventBridge Pipes verwendet die IAM-Rolle, die Sie für die Pipe angeben, für Enrichment- und Target-Aufrufe mithilfe des IAM-Prinzipals. `pipes.amazonaws.com`

Aufrufen von Zielen

EventBridge hat die folgenden Möglichkeiten, ein Ziel aufzurufen:

- Synchron (Aufruftyp auf gesetzt`REQUEST_RESPONSE`) — EventBridge wartet auf eine Antwort vom Ziel, bevor der Vorgang fortgesetzt wird.
- Asynchron (Aufruftyp auf gesetzt`FIRE_AND_FORGET`) — wartet EventBridge nicht auf eine Antwort, bevor der Vorgang fortgesetzt wird.

EventBridge Ruft Ziele bei Pipes mit geordneten Quellen standardmäßig synchron auf, da eine Antwort vom Ziel erforderlich ist, bevor mit dem nächsten Ereignis fortgefahren werden kann.

Wenn eine Quelle keine Reihenfolge erzwingt, wie z. B. eine standardmäßige Amazon SQS SQS-Warteschlange, EventBridge kann ein unterstütztes Ziel synchron oder asynchron aufgerufen werden.

Mit Lambda-Funktionen und Step-Functions-Zustandsmaschinen können Sie den Aufruftyp konfigurieren.

> **Note**
>
> Für Step-Functions-Zustandsmaschinen müssen Standard-Workflows asynchron aufgerufen werden.

EventBridge Leitet die Besonderheiten des Ziels weiter

AWS Batch Job-Warteschlangen

Alle AWS Batch `submitJob` Parameter werden explizit mit konfiguriert`BatchParameters`, und wie alle Pipe-Parameter können sie dynamisch sein, indem sie einen JSON-Pfad zu Ihrer Nutzlast für eingehende Ereignisse verwenden.

CloudWatch Gruppe „Protokolle“

Unabhängig davon, ob Sie einen Eingabe-Transformator nutzen oder nicht, wird die Ereignisnutzlast als Protokollnachricht verwendet. Sie können den `Timestamp` (oder den expliziten `LogStreamName` des Ziels) über `CloudWatchLogsParameters` in `PipeTarget` festlegen. Wie bei allen Pipe-Parametern können diese Parameter bei Verwendung eines JSON-Pfads zur eingehenden Ereignisnutzlast dynamisch sein.

Amazon-ECS-Aufgabe

Alle Amazon-ECS-`runTask`-Parameter werden explizit über `EcsParameters` konfiguriert. Wie bei allen Pipe-Parametern können diese Parameter bei Verwendung eines JSON-Pfads zur eingehenden Ereignisnutzlast dynamisch sein.

Workflows für Lambda-Funktionen und Step Functions

Lambda und Step Functions haben keine Stapel-API. Zum Verarbeiten von Ereignisstapeln aus einer Pipe-Quelle wird der Stapel in ein JSON-Array konvertiert und als Eingabe an das Lambda- oder Step-Functions-Ziel übergeben. Weitere Informationen finden Sie unter ???.

Timestream für LiveAnalytics Tabelle

Bei der Angabe einer Timestream LiveAnalytics for-Tabelle als Pipe-Ziel sollten unter anderem folgende Punkte berücksichtigt werden:

- Apache Kafka-Streams (auch von Anbietern Amazon MSK oder Drittanbietern) werden derzeit nicht als Pipe-Quelle unterstützt.
- Wenn Sie einen DynamoDB Stream Kinesis oder als Pipe-Quelle angegeben haben, müssen Sie die Anzahl der Wiederholungsversuche angeben.

  Weitere Informationen finden Sie unter ???.

Stapelverarbeitung und Gleichzeitigkeit von Amazon EventBridge Pipes

Batching-Verhalten

EventBridge Pipes unterstützt Batching von der Quelle und zu Zielen, die es unterstützen. Darüber hinaus wird die Stapelverarbeitung bis zur Anreicherung für AWS Lambda und AWS Step Functions unterstützt. Da verschiedene Services unterschiedliche Ebenen der Stapelverarbeitung unterstützen, können Sie eine Pipe nicht mit einer größeren Stapelgröße konfigurieren, als das Ziel unterstützt. Amazon-Kinesis-Streamquellen unterstützen beispielsweise eine maximale Stapelgröße von 10 000 Datensätzen, Amazon Simple Queue Service unterstützt jedoch maximal 10 Nachrichten pro Stapel als Ziel. Daher kann eine Pipe von einem Kinesis-Stream zu einer Amazon-SQS-Warteschlange eine maximal konfigurierte Stapelgröße in der Quelle von 10 haben.

Wenn Sie eine Pipe mit einer Anreicherung oder einem Ziel konfigurieren, das die Stapelverarbeitung nicht unterstützt, können Sie die Stapelverarbeitung in der Quelle nicht aktivieren.

Wenn die Stapelverarbeitung in der Quelle aktiviert ist, werden Arrays von JSON-Datensätzen durch die Pipe geleitet und dann der Stapel-API einer unterstützten Anreicherung oder eines unterstützten Ziels zugeordnet. Eingabe-Transformatoren werden separat auf jeden einzelnen JSON-Datensatz im Array angewendet, nicht auf das gesamte Array. Beispiele für diese Arrays finden Sie unter ??? und wählen Sie eine bestimmte Quelle aus. Pipes verwenden die Stapel-API für die unterstützte Anreicherung oder das unterstützte Ziel, auch wenn die Stapelgröße 1 ist. Wenn die Anreicherung oder das Ziel keine Stapel-API hat, aber vollständige JSON-Nutzlasten wie Lambda und Step Functions empfängt, wird das gesamte JSON-Array in einer Anfrage gesendet. Die Anfrage wird als JSON-Array gesendet, auch wenn die Stapelgröße 1 ist.

Wenn eine Pipe für die Stapelverarbeitung an der Quelle konfiguriert ist und das Ziel die Stapelverarbeitung unterstützt, können Sie ein Array von JSON-Elementen aus Ihrer Anreicherung zurückgeben. Dieses Array kann ein kürzeres oder längeres Array als die ursprüngliche Quelle

enthalten. Wenn das Array jedoch größer als die vom Ziel unterstützte Stapelgröße ist, ruft die Pipe das Ziel nicht auf.

Unterstützte stapelbare Ziele

Ziel	Maximale Stapelgröße
CloudWatch Protokolle	10.000
EventBridge Event Bus	10
Firehose-Stream	500
Kinesis-Stream	500
Lambda-Funktion	Kundendefiniert
Step-Functions-Zustandsautomat	Kundendefiniert
Amazon SNS-Thema	10
Amazon-SQS-Warteschlange	10

Die folgenden Anreicherungen und Ziele erhalten die vollständige Ereignisnutzlast des Stapels zur Verarbeitung und sind durch die Gesamtgröße der Nutzlast des Ereignisses und nicht durch die Größe des Stapels eingeschränkt:

- Step-Functions-Zustandsautomat (262144 Zeichen)
- Lambda-Funktion (6 MB)

Teilweiser Stapelfehler

Für Amazon SQS- und Stream-Quellen wie Kinesis und DynamoDB EventBridge unterstützt Pipes die teilweise Batch-Fehlerbehandlung von Zielfehlern. Wenn das Ziel das Batching unterstützt und nur ein Teil des Batches erfolgreich ist, versucht EventBridge automatisch, den Rest der Nutzlast zu batchen. Für den am meisten up-to-date anreicherten Inhalt erfolgt dieser Wiederholungsversuch über die gesamte Pipe, einschließlich des erneuten Aufrufs jeder konfigurierten Anreicherung.

Die Behandlung teilweiser Stapelfehler bei der Anreicherung wird nicht unterstützt.

Für Lambda- und Step-Functions-Ziele können Sie auch einen Teilfehler angeben, indem Sie eine Nutzlast mit definierter Struktur aus dem Ziel zurückgeben. Dies weist auf Ereignisse hin, die erneut versucht werden müssen.

Beispiel für eine Nutzlaststruktur bei einem Teilfehler

```
{
  "batchItemFailures": [
    {
      "itemIdentifier": "id2"
    },
    {
      "itemIdentifier": "id4"
    }
]
```

In diesem Beispiel entspricht die `itemIdentifier` der ID der von Ihrem Ziel verarbeiteten Ereignisse aus ihrer ursprünglichen Quelle. Für Amazon SQS ist dies die `messageId`. Für Kinesis und DynamoDB ist dies die `eventID`. Damit EventBridge Pipes teilweise Batchfehler von den Zielen angemessen behandeln kann, müssen diese Felder in jede Array-Nutzlast aufgenommen werden, die von der Anreicherung zurückgegeben wird.

Verhalten des Durchsatzes und der Gleichzeitigkeit

Jedes Ereignis oder jeder Ereignisstapel, der von einer Pipe empfangen und zu einer Anreicherung oder einem Ziel weitergeleitet wird, wird als Pipe-Ausführung betrachtet. Eine Pipe im Status `STARTED` fragt kontinuierlich Ereignisse aus der Quelle ab und skaliert dabei je nach verfügbarem Backlog und konfigurierten Einstellungen der Stapelverarbeitung nach oben oder unten.

Informationen zu Kontingenten für gleichzeitige Pipe-Ausführungen und zur Anzahl der Pipes pro Konto und Region finden Sie unter ???.

Standardmäßig wird eine einzelne Pipe je nach Quelle auf die folgende maximale Anzahl gleichzeitiger Ausführungen skaliert:

- DynamoDB – Die gleichzeitigen Ausführungen können bis zur Anzahl des in der Pipe konfigurierten `ParallelizationFactor` multipliziert mit der Anzahl der Shards im Stream ansteigen.
- Apache Kafka – Die gleichzeitigen Ausführungen können bis zur Anzahl der Partitionen im Thema steigen, bis zu 1000.

- Kinesis – Die gleichzeitigen Ausführungen können bis zur Anzahl des in der Pipe konfigurierten `ParallelizationFactor` multipliziert mit der Anzahl der Shards im Stream ansteigen.
- Amazon MQ – 5
- Amazon SQS – 1250

Wenn Sie höhere maximale Abfragedurchsätze oder Gleichzeitigkeitslimits benötigen, wenden Sie sich an den Support.

> **Note**
>
> Die Ausführungslimits gelten nach bestem Bemühen um Sicherheit. Obwohl das Abfragen nicht unter diese Werte gedrosselt wird, kann es passieren, dass eine Pipe oder ein Konto höher als diese empfohlenen Werte gesteigert wird.

Die Pipe-Ausführungen sind auf maximal 5 Minuten begrenzt, einschließlich der Anreicherung und Zielverarbeitung. Dieses Limit kann derzeit nicht erhöht werden.

Die Gleichzeitigkeit von Pipes mit streng geordneten Quellen (wie Amazon-SQS-FIFO-Warteschlangen, Kinesis- und DynamoDB-Streams oder Apache-Kafka-Themen) wird durch die Konfiguration der Quelle weiter eingeschränkt, z. B. durch die Anzahl der Nachrichtengruppen-IDs für FIFO-Warteschlangen oder die Anzahl der Shards für Kinesis-Warteschlangen. Da die Reihenfolge innerhalb dieser Einschränkungen strikt garantiert ist, kann eine Pipe mit einer geordneten Quelle diese Gleichzeitigkeitslimits nicht überschreiten.

Amazon-EventBridge-Pipes-Eingabetransformation

Amazon EventBridge Pipes unterstützt optionale Eingabe-Transformatoren bei der Weitergabe von Daten an die Anreicherung und das Ziel. Sie können Eingabe-Transformatoren verwenden, um die Nutzlast der JSON-Ereigniseingabe so umzugestalten, dass sie den Anforderungen des Anreicherungs- oder Zielservice gerecht wird. Für Amazon API Gateway und API-Ziele passen Sie das Eingabeereignis auf diese Weise an das RESTful-Modell der API an. Eingabe-Transformatoren werden als `InputTemplate`-Parameter modelliert. Sie können Freitext, ein JSON-Pfad zur Ereignisnutzlast oder ein JSON-Objekt sein, das Inline-JSON-Pfade zur Ereignisnutzlast enthält. Zur Anreicherung stammt die Ereignisnutzlast aus der Quelle. Bei Zielen ist die Ereignisnutzlast das, was von der Anreicherung zurückgegeben wird, sofern eine solche für die Pipe konfiguriert ist. Zusätzlich

zu den servicespezifischen Daten in der Ereignisnutzlast können Sie reservierte Variablen in Ihrer `InputTemplate` verwenden, um auf Daten für die Pipe zu verweisen.

Verwenden Sie die Notation mit eckigen Klammern, um auf Elemente in einem Array zuzugreifen.

> **Note**
>
> EventBridge unterstützt nicht die gesamte JSON-Pfadsyntax und wertet sie nicht zur Laufzeit aus. Die unterstützte Syntax umfasst:
>
> - Punktnotation (zum Beispiel `$.detail`)
> - Bindestriche
> - Unterstriche
> - Alphanumerische Zeichen
> - Array-Indizes
> - Platzhalter (*)

Im Folgenden finden Sie `InputTemplate`-Beispielparameter, die auf eine Amazon-SQS-Ereignisnutzlast verweisen:

Statische Zeichenfolge

```
InputTemplate: "Hello, sender"
```

JSON-Pfad

```
InputTemplate: <$.attributes.SenderId>
```

Dynamische Zeichenfolge

```
InputTemplate: "Hello, <$.attributes.SenderId>"
```

Statisches JSON

```
InputTemplate: >
{
```

```
  "key1": "value1",
  "key2": "value2",
  "key3": "value3",
}
```

Dynamisches JSON

```
InputTemplate: >
{
  "key1": "value1"
  "key2": <$.body.key>,
  "d": <aws.pipes.event.ingestion-time>
}
```

Verwenden der Notation mit eckigen Klammern, um auf ein Element in einem Array zuzugreifen:

```
InputTemplate: >
{
  "key1": "value1"
  "key2": <$.body.Records[3]>,
  "d": <aws.pipes.event.ingestion-time>
}
```

> **Note**
>
> EventBridge ersetzt Eingabe-Transformatoren zur Laufzeit, um eine gültige JSON-Ausgabe sicherzustellen. Setzen Sie aus diesem Grund Variablen, die auf JSON-Pfadparameter verweisen, in Anführungszeichen, Variablen, die sich auf JSON-Objekte oder -Arrays beziehen, jedoch nicht in Anführungszeichen.

Reservierte Variablen

Eingabevorlagen können die folgenden reservierten Variablen verwenden:

- `<aws.pipes.pipe-arn>` – Der Amazon-Ressourcenname (ARN) für die Pipe
- `<aws.pipes.pipe-name>` – Der Name der Pipe
- `<aws.pipes.source-arn>` – Der ARN der Ereignisquelle der Pipe
- `<aws.pipes.enrichment-arn>` – Der ARN der Anreicherung der Pipe

- `<aws.pipes.target-arn>` – Der ARN des Ziels der Pipe
- `<aws.pipes.event.ingestion-time>` – Der Zeitpunkt, zu dem das Ereignis vom Eingabe-Transformator empfangen wurde Dies ist ein ISO-8601-Zeitstempel. Diese Zeit ist für den Eingabe-Transformator der Anreicherung und den Eingabe-Transformator des Ziels unterschiedlich, je nachdem, wann die Anreicherung die Verarbeitung des Ereignisses abgeschlossen hat.
- `<aws.pipes.event>` – Das Ereignis, wie es vom Eingabe-Transformator empfangen wurde

  Bei einem Eingabe-Transformator der Anreicherung ist dies das Ereignis aus der Quelle. Dies enthält die ursprüngliche Nutzlast aus der Quelle sowie zusätzliche servicespezifische Metadaten. Weitere servicespezifische Beispiele finden Sie in den Themen unter ???.

  Bei einem Eingabe-Transformator des Ziels ist dies das Ereignis, das von der Anreicherung zurückgegeben wird, sofern eine solche konfiguriert ist, ohne zusätzliche Metadaten. Daher kann es sich bei einer Nutzlast, die durch eine Anreicherung zurückgegeben wurde, um Nicht-JSON handeln. Wenn für die Pipe keine Anreicherung konfiguriert ist, ist dies das Ereignis aus der Quelle mit Metadaten.
- `<aws.pipes.event.json>` – Das Gleiche wie `aws.pipes.event`, aber die Variable hat nur dann einen Wert, wenn die ursprüngliche Nutzlast, entweder aus der Quelle oder von der Anreicherung zurückgegeben, JSON ist. Wenn die Pipe ein codiertes Feld hat, z. B. das Amazon-SQS-Feld `body` oder die Kinesis-`data`, werden diese Felder decodiert und in gültiges JSON umgewandelt. Da sie nicht maskiert ist, kann die Variable nur als Wert für ein JSON-Feld verwendet werden. Weitere Informationen finden Sie unter ???.

Beispiel für die Eingabetransformation

Im Folgenden finden Sie ein Beispiel für ein Amazon-EC2-Ereignis, das wir als Beispielereignis verwenden können.

```
{
  "version": "0",
  "id": "7bf73129-1428-4cd3-a780-95db273d1602",
  "detail-type": "EC2 Instance State-change Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2015-11-11T21:29:54Z",
  "region": "us-east-1",
  "resources": [
```

```
    "arn:aws:ec2:us-east-1:123456789012:instance/i-abcd1111"
  ],
  "detail": {
    "instance-id": "i-0123456789",
    "state": "RUNNING"
  }
}
```

Lassen Sie uns den folgenden JSON als unseren Transformator verwenden.

```
{
  "instance" : <$.detail.instance-id>,
  "state": <$.detail.state>,
  "pipeArn" : <aws.pipes.pipe-arn>,
  "pipeName" : <aws.pipes.pipe-name>,
  "originalEvent" : <aws.pipes.event.json>
}
```

Die resultierende Ausgabe sieht wie folgt aus:

```
{
  "instance" : "i-0123456789",
  "state": "RUNNING",
  "pipeArn" : "arn:aws:pipe:us-east-1:123456789012:pipe/example",
  "pipeName" : "example",
  "originalEvent" : {
    ... // commented for brevity
  }
}
```

Implizites Textdatenparsen

Die folgenden Felder in der eingehenden Nutzlast können JSON-maskiert, wie das Amazon-SQS-Objekt `body`, oder base64-kodiert sein, wie das Kinesis-Objekt `data`. Sowohl für die Filterung als auch für die Eingabetransformation wandelt EventBridge diese Felder in gültiges JSON um, sodass Unterwerte direkt referenziert werden können. Zum Beispiel <$.data.someKey> für Kinesis.

Damit das Ziel die ursprüngliche Nutzlast ohne zusätzliche Metadaten erhält, verwenden Sie einen Eingabe-Transformator mit diesen Textdaten, die für die Quelle spezifisch sind. Zum Beispiel <$.body> für Amazon SQS oder <$.data> für Kinesis. Wenn die ursprüngliche Nutzlast eine

gültige JSON-Zeichenfolge ist (zum Beispiel `{"key": "value"}`), führt die Verwendung des Eingabe-Transformators mit quellspezifischen Textdaten dazu, dass die Anführungszeichen innerhalb der ursprünglichen Quellnutzlast entfernt werden. Zum Beispiel wird `{"key": "value"}` zu `"{key: value}"`, wenn es an das Ziel übermittelt wird. Wenn das Ziel gültige JSON-Nutzlasten benötigt (z. B. EventBridge Lambda oder Step Functions), führt dies zu einer fehlgeschlagenen Übermittlung. Damit das Ziel die ursprünglichen Quelldaten empfängt, ohne ungültiges JSON zu generieren, umschließen Sie den Dateneingabe-Transformator des Quelltextes in JSON. Zum Beispiel `{"data": <$.data>}`.

Implizites Textparsen kann auch verwendet werden, um Werte für die meisten Pipe-Ziel- oder Anreicherungsparameter dynamisch aufzufüllen. Weitere Informationen finden Sie unter ???.

> **Note**
>
> Wenn es sich bei der ursprünglichen Nutzlast um gültiges JSON handelt, enthält dieses Feld das nicht maskierte, nicht base64-kodierte JSON. Wenn es sich bei der Nutzlast jedoch nicht um gültiges JSON handelt, base64-codiert EventBridge für die unten aufgeführten Felder, mit Ausnahme von Amazon SQS.

- Aktives MQ – `data`
- Kinesis – `data`
- Amazon MSK – `key` und `value`
- Rabbit MQ – `data`
- Selbstverwaltetes Apache Kafka – `key` und `value`
- Amazon SQS – `body`

Häufige Probleme beim Transformieren von Eingaben

Dies sind einige häufige Probleme beim Transformieren von Eingaben in EventBridge-Pipes:

- Für Zeichenfolgen sind Anführungszeichen erforderlich.
- Beim Erstellen des JSON-Pfads für Ihre Vorlage erfolgt keine Validierung.
- Wenn Sie eine Variable angeben, die einem JSON-Pfad entspricht, der im Ereignis nicht vorhanden ist, wird diese Variable nicht erstellt und nicht in der Ausgabe angezeigt.

- JSON-Eigenschaften wie `aws.pipes.event.json` können nur als Wert eines JSON-Felds verwendet werden, nicht inline in anderen Zeichenfolgen.
- EventBridge maskiert Werte, die vom Eingabepfad extrahiert werden, nicht, wenn die Eingabevorlage für ein Ziel gefüllt wird.
- Wenn ein JSON-Pfad auf ein JSON-Objekt oder ein JSON-Array verweist, die Variable jedoch in einer Zeichenfolge referenziert wird, entfernt EventBridge alle internen Anführungszeichen, um sicherzustellen, dass eine gültige Zeichenfolge vorliegt. Zum Beispiel würde "Body is **<**$.body**>**" dazu führen, dass EventBridge Anführungszeichen aus dem Objekt entfernt.

  Wenn Sie also ein JSON-Objekt ausgeben möchten, das auf einer einzelnen JSON-Pfadvariablen basiert, müssen Sie es als Schlüssel platzieren. In diesem Beispiel `{"body": <$.body>}`.
- Für Variablen, die Zeichenfolgen darstellen, sind keine Anführungszeichen erforderlich. Sie sind zulässig, aber EventBridge Pipes fügt Werten für Zeichenfolgenvariablen während der Transformation automatisch Anführungszeichen hinzu, um sicherzustellen, dass die Transformationsausgabe gültiges JSON ist. EventBridge Pipes fügt Variablen, die JSON-Objekte oder -Arrays darstellen, keine Anführungszeichen hinzu. Fügen Sie für Variablen, die JSON-Objekte oder -Arrays darstellen, keine Anführungszeichen hinzu.

  Die folgende Eingabevorlage enthält beispielsweise Variablen, die sowohl Zeichenfolgen als auch JSON-Objekte darstellen:

```
{
  "pipeArn" : <aws.pipes.pipe-arn>,
  "pipeName" : <aws.pipes.pipe-name>,
  "originalEvent" : <aws.pipes.event.json>
}
```

  Das Ergebnis ist gültiges JSON mit den richtigen Anführungszeichen:

```
{
  "pipeArn" : "arn:aws:events:us-east-2:123456789012:pipe/example",
  "pipeName" : "example",
  "originalEvent" : {
    ... // commented for brevity
  }
}
```

- Bei Lambda- oder Step-Functions-Anreicherungen oder -Zielen werden Stapel als JSON-Arrays an das Ziel übermittelt, auch wenn die Stapelgröße 1 ist. Eingabe-Transformatoren werden jedoch

weiterhin auf einzelne Datensätze im JSON-Array angewendet, nicht auf das gesamte Array. Weitere Informationen finden Sie unter ???.

Amazon EventBridge Pipes protokollieren

EventBridge Mit der Pipes-Protokollierung können Sie festlegen, dass EventBridge Pipes Aufzeichnungen über die Pipe-Leistung an unterstützte AWS Dienste sendet. Verwenden Sie Protokolle, um einen Einblick in die Ausführungsleistung Ihrer Pipe zu erhalten und um bei der Fehlerbehebung und beim Debuggen zu helfen.

Sie können die folgenden AWS Dienste als Protokollziele auswählen, an die EventBridge Pipes Datensätze liefert:

- CloudWatch Logs

  EventBridge übermittelt Protokolldatensätze an die angegebene CloudWatch Logs-Protokollgruppe.

  Verwenden Sie CloudWatch Logs, um die Protokolle all Ihrer Systeme, Anwendungen und AWS Dienste, die Sie verwenden, in einem einzigen, hoch skalierbaren Service zu zentralisieren. Weitere Informationen finden Sie unter Arbeiten mit Protokollgruppen und Protokollstreams im Amazon CloudWatch Logs-Benutzerhandbuch.
- Firehose-Stream-Protokolle

  EventBridge liefert Protokolldatensätze an einen Firehose-Lieferstream.

  Amazon Data Firehose ist ein vollständig verwalteter Service für die Bereitstellung von Echtzeit-Streaming-Daten an Ziele wie bestimmte AWS Dienste sowie an alle benutzerdefinierten HTTP-Endpunkte oder HTTP-Endpunkte, die unterstützten Drittanbietern gehören. Weitere Informationen finden Sie unter Erstellen eines Amazon Data Firehose-Lieferdatenstroms im Amazon Data Firehose-Benutzerhandbuch.
- Amazon-S3-Protokolle

  EventBridge liefert Protokolldatensätze als Amazon S3 S3-Objekte an den angegebenen Bucket.

  Amazon S3 ist ein Objektspeicherservice, der branchenführende Skalierbarkeit, Datenverfügbarkeit, Sicherheit und Leistung bietet. Weitere Informationen finden Sie unter Hochladen, Herunterladen und Arbeiten mit Objekten in Amazon S3 im Benutzerhandbuch für Amazon Simple Storage Service.

So funktioniert die Amazon EventBridge Pipes-Protokollierung

Eine Pipe-Ausführung ist ein Ereignis oder ein Ereignisstapel, der von einer Pipe empfangen und zu einer Anreicherung und/oder einem Ziel weitergeleitet wird. Wenn diese Option aktiviert ist, EventBridge wird für jeden Ausführungsschritt, den sie bei der Verarbeitung des Ereignisstapels ausführt, ein Protokolldatensatz generiert. Die im Datensatz enthaltenen Informationen beziehen sich auf den Ereignisstapel, unabhängig davon, ob es sich um ein einzelnes Ereignis oder um bis zu 10 000 Ereignisse handelt.

Sie können die Größe des Ereignisstapels für die Quelle und das Ziel der Pipe konfigurieren. Weitere Informationen finden Sie unter ???.

Die an jedes Protokollziel gesendeten Datensatzdaten sind identisch.

Wenn ein Amazon CloudWatch Logs-Ziel konfiguriert ist, haben die Protokolldatensätze, die an alle Ziele gesendet werden, ein Limit von 256 KB. Felder werden nach Bedarf gekürzt.

Sie können die an die ausgewählten Protokollziele EventBridge gesendeten Datensätze wie folgt anpassen:

- Sie können die Protokollebene angeben, die die Ausführungsschritte bestimmt, für die Datensätze EventBridge an die ausgewählten Protokollziele gesendet werden. Weitere Informationen finden Sie unter ???.
- Sie können angeben, ob EventBridge Pipes Ausführungsdaten in Datensätze für Ausführungsschritte aufnimmt, sofern sie relevant sind. Diese Daten umfassen Folgendes:
  - Die Nutzlast des Ereignisstapels
  - Die Anfrage wurde an den AWS Anreicherungs- oder Zieldienst gesendet
  - Die vom AWS Anreicherungs- oder Zieldienst zurückgegebene Antwort

  Weitere Informationen finden Sie unter ???.

Geben Sie die EventBridge Pipes-Protokollebene an

Sie können die Arten von Ausführungsschritten angeben, für die Datensätze EventBridge an die ausgewählten Protokollziele gesendet werden.

Wählen Sie aus den folgenden Detailstufen, die in Protokolldatensätze aufgenommen werden sollen. Die Protokollebene gilt für alle für die Pipe angegebenen Protokollziele. Jede Protokollebene umfasst die Ausführungsschritte der vorherigen Protokollebenen.

- AUS — sendet EventBridge keine Datensätze an angegebene Protokollziele. Dies ist die Standardeinstellung.
- FEHLER — EventBridge sendet alle Datensätze, die sich auf Fehler beziehen, die während der Pipe-Ausführung generiert wurden, an die angegebenen Protokollziele.
- INFO — EventBridge sendet alle Datensätze, die sich auf Fehler beziehen, sowie ausgewählte andere Schritte, die während der Pipe-Ausführung ausgeführt wurden, an die angegebenen Protokollziele.
- TRACE — EventBridge sendet alle Datensätze, die während der einzelnen Schritte der Pipe-Ausführung generiert wurden, an die angegebenen Protokollziele.

In der EventBridge Konsole ist CloudWatch Logs standardmäßig als Log-Ziel ausgewählt, ebenso wie die `ERROR` Protokollebene. Daher erstellt EventBridge Pipes standardmäßig eine neue CloudWatch Protokollgruppe, an die Protokolldatensätze gesendet werden, die den `ERROR` Detaillierungsgrad enthalten. Bei der programmgesteuerten Konfiguration von Protokollen ist kein Standard ausgewählt.

Die folgende Tabelle enthält die Ausführungsschritte, die in den einzelnen Protokollebenen enthalten sind.

Schritt	TRACE	INFO	ERROR	OFF
Ausführung fehlgeschlagen	x	x	x	
Ausführung teilweise fehlgeschlagen	x	x	x	
Ausführung gestartet	x	x		
Ausführung erfolgreich	x	x		
Ausführung gedrosselt	x	x	x	
Execution Timeout	x	x	x	
Anreicherungsaufruf fehlgeschlagen	x	x	x	
Anreicherungsaufruf übersprungen	x	x		

Schritt	TRACE	INFO	ERROR	OFF
Anreicherungsaufruf gestartet	x			
Anreicherungsaufruf erfolgreich	x			
Anreicherungsstufe gestartet	x	x		
Anreicherungsstufe fehlgeschlagen	x	x	x	
Anreicherungsstufe erfolgreich	x	x		
Anreicherungstransformation fehlgeschlagen	x	x	x	
Anreicherungstransformation gestartet	x			
Anreicherungstransformation erfolgreich	x			
Zielaufruf fehlgeschlagen	x	x	x	
Zielaufruf teilweise fehlgeschlagen	x	x	x	
Zielaufruf übersprungen	x			
Zielaufruf gestartet	x			
Zielaufruf erfolgreich	x			
Zielstufe gestartet	x	x		
Zielstufe fehlgeschlagen	x	x	x	
Zielstufe teilweise fehlgeschlagen	x	x	x	

Schritt	TRACE	INFO	ERROR	OFF
Zielstufe übersprungen	x			
Zielstufe erfolgreich	x	x		
Zieltransformation fehlgesch lagen	x	x	x	
Zieltransformation gestartet	x			
Zieltransformation erfolgreich	x			

Inklusive Ausführungsdaten in EventBridge Pipes-Protokollen

Sie können angeben EventBridge , dass Ausführungsdaten in die generierten Datensätze aufgenommen werden sollen. Zu den Ausführungsdaten gehören Felder, die die Nutzlast des Ereignisstapels sowie die an die Anreicherung und das Ziel gesendete Anfrage und deren Antwort darstellen.

Ausführungsdaten sind für die Fehlerbehebung und das Debuggen nützlich. Das `payload`-Feld enthält den tatsächlichen Inhalt jedes im Stapel enthaltenen Ereignisses, sodass Sie einzelne Ereignisse mit einer bestimmten Pipe-Ausführung korrelieren können.

Wenn Sie sich dafür entscheiden, Ausführungsdaten einzubeziehen, sind diese für alle für die Pipe angegebenen Protokollziele enthalten.

> **Important**
>
> Diese Felder können vertrauliche Informationen enthalten. EventBridge unternimmt keinen Versuch, den Inhalt dieser Felder während der Protokollierung zu redigieren.

EventBridge Fügt beim Einbeziehen von Ausführungsdaten die folgenden Felder zu den entsprechenden Datensätzen hinzu:

- **`payload`**

  Stellt den Inhalt des Ereignisstapels dar, der von der Pipe verarbeitet wird.

EventBridge schließt das `payload` Feld in Datensätze ein, die in Schritten generiert wurden, in denen der Inhalt des Ereignisstapels möglicherweise aktualisiert wurde. Dazu gehören die folgenden Schritte:

- `EXECUTION_STARTED`
- `ENRICHMENT_TRANSFORMATION_SUCCEEDED`
- `ENRICHMENT_STAGE_SUCCEEDED`
- `TARGET_TRANSFORMATION_SUCCEEDED`
- `TARGET_STAGE_SUCCEEDED`

- **`awsRequest`**

  Stellt die an die Anreicherung oder das Ziel gesendete Anfrage als JSON-Zeichenfolge dar. Bei Anfragen, die an ein API-Ziel gesendet werden, stellt dies die an diesen Endpunkt gesendete HTTP-Anfrage dar.

  EventBridge schließt das `awsRequest` Feld in Datensätze ein, die in den letzten Schritten der Anreicherung und Targeting generiert wurden, d. h. nachdem die Anfrage für den angegebenen Anreicherungs- oder Zieldienst ausgeführt EventBridge wurde oder versucht wurde, sie auszuführen. Dazu gehören die folgenden Schritte:

  - `ENRICHMENT_INVOCATION_FAILED`
  - `ENRICHMENT_INVOCATION_SUCCEEDED`
  - `TARGET_INVOCATION_FAILED`
  - `TARGET_INVOCATION_PARTIALLY_FAILED`
  - `TARGET_INVOCATION_SUCCEEDED`

- **`awsResponse`**

  Stellt die von der Anreicherung oder dem Ziel zurückgegebene Antwort im JSON-Format dar. Bei Anfragen, die an ein API-Ziel gesendet werden, stellt dies die von diesem Endpunkt zurückgegebene HTTP-Antwort dar.

  Wie bei`awsRequest`, EventBridge schließt das `awsResponse` Feld in Datensätzen ein, die in den letzten Schritten von Anreicherung und Targeting generiert wurden, d. h. nachdem eine Anfrage für den angegebenen Anreicherungs- oder Zieldienst ausgeführt oder versucht wurde, sie auszuführen, und eine Antwort erhalten EventBridge hat. Dazu gehören die folgenden Schritte:

  - `ENRICHMENT_INVOCATION_FAILED`

- ENRICHMENT_INVOCATION_SUCCEEDED
- TARGET_INVOCATION_FAILED
- TARGET_INVOCATION_PARTIALLY_FAILED
- TARGET_INVOCATION_SUCCEEDED

Eine Erläuterung der Schritte zur Pipe-Ausführung finden Sie unter ???.

Kürzen von Ausführungsdaten in Pipes-Protokolldatensätzen EventBridge

Wenn Sie sich dafür entscheiden, Ausführungsdaten in die Protokolldatensätze einer Pipe EventBridge aufzunehmen, besteht die Möglichkeit, dass ein Datensatz die Größenbeschränkung von 256 KB überschreitet. Um dies zu verhindern, EventBridge werden die Felder mit den Ausführungsdaten automatisch in der folgenden Reihenfolge gekürzt. EventBridge schneidet jedes Feld vollständig ab, bevor das nächste Feld gekürzt wird. EventBridge kürzt Felddaten, indem einfach Zeichen am Ende der Datenzeichenfolge entfernt werden. Es wird nicht versucht, die Daten aufgrund der Wichtigkeit der Daten zu kürzen, und das Kürzen macht die JSON-Formatierung ungültig.

- payload
- awsRequest
- awsResponse

Wenn Felder im Ereignis EventBridge gekürzt werden, enthält das truncatedFields Feld eine Liste der gekürzten Datenfelder.

Fehlerberichterstattung in EventBridge Pipes-Protokolldatensätzen

EventBridge schließt, sofern verfügbar, auch Fehlerdaten in Schritten zur Pipe-Ausführung ein, die Fehlerzustände darstellen. Zu diesen Schritten gehören:

- ExecutionThrottled
- ExecutionTimeout
- ExecutionFailed
- ExecutionPartiallyFailed
- EnrichmentTransformationFailed
- EnrichmentInvocationFailed

- `EnrichmentStageFailed`
- `TargetTransformationFailed`
- `TargetInvocationFailed`
- `TargetInvocationPartiallyFailed`
- `TargetStageFailed`
- `TargetStagePartiallyFailed`

EventBridge Schritte zur Pipeline-Ausführung

Das Verständnis des Ablaufs der Pipe-Ausführungsschritte kann Ihnen bei der Fehlerbehebung oder beim Debuggen der Leistung Ihrer Pipe mithilfe von Protokollen helfen.

Eine Pipe-Ausführung ist ein Ereignis oder ein Ereignisstapel, der von einer Pipe empfangen und zu einer Anreicherung oder einem Ziel weitergeleitet wird. Wenn diese Option aktiviert ist, EventBridge wird für jeden Ausführungsschritt, der bei der Verarbeitung des Ereignisbatches ausgeführt wird, ein Protokolldatensatz generiert.

Auf hoher Ebene besteht die Ausführung aus zwei Stufen oder einer Reihe von Schritten: Anreicherung und Ziel. Jede dieser Stufen besteht aus Transformations- und Aufrufschritten.

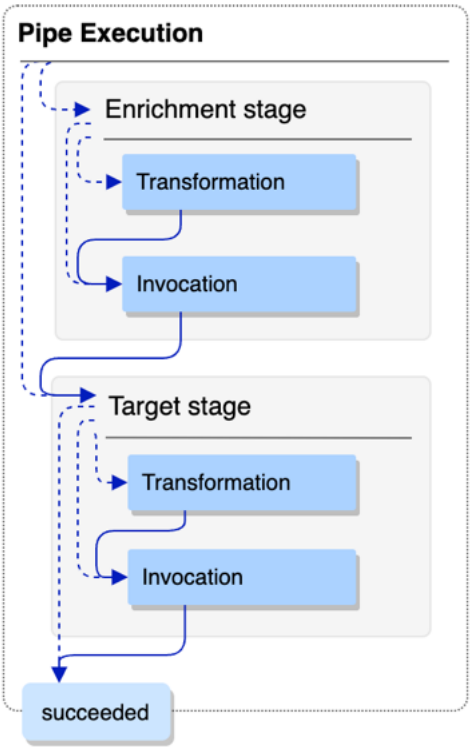
Die wichtigsten Schritte einer erfolgreichen Pipe-Ausführung folgen diesem Ablauf:

- Die Pipe-Ausführung wird gestartet.
- Die Ausführung geht in die Anreicherungsstufe über, wenn Sie eine Anreicherung für die Ereignisse angegeben haben. Wenn Sie keine Anreicherung angegeben haben, wird die Ausführung mit der Zielstufe fortgesetzt.

  In der Anreicherungsstufe führt die Pipe jede von Ihnen angegebene Transformation durch und ruft dann die Anreicherung auf.
- In der Zielstufe führt die Pipe jede von Ihnen angegebene Transformation durch und ruft dann das Ziel auf.

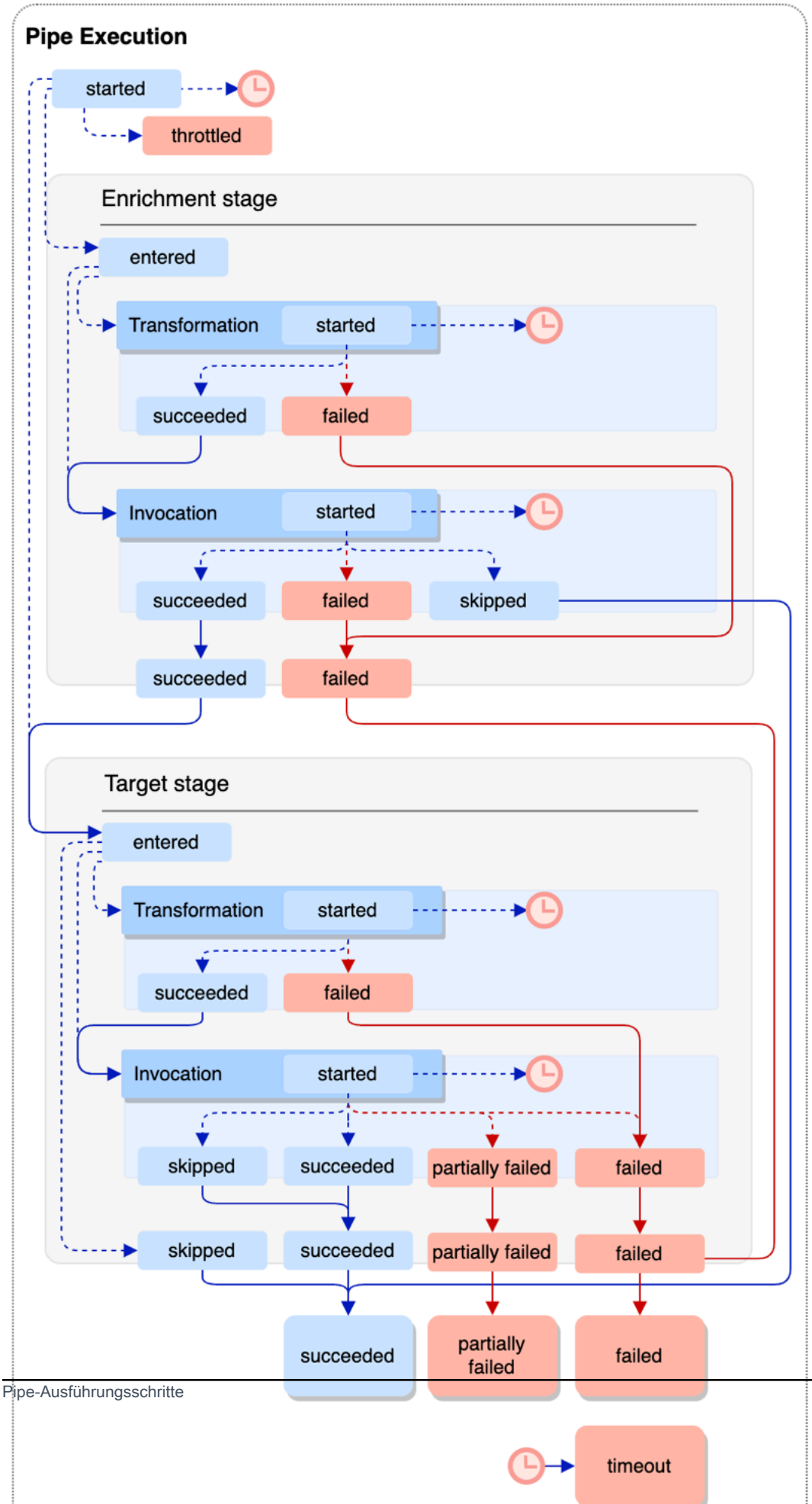
  Wenn Sie keine Transformation oder kein Ziel angegeben haben, überspringt die Ausführung die Zielstufe.
- Die Pipe-Ausführung wird erfolgreich abgeschlossen.

Das folgende Diagramm veranschaulicht diesen Ablauf. Divergierende Pfade werden als gepunktete Linien formatiert.



Das folgende Diagramm zeigt eine detaillierte Ansicht des Ablaufs der Pipe-Ausführung, wobei alle möglichen Ausführungsschritte dargestellt sind. Auch hier werden divergierende Pfade als gepunktete Linien formatiert

Eine vollständige Liste der Pipe-Ausführungsschritte finden Sie unter ???.


Pipe Execution
started
throttled
Enrichment stage
entered
Transformation
started
succeeded
failed
Invocation
started
succeeded
failed
skipped
succeeded
failed
Target stage
entered
Transformation
started
succeeded
failed
Invocation
started
skipped
succeeded
partially failed
failed
skipped
succeeded
partially failed
failed
succeeded
partially failed
failed
timeout

Beachten Sie, dass ein Zielaufruf zu einem teilweisen Fehler des Stapels führen kann. Weitere Informationen finden Sie unter ???.

EventBridge Referenz zum Pipes-Protokollschema

In der folgenden Referenz wird das Schema für EventBridge Pipes-Protokolldatensätze detailliert beschrieben.

Jeder Protokolldatensatz stellt einen Pipe-Ausführungsschritt dar und kann bis zu 10 000 Ereignisse enthalten, wenn die Pipe-Quelle und das Pipe-Ziel für die Stapelverarbeitung konfiguriert wurden.

Weitere Informationen finden Sie unter ???.

```
{
    "executionId": "guid",
    "timestamp": "date_time",
    "messageType": "execution_step",
    "resourceArn": "arn:aws:pipes:region:account:pipe/pipe-name",
    "logLevel": "TRACE | INFO | ERROR",
    "payload": "{}",
    "awsRequest": "{}"
    "awsResponse":"{}"
    "truncatedFields": ["awsRequest","awsResponse","payload"],
    "error": {
        "httpStatusCode": code,
        "message": "error_message",
        "details": "",
        "awsService": "service_name",
        "requestId": "service_request_id"
    }
}
```

executionId

Die ID der Pipe-Ausführung.

Eine Pipe-Ausführung ist ein Ereignis oder ein Ereignisstapel, der von einer Pipe empfangen und zu einer Anreicherung oder einem Ziel weitergeleitet wird. Weitere Informationen finden Sie unter ???.

timestamp

Das Datum und die Uhrzeit, an denen das Protokollereignis ausgegeben wurde.

Einheit: Millisekunden

messageType

Der Pipe-Ausführungsschritt, für den der Datensatz generiert wurde.

Weitere Informationen zu den Pipe-Ausführungsschritten finden Sie unter ???.

resourceArn

Der Amazon-Ressourcenname (ARN) für die Pipe.

logLevel

Die für das Pipe-Protokoll angegebene Detailstufe.

Zulässige Werte: `ERROR` | `INFO` | `TRACE`

Weitere Informationen finden Sie unter ???.

Nutzlast

Der Inhalt des Ereignisstapels, der von der Pipe verarbeitet wird.

EventBridge schließt dieses Feld nur ein, wenn Sie angegeben haben, dass Ausführungsdaten in die Protokolle für diese Pipe aufgenommen werden sollen. Weitere Informationen finden Sie unter ???.

> **Important**
>
> Diese Felder können vertrauliche Informationen enthalten. EventBridge unternimmt keinen Versuch, den Inhalt dieser Felder während der Protokollierung zu redigieren.

Weitere Informationen finden Sie unter ???.

awsRequest

Die an die Anreicherung oder das Ziel gesendete Anfrage im JSON-Format. Bei Anfragen, die an ein API-Ziel gesendet werden, stellt dies die an diesen Endpunkt gesendete HTTP-Anfrage dar.

EventBridge schließt dieses Feld nur ein, wenn Sie angegeben haben, dass Ausführungsdaten in die Protokolle für diese Pipe aufgenommen werden sollen. Weitere Informationen finden Sie unter ???.

> **Important**
>
> Diese Felder können vertrauliche Informationen enthalten. EventBridge unternimmt keinen Versuch, den Inhalt dieser Felder während der Protokollierung zu redigieren.

Weitere Informationen finden Sie unter ???.

awsResponse

Die von der Anreicherung oder dem Ziel zurückgegebene Antwort im JSON-Format. Bei Anfragen, die an ein API-Ziel gesendet werden, stellt dies die HTTP-Antwort dar, die von diesem Endpunkt zurückgegeben wird, und nicht die Antwort, die vom API-Zielservice selbst zurückgegeben wird.

EventBridge schließt dieses Feld nur ein, wenn Sie angegeben haben, dass Ausführungsdaten in die Protokolle für diese Pipe aufgenommen werden sollen. Weitere Informationen finden Sie unter ???.

> **Important**
>
> Diese Felder können vertrauliche Informationen enthalten. EventBridge unternimmt keinen Versuch, den Inhalt dieser Felder während der Protokollierung zu redigieren.

Weitere Informationen finden Sie unter ???.

truncatedFields

Eine Liste aller Ausführungsdatenfelder EventBridge wurde gekürzt, um den Datensatz unter der Größenbeschränkung von 256 KB zu halten.

Wenn EventBridge keines der Ausführungsdatenfelder gekürzt werden musste, ist dieses Feld aber vorhanden. `null`

Weitere Informationen finden Sie unter ???.

error

Enthält Informationen zu allen Fehlern, die während dieses Pipe-Ausführungsschritts generiert wurden.

Wenn bei diesem Pipe-Ausführungsschritt kein Fehler generiert wurde, ist dieses Feld vorhanden, aber `null`.

httpStatusCode

Der vom aufgerufenen Service zurückgegebene HTTP-Statuscode.

message

Die vom aufgerufenen Service zurückgegebene Fehlermeldung.

details

Alle detaillierten Fehlerinformationen, die vom aufgerufenen Service zurückgegeben wurden.

awsService

Der Name des aufgerufenen Service.

requestId

Die Anfrage-ID für diese Anfrage vom aufgerufenen Service.

Protokollierung und Überwachung von Amazon EventBridge Pipes mithilfe von AWS CloudTrail Amazon CloudWatch Logs

Sie können EventBridge Pipes-Aufrufe und die Verwendung von Pipes protokollieren CloudTrail und den Zustand Ihrer Pipes anhand CloudWatch von Metriken überwachen.

CloudWatch Metriken

EventBridge Pipes sendet CloudWatch jede Minute Metriken an Amazon für alles, von der Drosselung von Pipe-Ausführungen bis hin zu einem erfolgreich aufgerufenen Ziel.

Metrik	Beschreibung	Dimensionen	Einheiten
`Concurrency`	Die Anzahl der gleichzeitigen Ausführungen einer Pipe.	AwsAccountId	None
`Duration`	Dauer, die die Ausführung der Pipe in Anspruch nahm.	PipeName	Millisekunden
`EventCount`	Die Anzahl der Ereignisse, die eine Pipe verarbeitet hat.	PipeName	None

Metrik	Beschreibung	Dimensionen	Einheiten
`EventSize`	Die Größe der Nutzlast des Ereignisses, das die Pipe aufgerufen hat.	PipeName	Bytes
`ExecutionThrottled`	Wie viele Ausführungen einer Pipe wurden gedrosselt. **Note** Dieser Wert lautet `0`, wenn keine Ausführungen gedrosselt wurden.	AwsAccountId, PipeName	None
`ExecutionTimeout`	Bei wie vielen Ausführungen einer Pipe wurde das Timeout überschritten, bevor die Ausführung abgeschlossen wurde. **Note** Dieser Wert lautet `0`, wenn bei keiner Ausführung ein Timeout aufgetreten ist.	PipeName	None
`ExecutionFailed`	Wie viele Ausführungen einer Pipe sind fehlgeschlagen. **Note** Dieser Wert lautet `0`, wenn keine Ausführung fehlgeschlagen ist.	PipeName	None

Metrik	Beschreibung	Dimensionen	Einheiten
`ExecutionPartiallyFailed`	Wie viele Ausführungen einer Pipe sind teilweise fehlgeschlagen. **Note** Dieser Wert lautet `0`, wenn keine Ausführung teilweise fehlgeschlagen ist.	PipeName	None
`EnrichmentStageDuration`	Wie lange es gedauert hat, bis die Anreicherungsstufe abgeschlossen war.	PipeName	Millisekunden
`EnrichmentStageFailed`	Wie viele Ausführungen der Anreicherungsstufe einer Pipe sind fehlgeschlagen. **Note** Dieser Wert lautet `0`, wenn keine Ausführung fehlgeschlagen ist.	PipeName	None
`Invocations`	Gesamtzahl der Aufrufe.	AwsAccountId, PipeName	None
`TargetStageDuration`	Wie lange es gedauert hat, bis die Zielstufe abgeschlossen war.	PipeName	Millisekunden

Metrik	Beschreibung	Dimensionen	Einheiten
`TargetStageFailed`	Wie viele Ausführungen der Zielstufe einer Pipe sind fehlgeschlagen. **Note** Dieser Wert lautet `0`, wenn keine Ausführung fehlgeschlagen ist.	PipeName	None
`TargetStagePartiallyFailed`	Wie viele Ausführungen der Zielstufe einer Pipe sind teilweise fehlgeschlagen. **Note** Dieser Wert lautet `0`, wenn keine Ausführung der Zielstufe teilweise fehlgeschlagen ist.	PipeName	None
`TargetStageSkipped`	Wie viele Ausführungen der Zielstufe einer Pipe wurden übersprungen (z. B. weil die Anreicherung eine leere Nutzlast zurückgab).	PipeName	Anzahl

Dimensionen für CloudWatch Metriken

CloudWatch Metriken haben Dimensionen oder sortierbare Attribute, die unten aufgeführt sind.

Dimension	Beschreibung
`AwsAccountId`	Filtert die verfügbaren Metriken nach Konto-ID.
`PipeName`	Filtert die verfügbaren Metriken nach Pipe-Name.

Amazon EventBridge Pipes Fehlerbehandlung und Fehlerbehebung

Wiederholungsverhalten und Fehlerbehandlung

EventBridge Pipes wiederholt automatisch die Anreicherung und den Zielaufruf bei allen wiederholbaren AWS Fehlern mit dem Quelldienst, dem Anreicherungsdienst oder den Zieldiensten oder. EventBridge Wenn jedoch Fehler von der Anreicherung oder den Ziel-Kunden-Implementierungen zurückgegeben werden, wird der Durchsatz bei der Pipe-Abfrage allmählich reduziert. Bei fast kontinuierlichen 4xx-Fehlern (z. B. Autorisierungsprobleme mit IAM oder fehlende Ressourcen) kann die Pipe automatisch deaktiviert werden, wobei eine erläuternde Meldung im `StateReason` angezeigt wird.

Pipe-Aufruffehler und Wiederholungsverhalten

Wenn Sie eine Pipe aufrufen, können zwei Haupttypen von Fehlern auftreten: Pipe-interne Fehler und Kundenaufruffehler.

Interne Pipe-Fehler

Interne Pipe-Fehler sind Fehler, die auf Aspekte des vom Pipes-Dienst verwalteten Aufrufs zurückzuführen sind. EventBridge

Zu diesen Fehlern können unter anderem folgende Probleme gehören:

- Ein HTTP-Verbindungsfehler beim Versuch, den Kundenzielservice aufzurufen
- Ein vorübergehender Rückgang der Verfügbarkeit des Pipe-Service selbst

Im Allgemeinen wiederholt EventBridge Pipes interne Fehler auf unbestimmte Zeit und stoppt erst, wenn der Datensatz in der Quelle abläuft.

Bei Pipes mit einer Streamquelle zählt EventBridge Pipes die Wiederholungen für interne Fehler nicht auf die maximale Anzahl von Wiederholungen, die in der Wiederholungsrichtlinie für die Streamquelle angegeben ist. Bei Pipes mit einer Amazon SQS SQS-Quelle zählt EventBridge Pipes keine Wiederholungen für interne Fehler auf die maximale Empfangsanzahl für die Amazon SQS SQS-Quelle.

Kundenaufruffehler

Kundenaufruffehler sind Fehler, die auf die vom Benutzer verwaltete Konfiguration oder auf den vom Benutzer verwalteten Code zurückzuführen sind.

Zu diesen Fehlern können unter anderem folgende Probleme gehören:

- Unzureichende Berechtigungen für die Pipe, um das Ziel aufzurufen.
- Ein Logikfehler in einem synchron aufgerufenen Lambda-, Step-Functions-, API-Ziel- oder API-Gateway-Endpunkt eines Kunden.

Bei Fehlern beim Kundenaufruf geht EventBridge Pipes wie folgt vor:

- Bei Pipes mit einer Streamquelle versucht EventBridge Pipes bis zu den in der Pipe-Wiederholungsrichtlinie konfigurierten maximalen Wiederholungszeiten oder bis zum Ablauf des maximalen Datensatzalters, je nachdem, was zuerst eintritt.
- Bei Pipes mit einer Amazon SQS SQS-Quelle versucht EventBridge Pipes erneut, einen Kundenfehler bis zur maximalen Empfangsanzahl in der Quellwarteschlange zu erreichen.
- Bei Pipes mit einer Apache Kafka- oder Amazon MQ MQ-Quelle werden Kundenfehler genauso wiederholt wie interne Fehler. EventBridge

Bei Pipes mit Rechenzielen müssen Sie die Pipe synchron aufrufen, damit EventBridge Pipes alle Laufzeitfehler erkennt, die von der Rechenlogik des Kunden ausgelöst werden, und es bei solchen Fehlern erneut versuchen kann. Pipes können bei Fehlern, die durch die Logik eines Step-Functions-Standard-Workflows ausgelöst werden, nicht wiederholen, da dieses Ziel asynchron aufgerufen werden muss.

Für Amazon SQS und Stream-Quellen wie Kinesis und DynamoDB unterstützt EventBridge Pipes die teilweise Batch-Fehlerbehandlung von Zielausfällen. Weitere Informationen finden Sie unter Teilweiser Stapelfehler.

Pipe-DLQ-Verhalten

Eine Pipe erbt das Verhalten der Warteschlange für unzustellbare Nachrichten von der Quelle:

- Wenn die Amazon-SQS-Quellwarteschlange über eine konfigurierte Warteschlange für unzustellbare Nachrichten verfügt, werden Nachrichten nach der angegebenen Anzahl von Versuchen automatisch dorthin zugestellt.

- Für Streaming-Quellen wie DynamoDB- und Kinesis-Streams können Sie eine Warteschlange für unzustellbare Nachrichten für die Pipe- und Weiterleitungsereignisse konfigurieren. DynamoDB- und Kinesis-Streaming-Quellen unterstützen Amazon-SQS-Warteschlangen und Amazon-SNS-Themen als Ziele der Warteschlange für unzustellbare Nachrichten.

Wenn Sie eine `DeadLetterConfig` für eine Pipe mit einer Kinesis- oder DynamoDB-Quelle angeben, stellen Sie sicher, dass die `MaximumRecordAgeInSeconds`-Eigenschaft der Pipe kleiner als das `MaximumRecordAge` des Quellereignisses ist. `MaximumRecordAgeInSeconds` steuert, wann der Pipe-Poller das Ereignis aufgibt und es an die Warteschlange für unzustellbare Nachrichten weiterleitet, und das `MaximumRecordAge` steuert, wie lange die Nachricht im Quell-Stream sichtbar ist, bevor sie gelöscht wird. Legen Sie daher `MaximumRecordAgeInSeconds` auf einen Wert fest, der kleiner als das `MaximumRecordAge` der Quelle ist, sodass zwischen dem Zeitpunkt, an dem das Ereignis an die Warteschlange für unzustellbare Nachrichten gesendet wird, und dem Zeitpunkt, an dem es automatisch von der Quelle gelöscht wird, ausreichend Zeit verbleibt, damit Sie feststellen können, warum das Ereignis an die Warteschlange für unzustellbare Nachrichten ging.

Für Amazon-MQ-Quellen kann die Warteschlange für unzustellbare Nachrichten direkt im Message Broker konfiguriert werden.

EventBridge Pipes unterstützt keine First-In-First-Out-DLQs (FIFO) für Stream-Quellen.

EventBridge Pipes unterstützt DLQ nicht für Amazon MSK-Stream- und selbstverwaltete Apache Kafka-Stream-Quellen.

Pipe-Fehlerzustände

Das Erstellen, Löschen und Aktualisieren von Pipes sind asynchrone Operationen, die zu einem Fehlerstatus führen können. Ebenso kann eine Pipe aufgrund von Fehlern automatisch deaktiviert werden. In allen Fällen stellt der Pipe-`StateReason` Informationen zur Verfügung, die bei der Behebung des Fehlers helfen.

Im Folgenden finden Sie ein Beispiel der möglichen `StateReason`-Werte:

- Stream nicht gefunden Um die Bearbeitung fortzusetzen, löschen Sie bitte die Pipe und erstellen Sie eine neue.
- Pipes verfügt nicht über die erforderlichen Berechtigungen zur Ausführung von Warteschlangenoperationen (sqs:ReceiveMessage, sqs: und sqs:) DeleteMessage GetQueueAttributes

- Verbindungsfehler Ihre VPC muss eine Verbindung zu Pipes herstellen können. Sie können Zugriff gewähren, indem Sie ein NAT-Gateway oder einen VPC-Endpunkt für Pipes-Daten konfigurieren. Informationen zur Einrichtung eines NAT-Gateways oder VPC-Endpunkts für Pipes-Daten finden Sie in der Dokumentation. AWS
- Dem MSK-Cluster sind keine Sicherheitsgruppen zugeordnet

Eine Pipe kann mit einem aktualisierten `StateReason` automatisch gestoppt werden. Mögliche Gründe sind:

- Ein Step-Functions-Standard-Workflow, der als Anreicherung konfiguriert wurde.
- Ein Step-Functions-Standard-Workflow, der als Ziel konfiguriert ist und synchron aufgerufen werden soll.

Fehler bei der benutzerdefinierten Verschlüsselung

Wenn Sie eine Quelle so konfigurieren, dass sie einen AWS KMS benutzerdefinierten Verschlüsselungsschlüssel (CMK) anstelle eines AWS verwalteten Schlüssels verwendet, müssen Sie der Execution AWS KMS Role Ihrer Pipe ausdrücklich die Berechtigung zur Entschlüsselung erteilen. Nehmen Sie dazu die folgende zusätzliche Berechtigung in die benutzerdefinierte CMK-Richtlinie auf:

```
  {
      "Sid": "Allow Pipes access",
      "Effect": "Allow",
      "Principal": {
          "AWS": "arn:aws:iam::01234567890:role/service-role/
Amazon_EventBridge_Pipe_DDBStreamSourcePipe_12345678"
      },
      "Action": "kms:Decrypt",
      "Resource": "*"
  }
```

Ersetzen Sie die obige Rolle durch die Ausführungsrolle Ihrer Pipe.

Dies gilt für alle Pipe-Quellen mit AWS KMS CMK, einschließlich:

- Amazon DynamoDB Streams
- Amazon Kinesis Data Streams

- Amazon MQ
- Amazon MSK
- Amazon SQS

Tutorial: Erstellen einer EventBridge-Pipe, die Quellereignisse filtert

In diesem Tutorial erstellen Sie eine Pipe, die eine DynamoDB-Stream-Quelle mit einem Amazon-SQS-Warteschlangenziel verbindet. Dazu gehört die Angabe eines Ereignismusters für die Pipe, das beim Filtern von Ereignissen zur Übermittlung an die Warteschlange verwendet werden soll. Anschließend testen Sie die Pipe, um sicherzustellen, dass nur die gewünschten Ereignisse übermittelt werden.

Voraussetzungen: Erstellen der Quelle und des Ziels

Bevor Sie die Pipe erstellen, müssen Sie die Quelle und das Ziel erstellen, mit denen die Pipe verbunden werden soll. In diesem Fall ein Amazon-DynamoDB-Datenstrom als Pipe-Quelle und eine Amazon-SQS-Warteschlange als Pipe-Ziel.

Zur Vereinfachung dieses Schrittes können Sie mit AWS CloudFormation die Quell- und Zielressourcen bereitstellen. Dazu erstellen Sie eine CloudFormation-Vorlage, die die folgenden Ressourcen definiert:

- Die Pipe-Quelle

  Eine Amazon-DynamoDB-Tabelle namens `pipe-tutorial-source` mit einem aktivierten Stream, um einen geordneten Informationsfluss zu Elementänderungen in der DynamoDB-Tabelle bereitzustellen.

- Das Pipe-Ziel

  Eine Amazon-SQS-Warteschlange namens `pipe-tutorial-target`, um den DynamoDB-Stream von Ereignissen aus der Pipe zu empfangen.

So erstellen Sie die CloudFormation-Vorlage für die Bereitstellung von Pipe-Ressourcen

1. Kopieren Sie den Text der JSON-Vorlage im folgenden Abschnitt ???.
2. Speichern Sie die Vorlage als JSON-Datei (z. B. `~/pipe-tutorial-resources.json`).

Verwenden Sie als Nächstes die Vorlagendatei, die Sie gerade erstellt haben, um einen CloudFormation-Stack bereitzustellen.

> **Note**
>
> Sobald Sie den CloudFormation-Stack erstellt haben, werden Ihnen die bereitgestellten AWS-Ressourcen berechnet.

Bereitstellen der Voraussetzungen für das Tutorial mithilfe der AWS-CLI

- Führen Sie den folgenden CLI-Befehl aus, wobei `--template-body` den Speicherort der Vorlagendatei angibt:

```
aws cloudformation create-stack --stack-name pipe-tuturial-resources --template-body file://~/pipe-tutorial-resources.json
```

Bereitstellen der Voraussetzungen für das Tutorial mit der CloudFormation-Konsole

1. Öffnen Sie die AWS CloudFormation-Konsole unter https://console.aws.amazon.com/cloudformation.
2. Wählen Sie Stacks und dann Stack erstellen und Mit neuen Ressourcen (Standard) aus.

   CloudFormation zeigt den Assistenten Stack erstellen an.
3. Lassen Sie für Voraussetzung – Vorlage vorbereiten die Standardeinstellung Vorlage ist bereit ausgewählt.
4. Wählen Sie unter Vorlage festlegen die Option Vorlagendatei hochladen und dann die Datei und Weiter aus.
5. Konfigurieren Sie den Stack und die Ressourcen, die er bereitstellen soll:
   - Geben Sie unter Stack name (Stack-Name) `pipe-tuturial-resources` ein.
   - Behalten Sie für Parameter die Standardnamen für die DynamoDB-Tabelle und die Amazon-SQS-Warteschlange bei.
   - Wählen Sie Next (Weiter).
6. Wählen Sie Weiter und dann Absenden aus.

CloudFormation erstellt den Stack und stellt die Ressourcen bereit, die in der Vorlage definiert sind.

Weitere Informationen zu CloudFormation finden Sie unter Was ist AWS CloudFormation? im AWS CloudFormation-Benutzerhandbuch.

Schritt 1: Erstellen der Pipe

Nachdem die Pipe-Quelle und das Pipe-Ziel bereitgestellt wurden, können Sie jetzt die Pipe erstellen, um die beiden Services zu verbinden.

Erstellen der Pipe mit der EventBridge-Konsole

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Pipes aus.
3. Wählen Sie Pipe erstellen aus.
4. Geben Sie für Name den Namen `pipe-tutorial` für die Pipe ein.
5. Geben Sie die DynamoDB-Datenstromquelle an:
   a. Wählen Sie unter Details für Quelle die Option DynamoDB-Datenstrom aus.

      EventBridge zeigt DynamoDB-spezifische Quellkonfigurationseinstellungen an.
   b. Wählen Sie für DynamoDB-Stream `pipe-tutorial-source` aus.

      Behalten Sie die Standardeinstellung `Latest` für Startposition bei.
   c. Wählen Sie Next (Weiter).
6. Geben Sie ein Ereignismuster an und testen Sie es, um Ereignisse zu filtern:

   Durch Filtern können Sie steuern, welche Ereignisse die Pipe an die Anreicherung oder an das Ziel sendet. Die Pipe sendet nur Ereignisse, die dem Ereignismuster entsprechen, an die Anreicherung oder das Ziel.

   Weitere Informationen finden Sie unter ???.

> **Note**
>
> Ihnen werden nur die Ereignisse in Rechnung gestellt, die an die Anreicherung oder das Ziel gesendet wurden.

a. Lassen Sie unter Beispielereignis – optional die Option AWS-Ereignisse ausgewählt und stellen Sie sicher, dass DynamoDB-Stream-Beispielereignis 1 ausgewählt ist.

   Dies ist das Beispielereignis, das Sie verwenden werden, um unser Ereignismuster zu testen.

b. Geben Sie unter Ereignismuster das folgende Ereignismuster ein:

```
{
  "eventName": ["INSERT", "MODIFY"]
}
```

c. Wählen Sie Testmuster aus.

   EventBridge zeigt eine Meldung an, dass das Beispielereignis dem Ereignismuster entspricht. Dies liegt daran, dass das Beispielereignis den `eventName`-Wert `INSERT` aufweist.

d. Wählen Sie Next (Weiter).

7. Wählen Sie Weiter aus, um die Angabe einer Anreicherung zu überspringen.

   In diesem Beispiel wählen Sie keine Anreicherung aus. Mit Anreicherungen können Sie einen Service auswählen, um die Daten aus der Quelle zu optimieren, bevor Sie sie an das Ziel senden. Weitere Details finden Sie unter ???.

8. Geben Sie Ihre Amazon-SQS-Warteschlange als Pipe-Ziel an:

   a. Wählen Sie unter Details für Zielservice die Option Amazon-SQS-Warteschlange aus.

   b. Wählen Sie für Warteschlange `pipe-tutorial-target` aus.

   c. Lassen Sie den Abschnitt Zieleingabe-Transformator leer.

      Weitere Informationen finden Sie unter ???.

9. Wählen Sie Pipe erstellen aus.

EventBridge erstellt die Pipe und zeigt die Pipe-Detailseite an. Die Pipe ist bereit, sobald ihr Status auf `Running` aktualisiert wurde.

Schritt 2: Bestätigen der Pipe-Filterereignisse

Die Pipe ist eingerichtet, hat aber noch keine Ereignisse aus der Tabelle empfangen.

Zum Testen der Pipe aktualisieren Sie die Einträge in der DynamoDB-Tabelle. Bei jeder Aktualisierung werden Ereignisse generiert, die der DynamoDB-Stream an unsere Pipe sendet. Einige entsprechen dem von Ihnen angegebenen Ereignismuster, andere nicht. Anschließend können Sie die Amazon-SQS-Warteschlange untersuchen, um sicherzustellen, dass die Pipe nur die Ereignisse übermittelt hat, die unserem Ereignismuster entsprachen.

Aktualisieren der Tabellenelemente, um Ereignisse zu generieren

1. Öffnen Sie die DynamoDB-Konsole unter https://console.aws.amazon.com/dynamodb/.
2. Wählen Sie in der linken Navigationsleiste die Option Tabellen aus. Wählen Sie die Tabelle `pipe-tutorial-source` aus.

   DynamoDB zeigt die Tabellendetailseite für `pipe-tutorial-source` an.
3. Wählen Sie Tabellenelemente durchsuchen und dann Element erstellen aus.

   DynamoDB zeigt die Seite Element erstellen an.
4. Erstellen Sie unter Attribute ein neues Tabellenelement:

   a. Geben Sie für Album `Album A` ein.

   b. Geben Sie für Künstler `Artist A` ein.

   c. Wählen Sie Create item (Element erstellen) aus.
5. Aktualisieren Sie das Tabellenelement:

   a. Wählen Sie unter Zurückgegebene Elemente die Option Album A aus.

   b. Wählen Sie Neues Attribut hinzufügen und anschließend Zeichenfolge aus.

   c. Geben Sie den neuen Wert `Song` mit dem Wert `Song A` ein.

   d. Wählen Sie Save Changes.
6. Löschen Sie das Tabellenelement:

a. Aktivieren Sie unter Zurückgegebene Elemente die Option Album A.

b. Wählen Sie im Menü Aktionen die Option Elemente löschen aus.

Sie haben drei Aktualisierungen am Tabellenelement vorgenommen. Dadurch werden drei Ereignisse für den DynamoDB-Datenstrom generiert:

- Ein `INSERT`-Ereignis, als Sie das Element erstellt haben
- Ein `MODIFY`-Ereignis, als Sie dem Element ein Attribut hinzugefügt haben
- Ein `REMOVE`-Ereignis, als Sie das Element gelöscht haben

Das von Ihnen für die Pipe angegebene Ereignismuster sollte jedoch alle Ereignisse herausfiltern, bei denen es sich nicht um `INSERT`- oder `MODIFY`-Ereignisse handelt. Stellen Sie als Nächstes sicher, dass die Pipe die erwarteten Ereignisse an die Warteschlange übermittelt hat.

Bestätigen, dass die erwarteten Ereignisse an die Warteschlange übermittelt wurden

1. Öffnen Sie die Amazon-SQS-Konsole unter https://console.aws.amazon.com/sqs/.
2. Wählen Sie die Warteschlange `pipe-tutorial-target` aus.

   Amazon SQS zeigt die Seite mit den Warteschlangendetails an.
3. Wählen Sie Nachrichten senden und empfangen und dann unter Nachrichten empfangen die Option Abrufen von Nachrichten aus.

   Die Warteschlange fragt die Pipe ab und listet dann die Ereignisse auf, die sie empfängt.
4. Wählen Sie den Namen des Ereignisses aus, um das übermittelte Ereignis-JSON zu sehen.

In der Warteschlange sollten sich zwei Ereignisse befinden: eines mit dem `eventName` `INSERT` und eines mit dem `eventName` `MODIFY`. Die Pipe hat das Ereignis für das Löschen des Tabellenelements jedoch nicht übermittelt, da dieses Ereignis den `eventName` `REMOVE` aufwies, der nicht dem von Ihnen in der Pipe angegebenen Ereignismuster entsprach.

Schritt 3: Bereinigen Ihrer Ressourcen

Löschen Sie zunächst die Pipe selbst.

Löschen der Pipe mit der EventBridge-Konsole

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Pipes aus.
3. Wählen Sie die Pipe `pipe-tutorial` und dann Löschen aus.

Löschen Sie anschließend den CloudFormation-Stack, um zu verhindern, dass Ihnen die fortgesetzte Nutzung der darin bereitgestellten Ressourcen berechnet wird.

Löschen der Voraussetzungen für das Tutorial mithilfe der AWS-CLI

- Führen Sie den folgenden CLI-Befehl aus, wobei `--stack-name` den Namen Ihres Stacks angibt:

```
aws cloudformation delete-stack --stack-name pipe-tuturial-resources
```

Löschen der Voraussetzungen für das Tutorial mithilfe der AWS CloudFormation-Konsole

1. Öffnen Sie die AWS CloudFormation-Konsole unter https://console.aws.amazon.com/cloudformation.
2. Wählen Sie auf der Seite Stacks den Stack und dann Löschen aus.
3. Wählen Sie Löschen aus, um Ihre Aktion zu bestätigen.

AWS CloudFormation-Vorlage für die Generierung der Voraussetzungen

Verwenden Sie den folgenden JSON-Code, um eine CloudFormation-Vorlage für die Bereitstellung der für dieses Tutorial erforderlichen Quell- und Zielressourcen zu erstellen.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",

 "Description" : "Provisions resources to use with the EventBridge Pipes tutorial. You
 will be billed for the AWS resources used if you create a stack from this template.",

  "Parameters" : {
    "SourceTableName" : {
      "Type" : "String",
      "Default" : "pipe-tutorial-source",
```

```
      "Description" : "Specify the name of the table to provision as the pipe source,
 or accept the default."
    },
  "TargetQueueName" : {
    "Type" : "String",
    "Default" : "pipe-tutorial-target",
    "Description" : "Specify the name of the queue to provision as the pipe target, or
 accept the default."
  }
},
  "Resources": {
    "PipeTutorialSourceDynamoDBTable": {
      "Type": "AWS::DynamoDB::Table",
      "Properties": {
        "AttributeDefinitions": [{
            "AttributeName": "Album",
            "AttributeType": "S"
          },
          {
            "AttributeName": "Artist",
            "AttributeType": "S"
          }

        ],
        "KeySchema": [{
            "AttributeName": "Album",
            "KeyType": "HASH"

          },
          {
            "AttributeName": "Artist",
            "KeyType": "RANGE"
          }
        ],
        "ProvisionedThroughput": {
          "ReadCapacityUnits": 10,
          "WriteCapacityUnits": 10
        },
        "StreamSpecification": {
          "StreamViewType": "NEW_AND_OLD_IMAGES"
        },
        "TableName": { "Ref" : "SourceTableName" }
      }
    },
```

```
    "PipeTutorialTargetQueue": {
      "Type": "AWS::SQS::Queue",
      "Properties": {
        "QueueName": { "Ref" : "TargetQueueName" }
      }
    }
  }
}
```

Generieren Sie eine AWS CloudFormation Vorlage aus EventBridge Pipes

AWS CloudFormation ermöglicht es Ihnen, Ihre AWS Ressourcen konto- und regionsübergreifend zentral und wiederholbar zu konfigurieren und zu verwalten, indem Infrastruktur als Code behandelt wird. CloudFormation ermöglicht dies, indem Sie Vorlagen erstellen können, die die Ressourcen definieren, die Sie bereitstellen und verwalten möchten.

EventBridge ermöglicht es Ihnen, Vorlagen aus den vorhandenen Pipes in Ihrem Konto zu generieren, um Ihnen den Einstieg in die Entwicklung von CloudFormation Vorlagen zu erleichtern. Sie können eine einzelne Pipe oder mehrere Pipes auswählen, die in die Vorlage aufgenommen werden sollen. Sie können diese Vorlagen dann als Grundlage für die Erstellung von Stapel verwalteter Ressourcen verwenden. CloudFormation

Weitere Informationen zu CloudFormation finden Sie im AWS CloudFormation Benutzerhandbuch.

Für Event-Busse können Sie CloudFormation Vorlagen aus Event-Bussen und Event-Bus-Regeln generieren.

In EventBridge Pipe-Vorlagen enthaltene Ressourcen

Beim EventBridge Generieren der CloudFormation Vorlage wird für jede ausgewählte Leitung eine AWS::Pipes::PipeRessource erstellt. EventBridge Enthält außerdem die folgenden Ressourcen unter den beschriebenen Bedingungen:

- AWS::Events::ApiDestination

  Wenn Ihre Pipes API-Ziele enthalten, entweder als Anreicherungen oder als Ziele, EventBridge werden diese als AWS::Events::ApiDestination Ressourcen in die CloudFormation Vorlage aufgenommen.

- AWS::Events::EventBus

  Wenn Ihre Pipes einen Event-Bus als Ziel EventBridge enthalten, nehmen Sie ihn als AWS::Events::EventBus Ressource in die CloudFormation Vorlage auf.

- AWS::IAM::Role

  Wenn Sie bei der Konfiguration der Pipe eine neue Ausführungsrolle EventBridge erstellt haben, können Sie wählen, ob Sie diese Rolle als AWS::IAM::Role Ressource in die Vorlage EventBridge aufnehmen möchten. EventBridge beinhaltet keine Rollen, die Sie erstellen. (In beiden Fällen enthält die `RoleArn` Eigenschaft der AWS::Pipes::Pipe Ressource den ARN der Rolle.)

Überlegungen bei der Verwendung von CloudFormation Vorlagen, die aus EventBridge Pipes generiert wurden

Berücksichtigen Sie bei der Verwendung einer CloudFormation Vorlage, aus der Sie generiert haben, die folgenden Faktoren EventBridge:

- EventBridge enthält keine Passwörter in der generierten Vorlage.

  Sie können die Vorlage so bearbeiten, dass sie Vorlagenparameter enthält, mit denen Benutzer Passwörter oder andere vertrauliche Informationen angeben können, wenn sie die Vorlage zum Erstellen oder Aktualisieren eines CloudFormation Stacks verwenden.

  Darüber hinaus können Benutzer Secrets Manager verwenden, um ein Secret in der gewünschten Region zu erstellen und dann die generierte Vorlage so zu bearbeiten, dass dynamische Parameter eingesetzt werden.

- Die Ziele in der generierten Vorlage bleiben genau so, wie sie in der ursprünglichen Pipe angegeben wurden. Dies kann zu regionsübergreifenden Problemen führen, wenn Sie die Vorlage nicht entsprechend bearbeiten, bevor Sie sie zum Erstellen von Stacks in anderen Regionen verwenden.

  Darüber hinaus erstellt die generierte Vorlage die nachgelagerten Ziele nicht automatisch.

Eine CloudFormation Vorlage aus EventBridge Pipes generieren

Gehen Sie wie folgt vor, um mithilfe der EventBridge Konsole eine CloudFormation Vorlage aus einer oder mehreren Pipes zu generieren:

Um eine CloudFormation Vorlage aus einer oder mehreren Leitungen zu generieren

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Pipes aus.
3. Wählen Sie unter Pipes eine oder mehrere Pipes aus, die Sie in die generierte CloudFormation Vorlage aufnehmen möchten.

   Bei einer einzelnen Pipe können Sie auch den Namen der Pipe auswählen, um die Seite mit den Details zu der Pipe anzuzeigen.
4. Wählen Sie CloudFormation Vorlage und wählen Sie dann aus, in welchem Format Sie die Vorlage generieren EventBridge möchten: JSON oder YAML.

   EventBridge zeigt die Vorlage an, die im ausgewählten Format generiert wurde.
5. Wenn Sie eine neue Ausführungsrolle für eine der ausgewählten Pipes EventBridge erstellt haben und diese Rollen in die Vorlage aufnehmen EventBridge möchten, wählen Sie IAM Rollen einbeziehen, die von der Konsole in Ihrem Namen erstellt wurden.
6. EventBridge bietet Ihnen die Möglichkeit, die Vorlagendatei herunterzuladen oder die Vorlage in die Zwischenablage zu kopieren.

   - Wählen Sie zum Herunterladen der Vorlagendatei Herunterladen aus.
   - Wählen Sie zum Kopieren der Vorlage in die Zwischenablage Kopieren aus.
7. Wählen Sie zum Beenden der Vorlage Abbrechen aus.

Festlegen von Anwendungen als regional fehlertolerant mit globalen Endpunkten und der Ereignisreplikation

Sie können die Verfügbarkeit Ihrer Anwendung mit den EventBridge globalen Endpunkten von Amazon verbessern. Mit globalen Endpunkten können Sie Ihre Anwendung ohne zusätzliche Kosten regional fehlertolerant machen. Zunächst weisen Sie dem Endpunkt eine Amazon-Route-53-Zustandsprüfung zu. Wenn ein Failover eingeleitet wird, meldet die Zustandsprüfung einen fehlerhaften Zustand. Innerhalb weniger Minuten nach der Einleitung des Failovers werden alle benutzerdefinierten Ereignisse an einen Event Bus in der sekundären Region weitergeleitet und von diesem Event Bus verarbeitet. Sobald die Zustandsprüfung einen fehlerfreien Zustand meldet, werden die Ereignisse vom Event Bus in der primären Region verarbeitet.

Wenn Sie globale Endpunkte verwenden, können Sie die Ereignisreplikation aktivieren. Bei der Ereignisreplikation werden alle benutzerdefinierten Ereignisse mithilfe verwalteter Regeln an die Event Buses in der primären und sekundären Region gesendet.

> **Note**
>
> Wenn Sie benutzerdefinierte Buses verwenden, benötigen Sie in jeder Region einen benutzerdefinierten Bus mit demselben Namen und demselben Konto, damit der Failover ordnungsgemäß funktioniert.

Themen

Recovery Time und Recovery Point Objectives

Das Recovery Time Objective (RTO) ist die Zeit, die benötigt wird, bis die sekundäre Region nach einem Fehler mit dem Empfang von Ereignissen beginnt. Bei RTO umfasst der Zeitraum den Zeitraum für das Auslösen von CloudWatch Alarmen und das Aktualisieren des Status für Route 53-Zustandsprüfungen. Das Recovery Point Objective (RPO) ist das Maß für die Daten, die bei einem Fehler unbearbeitet bleiben. Bei RPO umfasst die Zeit Ereignisse, die nicht in die sekundäre Region repliziert werden und in der primären Region feststecken, bis der Service oder die Region wiederhergestellt ist. Wenn Sie bei globalen Endpunkten unsere ausführlichen Anleitungen zur Alarmkonfiguration befolgen, können Sie davon ausgehen, dass RTO sowie RPO 360 Sekunden und maximal 420 Sekunden betragen.

Ereignisreplikation

Ereignisse werden in der sekundären Region asynchron verarbeitet. Dies bedeutet, dass nicht garantiert werden kann, dass Ereignisse in beiden Regionen gleichzeitig verarbeitet werden. Wenn ein Failover ausgelöst wird, werden die Ereignisse von der sekundären Region verarbeitet und von der primären Region verarbeitet, sobald diese verfügbar ist. Wenn Sie die Ereignisreplikation aktivieren, erhöhen sich Ihre monatlichen Kosten. Weitere Informationen finden Sie unter EventBridgeAmazon-Preise

Aus den folgenden Gründen empfehlen wir, die Ereignisreplikation bei der Einrichtung globaler Endpunkte zu aktivieren:

- Mithilfe der Ereignisreplikation können Sie überprüfen, ob Ihre globalen Endpunkte korrekt konfiguriert sind. Auf diese Weise können Sie sicherstellen, dass Sie im Falle eines Failovers abgesichert sind.
- Für die automatische Wiederherstellung nach einem Failover-Ereignis ist die Ereignisreplikation erforderlich. Wenn Sie die Ereignisreplikation nicht aktiviert haben, müssen Sie die Route-53-Zustandsprüfung manuell auf den fehlerfreien Zustand zurücksetzen, bevor die Ereignisse in die primäre Region zurückkehren.

Replizierte Ereignisnutzlast

Im Folgenden sehen Sie ein Beispiel für eine replizierte Ereignisnutzlast:

> **Note**
>
> Für `region` ist die Region aufgeführt, aus der das Ereignis repliziert wurde.

```
{
    "version": "0",
    "id": "a908baa3-65e5-ab77-367e-527c0e71bbc2",
    "detail-type": "Test",
    "source": "test.service.com",
    "account": "0123456789",
    "time": "1900-01-01T00:00:00Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:events:us-east-1:0123456789:endpoint/MyEndpoint"
    ],
    "detail": {
        "a": "b"
    }
}
```

Erstellen eines globalen Endpunkts

Gehen Sie folgendermaßen vor, um einen globalen Endpunkt einzurichten:

1. Stellen Sie sicher, dass Sie sowohl in der primären als auch in der sekundären Region über passende Event Buses und Regeln verfügen.
2. Erstellen Sie eine Route-53-Zustandsprüfung, um Ihre Event Buses zu überwachen. Wenn Sie Unterstützung bei der Erstellung Ihrer Zustandsprüfung benötigen, wählen Sie bei der Erstellung Ihres globalen Endpunkts die Option Neue Zustandsprüfung.
3. Erstellen Sie Ihren globalen Endpunkt.

Sobald Sie die Route-53-Zustandsprüfung eingerichtet haben, können Sie einen globalen Endpunkt erstellen.

So erstellen Sie einen globalen Endpunkt mit der Konsole

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.

2. Klicken Sie im Navigationsbereich auf Globale Endpunkte.
3. Klicken Sie auf Endpunkt erstellen.
4. Geben Sie einen Namen und eine Beschreibung für den Endpunkt ein.
5. Wählen Sie für Event Bus in primärer Region den Event Bus aus, dem der Endpunkt zugeordnet werden soll.
6. Wählen Sie für Sekundäre Region die Region aus, in die Sie Ereignisse im Falle eines Failovers weiterleiten möchten.

   > **Note**
   >
   > Die Option Event Bus in sekundärer Region wird automatisch gefüllt und kann nicht bearbeitet werden.

7. Wählen Sie für Route-53-Zustandsprüfung zum Auslösen von Failover und Wiederherstellung die Zustandsprüfung aus, die der Endpunkt überwachen soll. Wenn Sie noch keinen Integritätscheck haben, wählen Sie Neue Integritätsprüfung aus, um die AWS CloudFormation Konsole zu öffnen und mithilfe einer CloudFormation Vorlage eine Integritätsprüfung zu erstellen.

   > **Note**
   >
   > Fehlende Daten führen dazu, dass die Zustandsprüfung fehlschlägt. Wenn Sie Ereignisse nur zeitweise versenden müssen, sollten Sie eine längere MinimumEvaluationPeriodVersion in Betracht ziehen oder fehlende Daten als „fehlend“ und nicht als „fehlerhaft“ behandeln.

8. (Optional) Gehen Sie für Ereignisreplikation wie folgt vor:

   a. Wählen Sie Ereignisreplikation aktiviert aus.
   b. Wählen Sie für Ausführungsrolle aus, ob Sie eine neue AWS Identity and Access Management -Rolle erstellen oder eine vorhandene Rolle verwenden möchten. Gehen Sie wie folgt vor:

      - Wählen Sie Create a new role for this specific resource aus. Optional können Sie den Rollennamen aktualisieren, um eine neue Rolle zu erstellen.
      - Wählen Sie Vorhandene Rolle verwenden aus. Wählen Sie dann für Ausführungsrolle die gewünschte Rolle aus.

9. Wählen Sie Erstellen.

So erstellen Sie einen globalen Endpunkt mit der API

Informationen zum Erstellen eines globalen Endpunkts mithilfe der EventBridge API finden Sie CreateEndpointin der Amazon EventBridge API-Referenz.

So erstellen Sie einen globalen Endpunkt mit AWS CloudFormation

Informationen zum Erstellen eines globalen Endpunkts mithilfe der AWS CloudFormation API finden Sie AWS::Events::Endpointsim AWS CloudFormation Benutzerhandbuch.

Arbeiten mit globalen Endpunkten mithilfe eines SDK AWS

> **Note**
>
> Unterstützung für C++ ist bald verfügbar.

Beachten Sie bei der Verwendung eines AWS SDK für die Arbeit mit globalen Endpunkten Folgendes:

- Sie müssen die AWS Common Runtime (CRT) -Bibliothek für Ihr spezielles SDK installiert haben. Wenn Sie das CRT nicht installiert haben, erhalten Sie eine Ausnahmenachricht, die angibt, was installiert werden muss. Weitere Informationen finden Sie hier:
  - AWS Common Runtime (CRT)-Bibliotheken
  - awslabs/ aws-crt-java
  - awslabs/ aws-crt-nodejs
  - awslabs/ aws-crt-python
- Sobald Sie einen globalen Endpunkt erstellt haben, müssen Sie allen `PutEvents`-Aufrufen, die Sie verwenden, die `endpointId` und den `EventBusName` hinzufügen.
- Globale Endpunkte unterstützen Signature Version 4A. Diese Version von SigV4 ermöglicht das Signieren von Anforderungen für mehrere AWS-Regionen. Dies ist nützlich bei API-Vorgängen, die zu Datenzugriff von einer von mehreren Regionen führen können. Wenn Sie das AWS SDK verwenden, geben Sie Ihre Anmeldeinformationen ein und für Anfragen an globale Endpunkte wird Signature Version 4A ohne zusätzliche Konfiguration verwendet. Weitere Informationen zu SigV4A finden Sie unter Signing AWS API Requests (Signieren von API-Anforderungen) in der AWS Allgemeinen Referenz.

Wenn Sie temporäre Anmeldeinformationen vom globalen AWS STS Endpunkt (sts.amazonaws.com) anfordern, werden Anmeldeinformationen bereitgestellt, die AWS STS SigV4a standardmäßig nicht unterstützen. Weitere Informationen finden Sie AWS STS im Benutzerhandbuch unter Verwaltung in einer AWS Region.AWS Identity and Access Management

Verfügbare Regionen

Die folgenden Regionen unterstützen globale Endpunkten:

- USA Ost (Nord-Virginia)
- USA Ost (Ohio)
- USA West (Nordkalifornien)
- USA West (Oregon)
- Kanada (Zentral)
- Europe (Frankfurt)
- Europa (Irland)
- Europa (London)
- Europa (Milan)
- Europe (Paris)
- Europa (Stockholm)
- Asien-Pazifik (Mumbai)
- Asia Pacific (Osaka)
- Asia Pacific (Seoul)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Tokio)
- Südamerika (São Paulo)

Bewährte Methoden für die Arbeit mit globalen Amazon-EventBridge-Endpunkten

Die folgenden bewährten Methoden werden empfohlen, wenn Sie globale Endpunkte einrichten.

Themen

Aktivieren der Ereignisreplikation

Es wird dringend empfohlen, die Replikation zu aktivieren und Ihre Ereignisse in der sekundären Region zu verarbeiten, die Sie Ihrem globalen Endpunkt zuweisen. Dadurch wird sichergestellt, dass Ihre Anwendung in der sekundären Region korrekt konfiguriert ist. Sie sollten auch die Replikation aktivieren, um eine automatische Wiederherstellung in der primären Region sicherzustellen, nachdem ein Problem behoben wurde.

Ereignis-IDs können sich bei API-Aufrufen ändern, sodass Sie für die Korrelation von Ereignissen in verschiedenen Regionen eine unveränderliche, eindeutige Kennung benötigen. Konsumenten sollten auch im Hinblick auf Idempotenz konzipiert werden. Wenn Sie Ereignisse replizieren oder sie aus Archiven wiederholen, gibt es auf diese Weise keine Nebeneffekte, wenn die Ereignisse in beiden Regionen verarbeitet werden.

Verhindern der Drosselung von Ereignissen

Um zu verhindern, dass Ereignisse gedrosselt werden, empfehlen wir, Ihre Limits für `PutEvents` und Ziele so zu aktualisieren, dass sie in allen Regionen einheitlich sind.

Verwenden von Subscriber-Metriken bei Amazon-Route-53-Zustandsprüfungen

Geben Sie in Ihren Amazon-Route-53-Zustandsprüfungen keine Subscriber-Metriken an. Das Einbeziehen dieser Metriken kann dazu führen, dass Ihr Publisher ein Failover zu sekundären Regionen durchführt, wenn ein Subscriber auf ein Problem stößt, obwohl alle anderen Subscriber in der primären Region fehlerlos geblieben sind. Wenn einer Ihrer Subscriber Ereignisse in der primären Region nicht verarbeitet, sollten Sie die Replikation aktivieren, um sicherzustellen, dass Ihr Subscriber Ereignisse in der sekundären Region erfolgreich verarbeiten kann.

AWS CloudFormation-Vorlage für die Einrichtung der Route-53-Zustandsprüfung

Wenn Sie globale Endpunkte verwenden, benötigen Sie eine Route-53-Zustandsprüfung, um den Status Ihrer Regionen zu überwachen. Die folgende Vorlage definiert einen Amazon-CloudWatch-Alarm und verwendet ihn, um eine Route-53-Zustandsprüfung zu definieren.

Themen

AWS CloudFormation-Vorlage für die Definition einer Route-53-Zustandsprüfung

Verwenden Sie die folgende Vorlage, um Ihre Route-53-Zustandsprüfung zu definieren.

```
Description: |-
  Global endpoints health check that will fail when the average Amazon EventBridge
  latency is above 30 seconds for a duration of 5 minutes. Note, missing data will
  cause the health check to fail, so if you only send events intermittently, consider
  changing the heath check to use a longer evaluation period or instead treat missing
  data as 'missing' instead of 'breaching'.

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      - Label:
          default: "Global endpoint health check alarm configuration"
        Parameters:
          - HealthCheckName
          - HighLatencyAlarmPeriod
          - MinimumEvaluationPeriod
          - MinimumThreshold
          - TreatMissingDataAs
    ParameterLabels:
      HealthCheckName:
        default: Health check name
      HighLatencyAlarmPeriod:
```

```
        default: High latency alarm period
      MinimumEvaluationPeriod:
        default: Minimum evaluation period
      MinimumThreshold:
        default: Minimum threshold
      TreatMissingDataAs:
        default: Treat missing data as

Parameters:
  HealthCheckName:
    Description: Name of the health check
    Type: String
    Default: LatencyFailuresHealthCheck
  HighLatencyAlarmPeriod:
    Description: The period, in seconds, over which the statistic is applied. Valid
 values are 10, 30, 60, and any multiple of 60.
    MinValue: 10
    Type: Number
    Default: 60
  MinimumEvaluationPeriod:
    Description: The number of periods over which data is compared to the specified
 threshold. You must have at least one evaluation period.
    MinValue: 1
    Type: Number
    Default: 5
  MinimumThreshold:
    Description: The value to compare with the specified statistic.
    Type: Number
    Default: 30000
  TreatMissingDataAs:
    Description: Sets how this alarm is to handle missing data points.
    Type: String
    AllowedValues:
      - breaching
      - notBreaching
      - ignore
      - missing
    Default: breaching

Mappings:
  "InsufficientDataMap":
    "missing":
      "HCConfig": "LastKnownStatus"
    "breaching":
```

```
      "HCConfig": "Unhealthy"

Resources:
  HighLatencyAlarm:
      Type: AWS::CloudWatch::Alarm
      Properties:
        AlarmDescription: High Latency in Amazon EventBridge
        MetricName: IngestionToInvocationStartLatency
        Namespace: AWS/Events
        Statistic: Average
        Period: !Ref HighLatencyAlarmPeriod
        EvaluationPeriods: !Ref MinimumEvaluationPeriod
        Threshold: !Ref MinimumThreshold
        ComparisonOperator: GreaterThanThreshold
        TreatMissingData: !Ref TreatMissingDataAs

  LatencyHealthCheck:
      Type: AWS::Route53::HealthCheck
      Properties:
        HealthCheckTags:
          - Key: Name
            Value: !Ref HealthCheckName
        HealthCheckConfig:
          Type: CLOUDWATCH_METRIC
          AlarmIdentifier:
            Name:
              Ref: HighLatencyAlarm
            Region: !Ref AWS::Region
          InsufficientDataHealthStatus: !FindInMap [InsufficientDataMap, !Ref
 TreatMissingDataAs, HCConfig]

Outputs:
  HealthCheckId:
    Description: The identifier that Amazon Route 53 assigned to the health check when
 you created it.
    Value: !GetAtt LatencyHealthCheck.HealthCheckId
```

Ereignis-IDs können sich bei API-Aufrufen ändern, sodass Sie für die Korrelation von Ereignissen in verschiedenen Regionen eine unveränderliche, eindeutige Kennung benötigen. Konsumenten sollten auch im Hinblick auf Idempotenz konzipiert werden. Wenn Sie Ereignisse replizieren oder sie aus Archiven wiederholen, gibt es auf diese Weise keine Nebeneffekte, wenn die Ereignisse in beiden Regionen verarbeitet werden.

Eigenschaften der Vorlage für einen CloudWatch-Alarm

> **Note**
>
> Berücksichtigen Sie bei allen **`editable`**-Feldern Ihren Durchsatz pro Sekunde. Wenn Sie Ereignisse nur sporadisch versenden, sollten Sie erwägen, die Zustandsprüfung so zu ändern, dass ein längerer Auswertungszeitraum verwendet wird, oder fehlende Daten als `missing` und nicht `breaching` zu behandeln.

Die folgenden Eigenschaften werden im Abschnitt CloudWatch-Alarm der Vorlage verwendet:

Metrik	Beschreibung
`AlarmDescription`	Die Beschreibung des Alarms. Standard: **`High Latency in Amazon EventBridge`**
`MetricName`	Der Name der dem Alarm zugehörigen Metrik. Dies ist für einen Alarm auf der Grundlage einer Metrik erforderlich. Für einen Alarm auf der Grundlage eines mathematischen Ausdrucks verwenden Sie stattdessen `Metrics` und Sie können `MetricName` nicht angeben. Standard: IngestionToInvocationStartLatency
`Namespace`	Der Namespace der Metrik, die dem Alarm zugeordnet ist. Dies ist für einen Alarm auf der Grundlage einer Metrik erforderlich. Für einen Alarm auf der Grundlage eines mathematischen Ausdrucks können Sie `Namespace` nicht angeben. Verwenden Sie stattdessen `Metrics`. Standard: `AWS/Events`
`Statistic`	Die Statistik für die Metrik, die mit dem Alarm verbunden ist, außer Perzentil. Standard: Average

Metrik	Beschreibung
`Period`	Der Zeitraum in Sekunden, in dem die Statistik angewendet wird. Dies ist für einen Alarm auf der Grundlage einer Metrik erforderlich. Gültige Werte sind 10, 30, 60 und jedes Vielfache von 60. Standard: **`60`**
`EvaluationPeriods`	Die Anzahl der Zeiträume, über die Daten mit dem angegebenen Schwellenwert verglichen werden. Wenn Sie einen Alarm einstellen, der erfordert, dass mehrere aufeinander folgende Datenpunkte verletzt werden, um den Alarm auszulösen, gibt dieser Wert diese Zahl an. Wenn Sie einen „M out of N“-Alarm einstellen, ist dieser Wert das N und `DatapointsToAlarm` das M. Standard: **`5`**
`Threshold`	Der Wert für den Vergleich mit der angegebenen Statistik. Standard: **`30,000`**
`ComparisonOperator`	Die arithmetische Operation, die beim Vergleichen der angegebenen Statistik und des Schwellenwerts zu verwenden ist. Der angegebene Statistikwert wird als erster Operand verwendet. Standard: `GreaterThanThreshold`
`TreatMissingData`	Legt fest, wie dieser Alarm fehlende Datenpunkte behandeln soll. Gültige Werte: `breaching` , `notBreaching` , `ignore` und `missing` Standard: `breaching`

Eigenschaften der Vorlage für eine Route-53-Zustandsprüfung

> **Note**
>
> Berücksichtigen Sie bei allen **`editable`**-Feldern Ihren Durchsatz pro Sekunde. Wenn Sie Ereignisse nur sporadisch versenden, sollten Sie erwägen, die Zustandsprüfung so zu

ändern, dass ein längerer Auswertungszeitraum verwendet wird, oder fehlende Daten als `missing` und nicht `breaching` zu behandeln.

Die folgenden Eigenschaften werden im Abschnitt Route-53-Zustandsprüfung der Vorlage verwendet:

Metrik	Beschreibung
`HealthCheckName`	Die Bezeichnung der Zustandsprüfung. Standard: **`LatencyFailuresHealthCheck`**
`InsufficientDataHealthStatus`	Der Status, der der Zustandsprüfung von Amazon Route 53 zugewiesen werden soll, wenn CloudWatch unzureichende Daten über die Metrik besitzt, um den Alarmzustand zu bestimmen Zulässige Werte: • `Healthy`: Route 53 betrachtet die Zustandsprüfung als fehlerfrei. • `Unhealthy` : Route 53 betrachtet die Zustandsprüfung als fehlerhaft. • `LastKnownStatus` : Route 53 verwendet den Status der Zustandsprüfung des letzten Zeitpunkts, zu dem CloudWatch über ausreichend Daten zur Ermittlung des Alarms verfügt hat. Bei neuen Zustandsprüfungen, die keinen letzten bekannten Status haben, ist der Standardstatus für die Zustandsprüfung „fehlerfrei“. Standard: Unhealthy **Note** Dieses Feld wird auf der Grundlage der Eingabe in das `TreatMissingData` -Feld aktualisiert. Wenn `TreatingMissingData` auf `Missing` gesetzt ist, wird es auf `LastKnownStatus` aktualisiert. Wenn `TreatingMissingData` auf `Breaching` gesetzt ist, wird es auf `Unhealthy` aktualisiert.

EventBridge Amazon-Schemas

Ein Schema definiert die Struktur von Ereignissen, an die gesendet EventBridge werden. EventBridge stellt Schemas für alle Ereignisse bereit, die von AWS Diensten generiert werden. Sie können auch benutzerdefinierte Schemata erstellen oder hochladen oder direkt aus Ereignissen in einem Event Bus Schemata ableiten. Sobald Sie über ein Schema für ein Ereignis verfügen, können Sie Codebindungen für gängige Programmiersprachen herunterladen und die Entwicklung beschleunigen. Sie können mit Codebindungen für Schemas arbeiten und Schemas von der EventBridge Konsole aus verwalten, indem Sie die API verwenden, oder direkt in Ihrer IDE, indem Sie die Toolkits verwenden. AWS Wenn Sie Serverless-Apps erstellen möchten, die Ereignisse verwenden, verwenden Sie AWS Serverless Application Model.

> **Note**
>
> Wenn Sie das Eingabe-Transformator-Feature verwenden, wird das ursprüngliche Ereignis durch die Schemaerkennung abgeleitet, nicht das transformierte Ereignis, das an das Ziel gesendet wird.

EventBridge unterstützt sowohl die Formate OpenAPI 3 als auch JSONSchema Draft4.

Für AWS Toolkit for JetBrains und AWS Toolkit for VS Code können Sie Schemas durchsuchen oder suchen und Codebindungen für Schemas direkt in Ihrer IDE herunterladen.

Das folgende Video gibt einen Überblick über Schemata und Schemaregistrierungen: Verwenden der Schemaregistrierung

Themen

Maskieren von Eigenschaftswerten der Schemaregistrierungs-API

Einige Eigenschaftswerte von Ereignissen, die zur Erstellung einer Schemaregistrierung verwendet werden, können vertrauliche Kundeninformationen enthalten. Zum Schutz der Kundeninformationen werden die Werte mit Sternchen (*) maskiert. Da wir diese Werte maskieren, EventBridge empfiehlt es sich, keine Anwendungen zu erstellen, die explizit von den folgenden Eigenschaften oder deren Werten abhängen:

- CreateSchema— Die `Content` Eigenschaft des Körpers `requestParameters`
- GetDiscoveredSchema— Das `Events` Eigentum des `requestParameters` Körpers und das `Content` Eigentum des `responseElements` Körpers
- SearchSchemas— Das `keywords` Eigentum des `requestParameters`
- UpdateSchema— Das `Content` Eigentum der `requestParameters`

Ein EventBridge Amazon-Schema finden

EventBridge beinhaltet Schemas für alle AWS Dienste, die Ereignisse generieren. Sie finden diese Schemas in der EventBridge Konsole oder mithilfe der API-Aktion. `SearchSchemas`

Um Schemas für AWS Dienste in der Konsole zu finden EventBridge

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Schemas aus.
3. Wählen Sie auf der Seite Schemata die Option AWS -Ereignisschemaregistrierung aus.
   <result>

   Die erste Seite der verfügbaren Schemas wird angezeigt.
   </result>
4. Um ein Schema zu finden, geben Sie unter AWS Suchereignisschemas einen Suchbegriff ein.

   Eine Suche gibt Übereinstimmungen sowohl für den Namen als auch für den Inhalt der verfügbaren Schemata zurück und zeigt dann an, welche Versionen des Schemas Übereinstimmungen enthalten.
5. Öffnen Sie ein Ereignisschema, indem Sie den Namen des Schemas auswählen.

EventBridge Amazon-Schemaregister

Schemaregistrierungen sind Container für Schemata. Schemaregistrierungen sammeln und organisieren Schemata so, dass sich Ihre Schemata in logischen Gruppen befinden. Die standardmäßigen Schemaregistrierungen sind:

- Alle Schemas — Alle Schemas aus den Registern „ AWS Ereignis“, „Ermittelt“ und „Benutzerdefiniertes Schema“.
- AWS Registrierung des Ereignisschemas — Die integrierten Schemas.
- Erkannte Schemaregistrierung – Die Schemata, die bei der Schemaerkennung erkannt wurden

Sie können benutzerdefinierte Registrierungen erstellen, um die Schemata zu organisieren, die Sie erstellen oder hochladen.

So erstellen Sie eine benutzerdefinierte Registrierung

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Schemata und dann Registrierung erstellen aus.
3. Geben Sie auf der Seite Registrierungsdetails unter Name einen Namen ein.
4. (Optional) Geben Sie eine Beschreibung für die neue Registrierung ein.
5. Wählen Sie Erstellen.

Wenn Sie in der neuen Registrierung ein benutzerdefiniertes Schema erstellen möchten, wählen Sie Benutzerdefiniertes Schema erstellen aus. Wenn Sie der Registrierung ein Schema hinzufügen möchten, wählen Sie diese Registrierung aus, wenn Sie ein neues Schema erstellen.

Wenn Sie eine Registrierung mithilfe der API erstellen möchten, verwenden Sie `CreateRegistry`. Weitere Informationen finden Sie unter Amazon EventBridge Schema Registry API Reference.

Informationen zur Verwendung der EventBridge Schemaregistrierung finden AWS CloudFormation Sie unter EventSchemas Resource Type Reference unter AWS CloudFormation.

Ein EventBridge Amazon-Schema erstellen

Sie erstellen Schemata, indem Sie JSON-Dateien mit der OpenAPI-Spezifikation oder der JSONSchema-Draft4-Spezifikation verwenden. Sie können Ihre eigenen Schemas erstellen oder hochladen, EventBridge indem Sie eine Vorlage verwenden oder ein Schema generieren, das auf dem JSON eines Ereignisses basiert. Sie können das Schema auch aus Ereignissen in einem Event Bus ableiten. Verwenden Sie die API-Aktion, um mithilfe der EventBridge Schema Registry API ein Schema zu erstellen. `CreateSchema`

Wenn Sie zwischen den Formaten OpenAPI 3 und JSONSchema Draft4 wählen, sollten Sie die folgenden Unterschiede berücksichtigen:

- Das Format JSONSchema unterstützt zusätzliche Schlüsselwörter, die in OpenAPI nicht unterstützt werden, wie z. B. `$schema, additionalItems`.
- Es gibt geringfügige Unterschiede in der Art und Weise, wie Schlüsselwörter behandelt werden, z. B. `type` und `format`.
- OpenAPI unterstützt keine JSONSchema-Hyper-Schema-Hyperlinks in JSON-Dokumenten.
- Tools für OpenAPI konzentrieren sich in der Regel auf die Erstellungszeit, während sich Tools für JSONSchema eher auf Laufzeitoperationen konzentrieren, wie z. B. Client-Tools für die Schemavalidierung.

Wir empfehlen, das JSONSchema-Format zu verwenden, um die clientseitige Validierung zu implementieren, sodass Ereignisse, die gesendet werden, dem Schema EventBridge entsprechen. Sie können JSONSchema verwenden, um einen Vertrag für gültige JSON-Dokumente zu definieren, und dann eine JSON-Schemavalidierung verwenden, bevor Sie die zugehörigen Ereignisse senden.

Nachdem Sie über ein neues Schema verfügen, können Sie Codebindungen herunterladen, um Anwendungen für Ereignisse mit diesem Schema zu erstellen.

Themen

- Erstellen eines Schemas mithilfe einer Vorlage
- Bearbeiten einer Schemavorlage direkt in der Konsole
- Erstellen eines Schemas aus dem JSON eines Ereignisses
- Erstellen eines Schemas aus Ereignissen in einem Event Bus

Erstellen eines Schemas mithilfe einer Vorlage

Sie können ein Schema anhand einer Vorlage erstellen oder indem Sie eine Vorlage direkt in der Konsole bearbeiten. EventBridge Wenn Sie die Vorlage erhalten möchten, laden Sie sie von der Konsole herunter. Sie können die Vorlage so bearbeiten, dass das Schema Ihren Ereignissen entspricht. Laden Sie dann die neue Vorlage über die Konsole hoch.

So laden Sie die Schemavorlage herunter

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Schema registry (Schemaregistrierung) aus.
3. Wählen Sie im Abschnitt Getting started (Erste Schritte) unter Schema template (Schemavorlage) die Option Download (Herunterladen) aus.

Alternativ können Sie die JSON-Vorlage aus dem folgenden Codebeispiel kopieren.

```
{
    "openapi": "3.0.0",
    "info": {
      "version": "1.0.0",
      "title": "Event"
    },
    "paths": {},
    "components": {
      "schemas": {
        "Event": {
          "type": "object",
          "properties": {
            "ordinal": {
              "type": "number",
              "format": "int64"
            },
            "name": {
              "type": "string"
            },
            "price": {
              "type": "number",
              "format": "double"
            },
            "address": {
              "type": "string"
```

```
            },
            "comments": {
              "type": "array",
              "items": {
                "type": "string"
              }
            },
            "created_at": {
              "type": "string",
              "format": "date-time"
            }
          }
        }
      }
    }
  }
```

So laden Sie eine Schemavorlage hoch

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Schemata und dann Schema erstellen aus.
3. (Optional) Wählen Sie eine Schemaregistrierung aus oder erstellen Sie sie.
4. Geben Sie unter Schemadetails einen Namen für das Schema ein.
5. (Optional) Geben Sie eine Beschreibung für das Schema ein.
6. Wählen Sie für Schematyp entweder OpenAPI 3.0 oder JSON Schema Draft 4 aus.
7. Ziehen Sie auf der Registerkarte Erstellen im Textfeld Ihre Schemadatei in das Textfeld oder fügen Sie die Schemaquelle ein.
8. Wählen Sie Erstellen aus.

Bearbeiten einer Schemavorlage direkt in der Konsole

So bearbeiten Sie ein Schema in der Konsole

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Schemata und dann Schema erstellen aus.
3. (Optional) Wählen Sie eine Schemaregistrierung aus oder erstellen Sie sie.
4. Geben Sie unter Schemadetails einen Namen für das Schema ein.

5. Wählen Sie für Schematyp entweder OpenAPI 3.0 oder JSON Schema Draft 4 aus.
6. (Optional) Geben Sie eine Beschreibung für das zu erstellende Schema ein.
7. Wählen Sie auf der Registerkarte Erstellen die Option Vorlage laden aus.
8. Bearbeiten Sie die Vorlage im Textfeld so, dass das Schema Ihren Ereignissen entspricht.
9. Wählen Sie Erstellen aus.

Erstellen eines Schemas aus dem JSON eines Ereignisses

Wenn Sie über die JSON-Datei eines Ereignisses verfügen, können Sie automatisch ein Schema für diesen Ereignistyp erstellen.

So erstellen Sie ein Schema basierend auf dem JSON eines Ereignisses

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Schemata und dann Schema erstellen aus.
3. (Optional) Wählen Sie eine Schemaregistrierung aus oder erstellen Sie sie.
4. Geben Sie unter Schema details (Schemadetails) einen Namen für Ihr Schema ein.
5. (Optional) Geben Sie eine Beschreibung für das erstellte Schema ein.
6. Wählen Sie für Schematyp die Option OpenAPI 3.0 aus.

   Sie können JSONSchema nicht verwenden, wenn Sie ein Schema aus dem JSON eines Ereignisses erstellen.
7. Wählen Sie Discover from JSON (Aus JSON-Code erkennen) aus.
8. Fügen Sie in das Textfeld unter JSON, die JSON-Quelle eines Ereignisses ein oder ziehen Sie sie dorthin.

   Sie könnten beispielsweise die Quelle aus diesem AWS Step Functions Ereignis für eine fehlgeschlagene Ausführung einfügen.

```
{
    "version": "0",
    "id": "315c1398-40ff-a850-213b-158f73e60175",
    "detail-type": "Step Functions Execution Status Change",
    "source": "aws.states",
    "account": "012345678912",
    "time": "2019-02-26T19:42:21Z",
    "region": "us-east-1",
```

```
    "resources": [
      "arn:aws:states:us-east-1:012345678912:execution:state-machine-
name:execution-name"
    ],
    "detail": {
        "executionArn": "arn:aws:states:us-east-1:012345678912:execution:state-
machine-name:execution-name",
        "stateMachineArn": "arn:aws:states:us-
east-1:012345678912:stateMachine:state-machine",
        "name": "execution-name",
        "status": "FAILED",
        "startDate": 1551225146847,
        "stopDate": 1551225151881,
        "input": "{}",
        "output": null
    }
}
```

9. Klicken Sie auf Discover schema (Schema erkennen).
10. EventBridge generiert ein OpenAPI-Schema für das Ereignis. Beispielsweise wird das folgende Schema für das vorhergehende Step-Functions-Ereignis generiert.

```
{
  "openapi": "3.0.0",
  "info": {
    "version": "1.0.0",
    "title": "StepFunctionsExecutionStatusChange"
  },
  "paths": {},
  "components": {
    "schemas": {
      "AWSEvent": {
        "type": "object",
        "required": ["detail-type", "resources", "detail", "id", "source", "time",
 "region", "version", "account"],
        "x-amazon-events-detail-type": "Step Functions Execution Status Change",
        "x-amazon-events-source": "aws.states",
        "properties": {
          "detail": {
            "$ref": "#/components/schemas/StepFunctionsExecutionStatusChange"
          },
          "account": {
            "type": "string"
```

```
          },
          "detail-type": {
            "type": "string"
          },
          "id": {
            "type": "string"
          },
          "region": {
            "type": "string"
          },
          "resources": {
            "type": "array",
            "items": {
              "type": "string"
            }
          },
          "source": {
            "type": "string"
          },
          "time": {
            "type": "string",
            "format": "date-time"
          },
          "version": {
            "type": "string"
          }
        }
      },
      "StepFunctionsExecutionStatusChange": {
        "type": "object",
        "required": ["output", "input", "executionArn", "name", "stateMachineArn",
"startDate", "stopDate", "status"],
        "properties": {
          "executionArn": {
            "type": "string"
          },
          "input": {
            "type": "string"
          },
          "name": {
            "type": "string"
          },
          "output": {},
          "startDate": {
```

```
            "type": "integer",
            "format": "int64"
          },
          "stateMachineArn": {
            "type": "string"
          },
          "status": {
            "type": "string"
          },
          "stopDate": {
            "type": "integer",
            "format": "int64"
          }
        }
      }
    }
  }
}
```

11. Nachdem das Schema generiert wurde, wählen Sie Erstellen aus.

Erstellen eines Schemas aus Ereignissen in einem Event Bus

EventBridge kann Schemas ableiten, indem Ereignisse erkannt werden. Wenn Sie Schemata ableiten möchten, aktivieren Sie die Ereigniserkennung in einem Event Bus und jedes eindeutige Schema wird der Schemaregistrierung hinzugefügt, einschließlich der Schemata für kontoübergreifende Ereignisse. Schemas, die von entdeckt wurden, EventBridge werden in der Registrierung Entdeckte Schemas auf der Seite Schemas angezeigt.

Wenn sich der Inhalt von Ereignissen auf dem Event-Bus ändert, werden neue Versionen des zugehörigen Schemas EventBridge erstellt. EventBridge

> **Note**
>
> Bei Aktivierung der Ereigniserkennung in einem Event Bus können Kosten entstehen. Die ersten fünf Millionen verarbeiteten Ereignisse in jedem Monat sind kostenlos.

> **Note**
>
> EventBridge leitet Schemas standardmäßig aus kontenübergreifenden Ereignissen ab. Sie können dies jedoch deaktivieren, indem Sie die Eigenschaft aktualisieren. `cross-account` Weitere Informationen finden Sie unter API-Referenz zu Discoverers in the EventBridge Schema Registry.

So aktivieren Sie die Schemaerkennung in einem Ereignisbus

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Event Buses (Ereignisbusse) aus.
3. Führen Sie eine der folgenden Aktionen aus:
   - Wenn Sie die Erkennung im Standard-Event-Bus aktivieren möchten, wählen Sie Erkennung starten aus.
   - Wenn Sie die Erkennung im benutzerdefinierten Event Bus aktivieren möchten, aktivieren Sie das Optionsfeld für den benutzerdefinierten Event Bus und wählen Sie dann Erkennung starten aus.

EventBridge Amazon-Codebindungen

Sie können Codebindungen für Ereignisschemas generieren, um die Entwicklung in Golang, Java, Python und zu beschleunigen. TypeScript Codebindungen sind vefügbar für AWS -Serviceereignisse, von Ihnen erstellte Schemata und für Schemata, die Sie basierend auf Ereignissen in einem Event Bus generieren. Sie können Codebindungen für ein Schema mithilfe der EventBridge Konsole, der EventBridge Schema Registry API oder in Ihrer IDE mit einem Toolkit generieren. AWS

Um Codebindungen aus einem Schema zu generieren EventBridge

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Schemas aus.
3. Suchen Sie ein Schema, für das Sie Codebindungen benötigen. Durchsuchen Sie dazu entweder die Schemaregistrierungen oder suchen Sie nach einem Schema.
4. Wählen Sie den Schemanamen aus.
5. Wählen Sie auf der Seite Schemadetails im Abschnitt Version die Option Codebindungen herunterladen aus.
6. Wählen Sie auf der Seite Download code bindings (Codebindungen herunterladen) die Sprache der Codebindungen aus, die Sie herunterladen möchten.
7. Wählen Sie Download (Herunterladen) aus.

   Es kann einige Sekunden dauern, bis der Download beginnt. Bei der heruntergeladenen Datei handelt es sich um eine ZIP-Datei mit Codebindungen für die ausgewählte Sprache.

Mit Amazon EventBridge verwandte Services und Tools

Amazon EventBridge funktioniert mit anderen AWS-Services und -Tools, um Ereignisse zu verarbeiten oder eine Ressource als Ziel einer Regel aufzurufen. Weitere Informationen zu EventBridge-Integrationen mit anderen AWS-Services und -Tools finden Sie unter:

Themen

- Verwenden von Amazon EventBridge mit Schnittstellen-VPC-Endpunkten
- Amazon-EventBridge-Integration mit AWS X-Ray
- Verwendung EventBridge mit dem AWS Integrated Application Test Kit
- EventBridge Amazon-Ressourcen in AWS CloudFormation Stapeln einbeziehen

Verwenden von Amazon EventBridge mit Schnittstellen-VPC-Endpunkten

Wenn Sie Amazon Virtual Private Cloud (Amazon VPC) zum Hosten Ihrer AWS-Ressourcen verwenden, können Sie eine private Verbindung zwischen Ihrer VPC und EventBridge herstellen. Ihre Ressourcen in der VPC können diese Verbindung verwenden, um mit EventBridge zu kommunizieren.

Mit einer VPC haben Sie die Kontrolle über Ihre Netzwerkeinstellungen, wie IP-Adressbereich, Subnetze, Routing-Tabellen und Netzwerk-Gateways. Zum Herstellen einer Verbindung der VPC mit EventBridge definieren Sie einen Schnittstellen-VPC-Endpunkt für EventBridge. Der Endpunkt bietet zuverlässige, skalierbare Konnektivität zu EventBridge, ohne dass ein Internet-Gateway, eine NAT-Instance (Network Address Translation) oder eine VPN-Verbindung erforderlich ist. Weitere Informationen finden Sie unter Was ist Amazon VPC im Benutzerhandbuch zu Amazon VPC.

Schnittstellen-VPC-Endpunkte werden über AWS PrivateLink bereitgestellt, das eine private Kommunikation zwischen AWS-Services unter Verwendung einer Elastic-Network-Schnittstelle mit privaten IP-Adressen ermöglicht. Weitere Informationen finden Sie unter AWS PrivateLink und VPC-Endpunkte.

Wenn Sie einen VPC-Endpunkt einer privaten Schnittstelle nutzen, verwenden benutzerdefinierte Ereignisse, die Ihre VPC an EventBridge sendet, diesen Endpunkt. EventBridge sendet diese Ereignisse dann auf der Grundlage der von Ihnen konfigurierten Regeln und Ziele an andere AWS-Services. Sobald Ereignisse an einen anderen Service gesendet wurden, können Sie sie entweder über den öffentlichen Endpunkt oder einen VPC-Endpunkt für diesen Service empfangen. Wenn Sie beispielsweise eine Regel zum Senden von Ereignissen an eine Amazon-SQS-Warteschlange erstellen, können Sie einen Schnittstellen-VPC-Endpunkt konfigurieren, damit Amazon SQS Nachrichten von dieser Warteschlange in Ihrer VPC empfängt, ohne den öffentlichen Endpunkt zu verwenden.

Verfügbarkeit

Zurzeit unterstützt EventBridge VPC-Endpunkte in den folgenden Regionen:

- US East (Ohio)
- USA Ost (Nord-Virginia)
- USA West (Nordkalifornien)

- USA West (Oregon)
- Afrika (Kapstadt)
- Asien-Pazifik (Mumbai)
- Asien-Pazifik (Hyderabad)
- Asien-Pazifik (Hongkong)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Jakarta)
- Asien-Pazifik (Melbourne)
- Asien-Pazifik (Tokio)
- Asien-Pazifik (Seoul)
- Asien-Pazifik (Osaka)
- Kanada (Zentral)
- Kanada West (Calgary)
- China (Peking)
- China (Ningxia)
- Europa (Frankfurt)
- Europa (Zürich)
- Europa (Irland)
- Europa (London)
- Europa (Milan)
- Europa (Spain)
- Europa (Paris)
- Europa (Stockholm)
- Naher Osten (VAE)
- Naher Osten (Bahrain)
- Südamerika (São Paulo)
- Israel (Tel Aviv)
- AWS GovCloud (USA-West)
- AWS GovCloud (USA-Ost)

Erstellen eines VPC-Endpunkts für EventBridge

Zur Verwendung von EventBridge mit Ihrer VPC erstellen Sie einen Schnittstellen-VPC-Endpunkt für EventBridge und wählen com.amazonaws.***Region***.events als Servicename aus. Weitere Informationen finden Sie unter Erstellen eines Schnittstellenendpunkts im Amazon VPC Leitfaden.

Einzelheiten zu EventBridge Pipes

Die vollständige EventBridge-Pipes-Unterstützung für Schnittstellen-VPC-Endpunkte ist nicht verfügbar. Informationen zur Verwendung der folgenden Quellen innerhalb einer VPC mit EventBridge Pipes finden Sie hier:

- Amazon-MSK-Netzwerkkonfiguration
- Selbstverwaltete Apache-Kafka-Netzwerkkonfiguration
- Amazon-MQ-Netzwerkkonfiguration

Amazon-EventBridge-Integration mit AWS X-Ray

Sie können AWS X-Ray verwenden, um Ereignisse zu verfolgen, die über EventBridge geleitet werden. EventBridge übergibt den ursprünglichen Ablaufverfolgungs-Header an das Ziel, damit die Zielservices verfolgen, analysieren und debuggen können.

EventBridge kann nur dann einen Ablaufverfolgungs-Header für ein Ereignis übergeben, wenn das Ereignis aus einer `PutEvents`-Anfrage stammt, die den Ablaufverfolgungskontext übergeben hat. X-Ray verfolgt keine Ereignisse, die von Drittanbietern, geplanten Ereignissen oder AWS-Services stammen, und diese Ereignisquellen erscheinen nicht auf Ihrer X-Ray-Servicekarte.

X-Ray validiert Ablaufverfolgungs-Header, und Ablaufverfolgungs-Header, die nicht gültig sind, werden gelöscht. Das Ereignis wird jedoch weiterhin verarbeitet.

> **Important**
>
> Der Ablaufverfolgungs-Header ist für das Ereignis, das an das Aufrufziel übermittelt wird, nicht verfügbar.
>
> - Wenn Sie über ein Ereignisarchiv verfügen, ist der Ablaufverfolgungs-Header für archivierte Ereignisse nicht verfügbar. Wenn Sie archivierte Ereignisse wiederholen, ist der Ablaufverfolgungs-Header nicht enthalten.
> - Wenn Sie eine Warteschlange für unzustellbare Nachrichten haben, ist der Ablaufverfolgungs-Header in der `SendMessage`-Anfrage enthalten, die das Ereignis an die Warteschlange für unzustellbare Nachrichten sendet. Wenn Sie mithilfe von `ReceiveMessage` Ereignisse (Nachrichten) aus der Warteschlange für unzustellbare Nachrichten abrufen, ist der mit dem Ereignis verknüpfte Ablaufverfolgungs-Header im Amazon-SQS-Nachrichtenattribut enthalten, jedoch nicht in der Ereignisnachricht.

Informationen darüber, wie ein EventBridge-Ereignisknoten Quell- und Zielservices verbindet, finden Sie unter Anzeigen von Quell- und Zielservices in der X-Ray-Servicekarte im AWS X-Ray-Entwicklerhandbuch.

Sie können die folgenden Ablaufverfolgungs-Header-Informationen über EventBridge übergeben:

- Standard-HTTP-Header – Das X-Ray-SDK füllt den Ablaufverfolgungs-Header automatisch als `X-Amzn-Trace-Id`-HTTP-Header für alle Aufrufziele auf. Weitere Informationen zum Standard-HTTP-Header finden Sie unter Ablaufverfolgungs-Header im AWS X-Ray-Entwicklerhandbuch.

- **`TraceHeader`**-Systemattribut – `TraceHeader` ist ein PutEventsRequestEntry-Attribut, das von EventBridge reserviert wird, um den X-Ray-Ablaufverfolgungs-Header an ein Ziel zu übertragen. Wenn Sie auch `PutEventsRequestEntry` verwenden, überschreibt `PutEventsRequestEntry` den HTTP-Ablaufverfolgungs-Header.

> **Note**
>
> Der Ablaufverfolgungs-Header wird nicht auf die `PutEventsRequestEntry`-Ereignisgröße angerechnet. Weitere Informationen finden Sie unter Berechnung der Größe des EventBridge PutEvents Amazon-Eventeintrags.

Das folgende Video zeigt die gemeinsame Verwendung von X-Ray und EventBridge: Verwenden von AWS X-Ray für die Ablaufverfolgung

Verwendung EventBridge mit dem AWS Integrated Application Test Kit

Wenn Sie Anwendungen erstellen, EventBridge die aus serverlosen Diensten wie Lambda oder Step Functions bestehen, können viele Ihrer Architekturkomponenten nicht auf Ihrem Desktop bereitgestellt werden, sondern existieren nur in der AWS Cloud. Im Gegensatz zur Arbeit mit lokal bereitgestellten Anwendungen profitieren diese Arten von Anwendungen von cloudbasierten Strategien zur Durchführung automatisierter Tests. AWS Das Integrated Application Test Kit (AWS IATK) unterstützt Sie bei der Implementierung einiger dieser Strategien für Ihre Anwendungen.

AWS IATK ist eine Softwarebibliothek, mit der Sie automatisierte Tests für cloudbasierte Anwendungen schreiben können.

EventBridge Integration mit IATK AWS

Sie können EventBridge Ereignisse und Event-Busse mit AWS IATK verwenden, um Ihre automatisierten Tests zu implementieren, darunter:

Implementieren von Testumgebungen

Um Integrationstests für ereignisgesteuerte Architekturen zu schreiben, setzen Sie logische Grenzen, indem Sie Ihre Anwendung in Subsysteme aufteilen. Eine nützliche Methode zum Testen von Subsystemen ist die Erstellung von Testumgebungen, d. h. Ressourcen, die Sie speziell für das Testen von Subsystemen erstellen.

Ein Integrationstest kann beispielsweise einen Subsystemprozess starten, indem ein Eingabetestereignis an dieses übergeben wird. AWS IATK kann für Sie ein Testsystem erstellen, das auf Ausgabeereignisse wartet. EventBridge (Unter der Haube besteht der Kabelbaum aus einer EventBridge Regel, die das Ausgabeereignis an Amazon SQS weiterleitet.) Ihr Integrationstest fragt dann die Testumgebung ab, um die Ausgabe zu untersuchen und festzustellen, ob der Test erfolgreich ist oder nicht.

Generieren von Pseudoereignissen

AWS IATK bietet Ihnen die Möglichkeit, Scheinereignisse anhand eines in der Schemaregistrierung gespeicherten Schemas zu generieren. EventBridge Auf diese Weise können Sie ein Pseudoereignis generieren und einen beliebigen Verbraucher (z. B. eine Lambda-Funktion oder eine Step-Functions-Zustandsmaschine) mit dem generierten Ereignis aufrufen.

Weitere Informationen finden Sie unter Überblick über das AWS Integrated Application Test Kit unter. GitHub

EventBridge Amazon-Ressourcen in AWS CloudFormation Stapeln einbeziehen

AWS CloudFormation ermöglicht es Ihnen, Ihre AWS Ressourcen konto- und regionsübergreifend auf zentralisierte und wiederholbare Weise zu konfigurieren und zu verwalten, indem die Infrastruktur als Code behandelt wird. CloudFormation ermöglicht dies, indem Sie Vorlagen erstellen können, die die Ressourcen definieren, die Sie bereitstellen und verwalten möchten. Zu diesen Ressourcen können unter anderem EventBridge Artefakte wie Event-Busse und Regeln, Pipes, Schemas und Zeitpläne gehören. Verwenden Sie diese Ressourcen, um EventBridge Funktionen in die Technologie-Stacks aufzunehmen, die Sie bereitstellen und verwalten. CloudFormation

EventBridge Amazon-Ressourcen verfügbar in AWS CloudFormation

EventBridge stellt Ressourcen zur Verwendung in CloudFormation Vorlagen in den folgenden Ressourcen-Namespaces bereit:

- AWS::Events

  Einige Vorlagenbeispiele:

  - Erstellen Sie ein API-Ziel für PagerDuty
  - Erstellen eines API-Ziels für Slack
  - Erstellen Sie eine Verbindung mit ApiKey Autorisierungsparametern
  - Erstellen einer Verbindung mit OAuth-Autorisierungsparametern
  - Erstellen eines globalen Endpunkts mit Ereignisreplikation
  - Deny-Richtlinie mit mehreren Prinzipalen und Aktionen
  - Erteilen der Berechtigung an eine Organisation unter Verwendung eines benutzerdefinierten Event Bus
  - Erstellen einer regionsübergreifenden Regel
  - Erstellen einer Regel, die eine Warteschlange für unzustellbare Nachrichten für ein Ziel enthält
  - Regelmäßiger Aufruf einer Lambda-Funktion
  - Aufruf einer Lambda-Funktion als Reaktion auf ein Ereignis
  - Benachrichtigen eines Themas als Reaktion auf einen Protokolleintrag
- AWS::EventSchemas
- AWS::Pipes

  Einige Vorlagenbeispiele:

  - Erstellen einer Pipe mit einem Ereignisfilter
- AWS::Scheduler

Generieren von EventBridge Amazon-Ressourcendefinitionen für AWS CloudFormation Vorlagen

Um Ihnen den Einstieg in die Entwicklung von CloudFormation Vorlagen zu erleichtern, können Sie mit der EventBridge Konsole CloudFormation Vorlagen aus den vorhandenen Event-Bussen, Regeln und Pipes in Ihrem Konto erstellen.

- ???
- ???
- ???

Den Standard-Event-Bus unter AWS CloudFormation Kontrolle bringen

Da EventBridge der Standard-Event-Bus automatisch für Ihr Konto bereitgestellt wird, können Sie ihn nicht mithilfe einer CloudFormation Vorlage erstellen, wie Sie es normalerweise für jede Ressource tun würden, die Sie in einen CloudFormation Stack aufnehmen möchten. Um den Standard-Event-Bus in einen CloudFormation Stack aufzunehmen, müssen Sie ihn zuerst in einen Stack importieren. Nachdem Sie den Standard-Event-Bus in einen Stack importiert haben, können Sie die Eigenschaften des Event-Busses nach Bedarf aktualisieren.

Weitere Informationen finden Sie unter ???.

Verwaltung von AWS CloudFormation Stack-Ereignissen mit EventBridge

Sie können nicht nur EventBridge Ressourcen in Ihre CloudFormation Stacks aufnehmen, sondern auch die von den CloudFormation Stacks selbst generierten Ereignisse verwalten. EventBridge CloudFormation sendet Ereignisse an EventBridge jedes Mal, wenn ein Vorgang zum Erstellen, Aktualisieren, Löschen oder zur Erkennung von Abweichungen an einem Stack ausgeführt wird. CloudFormation sendet auch Ereignisse an, EventBridge um Statusänderungen an Stack-Sets und Stack-Set-Instanzen vorzunehmen. Sie können EventBridge Regeln verwenden, um Ereignisse an Ihre definierten Ziele weiterzuleiten.

Weitere Informationen finden Sie unter CloudFormation Ereignisse mithilfe von verwalten EventBridge im AWS CloudFormation Benutzerhandbuch.

Amazon-EventBridge-Tutorials

EventBridge lässt sich in eine Reihe von AWS-Services und SaaS-Partnern integrieren. Diese Tutorials sollen Ihnen helfen, sich mit den Grundlagen von EventBridge vertraut zu machen und zu erfahren, wie EventBridge Teil Ihrer Serverless-Architektur sein kann.

Tutorials:

- Erste-Schritte-Tutorials für Amazon EventBridge
- Amazon-EventBridge-Tutorials zur Integration mit anderen AWS-Services
- Amazon-EventBridge-Tutorials zur Integration mit SaaS-Anbietern

Erste-Schritte-Tutorials für Amazon EventBridge

Die folgenden Tutorials helfen Ihnen, die Funktionen von EventBridge kennenzulernen und wie man sie benutzt.

Tutorials:

- Archivieren und Wiederholen von Amazon-EventBridge-Ereignissen
- Erstellen einer Amazon-EventBridge-Beispielanwendung
- Tutorial: Herunterladen von Codebindungen für Ereignisse mithilfe der EventBridge-Schemaregistrierung
- Tutorial: Verwenden des Eingabe-Transformators, um die EventBridge-Ausgabe an das Ereignisziel anzupassen

Archivieren und Wiederholen von Amazon-EventBridge-Ereignissen

Sie können EventBridge verwenden, um Ereignisse mithilfe von Regeln an bestimmte AWS Lambda-Funktionen weiterzuleiten.

In diesem Tutorial erstellen Sie mithilfe der Lambda-Konsole eine Funktion, die als Ziel für die EventBridge-Regel verwendet werden soll. Anschließend erstellen Sie ein Archiv und eine Regel, mit der Testereignisse mithilfe der EventBridge-Konsole archiviert werden. Sobald Ereignisse in diesem Archiv vorhanden sind, wiederholen Sie sie.

Schritte:

- Schritt 1: Erstellen einer Lambda-Funktion
- Schritt 2: Erstellen eines Archivs
- Schritt 3: Erstellen einer Regel
- Schritt 4: Senden von Testereignissen
- Schritt 5: Wiederholen von Ereignissen
- Schritt 6: Bereinigen Ihrer Ressourcen

Schritt 1: Erstellen einer Lambda-Funktion

Erstellen Sie zunächst eine Lambda-Funktion, um die Ereignisse zu protokollieren.

So erstellen Sie eine Lambda-Funktion:

1. Öffnen Sie die AWS Lambda-Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie Create function (Funktion erstellen).
3. Wählen Sie Author from scratch aus.
4. Geben Sie einen Namen und eine Beschreibung für die Lambda-Funktion ein. Geben Sie der Funktion beispielsweise den Namen `LogScheduledEvent`.
5. Behalten Sie die übrigen Optionen als Standardwerte bei und wählen Sie Funktion erstellen aus.
6. Doppelklicken Sie auf der Registerkarte Code der Funktionsseite auf index.js.
7. Ersetzen Sie den vorhandenen JavaScript-Code mit folgendem Code:

```
'use strict';

exports.handler = (event, context, callback) => {
```

```
    console.log('LogScheduledEvent');
    console.log('Received event:', JSON.stringify(event, null, 2));
    callback(null, 'Finished');
};
```

8. Wählen Sie Bereitstellen aus.

Schritt 2: Erstellen eines Archivs

Erstellen Sie als Nächstes das Archiv, in dem alle Testereignisse gespeichert werden.

So erstellen Sie ein Archiv

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Archive aus.
3. Wählen Sie Archiv erstellen.
4. Geben Sie einen Namen und eine Beschreibung für das Archiv ein. Nennen Sie das Archiv beispielsweise `ArchiveTest`.
5. Behalten Sie die übrigen Optionen als Standardwerte bei und wählen Sie Weiter aus.
6. Wählen Sie Archiv erstellen.

Schritt 3: Erstellen einer Regel

Erstellen Sie eine Regel zum Archivieren von Ereignissen, die an den Event Bus gesendet werden.

So erstellen Sie eine Regel

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Nennen Sie die Regel beispielsweise `ARTestRule`.

   Eine Regel darf nicht denselben Namen wie eine andere Regel in derselben Region und auf demselben Event Bus haben.
5. Wählen Sie für Event Bus den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen

Sie Standard aus. Wenn ein AWS-Service in Ihrem Konto ein Ereignis ausgibt, wird es stets an den Standard-Event-Bus Ihres Kontos weitergeleitet.

6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Next (Weiter).
8. Wählen Sie für Event source (Ereignisquelle) Other (Andere) aus.
9. Geben Sie für Ereignismuster Folgendes ein:

```
{
  "detail-type": [
    "customerCreated"
  ]
}
```

10. Wählen Sie Next (Weiter).
11. Bei Target types (Zieltypen) wählen Sie AWS-Service aus.
12. Wählen Sie für Ziel auswählen die Option Lambda-Funktion aus der Dropdown-Liste aus.
13. Wählen Sie für Funktion die Lambda-Funktion aus, die Sie im Abschnitt Schritt 1: Erstellen einer Lambda-Funktion erstellt haben. Wählen Sie in diesem Beispiel `LogScheduledEvent` aus.
14. Wählen Sie Next (Weiter).
15. Wählen Sie Next (Weiter).
16. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 4: Senden von Testereignissen

Nachdem Sie das Archiv und die Regel eingerichtet haben, senden wir Testereignisse, um sicherzustellen, dass das Archiv ordnungsgemäß funktioniert.

> **Note**
>
> Es kann einige Zeit dauern, bis Ereignisse in das Archiv aufgenommen werden.

So senden Sie Testereignisse (Konsole)

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.

2. Wählen Sie im Navigationsbereich die Option Event Buses (Ereignisbusse) aus.
3. Wählen Sie in der Kachel Standard-Event-Bus die Optionen Aktionen, Ereignisse senden aus.
4. Geben Sie eine Ereignisquelle ein. Zum Beispiel `TestEvent`.
5. Geben Sie für Detailtyp `customerCreated` ein.
6. Geben Sie für Ereignisdetail `{}` ein.
7. Wählen Sie Send (Senden) aus.

Schritt 5: Wiederholen von Ereignissen

Sobald sich die Testereignisse im Archiv befinden, können Sie sie wiederholen.

So wiederholen Sie archivierte Ereignisse (Konsole)

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Wiederholungen.
3. Wählen Sie Neue Wiederholung starten.
4. Geben Sie einen Namen und eine Beschreibung für die Wiederholung ein. Nennen Sie die Wiederholung beispielsweise `ReplayTest`.
5. Wählen Sie für Quelle das Archiv aus, das Sie im Abschnitt Schritt 2: Erstellen eines Archivs erstellt haben.
6. Gehen Sie für Zeitrahmen der Wiederholung wie folgt vor.
   a. Wählen Sie für Startzeit das Datum aus, an dem Sie die Testereignisse gesendet haben, und eine Uhrzeit, bevor Sie sie gesendet haben. Beispiel: `2021/08/11` und `08:00:00`.
   b. Wählen Sie für Endzeit das aktuelle Datum und die aktuelle Uhrzeit aus. Beispiel: `2021/08/11` und `09:15:00`.
7. Wählen Sie Wiederholung starten.

Schritt 6: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Durch das Löschen von AWS-Ressourcen, die Sie nicht mehr verwenden, können Sie verhindern, dass unnötige Gebühren in Ihrem AWS-Konto anfallen.

So löschen Sie die Lambda-Funktion(en)

1. Öffnen Sie die Seite Funktionen der Lambda-Konsole.
2. Wählen Sie die Funktion(en) aus, die Sie erstellt haben.
3. Wählen Sie Aktionen, Löschen aus.
4. Wählen Sie Delete (Löschen).

So löschen Sie das/die EventBridge-Archiv(e)

1. Öffnen Sie die Seite Archive der EventBridge-Konsole.
2. Wählen Sie das/die Archiv(e) aus, das/die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Geben Sie den Archivnamen ein und wählen Sie Löschen aus.

So löschen Sie die EventBridge-Regel(n)

1. Öffnen Sie die Seite Regeln der EventBridge-Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

Erstellen einer Amazon-EventBridge-Beispielanwendung

Sie können EventBridge verwenden, um Ereignisse mithilfe von Regeln an bestimmte Lambda-Funktionen weiterzuleiten.

In diesem Tutorial verwenden Sie die AWS CLI, Node.js und den Code im GitHub-Repo, um Folgendes zu erstellen:

- Eine AWS Lambda-Funktion, die Ereignisse für ATM-Banktransaktionen produziert.
- Drei Lambda-Funktionen zur Verwendung als Ziele einer EventBridge-Regel.
- Und die Regel, die die erstellten Ereignisse auf der Grundlage eines Ereignismusters an die richtige nachgelagerte Funktion weiterleitet.

In diesem Beispiel werden AWS SAM-Vorlagen verwendet, um die EventBridge-Regeln zu definieren. Weitere Informationen zur Verwendung von AWS SAM-Vorlagen mit EventBridge finden Sie unter ???.

Im Repo enthält das Unterverzeichnis atmProducer das `handler.js`, das den ATM-Service darstellt, der Ereignisse produziert. Dieser Code ist ein in Node.js geschriebener Lambda-Handler, der Ereignisse mithilfe dieser JavaScript-Codezeile über das AWS SDK in EventBridge veröffentlicht.

```
const result = await eventbridge.putEvents(params).promise()
```

Dieses Verzeichnis enthält auch ein `events.js`, das mehrere Testtransaktionen in einem Antwort-Array auflistet. Ein einzelnes Ereignis ist in JavaScript wie folgt definiert:

```
{
  // Event envelope fields
  Source: 'custom.myATMapp',
  EventBusName: 'default',
  DetailType: 'transaction',
  Time: new Date(),

  // Main event body
  Detail: JSON.stringify({
    action: 'withdrawal',
    location: 'MA-BOS-01',
    amount: 300,
    result: 'approved',
    transactionId: '123456',
```

```
    cardPresent: true,
    partnerBank: 'Example Bank',
    remainingFunds: 722.34
  })
}
```

Der Abschnitt Details des Ereignisses spezifiziert Transaktionsattribute. Dazu gehören der Standort des ATM, der Betrag, die Partnerbank und das Ergebnis der Transaktion.

Die `handler.js`-Datei im Unterverzeichnis atmConsumer enthält drei Funktionen:

```
exports.case1Handler = async (event) => {
  console.log('--- Approved transactions ---')
  console.log(JSON.stringify(event, null, 2))
}

exports.case2Handler = async (event) => {
  console.log('--- NY location transactions ---')
  console.log(JSON.stringify(event, null, 2))
}

exports.case3Handler = async (event) => {
  console.log('--- Unapproved transactions ---')
  console.log(JSON.stringify(event, null, 2))
}
```

Jede Funktion empfängt Transaktionsereignisse, die über die `console.log`-Anweisungen in Amazon CloudWatch Logs protokolliert werden. Die Konsumentenfunktionen arbeiten unabhängig vom Produzenten und sind sich der Quelle der Ereignisse nicht bewusst.

Die Weiterleitungslogik ist in den EventBridge-Regeln enthalten, die von der AWS SAM-Vorlage der Anwendung bereitgestellt werden. Die Regeln werten den eingehenden Stream von Ereignissen aus und leiten übereinstimmende Ereignisse an die Lambda-Zielfunktionen weiter.

Die Regeln verwenden Ereignismuster, bei denen es sich um JSON-Objekte mit derselben Struktur wie die Ereignisse handelt, mit denen sie übereinstimmen. Hier ist das Ereignismuster für eine der Regeln.

```
{
  "detail-type": ["transaction"],
  "source": ["custom.myATMapp"],
  "detail": {
```

```
    "location": [{
      "prefix": "NY-"
    }]
  }
}
```

Schritte:

- Voraussetzungen
- Schritt 1: Erstellen einer Anwendung
- Schritt 2: Ausführen einer Anwendung
- Schritt 3: Überprüfen der Protokolle und Funktionsfähigkeit der Anwendung
- Schritt 4: Bereinigen Ihrer Ressourcen

Voraussetzungen

Zum Durcharbeiten dieses Tutorials benötigen Sie die folgenden Ressourcen:

- Ein AWS-Konto. Erstellen Sie ein AWS-Konto, wenn noch keines vorhanden ist.
- Installierte AWS CLI Informationen zur Installation der AWS CLI finden Sie unter Installieren, Aktualisieren und Deinstallieren der AWS CLI Version 2.
- Installiertes Node.js 12.x Informationen zur Installation von Node.js finden Sie unter Downloads.

Schritt 1: Erstellen einer Anwendung

Um die Beispielanwendung einzurichten, verwenden Sie die AWS CLI und Git, um die benötigten AWS-Ressourcen zu erstellen.

So erstellen Sie die Anwendung

1. Melden Sie sich bei AWS an.
2. Installieren Sie Git und die AWS Serverless Application Model-CLI auf Ihrem lokalen Computer.
3. Erstellen Sie ein neues Verzeichnis und navigieren Sie dann in einem Terminal zu diesem Verzeichnis.
4. Geben Sie `git clone https://github.com/aws-samples/amazon-eventbridge-producer-consumer-example` in die Befehlszeile ein.
5. Führen Sie in der Befehlszeile den folgenden Befehl aus:

```
cd ./amazon-eventbridge-producer-consumer-example
sam deploy --guided
```

6. Führen Sie die folgenden Schritte im Terminal aus:

   a. Geben Sie für **`Stack Name`** einen Namen für den Stack ein. Nennen Sie den Stack beispielsweise `Test`.

   b. Geben Sie für **`AWS Region`** die Region ein. Zum Beispiel `us-west-2`.

   c. Geben Sie unter **`Confirm changes before deploy`** den Wert Y ein.

   d. Geben Sie für **`Allow SAM CLI IAM role creation`** den Wert Y ein.

   e. Geben Sie für **`Save arguments to configuration file`** den Wert Y ein.

   f. Geben Sie unter **`SAM configuration file`** den Wert `samconfig.toml` ein.

   g. Geben Sie unter **`SAM configuration environment`** den Wert `default` ein.

Schritt 2: Ausführen einer Anwendung

Nachdem die Ressourcen eingerichtet sind, verwenden Sie die Konsole, um die Funktionen zu testen.

So führen Sie die Anwendung aus

1. Öffnen Sie die Lambda-Konsole in derselben Region, in der Sie die AWS SAM-Anwendung bereitgestellt haben.
2. Es gibt vier Lambda-Funktionen mit dem Präfix atm-demo. Wählen Sie die Funktion atmProducerFn und dann Aktionen, Test aus.
3. Geben Sie `Test` für den Namen ein.
4. Wählen Sie Test aus.

Schritt 3: Überprüfen der Protokolle und Funktionsfähigkeit der Anwendung

Nachdem Sie die Anwendung ausgeführt haben, verwenden Sie die Konsole, um die CloudWatch-Protokolle zu überprüfen.

So überprüfen Sie die Protokolle

1. Öffnen Sie die CloudWatch-Konsole in derselben Region, in der Sie die AWS SAM-Anwendung ausgeführt haben.
2. Wählen Sie Logs (Protokolle) und anschließend Log groups (Protokollgruppen) aus.
3. Wählen Sie die Protokollgruppe aus, die atmConsumerCase1 enthält. Sie sehen zwei Streams, die die beiden vom ATM genehmigten Transaktionen darstellen. Wählen Sie einen Protokollstream aus, um die Ausgabe anzuzeigen.
4. Navigieren Sie zurück zur Liste der Protokollgruppen und wählen Sie dann die Protokollgruppe aus, die atmConsumerCase2 enthält. Es werden zwei Streams angezeigt, die die beiden Transaktionen darstellen, die dem Standortfilter New York entsprechen.
5. Navigieren Sie zurück zur Liste der Protokollgruppen und wählen Sie die Protokollgruppe aus, die atmConsumerCase3 enthält. Öffnen Sie den Stream, um die abgelehnten Transaktionen zu sehen.

Schritt 4: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Durch das Löschen von AWS-Ressourcen, die Sie nicht mehr verwenden, können Sie verhindern, dass unnötige Gebühren in Ihrem AWS-Konto anfallen.

So löschen Sie die EventBridge-Regel(n)

1. Öffnen Sie die Seite Regeln der EventBridge-Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

So löschen Sie die Lambda-Funktion(en)

1. Öffnen Sie die Seite Funktionen der Lambda-Konsole.
2. Wählen Sie die Funktion(en) aus, die Sie erstellt haben.
3. Wählen Sie Aktionen, Löschen aus.
4. Wählen Sie Delete (Löschen).

So löschen Sie die CloudWatch-Logs-Protokollgruppe(n)

1. Öffnen Sie die CloudWatch-Konsole.
2. Wählen Sie Protokolle, Protokollgruppen aus.
3. Wählen Sie die Protokollgruppe(n) aus, die in diesem Tutorial erstellt wurde(n).
4. Wählen Sie Actions (Aktionen), Delete log group(s) (Protokollgruppe(n) löschen) aus.
5. Wählen Sie Delete (Löschen).

Tutorial: Herunterladen von Codebindungen für Ereignisse mithilfe der EventBridge-Schemaregistrierung

Sie können Codebindungen für Ereignisschemata generieren, um die Entwicklung für Golang, Java, Python und TypeScript zu beschleunigen. Sie können Codebindungen für vorhandene AWS-Services, von Ihnen erstellte Schemata und für Schemata abrufen, die Sie basierend auf Ereignissen in einem Event Bus generieren. Sie können Codebindungen für ein Schema mit einer der folgenden Komponenten generieren:

- EventBridge-Konsole
- EventBridge-Schemaregistrierungs-API
- Ihre IDE mit einem AWS Toolkit

In diesem Tutorial generieren Sie Codebindungen aus einem EventBridge-Schema für die Ereignisse eines AWS-Service und laden sie herunter.

So generieren Sie Codebindungen aus einem EventBridge-Schema

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Schemas aus.
3. Wählen Sie die Registerkarte AWS-Ereignisschemaregistrierung aus.
4. Suchen Sie das Schema für den AWS-Service, für das Sie Codebindungen verwenden möchten. Durchsuchen Sie dazu entweder die Schemaregistrierung oder suchen Sie nach einem Schema.
5. Wählen Sie den Schemanamen.
6. Wählen Sie auf der Seite Schemadetails im Abschnitt Version die Option Codebindungen herunterladen aus.
7. Wählen Sie auf der Seite Download code bindings (Codebindungen herunterladen) die Sprache der Codebindungen aus, die Sie herunterladen möchten.
8. Wählen Sie Download (Herunterladen) aus.

   Es kann einige Sekunden dauern, bis der Download beginnt. Bei der Download-Datei handelt es sich um eine ZIP-Datei mit Codebindungen für die ausgewählte Sprache.
9. Entpacken Sie die heruntergeladene Datei und fügen Sie sie Ihrem Projekt hinzu.

   Das heruntergeladene Paket enthält eine README-Datei, in der erklärt wird, wie die Abhängigkeiten des Pakets in verschiedenen Frameworks konfiguriert werden.

Verwenden Sie diese Codebindungen in Ihrem eigenen Code, um schnell Anwendungen mit diesem EventBridge-Ereignis zu erstellen.

Tutorial: Verwenden des Eingabe-Transformators, um die EventBridge-Ausgabe an das Ereignisziel anzupassen

Sie können mit dem Eingabe-Transformator in EventBridge Text aus einem Ereignis anpassen, bevor Sie ihn an das Ziel einer Regel senden.

Dazu definieren Sie JSON-Pfade aus dem Ereignis und weisen ihre Ausgaben unterschiedlichen Variablen zu. Anschließend können Sie diese Variablen in der Eingabevorlage verwenden. Die Zeichen < und > können nicht durch Escape-Zeichen geschützt werden. Weitere Informationen finden Sie unter Transformation Amazon EventBridge Amazon-Eingaben.

> **Note**
>
> Wenn Sie eine Variable angeben, die einem JSON-Pfad entspricht, der im Ereignis nicht vorhanden ist, wird diese Variable nicht erstellt und nicht in der Ausgabe angezeigt.

In diesem Tutorial erstellen Sie eine Regel, die einem Ereignis mit `detail-type: "customerCreated"` entspricht. Der Eingabe-Tansformator ordnet die Variable `type` dem JSON-Pfad $.detail-type aus dem Ereignis zu. Anschließend fügt EventBridge die Variable in die Eingabevorlage "This event was <type>" ein. Das Ergebnis ist die folgende Amazon-SNS-Nachricht.

```
"This event was of customerCreated type."
```

Schritte:

- Schritt 1: Erstellen eines Amazon-SNS-Themas
- Schritt 2: Erstellen eines Amazon-SNS-Abonnements
- Schritt 3: Erstellen einer Regel
- Schritt 4: Senden von Testereignissen
- Schritt 5: Bestätigen des Erfolgs
- Schritt 6: Bereinigen Ihrer Ressourcen

Schritt 1: Erstellen eines Amazon-SNS-Themas

Erstellen Sie ein Thema, um die Ereignisse von EventBridge zu erhalten.

Erstellen Sie ein Thema wie folgt

1. Öffnen Sie die Amazon SNS-Konsole unter https://console.aws.amazon.com/sns/v3/home.
2. Wählen Sie im Navigationsbereich Topics (Themen) aus.
3. Wählen Sie Create topic (Thema erstellen) aus.
4. Wählen Sie unter Type (Typ) die Option Standard aus.
5. Geben Sie **`eventbridge-IT-test`** als Namen des Themas ein.
6. Wählen Sie Create topic (Thema erstellen) aus.

Schritt 2: Erstellen eines Amazon-SNS-Abonnements

Erstellen Sie ein Abonnement, um E-Mails mit den transformierten Informationen zu erhalten.

Erstellen eines Abonnements

1. Öffnen Sie die Amazon SNS-Konsole unter https://console.aws.amazon.com/sns/v3/home.
2. Wählen Sie im Navigationsbereich Subscriptions aus.
3. Wählen Sie Create subscription.
4. Wählen Sie für Thema-ARN das in Schritt 1 erstellte Thema aus. Wählen Sie für dieses Tutorial eventbridge-IT-test aus.
5. Wählen Sie unter Protocol (Protokoll) die Option Email (E-Mail) aus.
6. Geben Sie unter Endpunkt Ihre E-Mail-Adresse ein.
7. Klicken Sie auf Create subscription (Abonnement erstellen).
8. Bestätigen Sie das Abonnement, indem Sie in der E-Mail, die Sie von AWS-Benachrichtigungen erhalten, die Option Abonnement bestätigen auswählen.

Schritt 3: Erstellen einer Regel

Erstellen Sie eine Regel, um mithilfe des Eingabe-Transformators die Informationen zum Instance-Status anzupassen, die an ein Ziel gesendet werden.

So erstellen Sie eine Regel

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.

3. Wählen Sie Regel erstellen.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Nennen Sie die Regel beispielsweise `ARTestRule`.
5. Wählen Sie für Event Bus den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie Standard aus. Wenn ein AWS-Service in Ihrem Konto ein Ereignis ausgibt, wird es stets an den Standard-Event-Bus Ihres Kontos weitergeleitet.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Next (Weiter).
8. Wählen Sie für Event source (Ereignisquelle) Other (Andere) aus.
9. Geben Sie für Ereignismuster Folgendes ein:

```
{
  "detail-type": [
    "customerCreated"
  ]
}
```

10. Wählen Sie Next (Weiter).
11. Bei Target types (Zieltypen) wählen Sie AWS-Service aus.
12. Wählen Sie für Ziel auswählen die Option SNS-Thema aus der Dropdown-Liste aus.
13. Wählen Sie für Thema das Amazon-SNS-Thema aus, das Sie in Schritt 1 erstellt haben. Wählen Sie für dieses Tutorial eventbridge-IT-test aus.
14. Gehen Sie für Weitere Einstellungen wie folgt vor:

    a. Wählen Sie für Zieleingabe konfigurieren die Option Eingabe-Transformator aus der Dropdown-Liste aus.

    b. Wählen Sie Eingabe-Transformator konfigurieren aus.

    c. Geben Sie für Beispielereignisse Folgendes ein:

```
{
  "detail-type": "customerCreated"
}
```

    d. Gehen Sie für Zieleingabe-Transformator wie folgt vor:

i. Geben Sie für Eingabepfad Folgendes ein:

```
{"detail-type":"$.detail-type"}
```

ii. Geben Sie für Eingabevorlage Folgendes ein:

```
"This event was of <detail-type> type."
```

e. Wählen Sie Bestätigen aus.

15. Wählen Sie Next (Weiter).
16. Wählen Sie Next (Weiter).
17. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 4: Senden von Testereignissen

Nachdem Sie das SNS-Thema und die Regel eingerichtet haben, senden wir Testereignisse, um sicherzustellen, dass die Regel ordnungsgemäß funktioniert.

So senden Sie Testereignisse (Konsole)

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich die Option Event Buses (Ereignisbusse) aus.
3. Wählen Sie in der Kachel Standard-Event-Bus die Optionen Aktionen, Ereignisse senden aus.
4. Geben Sie eine Ereignisquelle ein. Zum Beispiel `TestEvent`.
5. Geben Sie für Detailtyp `customerCreated` ein.
6. Geben Sie für Ereignisdetail `{}` ein.
7. Wählen Sie Send (Senden) aus.

Schritt 5: Bestätigen des Erfolgs

Wenn Sie von AWS-Benachrichtigungen eine E-Mail erhalten, die der erwarteten Ausgabe entspricht, haben Sie das Tutorial erfolgreich abgeschlossen.

Schritt 6: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Durch das Löschen von AWS-Ressourcen, die Sie nicht mehr verwenden, können Sie verhindern, dass unnötige Gebühren in Ihrem AWS-Konto anfallen.

So löschen Sie das SNS-Thema

1. Öffnen Sie die Seite Themen der SNS-Konsole.
2. Wählen Sie das Thema aus, das Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Geben Sie **`delete me`** ein.
5. Wählen Sie Delete (Löschen).

So löschen Sie das SNS-Abonnement

1. Öffnen Sie die Seite Abonnements der SNS-Konsole.
2. Wählen Sie das von Ihnen erstellte Abonnement aus.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

So löschen Sie die EventBridge-Regel(n)

1. Öffnen Sie die Seite Regeln der EventBridge-Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

Amazon-EventBridge-Tutorials zur Integration mit anderen AWS-Services

Amazon EventBridge funktioniert mit anderen AWS-Services, um Ereignisse zu verarbeiten oder eine AWS-Ressource als Ziel einer Regel aufzurufen. Die folgenden Tutorials zeigen, wie Sie EventBridge mit anderen AWS-Services integrieren.

Tutorials:

- Tutorial: Protokollieren des Status einer Auto-Scaling-Gruppe mit EventBridge
- Tutorial: AWS API-Aufrufe protokollieren mit EventBridge
- Tutorial: Den Status einer Amazon EC2 EC2-Instance protokollieren mit EventBridge
- Tutorial: Protokollieren von Amazon-S3-Operationen auf Objektebene mit EventBridge
- Tutorial: Ereignisse mithilfe EventBridge des Schemas an einen Amazon Kinesis-Stream senden aws.events
- Tutorial: Planen automatisierter Amazon-EBS-Snapshots mit EventBridge
- Tutorial: Senden einer Benachrichtigung, wenn ein Amazon-S3-Objekt erstellt wird
- Tutorial: Planen von AWS Lambda-Funktionen mit EventBridge

Tutorial: Protokollieren des Status einer Auto-Scaling-Gruppe mit EventBridge

Sie können eine AWS Lambda-Funktion ausführen, die Ereignisse protokolliert, sobald eine Auto-Scaling-Gruppe eine Amazon-EC2-Instance startet oder beendet, und angibt, ob ein Ereignis erfolgreich war.

Weitere Informationen zu weiteren Szenarien, die Ereignisse von Amazon EC2 Auto Scaling verwenden, finden Sie unter Verwenden von EventBridge, um Auto-Scaling-Ereignisse zu behandeln im Benutzerhandbuch für Amazon EC2 Auto Scaling.

In diesem Tutorial erstellen Sie eine Lambda-Funktion und in der EventBridge-Konsole eine Regel, die diese Funktion aufruft, wenn eine Gruppe von Amazon EC2 Auto Scaling eine Instance startet oder beendet.

Schritte:

- Voraussetzungen
- Schritt 1: Erstellen einer Lambda-Funktion
- Schritt 2: Erstellen einer Regel
- Schritt 3: Testen der Regel
- Schritt 4: Bestätigen des Erfolgs
- Schritt 5: Bereinigen Ihrer Ressourcen

Voraussetzungen

Zum Durcharbeiten dieses Tutorials benötigen Sie die folgenden Ressourcen:

- Eine Auto-Scaling-Gruppe Weitere Informationen zum Erstellen einer Auto-Scaling-Gruppe finden Sie unter Erstellen einer Auto-Scaling-Gruppe mithilfe einer Startkonfiguration im Benutzerhandbuch für Amazon EC2 Auto Scaling.

Schritt 1: Erstellen einer Lambda-Funktion

Erstellen Sie eine Lambda-Funktion, um die Skalierungsereignisse für die Auto Scaling-Gruppe zu protokollieren.

So erstellen Sie eine Lambda-Funktion:

1. Öffnen Sie die AWS Lambda-Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie Create function (Funktion erstellen).
3. Wählen Sie Author from scratch aus.
4. Geben Sie einen Namen für die Lambda-Funktion ein. Geben Sie der Funktion beispielsweise den Namen `LogAutoScalingEvent`.
5. Behalten Sie die übrigen Optionen als Standardwerte bei und wählen Sie Funktion erstellen aus.
6. Doppelklicken Sie auf der Registerkarte Code der Funktionsseite auf index.js.
7. Ersetzen Sie den vorhandenen Code mit folgendem Code.

```
'use strict';

exports.handler = (event, context, callback) => {
    console.log('LogAutoScalingEvent');
    console.log('Received event:', JSON.stringify(event, null, 2));
    callback(null, 'Finished');
};
```

8. Wählen Sie Deploy (Bereitstellen) aus.

Schritt 2: Erstellen einer Regel

Erstellen Sie eine Regel, um die Lambda-Funktion auszuführen, die Sie in Schritt 1 erstellt haben. Die Regel wird ausgeführt, wenn Ihre Auto-Scaling-Gruppe eine Instance startet oder stoppt.

So erstellen Sie eine Regel

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Nennen Sie die Regel beispielsweise `TestRule`.
5. Wählen Sie für Event Bus den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie Standard aus. Wenn ein AWS-Service in Ihrem Konto ein Ereignis ausgibt, wird es stets an den Standard-Event-Bus Ihres Kontos weitergeleitet.

6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Next (Weiter).
8. Als Event source (Ereignisquelle) wählen Sie AWS-Services aus.
9. Gehen Sie bei Event pattern (Ereignismuster) wie folgt vor:

    a. Wählen Sie für Ereignisquelle die Option Auto Scaling aus der Dropdown-Liste aus.
    b. Wählen Sie für Ereignistyp die Option Instance starten und beenden aus der Dropdown-Liste aus.
    c. Wählen Sie Beliebiges Instance-Ereignis und Beliebiger Gruppenname aus.
10. Wählen Sie Next (Weiter).
11. Bei Target types (Zieltypen) wählen Sie AWS-Service aus.
12. Wählen Sie für Ziel auswählen die Option Lambda-Funktion aus der Dropdown-Liste aus.
13. Wählen Sie für Funktion die Lambda-Funktion aus, die Sie im Abschnitt Schritt 1: Erstellen einer Lambda-Funktion erstellt haben. Wählen Sie in diesem Beispiel `LogAutoScalingEvent` aus.
14. Wählen Sie Next (Weiter).
15. Wählen Sie Next (Weiter).
16. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 3: Testen der Regel

Sie können die Regel durch manuelle Skalierung einer to Auto Scaling-Gruppe testen, sodass eine Instance gestartet wird. Warten Sie einige Minuten, damit das Aufskalierungsereignis eintritt. Dann können Sie überprüfen, ob die Lambda-Funktion aufgerufen wurde.

So testen Sie die Regel mit einer Auto Scaling-Gruppe

1. So erhöhen Sie die Größe der Auto Scaling-Gruppe:

    a. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
    b. Wählen Sie im Navigationsbereich Auto Scaling (Auto Scaling) und Auto Scaling Groups (Auto Scaling-Gruppen) aus.
    c. Aktivieren Sie das Kontrollkästchen für die Auto Scaling-Gruppe.
    d. Wählen Sie auf der Registerkarte Details die Option Edit (Bearbeiten) aus. Erhöhen Sie für Desired die gewünschte Kapazität um eine Einheit. Wenn der aktuelle Wert beispielsweise

2 ist, geben Sie 3 ein. Die gewünschte Kapazität darf maximal so groß wie die Höchstgröße der Gruppe sein. Wenn Ihr neuer Wert für Desired größer ist als Max, müssen Sie Max aktualisieren. Wenn Sie fertig sind, wählen Sie Speichern.

2. So zeigen Sie die Ausgabe aus der Lambda-Funktion an:

   a. Öffnen Sie die CloudWatch-Konsole unter https://console.aws.amazon.com/cloudwatch/.

   b. Wählen Sie im Navigationsbereich Logs (Logs) aus.

   c. Wählen Sie den Namen der Protokollgruppe für Ihre Lambda-Funktion aus (`/aws/lambda/function-name`).

   d. Wählen Sie den Namen des Protokoll-Streams aus, um die von der Funktion für die von Ihnen gestartete Instance bereitgestellten Daten anzuzeigen.

3. (Optional) Wenn Sie fertig sind, können Sie die gewünschte Kapazität verringern, sodass die Auto-Scaling-Gruppe auf die vorherige Größe zurückgesetzt wird.

Schritt 4: Bestätigen des Erfolgs

Wenn Sie das Lambda-Ereignis in den CloudWatch-Protokollen sehen, haben Sie dieses Tutorial erfolgreich abgeschlossen. Wenn das Ereignis nicht in Ihren CloudWatch-Protokollen enthalten ist, beginnen Sie mit der Fehlerbehebung, indem Sie überprüfen, ob die Regel erfolgreich erstellt wurde. Wenn die Regel korrekt aussieht, überprüfen Sie, ob der Code Ihrer Lambda-Funktion korrekt ist.

Schritt 5: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Durch das Löschen von AWS-Ressourcen, die Sie nicht mehr verwenden, können Sie verhindern, dass unnötige Gebühren in Ihrem AWS-Konto anfallen.

So löschen Sie die EventBridge-Regel(n)

1. Öffnen Sie die Seite Regeln der EventBridge-Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

So löschen Sie die Lambda-Funktion(en)

1. Öffnen Sie die Seite Funktionen der Lambda-Konsole.

2. Wählen Sie die Funktion(en) aus, die Sie erstellt haben.
3. Wählen Sie Aktionen, Löschen aus.
4. Wählen Sie Delete (Löschen).

Tutorial: AWS API-Aufrufe protokollieren mit EventBridge

Sie können EventBridge Amazon-Regeln verwenden, um auf API-Aufrufe eines AWS Dienstes zu reagieren, die von aufgezeichnet wurden AWS CloudTrail.

In diesem Tutorial erstellen Sie einen AWS CloudTrailTrail, eine Lambda-Funktion und eine Regel in der EventBridge Konsole. Die Regel ruft die Lambda-Funktion auf, wenn eine Amazon-EC2-Instance gestoppt wird.

Schritte:

- Schritt 1: Erstellen Sie einen Trail AWS CloudTrail
- Schritt 2: Erstellen einer AWS Lambda -Funktion
- Schritt 3: Erstellen einer Regel
- Schritt 4: Testen der Regel
- Schritt 5: Bestätigen des Erfolgs
- Schritt 6: Bereinigen Ihrer Ressourcen

Schritt 1: Erstellen Sie einen Trail AWS CloudTrail

Wenn Sie bereits einen Trail eingerichtet haben, gehen Sie weiter zu Schritt 2.

Sie erstellen einen Trail wie folgt:

1. Öffnen Sie die CloudTrail Konsole unter https://console.aws.amazon.com/cloudtrail/.
2. Wählen Sie Trails, Create trail (Trail erstellen).
3. Geben Sie unter Trail name einen Namen für den Trail ein.
4. Wählen Sie für Speicherort die Option Neuen S3-Bucket erstellen aus.
5. Geben Sie einen AWS KMS -Alias für den KMS-Schlüssel ein.
6. Wählen Sie Weiter.
7. Wählen Sie Weiter.
8. Wählen Sie Create Trail (Trail erstellen) aus.

Schritt 2: Erstellen einer AWS Lambda -Funktion

Erstellen Sie eine Lambda-Funktion, um den API-Aufruf zu protokollieren.

Eine Lambda-Funktion erstellen

1. Öffnen Sie die AWS Lambda Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie Funktion erstellen.
3. Wählen Sie Von Grund auf neu schreiben aus.
4. Geben Sie einen Namen und eine Beschreibung für die Lambda-Funktion ein. Geben Sie der Funktion beispielsweise den Namen `LogEC2StopInstance`.
5. Behalten Sie die übrigen Optionen als Standardwerte bei und wählen Sie Funktion erstellen aus.
6. Doppelklicken Sie auf der Registerkarte Code der Funktionsseite auf index.js.
7. Ersetzen Sie den vorhandenen Code mit folgendem Code.

```
'use strict';

exports.handler = (event, context, callback) => {
    console.log('LogEC2StopInstance');
    console.log('Received event:', JSON.stringify(event, null, 2));
    callback(null, 'Finished');
};
```

8. Wählen Sie Bereitstellen.

Schritt 3: Erstellen einer Regel

Erstellen Sie eine Regel, damit die in Schritt 2 angelegte Lambda-Funktion ausgeführt wird, wenn Sie eine Amazon-EC2-Instance stoppen.

So erstellen Sie eine Regel

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen aus.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Nennen Sie die Regel beispielsweise `TestRule`.
5. Wählen Sie für Event Bus den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie Standard aus. Wenn ein AWS -Service in Ihrem Konto ein Ereignis ausgibt, wird es stets an den Standard-Event-Bus Ihres Kontos weitergeleitet.

6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Weiter aus.
8. Als Event source (Ereignisquelle) wählen Sie AWS -Services aus.
9. Gehen Sie bei Event pattern (Ereignismuster) wie folgt vor:
   a. Wählen Sie für Ereignisquelle die Option EC2 aus der Dropdown-Liste aus.
   b. Wählen Sie als Ereignistyp in der Drop-down-Liste die Option AWS API-Aufruf über CloudTrail aus.
   c. Wählen Sie Spezifische Operation(en) aus und geben Sie `StopInstances` ein.
10. Wählen Sie Weiter aus.
11. Bei Target types (Zieltypen) wählen Sie AWS -Service aus.
12. Wählen Sie für Ziel auswählen die Option Lambda-Funktion aus der Dropdown-Liste aus.
13. Wählen Sie für Funktion die Lambda-Funktion aus, die Sie im Abschnitt Schritt 1: Erstellen einer Lambda-Funktion erstellt haben. Wählen Sie in diesem Beispiel `LogEC2StopInstance` aus.
14. Wählen Sie Weiter.
15. Wählen Sie Weiter.
16. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 4: Testen der Regel

Sie können Ihre Regeln testen, indem Sie eine Amazon EC2-Instance mithilfe der Amazon EC2-Konsole anhalten. Warten Sie einige Minuten, bis die Instance beendet ist, und überprüfen Sie dann Ihre AWS Lambda Metriken auf der CloudWatch Konsole, um sicherzustellen, dass Ihre Funktion ausgeführt wurde.

Testen der Regel durch Anhalten einer Instance

1. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
2. Starten Sie eine Instance. Weitere Informationen finden Sie unter Launch Your Instance im Amazon EC2 EC2-Benutzerhandbuch.
3. Halten Sie die Instance an. Weitere Informationen finden Sie unter Stoppen und Starten Ihrer Instance im Amazon EC2 EC2-Benutzerhandbuch.
4. So zeigen Sie die Ausgabe aus der Lambda-Funktion an:

a. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
b. Wählen Sie im Navigationsbereich Protokolle aus.
c. Wählen Sie den Namen der Protokollgruppe für Ihre Lambda-Funktion aus (`/aws/lambda/function-name`).
d. Wählen Sie den Namen des Protokoll-Streams aus, um die von der Funktion für die von Ihnen angehaltene Instance bereitgestellten Daten anzuzeigen.

5. (Optional) Beenden Sie die angehaltene Instance, wenn Sie fertig sind. Weitere Informationen finden Sie unter Terminate Your Instance im Amazon EC2 EC2-Benutzerhandbuch.

Schritt 5: Bestätigen des Erfolgs

Wenn Sie das Lambda-Ereignis in den CloudWatch Protokollen sehen, haben Sie dieses Tutorial erfolgreich abgeschlossen. Wenn das Ereignis nicht in Ihren CloudWatch Protokollen enthalten ist, beginnen Sie mit der Fehlerbehebung, indem Sie überprüfen, ob die Regel erfolgreich erstellt wurde. Wenn die Regel korrekt aussieht, überprüfen Sie, ob der Code Ihrer Lambda-Funktion korrekt ist.

Schritt 6: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Indem Sie AWS Ressourcen löschen, die Sie nicht mehr verwenden, verhindern Sie, dass Ihr AWS Konto unnötig belastet wird.

Um die EventBridge Regel (n) zu löschen

1. Öffnen Sie die Seite Regeln der EventBridge Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

So löschen Sie die Lambda-Funktion(en)

1. Öffnen Sie die Seite Funktionen der Lambda-Konsole.
2. Wählen Sie die Funktion(en) aus, die Sie erstellt haben.
3. Wählen Sie Aktionen, Löschen aus.
4. Wählen Sie Delete (Löschen) aus.

Um die CloudTrail Spur (en) zu löschen

1. Öffnen Sie die Trails-Seite der CloudTrail Konsole.
2. Wählen Sie den/die Trail(s) aus, den/die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Löschen.

Tutorial: Den Status einer Amazon EC2 EC2-Instance protokollieren mit EventBridge

Sie können eine AWS Lambda-Funktion erstellen, die eine Statusänderung einer Amazon-EC2-Instance protokolliert. Anschließend können Sie eine Regel erstellen, die Ihre Lambda-Funktion ausführt, sobald ein Statusübergang oder ein Übergang zu einem oder mehreren Status stattfindet, die für sie von Interesse sind. In diesem Tutorial protokollieren Sie den Start einer neuen Instance.

Schritte:

Schritt 1: Erstellen einer AWS Lambda -Funktion

Erstellen Sie eine Lambda-Funktion, um die Ereignisse bezüglich der Änderung des Status zu protokollieren. Wenn Sie die Regel in Schritt 2 erstellen, geben Sie diese Funktion an.

Eine Lambda-Funktion erstellen

1. Öffnen Sie die AWS Lambda Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie Funktion erstellen.
3. Wählen Sie Von Grund auf neu schreiben aus.
4. Geben Sie einen Namen und eine Beschreibung für die Lambda-Funktion ein. Geben Sie der Funktion beispielsweise den Namen `LogEC2InstanceStateChange`.
5. Behalten Sie die übrigen Optionen als Standardwerte bei und wählen Sie Funktion erstellen aus.
6. Doppelklicken Sie auf der Registerkarte Code der Funktionsseite auf index.js.
7. Ersetzen Sie den vorhandenen Code mit folgendem Code.

```
'use strict';

exports.handler = (event, context, callback) => {
    console.log('LogEC2InstanceStateChange');
```

```
    console.log('Received event:', JSON.stringify(event, null, 2));
    callback(null, 'Finished');
};
```

8. Wählen Sie Deploy (Bereitstellen) aus.

Schritt 2: Erstellen einer Regel

Erstellen Sie eine Regel, um die Lambda-Funktion auszuführen, die Sie in Schritt 1 erstellt haben. Die Regel wird ausgeführt, wenn Sie eine Amazon-EC2-Instance starten.

Um die EventBridge Regel zu erstellen

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen aus.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Nennen Sie die Regel beispielsweise `TestRule`.
5. Wählen Sie für Event Bus den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie Standard aus. Wenn ein AWS -Service in Ihrem Konto ein Ereignis ausgibt, wird es stets an den Standard-Event-Bus Ihres Kontos weitergeleitet.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Weiter aus.
8. Als Event source (Ereignisquelle) wählen Sie AWS -Services aus.
9. Gehen Sie bei Event pattern (Ereignismuster) wie folgt vor:
    a. Wählen Sie für Ereignisquelle die Option EC2 aus der Dropdown-Liste aus.
    b. Wählen Sie für Ereignistyp die Option Benachrichtigung über die Statusänderung der EC2-Instance aus der Dropdown-Liste aus
    c. Wählen Sie Spezifische Statusart(en) und anschließend Wird ausgeführt aus der Dropdown-Liste aus.
    d. Wählen Sie Beliebige Instance aus.
10. Wählen Sie Weiter aus.
11. Bei Target types (Zieltypen) wählen Sie AWS -Service aus.

12. Wählen Sie für Ziel auswählen die Option Lambda-Funktion aus der Dropdown-Liste aus.
13. Wählen Sie für Funktion die Lambda-Funktion aus, die Sie im Abschnitt Schritt 1: Erstellen einer Lambda-Funktion erstellt haben. Wählen Sie in diesem Beispiel `LogEC2InstanceStateChange` aus.
14. Wählen Sie Weiter.
15. Wählen Sie Weiter.
16. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 3: Testen der Regel

Sie können Ihre Regeln testen, indem Sie eine Amazon EC2-Instance mithilfe der Amazon EC2-Konsole anhalten. Warten Sie einige Minuten, bis die Instance beendet ist, und überprüfen Sie dann Ihre AWS Lambda Messwerte auf der CloudWatch Konsole, um sicherzustellen, dass Ihre Funktion ausgeführt wurde.

Testen der Regel durch Anhalten einer Instance

1. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
2. Starten Sie eine Instance. Weitere Informationen finden Sie unter Launch Your Instance im Amazon EC2 EC2-Benutzerhandbuch.
3. Halten Sie die Instance an. Weitere Informationen finden Sie unter Stoppen und Starten Ihrer Instance im Amazon EC2 EC2-Benutzerhandbuch.
4. So zeigen Sie die Ausgabe aus der Lambda-Funktion an:
    a. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
    b. Wählen Sie im Navigationsbereich Protokolle aus.
    c. Wählen Sie den Namen der Protokollgruppe für Ihre Lambda-Funktion aus (`/aws/lambda/function-name`).
    d. Wählen Sie den Namen des Protokoll-Streams aus, um die von der Funktion für die von Ihnen angehaltene Instance bereitgestellten Daten anzuzeigen.
5. (Optional) Beenden Sie die angehaltene Instance, wenn Sie fertig sind. Weitere Informationen finden Sie unter Terminate Your Instance im Amazon EC2 EC2-Benutzerhandbuch.

Schritt 4: Bestätigen des Erfolgs

Wenn Sie das Lambda-Ereignis in den CloudWatch Protokollen sehen, haben Sie dieses Tutorial erfolgreich abgeschlossen. Wenn das Ereignis nicht in Ihren CloudWatch Protokollen enthalten ist, beginnen Sie mit der Fehlerbehebung, indem Sie überprüfen, ob die Regel erfolgreich erstellt wurde. Wenn die Regel korrekt aussieht, überprüfen Sie, ob der Code Ihrer Lambda-Funktion korrekt ist.

Schritt 5: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Indem Sie AWS Ressourcen löschen, die Sie nicht mehr verwenden, verhindern Sie, dass Ihr AWS Konto unnötig belastet wird.

Um die EventBridge Regel (n) zu löschen

1. Öffnen Sie die Seite Regeln der EventBridge Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

So löschen Sie die Lambda-Funktion(en)

1. Öffnen Sie die Seite Funktionen der Lambda-Konsole.
2. Wählen Sie die Funktion(en) aus, die Sie erstellt haben.
3. Wählen Sie Aktionen, Löschen aus.
4. Wählen Sie Löschen.

Tutorial: Protokollieren von Amazon-S3-Operationen auf Objektebene mit EventBridge

Sie können die API-Operationen auf Objektebene der Amazon-S3-Buckets protokollieren. Bevor Amazon EventBridge diese Ereignisse abgleichen kann, müssen Sie mit AWS CloudTrail einen Trail einrichten und für den Empfang dieser Ereignisse konfigurieren.

In diesem Tutorial erstellen Sie einen CloudTrail-Trail, eine AWS Lambda-Funktion und anschließend eine Regel in der EventBridge-Konsole, die diese Funktion als Antwort auf ein S3-Datenereignis aufruft.

Schritte:

- Schritt 1: Konfigurieren des AWS CloudTrail-Trail
- Schritt 2: Erstellen einer AWS Lambda-Funktion
- Schritt 3: Erstellen einer Regel
- Schritt 4: Testen der Regel
- Schritt 5: Bestätigen des Erfolgs
- Schritt 6: Bereinigen Ihrer Ressourcen

Schritt 1: Konfigurieren des AWS CloudTrail-Trail

Zum Protokollieren von Datenereignissen für einen S3 Bucket in AWS CloudTrail und EventBridge erstellen Sie zunächst einen Trail. Ein Trail erfasst API-Aufrufe und zugehörige Ereignisse für Ihr Konto und stellt die Protokolldateien anschließend in einem S3 Bucket bereit, den Sie angegeben haben. Sie können einen vorhandenen Trail aktualisieren oder einen Trail erstellen.

Weitere Informationen finden Sie unter Datenereignisse im Benutzerhandbuch für AWS CloudTrail.

Sie erstellen einen Trail wie folgt:

1. Öffnen Sie die CloudTrail-Konsole unter https://console.aws.amazon.com/cloudtrail/.
2. Wählen Sie Trails, Create trail (Trail erstellen).
3. Geben Sie unter Trail name einen Namen für den Trail ein.
4. Wählen Sie für Speicherort die Option Neuen S3-Bucket erstellen aus.
5. Geben Sie einen AWS KMS-Alias für den KMS-Schlüssel ein.

6. Wählen Sie Next (Weiter).
7. Wählen Sie für Ereignistyp die Option Datenereignisse aus.
8. Führen Sie für Datenereignisse eine der folgenden Aktionen aus:

   - Um Datenereignisse für alle Amazon S3-Objekte in einem Bucket zu protokollieren, geben Sie einen S3-Bucket und ein leeres Präfix an. Wenn ein Ereignis auf einem Objekt in diesem -Bucket eintritt, wird das Ereignis vom Trail verarbeitet und protokolliert.
   - Geben Sie einen S3-Bucket und das Objektpräfix an, um Datenereignisse für bestimmte Amazon-S3-Objekte in einem Bucket zu protokollieren. Tritt ein Ereignis auf einem Objekt in dem -Bucket auf und das Objekt beginnt mit dem angegebenen Präfix, wird das Ereignis vom Trail verarbeitet und protokolliert.
9. Wählen Sie für jede Ressource aus, ob Lese-Ereignisse, Schreib-Ereignisse oder beide protokolliert werden sollen.
10. Wählen Sie Next (Weiter).
11. Wählen Sie Create Trail (Trail erstellen) aus.

Schritt 2: Erstellen einer AWS Lambda-Funktion

Erstellen Sie eine Lambda-Funktion, um Datenereignisse für die S3 Buckets zu protokollieren.

So erstellen Sie eine Lambda-Funktion:

1. Öffnen Sie die AWS Lambda-Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie Create function (Funktion erstellen).
3. Wählen Sie Author from scratch aus.
4. Geben Sie einen Namen und eine Beschreibung für die Lambda-Funktion ein. Geben Sie der Funktion beispielsweise den Namen `LogS3DataEvents`.
5. Behalten Sie die übrigen Optionen als Standardwerte bei und wählen Sie Funktion erstellen aus.
6. Doppelklicken Sie auf der Registerkarte Code der Funktionsseite auf index.js.
7. Ersetzen Sie den vorhandenen Code mit folgendem Code.

```
'use strict';

exports.handler = (event, context, callback) => {
    console.log('LogS3DataEvents');
    console.log('Received event:', JSON.stringify(event, null, 2));
```

```
    callback(null, 'Finished');
};
```

8. Wählen Sie Bereitstellen aus.

Schritt 3: Erstellen einer Regel

Erstellen Sie eine Regel, um die Lambda-Funktion auszuführen, die Sie in Schritt 2 erstellt haben. Diese Regel wird als Antwort auf ein Amazon-S3-Datenereignis ausgeführt.

So erstellen Sie eine Regel

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Nennen Sie die Regel beispielsweise `TestRule`.
5. Wählen Sie für Event Bus den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie Standard aus. Wenn ein AWS-Service in Ihrem Konto ein Ereignis ausgibt, wird es stets an den Standard-Event-Bus Ihres Kontos weitergeleitet.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Next (Weiter).
8. Als Event source (Ereignisquelle) wählen Sie AWS-Services aus.
9. Gehen Sie bei Event pattern (Ereignismuster) wie folgt vor:
    a. Wählen Sie für Ereignisquelle die Option Simple Storage Service (S3) aus der Dropdown-Liste aus.
    b. Wählen Sie für Ereignistyp die Option API-Aufruf auf Objektebene über CloudTrail aus der Dropdown-Liste aus.
    c. Wählen Sie Spezifische Operation(en) und dann die Option PutObject aus.
    d. Standardmäßig gilt die Regel für Datenereignisse in allen Buckets in der Region. Damit die Datenereignisse für bestimmte Buckets gelten, wählen Sie Specify bucket(s) by name (Bucket(s) nach Name angeben) aus und geben dann einen oder mehrere Buckets ein.
10. Wählen Sie Next (Weiter).

11. Bei Target types (Zieltypen) wählen Sie AWS-Service aus.
12. Wählen Sie für Ziel auswählen die Option Lambda-Funktion aus der Dropdown-Liste aus.
13. Wählen Sie für Funktion die `LogS3DataEvents`-Lambda-Funktion aus, die Sie in Schritt 1 erstellt haben.
14. Wählen Sie Next (Weiter).
15. Wählen Sie Next (Weiter).
16. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 4: Testen der Regel

Um die Regel zu testen, fügen Sie ein Objekt in Ihrem S3-Bucket ein. Sie können überprüfen, ob die Lambda-Funktion aufgerufen wurde.

So zeigen Sie die Protokolle für die Lambda-Funktion an

1. Öffnen Sie die CloudWatch-Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Logs (Logs) aus.
3. Wählen Sie den Namen der Protokollgruppe für Ihre Lambda-Funktion aus (`/aws/lambda/`*`function-name`*).
4. Wählen Sie den Namen des Protokoll-Streams aus, um die von der Funktion für die von Ihnen gestartete Instance bereitgestellten Daten anzuzeigen.

Sie können auch die CloudTrail-Protokolle im S3-Bucket überprüfen, den Sie für den Trail angegeben haben. Weitere Informationen finden Sie unter Abrufen und Anzeigen der CloudTrail-Protokolldateien im AWS CloudTrail-Benutzerhandbuch.

Schritt 5: Bestätigen des Erfolgs

Wenn Sie das Lambda-Ereignis in den CloudWatch-Protokollen sehen, haben Sie dieses Tutorial erfolgreich abgeschlossen. Wenn das Ereignis nicht in Ihren CloudWatch-Protokollen enthalten ist, beginnen Sie mit der Fehlerbehebung, indem Sie überprüfen, ob die Regel erfolgreich erstellt wurde. Wenn die Regel korrekt aussieht, überprüfen Sie, ob der Code Ihrer Lambda-Funktion korrekt ist.

Schritt 6: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Durch das Löschen von AWS-Ressourcen, die Sie nicht mehr verwenden, können Sie verhindern, dass unnötige Gebühren in Ihrem AWS-Konto anfallen.

So löschen Sie die EventBridge-Regel(n)

1. Öffnen Sie die Seite Regeln der EventBridge-Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

So löschen Sie die Lambda-Funktion(en)

1. Öffnen Sie die Seite Funktionen der Lambda-Konsole.
2. Wählen Sie die Funktion(en) aus, die Sie erstellt haben.
3. Wählen Sie Aktionen, Löschen aus.
4. Wählen Sie Delete (Löschen).

So löschen Sie den/die CloudTrail-Trail(s)

1. Öffnen Sie die Seite Trails der CloudTrail-Konsole.
2. Wählen Sie den/die Trail(s) aus, den/die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

Tutorial: Ereignisse mithilfe EventBridge des Schemas an einen Amazon Kinesis-Stream senden `aws.events`

Sie können AWS API-Aufrufereignisse EventBridge an einen Amazon Kinesis Kinesis-Stream senden, Kinesis Data Streams Streams-Anwendungen erstellen und große Datenmengen verarbeiten. In diesem Tutorial erstellen Sie einen Kinesis-Stream und anschließend eine Regel in der EventBridge Konsole, die Ereignisse an diesen Stream sendet, wenn eine Amazon EC2 EC2-Instance stoppt.

Schritte:

- Voraussetzungen
- Schritt 1: Erstellen eines Amazon-Kinesis-Streams
- Schritt 2: Erstellen einer Regel
- Schritt 3: Testen der Regel
- Schritt 4: Überprüfen, ob das Ereignis gesendet wurde
- Schritt 5: Bereinigen Ihrer Ressourcen

Voraussetzungen

In diesem Tutorial verwenden Sie Folgendes:

- Verwenden Sie den AWS CLI , um mit Kinesis-Streams zu arbeiten.

  Informationen zur AWS CLI Installation von finden Sie unter Installation, Aktualisierung und Deinstallation der AWS CLI Version 2.

> **Note**
>
> In diesem Tutorial werden AWS Ereignisse und die integrierte `aws.events` Schemaregistrierung verwendet. Sie können auch eine EventBridge Regel erstellen, die auf dem Schema Ihrer benutzerdefinierten Ereignisse basiert, indem Sie sie manuell zu einer benutzerdefinierten Schemaregistrierung hinzufügen oder die Schemaerkennung verwenden. Weitere Informationen zu Schemata finden Sie unter ???. Weitere Informationen zum Erstellen einer Regel mithilfe anderer Ereignismusteroptionen finden Sie unter ???.

Schritt 1: Erstellen eines Amazon-Kinesis-Streams

Verwenden Sie an einer Befehlszeile den Befehl, um einen Stream zu `create-stream` AWS CLI erstellen.

```
aws kinesis create-stream --stream-name test --shard-count 1
```

Wenn der Stream-Status `ACTIVE` ist, ist der Stream bereit. Verwenden Sie den Befehl `describe-stream`, um den Status eines Streams zu überprüfen.

```
aws kinesis describe-stream --stream-name test
```

Schritt 2: Erstellen einer Regel

Erstellen Sie eine Regel, mit der Ereignisse an den Stream gesendet werden, wenn eine Amazon EC2-Instance gestoppt wird.

So erstellen Sie eine Regel

1. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen aus.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Nennen Sie die Regel beispielsweise `TestRule`.
5. Wählen Sie für Event Bus die Option Standard aus.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Weiter aus.
8. Wählen Sie als Eventquelle AWS Events oder EventBridge Partnerevents aus.
9. Wählen Sie für Erstellungsmethode die Option Schema verwenden aus.
10. Gehen Sie bei Event pattern (Ereignismuster) wie folgt vor:

    a. Wählen Sie für Schematyp die Option Schema aus der Schemaregistrierung auswählen aus.
    b. Wählen Sie für Schemaregistrierung die Option aws.events aus der Dropdown-Liste aus.
    c. Wählen Sie für Schema aws.ec2 @EC2 InstanceStateChangeNotification aus der Dropdownliste aus.

EventBridge zeigt das Ereignisschema unter Models an.

EventBridge zeigt neben allen Eigenschaften, die für das Ereignis und nicht für das Ereignismuster erforderlich sind, ein rotes Sternchen an.

d. Legen Sie für Modelle die folgenden Eigenschaften für den Ereignisfilter fest:

   i. Wählen Sie + Bearbeiten neben der Statuseigenschaft aus.

      Lassen Sie das Feld Beziehung leer. Geben Sie für Wert `running` ein. Wählen Sie Festlegen aus.

   ii. Wählen Sie + Bearbeiten neben der Quelleigenschaft aus.

      Lassen Sie das Feld Beziehung leer. Geben Sie für Wert `aws.ec2` ein. Wählen Sie Festlegen aus.

   iii. Wählen Sie + Bearbeiten neben der Detailtypeigenschaft aus.

      Lassen Sie das Feld Beziehung leer. Geben Sie für Wert `EC2 Instance State-change Notification` ein. Wählen Sie Festlegen aus.

e. Wählen Sie Ereignismuster in JSON generieren aus, um das von Ihnen erstellte Ereignismuster anzuzeigen.

   EventBridge zeigt das Ereignismuster in JSON an:

```
{
  "detail": {
    "state": ["running"]
  },
  "detail-type": ["EC2 Instance State-change Notification"],
  "source": ["aws.ec2"]
}
```

11. Wählen Sie Weiter aus.
12. Bei Target types (Zieltypen) wählen Sie AWS -Service aus.
13. Wählen Sie für Ziel auswählen die Option Kinesis-Stream aus der Dropdown-Liste aus.
14. Wählen Sie für Stream den Kinesis-Stream aus, den Sie im Abschnitt Schritt 1: Erstellen eines Amazon-Kinesis-Streams erstellt haben. Wählen Sie in diesem Beispiel `test` aus.
15. Wählen Sie für Ausführungsrolle die Option Eine neue Rolle für diese spezifische Ressource erstellen aus.

16. Wählen Sie Weiter.
17. Wählen Sie Weiter.
18. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 3: Testen der Regel

Halten Sie zum Testen der Regel eine Amazon EC2-Instance an. Warten Sie einige Minuten, bis die Instanz beendet ist, und überprüfen Sie dann CloudWatch anhand Ihrer Messwerte, ob Ihre Funktion ausgeführt wurde.

Testen der Regel durch Anhalten einer Instance

1. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
2. Starten Sie eine Instance. Weitere Informationen finden Sie unter Launch Your Instance im Amazon EC2 EC2-Benutzerhandbuch.
3. Öffnen Sie die EventBridge Amazon-Konsole unter https://console.aws.amazon.com/events/.
4. Wählen Sie im Navigationsbereich Regeln aus.

   Wählen Sie den Namen der von Ihnen erstellten Regel und Metrics for the rule (Metriken für die Regel) aus.
5. (Optional) Beenden Sie die Instance, wenn Sie fertig sind. Weitere Informationen finden Sie unter Terminate Your Instance im Amazon EC2 EC2-Benutzerhandbuch.

Schritt 4: Überprüfen, ob das Ereignis gesendet wurde

Mit dem können Sie AWS CLI den Datensatz aus dem Stream abrufen, um zu überprüfen, ob das Ereignis gesendet wurde.

So rufen Sie den Datensatz ab

1. Verwenden Sie an einer Eingabeaufforderung den Befehl `get-shard-iterator`, um mit dem Lesen aus dem Kinesis-Stream zu beginnen.

   ```
   aws kinesis get-shard-iterator --shard-id shardId-000000000000 --shard-iterator-
   type TRIM_HORIZON --stream-name test
   ```

   Es folgt eine Beispielausgabe.

```
{
    "ShardIterator": "AAAAAAAAAAHSywljv0zEgPX4NyKdZ5wryMzP9yALs8NeKbUjp1IxtZs1Sp
+KEd9I6AJ9ZG4lNR1EMi+9Md/nHvtLyxpfhEzYvkTZ4D9DQVz/mBYWRO6OTZRKnW9gd
+efGN2aHFdkH1rJl4BL9Wyrk+ghYG22D2T1Da2EyNSH1+LAbK33gQweTJADBdyMwlo5r6PqcP2dzhg="
}
```

2. Um den Datensatz abzurufen, verwenden Sie den folgenden `get-records`-Befehl. Verwenden Sie den Shard-Iterator aus der Ausgabe im vorherigen Schritt.

```
aws kinesis get-records --shard-
iterator AAAAAAAAAAHSywljv0zEgPX4NyKdZ5wryMzP9yALs8NeKbUjp1IxtZs1Sp
+KEd9I6AJ9ZG4lNR1EMi+9Md/nHvtLyxpfhEzYvkTZ4D9DQVz/mBYWRO6OTZRKnW9gd
+efGN2aHFdkH1rJl4BL9Wyrk+ghYG22D2T1Da2EyNSH1+LAbK33gQweTJADBdyMwlo5r6PqcP2dzhg=
```

   Wenn der Befehl erfolgreich war, werden Datensätze aus dem Stream für eine bestimmte Shard abgefragt. Sie können null oder mehr Datensätze erhalten. Die zurückgegebenen Datensätze stellen möglicherweise nicht alle Datensätze in Ihrem Stream dar. Wenn Sie nicht die erwarteten Daten erhalten, rufen Sie `get-records` weiter auf.

3. Datensätze in Kinesis sind in Base64 codiert. Verwenden Sie einen Base64-Decoder, um die Daten zu decodieren, sodass Sie überprüfen können, ob es sich um das Ereignis handelt, das im JSON-Format an den Stream gesendet wurde.

Schritt 5: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Indem Sie AWS Ressourcen löschen, die Sie nicht mehr verwenden, verhindern Sie, dass Ihr AWS Konto unnötig belastet wird.

Um die EventBridge Regel (n) zu löschen

1. Öffnen Sie die Regelseite der EventBridge Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

So löschen Sie den/die Kinesis-Stream(s)

1. Öffnen Sie die Datenstromseite der Kinesis-Konsole.
2. Wählen Sie den/die Stream(s) aus, den/die Sie erstellt haben.
3. Wählen Sie Aktionen, Löschen aus.
4. Geben Sie delete in das Feld ein und wählen Sie Löschen aus.

Tutorial: Planen automatisierter Amazon-EBS-Snapshots mit EventBridge

Sie können EventBridge-Regeln nach einem Zeitplan ausführen. In diesem Tutorial erstellen Sie einen Snapshot eines vorhandenen Amazon Elastic Block Store (Amazon EBS)-Volumes nach einem Zeitplan. Sie können festlegen, dass alle paar Minuten ein Snapshot erstellt wird, oder einen Cron-Ausdruck verwenden, um den Snapshot zu einer bestimmten Zeit zu erstellen.

> **Important**
>
> Sie müssen die AWS Management Console verwenden, um Regeln mit integrierten Zielen zu erstellen.

Schritte:

- Schritt 1: Erstellen der Regel
- Schritt 2: Testen der Regel
- Schritt 3: Bestätigen des Erfolgs
- Schritt 4: Bereinigen Ihrer Ressourcen

Schritt 1: Erstellen der Regel

Erstellen Sie eine Regel, mit der Snapshots nach einem Zeitplan erstellt werden. Sie können einen Rate-Ausdruck oder einen Cron-Ausdruck verwenden, um den Zeitplan anzugeben. Weitere Informationen finden Sie unter Erstellen einer Amazon-EventBridge-Regel, die nach einem Zeitplan ausgeführt wird.

So erstellen Sie eine Regel

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein.

   Eine Regel darf nicht denselben Namen wie eine andere Regel in derselben Region und auf demselben Event Bus haben.
5. Wählen Sie als Event bus (Event Bus) den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto

übereinstimmt, wählen Sie AWS-Standard-Event-Bus aus. Wenn ein AWS-Service in Ihrem Konto ein Ereignis ausgibt, wird es stets an den Standard-Event-Bus Ihres Kontos weitergeleitet.

6. Wählen Sie unter Rule type (Regeltyp) die Option Schedule (Zeitplan) aus.
7. Wählen Sie Next (Weiter).
8. Wählen Sie für Zeitplanmuster die Option Ein Zeitplan, der regelmäßig ausgeführt wird, z. B. alle 10 Minuten. aus, geben Sie **5** ein und wählen Sie Minuten in der Dropdown-Liste aus.
9. Wählen Sie Next (Weiter).
10. Bei Target types (Zieltypen) wählen Sie AWS-Service aus.
11. Wählen Sie für Ziel auswählen die Option EBS-Snapshot aus der Dropdown-Liste aus.
12. Geben Sie für Volume-ID die Volume-ID des Amazon-EBS-Volumes ein.
13. Wählen Sie für Ausführungsrolle die Option Eine neue Rolle für diese spezifische Ressource erstellen aus.
14. Wählen Sie Next (Weiter).
15. Wählen Sie Next (Weiter).
16. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 2: Testen der Regel

Sie können überprüfen, ob Ihre Regel funktioniert, indem Sie den ersten erstellten Snapshot anzeigen.

Testen Ihrer Regel

1. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
2. Wählen Sie im Navigationsbereich Elastic Block Store und die Option Snapshots aus.
3. Vergewissern Sie sich, dass der erste Snapshot in der Liste angezeigt wird.

Schritt 3: Bestätigen des Erfolgs

Wenn Sie den Snapshot in der Liste sehen, haben Sie dieses Tutorial erfolgreich abgeschlossen. Wenn der Snapshot nicht in der Liste enthalten ist, beginnen Sie mit der Fehlerbehebung, indem Sie überprüfen, ob die Regel erfolgreich erstellt wurde.

Schritt 4: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Durch das Löschen von AWS-Ressourcen, die Sie nicht mehr verwenden, können Sie verhindern, dass unnötige Gebühren in Ihrem AWS-Konto anfallen.

So löschen Sie die EventBridge-Regel(n)

1. Öffnen Sie die Seite Regeln der EventBridge-Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

Tutorial: Senden einer Benachrichtigung, wenn ein Amazon-S3-Objekt erstellt wird

Sie können E-Mail-Benachrichtigungen senden, wenn Amazon Simple Storage Service (Amazon S3)-Objekte mit Amazon EventBridge und Amazon SNS erstellt werden. In diesem Tutorial erstellen Sie ein SNS-Thema und -Abonnement. Anschließend erstellen Sie in der EventBridge-Konsole eine Regel, die Ereignisse an dieses Thema sendet, wenn Amazon-S3-`Object Created`-Ereignisse empfangen werden.

Schritte:

- Voraussetzungen
- Schritt 1: Erstellen eines Amazon-SNS-Themas
- Schritt 2: Erstellen eines Amazon-SNS-Abonnements
- Schritt 3: Erstellen einer Regel
- Schritt 4: Testen der Regel
- Schritt 5: Bereinigen Ihrer Ressourcen

Voraussetzungen

Zum Empfangen von Amazon-S3-Ereignissen in EventBridge müssen Sie EventBridge in der Amazon-S3-Konsole aktivieren. In diesem Tutorial wird davon ausgegangen, dass EventBridge aktiviert ist. Weitere Informationen finden Sie unter Aktivieren von Amazon EventBridge in der S3-Konsole.

Schritt 1: Erstellen eines Amazon-SNS-Themas

Erstellen Sie ein Thema, um die Ereignisse von EventBridge zu erhalten.

Erstellen Sie ein Thema wie folgt

1. Öffnen Sie die Amazon SNS-Konsole unter https://console.aws.amazon.com/sns/v3/home.
2. Wählen Sie im Navigationsbereich Topics (Themen) aus.
3. Wählen Sie Create topic (Thema erstellen) aus.
4. Wählen Sie unter Type (Typ) die Option Standard aus.
5. Geben Sie **`eventbridge-test`** als Namen des Themas ein.

6. Wählen Sie Create topic (Thema erstellen) aus.

Schritt 2: Erstellen eines Amazon-SNS-Abonnements

Erstellen Sie ein Abonnement, um E-Mail-Benachrichtigungen von Amazon S3 zu erhalten, wenn Ereignisse vom Thema empfangen werden.

Erstellen eines Abonnements

1. Öffnen Sie die Amazon SNS-Konsole unter https://console.aws.amazon.com/sns/v3/home.
2. Wählen Sie im Navigationsbereich Subscriptions aus.
3. Wählen Sie Create subscription.
4. Wählen Sie für Thema-ARN das in Schritt 1 erstellte Thema aus. Wählen Sie für dieses Tutorial eventbridge-test aus.
5. Wählen Sie unter Protocol (Protokoll) die Option Email (E-Mail) aus.
6. Geben Sie unter Endpunkt Ihre E-Mail-Adresse ein.
7. Klicken Sie auf Create subscription (Abonnement erstellen).
8. Bestätigen Sie das Abonnement, indem Sie in der E-Mail, die Sie von AWS-Benachrichtigungen erhalten, die Option Abonnement bestätigen auswählen.

Schritt 3: Erstellen einer Regel

Erstellen Sie eine Regel zum Senden von Ereignissen an das Thema, wenn ein Amazon-S3-Objekt erstellt wird.

So erstellen Sie eine Regel

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Nennen Sie die Regel beispielsweise `s3-test`.
5. Wählen Sie für Event Bus die Option Standard aus.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.

7. Wählen Sie Next (Weiter).
8. Wählen Sie für Event source (Ereignisquelle) AWS events or EventBridge partner events (-Ereignisse oder EventBridge-Partnerereignisse).
9. Wählen Sie für Erstellungsmethode die Option Musterformular verwenden aus.
10. Gehen Sie bei Event pattern (Ereignismuster) wie folgt vor:
    a. Wählen Sie für Ereignisquelle die Option AWS-Services aus der Dropdown-Liste aus.
    b. Wählen Sie für AWS-Service die Option Simple Storage Service (S3) aus der Dropdown-Liste aus.
    c. Wählen Sie für Ereignistyp die Option Amazon-S3-Ereignisbenachrichtigung aus der Dropdown-Liste aus.
    d. Wählen Sie Spezifische(s) Ereignis(se) und die Option Objekt erstellt aus der Dropdown-Liste aus.
    e. Wählen Sie Beliebiger Bucket aus.
11. Wählen Sie Next (Weiter).
12. Bei Target types (Zieltypen) wählen Sie AWS-Service aus.
13. Wählen Sie für Ziel auswählen die Option SNS-Thema aus der Dropdown-Liste aus.
14. Wählen Sie für Thema das Amazon-SNS-Thema aus, das Sie im Abschnitt Schritt 1: Erstellen eines SNS-Themas erstellt haben. Wählen Sie in diesem Beispiel `eventbridge-test` aus.
15. Wählen Sie Next (Weiter).
16. Wählen Sie Next (Weiter).
17. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 4: Testen der Regel

Um Ihre Regel zu testen, erstellen Sie ein Amazon-S3-Objekt, indem Sie eine Datei in einen EventBridge-fähigen Bucket hochladen. Warten Sie dann einige Minuten und überprüfen Sie, ob Sie eine E-Mail von AWS-Benachrichtigungen erhalten.

Schritt 5: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Durch das Löschen von AWS-Ressourcen, die Sie nicht mehr verwenden, können Sie verhindern, dass unnötige Gebühren in Ihrem AWS-Konto anfallen.

So löschen Sie das SNS-Thema

1. Öffnen Sie die Seite Themen der SNS-Konsole.
2. Wählen Sie das Thema aus, das Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Geben Sie **`delete me`** ein.
5. Wählen Sie Delete (Löschen).

So löschen Sie das SNS-Abonnement

1. Öffnen Sie die Seite Abonnements der SNS-Konsole.
2. Wählen Sie das von Ihnen erstellte Abonnement aus.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

So löschen Sie die EventBridge-Regel(n)

1. Öffnen Sie die Seite Regeln der EventBridge-Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

Tutorial: Planen von AWS Lambda-Funktionen mit EventBridge

Sie können eine Regel für die Ausführung einer AWS Lambda-Funktion nach einem Zeitplan einrichten. In diesem Tutorial erfahren Sie, wie Sie die AWS Management Console oder AWS CLI verwenden, um die Regel zu erstellen. Wenn Sie die AWS CLI verwenden möchten, sie aber noch nicht installiert haben, finden Sie weitere Informationen unter Installieren, Aktualisieren und Deinstallieren der AWS CLI Version 2.

EventBridge stellt für Zeitpläne in Planungsausdrücken keine Präzision der zweiten Ebene bereit. Die feinste Zeitauflösung bei Verwendung eines Cron-Ausdrucks ist eine Minute. Aufgrund der verteilten Natur von EventBridge und den Zielservices kann es zwischen dem Zeitpunkt der Auslösung der geplanten Regel und dem Zeitpunkt, zu dem der Zielservice die Zielressource ausführt, zu einer Verzögerung von einigen Sekunden kommen.

Schritte:

- Schritt 1: Erstellen einer Lambda-Funktion
- Schritt 2: Erstellen einer Regel
- Schritt 3: Überprüfen der Regel
- Schritt 4: Bestätigen des Erfolgs
- Schritt 5: Bereinigen Ihrer Ressourcen

Schritt 1: Erstellen einer Lambda-Funktion

Erstellen Sie eine Lambda-Funktion, um die geplanten Ereignisse zu protokollieren.

So erstellen Sie eine Lambda-Funktion:

1. Öffnen Sie die AWS Lambda-Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie Create function (Funktion erstellen).
3. Wählen Sie Author from scratch aus.
4. Geben Sie einen Namen und eine Beschreibung für die Lambda-Funktion ein. Geben Sie der Funktion beispielsweise den Namen `LogScheduledEvent`.
5. Behalten Sie die übrigen Optionen als Standardwerte bei und wählen Sie Funktion erstellen aus.
6. Doppelklicken Sie auf der Registerkarte Code der Funktionsseite auf index.js.
7. Ersetzen Sie den vorhandenen Code mit folgendem Code.

```
'use strict';

exports.handler = (event, context, callback) => {
    console.log('LogScheduledEvent');
    console.log('Received event:', JSON.stringify(event, null, 2));
    callback(null, 'Finished');
};
```

8. Wählen Sie Deploy (Bereitstellen) aus.

Schritt 2: Erstellen einer Regel

Erstellen Sie eine Regel, um die Lambda-Funktion, die Sie in Schritt 1 erstellt haben, nach einem Zeitplan auszuführen.

Sie können entweder die Konsole oder die AWS CLI verwenden, um die Regel zu erstellen. Zur Nutzung der AWS CLI erteilen Sie der Regel zunächst die Berechtigung, die Lambda-Funktion aufzurufen. Anschließend können Sie die Regel erstellen und Lambda-Funktion als Ziel hinzufügen.

So erstellen Sie eine Regel (Konsole)

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein.

   Eine Regel darf nicht denselben Namen wie eine andere Regel in derselben Region und auf demselben Event Bus haben.
5. Wählen Sie als Event bus (Event Bus) den Event Bus aus, den Sie dieser Regel zuordnen möchten. Wenn Sie möchten, dass diese Regel mit Ereignissen aus Ihrem eigenen Konto übereinstimmt, wählen Sie AWS-Standard-Event-Bus aus. Wenn ein AWS-Service in Ihrem Konto ein Ereignis ausgibt, wird es stets an den Standard-Event-Bus Ihres Kontos weitergeleitet.
6. Wählen Sie unter Rule type (Regeltyp) die Option Schedule (Zeitplan) aus.
7. Wählen Sie Next (Weiter).
8. Wählen Sie für Zeitplanmuster die Option Ein Zeitplan, der regelmäßig ausgeführt wird, z. B. alle 10 Minuten. aus, geben Sie **5** ein und wählen Sie Minuten in der Dropdown-Liste aus.
9. Wählen Sie Next (Weiter).

10. Bei Target types (Zieltypen) wählen Sie AWS-Service aus.
11. Wählen Sie für Ziel auswählen die Option Lambda-Funktion aus der Dropdown-Liste aus.
12. Wählen Sie für Funktion die Lambda-Funktion aus, die Sie im Abschnitt Schritt 1: Erstellen einer Lambda-Funktion erstellt haben. Wählen Sie in diesem Beispiel `LogScheduledEvent` aus.
13. Wählen Sie Next (Weiter).
14. Wählen Sie Next (Weiter).
15. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

So erstellen Sie eine Regel (AWS CLI)

1. Verwenden Sie den Befehl `put-rule`, um eine Regel zu erstellen, die nach einem Zeitplan ausgeführt wird.

```
aws events put-rule \
--name my-scheduled-rule \
--schedule-expression 'rate(5 minutes)'
```

Wenn diese Regel ausgeführt wird, erstellt sie ein Ereignis und sendet es dann an die Ziele. Es folgt ein Beispielereignis.

```
{
    "version": "0",
    "id": "53dc4d37-cffa-4f76-80c9-8b7d4a4d2eaa",
    "detail-type": "Scheduled Event",
    "source": "aws.events",
    "account": "123456789012",
    "time": "2015-10-08T16:53:06Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:events:us-east-1:123456789012:rule/my-scheduled-rule"
    ],
    "detail": {}
}
```

2. Verwenden Sie den Befehl `add-permission`, um dem EventBridge-Service-Prinzipal (`events.amazonaws.com`) die Berechtigung zur Ausführung der Regel zu gewähren.

```
aws lambda add-permission \
--function-name LogScheduledEvent \
```

```
--statement-id my-scheduled-event \
--action 'lambda:InvokeFunction' \
--principal events.amazonaws.com \
--source-arn arn:aws:events:us-east-1:123456789012:rule/my-scheduled-rule
```

3. Erstellen Sie die Datei `targets.json` mit folgendem Inhalt.

```
[
  {
    "Id": "1",
    "Arn": "arn:aws:lambda:us-east-1:123456789012:function:LogScheduledEvent"
  }
]
```

4. Verwenden Sie den Befehl `put-targets`, um die Lambda-Funktion, die Sie in Schritt 1 erstellt haben, zur Regel hinzuzufügen.

```
aws events put-targets --rule my-scheduled-rule --targets file://targets.json
```

Schritt 3: Überprüfen der Regel

Warten Sie mindestens fünf Minuten nach Abschluss von Schritt 2, und dann können Sie prüfen, ob die Lambda-Funktion aufgerufen wurde.

Anzeigen der Ausgabe aus der Lambda-Funktion

1. Öffnen Sie die CloudWatch-Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Logs (Logs) aus.
3. Wählen Sie den Namen der Protokollgruppe für Ihre Lambda-Funktion aus (`/aws/lambda/function-name`).
4. Wählen Sie den Namen des Protokoll-Streams aus, um die von der Funktion für die von Ihnen gestartete Instance bereitgestellten Daten anzuzeigen.

Schritt 4: Bestätigen des Erfolgs

Wenn Sie das Lambda-Ereignis in den CloudWatch-Protokollen sehen, haben Sie dieses Tutorial erfolgreich abgeschlossen. Wenn das Ereignis nicht in Ihren CloudWatch-Protokollen enthalten ist,

beginnen Sie mit der Fehlerbehebung, indem Sie überprüfen, ob die Regel erfolgreich erstellt wurde. Wenn die Regel korrekt aussieht, überprüfen Sie, ob der Code Ihrer Lambda-Funktion korrekt ist.

Schritt 5: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Durch das Löschen von AWS-Ressourcen, die Sie nicht mehr verwenden, können Sie verhindern, dass unnötige Gebühren in Ihrem AWS-Konto anfallen.

So löschen Sie die EventBridge-Regel(n)

1. Öffnen Sie die Seite Regeln der EventBridge-Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

So löschen Sie die Lambda-Funktion(en)

1. Öffnen Sie die Seite Funktionen der Lambda-Konsole.
2. Wählen Sie die Funktion(en) aus, die Sie erstellt haben.
3. Wählen Sie Aktionen, Löschen aus.
4. Wählen Sie Delete (Löschen).

Amazon-EventBridge-Tutorials zur Integration mit SaaS-Anbietern

EventBridge kann direkt mit SaaS-Partneranwendungen und -services arbeiten, um Ereignisse zu senden und zu empfangen. Die folgenden Tutorials zeigen, wie Sie EventBridge mit SaaS-Partnern integrieren.

Tutorials:

- Tutorial: Erstellen einer Verbindung zu Datadog als API-Ziel
- Tutorial: Erstellen einer Verbindung zu Salesforce als API-Ziel
- Tutorial: Erstellen einer Verbindung zu Zendesk als API-Ziel

Tutorial: Erstellen einer Verbindung zu Datadog als API-Ziel

Sie können EventBridge verwenden, um Ereignisse an Drittanbieterservices weiterzuleiten, z. B. Datadog.

In diesem Tutorial verwenden Sie die EventBridge-Konsole, um eine Verbindung zu Datadog, ein API-Ziel, das auf Datadog verweist, und eine Regel zum Weiterleiten von Ereignissen an Datadog zu erstellen.

Schritte:

Voraussetzungen

Zum Durcharbeiten dieses Tutorials benötigen Sie die folgenden Ressourcen:

- Ein Datadog-Konto
- Einen Datadog-API-Schlüssel
- Einen EventBridge-fähigen Amazon Simple Storage Service (Amazon S3)-Bucket

Schritt 1: Erstellen einer Verbindung

Zum Senden von Ereignissen an Datadog müssen Sie zunächst eine Verbindung zur Datadog-API herstellen.

So erstellen Sie die Verbindung

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich API-Ziele aus.
3. Wählen Sie die Registerkarte Verbindungen und dann Verbindung erstellen aus.

4. Geben Sie einen Namen und eine Beschreibung für die Verbindung ein. Geben Sie beispielsweise **`Datadog`** als Name und **`Datadog API Connection`** als Beschreibung ein.
5. Wählen Sie für Autorisierungstyp die Option API Key aus.
6. Geben Sie für API-Schlüsselname **`DD-API-KEY`** ein.
7. Fügen Sie für Wert Ihren geheimen Datadog-API-Schlüssel ein.
8. Wählen Sie Erstellen aus.

Schritt 2: Erstellen eines API-Ziels

Nachdem Sie die Verbindung erstellt haben, erstellen Sie als Nächstes das API-Ziel, das als Ziel der Regel verwendet werden soll.

So erstellen Sie das API-Ziel

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich API-Ziele aus.
3. Wählen Sie API-Ziel erstellen aus.
4. Geben Sie einen Namen und eine Beschreibung für das API-Ziel ein. Geben Sie beispielsweise **`DatadogAD`** für den Namen und **`Datadog API Destination`** für die Beschreibung ein.
5. Geben Sie für API-Zielendpunkt **`https://http-intake.logs.datadoghq.com/api/v2/logs`** ein.
6. Wählen Sie in HTTP method POST.
7. Geben Sie für Aufrufratenlimit **`300`** ein.
8. Wählen Sie für Verbindung die Option Vorhandene Verbindung verwenden und die `Datadog`-Verbindung aus, die Sie in Schritt 1 erstellt haben.
9. Wählen Sie Erstellen aus.

Schritt 3: Erstellen einer Regel

Als Nächstes erstellen Sie eine Regel zum Senden von Ereignissen an Datadog, wenn ein Amazon-S3-Objekt erstellt wird.

So erstellen Sie eine Regel

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.

2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Geben Sie beispielsweise **`DatadogRule`** für den Namen und **`Rule to send events to Datadog for S3 object creation`** für die Beschreibung ein.
5. Bei Event bus (Ereignisbus) wählen Sie default (Standard) aus.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Next (Weiter).
8. Wählen Sie für Event source (Ereignisquelle) Other (Andere) aus.
9. Geben Sie für Ereignismuster Folgendes ein:

```
{
  "source": ["aws.s3"]
}
```

10. Wählen Sie Next (Weiter).
11. Wählen Sie für Zieltypen die Option EventBridge-API-Ziel aus.
12. Wählen Sie für API-Ziel die Option Vorhandenes API-Ziel verwenden und dann das `DatadogAD`-Ziel aus, das Sie in Schritt 2 erstellt haben.
13. Wählen Sie für Ausführungsrolle die Option Eine neue Rolle für diese spezifische Ressource erstellen aus.
14. Gehen Sie für Weitere Einstellungen wie folgt vor:

    a. Wählen Sie für Zieleingabe konfigurieren die Option Eingabe-Transformator aus der Dropdown-Liste aus.

    b. Wählen Sie Eingabe-Transformator konfigurieren aus.

    c. Geben Sie für Beispielereignisse Folgendes ein:

```
{
  "detail":[]
}
```

    d. Gehen Sie für Zieleingabe-Transformator wie folgt vor:

        i. Geben Sie für Eingabepfad Folgendes ein:

```
{"detail":"$.detail"}
```

ii. Geben Sie für Eingabevorlage Folgendes ein:

```
{"message": <detail>}
```

e. Wählen Sie Bestätigen aus.

15. Wählen Sie Next (Weiter).
16. Wählen Sie Next (Weiter).
17. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 4: Testen der Regel

Um Ihre Regel zu testen, erstellen Sie ein Amazon-S3-Objekt, indem Sie eine Datei in einen EventBridge-fähigen Bucket hochladen. Das erstellte Objekt wird in der Datadog-Logs-Konsole protokolliert.

Schritt 5: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Durch das Löschen von AWS-Ressourcen, die Sie nicht mehr verwenden, können Sie verhindern, dass unnötige Gebühren in Ihrem AWS-Konto anfallen.

So löschen Sie die EventBridge-Verbindung(en)

1. Öffnen Sie die Seite API-Ziel der EventBridge-Konsole.
2. Wählen Sie die Registerkarte Connections (Verbindungen) aus.
3. Wählen Sie die Verbindung(en) aus, die Sie erstellt haben.
4. Wählen Sie Delete (Löschen).
5. Geben Sie den Namen der Verbindung ein und wählen Sie Löschen aus.

So löschen Sie das/die EventBridge-API-Ziel(e)

1. Öffnen Sie die Seite API-Ziel der EventBridge-Konsole.
2. Wählen Sie das/die API-Ziel(e) aus, das/die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).

4. Geben Sie den Namen des API-Ziels ein und wählen Sie Löschen aus.

So löschen Sie die EventBridge-Regel(n)

1. Öffnen Sie die Seite Regeln der EventBridge-Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

Tutorial: Erstellen einer Verbindung zu Salesforce als API-Ziel

Sie können verwenden EventBridge , um Ereignisse an Drittanbieterservices wie weiterzuleitenSalesforce.

In diesem Tutorial verwenden Sie die EventBridge Konsole, um eine Verbindung zu Salesforce, ein API-Ziel, das auf verweistSalesforce, und eine Regel zum Weiterleiten von Ereignissen an herzustellenSalesforce.

Schritte:

- Voraussetzungen
- Schritt 1: Erstellen einer Verbindung
- Schritt 2: Erstellen eines API-Ziels
- Schritt 3: Erstellen einer Regel
- Schritt 4: Testen der Regel
- Schritt 5: Bereinigen Ihrer Ressourcen

Voraussetzungen

Zum Durcharbeiten dieses Tutorials benötigen Sie die folgenden Ressourcen:

- Ein Salesforce-Konto
- Eine mit Salesforce verbundene App
- Ein Salesforce-Sicherheitstoken
- Ein benutzerdefiniertes Salesforce-Plattformereignis
- Ein EventBridge-fähiger Amazon Simple Storage Service (Amazon S3)-Bucket.

Schritt 1: Erstellen einer Verbindung

Zum Senden von Ereignissen an Salesforce müssen Sie zunächst eine Verbindung zur Salesforce-API herstellen.

So erstellen Sie die Verbindung

1. Öffnen Sie die Amazon- EventBridge Konsole unter https://console.aws.amazon.com/events/.

2. Wählen Sie im Navigationsbereich API-Ziele aus.
3. Wählen Sie die Registerkarte Verbindungen und dann Verbindung erstellen aus.
4. Geben Sie einen Namen und eine Beschreibung für die Verbindung ein. Geben Sie beispielsweise **`Salesforce`** als Name und **`Salesforce API Connection`** als Beschreibung ein.
5. Wählen Sie für Zieltyp die Option Partner und für Partnerziele die Option Salesforce aus der Dropdown-Liste aus.
6. Geben Sie für Autorisierungsendpunkt einen der folgenden Werte ein:

   - Wenn Sie eine Produktionsorganisation verwenden, geben Sie **`https://`*`MyDomainName`*`.my.salesforce.com./services/oauth2/token`** ein.
   - Wenn Sie eine Sandbox ohne erweiterte Domains verwenden, geben Sie **`https://`*`MyDomainName`*`--`*`SandboxName`*`.my. salesforce.com/services /oauth2/token`** ein.
   - Wenn Sie eine Sandbox mit erweiterten Domains verwenden, geben Sie **`https://`*`MyDomainName`*`-- `*`SandboxName`*`.sandbox.my.salesforce.com/services/oauth2/token`** ein.
7. Wählen Sie für HTTP-Methode die Option POST aus der Dropdown-Liste aus.
8. Geben Sie für Client-ID die Client-ID aus der mit Salesforce verbundenen App ein.
9. Geben Sie für Client-Secret das Client-Secret aus der mit Salesforce verbundenen App ein.
10. Geben Sie für OAuth-Http-Parameter das folgende Schlüssel-Wert-Paar ein:

Key (Schlüssel)	Value (Wert)
Gewährungsart	client_credentials

11. Wählen Sie Erstellen.

Schritt 2: Erstellen eines API-Ziels

Nachdem Sie die Verbindung erstellt haben, erstellen Sie als Nächstes das API-Ziel, das als Ziel der Regel verwendet werden soll.

So erstellen Sie das API-Ziel

1. Öffnen Sie die Amazon- EventBridge Konsole unter https://console.aws.amazon.com/events/.

2. Wählen Sie im Navigationsbereich API-Ziele aus.
3. Wählen Sie API-Ziel erstellen aus.
4. Geben Sie einen Namen und eine Beschreibung für das API-Ziel ein. Geben Sie beispielsweise **`SalesforceAD`** für den Namen und **`Salesforce API Destination`** für die Beschreibung ein.
5. Geben Sie für API-Zielendpunkt **`https://`*`MyDomainName`*`.my.salesforce.com/services/data/v54.0/sobjects/`*`MyEvent__e`** ein, wobei Myevent__e das Plattformereignis ist, an das Sie Informationen senden möchten.
6. Wählen Sie für HTTP-Methode die Option POST aus der Dropdown-Liste aus.
7. Geben Sie für Begrenzung der Aufrufrate **`300`** ein.
8. Wählen Sie für Verbindung die Option Vorhandene Verbindung verwenden und die `Salesforce`-Verbindung aus, die Sie in Schritt 1 erstellt haben.
9. Wählen Sie Erstellen.

Schritt 3: Erstellen einer Regel

Als Nächstes erstellen Sie eine Regel zum Senden von Ereignissen an Salesforce, wenn ein Amazon-S3-Objekt erstellt wird.

So erstellen Sie eine Regel

1. Öffnen Sie die Amazon- EventBridge Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen aus.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Geben Sie beispielsweise **`SalesforceRule`** für den Namen und **`Rule to send events to Salesforce for S3 object creation`** für die Beschreibung ein.
5. Bei Event bus (Ereignisbus) wählen Sie default (Standard) aus.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Weiter aus.
8. Wählen Sie für Event source (Ereignisquelle) Other (Andere) aus.
9. Geben Sie für Ereignismuster Folgendes ein:

```
{
  "source": ["aws.s3"]
}
```

10. Wählen Sie Weiter aus.
11. Wählen Sie für Zieltypen die Option EventBridge API-Ziel aus.
12. Wählen Sie für API-Ziel die Option Vorhandenes API-Ziel verwenden und dann das `SalesforceAD`-Ziel aus, das Sie in Schritt 2 erstellt haben.
13. Wählen Sie für Ausführungsrolle die Option Eine neue Rolle für diese spezifische Ressource erstellen aus.
14. Gehen Sie für Weitere Einstellungen wie folgt vor:
    a. Wählen Sie für Zieleingabe konfigurieren die Option Eingabe-Transformator aus der Dropdown-Liste aus.
    b. Wählen Sie Eingabe-Transformator konfigurieren aus.
    c. Geben Sie für Beispielereignisse Folgendes ein:

```
{
  "detail":[]
}
```

    d. Gehen Sie für Zieleingabe-Transformator wie folgt vor:
        i. Geben Sie für Eingabepfad Folgendes ein:

```
{"detail":"$.detail"}
```

        ii. Geben Sie für Eingabevorlage Folgendes ein:

```
{"message": <detail>}
```

    e. Wählen Sie Bestätigen aus.
15. Wählen Sie Weiter.
16. Wählen Sie Weiter.
17. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 4: Testen der Regel

Um Ihre Regel zu testen, erstellen Sie ein Amazon S3-Objekt, indem Sie eine Datei in einen EventBridge-fähigen Bucket hochladen. Die Informationen über das erstellte Objekt werden an das Salesforce-Plattformereignis gesendet.

Schritt 5: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Durch das Löschen von AWS Ressourcen, die Sie nicht mehr verwenden, vermeiden Sie unnötige Gebühren für Ihr AWS Konto.

So löschen Sie die EventBridge Verbindungen(e)

1. Öffnen Sie die API-Zielseite der - EventBridge Konsole.
2. Wählen Sie die Registerkarte Connections (Verbindungen) aus.
3. Wählen Sie die Verbindung(en) aus, die Sie erstellt haben.
4. Wählen Sie Löschen aus.
5. Geben Sie den Namen der Verbindung ein und wählen Sie Löschen aus.

So löschen Sie das/die EventBridge API-Ziel(e)

1. Öffnen Sie die API-Zielseite der - EventBridge Konsole.
2. Wählen Sie das/die API-Ziel(e) aus, das/die Sie erstellt haben.
3. Wählen Sie Löschen aus.
4. Geben Sie den Namen des API-Ziels ein und wählen Sie Löschen aus.

So löschen Sie die EventBridge Regel(n)

1. Öffnen Sie die Seite Regeln der - EventBridge Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Löschen.

Tutorial: Erstellen einer Verbindung zu Zendesk als API-Ziel

Sie können EventBridge verwenden, um Ereignisse an Drittanbieterservices wie Zendesk weiterzuleiten.

In diesem Tutorial verwenden Sie die EventBridge-Konsole, um eine Verbindung zu Zendesk, ein API-Ziel, das auf Zendesk verweist, und eine Regel zum Weiterleiten von Ereignissen an Zendesk zu erstellen.

Schritte:

Voraussetzungen

Zum Durcharbeiten dieses Tutorials benötigen Sie die folgenden Ressourcen:

- Ein Zendesk-Konto
- Einen EventBridge-fähigen Amazon Simple Storage Service (Amazon S3)-Bucket

Schritt 1: Erstellen einer Verbindung

Zum Senden von Ereignissen an Zendesk müssen Sie zunächst eine Verbindung zur Zendesk-API herstellen.

So erstellen Sie die Verbindung

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich API-Ziele aus.
3. Wählen Sie die Registerkarte Verbindungen und dann Verbindung erstellen aus.

4. Geben Sie einen Namen und eine Beschreibung für die Verbindung ein. Geben Sie beispielsweise **`Zendesk`** für den Namen und **`Connection to Zendesk API`** für die Beschreibung ein.
5. Wählen Sie als Autorisierungstyp die Option Basic (Benutzername/Passwort) aus.
6. Geben Sie für Benutzername Ihren Zendesk-Benutzernamen ein.
7. Geben Sie für Passwort Ihr Zendesk-Passwort ein.
8. Wählen Sie Erstellen aus.

Schritt 2: Erstellen eines API-Ziels

Nachdem Sie die Verbindung erstellt haben, erstellen Sie als Nächstes das API-Ziel, das als Ziel der Regel verwendet werden soll.

So erstellen Sie das API-Ziel

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich API-Ziele aus.
3. Wählen Sie API-Ziel erstellen aus.
4. Geben Sie einen Namen und eine Beschreibung für das API-Ziel ein. Geben Sie beispielsweise **`ZendeskAD`** für den Namen und **`Zendesk API destination`** für die Beschreibung ein.
5. Geben Sie für API-Zielendpunkt **`https://`*`your-subdomain`*`.zendesk.com/api/v2/tickets.json`** ein, wobei *`your-subdomain`* die mit Ihrem Zendesk-Konto verknüpfte Subdomain ist.
6. Wählen Sie in HTTP method POST.
7. Geben Sie für Aufrufratenlimit **`10`** ein.
8. Wählen Sie für Verbindung die Option Vorhandene Verbindung verwenden und die `Zendesk`-Verbindung aus, die Sie in Schritt 1 erstellt haben.
9. Wählen Sie Erstellen aus.

Schritt 3: Erstellen einer Regel

Als Nächstes erstellen Sie eine Regel zum Senden von Ereignissen an Zendesk, wenn ein Amazon-S3-Objekt erstellt wird.

So erstellen Sie eine Regel

1. Öffnen Sie die Amazon EventBridge-Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie im Navigationsbereich Rules aus.
3. Wählen Sie Regel erstellen.
4. Geben Sie einen Namen und eine Beschreibung für die Regel ein. Geben Sie beispielsweise **`ZendeskRule`** für den Namen und **`Rule to send events to Zendesk when S3 objects are created`** für die Beschreibung ein.
5. Bei Event bus (Ereignisbus) wählen Sie default (Standard) aus.
6. Bei Rule type (Regeltyp) wählen Sie Rule with an event pattern (Regel mit einem Ereignismuster) aus.
7. Wählen Sie Next (Weiter).
8. Wählen Sie für Event source (Ereignisquelle) Other (Andere) aus.
9. Geben Sie für Ereignismuster Folgendes ein:

```
{
  "source": ["aws.s3"]
}
```

10. Wählen Sie Next (Weiter).
11. Wählen Sie für Zieltypen die Option EventBridge-API-Ziel aus.
12. Wählen Sie für API-Ziel die Option Vorhandenes API-Ziel verwenden und dann das `ZendeskAD`-Ziel aus, das Sie in Schritt 2 erstellt haben.
13. Wählen Sie für Ausführungsrolle die Option Eine neue Rolle für diese spezifische Ressource erstellen aus.
14. Gehen Sie für Weitere Einstellungen wie folgt vor:

    a. Wählen Sie für Zieleingabe konfigurieren die Option Eingabe-Transformator aus der Dropdown-Liste aus.

    b. Wählen Sie Eingabe-Transformator konfigurieren aus.

    c. Geben Sie für Beispielereignisse Folgendes ein:

```
{
  "detail":[]
}
```

d. Gehen Sie für Zieleingabe-Transformator wie folgt vor:

i. Geben Sie für Eingabepfad Folgendes ein:

```
{"detail":"$.detail"}
```

ii. Geben Sie für Eingabevorlage Folgendes ein:

```
{"message": <detail>}
```

e. Wählen Sie Bestätigen aus.

15. Wählen Sie Next (Weiter).
16. Wählen Sie Next (Weiter).
17. Überprüfen Sie die Details der Regel und wählen Sie dann Create rule (Regel erstellen) aus.

Schritt 4: Testen der Regel

Um Ihre Regel zu testen, erstellen Sie ein Amazon-S3-Objekt, indem Sie eine Datei in einen EventBridge-fähigen Bucket hochladen. Wenn das Ereignis der Regel entspricht, ruft EventBridge die Zendesk-API zur Ticketerstellung auf. Das neue Ticket wird im Zendesk-Dashboard angezeigt.

Schritt 5: Bereinigen Ihrer Ressourcen

Sie können jetzt die Ressourcen, die Sie für dieses Tutorial erstellt haben, löschen, es sei denn, Sie möchten sie behalten. Durch das Löschen von AWS-Ressourcen, die Sie nicht mehr verwenden, können Sie verhindern, dass unnötige Gebühren in Ihrem AWS-Konto anfallen.

So löschen Sie die EventBridge-Verbindung(en)

1. Öffnen Sie die Seite API-Ziel der EventBridge-Konsole.
2. Wählen Sie die Registerkarte Connections (Verbindungen) aus.
3. Wählen Sie die Verbindung(en) aus, die Sie erstellt haben.
4. Wählen Sie Delete (Löschen).
5. Geben Sie den Namen der Verbindung ein und wählen Sie Löschen aus.

So löschen Sie das/die EventBridge-API-Ziel(e)

1. Öffnen Sie die Seite API-Ziel der EventBridge-Konsole.

2. Wählen Sie das/die API-Ziel(e) aus, das/die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Geben Sie den Namen des API-Ziels ein und wählen Sie Löschen aus.

So löschen Sie die EventBridge-Regel(n)

1. Öffnen Sie die Seite Regeln der EventBridge-Konsole.
2. Wählen Sie die Regel(n) aus, die Sie erstellt haben.
3. Wählen Sie Delete (Löschen).
4. Wählen Sie Delete (Löschen).

Verwendung EventBridge mit einem AWS SDK

AWS Software Development Kits (SDKs) sind für viele gängige Programmiersprachen verfügbar. Jedes SDK bietet eine API, Codebeispiele und Dokumentation, die es Entwicklern erleichtern, Anwendungen in ihrer bevorzugten Sprache zu erstellen.

SDK-Dokumentation	Codebeispiele
AWS SDK for C++	AWS SDK for C++ Code-Beispiele
AWS CLI	AWS CLI Codebeispiele
AWS SDK for Go	AWS SDK for Go Codebeispiele
AWS SDK for Java	AWS SDK for Java Codebeispiele
AWS SDK for JavaScript	AWS SDK for JavaScript Codebeispiele
AWS SDK for Kotlin	AWS SDK for Kotlin Codebeispiele
AWS SDK for .NET	AWS SDK for .NET Codebeispiele
AWS SDK for PHP	AWS SDK for PHP Codebeispiele
AWS Tools for PowerShell	Tools für PowerShell Codebeispiele
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) Code-Beispiele
AWS SDK for Ruby	AWS SDK for Ruby Codebeispiele
AWS SDK for Rust	AWS SDK for Rust Codebeispiele
AWS SDK für SAP ABAP	AWS SDK für SAP ABAP Codebeispiele
AWS SDK for Swift	AWS SDK for Swift Codebeispiele

Spezifische Beispiele für finden Sie unterCodebeispiele für die EventBridge Verwendung von AWS SDKs. EventBridge

Beispiel für die Verfügbarkeit

Sie können nicht finden, was Sie brauchen? Fordern Sie ein Codebeispiel an, indem Sie unten den Link Feedback geben auswählen.

Codebeispiele für die EventBridge Verwendung von AWS SDKs

Die folgenden Codebeispiele zeigen, wie die Verwendung EventBridge mit einem AWS Software Development Kit (SDK) funktioniert.

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Servicefunktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarien und serviceübergreifenden Beispiele sehen.

Szenarien sind Codebeispiele, die Ihnen zeigen, wie Sie eine bestimmte Aufgabe ausführen können, indem Sie mehrere Funktionen innerhalb desselben Services aufrufen.

Serviceübergreifende Beispiele sind Beispielanwendungen, die über mehrere AWS-Services hinweg arbeiten.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erste Schritte

Hallo EventBridge

Die folgenden Codebeispiele zeigen, wie Sie mit der Verwendung beginnen EventBridge.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
using Amazon.EventBridge;
using Amazon.EventBridge.Model;
```

```
namespace EventBridgeActions;

public static class HelloEventBridge
{
    static async Task Main(string[] args)
    {
        var eventBridgeClient = new AmazonEventBridgeClient();

        Console.WriteLine($"Hello Amazon EventBridge! Following are some of your
 EventBuses:");
        Console.WriteLine();

        // You can use await and any of the async methods to get a response.
        // Let's get the first five event buses.
        var response = await eventBridgeClient.ListEventBusesAsync(
            new ListEventBusesRequest()
            {
                Limit = 5
            });

        foreach (var eventBus in response.EventBuses)
        {
            Console.WriteLine($"\tEventBus: {eventBus.Name}");
            Console.WriteLine($"\tArn: {eventBus.Arn}");
            Console.WriteLine($"\tPolicy: {eventBus.Policy}");
            Console.WriteLine();
        }
    }
}
```

- Einzelheiten zur API finden Sie ListEventBusesin der AWS SDK for .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 */
public class HelloEventBridge {
    public static void main(String[] args) {
        Region region = Region.US_WEST_2;
        EventBridgeClient eventBrClient = EventBridgeClient.builder()
                .region(region)
                .build();

        listBuses(eventBrClient);
        eventBrClient.close();
    }

    public static void listBuses(EventBridgeClient eventBrClient) {
        try {
            ListEventBusesRequest busesRequest = ListEventBusesRequest.builder()
                    .limit(10)
                    .build();

            ListEventBusesResponse response =
 eventBrClient.listEventBuses(busesRequest);
            List<EventBus> buses = response.eventBuses();
            for (EventBus bus : buses) {
                System.out.println("The name of the event bus is: " +
 bus.name());
                System.out.println("The ARN of the event bus is: " + bus.arn());
            }

        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie ListEventBusesin der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
import aws.sdk.kotlin.services.eventbridge.EventBridgeClient
import aws.sdk.kotlin.services.eventbridge.model.ListEventBusesRequest
import aws.sdk.kotlin.services.eventbridge.model.ListEventBusesResponse

suspend fun main() {
    listBusesHello()
}

suspend fun listBusesHello() {
    val request = ListEventBusesRequest {
        limit = 10
    }

    EventBridgeClient { region = "us-west-2" }.use { eventBrClient ->
        val response: ListEventBusesResponse =
 eventBrClient.listEventBuses(request)
        response.eventBuses?.forEach { bus ->
            println("The name of the event bus is ${bus.name}")
            println("The ARN of the event bus is ${bus.arn}")
        }
    }
}
```

- API-Details finden Sie ListEventBusesin der API-Referenz zum AWS SDK für Kotlin.

Codebeispiele

- Aktionen zur EventBridge Verwendung von SDKs AWS

Aktionen zur EventBridge Verwendung von SDKs AWS

Die folgenden Codebeispiele zeigen, wie einzelne EventBridge Aktionen mit AWS SDKs ausgeführt werden. Diese Auszüge rufen die EventBridge API auf und sind Codeauszüge aus größeren Programmen, die im Kontext ausgeführt werden müssen. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Eine vollständige Liste finden Sie in der Amazon EventBridge API-Referenz.

Beispiele

- Verwendung ListRules mit einem AWS SDK oder CLI
- Verwendung ListTargetsByRule mit einem AWS SDK oder CLI
- Verwendung PutEvents mit einem AWS SDK oder CLI
- Verwendung PutRule mit einem AWS SDK oder CLI
- Verwendung PutTargets mit einem AWS SDK oder CLI
- Verwendung RemoveTargets mit einem AWS SDK oder CLI

Verwendung **`DeleteRule`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DeleteRule`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit Regeln und Zielen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Löschen Sie eine Regel anhand ihres Namens.

```
/// <summary>
/// Delete an event rule by name.
/// </summary>
/// <param name="ruleName">The name of the event rule.</param>
/// <returns>True if successful.</returns>
public async Task<bool> DeleteRuleByName(string ruleName)
{
    var response = await _amazonEventBridge.DeleteRuleAsync(
        new DeleteRuleRequest()
        {
```

```
                Name = ruleName
            });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie DeleteRulein der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

Um eine CloudWatch Ereignisregel zu löschen

In diesem Beispiel wird die Regel mit dem Namen InstanceStateChanges EC2 gelöscht:

```
aws events delete-rule --name "EC2InstanceStateChanges"
```

- Einzelheiten zur API finden Sie unter DeleteRule AWS CLIBefehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   public static void deleteRuleByName(EventBridgeClient eventBrClient, String
ruleName) {
        DeleteRuleRequest ruleRequest = DeleteRuleRequest.builder()
                .name(ruleName)
                .build();

        eventBrClient.deleteRule(ruleRequest);
        System.out.println("Successfully deleted the rule");
    }
```

- Einzelheiten zur API finden Sie DeleteRulein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun deleteRuleByName(ruleName: String?) {
    val ruleRequest = DeleteRuleRequest {
        name = ruleName
    }
    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.deleteRule(ruleRequest)
        println("Successfully deleted the rule")
    }
}
```

- API-Details finden Sie DeleteRulein der API-Referenz zum AWS SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DescribeRule`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DescribeRule`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit Regeln und Zielen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie den Status einer Regel anhand der Regelbeschreibung ab.

```
    /// <summary>
    /// Get the state for a rule by the rule name.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <param name="eventBusName">The optional name of the event bus. If empty,
uses the default event bus.</param>
    /// <returns>The state of the rule.</returns>
    public async Task<RuleState> GetRuleStateByRuleName(string ruleName, string?
eventBusName = null)
    {
        var ruleResponse = await _amazonEventBridge.DescribeRuleAsync(
            new DescribeRuleRequest()
            {
                Name = ruleName,
                EventBusName = eventBusName
            });
        return ruleResponse.State;
    }
```

- Einzelheiten zur API finden Sie DescribeRulein der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So zeigen Sie Informationen zu einer CloudWatch Ereignisregel an

In diesem Beispiel werden Informationen zu der Regel mit dem Namen angezeigt DailyLambdaFunction:

```
aws events describe-rule --name "DailyLambdaFunction"
```

- Einzelheiten zur API finden Sie DescribeRuleunter AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   public static void checkRule(EventBridgeClient eventBrClient, String
eventRuleName) {
       try {
           DescribeRuleRequest ruleRequest = DescribeRuleRequest.builder()
                   .name(eventRuleName)
                   .build();

           DescribeRuleResponse response =
eventBrClient.describeRule(ruleRequest);
           System.out.println("The state of the rule is " +
response.stateAsString());

       } catch (EventBridgeException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Einzelheiten zur API finden Sie DescribeRulein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun checkRule(eventRuleName: String?) {
    val ruleRequest = DescribeRuleRequest {
        name = eventRuleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.describeRule(ruleRequest)
        println("The state of the rule is $response")
    }
}
```

- API-Details finden Sie DescribeRulein der API-Referenz zum AWS SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DisableRule`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`DisableRule`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit Regeln und Zielen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Deaktivieren Sie eine Regel anhand ihres Regelnamens.

```
/// <summary>
/// Disable a particular rule on an event bus.
/// </summary
/// <param name="ruleName">The name of the rule.</param>
/// <returns>True if successful.</returns>
public async Task<bool> DisableRuleByName(string ruleName)
{
    var ruleResponse = await _amazonEventBridge.DisableRuleAsync(
        new DisableRuleRequest()
        {
            Name = ruleName
        });
    return ruleResponse.HttpStatusCode == HttpStatusCode.OK;
}
```

- Einzelheiten zur API finden Sie DisableRulein der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

Um eine CloudWatch Ereignisregel zu deaktivieren

In diesem Beispiel wird die genannte DailyLambdaFunction Regel deaktiviert. Die Regel wird nicht gelöscht:

```
aws events disable-rule --name "DailyLambdaFunction"
```

- Einzelheiten zur API finden Sie DisableRulein der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Deaktivieren Sie eine Regel anhand ihres Regelnamens.

```
   public static void changeRuleState(EventBridgeClient eventBrClient, String
eventRuleName, Boolean isEnabled) {
       try {
           if (!isEnabled) {
               System.out.println("Disabling the rule: " + eventRuleName);
               DisableRuleRequest ruleRequest = DisableRuleRequest.builder()
                       .name(eventRuleName)
                       .build();

               eventBrClient.disableRule(ruleRequest);
           } else {
               System.out.println("Enabling the rule: " + eventRuleName);
               EnableRuleRequest ruleRequest = EnableRuleRequest.builder()
                       .name(eventRuleName)
                       .build();
               eventBrClient.enableRule(ruleRequest);
           }

       } catch (EventBridgeException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Einzelheiten zur API finden Sie DisableRulein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun changeRuleState(eventRuleName: String, isEnabled: Boolean?) {
    if (!isEnabled!!) {
        println("Disabling the rule: $eventRuleName")
        val ruleRequest = DisableRuleRequest {
            name = eventRuleName
        }
        EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
            eventBrClient.disableRule(ruleRequest)
        }
    } else {
        println("Enabling the rule: $eventRuleName")
        val ruleRequest = EnableRuleRequest {
            name = eventRuleName
        }
        EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
            eventBrClient.enableRule(ruleRequest)
        }
    }
}
```

- API-Details finden Sie DisableRulein der API-Referenz zum AWS SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`EnableRule`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`EnableRule`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit Regeln und Zielen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Aktivieren Sie eine Regel anhand ihres Regelnamens.

```
/// <summary>
/// Enable a particular rule on an event bus.
/// </summary>
/// <param name="ruleName">The name of the rule.</param>
/// <returns>True if successful.</returns>
public async Task<bool> EnableRuleByName(string ruleName)
{
    var ruleResponse = await _amazonEventBridge.EnableRuleAsync(
        new EnableRuleRequest()
        {
            Name = ruleName
        });
    return ruleResponse.HttpStatusCode == HttpStatusCode.OK;
}
```

- Einzelheiten zur API finden Sie EnableRulein der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

Um eine CloudWatch Ereignisregel zu aktivieren

In diesem Beispiel wird die genannte Regel aktiviert DailyLambdaFunction, die zuvor deaktiviert wurde:

```
aws events enable-rule --name "DailyLambdaFunction"
```

- Einzelheiten zur API finden Sie EnableRulein der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Aktivieren Sie eine Regel anhand ihres Regelnamens.

```
   public static void changeRuleState(EventBridgeClient eventBrClient, String
eventRuleName, Boolean isEnabled) {
       try {
           if (!isEnabled) {
               System.out.println("Disabling the rule: " + eventRuleName);
               DisableRuleRequest ruleRequest = DisableRuleRequest.builder()
                       .name(eventRuleName)
                       .build();

               eventBrClient.disableRule(ruleRequest);
           } else {
               System.out.println("Enabling the rule: " + eventRuleName);
               EnableRuleRequest ruleRequest = EnableRuleRequest.builder()
                       .name(eventRuleName)
                       .build();
               eventBrClient.enableRule(ruleRequest);
           }

       } catch (EventBridgeException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
```

```
    }
```

- Einzelheiten zur API finden Sie EnableRulein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun changeRuleState(eventRuleName: String, isEnabled: Boolean?) {
    if (!isEnabled!!) {
        println("Disabling the rule: $eventRuleName")
        val ruleRequest = DisableRuleRequest {
            name = eventRuleName
        }
        EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
            eventBrClient.disableRule(ruleRequest)
        }
    } else {
        println("Enabling the rule: $eventRuleName")
        val ruleRequest = EnableRuleRequest {
            name = eventRuleName
        }
        EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
            eventBrClient.enableRule(ruleRequest)
        }
    }
}
```

- API-Details finden Sie EnableRulein der API-Referenz zum AWS SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListRuleNamesByTarget`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListRuleNamesByTarget`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit Regeln und Zielen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie alle Regelnamen mithilfe des Ziels auf.

```
/// <summary>
/// List names of all rules matching a target.
/// </summary>
/// <param name="targetArn">The ARN of the target.</param>
/// <returns>The list of rule names.</returns>
public async Task<List<string>> ListAllRuleNamesByTarget(string targetArn)
{
    var results = new List<string>();
    var request = new ListRuleNamesByTargetRequest()
    {
        TargetArn = targetArn
    };
    ListRuleNamesByTargetResponse response;
    do
    {
```

```
            response = await
_amazonEventBridge.ListRuleNamesByTargetAsync(request);
            results.AddRange(response.RuleNames);
            request.NextToken = response.NextToken;

        } while (response.NextToken is not null);

        return results;
    }
```

- Einzelheiten zur API finden Sie ListRuleNamesByTargetin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So zeigen Sie alle Regeln mit einem bestimmten Ziel an

In diesem Beispiel werden alle Regeln angezeigt, deren Ziel die Lambda-Funktion MyFunctionName "" ist:

```
aws events list-rule-names-by-target --target-arn "arn:aws:lambda:us-
east-1:123456789012:function:MyFunctionName"
```

- Einzelheiten zur API finden Sie unter ListRuleNamesByTarget AWS CLIBefehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie alle Regelnamen mithilfe des Ziels auf.

```
   public static void listTargetRules(EventBridgeClient eventBrClient, String
topicArn) {
       ListRuleNamesByTargetRequest ruleNamesByTargetRequest =
ListRuleNamesByTargetRequest.builder()
               .targetArn(topicArn)
               .build();

       ListRuleNamesByTargetResponse response =
eventBrClient.listRuleNamesByTarget(ruleNamesByTargetRequest);
       List<String> rules = response.ruleNames();
       for (String rule : rules) {
           System.out.println("The rule name is " + rule);
       }
   }
```

- Einzelheiten zur API finden Sie ListRuleNamesByTargetin der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun listTargetRules(topicArnVal: String?) {
    val ruleNamesByTargetRequest = ListRuleNamesByTargetRequest {
        targetArn = topicArnVal
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response =
eventBrClient.listRuleNamesByTarget(ruleNamesByTargetRequest)
        response.ruleNames?.forEach { rule ->
            println("The rule name is $rule")
        }
    }
```

```
}
```

- API-Details finden Sie ListRuleNamesByTargetin der API-Referenz zum AWS SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListRules`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListRules`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit Regeln und Zielen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie alle Regeln für einen Event Bus auf.

```
    /// <summary>
    /// List the rules on an event bus.
    /// </summary>
    /// <param name="eventBusArn">The optional ARN of the event bus. If empty,
uses the default event bus.</param>
    /// <returns>The list of rules.</returns>
    public async Task<List<Rule>> ListAllRulesForEventBus(string? eventBusArn =
null)
```

```
    {
        var results = new List<Rule>();
        var request = new ListRulesRequest()
        {
            EventBusName = eventBusArn
        };
        // Get all of the pages of rules.
        ListRulesResponse response;
        do
        {
            response = await _amazonEventBridge.ListRulesAsync(request);
            results.AddRange(response.Rules);
            request.NextToken = response.NextToken;

        } while (response.NextToken is not null);

        return results;
    }
```

- Einzelheiten zur API finden Sie ListRulesin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

Um eine Liste aller CloudWatch Event-Regeln anzuzeigen

In diesem Beispiel werden alle CloudWatch Event-Regeln in der Region angezeigt:

```
aws events list-rules
```

Um eine Liste von CloudWatch Event-Regeln anzuzeigen, die mit einer bestimmten Zeichenfolge beginnen.

In diesem Beispiel werden alle CloudWatch Event-Regeln in der Region angezeigt, deren Name mit „Täglich“ beginnt:

```
aws events list-rules --name-prefix "Daily"
```

- Einzelheiten zur API finden Sie ListRulesunter AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Aktivieren Sie eine Regel anhand ihres Regelnamens.

```
    public static void listRules(EventBridgeClient eventBrClient) {
        try {
            ListRulesRequest rulesRequest = ListRulesRequest.builder()
                    .eventBusName("default")
                    .limit(10)
                    .build();

            ListRulesResponse response = eventBrClient.listRules(rulesRequest);
            List<Rule> rules = response.rules();
            for (Rule rule : rules) {
                System.out.println("The rule name is : " + rule.name());
                System.out.println("The rule description is : " +
rule.description());
                System.out.println("The rule state is : " +
rule.stateAsString());
            }

        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Einzelheiten zur API finden Sie ListRulesin der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun listRules() {
    val rulesRequest = ListRulesRequest {
        eventBusName = "default"
        limit = 10
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.listRules(rulesRequest)
        response.rules?.forEach { rule ->
            println("The rule name is ${rule.name}")
            println("The rule ARN is ${rule.arn}")
        }
    }
}
```

- API-Details finden Sie ListRulesin der API-Referenz zum AWS SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListTargetsByRule`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`ListTargetsByRule`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit Regeln und Zielen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie alle Ziele für eine Regel mithilfe des Regelnamens auf.

```
/// <summary>
/// List all of the targets matching a rule by name.
/// </summary>
/// <param name="ruleName">The name of the rule.</param>
/// <returns>The list of targets.</returns>
public async Task<List<Target>> ListAllTargetsOnRule(string ruleName)
{
    var results = new List<Target>();
    var request = new ListTargetsByRuleRequest()
    {
        Rule = ruleName
    };
    ListTargetsByRuleResponse response;
    do
    {
        response = await _amazonEventBridge.ListTargetsByRuleAsync(request);
        results.AddRange(response.Targets);
        request.NextToken = response.NextToken;

    } while (response.NextToken is not null);

    return results;
}
```

- Einzelheiten zur API finden Sie ListTargetsByRulein der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

Um alle Ziele für eine CloudWatch Ereignisregel anzuzeigen

In diesem Beispiel werden alle Ziele der Regel mit dem Namen angezeigt DailyLambdaFunction:

```
aws events list-targets-by-rule --rule  "DailyLambdaFunction"
```

- Einzelheiten zur API finden Sie ListTargetsByRuleunter AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Listen Sie alle Ziele für eine Regel mithilfe des Regelnamens auf.

```
   public static void listTargets(EventBridgeClient eventBrClient, String
ruleName) {
       ListTargetsByRuleRequest ruleRequest = ListTargetsByRuleRequest.builder()
               .rule(ruleName)
               .build();

       ListTargetsByRuleResponse res =
eventBrClient.listTargetsByRule(ruleRequest);
       List<Target> targetsList = res.targets();
       for (Target target: targetsList) {
           System.out.println("Target ARN: "+target.arn());
       }
   }
```

- Einzelheiten zur API finden Sie ListTargetsByRulein der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun listTargets(ruleName: String?) {
    val ruleRequest = ListTargetsByRuleRequest {
        rule = ruleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.listTargetsByRule(ruleRequest)
        response.targets?.forEach { target ->
            println("Target ARN: ${target.arn}")
        }
    }
}
```

- API-Details finden Sie ListTargetsByRulein der API-Referenz zum AWS SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`PutEvents`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`PutEvents`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen und Auslösen einer Regel
- Erste Schritte mit Regeln und Zielen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Senden Sie ein Ereignis, das einem benutzerdefinierten Muster für eine Regel entspricht.

```
    /// <summary>
    /// Add an event to the event bus that includes an email, message, and time.
    /// </summary>
    /// <param name="email">The email to use in the event detail of the custom
event.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> PutCustomEmailEvent(string email)
    {
        var eventDetail = new
        {
            UserEmail = email,
            Message = "This event was generated by example code.",
            UtcTime = DateTime.UtcNow.ToString("g")
        };
        var response = await _amazonEventBridge.PutEventsAsync(
            new PutEventsRequest()
            {
                Entries = new List<PutEventsRequestEntry>()
                {
                    new PutEventsRequestEntry()
                    {
                        Source = "ExampleSource",
                        Detail = JsonSerializer.Serialize(eventDetail),
                        DetailType = "ExampleType"
                    }
                }
```

```
            });

        return response.FailedEntryCount == 0;
    }
```

- Einzelheiten zur API finden Sie PutEventsin der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Binden Sie die erforderlichen Dateien ein.

```
#include <aws/core/Aws.h>
#include <aws/events/EventBridgeClient.h>
#include <aws/events/model/PutEventsRequest.h>
#include <aws/events/model/PutEventsResult.h>
#include <aws/core/utils/Outcome.h>
#include <iostream>
```

Senden Sie das Ereignis.

```
        Aws::CloudWatchEvents::EventBridgeClient cwe;

        Aws::CloudWatchEvents::Model::PutEventsRequestEntry event_entry;
        event_entry.SetDetail(MakeDetails(event_key, event_value));
        event_entry.SetDetailType("sampleSubmitted");
        event_entry.AddResources(resource_arn);
        event_entry.SetSource("aws-sdk-cpp-cloudwatch-example");

        Aws::CloudWatchEvents::Model::PutEventsRequest request;
        request.AddEntries(event_entry);
```

```
        auto outcome = cwe.PutEvents(request);
        if (!outcome.IsSuccess())
        {
            std::cout << "Failed to post CloudWatch event: " <<
                outcome.GetError().GetMessage() << std::endl;
        }
        else
        {
            std::cout << "Successfully posted CloudWatch event" << std::endl;
        }
```

- Einzelheiten zur API finden Sie PutEventsin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Um ein benutzerdefiniertes Ereignis an CloudWatch Events zu senden

In diesem Beispiel wird ein benutzerdefiniertes Ereignis an CloudWatch Events gesendet. Das Ereignis ist in der Datei putevents.json enthalten:

```
aws events put-events --entries file://putevents.json
```

Die Datei putevents.json hat folgenden Inhalt:

```
[
  {
    "Source": "com.mycompany.myapp",
    "Detail": "{ \"key1\": \"value1\", \"key2\": \"value2\" }",
    "Resources": [
      "resource1",
      "resource2"
    ],
    "DetailType": "myDetailType"
  },
  {
    "Source": "com.mycompany.myapp",
    "Detail": "{ \"key1\": \"value3\", \"key2\": \"value4\" }",
    "Resources": [
      "resource1",
```

```
      "resource2"
    ],
    "DetailType": "myDetailType"
   }
]
```

- Einzelheiten zur API finden Sie PutEventsin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
   public static void triggerCustomRule(EventBridgeClient eventBrClient, String
email) {
       String json = "{" +
               "\"UserEmail\": \"" + email + "\"," +
               "\"Message\": \"This event was generated by example code.\"," +
               "\"UtcTime\": \"Now.\"" +
               "}";

       PutEventsRequestEntry entry = PutEventsRequestEntry.builder()
               .source("ExampleSource")
               .detail(json)
               .detailType("ExampleType")
               .build();

       PutEventsRequest eventsRequest = PutEventsRequest.builder()
               .entries(entry)
               .build();

       eventBrClient.putEvents(eventsRequest);
   }
```

- Einzelheiten zur API finden Sie PutEventsin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Importieren Sie das SDK- und Client-Module und rufen Sie die API auf.

```
import {
  EventBridgeClient,
  PutEventsCommand,
} from "@aws-sdk/client-eventbridge";

export const putEvents = async (
  source = "eventbridge.integration.test",
  detailType = "greeting",
  resources = [],
) => {
  const client = new EventBridgeClient({});

  const response = await client.send(
    new PutEventsCommand({
      Entries: [
        {
          Detail: JSON.stringify({ greeting: "Hello there." }),
          DetailType: detailType,
          Resources: resources,
          Source: source,
        },
      ],
    }),
  );

  console.log("PutEvents response:");
  console.log(response);
  // PutEvents response:
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
```

```
  //     requestId: '3d0df73d-dcea-4a23-ae0d-f5556a3ac109',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   Entries: [ { EventId: '51620841-5af4-6402-d9bc-b77734991eb5' } ],
  //   FailedEntryCount: 0
  // }

  return response;
};
```

- Einzelheiten zur API finden Sie PutEventsin der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create CloudWatchEvents service object
var ebevents = new AWS.EventBridge({ apiVersion: "2015-10-07" });

var params = {
  Entries: [
    {
      Detail: '{ "key1": "value1", "key2": "value2" }',
      DetailType: "appRequestSubmitted",
      Resources: ["RESOURCE_ARN"],
      Source: "com.company.app",
    },
  ],
};
```

```
ebevents.putEvents(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.Entries);
  }
});
```

- Einzelheiten zur API finden Sie PutEventsin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun triggerCustomRule(email: String) {
    val json = "{" +
        "\"UserEmail\": \"" + email + "\"," +
        "\"Message\": \"This event was generated by example code.\"" +
        "\"UtcTime\": \"Now.\"" +
        "}"

    val entry = PutEventsRequestEntry {
        source = "ExampleSource"
        detail = json
        detailType = "ExampleType"
    }

    val eventsRequest = PutEventsRequest {
        this.entries = listOf(entry)
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putEvents(eventsRequest)
    }
}
```

- API-Details finden Sie PutEventsin der API-Referenz zum AWS SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`PutRule`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`PutRule`.

Aktionsbeispiele sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Sie können diese Aktion in den folgenden Codebeispielen im Kontext sehen:

- Erstellen und Auslösen einer Regel
- Erste Schritte mit Regeln und Zielen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie eine Regel, die ausgelöst wird, wenn ein Objekt zu einem Amazon-Simple-Storage-Service-Bucket hinzugefügt wird.

```
    /// <summary>
    /// Create a new event rule that triggers when an Amazon S3 object is created
 in a bucket.
    /// </summary>
    /// <param name="roleArn">The ARN of the role.</param>
    /// <param name="ruleName">The name to give the rule.</param>
    /// <param name="bucketName">The name of the bucket to trigger the event.</
param>
```

```
    /// <returns>The ARN of the new rule.</returns>
    public async Task<string> PutS3UploadRule(string roleArn, string ruleName,
string bucketName)
    {
        string eventPattern = "{" +
                               "\"source\": [\"aws.s3\"]," +
                                   "\"detail-type\": [\"Object Created\"]," +
                                   "\"detail\": {" +
                                       "\"bucket\": {" +
                                           "\"name\": [\"" + bucketName + "\"]"
+
                                       "}" +
                                   "}" +
                               "}";

        var response = await _amazonEventBridge.PutRuleAsync(
            new PutRuleRequest()
            {
                Name = ruleName,
                Description = "Example S3 upload rule for EventBridge",
                RoleArn = roleArn,
                EventPattern = eventPattern
            });

        return response.RuleArn;
    }
```

Erstellen Sie eine Regel, die ein benutzerdefiniertes Muster verwendet.

```
    /// <summary>
    /// Update a rule to use a custom defined event pattern.
    /// </summary>
    /// <param name="ruleName">The name of the rule to update.</param>
    /// <returns>The ARN of the updated rule.</returns>
    public async Task<string> UpdateCustomEventPattern(string ruleName)
    {
        string customEventsPattern = "{" +
                                     "\"source\": [\"ExampleSource\"]," +
                                     "\"detail-type\": [\"ExampleType\"]" +
                                     "}";

        var response = await _amazonEventBridge.PutRuleAsync(
```

```
            new PutRuleRequest()
            {
                Name = ruleName,
                Description = "Custom test rule",
                EventPattern = customEventsPattern
            });

        return response.RuleArn;
    }
```

- Einzelheiten zur API finden Sie PutRulein der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Binden Sie die erforderlichen Dateien ein.

```
#include <aws/core/Aws.h>
#include <aws/events/EventBridgeClient.h>
#include <aws/events/model/PutRuleRequest.h>
#include <aws/events/model/PutRuleResult.h>
#include <aws/core/utils/Outcome.h>
#include <iostream>
```

Erstellen Sie die -Regel.

```
        Aws::CloudWatchEvents::EventBridgeClient cwe;
        Aws::CloudWatchEvents::Model::PutRuleRequest request;
        request.SetName(rule_name);
        request.SetRoleArn(role_arn);
        request.SetScheduleExpression("rate(5 minutes)");
        request.SetState(Aws::CloudWatchEvents::Model::RuleState::ENABLED);
```

```
        auto outcome = cwe.PutRule(request);
        if (!outcome.IsSuccess())
        {
            std::cout << "Failed to create CloudWatch events rule " <<
                rule_name << ": " << outcome.GetError().GetMessage() <<
                std::endl;
        }
        else
        {
            std::cout << "Successfully created CloudWatch events rule " <<
                rule_name << " with resulting Arn " <<
                outcome.GetResult().GetRuleArn() << std::endl;
        }
```

- Einzelheiten zur API finden Sie PutRulein der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

Um Regeln für CloudWatch Ereignisse zu erstellen

Im folgenden Beispiel wird eine Regel erstellt, die jeden Tag um 09:00 Uhr (UTC) ausgelöst wird. Wenn Sie put-targets verwenden, um eine Lambda-Funktion als Ziel dieser Regel hinzuzufügen, können Sie die Lambda-Funktion jeden Tag zur angegebenen Zeit ausführen:

```
aws events put-rule --name "DailyLambdaFunction" --schedule-expression "cron(0 9
 * * ? *)"
```

Im folgenden Beispiel wird eine Regel erstellt, die ausgelöst wird, wenn eine EC2-Instance in der Region den Status ändert:

```
aws events put-rule --name "EC2InstanceStateChanges" --event-pattern "{\"source
\":[\"aws.ec2\"],\"detail-type\":[\"EC2 Instance State-change Notification\"]}"
 --role-arn "arn:aws:iam::123456789012:role/MyRoleForThisRule"
```

Im folgenden Beispiel wird eine Regel erstellt, die ausgelöst wird, wenn eine EC2-Instance in der Region gestoppt oder beendet wird:

```
aws events put-rule --name "EC2InstanceStateChangeStopOrTerminate" --event-
pattern "{\"source\":[\"aws.ec2\"],\"detail-type\":[\"EC2 Instance State-change
 Notification\"],\"detail\":{\"state\":[\"stopped\",\"terminated\"]}}" --role-arn
 "arn:aws:iam::123456789012:role/MyRoleForThisRule"
```

- Einzelheiten zur API finden Sie PutRulein der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie eine geplante Regel.

```
   public static void createEBRule(EventBridgeClient eventBrClient, String
ruleName, String cronExpression) {
       try {
           PutRuleRequest ruleRequest = PutRuleRequest.builder()
                   .name(ruleName)
                   .eventBusName("default")
                   .scheduleExpression(cronExpression)
                   .state("ENABLED")
                   .description("A test rule that runs on a schedule created by
the Java API")
                   .build();

           PutRuleResponse ruleResponse = eventBrClient.putRule(ruleRequest);
           System.out.println("The ARN of the new rule is " +
ruleResponse.ruleArn());

       } catch (EventBridgeException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

Erstellen Sie eine Regel, die ausgelöst wird, wenn ein Objekt zu einem Amazon-Simple-Storage-Service-Bucket hinzugefügt wird.

```
   // Create a new event rule that triggers when an Amazon S3 object is created
in
   // a bucket.
   public static void addEventRule(EventBridgeClient eventBrClient, String
roleArn, String bucketName,
           String eventRuleName) {
       String pattern = "{\n" +
               "  \"source\": [\"aws.s3\"],\n" +
               "  \"detail-type\": [\"Object Created\"],\n" +
               "  \"detail\": {\n" +
               "    \"bucket\": {\n" +
               "      \"name\": [\"" + bucketName + "\"]\n" +
               "    }\n" +
               "  }\n" +
               "}";

       try {
           PutRuleRequest ruleRequest = PutRuleRequest.builder()
                   .description("Created by using the AWS SDK for Java v2")
                   .name(eventRuleName)
                   .eventPattern(pattern)
                   .roleArn(roleArn)
                   .build();

           PutRuleResponse ruleResponse = eventBrClient.putRule(ruleRequest);
           System.out.println("The ARN of the new rule is " +
ruleResponse.ruleArn());

       } catch (EventBridgeException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Einzelheiten zur API finden Sie PutRulein der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Importieren Sie das SDK- und Client-Module und rufen Sie die API auf.

```
import { EventBridgeClient, PutRuleCommand } from "@aws-sdk/client-eventbridge";

export const putRule = async (
  ruleName = "some-rule",
  source = "some-source",
) => {
  const client = new EventBridgeClient({});

  const response = await client.send(
    new PutRuleCommand({
      Name: ruleName,
      EventPattern: JSON.stringify({ source: [source] }),
      State: "ENABLED",
      EventBusName: "default",
    }),
  );

  console.log("PutRule response:");
  console.log(response);
  // PutRule response:
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: 'd7292ced-1544-421b-842f-596326bc7072',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   RuleArn: 'arn:aws:events:us-east-1:xxxxxxxxxxxx:rule/
EventBridgeTestRule-1696280037720'
```

```
  // }
  return response;
};
```

- Einzelheiten zur API finden Sie PutRulein der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create CloudWatchEvents service object
var ebevents = new AWS.EventBridge({ apiVersion: "2015-10-07" });

var params = {
  Name: "DEMO_EVENT",
  RoleArn: "IAM_ROLE_ARN",
  ScheduleExpression: "rate(5 minutes)",
  State: "ENABLED",
};

ebevents.putRule(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.RuleArn);
  }
});
```

- Einzelheiten zur API finden Sie PutRulein der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Erstellen Sie eine geplante Regel.

```
suspend fun createScRule(ruleName: String?, cronExpression: String?) {
    val ruleRequest = PutRuleRequest {
        name = ruleName
        eventBusName = "default"
        scheduleExpression = cronExpression
        state = RuleState.Enabled
        description = "A test rule that runs on a schedule created by the Kotlin
 API"
    }

    EventBridgeClient { region = "us-west-2" }.use { eventBrClient ->
        val ruleResponse = eventBrClient.putRule(ruleRequest)
        println("The ARN of the new rule is ${ruleResponse.ruleArn}")
    }
}
```

Erstellen Sie eine Regel, die ausgelöst wird, wenn ein Objekt zu einem Amazon-Simple-Storage-Service-Bucket hinzugefügt wird.

```
// Create a new event rule that triggers when an Amazon S3 object is created in a
 bucket.
suspend fun addEventRule(roleArnVal: String?, bucketName: String, eventRuleName:
 String?) {
    val pattern = """{
        "source": ["aws.s3"],
        "detail-type": ["Object Created"],
        "detail": {
        "bucket": {
            "name": ["$bucketName"]
```

```
            }
        }
    }"""

    val ruleRequest = PutRuleRequest {
        description = "Created by using the AWS SDK for Kotlin"
        name = eventRuleName
        eventPattern = pattern
        roleArn = roleArnVal
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val ruleResponse = eventBrClient.putRule(ruleRequest)
        println("The ARN of the new rule is ${ruleResponse.ruleArn}")
    }
}
```

- API-Details finden Sie PutRulein der API-Referenz zum AWS SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`PutTargets`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`PutTargets`.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit Regeln und Zielen

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Fügen Sie ein Amazon-SNS-Thema als Ziel für eine Regel hinzu.

```
    /// <summary>
    /// Add an Amazon SNS target topic to a rule.
    /// </summary>
    /// <param name="ruleName">The name of the rule to update.</param>
    /// <param name="targetArn">The ARN of the Amazon SNS target.</param>
    /// <param name="eventBusArn">The optional event bus name, uses default if
empty.</param>
    /// <returns>The ID of the target.</returns>
    public async Task<string> AddSnsTargetToRule(string ruleName, string
targetArn, string? eventBusArn = null)
    {
        var targetID = Guid.NewGuid().ToString();

        // Create the list of targets and add a new target.
        var targets = new List<Target>
        {
            new Target()
            {
                Arn = targetArn,
                Id = targetID
            }
        };

        // Add the targets to the rule.
        var response = await _amazonEventBridge.PutTargetsAsync(
            new PutTargetsRequest()
            {
                EventBusName = eventBusArn,
                Rule = ruleName,
                Targets = targets,
            });
```

```
        if (response.FailedEntryCount > 0)
        {
            response.FailedEntries.ForEach(e =>
            {
                _logger.LogError(
                    $"Failed to add target {e.TargetId}: {e.ErrorMessage}, code
{e.ErrorCode}");
            });
        }

        return targetID;
    }
```

Fügen Sie einen Eingabe-Transformator als Ziel für eine Regel hinzu.

```
    /// <summary>
    /// Update an Amazon S3 object created rule with a transform on the target.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <param name="targetArn">The ARN of the target.</param>
    /// <param name="eventBusArn">Optional event bus ARN. If empty, uses the
default event bus.</param>
    /// <returns>The ID of the target.</returns>
    public async Task<string> UpdateS3UploadRuleTargetWithTransform(string
ruleName, string targetArn, string? eventBusArn = null)
    {
        var targetID = Guid.NewGuid().ToString();

        var targets = new List<Target>
        {
            new Target()
            {
                Id = targetID,
                Arn = targetArn,
                InputTransformer = new InputTransformer()
                {
                    InputPathsMap = new Dictionary<string, string>()
                    {
                        {"bucket", "$.detail.bucket.name"},
                        {"time", "$.time"}
                    },
```

```
                    InputTemplate = "\"Notification: an object was uploaded to
bucket <bucket> at <time>.\""
                }
            }
        };
        var response = await _amazonEventBridge.PutTargetsAsync(
            new PutTargetsRequest()
            {
                EventBusName = eventBusArn,
                Rule = ruleName,
                Targets = targets,
            });
        if (response.FailedEntryCount > 0)
        {
            response.FailedEntries.ForEach(e =>
            {
                _logger.LogError(
                    $"Failed to add target {e.TargetId}: {e.ErrorMessage}, code
{e.ErrorCode}");
            });
        }
        return targetID;
    }
```

- Einzelheiten zur API finden Sie PutTargetsin der AWS SDK for .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Binden Sie die erforderlichen Dateien ein.

```
#include <aws/core/Aws.h>
#include <aws/events/EventBridgeClient.h>
#include <aws/events/model/PutTargetsRequest.h>
```

```
#include <aws/events/model/PutTargetsResult.h>
#include <aws/core/utils/Outcome.h>
#include <iostream>
```

Fügen Sie das Ziel hinzu.

```
        Aws::CloudWatchEvents::EventBridgeClient cwe;

        Aws::CloudWatchEvents::Model::Target target;
        target.SetArn(lambda_arn);
        target.SetId(target_id);

        Aws::CloudWatchEvents::Model::PutTargetsRequest request;
        request.SetRule(rule_name);
        request.AddTargets(target);

        auto putTargetsOutcome = cwe.PutTargets(request);
        if (!putTargetsOutcome.IsSuccess())
        {
            std::cout << "Failed to create CloudWatch events target for rule "
                << rule_name << ": " <<
                putTargetsOutcome.GetError().GetMessage() << std::endl;
        }
        else
        {
            std::cout <<
                "Successfully created CloudWatch events target for rule "
                << rule_name << std::endl;
        }
```

- Einzelheiten zur API finden Sie PutTargetsin der AWS SDK for C++ API-Referenz.

CLI

AWS CLI

So fügen Sie Ziele für CloudWatch Event-Regeln hinzu

Im folgenden Beispiel wird eine Lambda-Funktion als Ziel einer Regel hinzugefügt:

```
aws events put-targets --rule DailyLambdaFunction --targets
 "Id"="1","Arn"="arn:aws:lambda:us-east-1:123456789012:function:MyFunctionName"
```

Im folgenden Beispiel wird ein Amazon-Kinesis-Stream als Ziel festgelegt, sodass Ereignisse, die von dieser Regel erfasst werden, an den Stream weitergeleitet werden:

```
aws events put-targets --rule EC2InstanceStateChanges --targets
 "Id"="1","Arn"="arn:aws:kinesis:us-east-1:123456789012:stream/
MyStream","RoleArn"="arn:aws:iam::123456789012:role/MyRoleForThisRule"
```

Im folgenden Beispiel werden zwei Amazon-Kinesis-Streams als Ziele für eine Regel festgelegt:

```
aws events put-targets --rule DailyLambdaFunction --targets
 "Id"="Target1","Arn"="arn:aws:kinesis:us-east-1:379642911888:stream/
MyStream1","RoleArn"="arn:aws:iam::379642911888:role/ MyRoleToAccessLambda"
  "Id"="Target2"," Arn"="arn:aws:kinesis:us-east-1:379642911888:stream/
MyStream2","RoleArn"="arn:aws:iam::379642911888:role/MyRoleToAccessLambda"
```

- Einzelheiten zur API finden Sie PutTargetsin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Fügen Sie ein Amazon-SNS-Thema als Ziel für eine Regel hinzu.

```
    // Add a rule which triggers an SNS target when a file is uploaded to an S3
    // bucket.
    public static void addSnsEventRule(EventBridgeClient eventBrClient, String
 ruleName, String topicArn,
            String topicName, String eventRuleName, String bucketName) {
        String targetID = java.util.UUID.randomUUID().toString();
```

```
        Target myTarget = Target.builder()
                .id(targetID)
                .arn(topicArn)
                .build();

        List<Target> targets = new ArrayList<>();
        targets.add(myTarget);
        PutTargetsRequest request = PutTargetsRequest.builder()
                .eventBusName(null)
                .targets(targets)
                .rule(ruleName)
                .build();

        eventBrClient.putTargets(request);
        System.out.println("Added event rule " + eventRuleName + " with Amazon
SNS target " + topicName + " for bucket "
                + bucketName + ".");
    }
```

Fügen Sie einen Eingabe-Transformator als Ziel für eine Regel hinzu.

```
    public static void updateCustomRuleTargetWithTransform(EventBridgeClient
eventBrClient, String topicArn,
            String ruleName) {
        String targetId = java.util.UUID.randomUUID().toString();
        InputTransformer inputTransformer = InputTransformer.builder()
                .inputTemplate("\"Notification: sample event was received.\"")
                .build();

        Target target = Target.builder()
                .id(targetId)
                .arn(topicArn)
                .inputTransformer(inputTransformer)
                .build();

        try {
            PutTargetsRequest targetsRequest = PutTargetsRequest.builder()
                    .rule(ruleName)
                    .targets(target)
                    .eventBusName(null)
                    .build();
```

```
            eventBrClient.putTargets(targetsRequest);
        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Einzelheiten zur API finden Sie PutTargetsin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Importieren Sie das SDK- und Client-Module und rufen Sie die API auf.

```
import {
  EventBridgeClient,
  PutTargetsCommand,
} from "@aws-sdk/client-eventbridge";

export const putTarget = async (
  existingRuleName = "some-rule",
  targetArn = "arn:aws:lambda:us-east-1:000000000000:function:test-func",
  uniqueId = Date.now().toString(),
) => {
  const client = new EventBridgeClient({});
  const response = await client.send(
    new PutTargetsCommand({
      Rule: existingRuleName,
      Targets: [
        {
          Arn: targetArn,
          Id: uniqueId,
        },
      ],
```

```
    }),
  );

  console.log("PutTargets response:");
  console.log(response);
  // PutTargets response:
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: 'f5b23b9a-2c17-45c1-ad5c-f926c3692e3d',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   FailedEntries: [],
  //   FailedEntryCount: 0
  // }

  return response;
};
```

- Einzelheiten zur API finden Sie PutTargetsin der AWS SDK for JavaScript API-Referenz.

SDK für JavaScript (v2)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create CloudWatchEvents service object
var ebevents = new AWS.EventBridge({ apiVersion: "2015-10-07" });

var params = {
  Rule: "DEMO_EVENT",
```

```
  Targets: [
    {
      Arn: "LAMBDA_FUNCTION_ARN",
      Id: "myEventBridgeTarget",
    },
  ],
};

ebevents.putTargets(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Einzelheiten zur API finden Sie PutTargetsin der AWS SDK for JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
// Add a rule that triggers an SNS target when a file is uploaded to an S3
 bucket.
suspend fun addSnsEventRule(ruleName: String?, topicArn: String?, topicName:
 String, eventRuleName: String, bucketName: String) {
    val targetID = UUID.randomUUID().toString()
    val myTarget = Target {
        id = targetID
        arn = topicArn
    }

    val targetsOb = mutableListOf<Target>()
    targetsOb.add(myTarget)
```

```
    val request = PutTargetsRequest {
        eventBusName = null
        targets = targetsOb
        rule = ruleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putTargets(request)
        println("Added event rule $eventRuleName with Amazon SNS target
 $topicName for bucket $bucketName.")
    }
}
```

Fügen Sie einen Eingabe-Transformator als Ziel für eine Regel hinzu.

```
suspend fun updateCustomRuleTargetWithTransform(topicArn: String?, ruleName:
 String?) {
    val targetId = UUID.randomUUID().toString()

    val inputTransformerOb = InputTransformer {
        inputTemplate = "\"Notification: sample event was received.\""
    }

    val target = Target {
        id = targetId
        arn = topicArn
        inputTransformer = inputTransformerOb
    }

    val targetsRequest = PutTargetsRequest {
        rule = ruleName
        targets = listOf(target)
        eventBusName = null
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putTargets(targetsRequest)
    }
}
```

- API-Details finden Sie PutTargetsin der API-Referenz zum AWS SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`RemoveTargets`** mit einem AWS SDK oder CLI

Die folgenden Codebeispiele zeigen, wie es verwendet wird`RemoveTargets`.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Entfernen Sie alle Ziele für eine Regel mithilfe des Regelnamens.

```
    /// <summary>
    /// Delete an event rule by name.
    /// </summary>
    /// <param name="ruleName">The name of the event rule.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> RemoveAllTargetsFromRule(string ruleName)
    {
        var targetIds = new List<string>();
        var request = new ListTargetsByRuleRequest()
        {
            Rule = ruleName
        };
        ListTargetsByRuleResponse targetsResponse;
        do
        {
            targetsResponse = await
_amazonEventBridge.ListTargetsByRuleAsync(request);
            targetIds.AddRange(targetsResponse.Targets.Select(t => t.Id));
            request.NextToken = targetsResponse.NextToken;

        } while (targetsResponse.NextToken is not null);
```

```
        var removeResponse = await _amazonEventBridge.RemoveTargetsAsync(
            new RemoveTargetsRequest()
            {
                Rule = ruleName,
                Ids = targetIds
            });

        if (removeResponse.FailedEntryCount > 0)
        {
            removeResponse.FailedEntries.ForEach(e =>
            {
                _logger.LogError(
                    $"Failed to remove target {e.TargetId}: {e.ErrorMessage},
code {e.ErrorCode}");
            });
        }

        return removeResponse.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie RemoveTargetsin der AWS SDK for .NET API-Referenz.

CLI

AWS CLI

So entfernen Sie ein Ziel für ein Ereignis

In diesem Beispiel wird der Amazon Kinesis Kinesis-Stream mit dem Namen MyStream 1 aus dem Ziel der Regel DailyLambdaFunction entfernt. Bei DailyLambdaFunction seiner Erstellung wurde dieser Stream als Ziel mit der ID Target1 festgelegt:

```
aws events remove-targets --rule "DailyLambdaFunction" --ids "Target1"
```

- Einzelheiten zur API finden Sie RemoveTargetsin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Entfernen Sie alle Ziele für eine Regel mithilfe des Regelnamens.

```
   public static void deleteTargetsFromRule(EventBridgeClient eventBrClient,
String eventRuleName) {
       // First, get all targets that will be deleted.
       ListTargetsByRuleRequest request = ListTargetsByRuleRequest.builder()
               .rule(eventRuleName)
               .build();

       ListTargetsByRuleResponse response =
eventBrClient.listTargetsByRule(request);
       List<Target> allTargets = response.targets();

       // Get all targets and delete them.
       for (Target myTarget : allTargets) {
           RemoveTargetsRequest removeTargetsRequest =
RemoveTargetsRequest.builder()
                   .rule(eventRuleName)
                   .ids(myTarget.id())
                   .build();

           eventBrClient.removeTargets(removeTargetsRequest);
           System.out.println("Successfully removed the target");
       }
   }
```

- Einzelheiten zur API finden Sie RemoveTargetsin der AWS SDK for Java 2.x API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
suspend fun deleteTargetsFromRule(eventRuleName: String?) {
    // First, get all targets that will be deleted.
    val request = ListTargetsByRuleRequest {
        rule = eventRuleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.listTargetsByRule(request)
        val allTargets = response.targets

        // Get all targets and delete them.
        if (allTargets != null) {
            for (myTarget in allTargets) {
                val removeTargetsRequest = RemoveTargetsRequest {
                    rule = eventRuleName
                    ids = listOf(myTarget.id.toString())
                }
                eventBrClient.removeTargets(removeTargetsRequest)
                println("Successfully removed the target")
            }
        }
    }
}
```

- API-Details finden Sie RemoveTargetsin der API-Referenz zum AWS SDK für Kotlin.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Szenarien für die EventBridge Verwendung von AWS SDKs

Die folgenden Codebeispiele zeigen Ihnen, wie Sie allgemeine Szenarien EventBridge mit AWS SDKs implementieren. Diese Szenarien zeigen Ihnen, wie Sie bestimmte Aufgaben erledigen können, indem Sie darin mehrere Funktionen aufrufen. EventBridge Jedes Szenario enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Beispiele

- Eine Regel in Amazon EventBridge mithilfe eines AWS SDK erstellen und auslösen
- Erste Schritte mit EventBridge Regeln und Zielen mithilfe eines AWS SDK

Eine Regel in Amazon EventBridge mithilfe eines AWS SDK erstellen und auslösen

Das folgende Codebeispiel zeigt, wie eine Regel in Amazon erstellt und ausgelöst wird EventBridge.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Rufen Sie die Funktionen in der richtigen Reihenfolge auf.

```
require "aws-sdk-sns"
require "aws-sdk-iam"
require "aws-sdk-cloudwatchevents"
require "aws-sdk-ec2"
require "aws-sdk-cloudwatch"
require "aws-sdk-cloudwatchlogs"
require "securerandom"
```

Überprüfen Sie, ob das angegebene Amazon Simple Notification Service (Amazon SNS)-Thema unter den für diese Funktion bereitgestellten Themen vorhanden ist.

```
# Checks whether the specified Amazon SNS
# topic exists among those provided to this function.
# This is a helper function that is called by the topic_exists? function.
#
# @param topics [Array] An array of Aws::SNS::Types::Topic objects.
# @param topic_arn [String] The ARN of the topic to find.
# @return [Boolean] true if the topic ARN was found; otherwise, false.
# @example
#   sns_client = Aws::SNS::Client.new(region: 'us-east-1')
#   response = sns_client.list_topics
#   if topic_found?(
#     response.topics,
#     'arn:aws:sns:us-east-1:111111111111:aws-doc-sdk-examples-topic'
#   )
#     puts 'Topic found.'
#   end

def topic_found?(topics, topic_arn)
  topics.each do |topic|
    return true if topic.topic_arn == topic_arn
  end
  return false
end
```

Überprüfen Sie, ob das angegebene Thema unter den für den Aufrufer in Amazon SNS verfügbaren Themen vorhanden ist.

```
# Checks whether the specified topic exists among those available to the
# caller in Amazon SNS.
#
# @param sns_client [Aws::SNS::Client] An initialized Amazon SNS client.
# @param topic_arn [String] The ARN of the topic to find.
# @return [Boolean] true if the topic ARN was found; otherwise, false.
# @example
#   exit 1 unless topic_exists?(
#     Aws::SNS::Client.new(region: 'us-east-1'),
#     'arn:aws:sns:us-east-1:111111111111:aws-doc-sdk-examples-topic'
#   )
def topic_exists?(sns_client, topic_arn)
  puts "Searching for topic with ARN '#{topic_arn}'..."
  response = sns_client.list_topics
  if response.topics.count.positive?
```

```
    if topic_found?(response.topics, topic_arn)
      puts "Topic found."
      return true
    end
    while response.next_page? do
      response = response.next_page
      if response.topics.count.positive?
        if topic_found?(response.topics, topic_arn)
          puts "Topic found."
          return true
        end
      end
    end
  end
  puts "Topic not found."
  return false
rescue StandardError => e
  puts "Topic not found: #{e.message}"
  return false
end
```

Erstellen Sie ein Thema in Amazon SNS und abonnieren Sie dann eine E-Mail-Adresse, um Benachrichtigungen zu diesem Thema zu erhalten.

```
# Creates a topic in Amazon SNS
# and then subscribes an email address to receive notifications to that topic.
#
# @param sns_client [Aws::SNS::Client] An initialized Amazon SNS client.
# @param topic_name [String] The name of the topic to create.
# @param email_address [String] The email address of the recipient to notify.
# @return [String] The ARN of the topic that was created.
# @example
#   puts create_topic(
#     Aws::SNS::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-topic',
#     'mary@example.com'
#   )
def create_topic(sns_client, topic_name, email_address)
  puts "Creating the topic named '#{topic_name}'..."
  topic_response = sns_client.create_topic(name: topic_name)
  puts "Topic created with ARN '#{topic_response.topic_arn}'."
  subscription_response = sns_client.subscribe(
```

```
    topic_arn: topic_response.topic_arn,
    protocol: "email",
    endpoint: email_address,
    return_subscription_arn: true
  )
  puts "Subscription created with ARN " \
    "'#{subscription_response.subscription_arn}'. Have the owner of the " \
    "email address '#{email_address}' check their inbox in a few minutes " \
    "and confirm the subscription to start receiving notification emails."
  return topic_response.topic_arn
rescue StandardError => e
  puts "Error creating or subscribing to topic: #{e.message}"
  return "Error"
end
```

Prüfen Sie, ob die angegebene AWS Identity and Access Management (IAM-) Rolle unter den für diese Funktion bereitgestellten Rollen existiert.

```
# Checks whether the specified AWS Identity and Access Management (IAM)
# role exists among those provided to this function.
# This is a helper function that is called by the role_exists? function.
#
# @param roles [Array] An array of Aws::IAM::Role objects.
# @param role_arn [String] The ARN of the role to find.
# @return [Boolean] true if the role ARN was found; otherwise, false.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-east-1')
#   response = iam_client.list_roles
#   if role_found?(
#     response.roles,
#     'arn:aws:iam::111111111111:role/aws-doc-sdk-examples-ec2-state-change'
#   )
#     puts 'Role found.'
#   end
def role_found?(roles, role_arn)
  roles.each do |role|
    return true if role.arn == role_arn
  end
  return false
end
```

Überprüfen Sie, ob die angegebene Rolle unter den für den Aufrufer in IAM verfügbaren Rollen vorhanden ist.

```
# Checks whether the specified role exists among those available to the
# caller in AWS Identity and Access Management (IAM).
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param role_arn [String] The ARN of the role to find.
# @return [Boolean] true if the role ARN was found; otherwise, false.
# @example
#   exit 1 unless role_exists?(
#     Aws::IAM::Client.new(region: 'us-east-1'),
#     'arn:aws:iam::111111111111:role/aws-doc-sdk-examples-ec2-state-change'
#   )
def role_exists?(iam_client, role_arn)
  puts "Searching for role with ARN '#{role_arn}'..."
  response = iam_client.list_roles
  if response.roles.count.positive?
    if role_found?(response.roles, role_arn)
      puts "Role found."
      return true
    end
    while response.next_page? do
      response = response.next_page
      if response.roles.count.positive?
        if role_found?(response.roles, role_arn)
          puts "Role found."
          return true
        end
      end
    end
  end
  puts "Role not found."
  return false
rescue StandardError => e
  puts "Role not found: #{e.message}"
  return false
end
```

Erstellen Sie eine Rolle in IAM.

```
# Creates a role in AWS Identity and Access Management (IAM).
```

```
# This role is used by a rule in Amazon EventBridge to allow
# that rule to operate within the caller's account.
# This role is designed to be used specifically by this code example.
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param role_name [String] The name of the role to create.
# @return [String] The ARN of the role that was created.
# @example
#   puts create_role(
#     Aws::IAM::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-ec2-state-change'
#   )
def create_role(iam_client, role_name)
  puts "Creating the role named '#{role_name}'..."
  response = iam_client.create_role(
    assume_role_policy_document: {
      'Version': "2012-10-17",
      'Statement': [
        {
          'Sid': "",
          'Effect': "Allow",
          'Principal': {
            'Service': "events.amazonaws.com"
          },
          'Action': "sts:AssumeRole"
        }
      ]
    }.to_json,
    path: "/",
    role_name: role_name
  )
  puts "Role created with ARN '#{response.role.arn}'."
  puts "Adding access policy to role..."
  iam_client.put_role_policy(
    policy_document: {
      'Version': "2012-10-17",
      'Statement': [
        {
          'Sid': "CloudWatchEventsFullAccess",
          'Effect': "Allow",
          'Resource': "*",
          'Action': "events:*"
        },
        {
```

```
          'Sid': "IAMPassRoleForCloudWatchEvents",
          'Effect': "Allow",
          'Resource': "arn:aws:iam::*:role/AWS_Events_Invoke_Targets",
          'Action': "iam:PassRole"
        }
      ]
    }.to_json,
    policy_name: "CloudWatchEventsPolicy",
    role_name: role_name
  )
  puts "Access policy added to role."
  return response.role.arn
rescue StandardError => e
  puts "Error creating role or adding policy to it: #{e.message}"
  puts "If the role was created, you must add the access policy " \
    "to the role yourself, or delete the role yourself and try again."
  return "Error"
end
```

Überprüft, ob die angegebene EventBridge Regel unter den für diese Funktion bereitgestellten Regeln existiert.

```
# Checks whether the specified Amazon EventBridge rule exists among
# those provided to this function.
# This is a helper function that is called by the rule_exists? function.
#
# @param rules [Array] An array of Aws::CloudWatchEvents::Types::Rule objects.
# @param rule_arn [String] The name of the rule to find.
# @return [Boolean] true if the name of the rule was found; otherwise, false.
# @example
#   cloudwatchevents_client = Aws::CloudWatch::Client.new(region: 'us-east-1')
#   response = cloudwatchevents_client.list_rules
#   if rule_found?(response.rules, 'aws-doc-sdk-examples-ec2-state-change')
#     puts 'Rule found.'
#   end
def rule_found?(rules, rule_name)
  rules.each do |rule|
    return true if rule.name == rule_name
  end
  return false
end
```

Überprüft, ob die angegebene Regel zu den Regeln gehört, die dem Anrufer zur Verfügung stehen. EventBridge

```
# Checks whether the specified rule exists among those available to the
# caller in Amazon EventBridge.
#
# @param cloudwatchevents_client [Aws::CloudWatchEvents::Client]
#   An initialized Amazon EventBridge client.
# @param rule_name [String] The name of the rule to find.
# @return [Boolean] true if the rule name was found; otherwise, false.
# @example
#   exit 1 unless rule_exists?(
#     Aws::CloudWatch::Client.new(region: 'us-east-1')
#     'aws-doc-sdk-examples-ec2-state-change'
#   )
def rule_exists?(cloudwatchevents_client, rule_name)
  puts "Searching for rule with name '#{rule_name}'..."
  response = cloudwatchevents_client.list_rules
  if response.rules.count.positive?
    if rule_found?(response.rules, rule_name)
      puts "Rule found."
      return true
    end
    while response.next_page? do
      response = response.next_page
      if response.rules.count.positive?
        if rule_found?(response.rules, rule_name)
          puts "Rule found."
          return true
        end
      end
    end
  end
  puts "Rule not found."
  return false
rescue StandardError => e
  puts "Rule not found: #{e.message}"
  return false
end
```

Erstellen Sie eine Regel in EventBridge.

```
# Creates a rule in Amazon EventBridge.
# This rule is triggered whenever an available instance in
# Amazon EC2 changes to the specified state.
# This rule is designed to be used specifically by this code example.
#
# Prerequisites:
#
# - A role in AWS Identity and Access Management (IAM) that is designed
#   to be used specifically by this code example.
# - A topic in Amazon SNS.
#
# @param cloudwatchevents_client [Aws::CloudWatchEvents::Client]
#   An initialized Amazon EventBridge client.
# @param rule_name [String] The name of the rule to create.
# @param rule_description [String] Some description for this rule.
# @param instance_state [String] The state that available instances in
#   Amazon EC2 must change to, to
#   trigger this rule.
# @param role_arn [String] The Amazon Resource Name (ARN) of the IAM role.
# @param target_id [String] Some identifying string for the rule's target.
# @param topic_arn [String] The ARN of the Amazon SNS topic.
# @return [Boolean] true if the rule was created; otherwise, false.
# @example
#   exit 1 unless rule_created?(
#     Aws::CloudWatch::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-ec2-state-change',
#     'Triggers when any available EC2 instance starts.',
#     'running',
#     'arn:aws:iam::111111111111:role/aws-doc-sdk-examples-ec2-state-change',
#     'sns-topic',
#     'arn:aws:sns:us-east-1:111111111111:aws-doc-sdk-examples-topic'
#   )
def rule_created?(
  cloudwatchevents_client,
  rule_name,
  rule_description,
  instance_state,
  role_arn,
  target_id,
  topic_arn
)
  puts "Creating rule with name '#{rule_name}'..."
  put_rule_response = cloudwatchevents_client.put_rule(
```

```
    name: rule_name,
    description: rule_description,
    event_pattern: {
      'source': [
        "aws.ec2"
      ],
      'detail-type': [
        "EC2 Instance State-change Notification"
      ],
      'detail': {
        'state': [
          instance_state
        ]
      }
    }.to_json,
    state: "ENABLED",
    role_arn: role_arn
  )
  puts "Rule created with ARN '#{put_rule_response.rule_arn}'."

  put_targets_response = cloudwatchevents_client.put_targets(
    rule: rule_name,
    targets: [
      {
        id: target_id,
        arn: topic_arn
      }
    ]
  )
  if put_targets_response.key?(:failed_entry_count) &&
      put_targets_response.failed_entry_count > 0
    puts "Error(s) adding target to rule:"
    put_targets_response.failed_entries.each do |failure|
      puts failure.error_message
    end
    return false
  else
    return true
  end
rescue StandardError => e
  puts "Error creating rule or adding target to rule: #{e.message}"
  puts "If the rule was created, you must add the target " \
    "to the rule yourself, or delete the rule yourself and try again."
  return false
```

```
end
```

Prüfen Sie, ob die angegebene Protokollgruppe zu den Protokollgruppen gehört, die dem Anrufer in Amazon CloudWatch Logs zur Verfügung stehen.

```
# Checks to see whether the specified log group exists among those available
# to the caller in Amazon CloudWatch Logs.
#
# @param cloudwatchlogs_client [Aws::CloudWatchLogs::Client] An initialized
#   Amazon CloudWatch Logs client.
# @param log_group_name [String] The name of the log group to find.
# @return [Boolean] true if the log group name was found; otherwise, false.
# @example
#   exit 1 unless log_group_exists?(
#     Aws::CloudWatchLogs::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-cloudwatch-log'
#   )
def log_group_exists?(cloudwatchlogs_client, log_group_name)
  puts "Searching for log group with name '#{log_group_name}'..."
  response = cloudwatchlogs_client.describe_log_groups(
    log_group_name_prefix: log_group_name
  )
  if response.log_groups.count.positive?
    response.log_groups.each do |log_group|
      if log_group.log_group_name == log_group_name
        puts "Log group found."
        return true
      end
    end
  end
  puts "Log group not found."
  return false
rescue StandardError => e
  puts "Log group not found: #{e.message}"
  return false
end
```

Erstellen Sie eine Protokollgruppe in CloudWatch Logs.

```
# Creates a log group in Amazon CloudWatch Logs.
#
```

```
# @param cloudwatchlogs_client [Aws::CloudWatchLogs::Client] An initialized
#   Amazon CloudWatch Logs client.
# @param log_group_name [String] The name of the log group to create.
# @return [Boolean] true if the log group name was created; otherwise, false.
# @example
#   exit 1 unless log_group_created?(
#     Aws::CloudWatchLogs::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-cloudwatch-log'
#   )
def log_group_created?(cloudwatchlogs_client, log_group_name)
  puts "Attempting to create log group with the name '#{log_group_name}'..."
  cloudwatchlogs_client.create_log_group(log_group_name: log_group_name)
  puts "Log group created."
  return true
rescue StandardError => e
  puts "Error creating log group: #{e.message}"
  return false
end
```

Schreiben Sie ein Ereignis in einen Protokollstream in CloudWatch Logs.

```
# Writes an event to a log stream in Amazon CloudWatch Logs.
#
# Prerequisites:
#
# - A log group in Amazon CloudWatch Logs.
# - A log stream within the log group.
#
# @param cloudwatchlogs_client [Aws::CloudWatchLogs::Client] An initialized
#   Amazon CloudWatch Logs client.
# @param log_group_name [String] The name of the log group.
# @param log_stream_name [String] The name of the log stream within
#   the log group.
# @param message [String] The message to write to the log stream.
# @param sequence_token [String] If available, the sequence token from the
#   message that was written immediately before this message. This sequence
#   token is returned by Amazon CloudWatch Logs whenever you programmatically
#   write a message to the log stream.
# @return [String] The sequence token that is returned by
#   Amazon CloudWatch Logs after successfully writing the message to the
#   log stream.
# @example
```

```
#   puts log_event(
#     Aws::EC2::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-cloudwatch-log'
#     '2020/11/19/53f985be-199f-408e-9a45-fc242df41fEX',
#     "Instance 'i-033c48ef067af3dEX' restarted.",
#     '495426724868310740095796045676567882148068632824696073EX'
#   )
def log_event(
  cloudwatchlogs_client,
  log_group_name,
  log_stream_name,
  message,
  sequence_token
)
  puts "Attempting to log '#{message}' to log stream '#{log_stream_name}'..."
  event = {
    log_group_name: log_group_name,
    log_stream_name: log_stream_name,
    log_events: [
      {
        timestamp: (Time.now.utc.to_f.round(3) * 1_000).to_i,
        message: message
      }
    ]
  }
  unless sequence_token.empty?
    event[:sequence_token] = sequence_token
  end

  response = cloudwatchlogs_client.put_log_events(event)
  puts "Message logged."
  return response.next_sequence_token
rescue StandardError => e
  puts "Message not logged: #{e.message}"
end
```

Starten Sie eine Amazon Elastic Compute Cloud (Amazon EC2) -Instance neu und fügen Sie Informationen über die zugehörige Aktivität zu einem Protokollstream in CloudWatch Logs hinzu.

```

# Restarts an Amazon EC2 instance
```

```
# and adds information about the related activity to a log stream
# in Amazon CloudWatch Logs.
#
# Prerequisites:
#
# - The Amazon EC2 instance to restart.
# - The log group in Amazon CloudWatch Logs to add related activity
#   information to.
#
# @param ec2_client [Aws::EC2::Client] An initialized Amazon EC2 client.
# @param cloudwatchlogs_client [Aws::CloudWatchLogs::Client]
#   An initialized Amazon CloudWatch Logs client.
# @param instance_id [String] The ID of the instance.
# @param log_group_name [String] The name of the log group.
# @return [Boolean] true if the instance was restarted and the information
#   was written to the log stream; otherwise, false.
# @example
#   exit 1 unless instance_restarted?(
#     Aws::EC2::Client.new(region: 'us-east-1'),
#     Aws::CloudWatchLogs::Client.new(region: 'us-east-1'),
#     'i-033c48ef067af3dEX',
#     'aws-doc-sdk-examples-cloudwatch-log'
#   )
def instance_restarted?(
  ec2_client,
  cloudwatchlogs_client,
  instance_id,
  log_group_name
)
  log_stream_name = "#{Time.now.year}/#{Time.now.month}/#{Time.now.day}/" \
    "#{SecureRandom.uuid}"
  cloudwatchlogs_client.create_log_stream(
    log_group_name: log_group_name,
    log_stream_name: log_stream_name
  )
  sequence_token = ""

  puts "Attempting to stop the instance with the ID '#{instance_id}'. " \
    "This might take a few minutes..."
  ec2_client.stop_instances(instance_ids: [instance_id])
  ec2_client.wait_until(:instance_stopped, instance_ids: [instance_id])
  puts "Instance stopped."
  sequence_token = log_event(
    cloudwatchlogs_client,
```

```
    log_group_name,
    log_stream_name,
    "Instance '#{instance_id}' stopped.",
    sequence_token
  )

  puts "Attempting to restart the instance. This might take a few minutes..."
  ec2_client.start_instances(instance_ids: [instance_id])
  ec2_client.wait_until(:instance_running, instance_ids: [instance_id])
  puts "Instance restarted."
  sequence_token = log_event(
    cloudwatchlogs_client,
    log_group_name,
    log_stream_name,
    "Instance '#{instance_id}' restarted.",
    sequence_token
  )

  return true
rescue StandardError => e
  puts "Error creating log stream or stopping or restarting the instance: " \
    "#{e.message}"
  log_event(
    cloudwatchlogs_client,
    log_group_name,
    log_stream_name,
    "Error stopping or starting instance '#{instance_id}': #{e.message}",
    sequence_token
  )
  return false
end
```

Zeigt Informationen zur Aktivität für eine Regel in an EventBridge.

```
# Displays information about activity for a rule in Amazon EventBridge.
#
# Prerequisites:
#
# - A rule in Amazon EventBridge.
#
# @param cloudwatch_client [Amazon::CloudWatch::Client] An initialized
#   Amazon CloudWatch client.
```

```
# @param rule_name [String] The name of the rule.
# @param start_time [Time] The timestamp that determines the first datapoint
#   to return. Can also be expressed as DateTime, Date, Integer, or String.
# @param end_time [Time] The timestamp that determines the last datapoint
#   to return. Can also be expressed as DateTime, Date, Integer, or String.
# @param period [Integer] The interval, in seconds, to check for activity.
# @example
#   display_rule_activity(
#     Aws::CloudWatch::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-ec2-state-change',
#     Time.now - 600, # Start checking from 10 minutes ago.
#     Time.now, # Check up until now.
#     60 # Check every minute during those 10 minutes.
#   )
def display_rule_activity(
  cloudwatch_client,
  rule_name,
  start_time,
  end_time,
  period
)
  puts "Attempting to display rule activity..."
  response = cloudwatch_client.get_metric_statistics(
    namespace: "AWS/Events",
    metric_name: "Invocations",
    dimensions: [
      {
        name: "RuleName",
        value: rule_name
      }
    ],
    start_time: start_time,
    end_time: end_time,
    period: period,
    statistics: ["Sum"],
    unit: "Count"
  )

  if response.key?(:datapoints) && response.datapoints.count.positive?
    puts "The event rule '#{rule_name}' was triggered:"
    response.datapoints.each do |datapoint|
      puts "  #{datapoint.sum} time(s) at #{datapoint.timestamp}"
    end
  else
```

```
    puts "The event rule '#{rule_name}' was not triggered during the " \
      "specified time period."
  end
rescue StandardError => e
  puts "Error getting information about event rule activity: #{e.message}"
end
```

Zeigt Protokollinformationen für alle Protokolldatenströme in einer Protokollgruppe „ CloudWatch Protokolle“ an.

```
# Displays log information for all of the log streams in a log group in
# Amazon CloudWatch Logs.
#
# Prerequisites:
#
# - A log group in Amazon CloudWatch Logs.
#
# @param cloudwatchlogs_client [Amazon::CloudWatchLogs::Client] An initialized
#   Amazon CloudWatch Logs client.
# @param log_group_name [String] The name of the log group.
# @example
#   display_log_data(
#     Amazon::CloudWatchLogs::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-cloudwatch-log'
#   )
def display_log_data(cloudwatchlogs_client, log_group_name)
  puts "Attempting to display log stream data for the log group " \
    "named '#{log_group_name}'..."
  describe_log_streams_response = cloudwatchlogs_client.describe_log_streams(
    log_group_name: log_group_name,
    order_by: "LastEventTime",
    descending: true
  )
  if describe_log_streams_response.key?(:log_streams) &&
      describe_log_streams_response.log_streams.count.positive?
    describe_log_streams_response.log_streams.each do |log_stream|
      get_log_events_response = cloudwatchlogs_client.get_log_events(
        log_group_name: log_group_name,
        log_stream_name: log_stream.log_stream_name
      )
      puts "\nLog messages for '#{log_stream.log_stream_name}':"
      puts "-" * (log_stream.log_stream_name.length + 20)
```

```
      if get_log_events_response.key?(:events) &&
          get_log_events_response.events.count.positive?
        get_log_events_response.events.each do |event|
          puts event.message
        end
      else
        puts "No log messages for this log stream."
      end
    end
  end
rescue StandardError => e
  puts "Error getting information about the log streams or their messages: " \
    "#{e.message}"
end
```

Zeigt dem Anrufer eine Erinnerung an, alle zugehörigen AWS Ressourcen, die er nicht mehr benötigt, manuell zu bereinigen.

```
# Displays a reminder to the caller to manually clean up any associated
# AWS resources that they no longer need.
#
# @param topic_name [String] The name of the Amazon SNS topic.
# @param role_name [String] The name of the IAM role.
# @param rule_name [String] The name of the Amazon EventBridge rule.
# @param log_group_name [String] The name of the Amazon CloudWatch Logs log
 group.
# @param instance_id [String] The ID of the Amazon EC2 instance.
# @example
#   manual_cleanup_notice(
#     'aws-doc-sdk-examples-topic',
#     'aws-doc-sdk-examples-cloudwatch-events-rule-role',
#     'aws-doc-sdk-examples-ec2-state-change',
#     'aws-doc-sdk-examples-cloudwatch-log',
#     'i-033c48ef067af3dEX'
#   )
def manual_cleanup_notice(
  topic_name, role_name, rule_name, log_group_name, instance_id
)
  puts "-" * 10
  puts "Some of the following AWS resources might still exist in your account."
  puts "If you no longer want to use this code example, then to clean up"
```

```
  puts "your AWS account and avoid unexpected costs, you might want to"
  puts "manually delete any of the following resources if they exist:"
  puts "- The Amazon SNS topic named '#{topic_name}'."
  puts "- The IAM role named '#{role_name}'."
  puts "- The Amazon EventBridge rule named '#{rule_name}'."
  puts "- The Amazon CloudWatch Logs log group named '#{log_group_name}'."
  puts "- The Amazon EC2 instance with the ID '#{instance_id}'."
end

# Example usage:
def run_me
  # Properties for the Amazon SNS topic.
  topic_name = "aws-doc-sdk-examples-topic"
  email_address = "mary@example.com"
  # Properties for the IAM role.
  role_name = "aws-doc-sdk-examples-cloudwatch-events-rule-role"
  # Properties for the Amazon EventBridge rule.
  rule_name = "aws-doc-sdk-examples-ec2-state-change"
  rule_description = "Triggers when any available EC2 instance starts."
  instance_state = "running"
  target_id = "sns-topic"
  # Properties for the Amazon EC2 instance.
  instance_id = "i-033c48ef067af3dEX"
  # Properties for displaying the event rule's activity.
  start_time = Time.now - 600 # Go back over the past 10 minutes
                              # (10 minutes * 60 seconds = 600 seconds).
  end_time = Time.now
  period = 60 # Look back every 60 seconds over the past 10 minutes.
  # Properties for the Amazon CloudWatch Logs log group.
  log_group_name = "aws-doc-sdk-examples-cloudwatch-log"
  # AWS service clients for this code example.
  region = "us-east-1"
  sts_client = Aws::STS::Client.new(region: region)
  sns_client = Aws::SNS::Client.new(region: region)
  iam_client = Aws::IAM::Client.new(region: region)
  cloudwatchevents_client = Aws::CloudWatchEvents::Client.new(region: region)
  ec2_client = Aws::EC2::Client.new(region: region)
  cloudwatch_client = Aws::CloudWatch::Client.new(region: region)
  cloudwatchlogs_client = Aws::CloudWatchLogs::Client.new(region: region)

  # Get the caller's account ID for use in forming
  # Amazon Resource Names (ARNs) that this code relies on later.
  account_id = sts_client.get_caller_identity.account
```

```
# If the Amazon SNS topic doesn't exist, create it.
topic_arn = "arn:aws:sns:#{region}:#{account_id}:#{topic_name}"
unless topic_exists?(sns_client, topic_arn)
  topic_arn = create_topic(sns_client, topic_name, email_address)
  if topic_arn == "Error"
    puts "Could not create the Amazon SNS topic correctly. Program stopped."
    manual_cleanup_notice(
      topic_name, role_name, rule_name, log_group_name, instance_id
    )
    exit 1
  end
end

# If the IAM role doesn't exist, create it.
role_arn = "arn:aws:iam::#{account_id}:role/#{role_name}"
unless role_exists?(iam_client, role_arn)
  role_arn = create_role(iam_client, role_name)
  if role_arn == "Error"
    puts "Could not create the IAM role correctly. Program stopped."
    manual_cleanup_notice(
      topic_name, role_name, rule_name, log_group_name, instance_id
    )
  end
end

# If the Amazon EventBridge rule doesn't exist, create it.
unless rule_exists?(cloudwatchevents_client, rule_name)
  unless rule_created?(
    cloudwatchevents_client,
    rule_name,
    rule_description,
    instance_state,
    role_arn,
    target_id,
    topic_arn
  )
    puts "Could not create the Amazon EventBridge rule correctly. " \
      "Program stopped."
    manual_cleanup_notice(
      topic_name, role_name, rule_name, log_group_name, instance_id
    )
  end
end
```

```
  # If the Amazon CloudWatch Logs log group doesn't exist, create it.
  unless log_group_exists?(cloudwatchlogs_client, log_group_name)
    unless log_group_created?(cloudwatchlogs_client, log_group_name)
      puts "Could not create the Amazon CloudWatch Logs log group " \
      "correctly. Program stopped."
      manual_cleanup_notice(
        topic_name, role_name, rule_name, log_group_name, instance_id
      )
    end
  end

  # Restart the Amazon EC2 instance, which triggers the rule.
  unless instance_restarted?(
    ec2_client,
    cloudwatchlogs_client,
    instance_id,
    log_group_name
  )
    puts "Could not restart the instance to trigger the rule. " \
      "Continuing anyway to show information about the rule and logs..."
  end

  # Display how many times the rule was triggered over the past 10 minutes.
  display_rule_activity(
    cloudwatch_client,
    rule_name,
    start_time,
    end_time,
    period
  )

  # Display related log data in Amazon CloudWatch Logs.
  display_log_data(cloudwatchlogs_client, log_group_name)

  # Reminder the caller to clean up any AWS resources that are used
  # by this code example and are no longer needed.
  manual_cleanup_notice(
    topic_name, role_name, rule_name, log_group_name, instance_id
  )
end

run_me if $PROGRAM_NAME == __FILE__
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Ruby -API-Referenz.
  - PutEvents
  - PutRule

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erste Schritte mit EventBridge Regeln und Zielen mithilfe eines AWS SDK

Die folgenden Code-Beispiele veranschaulichen Folgendes:

- Erstellen Sie eine Regel und fügen Sie ihr ein Ziel hinzu.
- Aktivieren und deaktivieren Sie Regeln.
- Listen Sie Regeln und Ziele auf und aktualisieren Sie sie.
- Senden Sie Ereignisse und bereinigen Sie dann die Ressourcen.

.NET

AWS SDK for .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

Führen Sie ein interaktives Szenario an einer Eingabeaufforderung aus.

```
public class EventBridgeScenario
{
    /*
    Before running this .NET code example, set up your development environment,
 including your credentials.

    This .NET example performs the following tasks with Amazon EventBridge:
    - Create a rule.
    - Add a target to a rule.
```

```
    - Enable and disable rules.
    - List rules and targets.
    - Update rules and targets.
    - Send events.
    - Delete the rule.
    */

    private static ILogger logger = null!;
    private static EventBridgeWrapper _eventBridgeWrapper = null!;
    private static IConfiguration _configuration = null!;

    private static IAmazonIdentityManagementService? _iamClient = null!;
    private static IAmazonSimpleNotificationService? _snsClient = null!;
    private static IAmazonS3 _s3Client = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon EventBridge.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonEventBridge>()
            .AddAWSService<IAmazonIdentityManagementService>()
            .AddAWSService<IAmazonS3>()
            .AddAWSService<IAmazonSimpleNotificationService>()
            .AddTransient<EventBridgeWrapper>()
            )
            .Build();

        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally, load local settings.
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<EventBridgeScenario>();
```

```
        ServicesSetup(host);

        string topicArn = "";
        string roleArn = "";

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Welcome to the Amazon EventBridge example scenario.");
        Console.WriteLine(new string('-', 80));

        try
        {
            roleArn = await CreateRole();

            await CreateBucketWithEventBridgeEvents();

            await AddEventRule(roleArn);

            await ListEventRules();

            topicArn = await CreateSnsTopic();

            var email = await SubscribeToSnsTopic(topicArn);

            await AddSnsTarget(topicArn);

            await ListTargets();

            await ListRulesForTarget(topicArn);

            await UploadS3File(_s3Client);

            await ChangeRuleState(false);

            await GetRuleState();

            await UpdateSnsEventRule(topicArn);

            await ChangeRuleState(true);

            await UploadS3File(_s3Client);

            await UpdateToCustomRule(topicArn);

            await TriggerCustomRule(email);
```

```
            await CleanupResources(topicArn);
        }
        catch (Exception ex)
        {
            logger.LogError(ex, "There was a problem executing the scenario.");
            await CleanupResources(topicArn);
        }
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("The Amazon EventBridge example scenario is
complete.");
        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _eventBridgeWrapper =
host.Services.GetRequiredService<EventBridgeWrapper>();
        _snsClient =
host.Services.GetRequiredService<IAmazonSimpleNotificationService>();
        _s3Client = host.Services.GetRequiredService<IAmazonS3>();
        _iamClient =
host.Services.GetRequiredService<IAmazonIdentityManagementService>();
    }

    /// <summary>
    /// Create a role to be used by EventBridge.
    /// </summary>
    /// <returns>The role Amazon Resource Name (ARN).</returns>
    public static async Task<string> CreateRole()
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Creating a role to use with EventBridge and attaching
managed policy AmazonEventBridgeFullAccess.");
        Console.WriteLine(new string('-', 80));

        var roleName = _configuration["roleName"];

        var assumeRolePolicy = "{" +
                               "\"Version\": \"2012-10-17\"," +
```

```
                                    "\"Statement\": [{" +
                                    "\"Effect\": \"Allow\"," +
                                    "\"Principal\": {" +
                                    $"\"Service\": \"events.amazonaws.com\"" +
                                    "}," +
                                    "\"Action\": \"sts:AssumeRole\"" +
                                    "}]" +
                                    "}";

        var roleResult = await _iamClient!.CreateRoleAsync(
            new CreateRoleRequest()
            {
                AssumeRolePolicyDocument = assumeRolePolicy,
                Path = "/",
                RoleName = roleName
            });

        await _iamClient.AttachRolePolicyAsync(
            new AttachRolePolicyRequest()
            {
                PolicyArn = "arn:aws:iam::aws:policy/
AmazonEventBridgeFullAccess",
                RoleName = roleName
            });
        // Allow time for the role to be ready.
        Thread.Sleep(10000);
        return roleResult.Role.Arn;
    }

    /// <summary>
    /// Create an Amazon Simple Storage Service (Amazon S3) bucket with
 EventBridge events enabled.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task CreateBucketWithEventBridgeEvents()
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Creating an S3 bucket with EventBridge events
 enabled.");

        var testBucketName = _configuration["testBucketName"];

        var bucketExists = await
 Amazon.S3.Util.AmazonS3Util.DoesS3BucketExistV2Async(_s3Client,
```

```
            testBucketName);

        if (!bucketExists)
        {
            await _s3Client.PutBucketAsync(new PutBucketRequest()
            {
                BucketName = testBucketName,
                UseClientRegion = true
            });
        }

        await _s3Client.PutBucketNotificationAsync(new
PutBucketNotificationRequest()
        {
            BucketName = testBucketName,
            EventBridgeConfiguration = new EventBridgeConfiguration()
        });

        Console.WriteLine($"\tAdded bucket {testBucketName} with EventBridge
events enabled.");

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Create and upload a file to an S3 bucket to trigger an event.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task UploadS3File(IAmazonS3 s3Client)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Uploading a file to the test bucket. This will trigger
a subscription email.");

        var testBucketName = _configuration["testBucketName"];

        var fileName = $"example_upload_{DateTime.UtcNow.Ticks}.txt";

        // Create the file if it does not already exist.
        if (!File.Exists(fileName))
        {
            await using StreamWriter sw = File.CreateText(fileName);
            await sw.WriteLineAsync(
                "This is a sample file for testing uploads.");
```

```
        }

        await s3Client.PutObjectAsync(new PutObjectRequest()
        {
            FilePath = fileName,
            BucketName = testBucketName
        });

        Console.WriteLine($"\tPress Enter to continue.");
        Console.ReadLine();

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Create an Amazon Simple Notification Service (Amazon SNS) topic to use as
an EventBridge target.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task<string> CreateSnsTopic()
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "Creating an Amazon Simple Notification Service (Amazon SNS) topic
for email subscriptions.");

        var topicName = _configuration["topicName"];

        string topicPolicy = "{" +
                             "\"Version\": \"2012-10-17\"," +
                             "\"Statement\": [{" +
                             "\"Sid\": \"EventBridgePublishTopic\"," +
                             "\"Effect\": \"Allow\"," +
                             "\"Principal\": {" +
                             $"\"Service\": \"events.amazonaws.com\"" +
                             "}," +
                             "\"Resource\": \"*\"," +
                             "\"Action\": \"sns:Publish\"" +
                             "}]" +
                             "}";

        var topicAttributes = new Dictionary<string, string>()
        {
            { "Policy", topicPolicy }
```

```
        };

        var topicResponse = await _snsClient!.CreateTopicAsync(new
CreateTopicRequest()
        {
            Name = topicName,
            Attributes = topicAttributes

        });

        Console.WriteLine($"\tAdded topic {topicName} for email subscriptions.");

        Console.WriteLine(new string('-', 80));

        return topicResponse.TopicArn;
    }

    /// <summary>
    /// Subscribe a user email to an SNS topic.
    /// </summary>
    /// <param name="topicArn">The ARN of the SNS topic.</param>
    /// <returns>The user's email.</returns>
    private static async Task<string> SubscribeToSnsTopic(string topicArn)
    {
        Console.WriteLine(new string('-', 80));


        string email = "";
        while (string.IsNullOrEmpty(email))
        {
            Console.WriteLine("Enter your email to subscribe to the Amazon SNS
topic:");
            email = Console.ReadLine()!;
        }

        var subscriptions = new List<string>();
        var paginatedSubscriptions =
_snsClient!.Paginators.ListSubscriptionsByTopic(
            new ListSubscriptionsByTopicRequest()
            {
                TopicArn = topicArn
            });

        // Get the entire list using the paginator.
```

```
        await foreach (var subscription in paginatedSubscriptions.Subscriptions)
        {
            subscriptions.Add(subscription.Endpoint);
        }

        if (subscriptions.Contains(email))
        {
            Console.WriteLine($"\tYour email is already subscribed.");
            Console.WriteLine(new string('-', 80));
            return email;
        }

        await _snsClient.SubscribeAsync(new SubscribeRequest()
        {
            TopicArn = topicArn,
            Protocol = "email",
            Endpoint = email
        });

        Console.WriteLine($"Use the link in the email you received to confirm
your subscription, then press Enter to continue.");

        Console.ReadLine();

        Console.WriteLine(new string('-', 80));
        return email;
    }

    /// <summary>
    /// Add a rule which triggers when a file is uploaded to an S3 bucket.
    /// </summary>
    /// <param name="roleArn">The ARN of the role used by EventBridge.</param>
    /// <returns>Async task.</returns>
    private static async Task AddEventRule(string roleArn)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Creating an EventBridge event that sends an email when
an Amazon S3 object is created.");

        var eventRuleName = _configuration["eventRuleName"];
        var testBucketName = _configuration["testBucketName"];

        await _eventBridgeWrapper.PutS3UploadRule(roleArn, eventRuleName,
testBucketName);
```

```
        Console.WriteLine($"\tAdded event rule {eventRuleName} for bucket
{testBucketName}.");

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Add an SNS target to the rule.
    /// </summary>
    /// <param name="topicArn">The ARN of the SNS topic.</param>
    /// <returns>Async task.</returns>
    private static async Task AddSnsTarget(string topicArn)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Adding a target to the rule to that sends an email
when the rule is triggered.");

        var eventRuleName = _configuration["eventRuleName"];
        var testBucketName = _configuration["testBucketName"];
        var topicName = _configuration["topicName"];
        await _eventBridgeWrapper.AddSnsTargetToRule(eventRuleName, topicArn);
        Console.WriteLine($"\tAdded event rule {eventRuleName} with Amazon SNS
target {topicName} for bucket {testBucketName}.");

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// List the event rules on the default event bus.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task ListEventRules()
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Current event rules:");

        var rules = await _eventBridgeWrapper.ListAllRulesForEventBus();
        rules.ForEach(r => Console.WriteLine($"\tRule: {r.Name} Description:
{r.Description} State: {r.State}"));

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
```

```
    /// Update the event target to use a transform.
    /// </summary>
    /// <param name="topicArn">The SNS topic ARN target to update.</param>
    /// <returns>Async task.</returns>
    private static async Task UpdateSnsEventRule(string topicArn)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Let's update the event target with a transform.");

        var eventRuleName = _configuration["eventRuleName"];
        var testBucketName = _configuration["testBucketName"];

        await
_eventBridgeWrapper.UpdateS3UploadRuleTargetWithTransform(eventRuleName,
topicArn);
        Console.WriteLine($"\tUpdated event rule {eventRuleName} with Amazon SNS
target {topicArn} for bucket {testBucketName}.");

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Update the rule to use a custom event pattern.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task UpdateToCustomRule(string topicArn)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Updating the event pattern to be triggered by a custom
event instead.");

        var eventRuleName = _configuration["eventRuleName"];

        await _eventBridgeWrapper.UpdateCustomEventPattern(eventRuleName);

        Console.WriteLine($"\tUpdated event rule {eventRuleName} to custom
pattern.");
        await
_eventBridgeWrapper.UpdateCustomRuleTargetWithTransform(eventRuleName,
            topicArn);

        Console.WriteLine($"\tUpdated event target {topicArn}.");

        Console.WriteLine(new string('-', 80));
```

```
    }

    /// <summary>
    /// Send rule events for a custom rule using the user's email address.
    /// </summary>
    /// <param name="email">The email address to include.</param>
    /// <returns>Async task.</returns>
    private static async Task TriggerCustomRule(string email)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Sending an event to trigger the rule. This will
trigger a subscription email.");

        await _eventBridgeWrapper.PutCustomEmailEvent(email);

        Console.WriteLine($"\tEvents have been sent. Press Enter to continue.");
        Console.ReadLine();

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// List all of the targets for a rule.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task ListTargets()
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("List all of the targets for a particular rule.");

        var eventRuleName = _configuration["eventRuleName"];
        var targets = await
_eventBridgeWrapper.ListAllTargetsOnRule(eventRuleName);
        targets.ForEach(t => Console.WriteLine($"\tTarget: {t.Arn} Id: {t.Id}
Input: {t.Input}"));

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// List all of the rules for a particular target.
    /// </summary>
    /// <param name="topicArn">The ARN of the SNS topic.</param>
    /// <returns>Async task.</returns>
```

```
    private static async Task ListRulesForTarget(string topicArn)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("List all of the rules for a particular target.");

        var rules = await _eventBridgeWrapper.ListAllRuleNamesByTarget(topicArn);
        rules.ForEach(r => Console.WriteLine($"\tRule: {r}"));

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Enable or disable a particular rule.
    /// </summary>
    /// <param name="isEnabled">True to enable the rule, otherwise false.</param>
    /// <returns>Async task.</returns>
    private static async Task ChangeRuleState(bool isEnabled)
    {
        Console.WriteLine(new string('-', 80));
        var eventRuleName = _configuration["eventRuleName"];

        if (!isEnabled)
        {
            Console.WriteLine($"Disabling the rule: {eventRuleName}");
            await _eventBridgeWrapper.DisableRuleByName(eventRuleName);
        }
        else
        {
            Console.WriteLine($"Enabling the rule: {eventRuleName}");
            await _eventBridgeWrapper.EnableRuleByName(eventRuleName);
        }

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Get the current state of the rule.
    /// </summary>
    /// <returns>Async task.</returns>
    private static async Task GetRuleState()
    {
        Console.WriteLine(new string('-', 80));
        var eventRuleName = _configuration["eventRuleName"];
```

```
        var state = await
_eventBridgeWrapper.GetRuleStateByRuleName(eventRuleName);
        Console.WriteLine($"Rule {eventRuleName} is in current state {state}.");

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="topicArn">The ARN of the SNS topic to clean up.</param>
    /// <returns>Async task.</returns>
    private static async Task CleanupResources(string topicArn)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine($"Clean up resources.");

        var eventRuleName = _configuration["eventRuleName"];
        if (GetYesNoResponse($"\tDelete all targets and event rule
{eventRuleName}? (y/n)"))
        {
            Console.WriteLine($"\tRemoving all targets from the event rule.");
            await _eventBridgeWrapper.RemoveAllTargetsFromRule(eventRuleName);

            Console.WriteLine($"\tDeleting event rule.");
            await _eventBridgeWrapper.DeleteRuleByName(eventRuleName);
        }

        var topicName = _configuration["topicName"];
        if (GetYesNoResponse($"\tDelete Amazon SNS subscription topic
{topicName}? (y/n)"))
        {
            Console.WriteLine($"\tDeleting topic.");
            await _snsClient!.DeleteTopicAsync(new DeleteTopicRequest()
            {
                TopicArn = topicArn
            });
        }

        var bucketName = _configuration["testBucketName"];
        if (GetYesNoResponse($"\tDelete Amazon S3 bucket {bucketName}? (y/n)"))
        {
            Console.WriteLine($"\tDeleting bucket.");
            // Delete all objects in the bucket.
```

```
            var deleteList = await _s3Client.ListObjectsV2Async(new
 ListObjectsV2Request()
            {
                BucketName = bucketName
            });
            await _s3Client.DeleteObjectsAsync(new DeleteObjectsRequest()
            {
                BucketName = bucketName,
                Objects = deleteList.S3Objects
                    .Select(o => new KeyVersion { Key = o.Key }).ToList()
            });
            // Now delete the bucket.
            await _s3Client.DeleteBucketAsync(new DeleteBucketRequest()
            {
                BucketName = bucketName
            });
        }

        var roleName = _configuration["roleName"];
        if (GetYesNoResponse($"\tDelete role {roleName}? (y/n)"))
        {
            Console.WriteLine($"\tDetaching policy and deleting role.");

            await _iamClient!.DetachRolePolicyAsync(new DetachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = "arn:aws:iam::aws:policy/
AmazonEventBridgeFullAccess",
            });

            await _iamClient!.DeleteRoleAsync(new DeleteRoleRequest()
            {
                RoleName = roleName
            });
        }

        Console.WriteLine(new string('-', 80));
    }

    /// <summary>
    /// Helper method to get a yes or no response from the user.
    /// </summary>
    /// <param name="question">The question string to print on the console.</
param>
```

```
    /// <returns>True if the user responds with a yes.</returns>
    private static bool GetYesNoResponse(string question)
    {
        Console.WriteLine(question);
        var ynResponse = Console.ReadLine();
        var response = ynResponse != null &&
                       ynResponse.Equals("y",
                           StringComparison.InvariantCultureIgnoreCase);
        return response;
    }
}
```

Erstellen Sie eine Klasse, die EventBridge Operationen umschließt.

```
/// <summary>
/// Wrapper for Amazon EventBridge operations.
/// </summary>
public class EventBridgeWrapper
{
    private readonly IAmazonEventBridge _amazonEventBridge;
    private readonly ILogger<EventBridgeWrapper> _logger;

    /// <summary>
    /// Constructor for the EventBridge wrapper.
    /// </summary>
    /// <param name="amazonEventBridge">The injected EventBridge client.</param>
    /// <param name="logger">The injected logger for the wrapper.</param>
    public EventBridgeWrapper(IAmazonEventBridge amazonEventBridge,
 ILogger<EventBridgeWrapper> logger)

    {
        _amazonEventBridge = amazonEventBridge;
        _logger = logger;
    }

    /// <summary>
    /// Get the state for a rule by the rule name.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <param name="eventBusName">The optional name of the event bus. If empty,
 uses the default event bus.</param>
```

```
    /// <returns>The state of the rule.</returns>
    public async Task<RuleState> GetRuleStateByRuleName(string ruleName, string?
eventBusName = null)
    {
        var ruleResponse = await _amazonEventBridge.DescribeRuleAsync(
            new DescribeRuleRequest()
            {
                Name = ruleName,
                EventBusName = eventBusName
            });
        return ruleResponse.State;
    }

    /// <summary>
    /// Enable a particular rule on an event bus.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> EnableRuleByName(string ruleName)
    {
        var ruleResponse = await _amazonEventBridge.EnableRuleAsync(
            new EnableRuleRequest()
            {
                Name = ruleName
            });
        return ruleResponse.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Disable a particular rule on an event bus.
    /// </summary
    /// <param name="ruleName">The name of the rule.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DisableRuleByName(string ruleName)
    {
        var ruleResponse = await _amazonEventBridge.DisableRuleAsync(
            new DisableRuleRequest()
            {
                Name = ruleName
            });
        return ruleResponse.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
```

```
    /// List the rules on an event bus.
    /// </summary>
    /// <param name="eventBusArn">The optional ARN of the event bus. If empty,
uses the default event bus.</param>
    /// <returns>The list of rules.</returns>
    public async Task<List<Rule>> ListAllRulesForEventBus(string? eventBusArn =
null)
    {
        var results = new List<Rule>();
        var request = new ListRulesRequest()
        {
            EventBusName = eventBusArn
        };
        // Get all of the pages of rules.
        ListRulesResponse response;
        do
        {
            response = await _amazonEventBridge.ListRulesAsync(request);
            results.AddRange(response.Rules);
            request.NextToken = response.NextToken;

        } while (response.NextToken is not null);

        return results;
    }

    /// <summary>
    /// List all of the targets matching a rule by name.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <returns>The list of targets.</returns>
    public async Task<List<Target>> ListAllTargetsOnRule(string ruleName)
    {
        var results = new List<Target>();
        var request = new ListTargetsByRuleRequest()
        {
            Rule = ruleName
        };
        ListTargetsByRuleResponse response;
        do
        {
            response = await _amazonEventBridge.ListTargetsByRuleAsync(request);
            results.AddRange(response.Targets);
            request.NextToken = response.NextToken;
```

```
        } while (response.NextToken is not null);

        return results;
    }

    /// <summary>
    /// List names of all rules matching a target.
    /// </summary>
    /// <param name="targetArn">The ARN of the target.</param>
    /// <returns>The list of rule names.</returns>
    public async Task<List<string>> ListAllRuleNamesByTarget(string targetArn)
    {
        var results = new List<string>();
        var request = new ListRuleNamesByTargetRequest()
        {
            TargetArn = targetArn
        };
        ListRuleNamesByTargetResponse response;
        do
        {
            response = await
 _amazonEventBridge.ListRuleNamesByTargetAsync(request);
            results.AddRange(response.RuleNames);
            request.NextToken = response.NextToken;

        } while (response.NextToken is not null);

        return results;
    }

    /// <summary>
    /// Create a new event rule that triggers when an Amazon S3 object is created
 in a bucket.
    /// </summary>
    /// <param name="roleArn">The ARN of the role.</param>
    /// <param name="ruleName">The name to give the rule.</param>
    /// <param name="bucketName">The name of the bucket to trigger the event.</
param>
    /// <returns>The ARN of the new rule.</returns>
    public async Task<string> PutS3UploadRule(string roleArn, string ruleName,
 string bucketName)
    {
        string eventPattern = "{" +
```

```
                                "\"source\": [\"aws.s3\"]," +
                                    "\"detail-type\": [\"Object Created\"]," +
                                    "\"detail\": {" +
                                        "\"bucket\": {" +
                                            "\"name\": [\"" + bucketName + "\"]"
+
                                        "}" +
                                    "}" +
                                "}";

        var response = await _amazonEventBridge.PutRuleAsync(
            new PutRuleRequest()
            {
                Name = ruleName,
                Description = "Example S3 upload rule for EventBridge",
                RoleArn = roleArn,
                EventPattern = eventPattern
            });

        return response.RuleArn;
    }

    /// <summary>
    /// Update an Amazon S3 object created rule with a transform on the target.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <param name="targetArn">The ARN of the target.</param>
    /// <param name="eventBusArn">Optional event bus ARN. If empty, uses the
default event bus.</param>
    /// <returns>The ID of the target.</returns>
    public async Task<string> UpdateS3UploadRuleTargetWithTransform(string
ruleName, string targetArn, string? eventBusArn = null)
    {
        var targetID = Guid.NewGuid().ToString();

        var targets = new List<Target>
        {
            new Target()
            {
                Id = targetID,
                Arn = targetArn,
                InputTransformer = new InputTransformer()
                {
                    InputPathsMap = new Dictionary<string, string>()
```

```
                    {
                        {"bucket", "$.detail.bucket.name"},
                        {"time", "$.time"}
                    },
                    InputTemplate = "\"Notification: an object was uploaded to
bucket <bucket> at <time>.\""
                }
            }
        };
        var response = await _amazonEventBridge.PutTargetsAsync(
            new PutTargetsRequest()
            {
                EventBusName = eventBusArn,
                Rule = ruleName,
                Targets = targets,
            });
        if (response.FailedEntryCount > 0)
        {
            response.FailedEntries.ForEach(e =>
            {
                _logger.LogError(
                    $"Failed to add target {e.TargetId}: {e.ErrorMessage}, code
{e.ErrorCode}");
            });
        }
        return targetID;
    }

    /// <summary>
    /// Update a custom rule with a transform on the target.
    /// </summary>
    /// <param name="ruleName">The name of the rule.</param>
    /// <param name="targetArn">The ARN of the target.</param>
    /// <param name="eventBusArn">Optional event bus ARN. If empty, uses the
default event bus.</param>
    /// <returns>The ID of the target.</returns>
    public async Task<string> UpdateCustomRuleTargetWithTransform(string
ruleName, string targetArn, string? eventBusArn = null)
    {
        var targetID = Guid.NewGuid().ToString();

        var targets = new List<Target>
        {
            new Target()
```

```
            {
                Id = targetID,
                Arn = targetArn,
                InputTransformer = new InputTransformer()
                {
                    InputTemplate = "\"Notification: sample event was received.
\""
                }
            }
        };
        var response = await _amazonEventBridge.PutTargetsAsync(
            new PutTargetsRequest()
            {
                EventBusName = eventBusArn,
                Rule = ruleName,
                Targets = targets,
            });
        if (response.FailedEntryCount > 0)
        {
            response.FailedEntries.ForEach(e =>
            {
                _logger.LogError(
                    $"Failed to add target {e.TargetId}: {e.ErrorMessage}, code
 {e.ErrorCode}");
            });
        }
        return targetID;
    }

    /// <summary>
    /// Add an event to the event bus that includes an email, message, and time.
    /// </summary>
    /// <param name="email">The email to use in the event detail of the custom
 event.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> PutCustomEmailEvent(string email)
    {
        var eventDetail = new
        {
            UserEmail = email,
            Message = "This event was generated by example code.",
            UtcTime = DateTime.UtcNow.ToString("g")
        };
        var response = await _amazonEventBridge.PutEventsAsync(
```

```
            new PutEventsRequest()
            {
                Entries = new List<PutEventsRequestEntry>()
                {
                    new PutEventsRequestEntry()
                    {
                        Source = "ExampleSource",
                        Detail = JsonSerializer.Serialize(eventDetail),
                        DetailType = "ExampleType"
                    }
                }
            });

        return response.FailedEntryCount == 0;
    }

    /// <summary>
    /// Update a rule to use a custom defined event pattern.
    /// </summary>
    /// <param name="ruleName">The name of the rule to update.</param>
    /// <returns>The ARN of the updated rule.</returns>
    public async Task<string> UpdateCustomEventPattern(string ruleName)
    {
        string customEventsPattern = "{" +
                                     "\"source\": [\"ExampleSource\"]," +
                                     "\"detail-type\": [\"ExampleType\"]" +
                                     "}";

        var response = await _amazonEventBridge.PutRuleAsync(
            new PutRuleRequest()
            {
                Name = ruleName,
                Description = "Custom test rule",
                EventPattern = customEventsPattern
            });

        return response.RuleArn;
    }

    /// <summary>
    /// Add an Amazon SNS target topic to a rule.
    /// </summary>
    /// <param name="ruleName">The name of the rule to update.</param>
    /// <param name="targetArn">The ARN of the Amazon SNS target.</param>
```

```
   /// <param name="eventBusArn">The optional event bus name, uses default if
empty.</param>
   /// <returns>The ID of the target.</returns>
   public async Task<string> AddSnsTargetToRule(string ruleName, string
targetArn, string? eventBusArn = null)
   {
       var targetID = Guid.NewGuid().ToString();

       // Create the list of targets and add a new target.
       var targets = new List<Target>
       {
           new Target()
           {
               Arn = targetArn,
               Id = targetID
           }
       };

       // Add the targets to the rule.
       var response = await _amazonEventBridge.PutTargetsAsync(
           new PutTargetsRequest()
           {
               EventBusName = eventBusArn,
               Rule = ruleName,
               Targets = targets,
           });

       if (response.FailedEntryCount > 0)
       {
           response.FailedEntries.ForEach(e =>
           {
               _logger.LogError(
                   $"Failed to add target {e.TargetId}: {e.ErrorMessage}, code
{e.ErrorCode}");
           });
       }

       return targetID;
   }

   /// <summary>
   /// Delete an event rule by name.
   /// </summary>
   /// <param name="ruleName">The name of the event rule.</param>
```

```
    /// <returns>True if successful.</returns>
    public async Task<bool> RemoveAllTargetsFromRule(string ruleName)
    {
        var targetIds = new List<string>();
        var request = new ListTargetsByRuleRequest()
        {
            Rule = ruleName
        };
        ListTargetsByRuleResponse targetsResponse;
        do
        {
            targetsResponse = await
_amazonEventBridge.ListTargetsByRuleAsync(request);
            targetIds.AddRange(targetsResponse.Targets.Select(t => t.Id));
            request.NextToken = targetsResponse.NextToken;

        } while (targetsResponse.NextToken is not null);

        var removeResponse = await _amazonEventBridge.RemoveTargetsAsync(
            new RemoveTargetsRequest()
            {
                Rule = ruleName,
                Ids = targetIds
            });

        if (removeResponse.FailedEntryCount > 0)
        {
            removeResponse.FailedEntries.ForEach(e =>
            {
                _logger.LogError(
                    $"Failed to remove target {e.TargetId}: {e.ErrorMessage},
code {e.ErrorCode}");
            });
        }

        return removeResponse.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Delete an event rule by name.
    /// </summary>
    /// <param name="ruleName">The name of the event rule.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteRuleByName(string ruleName)
```

```
    {
        var response = await _amazonEventBridge.DeleteRuleAsync(
            new DeleteRuleRequest()
            {
                Name = ruleName
            });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for .NET -API-Referenz.
  - DeleteRule
  - DescribeRule
  - DisableRule
  - EnableRule
  - ListRuleNamesByTarget
  - ListRules
  - ListTargetsByRule
  - PutEvents
  - PutRule
  - PutTargets

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu. GitHub Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
```

```
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * This Java code example performs the following tasks:
 *
 * This Java V2 example performs the following tasks with Amazon EventBridge:
 *
 * 1. Creates an AWS Identity and Access Management (IAM) role to use with
 * Amazon EventBridge.
 * 2. Amazon Simple Storage Service (Amazon S3) bucket with EventBridge events
 * enabled.
 * 3. Creates a rule that triggers when an object is uploaded to Amazon S3.
 * 4. Lists rules on the event bus.
 * 5. Creates a new Amazon Simple Notification Service (Amazon SNS) topic and
 * lets the user subscribe to it.
 * 6. Adds a target to the rule that sends an email to the specified topic.
 * 7. Creates an EventBridge event that sends an email when an Amazon S3 object
 * is created.
 * 8. Lists Targets.
 * 9. Lists the rules for the same target.
 * 10. Triggers the rule by uploading a file to the Amazon S3 bucket.
 * 11. Disables a specific rule.
 * 12. Checks and print the state of the rule.
 * 13. Adds a transform to the rule to change the text of the email.
 * 14. Enables a specific rule.
 * 15. Triggers the updated rule by uploading a file to the Amazon S3 bucket.
 * 16. Updates the rule to be a custom rule pattern.
 * 17. Sending an event to trigger the rule.
 * 18. Cleans up resources.
 *
 */
public class EventbridgeMVP {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws InterruptedException,
 IOException {
        final String usage = """

                Usage:
                    <roleName> <bucketName> <topicName> <eventRuleName>
```

```
                Where:
                    roleName - The name of the role to create.
                    bucketName - The Amazon Simple Storage Service (Amazon S3)
bucket name to create.
                    topicName - The name of the Amazon Simple Notification
Service (Amazon SNS) topic to create.
                    eventRuleName - The Amazon EventBridge rule name to create.
                """;

        if (args.length != 5) {
            System.out.println(usage);
            System.exit(1);
        }

        String polJSON = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                "\"Service\": \"events.amazonaws.com\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        Scanner sc = new Scanner(System.in);
        String roleName = args[0];
        String bucketName = args[1];
        String topicName = args[2];
        String eventRuleName = args[3];

        Region region = Region.US_EAST_1;
        EventBridgeClient eventBrClient = EventBridgeClient.builder()
                .region(region)
                .build();

        S3Client s3Client = S3Client.builder()
                .region(region)
                .build();

        Region regionGl = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(regionGl)
```

```
                .build();

        SnsClient snsClient = SnsClient.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon EventBridge example
scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out
                .println("1. Create an AWS Identity and Access Management (IAM)
role to use with Amazon EventBridge.");
        String roleArn = createIAMRole(iam, roleName, polJSON);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("2. Create an S3 bucket with EventBridge events
enabled.");
        if (checkBucket(s3Client, bucketName)) {
            System.out.println("Bucket " + bucketName + " already exists. Ending
this scenario.");
            System.exit(1);
        }

        createBucket(s3Client, bucketName);
        Thread.sleep(3000);
        setBucketNotification(s3Client, bucketName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Create a rule that triggers when an object is
uploaded to Amazon S3.");
        Thread.sleep(10000);
        addEventRule(eventBrClient, roleArn, bucketName, eventRuleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. List rules on the event bus.");
        listRules(eventBrClient);
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("5. Create a new SNS topic for testing and let the
user subscribe to the topic.");
        String topicArn = createSnsTopic(snsClient, topicName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6. Add a target to the rule that sends an email to
the specified topic.");
        System.out.println("Enter your email to subscribe to the Amazon SNS
topic:");
        String email = sc.nextLine();
        subEmail(snsClient, topicArn, email);
        System.out.println(
                "Use the link in the email you received to confirm your
subscription. Then, press Enter to continue.");
        sc.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Create an EventBridge event that sends an email
when an Amazon S3 object is created.");
        addSnsEventRule(eventBrClient, eventRuleName, topicArn, topicName,
eventRuleName, bucketName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 8. List Targets.");
        listTargets(eventBrClient, eventRuleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 9. List the rules for the same target.");
        listTargetRules(eventBrClient, topicArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 10. Trigger the rule by uploading a file to the S3
bucket.");
        System.out.println("Press Enter to continue.");
        sc.nextLine();
        uploadTextFiletoS3(s3Client, bucketName);
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("11. Disable a specific rule.");
        changeRuleState(eventBrClient, eventRuleName, false);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("12. Check and print the state of the rule.");
        checkRule(eventBrClient, eventRuleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("13. Add a transform to the rule to change the text of
the email.");
        updateSnsEventRule(eventBrClient, topicArn, eventRuleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("14. Enable a specific rule.");
        changeRuleState(eventBrClient, eventRuleName, true);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 15. Trigger the updated rule by uploading a file to
the S3 bucket.");
        System.out.println("Press Enter to continue.");
        sc.nextLine();
        uploadTextFiletoS3(s3Client, bucketName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 16. Update the rule to be a custom rule pattern.");
        updateToCustomRule(eventBrClient, eventRuleName);
        System.out.println("Updated event rule " + eventRuleName + " to use a
custom pattern.");
        updateCustomRuleTargetWithTransform(eventBrClient, topicArn,
eventRuleName);
        System.out.println("Updated event target " + topicArn + ".");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("17. Sending an event to trigger the rule. This will
trigger a subscription email.");
        triggerCustomRule(eventBrClient, email);
        System.out.println("Events have been sent. Press Enter to continue.");
```

```
        sc.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("18. Clean up resources.");
        System.out.println("Do you want to clean up resources (y/n)");
        String ans = sc.nextLine();
        if (ans.compareTo("y") == 0) {
            cleanupResources(eventBrClient, snsClient, s3Client, iam, topicArn,
eventRuleName, bucketName, roleName);
        } else {
            System.out.println("The resources will not be cleaned up. ");
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The Amazon EventBridge example scenario has
successfully completed.");
        System.out.println(DASHES);
    }

    public static void cleanupResources(EventBridgeClient eventBrClient,
SnsClient snsClient, S3Client s3Client,
            IamClient iam, String topicArn, String eventRuleName, String
bucketName, String roleName) {
        System.out.println("Removing all targets from the event rule.");
        deleteTargetsFromRule(eventBrClient, eventRuleName);
        deleteRuleByName(eventBrClient, eventRuleName);
        deleteSNSTopic(snsClient, topicArn);
        deleteS3Bucket(s3Client, bucketName);
        deleteRole(iam, roleName);
    }

    public static void deleteRole(IamClient iam, String roleName) {
        String policyArn = "arn:aws:iam::aws:policy/AmazonEventBridgeFullAccess";
        DetachRolePolicyRequest policyRequest = DetachRolePolicyRequest.builder()
                .policyArn(policyArn)
                .roleName(roleName)
                .build();

        iam.detachRolePolicy(policyRequest);
        System.out.println("Successfully detached policy " + policyArn + " from
role " + roleName);
```

```
        // Delete the role.
        DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                .roleName(roleName)
                .build();

        iam.deleteRole(roleRequest);
        System.out.println("*** Successfully deleted " + roleName);
    }

    public static void deleteS3Bucket(S3Client s3Client, String bucketName) {
        // Remove all the objects from the S3 bucket.
        ListObjectsRequest listObjects = ListObjectsRequest.builder()
                .bucket(bucketName)
                .build();

        ListObjectsResponse res = s3Client.listObjects(listObjects);
        List<S3Object> objects = res.contents();
        ArrayList<ObjectIdentifier> toDelete = new ArrayList<>();

        for (S3Object myValue : objects) {
            toDelete.add(ObjectIdentifier.builder()
                    .key(myValue.key())
                    .build());
        }

        DeleteObjectsRequest dor = DeleteObjectsRequest.builder()
                .bucket(bucketName)
                .delete(Delete.builder()
                        .objects(toDelete).build())
                .build();

        s3Client.deleteObjects(dor);

        // Delete the S3 bucket.
        DeleteBucketRequest deleteBucketRequest = DeleteBucketRequest.builder()
                .bucket(bucketName)
                .build();

        s3Client.deleteBucket(deleteBucketRequest);
        System.out.println("You have deleted the bucket and the objects");
    }

    // Delete the SNS topic.
    public static void deleteSNSTopic(SnsClient snsClient, String topicArn) {
```

```
        try {
            DeleteTopicRequest request = DeleteTopicRequest.builder()
                    .topicArn(topicArn)
                    .build();

            DeleteTopicResponse result = snsClient.deleteTopic(request);
            System.out.println("\n\nStatus was " +
result.sdkHttpResponse().statusCode());

        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteRuleByName(EventBridgeClient eventBrClient, String
ruleName) {
        DeleteRuleRequest ruleRequest = DeleteRuleRequest.builder()
                .name(ruleName)
                .build();

        eventBrClient.deleteRule(ruleRequest);
        System.out.println("Successfully deleted the rule");
    }

    public static void deleteTargetsFromRule(EventBridgeClient eventBrClient,
String eventRuleName) {
        // First, get all targets that will be deleted.
        ListTargetsByRuleRequest request = ListTargetsByRuleRequest.builder()
                .rule(eventRuleName)
                .build();

        ListTargetsByRuleResponse response =
eventBrClient.listTargetsByRule(request);
        List<Target> allTargets = response.targets();

        // Get all targets and delete them.
        for (Target myTarget : allTargets) {
            RemoveTargetsRequest removeTargetsRequest =
RemoveTargetsRequest.builder()
                    .rule(eventRuleName)
                    .ids(myTarget.id())
                    .build();
```

```
            eventBrClient.removeTargets(removeTargetsRequest);
            System.out.println("Successfully removed the target");
        }
    }

    public static void triggerCustomRule(EventBridgeClient eventBrClient, String
email) {
        String json = "{" +
                "\"UserEmail\": \"" + email + "\"," +
                "\"Message\": \"This event was generated by example code.\"," +
                "\"UtcTime\": \"Now.\"" +
                "}";

        PutEventsRequestEntry entry = PutEventsRequestEntry.builder()
                .source("ExampleSource")
                .detail(json)
                .detailType("ExampleType")
                .build();

        PutEventsRequest eventsRequest = PutEventsRequest.builder()
                .entries(entry)
                .build();

        eventBrClient.putEvents(eventsRequest);
    }

    public static void updateCustomRuleTargetWithTransform(EventBridgeClient
eventBrClient, String topicArn,
            String ruleName) {
        String targetId = java.util.UUID.randomUUID().toString();
        InputTransformer inputTransformer = InputTransformer.builder()
                .inputTemplate("\"Notification: sample event was received.\"")
                .build();

        Target target = Target.builder()
                .id(targetId)
                .arn(topicArn)
                .inputTransformer(inputTransformer)
                .build();

        try {
            PutTargetsRequest targetsRequest = PutTargetsRequest.builder()
                    .rule(ruleName)
                    .targets(target)
```

```
                    .eventBusName(null)
                    .build();

            eventBrClient.putTargets(targetsRequest);
        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void updateToCustomRule(EventBridgeClient eventBrClient, String
ruleName) {
        String customEventsPattern = "{" +
                "\"source\": [\"ExampleSource\"]," +
                "\"detail-type\": [\"ExampleType\"]" +
                "}";

        PutRuleRequest request = PutRuleRequest.builder()
                .name(ruleName)
                .description("Custom test rule")
                .eventPattern(customEventsPattern)
                .build();

        eventBrClient.putRule(request);
    }

    // Update an Amazon S3 object created rule with a transform on the target.
    public static void updateSnsEventRule(EventBridgeClient eventBrClient, String
topicArn, String ruleName) {
        String targetId = java.util.UUID.randomUUID().toString();
        Map<String, String> myMap = new HashMap<>();
        myMap.put("bucket", "$.detail.bucket.name");
        myMap.put("time", "$.time");

        InputTransformer inputTransformer = InputTransformer.builder()
                .inputTemplate("\"Notification: an object was uploaded to bucket
<bucket> at <time>.\"")
                .inputPathsMap(myMap)
                .build();

        Target target = Target.builder()
                .id(targetId)
                .arn(topicArn)
                .inputTransformer(inputTransformer)
```

```
                .build();

        try {
            PutTargetsRequest targetsRequest = PutTargetsRequest.builder()
                    .rule(ruleName)
                    .targets(target)
                    .eventBusName(null)
                    .build();

            eventBrClient.putTargets(targetsRequest);

        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void checkRule(EventBridgeClient eventBrClient, String
eventRuleName) {
        try {
            DescribeRuleRequest ruleRequest = DescribeRuleRequest.builder()
                    .name(eventRuleName)
                    .build();

            DescribeRuleResponse response =
eventBrClient.describeRule(ruleRequest);
            System.out.println("The state of the rule is " +
response.stateAsString());

        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void changeRuleState(EventBridgeClient eventBrClient, String
eventRuleName, Boolean isEnabled) {
        try {
            if (!isEnabled) {
                System.out.println("Disabling the rule: " + eventRuleName);
                DisableRuleRequest ruleRequest = DisableRuleRequest.builder()
                        .name(eventRuleName)
                        .build();
```

```
                eventBrClient.disableRule(ruleRequest);
            } else {
                System.out.println("Enabling the rule: " + eventRuleName);
                EnableRuleRequest ruleRequest = EnableRuleRequest.builder()
                        .name(eventRuleName)
                        .build();
                eventBrClient.enableRule(ruleRequest);
            }

        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Create and upload a file to an S3 bucket to trigger an event.
    public static void uploadTextFiletoS3(S3Client s3Client, String bucketName)
throws IOException {
        // Create a unique file name.
        String fileSuffix = new SimpleDateFormat("yyyyMMddHHmmss").format(new
Date());
        String fileName = "TextFile" + fileSuffix + ".txt";

        File myFile = new File(fileName);
        FileWriter fw = new FileWriter(myFile.getAbsoluteFile());
        BufferedWriter bw = new BufferedWriter(fw);
        bw.write("This is a sample file for testing uploads.");
        bw.close();

        try {
            PutObjectRequest putOb = PutObjectRequest.builder()
                    .bucket(bucketName)
                    .key(fileName)
                    .build();

            s3Client.putObject(putOb, RequestBody.fromFile(myFile));

        } catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

```
   public static void listTargetRules(EventBridgeClient eventBrClient, String
topicArn) {
       ListRuleNamesByTargetRequest ruleNamesByTargetRequest =
ListRuleNamesByTargetRequest.builder()
               .targetArn(topicArn)
               .build();

       ListRuleNamesByTargetResponse response =
eventBrClient.listRuleNamesByTarget(ruleNamesByTargetRequest);
       List<String> rules = response.ruleNames();
       for (String rule : rules) {
           System.out.println("The rule name is " + rule);
       }
   }

   public static void listTargets(EventBridgeClient eventBrClient, String
ruleName) {
       ListTargetsByRuleRequest ruleRequest = ListTargetsByRuleRequest.builder()
               .rule(ruleName)
               .build();

       ListTargetsByRuleResponse res =
eventBrClient.listTargetsByRule(ruleRequest);
       List<Target> targetsList = res.targets();
       for (Target target: targetsList) {
           System.out.println("Target ARN: "+target.arn());
       }
   }

   // Add a rule which triggers an SNS target when a file is uploaded to an S3
   // bucket.
   public static void addSnsEventRule(EventBridgeClient eventBrClient, String
ruleName, String topicArn,
           String topicName, String eventRuleName, String bucketName) {
       String targetID = java.util.UUID.randomUUID().toString();
       Target myTarget = Target.builder()
               .id(targetID)
               .arn(topicArn)
               .build();

       List<Target> targets = new ArrayList<>();
       targets.add(myTarget);
       PutTargetsRequest request = PutTargetsRequest.builder()
               .eventBusName(null)
```

```
                .targets(targets)
                .rule(ruleName)
                .build();

        eventBrClient.putTargets(request);
        System.out.println("Added event rule " + eventRuleName + " with Amazon
SNS target " + topicName + " for bucket "
                + bucketName + ".");
    }

    public static void subEmail(SnsClient snsClient, String topicArn, String
email) {
        try {
            SubscribeRequest request = SubscribeRequest.builder()
                    .protocol("email")
                    .endpoint(email)
                    .returnSubscriptionArn(true)
                    .topicArn(topicArn)
                    .build();

            SubscribeResponse result = snsClient.subscribe(request);
            System.out.println("Subscription ARN: " + result.subscriptionArn() +
"\n\n Status is "
                    + result.sdkHttpResponse().statusCode());

        } catch (SnsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void listRules(EventBridgeClient eventBrClient) {
        try {
            ListRulesRequest rulesRequest = ListRulesRequest.builder()
                    .eventBusName("default")
                    .limit(10)
                    .build();

            ListRulesResponse response = eventBrClient.listRules(rulesRequest);
            List<Rule> rules = response.rules();
            for (Rule rule : rules) {
                System.out.println("The rule name is : " + rule.name());
                System.out.println("The rule description is : " +
rule.description());
```

```
                System.out.println("The rule state is : " +
rule.stateAsString());
            }

        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static String createSnsTopic(SnsClient snsClient, String topicName) {
        String topicPolicy = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Sid\": \"EventBridgePublishTopic\"," +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                "\"Service\": \"events.amazonaws.com\"" +
                "}," +
                "\"Resource\": \"*\"," +
                "\"Action\": \"sns:Publish\"" +
                "}]" +
                "}";

        Map<String, String> topicAttributes = new HashMap<>();
        topicAttributes.put("Policy", topicPolicy);
        CreateTopicRequest topicRequest = CreateTopicRequest.builder()
                .name(topicName)
                .attributes(topicAttributes)
                .build();

        CreateTopicResponse response = snsClient.createTopic(topicRequest);
        System.out.println("Added topic " + topicName + " for email
subscriptions.");
        return response.topicArn();
    }

    // Create a new event rule that triggers when an Amazon S3 object is created
in
    // a bucket.
    public static void addEventRule(EventBridgeClient eventBrClient, String
roleArn, String bucketName,
            String eventRuleName) {
        String pattern = "{\n" +
```

```
                "  \"source\": [\"aws.s3\"],\n" +
                "  \"detail-type\": [\"Object Created\"],\n" +
                "  \"detail\": {\n" +
                "    \"bucket\": {\n" +
                "      \"name\": [\"" + bucketName + "\"]\n" +
                "    }\n" +
                "  }\n" +
                "}";

        try {
            PutRuleRequest ruleRequest = PutRuleRequest.builder()
                    .description("Created by using the AWS SDK for Java v2")
                    .name(eventRuleName)
                    .eventPattern(pattern)
                    .roleArn(roleArn)
                    .build();

            PutRuleResponse ruleResponse = eventBrClient.putRule(ruleRequest);
            System.out.println("The ARN of the new rule is " +
ruleResponse.ruleArn());

        } catch (EventBridgeException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Determine if the S3 bucket exists.
    public static Boolean checkBucket(S3Client s3Client, String bucketName) {
        try {
            HeadBucketRequest headBucketRequest = HeadBucketRequest.builder()
                    .bucket(bucketName)
                    .build();

            s3Client.headBucket(headBucketRequest);
            return true;
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        return false;
    }

    // Set the S3 bucket notification configuration.
```

```
   public static void setBucketNotification(S3Client s3Client, String
bucketName) {
       try {
           EventBridgeConfiguration eventBridgeConfiguration =
EventBridgeConfiguration.builder()
                   .build();

           NotificationConfiguration configuration =
NotificationConfiguration.builder()
                   .eventBridgeConfiguration(eventBridgeConfiguration)
                   .build();

           PutBucketNotificationConfigurationRequest configurationRequest =
PutBucketNotificationConfigurationRequest
                   .builder()
                   .bucket(bucketName)
                   .notificationConfiguration(configuration)
                   .skipDestinationValidation(true)
                   .build();

           s3Client.putBucketNotificationConfiguration(configurationRequest);
           System.out.println("Added bucket " + bucketName + " with EventBridge
events enabled.");

       } catch (S3Exception e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void createBucket(S3Client s3Client, String bucketName) {
       try {
           S3Waiter s3Waiter = s3Client.waiter();
           CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
                   .bucket(bucketName)
                   .build();

           s3Client.createBucket(bucketRequest);
           HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
                   .bucket(bucketName)
                   .build();

           // Wait until the bucket is created and print out the response.
```

```
            WaiterResponse<HeadBucketResponse> waiterResponse =
 s3Waiter.waitUntilBucketExists(bucketRequestWait);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println(bucketName + " is ready");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static String createIAMRole(IamClient iam, String rolename, String
 polJSON) {
        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(polJSON)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            AttachRolePolicyRequest rolePolicyRequest =
 AttachRolePolicyRequest.builder()
                    .roleName(rolename)
                    .policyArn("arn:aws:iam::aws:policy/
AmazonEventBridgeFullAccess")
                    .build();

            iam.attachRolePolicy(rolePolicyRequest);
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Java 2.x -API-Referenz.
  - DeleteRule
  - DescribeRule

- DisableRule
- EnableRule
- ListRuleNamesByTarget
- ListRules
- ListTargetsByRule
- PutEvents
- PutRule
- PutTargets

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr GitHub. Sie sehen das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel-Repository einrichten und ausführen.

```
/*
 Before running this Kotlin code example, set up your development environment,
 including your credentials.

 For more information, see the following documentation topic:
 https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html

 This Kotlin example performs the following tasks with Amazon EventBridge:

 1. Creates an AWS Identity and Access Management (IAM) role to use with Amazon
 EventBridge.
 2. Creates an Amazon Simple Storage Service (Amazon S3) bucket with EventBridge
 events enabled.
 3. Creates a rule that triggers when an object is uploaded to Amazon S3.
 4. Lists rules on the event bus.
 5. Creates a new Amazon Simple Notification Service (Amazon SNS) topic and lets
 the user subscribe to it.
 6. Adds a target to the rule that sends an email to the specified topic.
```

```
 7. Creates an EventBridge event that sends an email when an Amazon S3 object is
 created.
 8. Lists targets.
 9. Lists the rules for the same target.
 10. Triggers the rule by uploading a file to the S3 bucket.
 11. Disables a specific rule.
 12. Checks and prints the state of the rule.
 13. Adds a transform to the rule to change the text of the email.
 14. Enables a specific rule.
 15. Triggers the updated rule by uploading a file to the S3 bucket.
 16. Updates the rule to a custom rule pattern.
 17. Sends an event to trigger the rule.
 18. Cleans up resources.
*/
val DASHES: String = String(CharArray(80)).replace("\u0000", "-")
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <roleName> <bucketName> <topicName> <eventRuleName>

    Where:
        roleName - The name of the role to create.
        bucketName - The Amazon Simple Storage Service (Amazon S3) bucket name to
 create.
        topicName - The name of the Amazon Simple Notification Service (Amazon
 SNS) topic to create.
        eventRuleName - The Amazon EventBridge rule name to create.
    """
    val polJSON = "{" +
        "\"Version\": \"2012-10-17\"," +
        "\"Statement\": [{" +
        "\"Effect\": \"Allow\"," +
        "\"Principal\": {" +
        "\"Service\": \"events.amazonaws.com\"" +
        "}," +
        "\"Action\": \"sts:AssumeRole\"" +
        "}]" +
        "}"

    if (args.size != 4) {
        println(usage)
        exitProcess(1)
    }
```

```
    val sc = Scanner(System.`in`)
    val roleName = args[0]
    val bucketName = args[1]
    val topicName = args[2]
    val eventRuleName = args[3]

    println(DASHES)
    println("Welcome to the Amazon EventBridge example scenario.")
    println(DASHES)

    println(DASHES)
    println("1. Create an AWS Identity and Access Management (IAM) role to use
with Amazon EventBridge.")
    val roleArn = createIAMRole(roleName, polJSON)
    println(DASHES)

    println(DASHES)
    println("2. Create an S3 bucket with EventBridge events enabled.")
    if (checkBucket(bucketName)) {
        println("$bucketName already exists. Ending this scenario.")
        exitProcess(1)
    }

    createBucket(bucketName)
    delay(3000)
    setBucketNotification(bucketName)
    println(DASHES)

    println(DASHES)
    println("3. Create a rule that triggers when an object is uploaded to Amazon
S3.")
    delay(10000)
    addEventRule(roleArn, bucketName, eventRuleName)
    println(DASHES)

    println(DASHES)
    println("4. List rules on the event bus.")
    listRules()
    println(DASHES)

    println(DASHES)
    println("5. Create a new SNS topic for testing and let the user subscribe to
the topic.")
    val topicArn = createSnsTopic(topicName)
```

```
    println(DASHES)

    println(DASHES)
    println("6. Add a target to the rule that sends an email to the specified
topic.")
    println("Enter your email to subscribe to the Amazon SNS topic:")
    val email = sc.nextLine()
    subEmail(topicArn, email)
    println("Use the link in the email you received to confirm your subscription.
Then press Enter to continue.")
    sc.nextLine()
    println(DASHES)

    println(DASHES)
    println("7. Create an EventBridge event that sends an email when an Amazon S3
object is created.")
    addSnsEventRule(eventRuleName, topicArn, topicName, eventRuleName,
bucketName)
    println(DASHES)

    println(DASHES)
    println("8. List targets.")
    listTargets(eventRuleName)
    println(DASHES)

    println(DASHES)
    println(" 9. List the rules for the same target.")
    listTargetRules(topicArn)
    println(DASHES)

    println(DASHES)
    println("10. Trigger the rule by uploading a file to the S3 bucket.")
    println("Press Enter to continue.")
    sc.nextLine()
    uploadTextFiletoS3(bucketName)
    println(DASHES)

    println(DASHES)
    println("11. Disable a specific rule.")
    changeRuleState(eventRuleName, false)
    println(DASHES)

    println(DASHES)
    println("12. Check and print the state of the rule.")
```

```
    checkRule(eventRuleName)
    println(DASHES)

    println(DASHES)
    println("13. Add a transform to the rule to change the text of the email.")
    updateSnsEventRule(topicArn, eventRuleName)
    println(DASHES)

    println(DASHES)
    println("14. Enable a specific rule.")
    changeRuleState(eventRuleName, true)
    println(DASHES)

    println(DASHES)
    println("15. Trigger the updated rule by uploading a file to the S3 bucket.")
    println("Press Enter to continue.")
    sc.nextLine()
    uploadTextFiletoS3(bucketName)
    println(DASHES)

    println(DASHES)
    println("16. Update the rule to a custom rule pattern.")
    updateToCustomRule(eventRuleName)
    println("Updated event rule $eventRuleName to use a custom pattern.")
    updateCustomRuleTargetWithTransform(topicArn, eventRuleName)
    println("Updated event target $topicArn.")
    println(DASHES)

    println(DASHES)
    println("17. Send an event to trigger the rule. This will trigger a
subscription email.")
    triggerCustomRule(email)
    println("Events have been sent. Press Enter to continue.")
    sc.nextLine()
    println(DASHES)

    println(DASHES)
    println("18. Clean up resources.")
    println("Do you want to clean up resources (y/n)")
    val ans = sc.nextLine()
    if (ans.compareTo("y") == 0) {
        cleanupResources(topicArn, eventRuleName, bucketName, roleName)
    } else {
        println("The resources will not be cleaned up. ")
```

```
    }
    println(DASHES)

    println(DASHES)
    println("The Amazon EventBridge example scenario has successfully
 completed.")
    println(DASHES)
}

suspend fun cleanupResources(topicArn: String?, eventRuleName: String?,
 bucketName: String?, roleName: String?) {
    println("Removing all targets from the event rule.")
    deleteTargetsFromRule(eventRuleName)
    deleteRuleByName(eventRuleName)
    deleteSNSTopic(topicArn)
    deleteS3Bucket(bucketName)
    deleteRole(roleName)
}

suspend fun deleteRole(roleNameVal: String?) {
    val policyArnVal = "arn:aws:iam::aws:policy/AmazonEventBridgeFullAccess"
    val policyRequest = DetachRolePolicyRequest {
        policyArn = policyArnVal
        roleName = roleNameVal
    }
    IamClient { region = "us-east-1" }.use { iam ->
        iam.detachRolePolicy(policyRequest)
        println("Successfully detached policy $policyArnVal from role
 $roleNameVal")

        // Delete the role.
        val roleRequest = DeleteRoleRequest {
            roleName = roleNameVal
        }

        iam.deleteRole(roleRequest)
        println("*** Successfully deleted $roleNameVal")
    }
}

suspend fun deleteS3Bucket(bucketName: String?) {
    // Remove all the objects from the S3 bucket.
    val listObjects = ListObjectsRequest {
        bucket = bucketName
```

```
    }
    S3Client { region = "us-east-1" }.use { s3Client ->
        val res = s3Client.listObjects(listObjects)
        val myObjects = res.contents
        val toDelete = mutableListOf<ObjectIdentifier>()

        if (myObjects != null) {
            for (myValue in myObjects) {
                toDelete.add(
                    ObjectIdentifier {
                        key = myValue.key
                    }
                )
            }
        }

        val delOb = Delete {
            objects = toDelete
        }

        val dor = DeleteObjectsRequest {
            bucket = bucketName
            delete = delOb
        }
        s3Client.deleteObjects(dor)

        // Delete the S3 bucket.
        val deleteBucketRequest = DeleteBucketRequest {
            bucket = bucketName
        }
        s3Client.deleteBucket(deleteBucketRequest)
        println("You have deleted the bucket and the objects")
    }
}

// Delete the SNS topic.
suspend fun deleteSNSTopic(topicArnVal: String?) {
    val request = DeleteTopicRequest {
        topicArn = topicArnVal
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        snsClient.deleteTopic(request)
        println(" $topicArnVal was deleted.")
```

```
    }
}

suspend fun deleteRuleByName(ruleName: String?) {
    val ruleRequest = DeleteRuleRequest {
        name = ruleName
    }
    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.deleteRule(ruleRequest)
        println("Successfully deleted the rule")
    }
}

suspend fun deleteTargetsFromRule(eventRuleName: String?) {
    // First, get all targets that will be deleted.
    val request = ListTargetsByRuleRequest {
        rule = eventRuleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.listTargetsByRule(request)
        val allTargets = response.targets

        // Get all targets and delete them.
        if (allTargets != null) {
            for (myTarget in allTargets) {
                val removeTargetsRequest = RemoveTargetsRequest {
                    rule = eventRuleName
                    ids = listOf(myTarget.id.toString())
                }
                eventBrClient.removeTargets(removeTargetsRequest)
                println("Successfully removed the target")
            }
        }
    }
}

suspend fun triggerCustomRule(email: String) {
    val json = "{" +
        "\"UserEmail\": \"" + email + "\"," +
        "\"Message\": \"This event was generated by example code.\"" +
        "\"UtcTime\": \"Now.\"" +
        "}"
```

```
    val entry = PutEventsRequestEntry {
        source = "ExampleSource"
        detail = json
        detailType = "ExampleType"
    }

    val eventsRequest = PutEventsRequest {
        this.entries = listOf(entry)
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putEvents(eventsRequest)
    }
}

suspend fun updateCustomRuleTargetWithTransform(topicArn: String?, ruleName:
 String?) {
    val targetId = UUID.randomUUID().toString()

    val inputTransformerOb = InputTransformer {
        inputTemplate = "\"Notification: sample event was received.\""
    }

    val target = Target {
        id = targetId
        arn = topicArn
        inputTransformer = inputTransformerOb
    }

    val targetsRequest = PutTargetsRequest {
        rule = ruleName
        targets = listOf(target)
        eventBusName = null
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putTargets(targetsRequest)
    }
}

suspend fun updateToCustomRule(ruleName: String?) {
    val customEventsPattern = "{" +
        "\"source\": [\"ExampleSource\"]," +
        "\"detail-type\": [\"ExampleType\"]" +
```

```
        "}"
    val request = PutRuleRequest {
        name = ruleName
        description = "Custom test rule"
        eventPattern = customEventsPattern
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putRule(request)
    }
}

// Update an Amazon S3 object created rule with a transform on the target.
suspend fun updateSnsEventRule(topicArn: String?, ruleName: String?) {
    val targetId = UUID.randomUUID().toString()
    val myMap = mutableMapOf<String, String>()
    myMap["bucket"] = "$.detail.bucket.name"
    myMap["time"] = "$.time"

    val inputTransOb = InputTransformer {
        inputTemplate = "\"Notification: an object was uploaded to bucket
 <bucket> at <time>.\""
        inputPathsMap = myMap
    }
    val targetOb = Target {
        id = targetId
        arn = topicArn
        inputTransformer = inputTransOb
    }

    val targetsRequest = PutTargetsRequest {
        rule = ruleName
        targets = listOf(targetOb)
        eventBusName = null
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putTargets(targetsRequest)
    }
}

suspend fun checkRule(eventRuleName: String?) {
    val ruleRequest = DescribeRuleRequest {
        name = eventRuleName
```

```
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.describeRule(ruleRequest)
        println("The state of the rule is $response")
    }
}

suspend fun changeRuleState(eventRuleName: String, isEnabled: Boolean?) {
    if (!isEnabled!!) {
        println("Disabling the rule: $eventRuleName")
        val ruleRequest = DisableRuleRequest {
            name = eventRuleName
        }
        EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
            eventBrClient.disableRule(ruleRequest)
        }
    } else {
        println("Enabling the rule: $eventRuleName")
        val ruleRequest = EnableRuleRequest {
            name = eventRuleName
        }
        EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
            eventBrClient.enableRule(ruleRequest)
        }
    }
}

// Create and upload a file to an S3 bucket to trigger an event.
@Throws(IOException::class)
suspend fun uploadTextFiletoS3(bucketName: String?) {
    val fileSuffix = SimpleDateFormat("yyyyMMddHHmmss").format(Date())
    val fileName = "TextFile$fileSuffix.txt"
    val myFile = File(fileName)
    val fw = FileWriter(myFile.absoluteFile)
    val bw = BufferedWriter(fw)
    bw.write("This is a sample file for testing uploads.")
    bw.close()

    val putOb = PutObjectRequest {
        bucket = bucketName
        key = fileName
        body = myFile.asByteStream()
    }
```

```
    S3Client { region = "us-east-1" }.use { s3Client ->
        s3Client.putObject(putOb)
    }
}

suspend fun listTargetRules(topicArnVal: String?) {
    val ruleNamesByTargetRequest = ListRuleNamesByTargetRequest {
        targetArn = topicArnVal
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response =
 eventBrClient.listRuleNamesByTarget(ruleNamesByTargetRequest)
        response.ruleNames?.forEach { rule ->
            println("The rule name is $rule")
        }
    }
}

suspend fun listTargets(ruleName: String?) {
    val ruleRequest = ListTargetsByRuleRequest {
        rule = ruleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.listTargetsByRule(ruleRequest)
        response.targets?.forEach { target ->
            println("Target ARN: ${target.arn}")
        }
    }
}

// Add a rule that triggers an SNS target when a file is uploaded to an S3
 bucket.
suspend fun addSnsEventRule(ruleName: String?, topicArn: String?, topicName:
 String, eventRuleName: String, bucketName: String) {
    val targetID = UUID.randomUUID().toString()
    val myTarget = Target {
        id = targetID
        arn = topicArn
    }

    val targetsOb = mutableListOf<Target>()
```

```
    targetsOb.add(myTarget)

    val request = PutTargetsRequest {
        eventBusName = null
        targets = targetsOb
        rule = ruleName
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        eventBrClient.putTargets(request)
        println("Added event rule $eventRuleName with Amazon SNS target
 $topicName for bucket $bucketName.")
    }
}

suspend fun subEmail(topicArnVal: String?, email: String?) {
    val request = SubscribeRequest {
        protocol = "email"
        endpoint = email
        returnSubscriptionArn = true
        topicArn = topicArnVal
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        val result = snsClient.subscribe(request)
        println(" Subscription ARN: ${result.subscriptionArn}")
    }
}

suspend fun createSnsTopic(topicName: String): String? {
    val topicPolicy = "{" +
        "\"Version\": \"2012-10-17\"," +
        "\"Statement\": [{" +
        "\"Sid\": \"EventBridgePublishTopic\"," +
        "\"Effect\": \"Allow\"," +
        "\"Principal\": {" +
        "\"Service\": \"events.amazonaws.com\"" +
        "}," +
        "\"Resource\": \"*\"," +
        "\"Action\": \"sns:Publish\"" +
        "}]" +
        "}"

    val topicAttributes = mutableMapOf<String, String>()
```

```
    topicAttributes["Policy"] = topicPolicy

    val topicRequest = CreateTopicRequest {
        name = topicName
        attributes = topicAttributes
    }

    SnsClient { region = "us-east-1" }.use { snsClient ->
        val response = snsClient.createTopic(topicRequest)
        println("Added topic $topicName for email subscriptions.")
        return response.topicArn
    }
}

suspend fun listRules() {
    val rulesRequest = ListRulesRequest {
        eventBusName = "default"
        limit = 10
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val response = eventBrClient.listRules(rulesRequest)
        response.rules?.forEach { rule ->
            println("The rule name is ${rule.name}")
            println("The rule ARN is ${rule.arn}")
        }
    }
}

// Create a new event rule that triggers when an Amazon S3 object is created in a
 bucket.
suspend fun addEventRule(roleArnVal: String?, bucketName: String, eventRuleName:
 String?) {
    val pattern = """{
        "source": ["aws.s3"],
        "detail-type": ["Object Created"],
        "detail": {
        "bucket": {
            "name": ["$bucketName"]
            }
        }
    }"""

    val ruleRequest = PutRuleRequest {
```

```
        description = "Created by using the AWS SDK for Kotlin"
        name = eventRuleName
        eventPattern = pattern
        roleArn = roleArnVal
    }

    EventBridgeClient { region = "us-east-1" }.use { eventBrClient ->
        val ruleResponse = eventBrClient.putRule(ruleRequest)
        println("The ARN of the new rule is ${ruleResponse.ruleArn}")
    }
}

// Set the Amazon S3 bucket notification configuration.
suspend fun setBucketNotification(bucketName: String) {
    val eventBridgeConfig = EventBridgeConfiguration {
    }

    val configuration = NotificationConfiguration {
        eventBridgeConfiguration = eventBridgeConfig
    }

    val configurationRequest = PutBucketNotificationConfigurationRequest {
        bucket = bucketName
        notificationConfiguration = configuration
        skipDestinationValidation = true
    }

    S3Client { region = "us-east-1" }.use { s3Client ->
        s3Client.putBucketNotificationConfiguration(configurationRequest)
        println("Added bucket $bucketName with EventBridge events enabled.")
    }
}

// Create an S3 bucket using a waiter.
suspend fun createBucket(bucketName: String) {
    val request = CreateBucketRequest {
        bucket = bucketName
    }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.createBucket(request)
        s3.waitUntilBucketExists {
            bucket = bucketName
        }
```

```
        println("$bucketName is ready")
    }
}

suspend fun checkBucket(bucketName: String?): Boolean {
    try {
        // Determine if the S3 bucket exists.
        val headBucketRequest = HeadBucketRequest {
            bucket = bucketName
        }

        S3Client { region = "us-east-1" }.use { s3Client ->
            s3Client.headBucket(headBucketRequest)
            return true
        }
    } catch (e: S3Exception) {
        System.err.println(e.message)
    }
    return false
}

suspend fun createIAMRole(rolenameVal: String?, polJSON: String?): String? {
    val request = CreateRoleRequest {
        roleName = rolenameVal
        assumeRolePolicyDocument = polJSON
        description = "Created using the AWS SDK for Kotlin"
    }

    val rolePolicyRequest = AttachRolePolicyRequest {
        roleName = rolenameVal
        policyArn = "arn:aws:iam::aws:policy/AmazonEventBridgeFullAccess"
    }

    IamClient { region = "us-east-1" }.use { iam ->
        val response = iam.createRole(request)
        iam.attachRolePolicy(rolePolicyRequest)
        return response.role?.arn
    }
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Kotlin.

- DeleteRule
- DescribeRule
- DisableRule
- EnableRule
- ListRuleNamesByTarget
- ListRules
- ListTargetsByRule
- PutEvents
- PutRule
- PutTargets

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Serviceübergreifende Beispiele für die EventBridge Verwendung von SDKs AWS

Die folgenden Beispielanwendungen verwenden AWS SDKs zur Kombination EventBridge mit anderen. AWS-Services Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen der Anwendung finden.

Beispiele

- Verwendung geplanter Ereignisse zum Aufrufen einer Lambda-Funktion

Verwendung geplanter Ereignisse zum Aufrufen einer Lambda-Funktion

Die folgenden Codebeispiele zeigen, wie eine AWS Lambda Funktion erstellt wird, die durch ein von Amazon EventBridge geplantes Ereignis aufgerufen wird.

Java

SDK für Java 2.x

Zeigt, wie ein von Amazon EventBridge geplantes Ereignis erstellt wird, das eine AWS Lambda Funktion aufruft. Konfigurieren Sie so EventBridge , dass ein Cron-Ausdruck verwendet wird, um zu planen, wann die Lambda-Funktion aufgerufen wird. In diesem Beispiel erstellen Sie eine Lambda-Funktion mithilfe der Lambda-Java-Laufzeit-API. In diesem Beispiel werden verschiedene AWS Dienste aufgerufen, um einen bestimmten Anwendungsfall auszuführen. Dieses Beispiel zeigt, wie man eine App erstellt, die eine mobile Textnachricht an Ihre Mitarbeiter sendet, um ihnen zum einjährigen Jubiläum zu gratulieren.

Den vollständigen Quellcode und Anweisungen zur Einrichtung und Ausführung finden Sie im vollständigen Beispiel unter GitHub.

In diesem Beispiel verwendete Dienste

- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

JavaScript

SDK für JavaScript (v3)

Zeigt, wie ein von Amazon EventBridge geplantes Ereignis erstellt wird, das eine AWS Lambda Funktion aufruft. Konfigurieren Sie so EventBridge , dass ein Cron-Ausdruck verwendet wird, um zu planen, wann die Lambda-Funktion aufgerufen wird. In diesem Beispiel erstellen Sie eine Lambda-Funktion mithilfe der JavaScript Lambda-Laufzeit-API. In diesem Beispiel werden verschiedene AWS Dienste aufgerufen, um einen bestimmten Anwendungsfall auszuführen. Dieses Beispiel zeigt, wie man eine App erstellt, die eine mobile Textnachricht an Ihre Mitarbeiter sendet, um ihnen zum einjährigen Jubiläum zu gratulieren.

Den vollständigen Quellcode und Anweisungen zur Einrichtung und Ausführung finden Sie im vollständigen Beispiel unter GitHub.

Dieses Beispiel ist auch verfügbar im AWS SDK for JavaScript Entwicklerhandbuch für v3.

In diesem Beispiel verwendete Dienste

- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

Python

SDK für Python (Boto3)

Dieses Beispiel zeigt, wie eine AWS Lambda Funktion als Ziel einer geplanten EventBridge Amazon-Veranstaltung registriert wird. Der Lambda-Handler schreibt eine freundliche Nachricht und die vollständigen Ereignisdaten für den späteren Abruf in Amazon CloudWatch Logs.

- Stellt eine Lambda-Funktion bereit.
- Erzeugt ein EventBridge geplantes Ereignis und macht die Lambda-Funktion zum Ziel.
- Erteilt die Erlaubnis, die EventBridge Lambda-Funktion aufrufen zu lassen.
- Druckt die neuesten Daten aus CloudWatch Logs, um das Ergebnis der geplanten Aufrufe anzuzeigen.
- Bereinigt alle Ressourcen, die während der Demo erstellt wurden.

Dieses Beispiel lässt sich am besten auf ansehen. GitHub Den vollständigen Quellcode und Anweisungen zur Einrichtung und Ausführung finden Sie im vollständigen Beispiel unter GitHub.

In diesem Beispiel verwendete Dienste

- CloudWatch Logs
- EventBridge
- Lambda

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwendung EventBridge mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Amazon EventBridge Sicherheit

Amazon EventBridge verwendet AWS Identity and Access Management , um den Zugriff auf andere AWS Dienste und Ressourcen zu kontrollieren. Eine Übersicht der Funktionsweise von IAM finden Sie unter Übersicht über die Zugriffsverwaltung im IAM-Benutzerhandbuch. Eine Übersicht der Sicherheitsanmeldeinformationen finden Sie unter AWS -Sicherheitsanmeldeinformationen in der Allgemeine Amazon Web Services-Referenz.

Themen

Datenschutz bei Amazon EventBridge

Das Modell der AWS gemeinsamen Verantwortung und geteilter Verantwortung gilt für den Datenschutz in Amazon EventBridge. Wie in diesem Modell beschrieben, AWS ist verantwortlich für den Schutz der globalen Infrastruktur, auf der alle Systeme laufen AWS Cloud. Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Sie sind auch für die Sicherheitskonfiguration und die Verwaltungsaufgaben für die von Ihnen verwendeten AWS-Services verantwortlich. Weitere Informationen zum Datenschutz finden Sie unter Häufig gestellte Fragen zum Datenschutz. Informationen zum Datenschutz in Europa finden Sie im Blog-Beitrag AWS -Modell der geteilten Verantwortung und in der DSGVO im AWS -Sicherheitsblog.

Aus Datenschutzgründen empfehlen wir, dass Sie AWS-Konto Anmeldeinformationen schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einrichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem empfehlen wir, die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- 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 der Konsole, der API EventBridge oder den SDKs arbeiten oder diese anderweitig AWS-Services verwenden. AWS CLI AWS 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.

Datenverschlüsselung für EventBridge Event-Busse

EventBridge bietet sowohl Verschlüsselung im Ruhezustand als auch Verschlüsselung bei der Übertragung, um Ihre Veranstaltungsdaten zu schützen:

- Verschlüsselung im Ruhezustand

  EventBridge lässt sich in AWS Key Management Service (KMS) integrieren, um auf Event-Bussen gespeicherte Ereignisdaten zu verschlüsseln. EventBridge Verwendet standardmäßig an, um AWS-eigener Schlüssel Ereignisdaten zu verschlüsseln. Sie können auch angeben EventBridge , dass stattdessen a Kundenverwalteter Schlüssel für benutzerdefinierte Ereignisse und Partnerereignisse verwendet werden soll.
- Verschlüsselung während der Übertragung

  EventBridge verschlüsselt Daten, die zwischen EventBridge und anderen Diensten übertragen werden, mithilfe von Transport Layer Security (TLS). Bei Ereignisbussen umfasst dies sowohl den Zeitpunkt, an den ein Ereignis gesendet wird EventBridge, als auch den Zeitpunkt, an den ein Ereignis an ein Regelziel EventBridge gesendet wird.

Verschlüsselung im Ruhezustand für Eventbusse

EventBridge bietet transparente serverseitige Verschlüsselung durch Integration mit AWS Key Management Service (KMS). Die standardmäßige Verschlüsselung von Daten im Ruhezustand trägt dazu bei, den betrieblichen Aufwand und die Komplexität zu reduzieren, die mit dem Schutz vertraulicher Daten verbunden sind. Gleichzeitig können Sie damit sichere Anwendungen erstellen, die strenge Verschlüsselungsvorschriften und gesetzliche Auflagen erfüllen.

Zu den Daten im Ereignisbus, die im Ruhezustand EventBridge verschlüsselt werden, gehören:

- Ereignisdaten für AWSbenutzerdefinierte Veranstaltungen und Partnerereignisse.

  Bei Event-Bussen umfassen die Ereignisdaten alle Felder, die im ??? Element der Veranstaltung enthalten sind.

  EventBridge verschlüsselt keine Event-Metadaten. Weitere Informationen zu Ereignismetadaten finden Sie unter???.

- Ereignismuster
- Eingangstransformatoren

EventBridge Verwendet standardmäßig an, AWS-eigener Schlüssel um Ereignisdaten zu verschlüsseln. Sie können auch angeben EventBridge , dass stattdessen a Kundenverwalteter Schlüssel für benutzerdefinierte Ereignisse und Partnerereignisse verwendet werden soll.

Sicherheitsüberlegungen zur Event-Bus-Verschlüsselung

Es wird dringend empfohlen, niemals vertrauliche oder sensible Informationen in die folgenden Felder einzugeben, da diese im Speicher nicht verschlüsselt werden:

- Namen der Event-Busse
- Namen der Regeln
- Gemeinsam genutzte Ressourcen wie z. B. Tags

KMS key Optionen für die Event-Bus-Verschlüsselung

EventBridge verwendet an AWS-eigener Schlüssel , um AWS Dienstereignisse zu verschlüsseln, die auf Event-Bussen gespeichert sind.

Für jeden Ereignisbus können Sie die Art der KMS key EventBridge Verwendung auswählen, mit der benutzerdefinierte Ereignisse und Partnerereignisse, die auf diesem Bus gespeichert sind, verschlüsselt werden sollen:

- AWS-eigener Schlüssel

  EventBridge Verschlüsselt Daten standardmäßig mit dem 256-Bit-Advanced Encryption Standard (AES-256) unter einem AWS-eigener Schlüssel, wodurch Ihre Daten vor unbefugtem Zugriff geschützt werden.

  Sie können ihre Verwendung nicht einsehen, verwalten AWS-eigene Schlüssel, verwenden oder überprüfen. Sie müssen jedoch keine Maßnahmen ergreifen oder Programme zum Schutz der Schlüssel ändern, die zur Verschlüsselung Ihrer Daten verwendet werden.

  Generell gilt: Sofern Sie nicht verpflichtet sind, den Verschlüsselungsschlüssel, der Ihre Ressourcen schützt, zu überprüfen oder zu kontrollieren, AWS-eigener Schlüssel ist an eine gute Wahl. AWS-eigene Schlüssel sind völlig kostenlos (keine monatlichen Gebühren oder

Nutzungsgebühren) und werden nicht auf die AWS KMS Kontingente für Ihr Konto angerechnet. Sie müssen den Schlüssel oder seine Schlüsselrichtlinie nicht erstellen oder pflegen.

Weitere Informationen finden Sie unter AWS -eigene Schlüssel im AWS Key Management Service -Entwicklerhandbuch.

- Kundenverwalteter Schlüssel

  EventBridge unterstützt die Verwendung einer Symmetrie Kundenverwalteter Schlüssel , die Sie selbst erstellen, besitzen und verwalten. Da Sie die volle Kontrolle über diesen Typ von haben KMS key, können Sie beispielsweise folgende Aufgaben ausführen:

  - Festlegung und Pflege wichtiger Richtlinien
  - Festlegung und Aufrechterhaltung von IAM-Richtlinien und -Zuschüssen
  - Aktivieren und Deaktivieren wichtiger Richtlinien
  - Kryptographisches Material mit rotierendem Schlüssel
  - Hinzufügen von Tags
  - Erstellen von Schlüsselaliasen
  - Schlüssel für das Löschen von Schlüsseln planen

  Weitere Informationen finden Sie unter Kundenverwaltete Schlüssel im AWS Key Management Service -Entwicklerhandbuch.

  EventBridge unterstützt Schlüssel mit mehreren Regionen und den kontoübergreifenden Zugriff auf Schlüssel.

  Kundenverwaltete Schlüssel fällt eine monatliche Gebühr an. Einzelheiten finden Sie unter AWS Key Management Service Preise und Kontingente im AWS Key Management Service Entwicklerhandbuch.

  > **Note**
  >
  > EventBridge unterstützt die folgenden Funktionen nicht auf Ereignisbussen, die mit verschlüsselt wurden Kundenverwaltete Schlüssel:
  >
  > - Archive
  > - Entdeckung von Schemas

Weitere Informationen finden Sie unter ???.

Ereignisse verschlüsseln mit Kundenverwaltete Schlüssel

Sie können angeben, dass EventBridge Ihre Daten (benutzerdefinierte Ereignisse und Partnerereignisse), die auf einem Event-Bus gespeichert sind, mit a AWS KMS Kundenverwalteter Schlüssel verschlüsselt werden, und nicht, AWS-eigener Schlüssel wie es die Standardeinstellung ist. Sie können einen angeben Kundenverwalteter Schlüssel , wenn Sie einen Event-Bus erstellen oder aktualisieren. Sie können den Standard-Event-Bus auch so aktualisieren, dass er auch Kundenverwalteter Schlüssel für benutzerdefinierte Veranstaltungen und Partnerveranstaltungen verwendet wird. Weitere Informationen finden Sie unter ???.

Wenn Sie a Kundenverwalteter Schlüssel für einen Event-Bus angeben, haben Sie die Möglichkeit, eine Dead-Letter-Queue (DLQ) für den Event-Bus anzugeben. EventBridge übermittelt dann alle benutzerdefinierten Ereignisse oder Partnerereignisse, die zu Verschlüsselungs- oder Entschlüsselungsfehlern führen, an diesen DLQ. Weitere Informationen finden Sie unter ???.

Geben Sie beim Erstellen eines Event-Busses (mithilfe der Konsole) einen Kundenverwalteter Schlüssel für die Verschlüsselung an

- Folgen Sie diesen Anweisungen:

  ???.

Angabe von a Kundenverwalteter Schlüssel für die Verschlüsselung beim Erstellen eines Event-Busses (mit der CLI)

- Verwenden Sie beim Aufrufen die `kms-key-identifier` Option`create-event-bus`, um den Kundenverwalteter Schlüssel für die Verschlüsselung auf dem Event-Bus EventBridge zu verwenden.

  Verwenden Sie diese Option optional, `dead-letter-config` um eine Warteschlange mit unerlaubten Briefen (DLQ) anzugeben.

Aktualisierung eines Event-Busses zur Verwendung von A Kundenverwalteter Schlüssel für die Verschlüsselung (mithilfe der Konsole)

- Folgen Sie diesen Anweisungen:

  ???.

Aktualisierung eines Event-Busses zur Verwendung von a Kundenverwalteter Schlüssel für die Verschlüsselung (mithilfe der CLI)

- Verwenden Sie beim Aufrufen die `kms-key-identifier` Option`update-event-bus`, um das Kundenverwalteter Schlüssel für die Verschlüsselung auf dem Event-Bus EventBridge zu verwenden.

  Verwenden Sie diese Option optional, `dead-letter-config` um eine Warteschlange mit unerlaubten Briefen (DLQ) anzugeben.

Aktualisierung des Standardereignisbusses zur Verwendung von a Kundenverwalteter Schlüssel für die Verschlüsselung mit CloudFormation

Da EventBridge der Standardereignisbus Ihrem Konto automatisch zugewiesen wird, können Sie ihn nicht mithilfe einer CloudFormation Vorlage erstellen, wie Sie es normalerweise für Ressourcen tun würden, die Sie in einen CloudFormation Stapel aufnehmen möchten. Um den Standard-Event-Bus in einen CloudFormation Stack aufzunehmen, müssen Sie ihn zuerst in einen Stack importieren. Nachdem Sie den Standard-Event-Bus in einen Stack importiert haben, können Sie die Eigenschaften des Event-Busses nach Bedarf aktualisieren.

- Folgen Sie diesen Anweisungen:

  ???.

Autorisierung EventBridge zur Verwendung eines Kundenverwalteter Schlüssel

Wenn Sie einen Kundenverwalteter Schlüssel in Ihrem Konto verwenden, um Ihren EventBridge Eventbus zu schützen, KMS key müssen die entsprechenden Richtlinien die EventBridge Erlaubnis enthalten, ihn in Ihrem Namen zu verwenden. Sie geben diese Berechtigungen in einer wichtigen Richtlinie an.

EventBridge benötigt keine zusätzliche Autorisierung, um die Standardeinstellung AWS-eigener Schlüssel zum Schutz der EventBridge Ressourcen in Ihrem AWS Konto zu verwenden.

EventBridge erfordert die folgenden Berechtigungen für eine Kundenverwaltete Schlüssel:

- `kms:DescribeKey`

  EventBridge benötigt diese Berechtigung, um den KMS key ARN für die angegebene Schlüssel-ID abzurufen und zu überprüfen, ob der Schlüssel symmetrisch ist.

- `kms:GenerateDataKey`

  EventBridge benötigt diese Berechtigung, um einen Datenschlüssel als Verschlüsselungsschlüssel für die Ereignisdaten zu generieren.

- `kms:Decrypt`

  EventBridge benötigt diese Berechtigung, um den Datenschlüssel zu entschlüsseln, der verschlüsselt und zusammen mit den verschlüsselten Ereignisdaten gespeichert ist.

  EventBridge verwendet dies für den Regelabgleich; Benutzer haben nie Zugriff auf die Daten.

Die folgende Beispiel-Schlüsselrichtlinie stellt die erforderlichen Berechtigungen bereit:

```
{
  "Sid": "Allow EventBridge to encrypt events",
  "Effect": "Allow",
  "Principal": {
    "Service": "events.amazonaws.com"
  },
  "Action": [
    "kms:DescribeKey",
    "kms:GenerateDataKey",
    "kms:Decrypt"
  ]
  "Resource": "*",
  "Condition": {
    "StringEquals": {
        "kms:EncryptionContext:aws:events:event-bus:arn":
 "arn:aws:events:region:account-id:event-bus/event-bus-arn",
        "aws:SourceArn": "arn:aws:events:region:account-id:event-bus/event-bus-name"
    }
  }
```

```
}
```

Sicherheit bei der Verwendung der Kundenverwaltete SchlüsselEventBridge Event-Bus-Verschlüsselung

Aus Sicherheitsgründen empfiehlt es sich, der Schlüsselrichtlinie einen `aws:SourceArnaws:sourceAccount`,, oder `kms:EncryptionContext:aws:events:event-bus:arn` AWS KMS Bedingungsschlüssel hinzuzufügen. Der IAM globale Bedingungsschlüssel trägt dazu bei, dass der KMS-Schlüssel nur für den angegebenen Bus oder das angegebene Konto EventBridge verwendet wird.

Das folgende Beispiel zeigt, wie Sie diese bewährte Methode in Ihrer IAM Richtlinie anwenden können:

```
{
      "Sid": "Allow the use of key",
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource": "*",
      "Condition" : {
        "StringEquals": {
          "aws:SourceAccount": "arn:aws:events:region:account-id",
          "aws:SourceArn": "arn:aws:events:region:account-id:event-bus/event-bus-name",
          "kms:EncryptionContext:aws:events:event-bus:arn":
 "arn:aws:events:region:account-id:event-bus/event-bus-arn"
        }
      }
```

Verwaltung der Kundenverwaltete SchlüsselEventBridge Event-Bus-Verschlüsselung

Um sicherzustellen, dass EventBridge immer der Zugriff auf das Notwendige erhalten bleibt Kundenverwalteter Schlüssel:

- Löschen Sie eine nicht, Kundenverwalteter Schlüssel bis Sie sicher sind, dass alle damit verschlüsselten Ereignisse verarbeitet wurden.

Wenn Sie einen der folgenden Vorgänge ausführen, bewahren Sie das vorherige Schlüsselmaterial auf, um sicherzustellen, dass Sie es weiterhin für zuvor verschlüsselte Ereignisse verwenden EventBridge können:

- Automatische Schlüsselrotation
- Manuelle Schlüsselrotation
- Einen Schlüsselalias aktualisieren

Generell gilt: Wenn Sie erwägen, einen AWS KMS Schlüssel zu löschen, deaktivieren Sie ihn zunächst und stellen Sie einen CloudWatch Alarm oder einen ähnlichen Mechanismus ein, um sicherzustellen, dass Sie den Schlüssel niemals zum Entschlüsseln verschlüsselter Daten verwenden müssen.

- Löschen Sie nicht die Schlüsselrichtlinie, die die Berechtigungen zur Verwendung EventBridge des Schlüssels bereitstellt.

Zu den weiteren Überlegungen gehören:

- Geben Sie je Kundenverwaltete Schlüssel nach Bedarf Regelziele an.

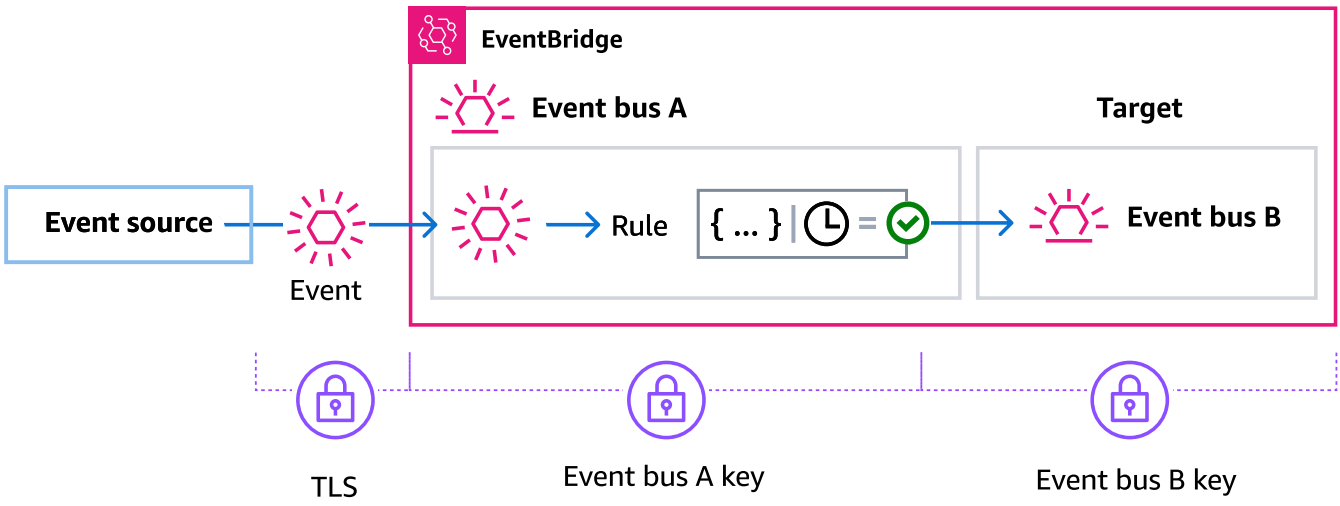
  Wenn ein Ereignis EventBridge an ein Regelziel gesendet wird, wird das Ereignis mithilfe von Transport Layer Security (TLS) gesendet. Welche Verschlüsselung auf das Ereignis angewendet wird, da es auf dem Ziel gespeichert ist, hängt jedoch von der Verschlüsselung ab, die Sie auf dem Ziel selbst konfiguriert haben.

Ereignisverschlüsselung, wenn ein Ereignisbus das Regelziel ist

Wenn ein benutzerdefiniertes Ereignis oder ein Partnerereignis an einen Eventbus gesendet wird, wird dieses Ereignis entsprechend der KMS-Schlüsselkonfiguration für Verschlüsselung im Ruhezustand für diesen Eventbus EventBridge verschlüsselt — entweder die Standardeinstellung AWS-eigener Schlüssel oder ein Kundenverwalteter Schlüssel, falls ein solches angegeben wurde. Wenn ein Ereignis mit einer Regel übereinstimmt, wird das Ereignis mit der KMS-Schlüsselkonfiguration für diesen Ereignisbus EventBridge verschlüsselt, bis das Ereignis an das Regelziel gesendet wird, sofern es sich bei dem Regelziel nicht um einen anderen Ereignisbus handelt.

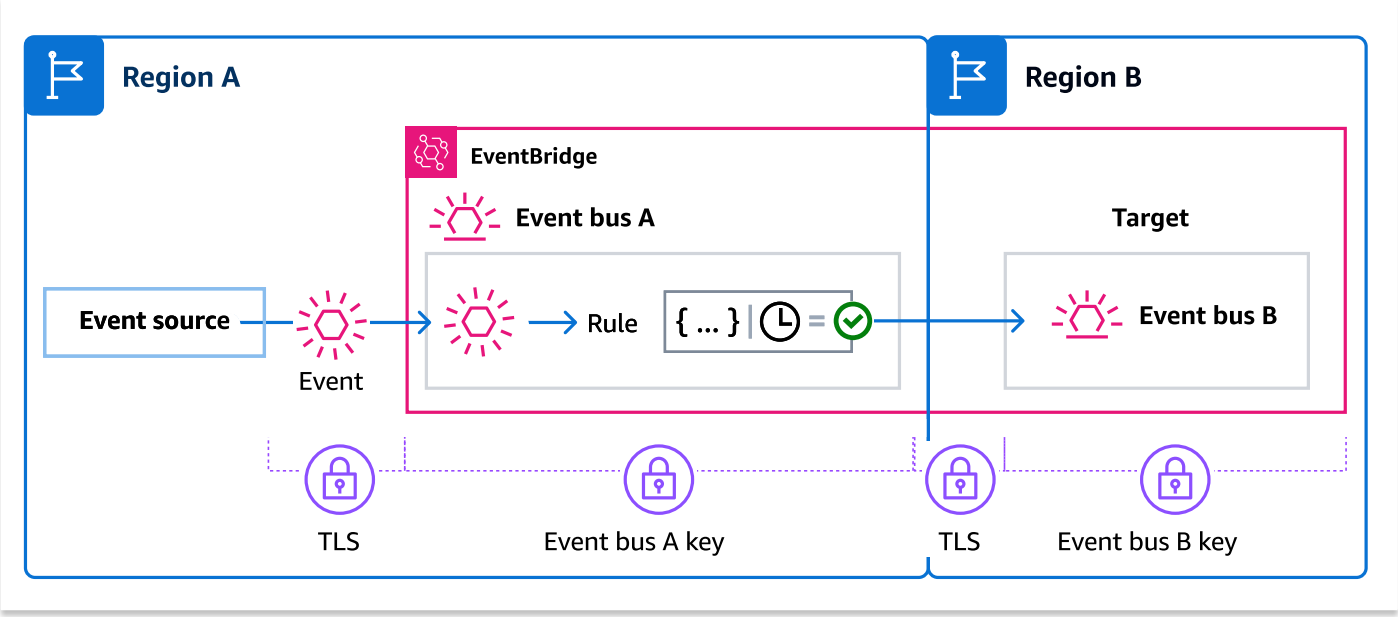
- Wenn das Ziel einer Regel ein anderer Event-Bus in derselben AWS Region ist:

Wenn für den Ziel-Event-Bus ein bestimmter Wert angegeben ist Kundenverwalteter Schlüssel, wird das Ereignis stattdessen mit dem Kundenverwalteter Schlüssel des Ziel-Event-Busses für die Zustellung EventBridge verschlüsselt.



- Wenn das Ziel einer Regel ein anderer Event-Bus in einer anderen AWS Region ist:

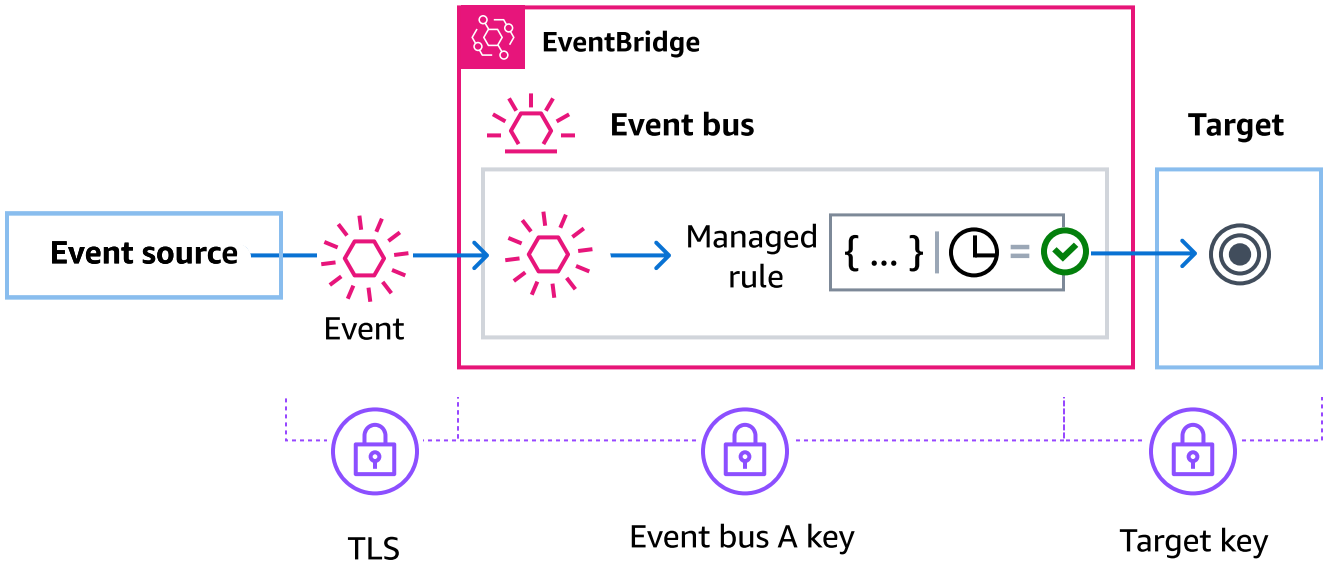
  EventBridge verschlüsselt das Ereignis im Ruhezustand gemäß der KMS-Schlüsselkonfiguration auf dem ersten Ereignisbus. EventBridge verwendet TLS, um das Ereignis an den zweiten Event-Bus in der anderen Region zu senden, wo es dann entsprechend der für den Ziel-Event-Bus angegebenen KMS-Schlüsselkonfiguration verschlüsselt wird.



Ereignisverschlüsselung für verwaltete Regeln

AWS Dienste können in Ihrem AWS Konto Event-Bus-Regeln erstellen und verwalten, die für bestimmte Funktionen in diesen Diensten benötigt werden. Im Rahmen einer verwalteten Regel kann der AWS Dienst angeben, dass das für die Regel Kundenverwalteter Schlüssel angegebene Ziel EventBridge verwendet wird. Auf diese Weise können Sie auf der Grundlage des Regelziels flexibel angeben, welche Option verwendet werden Kundenverwalteter Schlüssel soll.

In diesen Fällen wird, sobald ein benutzerdefiniertes Ereignis oder ein Partnerereignis mit der verwalteten Regel übereinstimmt, das in der verwalteten Regel Kundenverwalteter Schlüssel angegebene Ziel EventBridge verwendet, um das Ereignis zu verschlüsseln, bis es an das Regelziel gesendet wird. Dies gilt unabhängig davon, ob der Event-Bus so konfiguriert wurde, dass er seinen eigenen Bus Kundenverwalteter Schlüssel für die Verschlüsselung verwendet. Dies ist auch dann der Fall, wenn das Ziel der verwalteten Regel ein anderer Ereignisbus ist und für diesen Ereignisbus ein eigener für die Verschlüsselung Kundenverwalteter Schlüssel spezifiziert wurde. EventBridge verwendet weiterhin das in der verwalteten Regel Kundenverwalteter Schlüssel angegebene Ziel, bis das Ereignis an ein Ziel gesendet wird, bei dem es sich nicht um einen Ereignisbus handelt.



In Fällen, in denen das Regelziel ein Event-Bus in einer anderen Region ist, müssen Sie einen Schlüssel für mehrere Regionen angeben. Der Ereignisbus in der ersten Region verschlüsselt das Ereignis mit dem in der verwalteten Kundenverwalteter Schlüssel Regel angegebenen Wert. Anschließend sendet er das Ereignis an den Ziel-Event-Bus in der zweiten Region. Dieser

Ereignisbus muss in der Lage sein, den weiter zu verwenden, Kundenverwalteter Schlüssel bis er das Ereignis an sein Ziel sendet.

EventBridge Kontext der Ereignisbus-Verschlüsselung

Ein Verschlüsselungskontext ist eine Gruppe von Schlüssel/Wert-Paaren mit willkürlichen, nicht geheimen Daten. Wenn Sie einen Verschlüsselungskontext in eine Anforderung zur Verschlüsselung von Daten aufnehmen, bindet AWS KMS den Verschlüsselungskontext kryptografisch an die verschlüsselten Daten. Zur Entschlüsselung der Daten müssen Sie denselben Verschlüsselungskontext übergeben.

Sie können den Verschlüsselungskontext auch als Bedingung für die Autorisierung in Richtlinien und Zuschüssen verwenden.

EventBridge Verwendet für Ereignisbusse bei allen AWS KMS kryptografischen Vorgängen denselben Verschlüsselungskontext. Wenn Sie zum Schutz Ihrer EventBridge Ressourcen einen vom Kunden verwalteten Schlüssel verwenden, können Sie anhand des Verschlüsselungskontextes die Verwendung dieses Schlüssels KMS key in Prüfaufzeichnungen und Protokollen identifizieren. Er wird auch im Klartext in Protokollen wie AWS CloudTrail und Amazon CloudWatch Logs angezeigt.

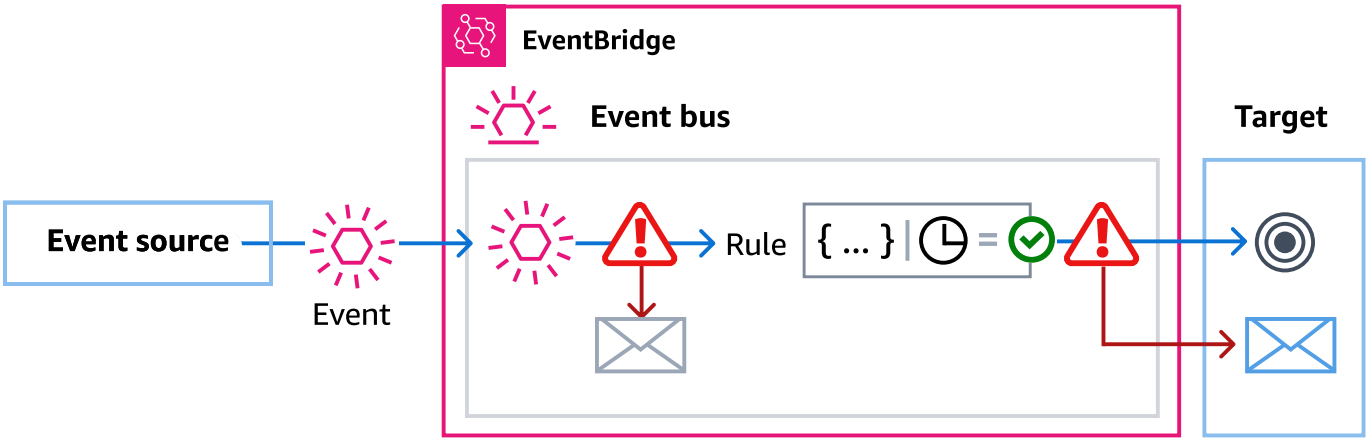
EventBridge Verwendet in seinen Anfragen an AWS KMS einen Verschlüsselungskontext mit einem einzigen Schlüssel-Wert-Paar, das den Event-Bus-ARN enthält:

```
"encryptionContext": {
    "kms:EncryptionContext:aws:events:event-bus:arn": "event-bus-arn"
}
```

Verwendung von Warteschlangen mit unverschlüsselten Buchstaben zur Erfassung verschlüsselter Ereignisfehler

Wenn Sie die Kundenverwalteter Schlüssel Verschlüsselung auf einem Event-Bus konfigurieren, empfehlen wir, eine Dead-Letter-Queue (DLQ) für diesen Event-Bus anzugeben. EventBridge sendet benutzerdefinierte Ereignisse und Partnerereignisse an diesen DLQ, wenn bei der Verarbeitung des Ereignisses auf dem Event-Bus ein Fehler auftritt, der nicht behoben werden kann. Ein nicht behebbarer Fehler ist ein Fehler, bei dem ein Benutzereingriff erforderlich ist, um das zugrundeliegende Problem zu beheben, z. B. wenn der angegebene Kundenverwalteter Schlüssel Fehler deaktiviert ist oder fehlt.

- Wenn während EventBridge der Verarbeitung des Ereignisses auf dem Event-Bus ein nicht wiederherstellbarer Verschlüsselungs- oder Entschlüsselungsfehler auftritt, wird das Ereignis an den DLQ für den Event-Bus gesendet, sofern einer angegeben wurde.
- Wenn beim EventBridge Versuch, das Ereignis an ein Ziel zu senden, ein nicht wiederherstellbarer Verschlüsselungs- oder Entschlüsselungsfehler auftritt, wird das Ereignis an den DLQ für das Ziel gesendet, sofern einer angegeben wurde.



Weitere Informationen, einschließlich Überlegungen zur Verwendung von DLQs und Anweisungen zum Einstellen von Berechtigungen, finden Sie unter. ???

Ereignisse in EventBridge Warteschlangen mit unzustellbaren Nachrichten entschlüsseln

Sobald Sie das zugrundeliegende Problem behoben haben, das einen Fehler verursacht, der nicht mehr rückgängig gemacht werden kann, können Sie die Ereignisse verarbeiten, die an den Event-Bus oder die Ziel-DLQs gesendet wurden. Bei verschlüsselten Ereignissen müssen Sie das Ereignis zuerst entschlüsseln, um es verarbeiten zu können.

Das folgende Beispiel zeigt, wie ein Ereignis entschlüsselt wird, das an einen Ereignisbus oder eine Ziel-DLQ gesendet EventBridge wurde.

```
// You will receive an encrypted event in the following json format.
        // ```
        //   {
        //     "version": "0",
        //     "id": "053afa53-cdd7-285b-e754-b0dfd0ac0bfb",  // New event id not the
 same as the original one
        //     "account": "123456789012",
        //     "time": "2020-02-10T10:22:00Z",
```

```
        //     "resources": [ ],
        //     "region": "us-east-1",
        //     "source": "aws.events",
        //     "detail-type": "Encrypted Events",
        //     "detail": {
        //       "event-bus-arn": "arn:aws:events:region:account:event-bus/bus-name",
        //       "rule-arn": "arn:aws:events:region:account:event-bus/bus-name/rule-
name",
        //       "kms-key-arn": "arn:aws:kms:region:account:key/key-arn",
        //       "encrypted-payload": "AgR4qiru/XNwTUyCgRHqP7rbbHn/
xpmVeVeRIAd12TDYYVwAawABABRhd3M6ZXZlbnRzOmV2ZW50LWJ1cwB
        //
 RYXJuOmF3czpldmVudHM6dXMtZWFzdC0xOjE0NjY4NjkwNDY3MzpldmVudC1idXMvY21rbXMtZ2EtY3Jvc3
        //
 MtYWNjb3VudC1zb3VyY2UtYnVzAAEAB2F3cy1rbXMAS2Fybjphd3M6a21zOnVzLWVhc3QtMToxNDY2ODY5"
        //     }
        //   }
        // ```

        // Construct an AwsCrypto object with the encryption algorithm
 `ALG_AES_256_GCM_HKDF_SHA512_COMMIT_KEY` which
        // is used by EventBridge for encryption operation. This object is an entry
 point for decryption operation.
        // It can later use decryptData(MasterKeyProvider, byte[]) method to decrypt
 data.
        final AwsCrypto crypto = AwsCrypto.builder()

 .withEncryptionAlgorithm(CryptoAlgorithm.ALG_AES_256_GCM_HKDF_SHA512_COMMIT_KEY)
                .build();

        // Construct AWS KMS master key provider with AWS KMS Client Supplier and AWS
 KMS Key ARN. The KMS Client Supplier can
        // implement a RegionalClientSupplier interface. The AWS KMS Key ARN can be
 fetched from kms-key-arn property in
        // encrypted event json detail.
        final KmsMasterKeyProvider kmsMasterKeyProvider =
 KmsMasterKeyProvider.builder()
                .customRegionalClientSupplier(...)
                .buildStrict(KMS_KEY_ARN);

        // The string of encrypted-payload is base64 encoded. Decode it into byte
 array, so it can be furthur
        // decrypted. The encrypted payload can be fetched from encrypted-payload field
 in encrypted event json detail.
```

```
        byte[] encryptedByteArray = Base64.getDecoder().decode(ENCRYPTED_PAYLOAD);

        // The decryption operation. It retrieves the encryption context and encrypted
 data key from the cipher
        // text headers, which is parsed from byte array encrypted data. Then it
 decrypts the data key, and
        // uses it to finally decrypt event payload. This encryption/decryption
 strategy is called envelope
        // encryption, https://docs.aws.amazon.com/kms/latest/developerguide/
concepts.html#enveloping
        final CryptoResult<byte[], KmsMasterKey> decryptResult =
 crypto.decryptData(kmsMasterKeyProvider, encryptedByteArray);

        final byte[] decryptedByteArray = decryptResult.getResult();

        // Decode the event json plaintext from byte array into string with UTF_8
 standard.
        String eventJson = new String(decryptedByteArray, StandardCharsets.UTF_8);
```

Tagbasierte Richtlinien

In Amazon EventBridge können Sie Richtlinien zum Steuern des Zugriffs auf Ressourcen basierend auf Tags verwenden.

Sie könnten z. B. den Zugriff auf Ressourcen beschränken, die einen Tag mit dem Schlüssel `environment` und dem Wert `production` enthalten. Die folgende Beispielrichtlinie verweigert jeder Ressource mit diesem Tag die Fähigkeit, Tags, Regeln oder Event Buses für Ressourcen zu erstellen, zu löschen oder zu ändern, die mit `environment/production` markiert wurden.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "events:PutRule",
                "events:DescribeRule",
                "events:DeleteRule",
                "events:CreateEventBus",
                "events:DescribeEventBus"
                "events:DeleteEventBus"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/environment": "production"}
            }
        }
    ]
}
```

Weitere Informationen zum Tagging finden Sie in den folgenden Abschnitten.

- Amazon- EventBridge Tags
- Zugriffssteuerung mit IAM-Tags

Amazon EventBridge und AWS Identity and Access Management

Um auf Amazon zuzugreifen EventBridge, benötigen Sie Anmeldeinformationen, die zur Authentifizierung Ihrer Anfragen verwenden AWS kann. Ihre Anmeldeinformationen müssen über Berechtigungen für den Zugriff auf AWS-Ressourcen wie das Abrufen von Ereignisdaten von anderen AWS-Ressourcen verfügen. In den folgenden Abschnitten erfahren Sie, wie Sie Ihre Ressourcen mit AWS Identity and Access Management (IAM) EventBridge sichern können, indem Sie steuern, wer darauf zugreifen kann.

Themen

Authentifizierung

Sie können mit einer der folgenden Identitäten auf AWS zugreifen:

- AWS-Konto-Root-Benutzer – Wenn Sie sich bei AWS registrieren, geben Sie eine E-Mail-Adresse und ein Passwort an, die mit Ihrem -Konto verknüpft sind. Dies sind Ihre Root-Anmeldeinformationen. Sie bieten vollständigen Zugriff auf alle Ihre AWS-Ressourcen.

> **Important**
>
> Aus Sicherheitsgründen empfehlen wir, die Root-Anmeldeinformationen nur zum Erstellen eines Administrators zu verwenden. Hierbei handelt es sich um einen IAM-Benutzer mit vollständigen Berechtigungen für Ihr Konto. Anschließend können Sie mit diesem Administrator weitere -Benutzer und -Rollen mit eingeschränkten Berechtigungen erstellen.

Weitere Informationen finden Sie unter Bewährte Methoden für IAM und Erstellen eines Administratorbenutzers und einer Gruppe im IAM-Benutzerhandbuch.

- IAM-Benutzer – Ein IAM-Benutzer ist eine Identität in Ihrem Konto mit bestimmten Berechtigungen, z. B. der Berechtigung zum Senden von Ereignisdaten an ein Ziel in EventBridge. Sie können IAM-Anmeldeinformationen verwenden, um sich auf sicheren AWS-Webseiten anzumelden, z. B. bei der AWS Management Console, bei AWS-Diskussionsforen oder beim AWS Support Center.

  Außer Anmeldeinformationen können Sie Zugriffsschlüssel für jeden Benutzer erstellen. Sie können diese Schlüssel verwenden, wenn Sie programmgesteuert auf AWS-Services zugreifen, um Ihre Anfrage kryptografisch zu signieren, entweder über eines der SDKs oder mithilfe der AWS Command Line Interface (AWS CLI). Wenn Sie die AWS-Tools nicht nutzen, müssen Sie die Anfrage mit Signature Version 4 selbst signieren, einem Protokoll für die Authentifizierung eingehender API-Anfragen. Weitere Informationen zur Authentifizierung von Anfragen finden Sie unter Signature Version 4-Signaturprozess im Allgemeine Amazon Web Services-Referenz.
- IAM-Rolle – Eine IAM-Rolle ist eine weitere IAM-Identität, die Sie in Ihrem Konto mit bestimmten Berechtigungen erstellen können. Sie ähnelt einem IAM-Benutzer, ist aber nicht mit einer bestimmten Person verknüpft. Mithilfe einer IAM-Rolle können Sie temporäre Zugriffsschlüssel für den Zugriff auf AWS-Services und -Ressourcen erhalten. IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:
  - Verbundener Benutzerzugriff – Statt einen Benutzer zu erstellen, können Sie Identitäten von AWS Directory Service, dem Benutzerverzeichnis Ihres Unternehmens oder eines Web-Identitätsanbieters (IDP) verwenden. Diese werden als Verbundbenutzer bezeichnet. AWS weist einem Verbundbenutzer eine Rolle zu, wenn der Benutzer Zugriff über einen Identitätsanbieter anfordert. Weitere Informationen zu Verbundbenutzern finden Sie unter Verbundbenutzer und Rollen im IAM-Leitfaden.
  - Kontoübergreifender Zugriff – Sie können eine IAM-Rolle in Ihrem Konto verwenden, um einem anderen Konto die Berechtigung für den Zugriff auf die Ressourcen Ihres Kontos zu erteilen. Ein Beispiel finden Sie unter Tutorial: Delegieren des Zugriffs in allen AWS-Konten mithilfe von IAM-Rollen im IAM-Benutzerhandbuch.
  - Zugriff auf AWS-Services – Sie können eine IAM-Rolle in Ihrem Konto verwenden, um einem AWS-Service die Berechtigung für den Zugriff auf die Ressourcen Ihres Konto zu erteilen. Sie können beispielsweise eine Rolle erstellen, mit der Amazon Redshift die in einem Amazon-S3-Bucket gespeicherten Daten in einen Amazon-Redshift-Cluster laden kann. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

- Anwendungen, die auf Amazon EC2 ausgeführt werden – Für Amazon EC2-Anwendungen, die Zugriff auf benötigen EventBridge, können Sie entweder Zugriffsschlüssel in der EC2-Instance speichern oder eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen zu verwalten. Um eine AWS-Rolle einer EC2-Instance zuzuweisen, erstellen Sie ein Instance-Profil, das der Instance angefügt ist. Ein Instance-Profil enthält die Rolle und stellt temporäre Anmeldeinformationen für Anwendungen bereit, die auf der EC2-Instance ausgeführt werden. Weitere Informationen finden Sie im Thema zum Verwenden von Rollen für Anwendungen in Amazon EC2 im IAM-Benutzerhandbuch.

Zugriffskontrolle

Um EventBridge Ressourcen zu erstellen oder darauf zuzugreifen, benötigen Sie sowohl gültige Anmeldeinformationen als auch Berechtigungen. Um beispielsweise AWS Lambda-, Amazon Simple Notification Service (Amazon SNS)- und Amazon Simple Queue Service (Amazon SQS)-Ziele aufzurufen, müssen Sie über Berechtigungen für diese Services verfügen.

Verwalten der Zugriffsberechtigungen für Ihre Amazon-EventBridge-Ressourcen

Sie verwalten den Zugriff auf EventBridge-Ressourcen wie Regeln oder Ereignisse mithilfe identitätsbasierter oder ressourcenbasierter Richtlinien.

EventBridge-Ressourcen

EventBridge-Ressourcen und -Subressourcen sind eindeutige Amazon-Ressourcennamen (ARNs) zugeordnet. Sie verwenden ARNs in EventBridge, um Ereignismuster zu erstellen. Weitere Informationen über ARNs finden Sie unter Amazon-Ressourcennamen (ARN) und AWS-Service-Namespaces im Allgemeine Amazon Web Services-Referenz.

Eine Liste der Operationen, die EventBridge für die Arbeit mit Ressourcen bereitstellt, finden Sie unter Referenz zu Amazon-EventBridge-Berechtigungen.

> **Note**
>
> Die meisten AWS-Services behandeln einen Doppelpunkt (`:`) oder einen Schrägstrich (`/`) in ARNs als genau diese Zeichen. EventBridge verwendet jedoch eine genaue Übereinstimmung in Ereignismustern und Regeln. Verwenden Sie also die richtigen ARN-Zeichen zum Erstellen von Ereignismustern, sodass sie mit der ARN-Syntax in dem Ereignis übereinstimmen.

Die folgende Tabelle zeigt die Ressourcen in EventBridge.

Ressourcentyp	ARN-Format
Archiv	`arn:aws:events:` *`region`*`:`*`account`*`:archive/` *`archive-name`*
Erneut abspielen	`arn:aws:events:` *`region`*`:`*`account`*`:replay/`*`replay-name`
Regel	`arn:aws:events:` *`region`*`:`*`account`*`:rule/`*`[event-bus-name]`*`/`*`rule-name`

Ressourcentyp	ARN-Format
Event Bus	`arn:aws:events:` *`region`*`:`*`account`*`:event-bus/` *`event-bus-name`*
Alle EventBridge-Ressourcen	`arn:aws:events:*`
Alle EventBridge-Ressourcen, die sich im Besitz des angegebenen Kontos in der angegebenen Region befinden	`arn:aws:events:` *`region`*`:`*`account`*`:*`

Im folgenden Beispiel wird gezeigt, wie eine bestimmte Regel (*`myRule`*) in der Anweisung mittels ihres ARN angegeben wird.

```
"Resource": "arn:aws:events:us-east-1:123456789012:rule/myRule"
```

Wenn Sie alle Regeln angeben möchten, die zu einem bestimmten Konto gehören, verwenden Sie das Sternchen (*) wie folgt.

```
"Resource": "arn:aws:events:us-east-1:123456789012:rule/*"
```

Wenn Sie alle Ressourcen angeben möchten oder eine bestimmte API-Aktion keine ARNs unterstützt, verwenden Sie das Sternchen (*) im `Resource`-Element wie folgt.

```
"Resource": "*"
```

Wenn Sie mehrere Ressourcen oder `PutTargets` in einer einzigen Anweisung angeben möchten, trennen Sie ihre ARNs mit Kommas wie folgt.

```
"Resource": ["arn1", "arn2"]
```

Ressourceneigentümerschaft

Unabhängig vom Ersteller ist ein Konto Eigentümer aller innerhalb des Kontos enthaltenen Ressourcen. Der Ressourceneigentümer ist das Konto der Prinzipal-Entität, d. h. der Root-Benutzer des Kontos, ein IAM-Benutzer oder eine IAM-Rolle, welche die Anforderung, die Ressource zu erstellen, authentifiziert. Die Funktionsweise wird anhand der folgenden Beispiele deutlich:

- Wenn Sie die Root-Benutzer-Anmeldeinformationen für Ihr Konto verwenden, um eine Regel zu erstellen, ist Ihr Konto der Besitzer der EventBridge-Ressource.
- Wenn Sie einen Benutzer in Ihrem Konto erstellen und diesem Berechtigungen zum Erstellen von EventBridge-Ressourcen erteilen, kann dieser Benutzer EventBridge-Ressourcen erstellen. Besitzer der EventBridge-Ressource ist jedoch das Konto, zu dem der Benutzer gehört.
- Wenn Sie in Ihrem Konto eine IAM-Rolle mit Berechtigungen zum Erstellen von EventBridge-Ressourcen erstellen, kann jeder Benutzer, der die Rolle übernehmen kann, EventBridge-Ressourcen erstellen. Besitzer der EventBridge-Ressourcen ist das Konto, zu dem die Rolle gehört.

Verwalten des Zugriffs auf Ressourcen

Eine Berechtigungsrichtlinie beschreibt, wer Zugriff auf welche Objekte hat. Im folgenden Abschnitt werden die verfügbaren Optionen zum Erstellen von Berechtigungsrichtlinien erläutert.

> **Note**
>
> Dieser Abschnitt behandelt die Verwendung von IAM um Zusammenhang mit EventBridge. Er enthält keine detaillierten Informationen über den IAM-Service. Eine umfassende IAM-Dokumentation finden Sie unter Was ist IAM? im IAM-Benutzerhandbuch. Informationen über die Syntax und Beschreibungen von IAM-Richtlinien finden Sie in der -IAM-Richtlinienreferenz im IAM-Benutzerhandbuch.

Richtlinien, die einer IAM-Identität zugeordnet sind, werden als identitätsbasierte Richtlinien (IAM-Richtlinien) bezeichnet, während Richtlinien, die einer Ressource zugeordnet sind, ressourcenbasierte Richtlinien genannt werden. In EventBridge können Sie sowohl identitätsbasierte (IAM-Richtlinien) als auch ressourcenbasierte Richtlinien verwenden.

Themen

- Identitätsbasierte Richtlinien (IAM-Richtlinien)
- Ressourcenbasierte Richtlinien (IAM-Richtlinien)

Identitätsbasierte Richtlinien (IAM-Richtlinien)

Richtlinien können IAM-Identitäten angefügt werden. Sie können z. B. Folgendes tun:

- Einem Benutzer oder einer Gruppe in Ihrem Konto eine Berechtigungsrichtlinie zuweisen – Wenn Sie einem Benutzer die Berechtigung zum Anzeigen von Regeln in der Amazon-CloudWatch-Konsole erteilen möchten, weisen Sie dem Benutzer oder der Gruppe, zu der er gehört, eine Berechtigungsrichtlinie zu.
- Einer Rolle eine Berechtigungsrichtlinie zuweisen (kontoübergreifende Berechtigungen gewähren) – Sie können einer IAM-Rolle eine identitätsbasierte Berechtigungsrichtlinie zuweisen, um kontoübergreifende Berechtigungen zu erteilen. Beispielsweise kann der Administrator in Konto A eine Rolle erstellen, um einem anderen Konto B oder einem AWS-Service kontoübergreifende Berechtigungen zu erteilen. Dazu geht er folgendermaßen vor:
  1. Der Administrator von Konto A erstellt eine IAM-Rolle und fügt ihr eine Berechtigungsrichtlinie an, die die Berechtigung für Ressourcen in Konto A erteilt.
  2. Der Administrator von Konto A weist der Rolle eine Vertrauensrichtlinie zu, die Konto B als den Prinzipal identifiziert, der die Rolle übernehmen kann.
  3. Der Administrator von Konto B kann nun Berechtigungen zur Übernahme der Rolle an alle Benutzer in Konto B delegieren. Daraufhin können die Benutzer in Konto B auf Ressourcen von Konto A zugreifen oder auch Ressourcen erstellen. Der Prinzipal in der Vertrauensrichtlinie kann auch ein AWS-Service-Prinzipal sein. Somit können Sie auch einem AWS-Service die Berechtigung zur Übernahme der Rolle erteilen.

  Weitere Informationen zum Delegieren von Berechtigungen mithilfe von IAM finden Sie unter Zugriffsverwaltung im IAM-Benutzerhandbuch.

Sie können spezifische IAM-Richtlinien erstellen, um die Aufrufe und Ressourcen zu beschränken, auf die Benutzer in Ihrem Konto Zugriff haben, und dann diese Richtlinien den Benutzern zuweisen. Weitere Informationen zum Erstellen von IAM-Rollen und Beispiele zu IAM-Richtlinienanweisungen für EventBridge finden Sie unter Verwalten der Zugriffsberechtigungen für Ihre Amazon-EventBridge-Ressourcen.

Ressourcenbasierte Richtlinien (IAM-Richtlinien)

Wenn eine Regel in EventBridge ausgeführt wird, werden alle mit der Regel verknüpften Ziele aufgerufen. Dies bedeutet das Aufrufen der AWS Lambda-Funktionen, das Veröffentlichen in den Amazon-SNS-Themen oder das Weiterleiten des Ereignisses an die Amazon-Kinesis-Streams. Um API-Aufrufe für die Ressourcen auszuführen, die Sie besitzen, muss EventBridge über die entsprechende Berechtigung verfügen. Für Lambda-, Amazon-SNS- und Amazon-SQS-Ressourcen verwendet EventBridge ressourcenbasierte Richtlinien. Für Kinesis-Streams verwendet EventBridge IAM-Rollen.

Weitere Informationen zum Erstellen von IAM-Rollen und Beispiele zu ressourcenbasierten Richtlinienanweisungen für EventBridge finden Sie unter Verwenden ressourcenbasierter Richtlinien für Amazon EventBridge.

Festlegen der Richtlinienelemente: Aktionen, Effekte und Prinzipale

Für jede EventBridge-Ressource definiert EventBridge eine Reihe von API-Operationen. Zur Erteilung von Berechtigungen für diese API-Operationen definiert EventBridge eine Reihe von Aktionen, die Sie in einer Richtlinie angeben können. Einige API-Operationen erfordern Berechtigungen für mehr als eine Aktion, um die API-Operation auszuführen. Weitere Informationen zu Ressourcen und API-Operationen finden Sie unter EventBridge-Ressourcen und Referenz zu Amazon-EventBridge-Berechtigungen.

Grundlegende Richtlinienelemente:

- Ressource – Verwenden Sie einen Amazon-Ressourcennamen (ARN), um die Ressource, für welche die Richtlinie gilt, zu identifizieren. Weitere Informationen finden Sie unter EventBridge-Ressourcen.
- Aktion – Mit Schlüsselwörtern geben Sie die Ressourcenoperationen an, die Sie zulassen oder verweigern möchten. Die `events:Describe`-Berechtigung erteilt dem Benutzer zum Beispiel Berechtigungen zum Ausführen der `Describe`-Operation.
- Effekt – Geben Sie entweder Zulassen oder Verweigern an. Wenn Sie den Zugriff auf eine Ressource nicht ausdrücklich gestatten („Allow“), wird er verweigert. Sie können den Zugriff auf eine Ressource auch explizit verweigern. So können Sie sicherstellen, dass Benutzer nicht darauf zugreifen können, auch wenn der Zugriff durch eine andere Richtlinie gestattet wird.
- Prinzipal – In identitätsbasierten Richtlinien (IAM-Richtlinien) ist der Benutzer, dem die Richtlinie zugewiesen ist, automatisch der Prinzipal. In ressourcenbasierten Richtlinien müssen Sie den

Benutzer, das Konto, den Service oder die sonstige Entität angeben, die die Berechtigungen erhalten soll (gilt nur für ressourcenbasierte Richtlinien).

Weitere Informationen zur IAM-Richtliniensyntax und Beschreibungen finden Sie in der IAM-JSON-Richtlinienreferenz im IAM-Benutzerhandbuch.

Informationen zu den EventBridge-API-Aktionen und den Ressourcen, für die sie gelten, finden Sie unter Referenz zu Amazon-EventBridge-Berechtigungen.

Angeben von Bedingungen in einer Richtlinie

Beim Erteilen von Berechtigungen können Sie mithilfe der Sprache der Zugriffsrichtlinie die Bedingungen angeben, wann die Richtlinie wirksam werden soll. Beispielsweise kann festgelegt werden, dass eine Richtlinie erst ab einem bestimmten Datum gilt. Weitere Informationen zum Angeben von Bedingungen in einer Richtliniensyntax finden Sie im Thema Bedingung im IAM Benutzerhandbuch.

Zum Festlegen von Bedingungen verwenden Sie Bedingungsschlüssel. Es gibt AWS-Bedingungsschlüssel und EventBridge-spezifische Schlüssel, die Sie gegebenenfalls verwenden können. Sie finden eine vollständige Liste der AWS-Schlüssel unter Available Keys for Conditions (Verfügbare Schlüssel für Bedingungen) im IAM User Guide (IAM-Benutzerhandbuch). Eine vollständige Liste der EventBridge-spezifischen Schlüssel finden Sie unter Verwenden von IAM-Richtlinienbedingungen für die differenzierte Zugriffskontrolle.

Verwendung identitätsbasierter Richtlinien (IAM-Richtlinien) für Amazon EventBridge

Identitätsbasierte Richtlinien sind Berechtigungsrichtlinien, die Sie an eine IAM-Identität anfügen können.

Themen

AWS verwaltete Richtlinien für EventBridge

AWS adressiert viele gängige Anwendungsfälle durch die Bereitstellung eigenständiger IAM-Richtlinien, die von erstellt und verwaltet AWS werden. Verwaltete oder vordefinierte Richtlinien erteilen die erforderlichen Berechtigungen für viele häufige Anwendungsfälle, sodass Sie nicht mühsam ermitteln müssen, welche Berechtigungen erforderlich sind. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien im IAM-Benutzerhandbuch.

Die folgenden AWS verwalteten Richtlinien, die Sie Benutzern in Ihrem Konto zuordnen können, sind spezifisch für EventBridge:

- AmazonEventBridgeFullAccess— Gewährt vollen Zugriff auf EventBridge, einschließlich EventBridge Pipes, EventBridge Schemas und EventBridge Scheduler.
- AmazonEventBridgeReadOnlyAccess— Gewährt schreibgeschützten Zugriff auf EventBridge, einschließlich EventBridge Pipes, Schemas und Scheduler. EventBridge EventBridge

AmazonEventBridgeFullAccess Richtlinie

Die AmazonEventBridgeFullAccess Richtlinie gewährt Berechtigungen zur Verwendung aller EventBridge Aktionen sowie die folgenden Berechtigungen:

- `iam:CreateServiceLinkedRole`— EventBridge benötigt diese Berechtigung, um die Servicerolle in Ihrem Konto für API-Ziele zu erstellen. Diese Berechtigung gewährt nur dem IAM-Service die Berechtigung, eine Rolle in Ihrem Konto speziell für API-Ziele zu erstellen.

- `iam:PassRole`— EventBridge benötigt diese Berechtigung, um eine Aufrufrolle zu übergeben, an die das Ziel einer Regel aufgerufen werden EventBridge soll.
- Secrets Manager Manager-Berechtigungen — EventBridge erfordert diese Berechtigungen, um Geheimnisse in Ihrem Konto zu verwalten, wenn Sie Anmeldeinformationen über die Verbindungsressource angeben, um API-Ziele zu autorisieren.

Die folgende JSON-Datei zeigt die AmazonEventBridgeFullAccess Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EventBridgeActions",
            "Effect": "Allow",
            "Action": [
                "events:*",
                "schemas:*",
                "scheduler:*",
                "pipes:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "IAMCreateServiceLinkedRoleForApiDestinations",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/
AmazonEventBridgeApiDestinationsServiceRolePolicy",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "apidestinations.events.amazonaws.com"
                }
            }
        },

        {
            "Sid": "SecretsManagerAccessForApiDestinations",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:CreateSecret",
                "secretsmanager:UpdateSecret",
                "secretsmanager:DeleteSecret",
```

```
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue"
            ],
            "Resource": "arn:aws:secretsmanager:*:*:secret:events!*"
        },
        {
            "Sid": "IAMPassRoleAccessForEventBridge",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "events.amazonaws.com"
                }
            }
        },
        {
            "Sid": "IAMPassRoleAccessForScheduler",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "scheduler.amazonaws.com"
                }
            }
        },
        {
            "Sid": "IAMPassRoleAccessForPipes",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "pipes.amazonaws.com"
                }
            }
        }
    ]
}
```

> **Note**
>
> Die Informationen in diesem Abschnitt gelten auch für die Richtlinie `CloudWatchEventsFullAccess`. Es wird jedoch dringend empfohlen, Amazon EventBridge anstelle von Amazon CloudWatch Events zu verwenden.

AmazonEventBridgeReadOnlyAccess Richtlinie

Die AmazonEventBridgeReadOnlyAccess Richtlinie gewährt Berechtigungen zur Verwendung aller EventBridge Leseaktionen.

Die folgende JSON-Datei zeigt die AmazonEventBridgeReadOnlyAccess Richtlinie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "events:DescribeRule",
                "events:DescribeEventBus",
                "events:DescribeEventSource",
                "events:ListEventBuses",
                "events:ListEventSources",
                "events:ListRuleNamesByTarget",
                "events:ListRules",
                "events:ListTargetsByRule",
                "events:TestEventPattern",
                "events:DescribeArchive",
                "events:ListArchives",
                "events:DescribeReplay",
                "events:ListReplays",
                "events:DescribeConnection",
                "events:ListConnections",
                "events:DescribeApiDestination",
                "events:ListApiDestinations",
                "events:DescribeEndpoint",
                "events:ListEndpoints",
                "schemas:DescribeCodeBinding",
                "schemas:DescribeDiscoverer",
                "schemas:DescribeRegistry",
```

```
                "schemas:DescribeSchema",
                "schemas:ExportSchema",
                "schemas:GetCodeBindingSource",
                "schemas:GetDiscoveredSchema",
                "schemas:GetResourcePolicy",
                "schemas:ListDiscoverers",
                "schemas:ListRegistries",
                "schemas:ListSchemas",
                "schemas:ListSchemaVersions",

                "schemas:ListTagsForResource",
                "schemas:SearchSchemas",
                "scheduler:GetSchedule",
                "scheduler:GetScheduleGroup",
                "scheduler:ListSchedules",
                "scheduler:ListScheduleGroups",
                "scheduler:ListTagsForResource",
                "pipes:DescribePipe",
                "pipes:ListPipes",
                "pipes:ListTagsForResource"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Die Informationen in diesem Abschnitt gelten auch für die Richtlinie `CloudWatchEventsReadOnlyAccess`. Es wird jedoch dringend empfohlen, Amazon EventBridge anstelle von Amazon CloudWatch Events zu verwenden.

EventBridge Schemaspezifische verwaltete Richtlinien

Ein Schema definiert die Struktur der Ereignisse, an die gesendet werden. EventBridge EventBridge stellt Schemas für alle Ereignisse bereit, die von AWS Diensten generiert werden. Die folgenden AWS verwalteten Richtlinien sind speziell für EventBridge Schemas verfügbar:

- AmazonEventBridgeSchemasServiceRolePolicy
- AmazonEventBridgeSchemasFullAccess

- AmazonEventBridgeSchemasReadOnlyAccess

EventBridge Scheduler-spezifische verwaltete Richtlinien

Amazon EventBridge Scheduler ist ein serverloser Scheduler, mit dem Sie Aufgaben von einem zentralen, verwalteten Service aus erstellen, ausführen und verwalten können. Informationen zu AWS verwalteten Richtlinien, die speziell für EventBridge Scheduler gelten, finden Sie unter AWS Verwaltete Richtlinien für EventBridge Scheduler im Scheduler-Benutzerhandbuch. EventBridge

EventBridge Rohrspezifische verwaltete Richtlinien

Amazon EventBridge Pipes verbindet Ereignisquellen mit Zielen. Pipes reduzieren den Bedarf an Fachwissen und Integrationscode bei der Entwicklung ereignisgesteuerter Architekturen. Dies trägt dazu bei, die Konsistenz der Anwendungen Ihres Unternehmens sicherzustellen. Die folgenden AWS verwalteten Richtlinien, die speziell für EventBridge Pipes gelten, sind verfügbar:

- AmazonEventBridgePipesFullAccess

  Bietet vollen Zugriff auf Amazon EventBridge Pipes.

  > **Note**
  >
  > Diese Richtlinie sieht vor, `iam:PassRole` dass EventBridge Pipes diese Berechtigung benötigt, um eine Aufrufrolle zu übergeben, EventBridge um Pipes zu erstellen und zu starten.

- AmazonEventBridgePipesReadOnlyAccess

  Bietet schreibgeschützten Zugriff auf Amazon EventBridge Pipes.

- AmazonEventBridgePipesOperatorAccess

  Bietet schreibgeschützten Zugriff und Bedienerzugriff (d. h. die Möglichkeit, die Ausführung von Pipes zu beenden und zu starten) auf Amazon EventBridge Pipes.

IAM-Rollen zum Senden von Ereignissen

Um Ereignisse an Ziele weiterzuleiten, ist eine EventBridge IAM-Rolle erforderlich.

Um eine IAM-Rolle für das Senden von Ereignissen zu erstellen EventBridge

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Um eine IAM-Rolle zu erstellen, folgen Sie den Schritten unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS Dienst im IAM-Benutzerhandbuch. Beachten Sie Folgendes, während Sie die Schritte ausführen:

   - Geben Sie unter Rollenname einen Namen ein, der innerhalb Ihres Kontos eindeutig ist.
   - Wählen Sie unter Rollentyp auswählen die Option AWS Service Roles und dann Amazon aus EventBridge. Dadurch werden EventBridge Berechtigungen zur Übernahme der Rolle erteilt.
   - Wählen Sie unter Richtlinie anhängen die Option aus AmazonEventBridgeFullAccess.

Sie können auch Ihre eigenen benutzerdefinierten IAM-Richtlinien erstellen, um Berechtigungen für EventBridge Aktionen und Ressourcen zu gewähren. Die benutzerdefinierten Richtlinien können Sie dann den IAM-Benutzern oder -Gruppen zuweisen, die diese Berechtigungen benötigen. Weitere Informationen zu IAM-Richtlinien finden Sie unter Übersicht über IAM-Richtlinien im IAM-Benutzerhandbuch. Weitere Informationen zum Verwalten und Erstellen von benutzerdefinierten IAM-Richtlinien finden Sie unter Verwalten von IAM-Richtlinien im IAM-Benutzerhandbuch.

Berechtigungen, die für den Zugriff EventBridge auf Ziele mithilfe von IAM-Rollen erforderlich sind

EventBridge Ziele benötigen in der Regel IAM-Rollen, die die Berechtigung EventBridge zum Aufrufen des Ziels gewähren. Im Folgenden finden Sie einige Beispiele für verschiedene AWS Dienste und Ziele. Andere Benutzer können die EventBridge Konsole verwenden, um eine Regel und eine neue Rolle zu erstellen, die dann mit einer Richtlinie mit vorkonfigurierten Berechtigungen für einen bestimmten Gültigkeitsbereich erstellt wird.

Amazon SQS, Amazon SNS, Lambda, CloudWatch Logs und EventBridge Busziele verwenden keine Rollen, und Berechtigungen dafür EventBridge müssen über eine Ressourcenrichtlinie erteilt werden. API-Gateway-Ziele können entweder Ressourcenrichtlinien oder IAM-Rollen verwenden.

Wenn das Ziel ein API-Ziel ist, muss die Rolle, die Sie angeben, die folgende Richtlinie enthalten.

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

```
            "Effect": "Allow",
             "Action": [ "events:InvokeApiDestination" ],
            "Resource": [ "arn:aws:events:::api-destination/*" ]
        }
     ]
}
```

Wenn das Ziel ein Kinesis-Stream ist, muss die Rolle, die zum Senden von Ereignisdaten an das Ziel verwendet wird, folgende Richtlinie enthalten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kinesis:PutRecord"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn das Ziel Systems Manager Run Command ist und Sie einen oder mehrere `InstanceIds`-Werte für den Befehl angeben, muss die Rolle, die Sie angeben, die folgende Richtlinie enthalten.

```
{
     "Version": "2012-10-17",
     "Statement": [
           {
                "Action": "ssm:SendCommand",
                "Effect": "Allow",
                "Resource": [
                     "arn:aws:ec2:region:accountId:instance/instanceIds",
                     "arn:aws:ssm:region:*:document/documentName"
                ]
           }
     ]
}
```

Wenn das Ziel Systems Manager Run Command ist und Sie einen oder mehrere Tags für den Befehl angeben, muss die Rolle, die Sie angeben, die folgende Richtlinie enthalten.

```
{
     "Version": "2012-10-17",
     "Statement": [
          {
               "Action": "ssm:SendCommand",
               "Effect": "Allow",
               "Resource": [
                    "arn:aws:ec2:region:accountId:instance/*"
               ],
               "Condition": {
                   "StringEquals": {
                       "ec2:ResourceTag/*": [
                           "[[tagValues]]"
                       ]
                   }
               }
          },
          {
               "Action": "ssm:SendCommand",
               "Effect": "Allow",
               "Resource": [
                    "arn:aws:ssm:region:*:document/documentName"
               ]
          }
     ]
}
```

Wenn es sich bei dem Ziel um eine AWS Step Functions Zustandsmaschine handelt, muss die von Ihnen angegebene Rolle die folgende Richtlinie enthalten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
             "Action": [ "states:StartExecution" ],
            "Resource": [ "arn:aws:states:*:*:stateMachine:*" ]
        }
     ]
}
```

Wenn das Ziel eine Amazon-ECS-Aufgabe ist, muss die Rolle, die Sie angeben, die folgende Richtlinie enthalten.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "ecs:RunTask"
        ],
        "Resource": [
            "arn:aws:ecs:*:account-id:task-definition/task-definition-name"
        ],
        "Condition": {
            "ArnLike": {
                "ecs:cluster": "arn:aws:ecs:*:account-id:cluster/cluster-name"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action":"iam:PassRole",
        "Resource": [
            "*"
        ],
        "Condition": {
            "StringLike": {
                "iam:PassedToService": "ecs-tasks.amazonaws.com"
            }
        }
    }]
}
```

Die folgende Richtlinie ermöglicht es integrierten Zielen EventBridge , Amazon EC2 EC2-Aktionen in Ihrem Namen durchzuführen. Sie müssen den verwenden, AWS Management Console um Regeln mit integrierten Zielen zu erstellen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TargetInvocationAccess",
            "Effect": "Allow",
```

```
            "Action": [
                "ec2:Describe*",
                "ec2:RebootInstances",
                "ec2:StopInstances",
                "ec2:TerminateInstances",
                "ec2:CreateSnapshot"
            ],
            "Resource": "*"
        }
    ]
}
```

Die folgende Richtlinie ermöglicht EventBridge die Weiterleitung von Ereignissen an die Kinesis-Streams in Ihrem Konto.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "KinesisAccess",
            "Effect": "Allow",
            "Action": [
                "kinesis:PutRecord"
            ],
            "Resource": "*"
        }
    ]
}
```

Beispiel für eine vom Kunden verwaltete Richtlinie: Verwenden der Markierung mit Tags zum Steuern des Zugriffs auf Regeln

Das folgende Beispiel zeigt eine Benutzerrichtlinie, die Berechtigungen für EventBridge Aktionen gewährt. Diese Richtlinie funktioniert, wenn Sie die EventBridge API, AWS SDKs oder die AWS CLI verwenden.

Sie können Benutzern Zugriff auf bestimmte EventBridge Regeln gewähren und sie gleichzeitig daran hindern, auf andere Regeln zuzugreifen. Hierzu markieren Sie beide Regelsätze mit Tags und verwenden dann IAM-Richtlinien, die auf diese Tags verweisen. Weitere Informationen zum Markieren von EventBridge Ressourcen finden Sie unterAmazon- EventBridge Tags.

Sie können einem Benutzer eine IAM-Richtlinie zuweisen, die ausschließlich den Zugriff auf Regeln mit einem bestimmten Tag zulässt. Sie wählen aus, auf welche Regeln Sie Zugriff gewähren möchten, indem Sie sie mit diesem bestimmten Tag markieren. Beispielweise gewährt die folgende Richtlinie einem Benutzer nur den Zugriff auf Regeln mit dem Wert `Prod` für den Tag-Schlüssel `Stack`.

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "events:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Stack": "Prod"
                }
            }
        }
    ]
}
```

Weitere Informationen zur Verwendung von IAM-Richtlinienanweisungen finden Sie unter Steuern des Zugriffs mithilfe von Richtlinien im IAM-Benutzerhandbuch.

EventBridge Aktualisierungen der AWS verwalteten Richtlinien durch Amazon

Hier finden Sie Informationen zu Aktualisierungen der AWS verwalteten Richtlinien, die EventBridge seit Beginn der Nachverfolgung dieser Änderungen durch diesen Service vorgenommen wurden. Abonnieren Sie den RSS-Feed auf der Seite EventBridge Dokumentenverlauf, um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhalten.

Änderung	Beschreibung	Datum
AmazonEventBridgeFullAccess— Aktualisierte Richtlinie	AWS GovCloud (US) Regions nur Die folgende Erlaubnis ist nicht enthalten, da sie nicht verwendet wird:	9. Mai 2024

Änderung	Beschreibung	Datum
	• `iam:CreateServiceLinkedRole` Erlaubnis für EventBridge Schema Registry	
AmazonEventBridgeSchemasFullAccess— Aktualisierte Richtlinie	AWS GovCloud (US) Regions nur Die folgende Erlaubnis ist nicht enthalten, da sie nicht verwendet wird: • `iam:CreateServiceLinkedRole` Erlaubnis für EventBridge Schema Registry	9. Mai 2024
AmazonEventBridgePipesFullAccess— Neue Richtlinie hinzugefügt	EventBridge Es wurde eine verwaltete Richtlinie für vollständige Berechtigungen zur Verwendung von EventBridge Pipes hinzugefügt.	01. Dezember 2022
AmazonEventBridgePipesReadOnlyAccess— Neue Richtlinie hinzugefügt	EventBridge Es wurde eine verwaltete Richtlinie für Berechtigungen zum Anzeigen von EventBridge Pipes-Informationsressourcen hinzugefügt.	01. Dezember 2022

Änderung	Beschreibung	Datum
AmazonEventBridgePipesOperatorAccess— Neue Richtlinie hinzugefügt	EventBridge Es wurde eine verwaltete Richtlinie für Berechtigungen zum Anzeigen von EventBridge Pipes-Informationen sowie zum Starten und Stoppen des Betriebs von Pipes hinzugefügt.	01. Dezember 2022
AmazonEventBridgeFullAccess – Aktualisierung auf eine bestehende Richtlinie	EventBridge Die Richtlinie wurde aktualisiert, sodass sie die für die Nutzung der EventBridge Pipes-Funktionen erforderlichen Berechtigungen enthält.	01. Dezember 2022
AmazonEventBridgeReadOnlyAccess – Aktualisierung auf eine bestehende Richtlinie	EventBridge Es wurden Berechtigungen hinzugefügt, die für das Anzeigen von EventBridge Pipes-Informationsressourcen erforderlich sind. Die folgenden Aktionen wurden hinzugefügt: • `pipes:DescribePipe` • `pipes:ListPipes` • `pipes:ListTagsForResource`	01. Dezember 2022
CloudWatchEventsReadOnlyAccess – Aktualisierung auf eine bestehende Richtlinie	Passend aktualisiert AmazonEventBridgeReadOnlyAccess.	01. Dezember 2022

Änderung	Beschreibung	Datum
CloudWatchEventsFullAccess – Aktualisierung auf eine bestehende Richtlinie	Passend aktualisiert AmazonEventBridgeFullAccess.	01. Dezember 2022
AmazonEventBridgeFullAccess – Aktualisierung auf eine bestehende Richtlinie	EventBridge Die Richtlinie wurde aktualisiert und enthält nun auch die für die Verwendung von Schemas und Scheduler-Funktionen erforderlichen Berechtigungen. Die folgenden Berechtigungen wurden hinzugefügt: • EventBridge Aktionen der Schemaregistrierung • EventBridge Scheduler-Aktionen • `iam:CreateServiceLinkedRole` Erlaubnis für EventBridge Schema Registry • `iam:PassRole` Erlaubnis für EventBridge Scheduler	10. November 2022

Änderung	Beschreibung	Datum
AmazonEventBridgeReadOnlyAccess – Aktualisierung auf eine bestehende Richtlinie	EventBridge Es wurden Berechtigungen hinzugefügt, die für das Anzeigen von Schema- und Scheduler -Informationsressourcen erforderlich sind. Die folgenden Aktionen wurden hinzugefügt: • `schemas:DescribeCodeBinding` • `schemas:DescribeDiscoverer` • `schemas:DescribeRegistry` • `schemas:DescribeSchema` • `schemas:ExportSchema` • `schemas:GetCodeBindingSource` • `schemas:GetDiscoveredSchema` • `schemas:GetResourcePolicy` • `schemas:ListDiscoverers` • `schemas:ListRegistries` • `schemas:ListSchemas` • `schemas:ListSchemaVersions`	10. November 2022

Änderung	Beschreibung	Datum
	• schemas:ListTagsForResource • schemas:SearchSchemas • scheduler:GetSchedule • scheduler:GetScheduleGroup • scheduler:ListSchedules • scheduler:ListScheduleGroups • scheduler:ListTagsForResource	
AmazonEventBridgeReadOnlyAccess – Aktualisierung auf eine bestehende Richtlinie	EventBridge Es wurden die für das Anzeigen von Endpunktinformationen erforderlichen Berechtigungen hinzugefügt. Die folgenden Aktionen wurden hinzugefügt: • events:ListEndpoints • events:DescribeEndpoint	7. April 2022

Änderung	Beschreibung	Datum
AmazonEventBridgeReadOnlyAccess – Aktualisierung auf eine bestehende Richtlinie	EventBridge Es wurden Berechtigungen hinzugefügt, die zum Anzeigen von Verbindungs- und API-Zielinformationen erforderlich sind. Die folgenden Aktionen wurden hinzugefügt: • `events:DescribeConnection` • `events:ListConnections` • `events:DescribeApiDestination` • `events:ListApiDestinations`	4. März 2021

Änderung	Beschreibung	Datum
AmazonEventBridgeFullAccess – Aktualisierung auf eine bestehende Richtlinie	EventBridge Die Richtlinie wurde aktualisiert `iam:CreateServiceLinkedRole` und umfasst nun auch die für die Verwendung von API-Zielen erforderlichen AWS Secrets Manager Berechtigungen. Die folgenden Aktionen wurden hinzugefügt: • `secretsmanager:CreateSecret` • `secretsmanager:UpdateSecret` • `secretsmanager:DeleteSecret` • `secretsmanager:GetSecretValue` • `secretsmanager:PutSecretValue`	4. März 2021
EventBridge hat begonnen, Änderungen zu verfolgen	EventBridge hat begonnen, Änderungen für die AWS verwalteten Richtlinien zu verfolgen.	4. März 2021

Verwenden ressourcenbasierter Richtlinien für Amazon EventBridge

Wenn eine Regel in EventBridge ausgeführt wird, werden alle Ziele aufgerufen, die mit der Regel verknüpft sind. Regeln können AWS Lambda-Funktionen aufrufen, in Amazon-SNS-Themen veröffentlichen oder das Ereignis an Kinesis-Streams weiterleiten. Um API-Aufrufe für die Ressourcen auszuführen, die Sie besitzen, muss EventBridge über die entsprechenden Berechtigungen verfügen. Für Lambda-, Amazon-SNS-, Amazon-SQS- und Amazon-CloudWatch-Logs-Ressourcen verwendet EventBridge ressourcenbasierte Richtlinien. Für Kinesis-Streams verwendet EventBridge identitätsbasierte Richtlinien.

Sie verwenden die AWS CLI, um Ihren Zielen Berechtigungen hinzuzufügen. Informationen zur Installation und Konfiguration der AWS CLI finden Sie unter Einrichtung mit der AWS Command Line Interface im AWS Command Line Interface-Benutzerhandbuch.

Themen

Amazon-API-Gateway-Berechtigungen

Um Ihren Amazon-API-Gateway-Endpunkt mithilfe einer EventBridge-Regel aufzurufen, fügen Sie der Richtlinie Ihres API-Gateway-Endpunkts die folgende Berechtigung hinzu.

```
{
  "Version": "2012-10-17",
  "Statement": [
       {
           "Effect": "Allow",
           "Principal": {
               "Service": "events.amazonaws.com"
           },
           "Action": "execute-api:Invoke",
           "Condition": {
```

```
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:events:region:account-id:rule/rule-name"
                }
            },
            "Resource": [
                "execute-api:/stage/GET/api"
            ]
        }
    ]
}
```

CloudWatch-Logs-Berechtigungen

Wenn CloudWatch Logs das Ziel einer Regel ist, erstellt EventBridge Protokollstreams und CloudWatch Logs speichert den Text aus den Ereignissen als Protokolleinträge. Um EventBridge das Erstellen des Protokollstreams und das Protokollieren der Ereignisse zu ermöglichen, muss CloudWatch Logs eine ressourcenbasierte Richtlinie enthalten, die EventBridge zum Schreiben in CloudWatch Logs berechtigt.

Wenn Sie die AWS Management Console verwenden, um CloudWatch Logs als Ziel einer Regel hinzuzufügen, wird diese Richtlinie automatisch erstellt. Wenn Sie das Ziel mithilfe der AWS CLI hinzufügen und die Richtlinie noch nicht vorhanden ist, müssen Sie sie erstellen.

Das folgende Beispiel ermöglicht EventBridge, in alle Protokollgruppen zu schreiben, deren Namen mit `/aws/events/` beginnen. Wenn Sie bei Protokollen dieser Art eine andere Benennungsrichtlinie verwenden, müssen Sie das Beispiel entsprechend anpassen.

```
{
    "Statement": [
        {
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Effect": "Allow",
            "Principal": {
                "Service": ["events.amazonaws.com", "delivery.logs.amazonaws.com"]
            },
            "Resource": "arn:aws:logs:region:account:log-group:/aws/events/*:*",
            "Sid": "TrustEventsToStoreLogEvent"
        }
    ],
```

```
    "Version": "2012-10-17"
}
```

Weitere Informationen finden Sie im API-Referenzhandbuch für CloudWatch Logs unter PutResourcePolicy.

AWS Lambda-Berechtigungen

Zum Aufrufen Ihrer AWS Lambda-Funktion mittels einer EventBridge-Regel fügen Sie der Richtlinie Ihrer Lambda-Funktion die folgende Berechtigung hinzu.

```
{
  "Effect": "Allow",
  "Action": "lambda:InvokeFunction",
  "Resource": "arn:aws:lambda:region:account-id:function:function-name",
  "Principal": {
    "Service": "events.amazonaws.com"
  },
  "Condition": {
    "ArnLike": {
      "AWS:SourceArn": "arn:aws:events:region:account-id:rule/rule-name"
    }
  },
  "Sid": "InvokeLambdaFunction"
}
```

So fügen Sie die oben genannten Berechtigungen hinzu, die es EventBridge ermöglichen, Lambda-Funktionen mit der AWS CLI aufzurufen

- Geben Sie in der Eingabeaufforderung den folgenden Befehl ein.

  ```
  aws lambda add-permission --statement-id "InvokeLambdaFunction" \
  --action "lambda:InvokeFunction" \
  --principal "events.amazonaws.com" \
  --function-name "arn:aws:lambda:region:account-id:function:function-name" \
  --source-arn "arn:aws:events:region:account-id:rule/rule-name"
  ```

Weitere Informationen zum Einrichten von Berechtigungen, mit denen EventBridge Lambda-Funktionen aufrufen kann, finden Sie unter AddPermission und Verwenden von Lambda mit geplanten Ereignissen im AWS Lambda-Entwicklerhandbuch.

Amazon-SNS-Berechtigungen

Damit EventBridge in einem Amazon-SNS-Thema veröffentlichen kann, verwenden Sie die Befehle `aws sns get-topic-attributes` und `aws sns set-topic-attributes`.

> **Note**
>
> Sie können keine `Condition`-Blöcke in Amazon-SNS-Themenrichtlinien für EventBridge verwenden.

So fügen Sie Berechtigungen hinzu, mit denen EventBridge SNS-Themen veröffentlichen kann

1. Verwenden Sie den folgenden Befehl, um die Attribute eines SNS-Themas aufzulisten.

   ```
   aws sns get-topic-attributes --topic-arn "arn:aws:sns:region:account-id:topic-name"
   ```

   Das folgende Beispiel zeigt das Ergebnis eines neuen SNS-Themas.

   ```
   {
       "Attributes": {
           "SubscriptionsConfirmed": "0",
           "DisplayName": "",
           "SubscriptionsDeleted": "0",
           "EffectiveDeliveryPolicy": "{\"http\":{\"defaultHealthyRetryPolicy\":
   {\"minDelayTarget\":20,\"maxDelayTarget\":20,\"numRetries\":3,\"numMaxDelayRetries
   \":0,\"numNoDelayRetries\":0,\"numMinDelayRetries\":0,\"backoffFunction\":\"linear
   \"},\"disableSubscriptionOverrides\":false}}",
           "Owner": "account-id",
           "Policy": "{\"Version\":\"2012-10-17\",\"Id\":\"__default_policy_ID\",
   \"Statement\":[{\"Sid\":\"__default_statement_ID\",\"Effect\":\"Allow\",\"Principal
   \":{\"AWS\":\"*\"},\"Action\":[\"SNS:GetTopicAttributes\",\"SNS:SetTopicAttributes
   \",\"SNS:AddPermission\",\"SNS:RemovePermission\",\"SNS:DeleteTopic\",
   \"SNS:Subscribe\",\"SNS:ListSubscriptionsByTopic\",\"SNS:Publish\"],\"Resource
   \":\"arn:aws:sns:region:account-id:topic-name\",\"Condition\":{\"StringEquals\":
   {\"AWS:SourceOwner\":\"account-id\"}}}]}",
           "TopicArn": "arn:aws:sns:region:account-id:topic-name",
           "SubscriptionsPending": "0"
       }
   }
   ```

2. Verwenden Sie einen JSON-in-Zeichenfolgen-Konverter, um die folgende Anweisung in eine Zeichenfolge zu konvertieren.

```
{
  "Sid": "PublishEventsToMyTopic",
  "Effect": "Allow",
  "Principal": {
    "Service": "events.amazonaws.com"
  },
  "Action": "sns:Publish",
  "Resource": "arn:aws:sns:region:account-id:topic-name"
}
```

   Nachdem Sie die Anweisung in eine Zeichenfolge konvertiert haben, sieht sie wie im folgenden Beispiel aus.

```
{\"Sid\":\"PublishEventsToMyTopic\",\"Effect\":\"Allow\",\"Principal\":
{\"Service\":\"events.amazonaws.com\"},\"Action\":\"sns:Publish\",\"Resource\":
\"arn:aws:sns:region:account-id:topic-name\"}
```

3. Fügen Sie die Zeichenfolge, die Sie im vorherigen Schritt erstellt haben, der `"Statement"`-Sammlung innerhalb des `"Policy"`-Attributs hinzu.
4. Verwenden Sie den Befehl `aws sns set-topic-attributes`, um die neue Richtlinie einzurichten.

```
aws sns set-topic-attributes --topic-arn "arn:aws:sns:region:account-id:topic-name"
 \
--attribute-name Policy \
--attribute-value "{\"Version\":\"2012-10-17\",\"Id\":\"__default_policy_ID\",
\"Statement\":[{\"Sid\":\"__default_statement_ID\",\"Effect\":\"Allow\",\"Principal
\":{\"AWS\":\"*\"},\"Action\":[\"SNS:GetTopicAttributes\",\"SNS:SetTopicAttributes
\",\"SNS:AddPermission\",\"SNS:RemovePermission\",\"SNS:DeleteTopic\",
\"SNS:Subscribe\",\"SNS:ListSubscriptionsByTopic\",\"SNS:Publish\"],\"Resource
\":\"arn:aws:sns:region:account-id:topic-name\",\"Condition\":{\"StringEquals
\":{\"AWS:SourceOwner\":\"account-id\"}}}, {\"Sid\":\"PublishEventsToMyTopic\",
\"Effect\":\"Allow\",\"Principal\":{\"Service\":\"events.amazonaws.com\"},\"Action
\":\"sns:Publish\",\"Resource\":\"arn:aws:sns:region:account-id:topic-name\"}]}"
```

Weitere Informationen finden Sie in der Aktion SetTopicAttributes in der API-Referenz zu Amazon Simple Notification Service.

Amazon-SQS-Berechtigungen

Damit eine EventBridge-Regel eine Amazon-SQS-Warteschlange aufrufen kann, verwenden Sie die Befehle `aws sqs get-queue-attributes` und `aws sqs set-queue-attributes`.

Wenn die Richtlinie für die SQS-Warteschlange leer ist, müssen Sie zuerst eine Richtlinie erstellen und dann können Sie ihr die Berechtigungsanweisung hinzufügen. Eine neue SQS-Warteschlange verfügt über eine leere Richtlinie.

Wenn die SQS-Warteschlange bereits über eine Richtlinie verfügt, müssen Sie die ursprüngliche Richtlinie kopieren und sie mit einer neuen Anweisung kombinieren, um ihr die Berechtigungsanweisung hinzuzufügen.

So fügen Sie Berechtigungen hinzu, mit denen EventBridge-Regeln eine SQS-Warteschlange aufrufen können

1. So listen Sie SQS-Warteschlangenattribute auf. Geben Sie in der Eingabeaufforderung den folgenden Befehl ein.

```
aws sqs get-queue-attributes \
--queue-url https://sqs.region.amazonaws.com/account-id/queue-name \
--attribute-names Policy
```

2. Fügen Sie die folgende Anweisung hinzu.

```
{
      "Sid": "AWSEvents_custom-eventbus-ack-sqs-rule_dlq_sqs-rule-target",
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Action": "sqs:SendMessage",
      "Resource": "arn:aws:sqs:region:account-id:queue-name",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": "arn:aws:events:region:account-id:rule/bus-name/rule-
name"
        }
      }
    }
```

3. Verwenden Sie einen JSON-in-Zeichenfolgen-Konverter, um die vorherige Anweisung in eine Zeichenfolge zu konvertieren. Nachdem Sie die Richtlinie in eine Zeichenfolge konvertiert haben, sieht sie wie folgt aus.

```
{\"Sid\": \"EventsToMyQueue\", \"Effect\": \"Allow\", \"Principal\": {\"Service
\": \"events.amazonaws.com\"}, \"Action\": \"sqs:SendMessage\", \"Resource\":
 \"arn:aws:sqs:region:account-id:queue-name\", \"Condition\": {\"ArnEquals\":
 {\"aws:SourceArn\": \"arn:aws:events:region:account-id:rule/rule-name\"}}
```

4. Erstellen Sie eine Datei mit dem Namen `set-queue-attributes.json` und folgendem Inhalt.

```
{
    "Policy": "{\"Version\":\"2012-10-17\",\"Id\":\"arn:aws:sqs:region:account-
id:queue-name/SQSDefaultPolicy\",\"Statement\":[{\"Sid\": \"EventsToMyQueue\",
 \"Effect\": \"Allow\", \"Principal\": {\"Service\": \"events.amazonaws.com\"},
 \"Action\": \"sqs:SendMessage\", \"Resource\": \"arn:aws:sqs:region:account-
id:queue-name\", \"Condition\": {\"ArnEquals\": {\"aws:SourceArn\":
 \"arn:aws:events:region:account-id:rule/rule-name\"}}}]}"
}
```

5. Legen Sie das Richtlinienattribut fest, indem Sie die soeben erstellte `set-queue-attributes.json`-Datei als Eingabe wie im folgenden Befehl gezeigt verwenden.

```
aws sqs set-queue-attributes \
--queue-url https://sqs.region.amazonaws.com/account-id/queue-name \
--attributes file://set-queue-attributes.json
```

Weitere Informationen finden Sie unter Beispiele für Amazon-SQS-Richtlinien im Entwicklerhandbuch für Amazon Simple Queue Service.

Einzelheiten zu EventBridge Pipes

EventBridge Pipes unterstützt keine ressourcenbasierten Richtlinien und hat keine APIs, die ressourcenbasierte Richtlinienbedingungen unterstützen.

Dienstübergreifende Confused-Deputy-Prävention

Das Confused-Deputy-Problem ist ein Sicherheitsproblem, bei dem eine juristische Stelle, die nicht über die Berechtigung zum Ausführen einer Aktion verfügt, eine privilegiertere juristische Stelle zwingen kann, die Aktion auszuführen. In AWS kann der dienstübergreifende Identitätswechsel zu

Confused-Deputy-Problem führen. Ein dienstübergreifender Identitätswechsel kann auftreten, wenn ein Dienst (der Anruf-Dienst) einen anderen Dienst anruft (den aufgerufenen Dienst). Der aufrufende Service kann manipuliert werden, um seine Berechtigungen zu verwenden, um Aktionen auf die Ressourcen eines anderen Kunden auszuführen, für die er sonst keine Zugriffsberechtigung haben sollte. Um dies zu verhindern, bietet AWS Tools, mit denen Sie Ihre Daten für alle Services mit Serviceprinzipalen schützen können, die Zugriff auf Ressourcen in Ihrem Konto erhalten haben.

Wir empfehlen die Verwendung der globalen Bedingungskontextschlüssel `aws:SourceArn` und `aws:SourceAccount` in ressourcenbasierten Richtlinien, um die Berechtigungen, die Amazon EventBridge einem anderen Service erteilt, auf eine bestimmte Ressource zu beschränken. Verwenden Sie `aws:SourceArn`, wenn Sie nur eine Ressource mit dem betriebsübergreifenden Zugriff verknüpfen möchten. Verwenden Sie `aws:SourceAccount`, wenn Sie zulassen möchten, dass Ressourcen in diesem Konto mit der betriebsübergreifenden Verwendung verknüpft werden.

Der effektivste Weg, um sich vor dem Confused-Deputy-Problem zu schützen, ist die Verwendung des globalen Bedingungskontextschlüssels `aws:SourceArn` mit dem vollständigen ARN der Ressource. Wenn Sie den vollständigen ARN der Ressource nicht kennen oder wenn Sie mehrere Ressourcen angeben, verwenden Sie den globalen Kontextbedingungsschlüssel `aws:SourceArn` mit Platzhalterzeichen (*) für die unbekannten Teile des ARN. Zum Beispiel `arn:aws:`*`servicename`*`:*:`*`123456789012`*`:*`.

Wenn der `aws:SourceArn`-Wert die Konto-ID nicht enthält, z. B. einen Amazon-S3-Bucket-ARN, müssen Sie beide globale Bedingungskontextschlüssel verwenden, um Berechtigungen einzuschränken.

Ereignisbusse

Für EventBridge-Event-Bus-Regelziele muss der Wert von `aws:SourceArn` der Regel-ARN sein.

Das folgende Beispiel zeigt, wie Sie die globalen Bedingungskontextschlüssel `aws:SourceArn` und `aws:SourceAccount` in EventBridge verwenden können, um das Problem des verwirrten Stellvertreters zu vermeiden. Dieses Beispiel findet Verwendung in einer Rollenvertrauensrichtlinie für eine Rolle, die von einer EventBridge-Regel verwendet wird.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
```

```
      "Service": "events.amazonaws.com"
    },
    "Action": "sts:AssumeRole"
    ],
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:events:*:123456789012:rule/myRule"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

EventBridge Pipes

Für EventBridge Pipes muss der Wert von `aws:SourceArn` der Pipe-ARN sein.

Das folgende Beispiel zeigt, wie Sie die globalen Bedingungskontextschlüssel `aws:SourceArn` und `aws:SourceAccount` in EventBridge verwenden können, um das Problem des verwirrten Stellvertreters zu vermeiden. Dieses Beispiel findet Verwendung in einer Rollenvertrauensrichtlinie für eine Rolle, die von EventBridge Pipes verwendet wird.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "events.amazonaws.com"
    },
    "Action": "sts:AssumeRole"
    ],
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:pipe:*:123456789012::pipe/example"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
```

```
}
```

Ressourcenbasierte Richtlinien für Amazon-EventBridge-Schemata

Die Amazon-EventBridge-Schemaregistrierung unterstützt ressourcenbasierte Richtlinien. Eine ressourcenbasierte Richtlinie ist eine Richtlinie, die an eine Ressource gebunden ist und nicht an eine IAM-Identität. Beispielsweise ist in Amazon Simple Storage Service (Amazon S3) eine Ressourcenrichtlinie an einen Amazon-S3-Bucket angefügt.

Weitere Informationen zu EventBridge-Schemata und ressourcenbasierten Richtlinien finden Sie im Folgenden.

- REST-API-Referenz zu Amazon-EventBridge-Schemata
- Identitätsbasierte Richtlinien und ressourcenbasierte Richtlinien im IAM-Benutzerhandbuch

Unterstützte APIs für ressourcenbasierte Richtlinien

Sie können die folgenden APIs mit ressourcenbasierten Richtlinien für die EventBridge-Schemaregistrierung verwenden.

- `DescribeRegistry`
- `UpdateRegistry`
- `DeleteRegistry`
- `ListSchemas`
- `SearchSchemas`
- `DescribeSchema`
- `CreateSchema`
- `DeleteSchema`
- `UpdateSchema`
- `ListSchemaVersions`
- `DeleteSchemaVersion`
- `DescribeCodeBinding`
- `GetCodeBindingSource`
- `PutCodeBinding`

Beispiel für eine Richtlinie, die einem AWS-Konto alle unterstützten Aktionen gewährt

Für die EventBridge-Schemaregistrierung müssen Sie immer eine ressourcenbasierte Richtlinie an eine Registrierung anhängen. Um Zugriff auf ein Schema zu gewähren, geben Sie den Schema-ARN und den Registrierungs-ARN in der Richtlinie an.

Um einem Benutzer Zugriff auf alle verfügbaren APIs für EventBridge-Schemata zu gewähren, verwenden Sie eine Richtlinie ähnlich der folgenden und ersetzen Sie den `"Principal"` durch die Konto-ID des Kontos, dem Sie Zugriff gewähren möchten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Test",
            "Effect": "Allow",
            "Action": [
                "schemas:*"
            ],
            "Principal": {
                "AWS": [
                    "109876543210"
                ]
            },
            "Resource": [
                "arn:aws:schemas:us-east-1:012345678901:registry/default",
                "arn:aws:schemas:us-east-1:012345678901:schema/default*"
            ]
        }
    ]
}
```

Beispielrichtlinie, die einem AWS-Konto schreibgeschützte Aktionen gewährt

Das folgende Beispiel gewährt Zugriff auf ein Konto nur für die schreibgeschützten APIs für EventBridge-Schemata.

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

```
            "Effect": "Allow",
            "Action": [
                "schemas:DescribeRegistry",
                "schemas:ListSchemas",
                "schemas:SearchSchemas",
                "schemas:DescribeSchema",
                "schemas:ListSchemaVersions",
                "schemas:DescribeCodeBinding",
                "schemas:GetCodeBindingSource"
            ],
            "Principal": {
                "AWS": [
                    "109876543210"
                ]
            },
            "Resource": [
                "arn:aws:schemas:us-east-1:012345678901:registry/default",
                "arn:aws:schemas:us-east-1:012345678901:schema/default*"
            ]
        }
    ]
}
```

Beispielrichtlinie, die einer Organisation alle Aktionen gewährt

Sie können ressourcenbasierte Richtlinien mit der EventBridge-Schemaregistrierung verwenden, um einer Organisation Zugriff zu gewähren. Weitere Informationen finden Sie im AWS Organizations-Benutzerhandbuch. Im folgenden Beispiel wird einer Organisation mit der ID `o-a1b2c3d4e5` Zugriff auf die Schemaregistrierung gewährt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Test",
            "Effect": "Allow",
            "Action": [
                "schemas:*"
            ],
            "Principal": "*",
            "Resource": [
                "arn:aws:schemas:us-east-1:012345678901:registry/default",
                "arn:aws:schemas:us-east-1:012345678901:schema/default*"
```

```
            ],
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": [
                        "o-a1b2c3d4e5"
                    ]
                }
            }
        }
    ]
}
```

Referenz zu Amazon-EventBridge-Berechtigungen

Um eine Aktion in einer EventBridge-Richtlinie anzugeben, verwenden Sie das Präfix `events:` gefolgt vom Namen der API-Operation, wie im folgenden Beispiel gezeigt.

```
"Action": "events:PutRule"
```

Um mehrere -Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie folgendermaßen durch Kommas.

```
"Action": ["events:action1", "events:action2"]
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen wie folgt festlegen, die mit dem Wort `"Put"` beginnen.

```
"Action": "events:Put*"
```

Wenn Sie alle EventBridge-API-Aktionen angeben möchten, verwenden Sie den Platzhalter * folgendermaßen.

```
"Action": "events:*"
```

In der folgenden Tabelle sind die EventBridge-API-Operationen und die entsprechenden Aktionen aufgeführt, die Sie in einer IAM-Richtlinie angeben können.

EventBridge-API-Operation	Erforderliche Berechtigungen	Beschreibung
DeleteRule	`events:DeleteRule`	Erforderlich zum Löschen einer Regel.
DescribeEventBus	`events:DescribeEventBus`	Erforderlich, um Konten aufzulisten, die Ereignisse in den Ereignisbus des aktuellen Kontos schreiben dürfen.
DescribeRule	`events:DescribeRule`	Erforderlich zum Auflisten der Details einer Regel.

EventBridge-API-Operation	Erforderliche Berechtigungen	Beschreibung
DisableRule	`events:DisableRule`	Erforderlich zum Deaktivieren einer Regel.
EnableRule	`events:EnableRule`	Erforderlich zum Aktivieren einer Regel.
ListRuleNamesByTarget	`events:ListRuleNamesByTarget`	Erforderlich zum Auflisten von Regeln, die mit einem Ziel verknüpft sind.
ListRules	`events:ListRules`	Erforderlich zum Auflisten aller Regeln in Ihrem Konto.
ListTagsForResource	`events:ListTagsForResource`	Erforderlich zum Auflisten aller Tags, die einer EventBridge-Ressource zugeordnet sind. Zurzeit können nur Regeln getaggt werden.
ListTargetsByRule	`events:ListTargetsByRule`	Erforderlich zum Auflisten aller Ziele im Zusammenhang mit einer Regel.
PutEvents	`events:PutEvents`	Erforderlich zum Hinzufügen von benutzerspezifischen Ereignissen, die Regeln zugeordnet werden können.
PutPermission	`events:PutPermission`	Erforderlich, um einem anderen Konto die Berechtigung zum Schreiben von Ereignissen in den Standardereignisbus dieses Kontos zu erteilen.
PutRule	`events:PutRule`	Erforderlich zum Erstellen oder Aktualisieren einer Regel.

EventBridge-API-Operation	Erforderliche Berechtigungen	Beschreibung
PutTargets	`events:PutTargets`	Erforderlich zum Hinzufügen von Zielen zu einer Regel.
RemovePermission	`events:RemovePermission`	Erforderlich, um einem anderen Konto die Berechtigungen zum Schreiben von Ereignissen in den Standardereignisbus dieses Kontos zu entziehen.
RemoveTargets	`events:RemoveTargets`	Erforderlich zum Entfernen eines Ziels aus einer Regel.
TestEventPattern	`events:TestEventPattern`	Erforderlich zum Testen eines Ereignismusters für ein bestimmtes Ereignis.

Verwenden von IAM-Richtlinienbedingungen für die differenzierte Zugriffskontrolle

Zum Erteilen von Berechtigungen geben Sie mithilfe der IAM-Richtliniensprache in einer Richtlinienanweisung die Bedingungen an, unter denen die Richtlinie wirksam werden soll. Beispielsweise kann festgelegt werden, dass eine Richtlinie erst ab einem bestimmten Datum gilt.

Eine Bedingung in einer Richtlinie besteht aus Schlüssel-Wert-Paaren. Bedingungsschlüssel unterscheiden nicht zwischen Groß- und Kleinschreibung.

Wenn Sie mehrere Bedingungen oder Schlüssel in einer einzelnen Bedingung angeben, müssen alle Bedingungen und Schlüssel erfüllt sein, damit EventBridge die Berechtigung erteilt. Wenn Sie eine einzelne Bedingung mit mehreren Werten für einen Schlüssel angeben, wird von EventBridge eine Genehmigung erteilt, wenn einer der Werte erfüllt ist.

Sie können Platzhalter oder Richtlinienvariablen verwenden, wenn Sie Bedingungen angeben. Weitere Informationen finden Sie unter Richtlinienvariablen im IAM-Benutzerhandbuch. Weitere Informationen zum Angeben von Bedingungen in einer IAM-Richtliniensprache finden Sie unter Bedingung im IAM-Benutzerhandbuch.

Standardmäßig können IAM-Benutzer und -Rollen nicht auf die Ereignisse in Ihrem Konto zugreifen. Damit Benutzer auf Ereignisse zugreifen können, müssen sie für die `PutRule`-API-Aktion autorisiert sein. Wenn IAM-Benutzer oder -Rollen für die `events:PutRule`-Aktion autorisiert sind, können diese eine Regel erstellen, die mit bestimmten Ereignissen übereinstimmt. Damit die Regel jedoch nützlich ist, muss der Benutzer auch über Berechtigungen für die `events:PutTargets`-Aktion verfügen, denn wenn Sie möchten, dass die Regel mehr kann als eine CloudWatch-Metrik zu veröffentlichen, müssen Sie auch ein Ziel zu einer Regel hinzufügen.

Sie können eine Bedingung in der Richtlinienanweisung eines IAM-Benutzers oder einer IAM-Rolle bereitstellen, mit der der Benutzer oder die Rolle eine Regel erstellen kann, die nur für eine bestimmte Gruppe von Quellen und bestimmte Ereignistypen gilt. Zum Gewähren von Zugriff auf bestimmte Quellen und Ereignistypen verwenden Sie die Bedingungsschlüssel `events:source` und `events:detail-type`.

Ebenso können Sie eine Bedingung in der Richtlinienanweisung eines IAM-Benutzers oder einer IAM-Rolle bereitstellen, mit der der Benutzer oder die Rolle eine Regel erstellen kann, die nur für eine bestimmte Ressource in Ihren Konten gilt. Zum Gewähren von Zugriff auf eine bestimmte Ressource verwenden Sie den Bedingungsschlüssel `events:TargetArn`.

Das folgende Beispiel ist eine Richtlinie, die es Benutzern ermöglicht, mithilfe einer Verweigerungsanweisung für die `PutRule`-API-Aktion auf alle Ereignisse außer Amazon-EC2-Ereignissen in EventBridge zuzugreifen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyPutRuleForAllEC2Events",
            "Effect": "Deny",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:source": "aws.ec2"
                }
            }
        }
    ]
}
```

EventBridge-Bedingungsschlüssel

Die folgende Tabelle zeigt die Bedingungsschlüssel und Schlüssel-Wert-Paare, die Sie in einer Richtlinie in EventBridge verwenden können.

Bedingungsschlüssel	Schlüssel-Wert-Paar	Bewertungstypen
aws:SourceAccount	Das Konto, in dem die von `aws:SourceArn` angegebene Regel vorhanden ist.	Account Id, Null
aws:SourceArn	Der ARN der Regel, die das Ereignis sendet.	ARN, Null
events:creatorAccount	`"events:creatorAccount":"` *`creatorAccount`* `"` Verwenden Sie für *`creatorAccount`* die Konto-ID für das Konto, das die Regel erstellt hat.	creatorAccount, Null

Bedingungsschlüssel	Schlüssel-Wert-Paar	Bewertungstypen
	Verwenden Sie diese Bedingung, um API-Aufrufe für Regeln aus einem bestimmten Konto zu autorisieren.	
events:detail-type	`"events:detail-type":"` *`detail-type`* `"` Dabei ist *`detail-type`* die Literalzeichenfolge für das Feld detail-type des Ereignisses, wie z. B. `"AWS API Call via CloudTrail"` und `"EC2 Instance State-change Notification"` .	Detailtyp, Null
events:detail.eventTypeCode	`"events:detail.eventTypeCode":"` *`eventTypeCode`* `"` Verwenden Sie für *`eventTypeCode`* die Literalzeichenfolge für das Feld detail.eventTypeCode des Ereignisses, wie z. B. `"AWS_ABUSE_DOS_REPORT"` .	eventTypeCode, Null
events:detail.service	`"events:detail.service":"` *`service`*`"` Verwenden Sie für *`service`* die Literalzeichenfolge für das Feld detail.service des Ereignisses, wie z. B. `"ABUSE"`.	service, Null

Bedingungsschlüssel	Schlüssel-Wert-Paar	Bewertungstypen
events:detail.user Identity.principalId	`"events:detail.use rIdentity.principa lId":" principal-id "` Verwenden Sie für *principal -id* die Literalzeichenfolge für das Feld detail.userIdentit y.principalId des Ereignisses mit dem Detailtyp `"AWS API Call via CloudTrail"` , z. B. `"AROAIDPPEZS35WEXA MPLE:AssumedRoleSe ssionName."` .	Principal-ID, Null
events:eventBusInv ocation	`"events:eventBusIn vocation":" boolean"` Verwenden Sie für *boolean* true, wenn eine Regel ein Ereignis an ein Ziel sendet, bei dem es sich um einen Event Bus in einem anderen Konto handelt. Verwenden Sie false, wenn ein `PutEvents` -API-Aufruf verwendet wird.	eventBusInvocation, Null
events:ManagedBy	Wird intern von AWS-Services verwendet. Für eine Regel, die von einem AWS-Service in Ihrem Namen erstellt wird, ist der Wert der Prinzipalname des Service, der die Regel erstellt hat.	Nicht für die Verwendung in Kundenrichtlinien vorgesehen.

Bedingungsschlüssel	Schlüssel-Wert-Paar	Bewertungstypen
events:source	`"events:source":" source "` Verwenden Sie für *source* die Literalzeichenfolge für das Quellfeld des Ereignisses, wie z. B. `"aws.ec2"` und `"aws.s3"`. Weitere mögliche Werte für *source* finden Sie in den Beispielereignissen in Ereignisse im Zusammenhang mit Dienstleistungen AWS.	Quelle, Null
events:TargetArn	`"events:TargetArn" :" target-arn "` Verwenden Sie für *target-arn* den ARN des Ziels für die Regel, z. B. `"arn:aws:lambda:*:*:function:*"` .	ArrayOfARN, Null

Beispiele mit Richtlinienanweisungen für EventBridge finden Sie unter Verwalten der Zugriffsberechtigungen für Ihre Amazon-EventBridge-Ressourcen.

Themen

- Einzelheiten zu EventBridge Pipes
- Beispiel: Verwenden der Bedingung creatorAccount
- Beispiel: Verwenden der Bedingung eventBusInvocation
- Beispiel: Einschränken des Zugriffs auf eine bestimmte Quelle
- Beispiel: Definieren mehrerer Quellen, die einzeln in einem Ereignismuster verwendet werden können
- Beispiel: Definieren einer Quelle und eines DetailType zur Verwendung in einem Ereignismuster
- Beispiel: Sicherstellen, dass die Quelle im Ereignismuster definiert ist

- Beispiel: Definieren einer Liste der zulässigen Quellen in einem Ereignismuster mit mehreren Quellen
- Beispiel: Beschränken des PutRule-Zugriffs durch detail.service
- Beispiel: Beschränken des PutRule-Zugriffs durch detail.eventTypeCode
- Beispiel: Sicherstellen, dass nur AWS CloudTrail-Ereignisse für API-Aufrufe von einer bestimmten PrincipalId zulässig sind
- Beispiel: Einschränken des Zugriffs auf Ziele

Einzelheiten zu EventBridge Pipes

EventBridge Pipes unterstützt keine zusätzlichen IAM-Richtlinienbedingungsschlüssel.

Beispiel: Verwenden der Bedingung **`creatorAccount`**

Das folgende Beispiel für eine Richtlinienanweisung zeigt, wie die Bedingung `creatorAccount` in einer Richtlinie verwendet wird, um die Erstellung von Regeln nur zuzulassen, wenn das als `creatorAccount` angegebene Konto das Konto ist, das die Regel erstellt hat.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleForOwnedRules",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "events:creatorAccount": "${aws:PrincipalAccount}"
                }
            }
        }
    ]
}
```

Beispiel: Verwenden der Bedingung **`eventBusInvocation`**

Der `eventBusInvocation` gibt an, ob der Aufruf von einem kontoübergreifenden Ziel oder einer `PutEvents`-API-Anfrage stammt. Der Wert ist true, wenn der Aufruf aus einer Regel resultiert,

die ein kontoübergreifendes Ziel beinhaltet, z. B. wenn es sich bei dem Ziel um einen Event Bus in einem anderen Konto handelt. Der Wert ist false, wenn der Aufruf aus einer `PutEvents`-API-Anfrage resultiert. Das folgende Beispiel zeigt einen Aufruf von einem kontoübergreifenden Ziel an.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCrossAccountInvocationEventsOnly",
      "Effect": "Allow",
      "Action": "events:PutEvents",
      "Resource": "*",
      "Condition": {
        "BoolIfExists": {
          "events:eventBusInvocation": "true"
        }
      }
    }
  ]
}
```

Beispiel: Einschränken des Zugriffs auf eine bestimmte Quelle

Folgende Beispielrichtlinien können einem IAM-Benutzer zugeordnet werden. Richtlinie A ermöglicht die `PutRule`-API-Aktion für alle Ereignisse, während Richtlinie B `PutRule` nur dann zulässt, wenn das Ereignismuster der erstellten Regel mit den Amazon-EC2-Ereignissen übereinstimmt.

Richtlinie A: alle Ereignisse zulassen

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleForAllEvents",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*"
        }
    ]
    }
```

Richtlinie B: nur Ereignisse von Amazon EC2 zulassen

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleForAllEC2Events",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:source": "aws.ec2"
                }
            }
        }
    ]
}
```

`EventPattern` ist ein obligatorisches Argument für `PutRule`. Wenn der Benutzer mit Richtlinie B `PutRule` mit einem Ereignismuster wie dem folgenden aufruft, gilt daher Folgendes.

```
{
    "source": [ "aws.ec2" ]
}
```

Die Regel würde erstellt werden, da die Richtlinie diese bestimmte Quelle zulässt, d. h. `"aws.ec2"`. Wenn der Benutzer mit Richtlinie B jedoch `PutRule` mit einem Ereignismuster wie dem folgenden aufruft, wird die Erstellung der Regel abgelehnt, da die Richtlinie diese bestimmte Quelle nicht zulässt, d. h. `"aws.s3"`.

```
{
    "source": [ "aws.s3" ]
}
```

Im Wesentlichen darf der Benutzer mit Richtlinie B nur eine Regel erstellen, die mit den Ereignissen aus Amazon EC2 übereinstimmt, weshalb er nur Zugriff auf die Ereignisse aus Amazon EC2 erhält.

In der folgenden Tabelle finden Sie einen Vergleich von Richtlinie A und Richtlinie B.

Ereignismuster	Zulässig durch Richtlinie A	Zulässig durch Richtlinie B

<pre>{
"source":
["aws.ec2"]
}</pre>
	Ja	Ja

<pre>{
"source":
["aws.ec2",
"aws.s3"]
}</pre>
	Ja	Nein (die Quelle aws.s3 ist nicht zulässig)

<pre>{
"source":
["aws.ec2"],
"detail-type":
["EC2 Instance
State-change
Notification"]
}</pre>
	Ja	Ja

<pre>{
"detail-type":
["EC2 Instance
State-change
Notification"]
}</pre>
	Ja	Keine (Quelle muss angegeben werden)

Beispiel: Definieren mehrerer Quellen, die einzeln in einem Ereignismuster verwendet werden können

Die folgende Richtlinie ermöglicht es einem IAM-Benutzer oder einer IAM-Rolle, eine Regel zu erstellen, deren Quelle im `EventPattern` entweder Amazon EC2 oder Amazon ECS ist.

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

```
    "Statement": [
        {
            "Sid": "AllowPutRuleIfSourceIsEC2OrECS",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:source": [ "aws.ec2", "aws.ecs" ]
                }
            }
        }
    ]
}
```

Die folgende Tabelle zeigt einige Beispiele für Ereignismuster, die durch diese Richtlinie zugelassen oder abgelehnt werden.

Ereignismuster	Zulässig durch die Richtlinie
`{ "source": [ "aws.ec2" ] }`	Ja
`{ "source": [ "aws.ecs" ] }`	Ja
`{ "source": [ "aws.s3" ] }`	Nein
`{ "source": [ "aws.ec2", "aws.ecs" ] }`	Nein
`{`	Nein

Ereignismuster	Zulässig durch die Richtlinie

<pre>{
"detail-type": ["AWS API Call via CloudTrail"]
}</pre>	

Beispiel: Definieren einer Quelle und eines **`DetailType`** zur Verwendung in einem Ereignismuster

Mit der folgenden Richtlinie werden nur Ereignisse von der `aws.ec2`-Quelle mit dem `DetailType` gleich `EC2 instance state change notification` zugelassen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid":
 "AllowPutRuleIfSourceIsEC2AndDetailTypeIsInstanceStateChangeNotification",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:source": "aws.ec2",
                    "events:detail-type": "EC2 Instance State-change Notification"
                }
            }
        }
    ]
}
```

Die folgende Tabelle zeigt einige Beispiele für Ereignismuster, die durch diese Richtlinie zugelassen oder abgelehnt werden.

Ereignismuster	Zulässig durch die Richtlinie

<pre>{
"source": ["aws.ec2"]
}</pre>
	Nein

Ereignismuster	Zulässig durch die Richtlinie
`{ "source": [ "aws.ecs" ] }`	Nein
`{ "source": [ "aws.ec2" ], "detail-type": [ "EC2 Instance State-change Notification" ] }`	Ja
`{ "source": [ "aws.ec2" ], "detail-type": [ "EC2 Instance Health Failed" ] }`	Nein
`{ "detail-type": [ "EC2 Instance State-change Notification" ] }`	Nein

Beispiel: Sicherstellen, dass die Quelle im Ereignismuster definiert ist

Die folgende Richtlinie ermöglicht Benutzern nur das Erstellen von Regeln mit `EventPatterns`, die über ein Quellfeld verfügen. Mit dieser Richtlinie können IAM-Benutzer oder IAM-Rollen keine Regel mit einem `EventPattern` erstellen, das keine bestimmte Quelle angibt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleIfSourceIsSpecified",
            "Effect": "Allow",
            "Action": "events:PutRule",
```

```
            "Resource": "*",
            "Condition": {
                "Null": {
                    "events:source": "false"
                }
            }
        }
    ]
}
```

Die folgende Tabelle zeigt einige Beispiele für Ereignismuster, die durch diese Richtlinie zugelassen oder abgelehnt werden.

Ereignismuster	Zulässig durch die Richtlinie
`{ "source": [ "aws.ec2" ], "detail-type": [ "EC2 Instance State-change Notification" ] }`	Ja
`{ "source": [ "aws.ecs", "aws.ec2" ] }`	Ja
`{ "detail-type": [ "EC2 Instance State-change Notification" ] }`	Nein

Beispiel: Definieren einer Liste der zulässigen Quellen in einem Ereignismuster mit mehreren Quellen

Die folgende Richtlinie ermöglicht Benutzern das Erstellen von Regeln mit `EventPatterns`, die über ein Quellfeld verfügen. Jede Quelle im Ereignismuster muss Mitglied der in der Bedingung

angegebenen Liste sein. Wenn Sie die Bedingung `ForAllValues` verwenden, müssen Sie mindestens eines der Elemente in der Bedingungsliste definieren.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleIfSourceIsSpecifiedAndIsEitherS3OrEC2OrBoth",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "events:source": [ "aws.ec2", "aws.s3" ]
                },
                "Null": {
                    "events:source": "false"
                }
            }
        }
    ]
}
```

Die folgende Tabelle zeigt einige Beispiele für Ereignismuster, die durch diese Richtlinie zugelassen oder abgelehnt werden.

Ereignismuster	Zulässig durch die Richtlinie
`{ "source": [ "aws.ec2" ] }`	Ja
`{ "source": [ "aws.ec2", "aws.s3" ] }`	Ja
`{ "source": [ "aws.ec2", "aws.autoscaling" ]`	Nein

Ereignismuster	Zulässig durch die Richtlinie
} 	

{
"detail-type": ["EC2
Instance State-change Notificat
ion"]
}
	Nein

Beispiel: Beschränken des **`PutRule`**-Zugriffs durch **`detail.service`**

Sie können einen IAM-Benutzer oder eine IAM-Rolle zum Erstellen von Regeln nur für Ereignisse beschränken, die einen bestimmten Wert im Feld `events:details.service` aufweisen. Der Wert von `events:details.service` ist nicht unbedingt der Name eines AWS-Service.

Diese Richtlinienbedingung ist hilfreich, wenn Sie mit Ereignissen von AWS Health arbeiten, die sich auf Sicherheit oder Missbrauch beziehen. Durch die Verwendung dieser Richtlinienbedingung können Sie den Zugriff auf diese sensiblen Warnungen auf ausschließlich diejenigen Benutzer einschränken, die diese unbedingt sehen müssen.

Beispiel: Die folgende Richtlinie ermöglicht das Erstellen von Regeln nur für Ereignisse, in denen der Wert von `events:details.service` ABUSE ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleEventsWithDetailServiceEC2",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:detail.service": "ABUSE"
                }
            }
        }
    ]
}
```

Beispiel: Beschränken des **`PutRule`**-Zugriffs durch **`detail.eventTypeCode`**

Sie können einen IAM-Benutzer oder eine IAM-Rolle zum Erstellen von Regeln nur für Ereignisse beschränken, die einen bestimmten Wert im Feld `events:details.eventTypeCode` aufweisen. Diese Richtlinienbedingung ist hilfreich, wenn Sie mit Ereignissen von AWS Health arbeiten, die sich auf Sicherheit oder Missbrauch beziehen. Durch die Verwendung dieser Richtlinienbedingung können Sie den Zugriff auf diese sensiblen Warnungen auf ausschließlich diejenigen Benutzer einschränken, die diese unbedingt sehen müssen.

Beispiel: Die folgende Richtlinie ermöglicht das Erstellen von Regeln nur für Ereignisse, in denen der Wert von `events:details.eventTypeCode` `AWS_ABUSE_DOS_REPORT` ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutRuleEventsWithDetailServiceEC2",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:detail.eventTypeCode": "AWS_ABUSE_DOS_REPORT"
                }
            }
        }
    ]
}
```

Beispiel: Sicherstellen, dass nur AWS CloudTrail-Ereignisse für API-Aufrufe von einer bestimmten **`PrincipalId`** zulässig sind

Alle AWS CloudTrail-Ereignisse haben die PrincipalId des Benutzers, der den API-Aufruf im `detail.userIdentity.principalId`-Pfad eines Ereignisses ausgeführt hat. Mithilfe des `events:detail.userIdentity.principalId`-Bedingungsschlüssels können Sie den Zugriff durch IAM-Benutzer oder IAM-Rollen auf CloudTrail-Ereignisse nur für Benutzer oder Rollen eines bestimmten Kontos einschränken.

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

```
    "Statement": [
        {
            "Sid": "AllowPutRuleOnlyForCloudTrailEventsWhereUserIsASpecificIAMUser",
            "Effect": "Allow",
            "Action": "events:PutRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:detail-type": [ "AWS API Call via CloudTrail" ],
                    "events:detail.userIdentity.principalId":
 [ "AIDAJ45Q7YFFAREXAMPLE" ]
                }
            }
        }
    ]
}
```

Die folgende Tabelle zeigt einige Beispiele für Ereignismuster, die durch diese Richtlinie zugelassen oder abgelehnt werden.

Ereignismuster	Zulässig durch die Richtlinie
`{ "detail-type": [ "AWS API Call via CloudTrail" ] }`	Nein
`{ "detail-type": [ "AWS API Call via CloudTrail" ], "detail.userIdentity.principalId": [ "AIDAJ45Q7YFFAREXAMPLE" ] }`	Ja
`{ "detail-type": [ "AWS API Call via CloudTrail" ], "detail.userIdentity.principalId": [ "AROAIDPPEZS35WEXA`	Nein

Ereignismuster	Zulässig durch die Richtlinie
MPLE:AssumedRoleSessionName "] }	

Beispiel: Einschränken des Zugriffs auf Ziele

Wenn ein IAM-Benutzer oder eine IAM-Rolle `events:PutTargets`-berechtigt ist, kann er oder sie im selben Konto den Regeln, auf die er oder sie zugreifen kann, beliebige Ziele hinzufügen. Die folgende Richtlinie beschränkt Benutzer darauf, Ziele nur einer bestimmten Regel hinzuzufügen: `MyRule` im Konto `123456789012`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutTargetsOnASpecificRule",
            "Effect": "Allow",
            "Action": "events:PutTargets",
            "Resource": "arn:aws:events:us-east-1:123456789012:rule/MyRule"
        }
    ]
}
```

Mit dem `events:TargetArn`-Bedingungsschlüssel legen Sie fest, welches Ziel der Regel hinzugefügt werden darf. Sie haben die Möglichkeit, Ziele nur auf Lambda-Funktionen zu beschränken (siehe folgendes Beispiel).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutTargetsOnASpecificRuleAndOnlyLambdaFunctions",
            "Effect": "Allow",
            "Action": "events:PutTargets",
            "Resource": "arn:aws:events:us-east-1:123456789012:rule/MyRule",
            "Condition": {
                "ArnLike": {
                    "events:TargetArn": "arn:aws:lambda:*:*:function:*"
```

```
                }
            }
        }
    ]
}
```

Verwenden von serviceverknüpften Rollen für EventBridge

Amazon EventBridge verwendet serviceverknüpfte Rollen von AWS Identity and Access Management (IAM). Eine serviceverknüpfte Rolle ist ein spezieller Typ einer IAM-Rolle, die direkt mit EventBridge verknüpft ist. Serviceverknüpfte Rollen werden von EventBridge vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer AWS-Services in Ihrem Namen erfordert.

Themen

- Verwenden von Rollen zum Erstellen von Secrets für API-Ziele
- Verwenden von Rollen für die Schemaerkennung

Verwenden von Rollen zum Erstellen von Secrets für API-Ziele

Amazon EventBridge verwendet serviceverknüpfte Rollen von AWS Identity and Access Management (IAM). Eine serviceverknüpfte Rolle ist ein spezieller Typ einer IAM-Rolle, die direkt mit EventBridge verknüpft ist. Serviceverknüpfte Rollen werden von EventBridge vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer AWS-Services in Ihrem Namen erfordert.

Eine serviceverknüpfte Rolle vereinfacht die Einrichtung von EventBridge, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. EventBridge definiert die Berechtigungen seiner serviceverknüpften Rollen. Sofern keine andere Konfiguration festgelegt wurde, kann nur EventBridge die Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dies schützt Ihre EventBridge-Ressourcen, da Sie nicht versehentlich die Berechtigung für den Zugriff auf die Ressourcen entfernen können.

Informationen zu anderen Services, die serviceorientierte Rollen unterstützen, finden Sie unter AWS services that work with IAM (-Services, die mit IAM funktionieren). Suchen Sie nach den Services, für die Yes (Ja) in der Spalte Service-linked roles (Serviceorientierte Rollen) angegeben ist. Wählen Sie

über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Berechtigungen von serviceverknüpften Rollen für EventBridge

EventBridge verwendet die serviceverknüpfte Rolle namens AWSServiceRoleForAmazonEventBridgeApiDestinations – Ermöglicht den Zugriff auf die von erstellten Secrets-Manager-SecretsEventBridge.

Die serviceverknüpfte Rolle AWSServiceRoleForAmazonEventBridgeApiDestinations vertraut darauf, dass die folgenden Services die Rolle annehmen:

- `apidestinations.events.amazonaws.com`

Die Rollenberechtigungsrichtlinie namens AmazonEventBridgeApiDestinationsServiceRoleRichtlinie erlaubt EventBridge, die folgenden Aktionen für die angegebenen Ressourcen durchzuführen:

- Aktion: `create, describe, update and delete secrets; get and put secret values` für `secrets created for all connections by EventBridge`

Sie müssen Berechtigungen konfigurieren, damit eine Benutzer, Gruppen oder Rollen eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen können. Weitere Informationen finden Sie unter serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

Erstellen einer serviceverknüpften Rolle für EventBridge

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie eine Verbindung in der AWS CLI, AWS Management Consoleder oder der -AWSAPI erstellen, EventBridge erstellt die serviceverknüpfte Rolle für Sie.

> **Important**
>
> Diese serviceverknüpfte Rolle kann in Ihrem Konto erscheinen, wenn Sie eine Aktion in einem anderen Service abgeschlossen haben, der die von dieser Rolle unterstützten Features verwendet. Wenn Sie den EventBridge Service vor dem 11. Februar 2021 verwendet haben, als dieser serviceverknüpfte Rollen unterstützt hat, hat die AWSServiceRoleForAmazonEventBridgeApiDestinations Rolle in Ihrem Konto EventBridge erstellt. Weitere Informationen finden Sie unter In meinem AWS-Konto wird eine neue Rolle angezeigt.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie eine Verbindung erstellen, EventBridge erstellt die serviceverknüpfte Rolle erneut für Sie.

Bearbeiten einer serviceverknüpften Rolle für EventBridge

EventBridge verhindert die Bearbeitung der serviceverknüpften Rolle AWSServiceRoleForAmazonEventBridgeApiDestinations. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach dem Erstellen einer serviceverknüpften Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für EventBridge

Wenn Sie ein Feature oder einen Dienst, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

Bereinigen einer serviceverknüpften Rolle

Bevor mit IAM eine serviceverknüpfte Rolle löschen können, müssen Sie zunächst alle von der Rolle verwendeten Ressourcen löschen.

> **Note**
>
> Wenn der EventBridge-Service 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.

So löschen Sie die von AWSServiceRoleForAmazonEventBridgeApiDestinations verwendeten EventBridge-Ressourcen (Konsole)

1. Öffnen Sie die Amazon- EventBridge Konsole unter https://console.aws.amazon.com/events/.
2. Wählen Sie unter Integrationen die Option API-Ziele und dann die Registerkarte Verbindungen aus.
3. Wählen Sie die Verbindung und dann Löschen aus.

So löschen Sie die von AWSServiceRoleForAmazonEventBridgeApiDestinations verwendeten EventBridge-Ressourcen (AWS-CLI)

- Verwenden Sie den folgenden Befehl: `delete-connection`.

So löschen Sie die von AWSServiceRoleForAmazonEventBridgeApiDestinations verwendeten EventBridge-Ressourcen (API)

- Verwenden Sie den folgenden Befehl: `DeleteConnection`.

Manuelles Löschen der -serviceverknüpften Rolle

Verwenden Sie die IAM-Konsole, AWS CLI- oder AWS-API, um die AWSServiceRoleForAmazonEventBridgeApiDestinations serviceverknüpfte Rolle zu löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für serviceverknüpfte EventBridge-Rollen

EventBridge unterstützt die Verwendung von serviceverknüpften Rollen in allen Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter AWS Regionen und Endpunkte.

Verwenden von Rollen für die Schemaerkennung

Amazon EventBridge verwendet serviceverknüpfte Rollen von AWS Identity and Access Management (IAM). Eine serviceverknüpfte Rolle ist ein spezieller Typ einer IAM-Rolle, die direkt mit EventBridge verknüpft ist. Serviceverknüpfte Rollen werden von EventBridge vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer AWS-Services in Ihrem Namen erfordert.

Eine serviceverknüpfte Rolle vereinfacht die Einrichtung von EventBridge, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. EventBridge definiert die Berechtigungen seiner serviceverknüpften Rollen. Sofern keine andere Konfiguration festgelegt wurde, kann nur EventBridge die Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dies schützt Ihre EventBridge-Ressourcen, da Sie nicht versehentlich die Berechtigung für den Zugriff auf die Ressourcen entfernen können.

Informationen zu anderen Services, die serviceorientierte Rollen unterstützen, finden Sie unter AWS services that work with IAM (-Services, die mit IAM funktionieren). Suchen Sie nach den Services, für die Yes (Ja) in der Spalte Service-linked roles (Serviceorientierte Rollen) angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Berechtigungen von serviceverknüpften Rollen für EventBridge

EventBridge verwendet die serviceverknüpfte Rolle namens AWSServiceRoleForSchemas – Gewährt Berechtigungen für verwaltete -RegelnAmazon EventBridge, die von Schemata erstellt wurden.

Die serviceverknüpfte Rolle AWSServiceRoleForSchemas vertraut darauf, dass die folgenden Services die Rolle annehmen:

- `schemas.amazonaws.com`

Die Rollenberechtigungsrichtlinie namens AmazonEventBridgeSchemasServiceRolePolicy ermöglicht es EventBridge, die folgenden Aktionen für die angegebenen Ressourcen durchzuführen:

- Aktion: `put, enable, disable, and delete rules; put and remove targets; list targets per rule` für `all managed rules created by EventBridge`

Sie müssen Berechtigungen konfigurieren, damit eine Benutzer, Gruppen oder Rollen eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen können. Weitere Informationen finden Sie unter serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

Erstellen einer serviceverknüpften Rolle für EventBridge

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie eine Schemaerkennung in der AWS CLI, AWS Management Consoleder oder der -AWSAPI durchführen, EventBridge erstellt die serviceverknüpfte Rolle für Sie.

> **Important**
>
> Diese serviceverknüpfte Rolle kann in Ihrem Konto erscheinen, wenn Sie eine Aktion in einem anderen Service abgeschlossen haben, der die von dieser Rolle unterstützten Features verwendet. Wenn Sie den EventBridge Service vor dem 27. November 2019 verwendet haben, als er serviceverknüpfte Rollen unterstützt hat, hat

die AWSServiceRoleForSchemas Rolle in Ihrem Konto EventBridge erstellt. Weitere Informationen finden Sie unter In meinem AWS-Konto wird eine neue Rolle angezeigt.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie eine Schemaerkennung durchführen, EventBridge erstellt die serviceverknüpfte Rolle erneut für Sie.

Bearbeiten einer serviceverknüpften Rolle für EventBridge

EventBridge verhindert die Bearbeitung der serviceverknüpften Rolle AWSServiceRoleForSchemas. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach dem Erstellen einer serviceverknüpften Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für EventBridge

Wenn Sie ein Feature oder einen Dienst, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

Bereinigen einer serviceverknüpften Rolle

Bevor mit IAM eine serviceverknüpfte Rolle löschen können, müssen Sie zunächst alle von der Rolle verwendeten Ressourcen löschen.

Note

Wenn der EventBridge-Service 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.

So löschen Sie die von AWSServiceRoleForSchemas verwendeten EventBridge-Ressourcen (Konsole)

1. Öffnen Sie die Amazon- EventBridge Konsole unter https://console.aws.amazon.com/events/.

2. Wählen Sie unter Buses die Option Event Buses und dann einen Event Bus aus.
3. Wählen Sie Erkennung beenden aus.

So löschen Sie die von AWSServiceRoleForSchemas verwendeten EventBridge-Ressourcen (AWS-CLI)

- Verwenden Sie den folgenden Befehl: `delete-discoverer`.

So löschen Sie die von AWSServiceRoleForSchemas verwendeten EventBridge-Ressourcen (API)

- Verwenden Sie den folgenden Befehl: `DeleteDiscoverer`.

Manuelles Löschen der -serviceverknüpften Rolle

Verwenden Sie die IAM-Konsole, AWS CLI- oder AWS-API, um die AWSServiceRoleForSchemas serviceverknüpfte Rolle zu löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für serviceverknüpfte EventBridge-Rollen

EventBridge unterstützt die Verwendung von serviceverknüpften Rollen in allen Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter AWS Regionen und Endpunkte.

Protokollieren von Amazon EventBridge API-Aufrufen mit AWS CloudTrail

Amazon EventBridge ist in integriert, einem ServiceAWS CloudTrail, der die Aktionen eines Benutzers, einer Rolle oder eines aufzeichnet AWS-Service. CloudTrail erfasst alle API-Aufrufe für EventBridge als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der EventBridge Konsole und Codeaufrufe der EventBridge API-Operationen. Anhand der von CloudTrailgesammelten Informationen können Sie die an gestellte Anfrage EventBridge, die IP-Adresse, von der die Anfrage gestellt wurde, den Zeitpunkt der Anfrage und zusätzliche Details bestimmen.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Ob die Anfrage mit Anmeldeinformationen des Root-Benutzers oder des Benutzers gestellt wurde.
- Ob die Anforderung im Namen eines IAM-Identity-Center-Benutzers gestellt wurde.
- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer gesendet wurde.
- Ob die Anforderung aus einem anderen AWS-Service gesendet wurde.

CloudTrail ist in Ihrem aktiv AWS-Konto , wenn Sie das Konto erstellen und automatisch Zugriff auf den CloudTrail Ereignisverlauf haben. Der CloudTrail Ereignisverlauf bietet eine anzeigbare, durchsuchbare, herunterladbare und unveränderliche Aufzeichnung der aufgezeichneten Verwaltungsereignisse der letzten 90 Tage in einem AWS-Region. Weitere Informationen finden Sie unter Arbeiten mit dem CloudTrail Ereignisverlauf im AWS CloudTrail -Benutzerhandbuch. Für die Anzeige des Ereignisverlaufs fallen keine CloudTrail Gebühren an.

Erstellen Sie für eine fortlaufende Aufzeichnung der Ereignisse in den AWS-Konto letzten 90 Tagen einen Trail oder einen CloudTrail Lake-Ereignisdatenspeicher.

CloudTrail Trails

> Ein Trail ermöglicht CloudTrail die Bereitstellung von Protokolldateien an einen Amazon S3-Bucket. Alle Trails, die mit der erstellt wurden, AWS Management Console sind multiregional. Sie können einen Trail für eine einzelne Region oder für mehrere Regionen erstellen, indem Sie die verwenden AWS CLI. Das Erstellen eines multiregionalen Trails wird empfohlen, da Sie Aktivitäten in allen AWS-Regionen in Ihrem Konto erfassen. Wenn Sie einen Trail für eine einzelne Region erstellen, können Sie nur die Ereignisse anzeigen, die im des Trails

protokolliert wurden AWS-Region. Weitere Informationen zu Trails finden Sie unter Erstellen eines Trails für Ihr AWS-Konto und Erstellen eines Trails für eine Organisation im AWS CloudTrail - Benutzerhandbuch.

Sie können eine Kopie Ihrer laufenden Verwaltungsereignisse in Ihrem Amazon S3-Bucket kostenlos von bereitstellen, CloudTrail indem Sie einen Trail erstellen. Es fallen jedoch Amazon S3-Speichergebühren an. Weitere Informationen zu CloudTrail Preisen finden Sie unter -AWS CloudTrail Preise. Informationen zu Amazon-S3-Preisen finden Sie unter Amazon S3-Preise.

CloudTrail Lake-Ereignisdatenspeicher

Mit CloudTrail Lake können Sie SQL-basierte Abfragen für Ihre Ereignisse ausführen. CloudTrail Lake konvertiert vorhandene Ereignisse im zeilenbasierten JSON-Format in das Apache-ORC-Format. ORC ist ein spaltenförmiges Speicherformat, das für den schnellen Abruf von Daten optimiert ist. Die Ereignisse werden in Ereignisdatenspeichern zusammengefasst, bei denen es sich um unveränderliche Sammlungen von Ereignissen handelt, die auf Kriterien basieren, die Sie mit Hilfe von erweiterten Ereignisselektoren auswählen. Die Selektoren, die Sie auf einen Ereignisdatenspeicher anwenden, steuern, welche Ereignisse bestehen bleiben und für Sie zur Abfrage verfügbar sind. Weitere Informationen zu CloudTrail Lake finden Sie unter Arbeiten mit AWS CloudTrail Lake im AWS CloudTrail -Benutzerhandbuch.

CloudTrail Für Lake-Ereignisdatenspeicher und Abfragen fallen Kosten an. Beim Erstellen eines Ereignisdatenspeichers wählen Sie die Preisoption aus, die für den Ereignisdatenspeicher genutzt werden soll. Die Preisoption bestimmt die Kosten für die Erfassung und Speicherung von Ereignissen sowie die standardmäßige und maximale Aufbewahrungsdauer für den Ereignisdatenspeicher. Weitere Informationen zu CloudTrail Preisen finden Sie unter -AWS CloudTrail Preise.

EventBridge -Datenereignisse in CloudTrail

Datenereignisse liefern Informationen über die Ressourcenoperationen, die auf oder in einer Ressource ausgeführt werden (z. B. Lesen oder Schreiben in ein Amazon-S3-Objekt). Sie werden auch als Vorgänge auf Datenebene bezeichnet. Datenereignisse sind oft Aktivitäten mit hohem Volume. Standardmäßig protokolliert CloudTrail keine Datenereignisse. Der CloudTrail Ereignisverlauf zeichnet keine Datenereignisse auf.

Für Datenereignisse werden zusätzliche Gebühren fällig. Weitere Informationen zu CloudTrail Preisen finden Sie unter -AWS CloudTrail Preise.

Sie können Datenereignisse für die EventBridge Ressourcentypen mithilfe der - CloudTrail Konsole AWS CLI oder API CloudTrail -Operationen protokollieren. Weitere Informationen zum Protokollieren von Datenereignissen finden Sie unter Protokollieren von Datenereignissen mit der AWS Management Console und Protokollieren von Datenereignissen mit der AWS Command Line Interface im AWS CloudTrail -Benutzerhandbuch.

In der folgenden Tabelle sind die EventBridge Ressourcentypen aufgeführt, für die Sie Datenereignisse protokollieren können. In der Spalte Datenereignistyp (Konsole) wird der Wert angezeigt, der aus der Liste Datenereignistyp in der CloudTrail Konsole ausgewählt werden kann. Die Spalte resources.type value zeigt den `resources.type` Wert an, den Sie bei der Konfiguration erweiterter Ereignisselektoren mit der AWS CLI oder CloudTrail APIs angeben würden. Die Spalte Daten-APIs, die in protokolliert CloudTrail wurden, zeigt die API-Aufrufe an, die CloudTrail für den - Ressourcentyp protokolliert wurden.

Typ des Datenereignisses (Konsole)	resources.type-Wert	Daten-APIs, die bei protokolliert wurden CloudTrail
Event Bus	`AWS::Events::EventBus`	• DescribeEventBus
Event-Bus-Regel	`AWS::Events::Rule`	• DeleteRule • DescribeRule • DisableRule • EnableRule • ListRuleNamesByTarget • ListRules • ListTargetsByRule • PutRule • PutTargets • RemoveTargets • TestEventPattern
Pipe	`AWS::Pipes::Pipe`	• CreatePipe • DeletePipe • DescribePipe

Typ des Datenereignisses (Konsole)	resources.type-Wert	Daten-APIs, die bei protokolliert wurden CloudTrail
		• ListPipes • StartPipe • StopPipe • UpdatePipe

Sie können erweiterte Ereignisselektoren so konfigurieren, dass sie nach den `resources.`ARN Feldern `eventName`, und filtern`readOnly`, um nur die Ereignisse zu protokollieren, die für Sie wichtig sind. Weitere Informationen zu diesen Feldern finden Sie unter AdvancedFieldSelector in der APIAWS CloudTrail -Referenz zu .

EventBridge -Verwaltungsereignisse in CloudTrail

Verwaltungsereignisse liefern Informationen zu Verwaltungsvorgängen, die für Ressourcen in Ihrem ausgeführt werden AWS-Konto. Sie werden auch als Vorgänge auf Steuerebene bezeichnet. CloudTrail Protokolliert standardmäßig Verwaltungsereignisse.

Amazon EventBridge protokolliert alle Operationen auf EventBridge Steuerebene als Verwaltungsereignisse. Eine Liste der Operationen auf Amazon EventBridge Steuerebene, die in EventBridge protokolliert CloudTrail, finden Sie in der APIAmazon EventBridge -Referenz zu .

EventBridge -Ereignisbeispiele

Ein Ereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar und enthält Informationen über die angeforderte API-Operation, das Datum und die Uhrzeit der Operation, die Anforderungsparameter usw. CloudTrail Protokolldateien sind kein geordnetes Stacktrace der öffentlichen API-Aufrufe, sodass Ereignisse nicht in einer bestimmten Reihenfolge angezeigt werden.

Das folgende Beispiel zeigt ein CloudTrail Ereignis, das die `PutRule` Operation demonstriert.

```
{
  "eventVersion":"1.03",
  "userIdentity":{
    "type":"Root",
    "principalId":"123456789012",
    "arn":"arn:aws:iam::123456789012:root",
```

```
    "accountId":"123456789012",
    "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
    "sessionContext":{
      "attributes":{
      "mfaAuthenticated":"false",
      "creationDate":"2015-11-17T23:56:15Z"
      }
    }
  },
  "eventTime":"2015-11-18T00:11:28Z",
  "eventSource":"events.amazonaws.com",
  "eventName":"PutRule",
  "awsRegion":"us-east-1",
  "sourceIPAddress":"AWS Internal",
  "userAgent":"AWS CloudWatch Console",
  "requestParameters":{
    "description":"",
    "name":"cttest2",
    "state":"ENABLED",
    "eventPattern":"{\"source\":[\"aws.ec2\"],\"detail-type\":[\"EC2 Instance State-
change Notification\"]}",
    "scheduleExpression":""
  },
  "responseElements":{
    "ruleArn":"arn:aws:events:us-east-1:123456789012:rule/cttest2"
  },
  "requestID":"e9caf887-8d88-11e5-a331-3332aa445952",
  "eventID":"49d14f36-6450-44a5-a501-b0fdcdfaeb98",
  "eventType":"AwsApiCall",
  "apiVersion":"2015-10-07",
  "recipientAccountId":"123456789012"
}
```


Weitere Informationen zu CloudTrail Datensatzinhalten finden Sie unter CloudTrail Datensatzinhalte im AWS CloudTrail -Benutzerhandbuch.

CloudTrail -Protokolleinträge für von EventBridge Pipes durchgeführte Aktionen

EventBridge Pipes übernimmt die bereitgestellte IAM-Rolle beim Lesen von Ereignissen aus Quellen, beim Aufrufen von Anreicherungen oder beim Aufrufen von Zielen. Für CloudTrail Einträge im Zusammenhang mit Aktionen in Ihrem Konto für alle Anreicherungen, Ziele und Amazon SQS,

Kinesis- und DynamoDB-Quellen enthalten die `invokedBy` Felder `sourceIPAddress` und `pipes.amazonaws.com`.

CloudTrail Beispielprotokolleintrag für alle Anreicherungen, Ziele und Amazon SQS, Kinesis- und DynamoDB-Quellen

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "...",
    "arn": "arn:aws:sts::111222333444:assumed-role/...",
    "accountId": "111222333444",
    "accessKeyId": "...",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "...",
        "arn": "...",
        "accountId": "111222333444",
        "userName": "userName"
      },
      "webIdFederationData": {},
      "attributes": {
        "creationDate": "2022-09-22T21:41:15Z",
        "mfaAuthenticated": "false"
      }
    },
    "invokedBy": "pipes.amazonaws.com"
  },
  "eventTime": ",,,",
  "eventName": "...",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "pipes.amazonaws.com",
  "userAgent": "pipes.amazonaws.com",
  "requestParameters": {
    ...
  },
  "responseElements": null,
  "requestID": "...",
  "eventID": "...",
  "readOnly": true,
  "eventType": "AwsApiCall",
```

```
  "managementEvent": true,
  "recipientAccountId": "...",
  "eventCategory": "Management"
}
```

Für alle anderen Quellen hat das -`sourceIPAddress`Feld der CloudTrail Protokolleinträge eine dynamische IP-Adresse und sollte nicht für Integrationen oder Ereigniskategorisierung verwendet werden. Außerdem enthalten diese Einträge das `invokedBy`-Feld nicht.

CloudTrail Beispielprotokolleintrag für alle anderen Quellen

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    ...
  },
  "eventTime": ",,,",
  "eventName": "...",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "Python-httplib2/0.8 (gzip)",
}
```

Compliance-Validierung in Amazon EventBridge

Die Auditoren Dritter, z. B. SOC, PCI, FedRAMP und HIPAA, bewerten die Sicherheit und die Compliance von AWS-Services im Rahmen mehrerer AWS-Compliance-Programme.

Eine Liste der AWS-Services im Bereich bestimmter Compliance-Programme finden Sie unter AWS-Services im Bereich nach Compliance-Programm. Allgemeine Informationen finden Sie unter AWS-Compliance-Programme.

Sie können Auditberichte von Drittanbietern unter AWS Artifact herunterladen. Weitere Informationen finden Sie unter Herunterladen von Berichten in AWS Artifact .

Ihre Compliance-Verantwortung bei der Verwendung von EventBridge ist von der Sensibilität Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften abhängig. AWS stellt die folgenden Ressourcen zur Unterstützung der Compliance bereit:

- Schnellstartanleitungen für Sicherheit und Compliance – Architektonische Überlegungen und Schritte für die Bereitstellung von sicherheits- und konformitätsorientierten Basisumgebungen auf AWS.
- Whitepaper zur Erstellung einer Architektur mit HIPAA-konformer Sicherheit und Compliance – Wie Unternehmen mithilfe von AWS HIPAA-konforme Anwendungen erstellen können.
- AWS-Compliance-Ressourcen – Eine Sammlung von Arbeitsbüchern und Leitfäden.
- Bewerten von Ressourcen mit Regeln im AWS Config-Entwicklerhandbuch – Informationen dazu, wie AWS Config bewertet, zu welchem Grad die Konfiguration Ihrer Ressourcen den internen Vorgehensweisen, Branchenrichtlinien und Vorschriften entspricht.
- AWS Security Hub: Eine umfassende Übersicht über Ihren Sicherheitsstatus innerhalb von AWS, die Ihnen hilft, Ihre Einhaltung von Standards und bewährten Methoden der Branche in Bezug auf Sicherheit zu überprüfen.

Amazon-EventBridge-Ausfallsicherheit

Im Zentrum der globalen AWS Infrastruktur stehen die AWS-Regionen und Availability Zones (Verfügbarkeitszonen, AZs). AWS -Regionen stellen mehrere physisch getrennte und isolierte Availability Zones bereit, die über hoch redundante Netzwerke mit niedriger Latenz und hohen Durchsätzen verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Zonen ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Weitere Informationen über AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Infrastruktursicherheit in Amazon EventBridge

Als verwalteter Service ist Amazon EventBridge durch die globale Netzwerksicherheit von AWS geschützt. Informationen zu AWS-Sicherheitsdiensten und wie AWS die Infrastruktur schützt, finden Sie unter AWSCloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS-Umgebung anhand der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastrukturschutz im Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS-veröffentlichte API-Aufrufe, um über das Netzwerk auf EventBridge zuzugreifen. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systemen wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit AWS Security Token Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Sie können diese API-Operationen von jedem Netzwerkstandort aus aufrufen und ressourcenbasierte Zugriffsrichtlinien in EventBridge verwenden, die Einschränkungen auf der Basis der Quell-IP-Adresse enthalten können. Sie können auch EventBridge-Richtlinien verwenden, um den Zugriff über bestimmte Amazon Virtual Private Cloud (Amazon VPC)-Endpunkte oder bestimmte VPCs zu steuern. Tatsächlich wird der Netzwerkzugriff hierdurch auf eine bestimmte EventBridge-Ressource eingeschränkt, sodass er ausschließlich über eine bestimmte VPC innerhalb des AWS-Netzwerks ausgeführt werden kann.

Konfigurations- und Schwachstellenanalyse in Amazon EventBridge

Konfiguration und IT-Steuerelemente unterliegen der übergreifenden Verantwortlichkeit von AWS und Ihnen, unserem Kunden. Weitere Informationen finden Sie unter AWSModell der übergreifenden Verantwortlichkeit.

Überwachung von Amazon EventBridge

EventBridge sendet CloudWatch jede Minute Metriken an Amazon für alles, von der Anzahl der übereinstimmenden Ereignisse bis hin zur Häufigkeit, mit der ein Ziel durch eine Regel aufgerufen wird.

Das folgende Video gibt einen Überblick über das Überwachungs- und EventBridge Prüfungsverhalten anhand von CloudWatch: Überwachung und Prüfung von Ereignissen

Themen

EventBridge Metriken

Der `AWS/Events`-Namespace enthält die folgenden Metriken.

Bei den Metriken, die Anzahl als Einheit verwenden, sind Summe und in der SampleCount Regel die nützlichsten Statistiken.

Metriken, die nur die `RuleName` Dimension angeben, beziehen sich auf den Standard-Event-Bus. Metriken, die `EventBusName` sowohl die als auch die `RuleName` Dimensionen angeben, beziehen sich auf einen benutzerdefinierten Event-Bus.

Metrik	Beschreibung	Dimensionen	Einheiten
`DeadLetterInvocations`	Die Häufigkeit, mit der das Ziel einer Regel nicht als Antwort auf ein Ereignis aufgerufen wird Dazu gehören Aufrufe, die zum erneuten Ausführen derselben Regel führen und damit eine Endlosschleife verursachen würden.	RuleName	Anzahl
`Events`	Die Anzahl der Partnerereignisse, die von aufgenommen wurden EventBridge.	EventSourceName	Anzahl
`FailedInvocations`	Die Anzahl der Aufrufe, die endgültig fehlgeschlagen sind Dies umfasst keine Aufrufe, die	RuleName	Anzahl

Metrik	Beschreibung	Dimensionen	Einheiten
	wiederholt werden, oder Aufrufe, die nach einem Wiederholungsversuch erfolgreich waren. Fehlgeschlagene Aufrufe, die in `DeadLetterInvocations` eingeschlossen werden, werden nicht gezählt. **Note** EventBridge sendet diese Metrik nur an, CloudWatch wenn sie nicht Null ist.		
`Invocations`	Die Häufigkeit, mit der ein Ziel von einer Regel als Antwort auf ein Ereignis aufgerufen wird Dies schließt erfolgreiche und fehlgeschlagene Aufrufe ein, nicht jedoch gedrosselte oder wiederholte Versuche, sofern sie nicht endgültig fehlschlagen. Es beinhaltet nicht `DeadLetterInvocations` . **Note** EventBridge sendet diese Metrik nur an, CloudWatch wenn sie nicht Null ist.	Keine, RuleName	Anzahl
`InvocationAttempts`	Anzahl der Versuche EventBridge , ein Ziel aufzurufen.	None	Anzahl

Metrik	Beschreibung	Dimensionen	Einheiten
`InvocationsCreated`	Die Gesamtzahl der Aufrufe, die als Antwort auf jedes Ereignis erstellt wurden Diese Metrik wird häufig verwendet, um die Nutzung des Invocations-Drosselllimits bei der Anzahl der Transaktionen pro Sekunde EventBridge für das Servicekontingent zu überwachen.	None	Anzahl
`InvocationsFailedToBeSentToDlq`	Die Anzahl der Aufrufe, die nicht in eine Warteschlange für unzustellbare Nachrichten verschoben werden konnten Fehler bei Warteschlangen für unzustellbare Nachrichten können aufgrund von Berechtigungsfehlern, nicht verfügbaren Ressourcen oder Größenbeschränkungen auftreten. **Note** EventBridge sendet diese Metrik nur an, CloudWatch wenn sie nicht Null ist.	RuleName	Anzahl
`IngestionToInvocationCompleteLatency`	Die Zeit von der Erfassung des Ereignisses bis zum Abschluss des ersten erfolgreichen Aufrufversuchs	EventBusName, Keine, RuleName	Millisekunden
`IngestionToInvocationStartLatency`	Die Zeit bis zur Verarbeitung von Ereignissen, gemessen von der Aufnahme eines Ereignisses EventBridge bis zum ersten Aufruf eines Ziels.	EventBusName, Keine, RuleName	Millisekunden

Metrik	Beschreibung	Dimensionen	Einheiten
`InvocationsSentToDlq`	Die Anzahl der Aufrufe, die in eine Warteschlange für unzustellbare Nachrichten verschoben werden **Note** EventBridge sendet diese Metrik nur an, CloudWatch wenn sie nicht Null ist.	RuleName	Anzahl
`MatchedEvents`	Wenn EventBusName oder angegeben EventSourceName ist, die Anzahl der Ereignisse, die mit einer Regel übereinstimmten. Wenn RuleName angegeben, die Anzahl der Ereignisse, die mit einer bestimmten Regel übereinstimmten.	EventBusName, EventSourceName, RuleName	Anzahl
`RetryInvocationAttempts`	Häufigkeit, mit der die Zielaufrufe wiederholt wurden **Note** EventBridge sendet diese Metrik nur an, CloudWatch wenn sie nicht Null ist.	None	Anzahl
`SuccessfulInvocationAttempts`	Häufigkeit, mit der das Ziel erfolgreich aufgerufen wurde	None	Anzahl

Metrik	Beschreibung	Dimensionen	Einheiten
`ThrottledRules`	Häufigkeit, mit der die Regelausführung gedrosselt wurde Das Aufrufen dieser Regeln kann sich verzögern. Weitere Informationen finden Sie unter Drossel-Limit für Aufrufe in Transaktionen pro Sekunde in ???.	EventBusName, Keine, RuleName	Anzahl
`TriggeredRules`	Die Anzahl der Regeln, die ausgeführt wurden und mit einem Ereignis übereinstimmten Diese Metrik wird CloudWatch erst angezeigt, wenn eine Regel ausgelöst wird.	EventBusName, Keine, RuleName	Anzahl

EventBridge PutEvents Metriken

Der Namespace `AWS/Events` enthält die folgenden Metriken, die sich auf `PutEvents`-API-Anforderungen beziehen.

Bei den Metriken, die Anzahl als Einheit verwenden, sind Summe und in der SampleCount Regel die nützlichsten Statistiken.

Metrik	Beschreibung	Dimensionen	Einheiten
`PutEventsApproximateCallCount`	Ungefähre Anzahl empfangener `PutEvents` -Anforderungen	None	Anzahl
`PutEventsApproximateFailedCount`	Ungefähre Anzahl fehlgeschlagener `PutEvents` -Anforderungen	None	Anzahl
`PutEventsApproxima`	Ungefähre Anzahl erfolgreicher `PutEvents` -Anforderungen	None	Anzahl

Metrik	Beschreibung	Dimensionen	Einheiten
teSuccessCount			
PutEventsApproximateThrottledCount	Anzahl der PutEvents -Anforderungen, die aufgrund von Drosselung abgelehnt wurden	None	Anzahl
PutEventsEntriesCount	Die Anzahl der in einer PutEvents -Anforderung enthaltenen Ereigniseinträge	None	Anzahl
PutEventsFailedEntriesCount	Die Anzahl der in einer PutEvents -Anforderung enthaltenen Ereigniseinträge, die nicht erfasst werden konnten	None	Anzahl
PutEventsLatency	Die pro PutEvents -Anforderung benötigte Zeit	None	Millisekunden
PutEventsRequestSize	Die Größe der PutEvents -Anforderung	None	Bytes

EventBridge PutPartnerEvents Metriken

Der Namespace `AWS/Events` enthält die folgenden Metriken, die sich auf PutPartnerEvents-API-Anforderungen beziehen.

> **Note**
>
> EventBridge enthält nur Metriken, die sich auf PutPartnerEventsAnfragen in SaaS-Partnerkonten beziehen, die Ereignisse senden. Weitere Informationen finden Sie unter ???.

Bei den Metriken, die Anzahl als Einheit verwenden, sind Summe und in der SampleCount Regel die nützlichsten Statistiken.

Metrik	Beschreibung	Dimensionen	Einheiten
PutPartnerEventsApproximateCallCount	Ungefähre Anzahl empfangener PutPartnerEvents -Anforderungen	None	Anzahl
PutPartnerEventsApproximateFailedCount	Ungefähre Anzahl fehlgeschlagener PutPartnerEvents -Anforderungen	None	Anzahl
PutPartnerEventsApproximateThrottledCount	Anzahl der PutPartnerEvents -Anforderungen, die aufgrund von Drosselung abgelehnt wurden	None	Anzahl
PutPartnerEventsApproximateSuccessCount	Ungefähre Anzahl erfolgreicher PutPartnerEvents -Anforderungen	None	Anzahl
PutPartnerEventsEntriesCount	Die Anzahl der in einer PutPartnerEvents -Anforderung enthaltenen Ereigniseinträge	None	Anzahl
PutPartnerEventsFailedEntriesCount	Die Anzahl der in einer PutPartnerEvents -Anforderung enthaltenen Ereigniseinträge, die nicht erfasst werden konnten	None	Anzahl

Metrik	Beschreibung	Dimensionen	Einheiten
`PutPartnerEventsLatency`	Die pro `PutPartnerEvents` -Anforderung benötigte Zeit	None	Millisekunden

Dimensionen für EventBridge Metriken

EventBridge Metriken haben Dimensionen oder sortierbare Attribute, die unten aufgeführt sind.

Dimension	Beschreibung
`EventBusName`	Filtert die verfügbaren Metriken nach Event-Bus-Name.
`EventSourceName`	Filtert die verfügbaren Metriken nach Partner-Ereignisquellenname.
`RuleName`	Filtert die verfügbaren Metriken nach Regelname.

Problembehebung bei Amazon EventBridge

Sie können die Schritte in diesem Abschnitt verwenden, um Probleme bei Amazon zu beheben EventBridge.

Themen

Meine Regel wurde ausgeführt, aber meine Lambda-Funktion wurde nicht aufgerufen

Ein Grund dafür, dass Ihre Lambda-Funktion möglicherweise nicht ausgeführt wird, liegt darin, dass Sie nicht über die richtigen Berechtigungen verfügen.

So überprüfen Sie Ihre Berechtigungen für die Lambda-Funktion

1. Führen Sie mit dem AWS CLI den folgenden Befehl mit Ihrer Funktion und Ihrer AWS Region aus:

   ```
   aws lambda get-policy --function-name MyFunction --region us-east-1
   ```

   Die Ausgabe sollte folgendermaßen aussehen.

   ```
   {
       "Policy": "{\"Version\":\"2012-10-17\",
       \"Statement\":[
           {\"Condition\":{\"ArnLike\":{\"AWS:SourceArn\":\"arn:aws:events:us-
   east-1:123456789012:rule/MyRule\"}},
           \"Action\":\"lambda:InvokeFunction\",
           \"Resource\":\"arn:aws:lambda:us-east-1:123456789012:function:MyFunction\",
           \"Effect\":\"Allow\",
           \"Principal\":{\"Service\":\"events.amazonaws.com\"},
           \"Sid\":\"MyId\"}
       ],
       \"Id\":\"default\"}"
   }
   ```

2. Wenn folgende Meldung angezeigt wird.

   ```
   A client error (ResourceNotFoundException) occurred when calling the GetPolicy
    operation: The resource you requested does not exist.
   ```

   Oder wenn Ihnen die Ausgabe zwar angezeigt wird, Sie aber events.amazonaws.com nicht als vertrauenswürdige Entität in der Richtlinie finden können, führen Sie den folgenden Befehl aus:

   ```
   aws lambda add-permission \
   --function-name MyFunction \
   --statement-id MyId \
   ```

```
--action 'lambda:InvokeFunction' \
--principal events.amazonaws.com \
--source-arn arn:aws:events:us-east-1:123456789012:rule/MyRule
```

3. Wenn die Ausgabe ein `SourceAccount`-Feld enthält, müssen Sie es entfernen. Eine `SourceAccount` Einstellung EventBridge verhindert, dass die Funktion aufgerufen werden kann.

> **Note**
>
> Wenn die Richtlinie falsch ist, können Sie die Regel in der EventBridge Konsole bearbeiten, indem Sie sie entfernen und dann wieder zur Regel hinzufügen. Die EventBridge Konsole legt dann die richtigen Berechtigungen für das Ziel fest.
> Wenn Sie einen bestimmten Lambda-Alias oder eine spezielle Version verwenden, fügen Sie die `--qualifier`-Parameter in den Befehlen `aws lambda get-policy` und `aws lambda add-permission` hinzu, wie im folgenden Befehl gezeigt.
>
> ```
> aws lambda add-permission \
> --function-name MyFunction \
> --statement-id MyId \
> --action 'lambda:InvokeFunction' \
> --principal events.amazonaws.com \
> --source-arn arn:aws:events:us-east-1:123456789012:rule/MyRule
> --qualifier alias or version
> ```

Ich habe gerade eine Regel erstellt oder bearbeitet, sie stimmt aber nicht mit einem Testereignis überein.

Wenn Sie an einer Regel oder deren Zielen Änderungen vornehmen, stimmen die Einstellungen der eingehenden Ereignisse möglicherweise nicht sofort mit den aktualisierten Regeln überein. Warten Sie einen Augenblick, bis die Änderungen wirksam werden.

Wenn die Ereignisse nach kurzer Zeit immer noch nicht übereinstimmen, überprüfen Sie die CloudWatch Metriken `TriggeredRules` und `FailedInvocations` Ihre Regel. `Invocations` Weitere Informationen zu diesen Kennzahlen finden Sie unter Amazon überwachen EventBridge.

Wenn die Regel einem Ereignis eines AWS Dienstes entsprechen soll, gehen Sie wie folgt vor:

- Verwenden Sie die Aktion `TestEventPattern`, um zu testen, ob das Ereignismuster Ihrer Regel mit einem Testereignis übereinstimmt. Weitere Informationen finden Sie TestEventPatternin der Amazon EventBridge API-Referenz.
- Verwenden Sie die Sandbox auf der EventBridge Konsole.

Meine Regel wurde nicht zu dem Zeitpunkt ausgeführt, den ich im **`ScheduleExpression`** angegeben habe.

Stellen Sie sicher, dass Sie den Zeitplan für die Regel auf die Zeitzone UTC+0 eingestellt haben. Wenn der `ScheduleExpression` korrekt ist, befolgen Sie die Schritte unter Ich habe gerade eine Regel erstellt oder bearbeitet, sie stimmt aber nicht mit einem Testereignis überein..

Meine Regel wurde nicht zum erwarteten Zeitpunkt ausgeführt.

EventBridge führt Regeln innerhalb einer Minute nach der von Ihnen festgelegten Startzeit aus. Der Countdown beginnt zu laufen, sobald Sie die Regel erstellen.

> **Note**
>
> Geplante Regeln weisen die Zustellungsart `guaranteed` auf, was bedeutet, dass Ereignisse für jeden erwarteten Zeitpunkt mindestens einmal ausgelöst werden.

Sie können einen Cron-Ausdruck verwenden, um Ziele zu einem bestimmten Zeitpunkt aufzurufen. Wenn Sie eine Regel erstellen möchten, die alle vier Stunden in der 0. Minute ausgeführt wird, gehen Sie wie folgt vor:

- In der EventBridge Konsole verwenden Sie den Cron-Ausdruck`0 0/4 * * ? *`.
- Mit dem AWS CLI verwenden Sie den Ausdruck`cron(0 0/4 * * ? *)`.

Um beispielsweise mithilfe von eine Regel mit dem Namen zu erstellen`TestRule`, die alle 4 Stunden ausgeführt wird AWS CLI, verwenden Sie den folgenden Befehl.

```
aws events put-rule --name TestRule --schedule-expression 'cron(0 0/4 * * ? *)'
```

Wenn Sie eine Regel alle fünf Minuten ausführen möchten, verwenden Sie den folgenden Cron-Ausdruck.

```
aws events put-rule --name TestRule --schedule-expression 'cron(0/5 * * * ? *)'
```

Die beste Auflösung für eine EventBridge Regel, die einen Cron-Ausdruck verwendet, ist eine Minute. Die geplante Regel wird innerhalb der angegebenen Minute ausgeführt, aber nicht exakt in der 0. Sekunde.

Da EventBridge die Zieldienste verteilt sind, kann es zwischen der Ausführung der geplanten Regel und dem Zeitpunkt, zu dem der Zieldienst die Aktion auf der Zielressource ausführt, zu einer Verzögerung von mehreren Sekunden kommen.

Meine Regel entspricht AWS globalen Service-API-Aufrufen, wurde aber nicht ausgeführt

AWS globale Dienste wie IAM und Amazon Route 53 sind nur in der Region USA Ost (Nord-Virginia) verfügbar, sodass Ereignisse aus AWS API-Aufrufen von globalen Diensten nur in dieser Region verfügbar sind. Weitere Informationen finden Sie unter Ereignisse im Zusammenhang mit Dienstleistungen AWS.

Die mit meiner Regel verknüpfte IAM-Rolle wird ignoriert, wenn die Regel ausgeführt wird.

EventBridge verwendet nur IAM-Rollen für Regeln, die Ereignisse an Kinesis-Streams senden. Für Regeln, die Lambda-Funktionen und Amazon SNS-Themen aufrufen, müssen Sie ressourcenbasierte Berechtigungen erteilen.

Stellen Sie sicher, dass Ihre regionalen AWS STS Endgeräte aktiviert sind, damit sie verwendet werden EventBridge können, wenn Sie die von Ihnen angegebene IAM-Rolle übernehmen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Aktivierung und Deaktivierung AWS STS in einer AWS Region.

Meine Regel verfügt über ein Ereignismuster, das einer Ressource entsprechen soll, aber es stimmen keine Ereignisse überein.

Die meisten Dienste in AWS behandeln einen Doppelpunkt (:) oder einen Schrägstrich (/) als dasselbe Zeichen in Amazon Resource Names (ARNs)., EventBridge verwenden jedoch eine exakte Übereinstimmung in den Ereignismustern und Regeln. Verwenden Sie also die richtigen ARN-Zeichen zum Erstellen von Ereignismustern, sodass sie mit der ARN-Syntax in dem Ereignis übereinstimmen.

Einige Ereignisse, wie z. B. AWS API-Aufrufereignisse von CloudTrail, haben im Feld Ressourcen nichts.

Die Bereitstellung meines Ereignisses an das Ziel verzögerte sich.

EventBridge versucht, ein Ereignis bis zu 24 Stunden lang an ein Ziel zu übermitteln, außer in Szenarien, in denen Ihre Zielressource eingeschränkt ist. Der erste Versuch erfolgt, sobald das Ereignis im Ereignis-Stream eintrifft. Wenn der Zieldienst Probleme hat, wird EventBridge automatisch ein neuer Termin für eine weitere Zustellung festgelegt. Wenn seit dem Eintreffen des Ereignisses 24 Stunden vergangen sind, EventBridge unterbricht der Versuch, das Ereignis zuzustellen, und veröffentlicht die `FailedInvocations` Metrik in CloudWatch. Wir empfehlen, dass Sie eine Warteschlange für unzustellbare Nachrichten einrichten, um Ereignisse zu speichern, die nicht erfolgreich an ein Ziel übermittelt werden konnten. Weitere Informationen finden Sie unter Verwenden von Warteschlangen mit unzustellbaren Buchstaben zur Verarbeitung nicht zugestellter Ereignisse.

Einige Ereignisse wurden nie in mein Ziel ausgeliefert

Wenn das Ziel einer EventBridge Regel über einen längeren Zeitraum eingeschränkt ist, wird die Übertragung EventBridge möglicherweise nicht erneut versucht. Wenn das Ziel beispielsweise nicht für die Verarbeitung des eingehenden Ereignisverkehrs vorgesehen ist und der Zieldienst Anfragen, die in Ihrem Namen gestellt werden, drosselt, wird die EventBridge Zustellung möglicherweise nicht erneut EventBridge versucht.

Meine Regel wurde als Antwort auf ein Ereignis mehr als einmal ausgeführt.

In seltenen Fällen kann die gleiche Regel mehr als einmal für ein einzelnes Ereignis oder eine geplante Zeit ausgeführt werden, oder dasselbe Ziel kann für eine bestimmte ausgelöste Regel mehr als einmal aufgerufen werden.

Verhindern von Endlosschleifen

In ist es möglich EventBridge, eine Regel zu erstellen, die zu Endlosschleifen führt, in denen die Regel wiederholt ausgeführt wird. Wenn Sie über eine Regel verfügen, die eine Endlosschleife verursacht, schreiben Sie sie so um, dass die Aktionen, die die Regel ausführt, nicht derselben Regel entsprechen.

Beispielsweise verursacht eine Regel, die erkennt, dass sich ACLs in einem Amazon-S3-Bucket geändert haben, und dann Software ausführt, um sie in einen neuen Zustand zu versetzen, eine Endlosschleife. Eine Möglichkeit, dieses Problem zu lösen, besteht darin, die Regel so umzuschreiben, dass sie nur mit ACLs übereinstimmt, die sich in einem fehlerhaften Zustand befinden.

Eine Endlosschleife kann schnell höhere Gebühren als erwartet verursachen. Wir empfehlen, dass Sie Budgetierung verwenden, um Warnungen zu erhalten, wenn die Gebühren das von Ihnen angegebene Limit überschreiten. Weitere Informationen finden Sie unter Verwalten der Kosten mit Budgets.

Meine Ereignisse werden nicht in die Amazon SQS-Zielwarteschlange ausgeliefert

Wenn Ihre Amazon-SQS-Warteschlange verschlüsselt ist, müssen Sie einen vom Kunden verwalteten KMS-Schlüssel erstellen und den folgenden Berechtigungsabschnitt in Ihre KMS-Schlüsselrichtlinie aufnehmen. Weitere Informationen finden Sie unter AWS KMS Berechtigungen konfigurieren.

```
{
  "Sid": "Allow EventBridge to use the key",
  "Effect": "Allow",
  "Principal": {
```

```
    "Service": "events.amazonaws.com"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
  ],
  "Resource": "*"
}
```

Meine Regel wird ausgeführt, aber mir werden keine im Amazon SNS-Thema veröffentlichten Nachrichten angezeigt.

Szenario 1

Sie benötigen die Berechtigung, Nachrichten im Amazon-SNS-Thema zu veröffentlichen. Verwenden Sie den folgenden Befehl AWS CLI, indem Sie us-east-1 durch Ihre Region ersetzen und Ihren Themen-ARN verwenden.

```
aws sns get-topic-attributes --region us-east-1 --topic-arn "arn:aws:sns:us-
east-1:123456789012:MyTopic"
```

Wenn Sie über die richtige Berechtigung verfügen möchten, müssen Ihre Richtlinienattribute den folgenden ähneln.

```
"{\"Version\":\"2012-10-17\",
\"Id\":\"__default_policy_ID\",
\"Statement\":[{\"Sid\":\"__default_statement_ID\",
\"Effect\":\"Allow\",
\"Principal\":{\"AWS\":\"*\"},
\"Action\":[\"SNS:Subscribe\",
\"SNS:ListSubscriptionsByTopic\",
\"SNS:DeleteTopic\",
\"SNS:GetTopicAttributes\",
\"SNS:Publish\",
\"SNS:RemovePermission\",
\"SNS:AddPermission\",
\"SNS:SetTopicAttributes\"],
\"Resource\":\"arn:aws:sns:us-east-1:123456789012:MyTopic\",
\"Condition\":{\"StringEquals\":{\"AWS:SourceOwner\":\"123456789012\"}}},{\"Sid\":
\"Allow_Publish_Events\",
```

```
\"Effect\":\"Allow\",
\"Principal\":{\"Service\":\"events.amazonaws.com\"},
\"Action\":\"sns:Publish\",
\"Resource\":\"arn:aws:sns:us-east-1:123456789012:MyTopic\"}]}"
```

Wenn in Ihrer Richtlinie `events.amazonaws.com` mit `Publish`-Berechtigung nicht angezeigt wird, kopieren Sie zunächst die aktuelle Richtlinie und fügen Sie die folgende Anweisung zur Liste der Anweisungen hinzu.

```
{\"Sid\":\"Allow_Publish_Events\",
\"Effect\":\"Allow\",\"Principal\":{\"Service\":\"events.amazonaws.com\"},
\"Action\":\"sns:Publish\",
\"Resource\":\"arn:aws:sns:us-east-1:123456789012:MyTopic\"}
```

Legen Sie dann die Themenattribute fest AWS CLI, indem Sie den folgenden Befehl verwenden.

```
aws sns set-topic-attributes --region us-east-1 --topic-arn "arn:aws:sns:us-
east-1:123456789012:MyTopic" --attribute-name Policy --attribute-
value NEW_POLICY_STRING
```

> **Note**
>
> Wenn die Richtlinie falsch ist, können Sie die Regel auch in der EventBridge Konsole bearbeiten, indem Sie sie entfernen und dann wieder zur Regel hinzufügen. EventBridge legt die richtigen Berechtigungen für das Ziel fest.

Szenario 2

Wenn Ihr SNS-Thema verschlüsselt ist, müssen Sie den folgenden Abschnitt in Ihre KMS-Schlüsselrichtlinie aufnehmen.

```
{
  "Sid": "Allow EventBridge to use the key",
  "Effect": "Allow",
  "Principal": {
    "Service": "events.amazonaws.com"
  },
  "Action": [
    "kms:Decrypt",
```

```
    "kms:GenerateDataKey"
  ],
  "Resource": "*"
}
```

Mein Amazon SNS SNS-Thema hat EventBridge auch nach dem Löschen der Regel, die mit dem Amazon SNS SNS-Thema verknüpft ist, weiterhin Berechtigungen

Wenn Sie eine Regel mit Amazon SNS als Ziel erstellen, EventBridge fügt Ihrem Amazon SNS SNS-Thema in Ihrem Namen Berechtigungen hinzu. Wenn Sie die Regel kurz nach ihrer Erstellung löschen, wird die Genehmigung EventBridge möglicherweise nicht aus Ihrem Amazon SNS SNS-Thema entfernt. Wenn dies geschieht, können Sie die Berechtigung aus dem Thema mithilfe des Befehls `aws sns set-topic-attributes` entfernen. Weitere Informationen zu ressourcenbasierten Berechtigungen für das Senden von Ereignissen finden Sie unter Verwenden ressourcenbasierter Richtlinien für Amazon EventBridge.

Mit welchen IAM-Bedingungsschlüsseln kann ich sie verwenden? EventBridge

EventBridge unterstützt die AWS-weiten Bedingungsschlüssel (siehe IAM- und AWS STS Bedingungskontextschlüssel im IAM-Benutzerhandbuch) sowie die unter aufgeführten Schlüssel. Verwenden von IAM-Richtlinienbedingungen für die differenzierte Zugriffskontrolle

Woran erkenne ich, dass EventBridge Regeln verletzt wurden?

Sie können den folgenden Alarm verwenden, um Sie zu benachrichtigen, wenn Ihre EventBridge Regeln verletzt werden.

Um einen Alarm zu erstellen, der warnt, wenn Regeln nicht eingehalten werden,

1. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie Alarm erstellen aus. Wählen Sie im Bereich CloudWatch Metriken nach Kategorie die Option Event-Metriken aus.
3. Wählen Sie in der Liste der Metriken die Option aus FailedInvocations.

4. Wählen Sie über dem Diagramm Statistic und Sum aus.
5. Wählen Sie einen Wert für Period aus, z. B. 5 Minuten. Wählen Sie Weiter aus.
6. Geben Sie unter Alarmschwellenwert für Name beispielsweise einen eindeutigen Namen für den Alarm ein myFailedRules. Geben Sie für Beschreibung eine Beschreibung des Alarms ein. Beispiel: Regeln übermitteln keine Ereignisse an das Ziel.
7. Wählen Sie für is die Option >= und 1 aus. Geben Sie für for die Option 10 ein.
8. Wählen Sie unter Actions (Aktionen) für Whenever this alarm die Option State is ALARM (Status ist ALARM) aus.
9. Wählen Sie für Send notification to (Benachrichtigung senden an) ein vorhandenes Amazon SNS-Thema aus oder erstellen Sie ein neues. Um ein neues Thema zu erstellen, wählen Sie New list aus. Geben Sie einen Namen für das neue Amazon SNS SNS-Thema ein, zum Beispiel: myFailedRules.
10. Geben Sie für Email list (E-Mail-Liste) eine durch Kommata getrennte Liste der E-Mail-Adressen ein, die benachrichtigt werden sollen, wenn der Alarm in den Status ALARM versetzt wird.
11. Wählen Sie Alarm erstellen aus.

Amazon-EventBridge-Kontingente

Für die meisten Aspekte von EventBridge gibt es Kontingente.

Themen

> **Note**
>
> Eine Liste der Kontingente für EventBridge Scheduler finden Sie unter Kontingente für EventBridge Scheduler im Benutzerhandbuch von EventBridge Scheduler.

EventBridge-Kontingente

Für EventBridge gelten die folgenden Kontingente.

Die Service-Quotas-Konsole stellt Informationen zu EventBridge-Kontingenten bereit. Neben der Anzeige der Standardkontingente können Sie die Servicekontingentkonsole verwenden, um Kontingenterhöhungen für einstellbare Kontingente anzufordern.

Name	Standard	Anpas	Beschreibung
API-Ziele	Jede unterstützte Region: 3 000	Ja	Die maximale Anzahl der API-Ziele pro Konto und Region.
Verbindungen	Jede unterstützte Region: 3 000	Ja	Die maximale Anzahl der Verbindungen pro Konto und Region.

Name	Standard	Anpas	Beschreibung
Drossel-Limit für CreateEndpoint in Transaktionen pro Sekunde	Jede unterstützte Region: 5 pro Sekunde	Nein	Die maximale Anzahl von Anforderungen pro Sekunde für CreateEndpoint API. Darüber hinausgehende Anfragen werden gedrosselt.
Drossel-Limit für DeleteEndpoint in Transaktionen pro Sekunde	Jede unterstützte Region: 5 pro Sekunde	Nein	Die maximale Anzahl von Anforderungen pro Sekunde für DeleteEndpoint API. Darüber hinausgehende Anfragen werden gedrosselt.
Endpunkte	Jede unterstützte Region: 100	Yes (Ja)	Die maximale Anzahl der Endpunkte pro Konto und Region.
Größe einer Event-Bus-Richtlinie	Jede unterstützte Region: 10 240	Ja	Maximalgröße einer Richtlinie in Zeichen Diese Richtliniengröße wird bei jedem Zugriff auf ein anderes Konto erhöht. Sie können Ihre aktuelle Richtlinie und ihre Größe mithilfe der DescribeEventBus-API anzeigen.
Ereignisbusse	Jede unterstützte Region: 100	Yes (Ja)	Maximale Event Buses pro Konto
Größe eines Ereignismusters	Jede unterstützte Region: 2 048	Ja	Maximalgröße eines Ereignismusters in Zeichen

Name	Standard	Anpas	Beschreibung
Drossel-Limit für Aufrufe in Transaktionen pro Sekunde	us-east-1: 18 750 pro Sekunde us-east-2: 4 500 pro Sekunde us-west-1: 2 250 pro Sekunde us-west-2: 18 750 pro Sekunde af-south-1: 750 pro Sekunde ap-northeast-1: 2 250 pro Sekunde ap-northeast-3: 750 pro Sekunde ap-southeast-1: 2 250 pro Sekunde ap-southeast-2: 2 250 pro Sekunde ap-southeast-3: 750 pro Sekunde eu-central-1: 4 500 pro Sekunde	Ja	Ein Aufruf ist ein Ereignis, das einer Regel zugeordnet ist und an die Ziele dieser Regel gesendet wird. Nachdem die Grenze erreicht ist, werden die Aufrufe gedrosselt, d. h. sie erfolgen weiterhin, aber verzögert.

Name	Standard	Anpas	Beschreibung
	eu-south-1: 750 pro Sekunde eu-west-1: 18 750 pro Sekunde eu-west-2: 2 250 pro Sekunde Jede der anderen unterstützten Regionen: 1 100 pro Sekunde		
Anzahl der Regeln	af-south-1: 100 eu-south-1: 100 Jede der anderen unterstützten Regionen: 300	Ja	Maximale Anzahl von Regeln, die ein Konto pro Event Bus haben kann

Name	Standard	Anpas	Beschreibung
Drossel-Limit für PutEvents in Transaktionen pro Sekunde	us-east-1: 10 000 pro Sekunde us-east-2: 2 400 pro Sekunde us-west-1: 1 200 pro Sekunde us-west-2: 10 000 pro Sekunde af-south-1: 400 pro Sekunde ap-northeast-1: 1 200 pro Sekunde ap-northeast-3: 400 pro Sekunde ap-southeast-1: 1 200 pro Sekunde ap-southeast-2: 1 200 pro Sekunde ap-southeast-3: 400 pro Sekunde eu-central-1: 2 400 pro Sekunde	Ja	Maximale Anzahl von Anforderungen pro Sekunde für PutEvents API. Darüber hinausgehende Anfragen werden gedrosselt.

Name	Standard	Anpas	Beschreibung
	eu-south-1: 400 pro Sekunde eu-west-1: 10 000 pro Sekunde eu-west-2: 1 200 pro Sekunde Jede der anderen unterstützten Regionen: 600 pro Sekunde		
Rate der Aufrufe pro API-Ziel	Jede unterstützte Region: 300 pro Sekunde	Ja	Die maximale Anzahl von Aufrufen pro Sekunde, die an jeden API-Ziele ndpunkt pro Konto und Region gesendet werden. Sobald das Kontingen t erreicht ist, werden zukünftige Aufrufe dieses API-Endpunkts gedrossel t. Die Aufrufe finden weiterhin statt, werden aber verzögert ausgeführ t.
Ziele pro Regel	Jede unterstützte Region: 5	Nein	Maximale Anzahl von Zielen, die einer Regel zugeordnet werden können

Name	Standard	Anpas	Beschreibung
Drossel-Limit in Transaktionen pro Sekunde	Jede unterstützte Region: 50 pro Sekunde	Ja	Maximale Anzahl von Anforderungen pro Sekunde für alle Operationen der EventBridge API außer PutEvents. Darüber hinausgehende Anfragen werden gedrosselt
Drossel-Limit für UpdateEndpoint in Transaktionen pro Sekunde	Jede unterstützte Region: 5 pro Sekunde	Nein	Die maximale Anzahl von Anforderungen pro Sekunde für UpdateEndpoint API. Darüber hinausgehende Anfragen werden gedrosselt.

Darüber hinaus gelten für EventBridge die folgenden Kontingente, die nicht über die Service-Quotas-Konsole verwaltet werden.

Name	Standard	Beschreibung
Ereignisbusse	Jede unterstützte Region: 100	Maximale Event Buses pro Konto
Größe einer Event-Bus-Richtlinie	Jede unterstützte Region: 10240	Maximalgröße einer Richtlinie in Zeichen Diese Richtliniengröße wird bei jedem Zugriff auf ein anderes Konto erhöht. Sie können Ihre aktuelle Richtlinie und ihre Größe mithilfe der API `DescribeEventBus` anzeigen.
Größe eines Ereignismusters	Jede unterstützte Region: 2048	Maximalgröße eines Ereignismusters in Zeichen

Name	Standard	Beschreibung
		Dies ist auf bis zu 4096 Zeichen einstellbar. Wenn Sie höhere maximale Limits benötigen, wenden Sie sich an den Support.
Regeln mit Platzhaltern	Jede unterstützte Region: 30 Reg pro Event Bus	Die maximale Anzahl von Regeln pro Event Bus und Konto, die Ereignisfilter enthalten können, die Platzhalter umfassen. Dieses Kontingent kann nicht angepasst werden. Weitere Informationen zur Verwendung von Platzhaltern in Ereignismustern finden Sie unter ???.
Ebenen der Schemaerk ennung	Jede unterstützte Region: 255 Eb	Die maximale Anzahl von Ebenen, bei der die Schemaerkennung Ereignisse ableitet, die verschachtelt sind. Alle Ereignisse, die über 255 Ebenen hinausgehen, werden ignoriert.

PutPartnerEvents-Kontingente nach Region

Wenn Sie höhere maximale Limits benötigen, wenden Sie sich an den Support.

Regionen	Transaktionen pro Sekunde
• AWS GovCloud (USA-West) • AWS GovCloud (USA-Ost) • USA Ost (Nord-Virginia) • USA Ost (Ohio) • USA West (Nordkalifornien) • USA West (Oregon) • Africa (Cape Town) • Asia Pacific (Hong Kong) • Asia Pacific (Mumbai)	PutPartnerEvents hat standardmäßig in allen Regionen ein weiches Limit von 1 400 Durchsatzanfragen pro Sekunde und 3 600 Burst-Anfragen pro Sekunde.

Regionen	Transaktionen pro Sekunde
• Asia Pacific (Osaka) • Asia Pacific (Seoul) • Asien-Pazifik (Singapur) • Asien-Pazifik (Sydney) • Asien-Pazifik (Tokio) • Canada (Central) • Europe (Frankfurt) • Europa (Irland) • Europa (London) • Europa (Milan) • Europe (Paris) • Europe (Stockholm) • Europa (Milan) • Südamerika (São Paulo) • China (Ningxia) • China (Peking)	

Kontigente für die EventBridge-Schemaregistrierung

Für die EventBridge-Schemaregistrierung gelten die folgenden Kontingente.

Die Service-Quotas-Konsole stellt Informationen zu EventBridge-Kontingenten bereit. Neben der Anzeige der Standardkontingente können Sie die Servicekontingentkonsole verwenden, um Kontingenterhöhungen für einstellbare Kontingente anzufordern.

Name	Standard	Anpas	Beschreibung
DiscoveredSchemas	Jede unterstützte Region: 200	Ja	Maximale Anzahl von Schemas für eine erkannte Schema Registry, die Sie in der

Name	Standard	Anpas	Beschreibung
			aktuellen Region erstellen können
Discoverers	Jede unterstützte Region: 10	Yes (Ja)	Maximale Anzahl von Discoverers, die Sie in der aktuellen Region erstellen können.
Registrierungen	Jede unterstützte Region: 10	Yes (Ja)	Die maximale Anzahl von Registrierungen, die Sie in der aktuellen Region erstellen können.
SchemaVersions	Jede unterstützte Region: 100	Yes (Ja)	Die maximale Anzahl von Versionen pro Schema, die Sie in der aktuellen Region erstellen können.
Schemata	Jede unterstützte Region: 100	Yes (Ja)	Die maximale Anzahl von Schemas pro Registrierung, die Sie in der aktuellen Region erstellen können. (Außer erkannte Schema Registry)

EventBridge-Pipes-Kontingente

Für EventBridge Pipes gelten die folgenden Kontingente. Wenn Sie höhere maximale Limits benötigen, wenden Sie sich an den Support.

Ressource	Regionen	Standardlimit
Gleichzeitige Pipe-Ausführungen pro Konto	• AWS GovCloud (USA-West) • AWS GovCloud (USA-Ost) • China (Ningxia)	1000

Ressource	Regionen	Standardlimit
	• China (Peking) • Asien-Pazifik (Osaka) • Afrika (Kapstadt) • Europa (Milan) • USA Ost (Ohio) • Europa (Frankfurt) • USA West (Nordkalifornien) • Europa (London) • Asien-Pazifik (Sydney) • Asien-Pazifik (Tokio) • Asien-Pazifik (Singapur) • Kanada (Zentral) • Europa (Paris) • Europa (Stockholm) • Südamerika (São Paulo) • Asien-Pazifik (Seoul) • Asien-Pazifik (Mumbai) • Asien-Pazifik (Hongkong) • Naher Osten (Bahrain) • China (Ningxia) • China (Peking) • Asien-Pazifik (Osaka) • Afrika (Kapstadt) • Europa (Milan)	
Gleichzeitige Pipe-Ausführungen pro Konto	• USA Ost (Nord-Virginia) • USA West (Oregon) • Europa (Irland)	3000
Pipes pro Konto	Alle	1000

Amazon- EventBridge Tags

Ein Tag ist eine benutzerdefinierte Attributbezeichnung, die Sie oder einer - AWS Ressource AWS zuweisen. In können EventBridgeSie Regel- und Event Buses Tags zuweisen. Jede Ressource kann maximal 50 Tags haben.

Sie verwenden Tags, um Ihre - AWS Ressourcen zu identifizieren und zu organisieren. Viele - AWS Services unterstützen das Markieren, sodass Sie Ressourcen aus verschiedenen Services dasselbe Tag zuweisen können, um anzuzeigen, dass die Ressourcen verwandt sind. Sie können beispielsweise dasselbe Tag einer EventBridge Regel zuweisen, die Sie einer EC2-Instance zuweisen.

Ein Tag besteht aus zwei Teilen:

- einem Tag-Schlüssel, z. B. `CostCenter`, `Environment` oder `Project`.
  - Bei Tag-Schlüsseln wird zwischen Groß- und Kleinschreibung unterschieden.
  - Die maximale Länge des Tag-Schlüssels beträgt 128 Unicode-Zeichen in UTF-8.
  - Für jede Ressource muss jeder Tag-Schlüssel eindeutig sein.
  - Erlaubte Zeichen sind Buchstaben, Ziffern und Leerzeichen, die in UTF-8 darstellbar sind, sowie die folgenden Zeichen: . : + = @ _ / - (Bindestrich).
  - Das `aws:` Präfix ist für Tags verboten, da es für die AWS Verwendung reserviert ist. Sie können keine Tag-Schlüssel oder -Werte mit diesem Präfix bearbeiten oder löschen. Tags mit diesem Präfix werden nicht zum Limit für Tags pro Ressource gezählt.
- einem optionalen Feld für den Tag-Wert, z. B. `111122223333` oder `Production`.
  - Jeder Tag-Schlüssel kann nur einen Wert haben.
  - Bei Tag-Werten muss die Groß- und Kleinschreibung beachtet werden.
  - Ein nicht angegebener Tag-Wert entspricht einer leeren Zeichenfolge.
  - Die maximale Länge des Tag-Wertes beträgt 256 Unicode-Zeichen in UTF-8.
  - Erlaubte Zeichen sind Buchstaben, Ziffern und Leerzeichen, die in UTF-8 darstellbar sind, sowie die folgenden Zeichen: . : + = @ _ / - (Bindestrich).

> **Tip**
>
> Eine bewährte Methode besteht darin, sich für eine einheitliche Schreibweise der Tag-Benennungen zu entscheiden und diese Strategie für alle Ressourcentypen umzusetzen.

Entscheiden Sie sich beispielsweise für `Costcenter`, `costcenter` oder `CostCenter` und verwenden Sie dann diese Konvention für alle Tags.

Sie können die EventBridge Konsole, die EventBridge API oder die verwenden, AWS CLI um Tags hinzuzufügen, zu bearbeiten oder zu löschen. Weitere Informationen finden Sie hier:

- TagResource, UntagResourceund ListTagsForResource in der Amazon EventBridge -API-Referenz
- tag-resource , untag-resource und list-tags-for-resource in der -AWS CLI Referenz
- Arbeiten mit dem Tag Editor im Ressourcengruppen-Benutzerhandbuch

Dokumentverlauf

In der folgenden Tabelle werden wichtige Änderungen in den einzelnen Versionen des EventBridge Amazon-Benutzerhandbuchs ab Juli 2019 beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

Änderung	Beschreibung	Veröffentlichungsdatum
Die AWS verwalteten Richtlinien wurden aktualisiert.	AWS GovCloud (US) Regions nur `AmazonEventBridgeFullAccess` und in den `AmazonEventBridgeSchemasFullAccess` Richtlinien nicht enthalten`iam:CreateServiceLinkedRole`, da es nicht verwendet wird. • the section called “Richtlinienaktualisierungen”	9. Mai 2024
Generieren Sie AWS CloudFormation Vorlagen aus Event-Bussen und Regeln.	Sie können jetzt AWS CloudFormation Vorlagen aus Ihren vorhandenen EventBridge Amazon-Eventbussen und -Regeln generieren. • Generieren einer AWS CloudFormation-Vorlage aus einem Amazon-EventBridge-Event-Bus	18. November 2022
Die EventBridge Pipes-Dokumentation wurde veröffentlicht.	Sie können jetzt Pipes erstellen, um Quellen mit Zielen zu verbinden, mit optionaler Filterung und Anreicherung. • Pipes	01. Dezember 2022
Generieren Sie AWS CloudFormation Vorlagen aus Event-Bussen und Regeln.	Sie können jetzt AWS CloudFormation Vorlagen aus Ihren vorhandenen EventBridge Amazon-Eventbussen und -Regeln generieren. • Generieren einer AWS CloudFormation-Vorlage aus einem Amazon-EventBridge-Event-Bus	18. November 2022

Änderung	Beschreibung	Veröffentlichungsdatum
Die AmazonEventBridgePipesFullAccess Richtlinie wurde hinzugefügt.	Bietet vollen Zugriff auf Amazon EventBridge Pipes. • EventBridge Rohrspezifische verwaltete Richtlinien	01. Dezember 2022
Die AmazonEventBridgePipesReadOnlyAccess Richtlinie wurde hinzugefügt.	Bietet schreibgeschützten Zugriff auf Amazon EventBridge Pipes. • EventBridge Rohrspezifische verwaltete Richtlinien	01. Dezember 2022
Die AmazonEventBridgePipesOperatorAccess Richtlinie wurde hinzugefügt.	Bietet schreibgeschützten Zugriff und Bedienerzugriff (d. h. die Möglichkeit, die Ausführung von Pipes zu beenden und zu starten) auf Amazon EventBridge Pipes. • EventBridge Rohrspezifische verwaltete Richtlinien	01. Dezember 2022
Die CloudWatchEventsFullAccess Richtlinie wurde aktualisiert.	Passend zu `AmazonEventBridgeFullAccess` aktualisiert. • AmazonEventBridgeFullAccess Richtlinie	01. Dezember 2022
Die CloudWatchEventsReadOnlyAccess Richtlinie wurde aktualisiert.	Passend zu `AmazonEventBridgeReadOnlyAccess` aktualisiert. • AmazonEventBridgeReadOnlyAccess Richtlinie	01. Dezember 2022

Änderung	Beschreibung	Veröffentlichungsdatum
Die Inhaltsfilterung in Ereignismustern wurde aktualisiert.	Sie können jetzt die Filteroptionen `suffix`, `equals-ignore-case` und `$or` verwenden, um Ereignismuster zu erstellen. • Inhaltsfilterung in EventBridge Amazon-Ereignismustern	14. November 2022
Die AmazonEventBridgeFullAccess Richtlinie wurde aktualisiert.	Es wurden die für die Verwendung von EventBridge Schema Registry und EventBridge Scheduler erforderlichen Berechtigungen hinzugefügt. • AmazonEventBridgeFullAccess Richtlinie	10. November 2022
Die AmazonEventBridgeReadOnlyAccess Richtlinie wurde aktualisiert.	Sie können jetzt Informationen zur EventBridge Schemaregistrierung und zum EventBridge Scheduler anzeigen. • AmazonEventBridgeReadOnlyAccess Richtlinie	10. November 2022
Die Inhaltsfilterung in Ereignismustern wurde aktualisiert.	Sie können jetzt die Filteroptionen `suffix`, `equals-ignore-case` und `$or` verwenden, um Ereignismuster zu erstellen. • Inhaltsfilterung in EventBridge Amazon-Ereignismustern	14. November 2022
Die AmazonEventBridgeFullAccess Richtlinie wurde aktualisiert.	Es wurden die für die Verwendung von EventBridge Schema Registry und EventBridge Scheduler erforderlichen Berechtigungen hinzugefügt. • AmazonEventBridgeFullAccess Richtlinie	10. November 2022

Änderung	Beschreibung	Veröffentlichungsdatum
Die AmazonEventBridgeReadOnlyAccess Richtlinie wurde aktualisiert.	Sie können jetzt Informationen zur EventBridge Schemaregistrierung und zum EventBridge Scheduler anzeigen. • AmazonEventBridgeReadOnlyAccess Richtlinie	10. November 2022
Die AmazonEventBridgeReadOnlyAccess Richtlinie wurde aktualisiert.	Sie können jetzt Endpunktinformationen einsehen. • AmazonEventBridgeReadOnlyAccess Richtlinie	07. April 2022
Unterstützung für globale Endpunkte hinzugefügt.	Amazon unterstützt EventBridge jetzt die Verwendung globaler Endpunkte, um Ihre Anwendung ohne zusätzliche Kosten regional fehlertolerant zu machen. Für weitere Informationen siehe: • Festlegen von Anwendungen als regional fehlertolerant mit globalen Endpunkten und der Ereignisreplikation • `CreateEndpoint`	07. April 2022
Unterstützung für Archive und Ereigniswiederholungen hinzugefügt.	Amazon unterstützt EventBridge jetzt die Verwendung von Archiven zum Speichern von Ereignissen und von Ereigniswiederholungen zum Wiedergeben von Ereignissen aus einem Archiv. Für weitere Informationen siehe: • Archivierung Amazon EventBridge Amazon-Ereignissen. • `CreateArchive` • `StartReplay`	05. November 2020

Änderung	Beschreibung	Veröffentlichungsdatum
Unterstützung für Warteschlangen für unzustellbare Nachrichten und Wiederholungsrichtlinien für Ziele hinzugefügt.	Amazon unterstützt EventBridge jetzt die Verwendung von Warteschlangen mit uneingeschränkten Briefen und die Definition einer Wiederholungsrichtlinie für Ziele. Für weitere Informationen siehe: • Verwenden von Warteschlangen mit unzustellbaren Buchstaben zur Verarbeitung nicht zugestellter Ereignisse. • `PutTargets`	12. Oktober 2020
Unterstützung für Schemata im JSONSchema-Draft4-Format hinzugefügt.	Amazon unterstützt EventBridge jetzt Schemas im Format JSONSchema Draft 4. Sie können jetzt auch Schemas mithilfe der API exportieren. EventBridge Für weitere Informationen siehe Folgendes. • EventBridge Amazon-Schemas • `Export`in der API-Referenz zur EventBridge Schemaregistrierung.	28. September 2020
Ressourcenbasierte Richtlinien für die Schema Registry EventBridge	Die Amazon EventBridge Schema Registry unterstützt jetzt ressourcenbasierte Richtlinien. Weitere Informationen finden Sie unter den folgenden Topics. • Ressourcenbasierte Richtlinien für Amazon-EventBridge-Schemata • `Policy`in der API-Referenz zur EventBridge Schemaregistry • RegistryPolicy Ressourcentyp im AWS CloudFormation Benutzerhandbuch	30. April 2020

Änderung	Beschreibung	Veröffentlichungsdatum
Tags für Ereignisbusse	Bei dieser Version können Sie Tags für Ereignisbusse erstellen und verwalten. Sie können Tags hinzufügen, wenn Sie einen Ereignisbus erstellen, und vorhandene Tags hinzufügen oder verwalten, indem Sie die betreffende API aufrufen. Weitere Informationen finden Sie unter den folgenden Topics. • Amazon- EventBridge Tags • Tagbasierte Richtlinien • `TagResource` • `UntagResource` • `ListTagsForResource`	24. Februar 2020
Erhöhte Servicekontingente	Amazon EventBridge hat die Kontingente für Aufrufe und für erhöht. `PutEvents` Die Kontingente sind je nach Region unterschiedlich und können bei Bedarf erhöht werden.	11. Februar 2020

Änderung	Beschreibung	Veröffentlichungsdatum
Neues Thema zum Transformieren der Zieleingabe sowie ein Link zu Anwendungs-Auto-Scaling-Ereignissen hinzugefügt.	Verbesserte Dokumentation zum Input-Transformer. • Transformation Amazon EventBridge Amazon-Eingaben • Verwenden des Input-Transformers, um Daten aus einem Ereignis zu extrahieren und diese Daten in das Ziel einzugeben • Tutorial: Verwenden des Eingabe-Transformators, um die EventBridge-Ausgabe an das Ereignisziel anzupassen Link zu Anwendungs-Auto-Scaling-Ereignissen hinzugefügt. • Auto Scaling-Ereignisse für Anwendungen und EventBridge • Ereignisse im Zusammenhang mit Dienstleistungen AWS	20. Dezember 2019
Inhaltsbasierte Filterung		19. Dezember 2019
Links zu Amazon Augmented AI-Ereignisbeispielen hinzugefügt.	Im Amazon SageMaker Developer Guide wurde ein Link zum Thema Amazon Augmented AI hinzugefügt, der Beispielveranstaltungen für Amazon Augmented AI enthält. Weitere Informationen finden Sie unter den folgenden Topics. • Verwenden von Ereignissen in Amazon Augmented AI • Ereignisse im Zusammenhang mit Dienstleistungen AWS	13. Dezember 2019

Änderung	Beschreibung	Veröffentlichungsdatum
Links zu Amazon-Chime-Ereignisbeispielen hinzugefügt.	Link zum Thema Amazon Chime hinzugefügt, das Beispielereignisse für diesen Service enthält. Weitere Informationen finden Sie unter den folgenden Topics. • Automatisieren von Amazon Chime mit EventBridge • Ereignisse im Zusammenhang mit Dienstleistungen AWS	12. Dezember 2019
EventBridge Amazon-Schemas	Sie können jetzt Schemas verwalten und Codebindungen für Ereignisse in Amazon generieren. EventBridge Weitere Informationen finden Sie unter den folgenden Topics. • EventBridge Amazon-Schemas • EventBridge API-Referenz für Schemas • EventSchemas Referenz zum Ressourcentyp in AWS CloudFormation	1. Dezember 2019
AWS CloudFormation Unterstützung für Event Buses	AWS CloudFormation unterstützt jetzt die EventBus Ressource. Es unterstützt auch den EventBusName Parameter sowohl in den Ressourcen als EventBusPolicy auch in der Regel. Weitere Informationen finden Sie unter Amazon EventBridge Resource Type Reference.	7. Oktober 2019
Neuer Service	Erste Version von Amazon EventBridge.	11. Juli 2019

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.