Benutzer-Leitfaden

AWS Elemental MediaStore

AWS Elemental MediaStore: Benutzer-Leitfaden

Table of Contents

Hinweis zum Ende des Supports: Am 13. November 2025 AWS wird der Support für AWS Elemental MediaStore eingestellt. Nach dem 13. November 2025 können Sie nicht mehr auf die MediaStore Konsole oder MediaStore die Ressourcen zugreifen. Weitere Informationen finden Sie in diesem Blogbeitrag.

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.

Was ist AWS Elemental MediaStore?

AWS Elemental MediaStore ist ein Service zur Erstellung und Speicherung von Videos, der die hohe Leistung und sofortige Konsistenz bietet, die für die Live-Erzeugung erforderlich sind. Mit MediaStore können Sie Video-Assets als Objekte in Containern verwalten, um zuverlässige, cloudbasierte Medien-Workflows zu erstellen.

Um den Service zu nutzen, laden Sie Ihre Objekte aus einer Quelle, z. B. von einem Encoder oder Datenfeed, in einen Container hoch, den Sie in MediaStore erstellen.

MediaStore ist eine hervorragende Wahl für das Speichern fragmentierter Videodateien, wenn Sie hohe Konsistenz, Lese- und Schreibvorgänge mit geringer Latenz und die Fähigkeit benötigen, große Mengen gleichzeitiger Anfragen zu verarbeiten. Wenn Sie keine Live-Streaming-Videos bereitstellen, sollten Sie stattdessen Amazon Simple Storage Service (Amazon S3) verwenden.

Themen

MediaStore Konzepte und Terminologie von AWS Elemental

ARN

Ein Amazon-Ressourcenname.

Fließtext

Die Daten, die in ein Objekt hochgeladen werden sollen.

(Byte)-Bereich

Eine Untermenge der Objektdaten, die angesprochen werden. Weitere Informationen finden Sie unter Bereich in der HTTP-Spezifikation.

Container

Ein Namespace, der Objekte enthält. Ein Container hat einen Endpunkt, den Sie zum Schreiben und Abrufen von Objekten und zum Anfügen von Zugriffsrichtlinien verwenden können.

Endpunkt

Ein Einstiegspunkt zum MediaStore Service, der als HTTPS-Root-URL angegeben wird.

ETag

Ein Entity-Tag, das ist ein Hash der Objektdaten.

Ordner

Eine Abschnitt eines Containers. Ein Ordner kann Objekte und andere Ordner enthalten.

Item

Ein Begriff für Objekte und Ordner.

Object

Ein Asset, ähnlich einem Amazon S3 S3-Objekt. Objekte sind die grundlegenden Einheiten, die in MediaStore gespeichert sind. Der Service akzeptiert alle Dateitypen.

Bereitstellungsservice

MediaStore wird als Ursprungsdienst betrachtet, da er die Vertriebsstelle für die Bereitstellung von Medieninhalten ist.

Pfad

Eine eindeutige ID für ein Objekt oder einen Ordner, die deren Position im Container angibt.

Teil

Eine Teilmenge der Daten (Block) eines Objekts.

Richtlinie

Eine IAM-Richtlinie.

Ressource

Eine Entität in AWS, mit der Sie arbeiten können. Jeder AWS-Ressource ist ein Amazon-Ressourcenname (ARN) zugeordnet, der als eindeutige Kennung fungiert. In MediaStore, das ist die Ressource und ihr ARN-Format:

- Container: `aws:mediastore:region:account-id:container/:containerName`

Zugehörige Services

- Amazon CloudFront ist ein globaler Content Delivery Network (CDN) -Service, der Daten und Videos sicher an Ihre Zuschauer liefert. Verwenden Sie CloudFront, um Inhalte mit der bestmöglichen Leistung bereitzustellen. Weitere Informationen finden Sie im Amazon CloudFront Developer Guide.
- CloudFormationist ein Service, der Sie bei der Modellierung und Einrichtung Ihrer AWS Ressourcen unterstützt. Sie erstellen eine Vorlage, die alle gewünschten AWS Ressourcen beschreibt (z. B. MediaStore Container), und CloudFormation kümmert sich um die Bereitstellung und Konfiguration dieser Ressourcen für Sie. Sie müssen AWS Ressourcen nicht einzeln erstellen und konfigurieren und herausfinden, was wovon abhängt, sondern CloudFormation kümmert sich um all das. Weitere Informationen finden Sie im AWS CloudFormation -Benutzerhandbuch.
- AWS CloudTrailist ein Service, mit dem Sie die CloudTrail API-Aufrufe für Ihr Konto, einschließlich der Aufrufe der AWS-Managementkonsole AWS CLI, und anderer Services überwachen können. Weitere Informationen finden Sie im AWS CloudTrail -Benutzerhandbuch.
- Amazon CloudWatch ist ein Überwachungsdienst für AWS Cloud-Ressourcen und die Anwendungen, auf denen Sie laufen AWS. Verwenden Sie CloudWatch Ereignisse, um Statusänderungen von Containern und Objekten in zu verfolgen MediaStore. Weitere Informationen finden Sie in der CloudWatch Amazon-Dokumentation.
- AWS Identity and Access Management (IAM) ist ein Webservice, mit dem Sie den Zugriff Ihrer Benutzer auf AWS Ressourcen sicher kontrollieren können. Verwenden Sie IAM, um zu kontrollieren, wer Ihre AWS Ressourcen verwenden kann (Authentifizierung) und welche Ressourcen Benutzer auf welche Weise verwenden können (Autorisierung). Weitere Informationen finden Sie unter Einrichtung von AWS Elemental MediaStore.
- Amazon Simple Storage Service (Amazon S3) ist ein Objektspeicher, der entwickelt wurde, um beliebige Datenmengen von überall zu speichern und abzurufen. Weitere Informationen finden Sie in der Amazon S3-Dokumentation.

Zugreifen auf AWS Elemental MediaStore

Sie können MediaStore mit einer der folgenden Methoden zugreifen:

- AWS-Managementkonsole — Die Verfahren in diesem Handbuch erläutern, wie Sie die AWS-Managementkonsole zur Ausführung von Aufgaben für verwenden MediaStore. So greifen MediaStore Sie über die Konsole zu:

```
https://<region>.console.aws.amazon.com/mediastore/home
```

- AWS Command Line Interface— Weitere Informationen finden Sie im AWS Command Line Interface Benutzerhandbuch. So greifen MediaStore Sie über den CLI-Endpunkt zu:

```
aws mediastore
```

- MediaStore API — Wenn Sie eine Programmiersprache verwenden, für die kein SDK verfügbar ist, finden Sie in der AWS Elemental MediaStore API-Referenz Informationen zu API-Aktionen und zum Stellen von API-Anfragen. So greifen MediaStore Sie über den REST-API-Endpunkt zu:

```
https://mediastore.<region>.amazonaws.com
```

- AWS SDKs — Wenn Sie eine Programmiersprache verwenden, für die AWS ein SDK bereitstellt, können Sie ein SDK für den Zugriff verwenden MediaStore. SDKs vereinfachen Sie die Authentifizierung, lassen sich problemlos in Ihre Entwicklungsumgebung integrieren und bieten Sie einfachen Zugriff auf MediaStore Befehle. Weitere Informationen finden Sie unter Tools für Amazon Web Services.
- AWS-Tools für Windows PowerShell — Weitere Informationen finden Sie im AWS -Tools für PowerShell Benutzerhandbuch.

Preise für AWS Elemental MediaStore

Wie bei anderen AWS Produkten gibt es keine Verträge oder Mindestverpflichtungen für die Nutzung MediaStore. Ihnen wird eine Gebühr pro GB für die Aufnahme von Inhalten in den Service und eine monatliche Gebühr pro GB für Inhalte, die Sie im Service speichern, in Rechnung gestellt. Weitere Informationen finden Sie unter AWS Elemental MediaStore Pricing.

Regionen und Endpunkte für AWS Elemental MediaStore

Um die Datenlatenz in Ihren Anwendungen zu reduzieren, MediaStore bietet es einen regionalen Endpunkt, an dem Sie Ihre Anfrage stellen können:

```
https://mediastore.<region>.amazonaws.com
```

Eine vollständige Liste der AWS-Regionen, in denen sie verfügbar MediaStore ist, finden Sie unter AWS Elemental MediaStore Endpoints and Quotas in der AWS General Reference.

Einrichtung von AWS Elemental MediaStore

Dieser Abschnitt führt Sie durch die Schritte, die erforderlich sind, um Benutzer für den Zugriff auf AWS Elemental MediaStore zu konfigurieren. Hintergrundinformationen und zusätzliche Informationen zur Identitäts- und Zugriffsverwaltung für finden Sie MediaStore unterIdentity and Access Management für AWS Elemental MediaStore.

Führen Sie die folgenden Schritte aus MediaStore, um mit der Nutzung von AWS Elemental zu beginnen.

Themen

Melden Sie sich für eine an 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/die Anmeldung.
2. Folgen Sie den Online-Anweisungen.

   Ein Teil des Anmeldevorgangs umfasst den Empfang eines Telefonanrufs oder einer Textnachricht und die Eingabe eines Bestätigungscodes auf der Telefontastatur.

   Wenn Sie sich für eine anmelden AWS-Konto, wird eine Root-Benutzer des AWS-Kontoserstellt. Der Root-Benutzer hat Zugriff auf alle AWS-Services und Ressourcen des Kontos. Als bewährte Sicherheitsmethode weisen Sie einem Administratorbenutzer Administratorzugriff zu und verwenden Sie nur den Root-Benutzer, um Aufgaben auszuführen, die Root-Benutzerzugriff erfordern.

AWS sendet Ihnen nach Abschluss des Anmeldevorgangs eine Bestätigungs-E-Mail. Du kannst jederzeit deine aktuellen Kontoaktivitäten einsehen und dein Konto verwalten, indem du zu https://aws.amazon.com/gehst und Mein Konto auswählst.

Erstellen eines Benutzers 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-Managementkonsoleals 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 eines Benutzers 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 Administratorbenutzer im IAM Identity Center Benutzerzugriff.

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

Anmelden als Administratorbenutzer

- 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 Access-Portal.

Weiteren Benutzern Zugriff zuweisen

1. Erstellen Sie im IAM-Identity-Center einen Berechtigungssatz, der den bewährten Vorgehensweisen für die Anwendung von geringsten Berechtigungen folgt.

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

   Eine genaue Anleitung finden Sie unter Gruppen hinzufügen im AWS IAM Identity Center Benutzerhandbuch.

Erste Schritte mit AWS Elemental MediaStore

In diesem Tutorial „Erste Schritte“ erfahren Sie, wie Sie mit AWS Elemental MediaStore einen Container erstellen und ein Objekt hochladen.

Themen

Schritt 1: Greifen Sie auf AWS Elemental zu MediaStore

Nachdem Sie Ihr AWS-Konto eingerichtet und Benutzer und Rollen erstellt haben, melden Sie sich bei der Konsole für AWS Elemental MediaStore an.

So greifen Sie auf AWS Elemental zu MediaStore

- Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.

> **Note**
>
> Sie können sich mit beliebigen IAM-Anmeldeinformationen anmelden, die Sie für dieses Konto erstellt haben. Weitere Informationen über das Erstellen IAM-Anmeldeinformationen finden Sie unter Einrichtung von AWS Elemental MediaStore.

Schritt 2: Erstellen eines Containers

Sie verwenden Container in AWS Elemental MediaStore , um Ihre Ordner und Objekte zu speichern. Sie können Container verwenden, um verwandte Objekte auf ähnliche Weise zu gruppieren, wie Sie ein Verzeichnis verwenden, um Dateien in einem Dateisystem zu gruppieren. Es entstehen Ihnen keine Kosten, wenn Sie Container erstellen; Gebühren fallen nur dann an, wenn Sie ein Objekt in einen Container hochladen.

Erstellen eines Containers

1. Wählen Sie auf der Seite Containers (Container) die Option Create Container (Container erstellen) aus.
2. Geben Sie für Container name (Containername) den Namen für Ihren Container ein. Weitere Informationen finden Sie unter Regeln für Containernamen.
3. Wählen Sie Container erstellen. AWS Elemental MediaStore fügt den neuen Container zu einer Liste von Containern hinzu. Anfänglich ist der Status des Containers Creating (Wird erstellt), dann wechselt er zu Active (Aktiv).

Schritt 3: Hochladen eines Objekts

Sie können Objekte (jeweils bis zu 25 MB) in einen Container oder in einen Ordner innerhalb eines Containers hochladen. Um ein Objekt in einen Ordner hochzuladen, geben Sie den Pfad zum Ordner an. Wenn der Ordner bereits existiert, MediaStore speichert AWS Elemental das Objekt in dem Ordner. Wenn der Ordner nicht vorhanden, legt der Service ihn an und speichert das Objekt in dem Ordner.

> **Note**
>
> Objektdateinamen dürfen nur Buchstaben, Ziffern, Punkte (.), Unterstriche (_), Tilden (~) und Bindestriche (-) enthalten.

So laden Sie ein Objekt hoch

1. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, den Sie soeben erstellt haben. Die Detailseite für den Container wird angezeigt.
2. Wählen Sie Upload object (Objekt hochladen).
3. Geben Sie für Target path (Zielpfad) einen Pfad für die Ordner ein. Beispiel, `premium/canada`. Wenn einer der Ordner im Pfad noch nicht existiert, MediaStore erstellt AWS Elemental ihn automatisch.
4. Wählen Sie für Object (Objekt) Browse (Durchsuchen).
5. Navigieren Sie zum entsprechenden Ordner und wählen Sie ein Objekt zum Hochladen aus.
6. Wählen Sie Open (Öffnen) und anschließend Upload (Hochladen).

Schritt 4: Zugreifen auf ein Objekt

Sie können Ihre Objekte auf einen bestimmten Endpunkt herunterladen.

1. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, der das Objekt enthält, das Sie herunterladen möchten.
2. Wenn sich das Objekt, das Sie herunterladen möchten, in einem Unterordner befindet, wählen Sie so lange Ordnernamen aus, bis Sie das Objekt sehen.
3. Wählen Sie den Namen des Objekts.
4. Wählen Sie auf der Detailseite für das Objekt die Option Download (Herunterladen).

Behälter in AWS Elemental MediaStore

Sie verwenden Container MediaStore , um Ihre Ordner und Objekte zu speichern. Verwandte Objekte können in Containern gruppiert werden, auf ähnliche Weise, wie Sie ein Verzeichnis verwenden, um Dateien in einem Dateisystem zu gruppieren. Es entstehen Ihnen keine Kosten, wenn Sie Container erstellen; Gebühren fallen nur dann an, wenn Sie ein Objekt in einen Container hochladen. Weitere Informationen zu Gebühren finden Sie unter AWS Elemental MediaStore Pricing.

Themen

Regeln für Containernamen

Wenn Sie einen Namen für Ihren Container wählen, beachten Sie Folgendes:

- Der Name muss innerhalb des aktuellen Kontos für die aktuelle AWS-Region eindeutig sein.
- Der Name kann Großbuchstaben, Kleinbuchstaben, Ziffern und Unterstriche (_) enthalten.
- Der Name muss zwischen 1 und 255 Zeichen lang sein.
- Bei den Namen muss die Groß- und Kleinschreibung beachtet werden. Sie können beispielsweise einen Container mit dem Namen `myContainer` und einen Ordner mit dem Namen `mycontainer` verwenden, weil diese Namen eindeutig sind.
- Ein Container kann nach dem Erstellen nicht mehr umbenannt werden.

Erstellen eines Containers

Sie können bis zu 100 Container für jedes AWS-Konto erstellen. Sie können eine beliebige Anzahl von Ordnern erstellen, solange sie nicht mehr als 10 Ebenen innerhalb eines Containers verschachtelt sind. Darüber hinaus können Sie beliebig viele Objekte in jeden Container hochladen.

> **Tip**
>
> Sie können einen Container auch automatisch mithilfe einer CloudFormation Vorlage erstellen. Die CloudFormation -Vorlage verwaltet Daten für fünf API-Aktionen: Erstellung eines Containers, Einstellung der Zugriffsprotokollierung, Aktualisierung der Standard-Containerrichtlinie, Hinzufügen einer CORS-Richtlinie (Cross-Origin Resource Sharing) und Hinzufügen einer Objektlebenszyklusrichtlinie. Weitere Informationen finden Sie im AWS CloudFormation -Benutzerhandbuch.

Einen Container erstellen (Konsole)

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.
2. Wählen Sie auf der Seite Containers (Container) die Option Create Container (Container erstellen) aus.
3. Geben Sie für Container den Namen für den Container ein. Weitere Informationen finden Sie unter Regeln für Containernamen.
4. Wählen Sie Container erstellen. AWS Elemental MediaStore fügt den neuen Container zu einer Liste von Containern hinzu. Anfänglich ist der Status des Containers Creating (Wird erstellt), dann wechselt er zu Active (Aktiv).

Erstellen eines Containers (AWS CLI)

- Verwenden Sie in AWS CLI der den `create-container` folgenden Befehl:

```
aws mediastore create-container --container-name ExampleContainer --region us-west-2
```

Im folgenden Beispiel finden Sie den Rückgabewert:

```
{
    "Container": {
        "AccessLoggingEnabled": false,
        "CreationTime": 1563557265.0,
        "Name": "ExampleContainer",
        "Status": "CREATING",
        "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleContainer"
```

```
    }
}
```

Anzeigen der Details für einen Container

Details für einen Container sind unter anderem die Container-Richtlinie, Endpunkt, ARN und den Zeitpunkt der Erstellung.

Anzeigen der Details für einen Container (Konsole)

1. Öffnen Sie die MediaStore Konsole unter. https://console.aws.amazon.com/mediastore/
2. Wählen Sie auf der Seite Containers (Container) den Containernamen aus.

   Die Seite mit den Containerdetails wird angezeigt. Diese Seite ist in zwei Abschnitte unterteilt:

   - Den Abschnitt Objects (Objekte), der die Objekte und Ordner im Container auflistet.
   - Den Container-Richtlinienabschnitt, der die ressourcenbasierte Richtlinie zeigt, die diesem Container zugeordnet ist. Weitere Informationen über Ressourcenrichtlinien finden Sie unter Containerrichtlinien.

Anzeigen der Details für einen Container (AWS CLI)

- Verwenden Sie im AWS CLI den folgenden `describe-container` Befehl:

  ```
  aws mediastore describe-container --container-name ExampleContainer --region us-
  west-2
  ```

  Im folgenden Beispiel finden Sie den Rückgabewert:

  ```
  {
      "Container": {
          "CreationTime": 1563558086.0,
          "AccessLoggingEnabled": false,
          "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
  ExampleContainer",
          "Status": "ACTIVE",
          "Name": "ExampleContainer",
          "Endpoint": "https://aaabbbcccdddee.data.mediastore.us-
  west-2.amazonaws.com"
  ```

```
    }
}
```

Anzeigen einer Liste von Containern

Sie können eine Liste aller Container anzeigen, die mit Ihrem Konto verknüpft sind.

Eine Liste der Container anzeigen (Konsole)

- Öffnen Sie die MediaStore Konsole unter. https://console.aws.amazon.com/mediastore/

  Die Seite Containers (Container) wird angezeigt und listet alle Container auf, die Ihrem Konto zugeordnet sind.

Eine Liste der Container anzeigen (AWS CLI)

- Verwenden Sie in der AWS CLI den `list-containers` Befehl.

  ```
  aws mediastore list-containers --region us-west-2
  ```

  Im folgenden Beispiel finden Sie den Rückgabewert:

  ```
  {
      "Containers": [
          {
              "CreationTime": 1505317931.0,
              "Endpoint": "https://aaabbbcccdddee.data.mediastore.us-
  west-2.amazonaws.com",
              "Status": "ACTIVE",
              "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
  ExampleLiveDemo",
              "AccessLoggingEnabled": false,
              "Name": "ExampleLiveDemo"
          },
          {
              "CreationTime": 1506528818.0,
              "Endpoint": "https://fffggghhhiiijj.data.mediastore.us-
  west-2.amazonaws.com",
              "Status": "ACTIVE",
  ```

```
            "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleContainer",
            "AccessLoggingEnabled": false,
            "Name": "ExampleContainer"
        }
    ]
}
```

Löschen eines Containers

Sie können einen Container nur löschen, wenn er keine Objekte enthält.

Einen Container löschen (Konsole)

1. Öffnen Sie die MediaStore Konsole unter. https://console.aws.amazon.com/mediastore/
2. Wählen Sie auf der Seite Containers (Container) die Option links neben dem Containernamen.
3. Wählen Sie Löschen aus.

Einen Container löschen (AWS CLI)

- Verwenden Sie im AWS CLI den folgenden `delete-container` Befehl:

  ```
  aws mediastore delete-container --container-name=ExampleLiveDemo --region us-west-2
  ```

  Dieser Befehl hat keinen Rückgabewert.

Richtlinien in AWS Elemental MediaStore

Sie können eine oder mehrere dieser Richtlinien auf Ihren AWS Elemental MediaStore Container anwenden:

- Container-Richtlinie — Legt die Zugriffsrechte für alle Ordner und Objekte innerhalb des Containers fest. MediaStore legt eine Standardrichtlinie fest, die es Benutzern ermöglicht, alle MediaStore Operationen am Container durchzuführen. Diese Richtlinie legt fest, dass alle Vorgänge über HTTPS ausgeführt werden müssen. Nachdem Sie einen Container erstellt haben, können Sie die Containerrichtlinie bearbeiten.
- CORS-Richtlinie (Cross-Origin Resource Sharing) — Ermöglicht Client-Webanwendungen aus einer Domäne, mit Ressourcen in einer anderen Domäne zu interagieren. MediaStore legt keine CORS-Standardrichtlinie fest.
- Metrik-Richtlinie — Ermöglicht MediaStore das Senden von Metriken an Amazon CloudWatch. MediaStore legt keine Standardrichtlinie für Metriken fest.
- Objektlebenszyklus-Richtlinie — Steuert, wie lange Objekte in einem MediaStore Container verbleiben. MediaStorelegt keine standardmäßige Objektlebenszyklusrichtlinie fest.

Container-Richtlinien in AWS Elemental MediaStore

Jeder Container hat eine ressourcenbasierte Richtlinie, die die Zugriffsrechte auf alle Ordner und Objekte in diesem Container regelt. Die Standardrichtlinie, die automatisch an alle neuen Container angehängt wird, ermöglicht den Zugriff auf alle AWS Elemental MediaStore Elemental-Operationen auf dem Container. Sie gibt vor, dass dieser Zugriff die Bedingung hat, dass HTTPS für die Operationen verwendet wird. Nach dem Erstellen eines Containers können Sie die Richtlinie bearbeiten, die an diesen Container angehängt ist.

Sie können auch eine Objektlebenszyklus-Richtlinie angeben, die das Ablaufdatum der Objekte in einem Container regelt. Wenn Objekte das angegebene Höchstalter erreichen, löscht der Service die Objekte aus dem Container.

Topics

- Anzeigen einer Containerrichtlinie
- Bearbeiten einer Containerrichtlinie
- Beispiel-Containerrichtlinien

Anzeigen einer Containerrichtlinie

Sie können die Konsole oder die verwenden, AWS CLI um die ressourcenbasierte Richtlinie eines Containers anzuzeigen.

Anzeige einer Containerrichtlinie (Konsole)

1. Öffnen Sie die MediaStore Konsole unter. https://console.aws.amazon.com/mediastore/
2. Wählen Sie auf der Seite Containers (Container) den Containernamen aus.

   Die Seite mit den Containerdetails wird angezeigt. Die Richtlinie wird im Abschnitt Container Policy (Containerrichtlinie) angezeigt.

Anzeige einer Containerrichtlinie (AWS CLI)

- Verwenden Sie in der AWS CLI den `get-container-policy` folgenden Befehl:

  ```
  aws mediastore get-container-policy --container-name ExampleLiveDemo --region us-west-2
  ```

  Im folgenden Beispiel finden Sie den Rückgabewert:

  ```
  {
    "Policy": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Sid": "PublicReadOverHttps",
          "Effect": "Allow",
          "Principal": {
            "AWS": "arn:aws:iam::111122223333:root",
          },
          "Action": [
            "mediastore:GetObject",
            "mediastore:DescribeObject",
          ],
          "Resource": "arn:aws:mediastore:us-west-2:111122223333:container/ExampleLiveDemo/*",
          "Condition": {
            "Bool": {
              "aws:SecureTransport": "true"
  ```

```
          }
        }
      }
    ]
  }
}
```

Bearbeiten einer Containerrichtlinie

Sie können die Berechtigungen in der Standard-Containerrichtlinie bearbeiten oder eine neue Richtlinie erstellen, die die Standardrichtlinie ersetzt. Es dauert bis zu fünf Minuten, bis die neue Richtlinie wirksam wird.

Bearbeiten einer Containerrichtlinie (Konsole)

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.
2. Wählen Sie auf der Seite Containers (Container) den Containernamen aus.
3. Wählen Sie Edit policy (Richtlinie bearbeiten). Beispiele, die zeigen, wie verschiedene Berechtigungen eingerichtet werden, finden Sie unter the section called “Beispiel-Containerrichtlinien”.
4. Nehmen Sie die entsprechenden Änderungen vor und wählen Sie dann Save (Speichern) aus.

Bearbeiten einer Containerrichtlinie (AWS CLI)

1. Erstellen Sie eine Datei, mit der die Containerrichtlinie definiert wird:

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadOverHttps",
      "Effect": "Allow",
      "Action": ["mediastore:GetObject", "mediastore:DescribeObject"],
      "Principal": "*",
      "Resource": "arn:aws:mediastore:us-
west-2:111122223333:container/ExampleLiveDemo/*",
```

```
      "Condition": {
        "Bool": {
            "aws:SecureTransport": "true"
        }
      }
    }
  ]
}
```

2. Verwenden Sie in der AWS CLI den `put-container-policy` folgenden Befehl:

```
aws mediastore put-container-policy --container-name ExampleLiveDemo --
policy file://ExampleContainerPolicy.json --region us-west-2
```

Dieser Befehl hat keinen Rückgabewert.

Beispiel-Containerrichtlinien

Die folgenden Beispiele zeigen Containerrichtlinien, die für verschiedene Benutzergruppen erstellt wurden.

Topics

- Beispiel-Containerrichtlinie: Standard
- Beispiel-Containerrichtlinie: Öffentlicher Lesezugriff über HTTPS
- Beispiel-Containerrichtlinie: Öffentlicher Lesezugriff über HTTP oder HTTPS
- Beispiel-Containerrichtlinie: Kontoübergreifender Lesezugriff – HTTP-fähig
- Beispiel-Containerrichtlinie: Kontoübergreifender Lesezugriff über HTTPS
- Beispiel-Containerrichtlinie: Kontoübergreifender Lesezugriff für eine Rolle
- Beispiel-Containerrichtlinie: Kontoübergreifender Vollzugriff für eine Rolle
- Beispiel-Containerrichtlinie: Zugriff auf bestimmte IP-Adressen beschränkt

Beispiel-Containerrichtlinie: Standard

Wenn Sie einen Container erstellen, hängt AWS Elemental MediaStore automatisch die folgende ressourcenbasierte Richtlinie an:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "MediaStoreFullAccess",
            "Action": [
                "mediastore:*"
            ],
            "Principal": {
                "AWS": "arn:aws:iam::333333333333:root"
            },
            "Effect": "Allow",
            "Resource": "arn:aws:mediastore:us-
east-2:333333333333:container/<container name>/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "true"
                }
            }
        }
    ]
}
```

Die Richtlinie ist in den Service integriert, Sie müssen sie also nicht erstellen. Sie können die Richtlinie für den Container jedoch bearbeiten, wenn die Berechtigungen in der Standardrichtlinie nicht mit den Berechtigungen übereinstimmen, die Sie für den Container verwenden möchten.

Die Standardrichtlinie, die allen neuen Containern zugewiesen wird, erlaubt den Zugriff auf alle MediaStore -Operationen für den Container. Sie gibt vor, dass dieser Zugriff die Bedingung hat, dass HTTPS für die Operationen verwendet wird.

Beispiel-Containerrichtlinie: Öffentlicher Lesezugriff über HTTPS

Diese Beispielrichtlinie erlaubt Benutzern das Abrufen eines Objekts über eine HTTPS-Anfrage. Sie ermöglicht allen Benutzern über eine sichere SSL/TLS Verbindung Lesezugriff: authentifizierte Benutzer und anonyme Benutzer (Benutzer, die nicht angemeldet sind). Die Anweisung hat den Namen `PublicReadOverHttps`. Sie erlaubt Zugriff auf die Operationen `GetObject` und `DescribeObject` sowie auf beliebige Objekte (wie durch den * am Ende des Ressourcenpfads

angegeben). Sie gibt vor, dass dieser Zugriff die Bedingung hat, dass HTTPS für die Operationen verwendet wird:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadOverHttps",
            "Effect": "Allow",
            "Action": [
                "mediastore:GetObject",
                "mediastore:DescribeObject"
            ],
            "Principal": "*",
            "Resource": "arn:aws:mediastore:us-
east-2:333333333333:container/<container name>/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "true"
                }
            }
        }
    ]
}
```

Beispiel-Containerrichtlinie: Öffentlicher Lesezugriff über HTTP oder HTTPS

Diese Beispielrichtlinie erlaubt Zugriff auf die Operationen `GetObject` und `DescribeObject` sowie auf beliebige Objekte (wie durch den * am Ende des Ressourcenpfads angegeben). Sie erlaubt allen Benutzern Lesezugriff: allen authentifizierten Benutzern und anonymen Benutzern (Benutzer, die nicht angemeldet sind):

JSON

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

```
            "Sid": "PublicReadOverHttpOrHttps",
            "Effect": "Allow",
            "Action": [
                "mediastore:GetObject",
                "mediastore:DescribeObject"
            ],
            "Principal": "*",
            "Resource": "arn:aws:mediastore:us-
east-2:333333333333:container/<container name>/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "false"
                }
            }
        }
    ]
}
```

Beispiel-Containerrichtlinie: Kontoübergreifender Lesezugriff – HTTP-fähig

Diese Beispielrichtlinie erlaubt Benutzern das Abrufen eines Objekts über eine HTTP-Anfrage. Sie erlaubt diesen Zugriff authentifizierten Benutzern mit kontoübergreifendem Zugriff. Das Objekt muss nicht auf einem Server mit einem SSL/TLS Zertifikat gehostet werden:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CrossAccountReadOverHttpOrHttps",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::333333333333:root"
            },
            "Action": [
                "mediastore:GetObject",
                "mediastore:DescribeObject"
            ],
            "Resource": "arn:aws:mediastore:us-
east-2:333333333333:container/<container name>/*",
```

```
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "true"
                }
            }
        }
    ]
}
```

Beispiel-Containerrichtlinie: Kontoübergreifender Lesezugriff über HTTPS

Diese Beispielrichtlinie ermöglicht den Zugriff auf die `DescribeObject` Operationen `GetObject` und für jedes Objekt (wie durch das * am Ende des Ressourcenpfads angegeben), das dem Root-Benutzer des angegebenen Objekts gehört<other acct number>. Sie gibt vor, dass dieser Zugriff die Bedingung hat, dass HTTPS für die Operationen verwendet wird:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CrossAccountReadOverHttps",
            "Effect": "Allow",
            "Action": [
                "mediastore:GetObject",
                "mediastore:DescribeObject"
            ],
            "Principal": {
                "AWS": "arn:aws:iam::333333333333:root"
            },
            "Resource": "arn:aws:mediastore:us-
east-2:333333333333:container/<container name>/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "true"
                }
            }
        }
    ]
```

```
}
```

Beispiel-Containerrichtlinie: Kontoübergreifender Lesezugriff für eine Rolle

Die Beispielrichtlinie erlaubt den Zugriff auf die Operationen `GetObject` und `DescribeObject` sowie auf beliebige Objekte (wie durch den * am Ende des Ressourcenpfads angegeben), die im Besitz der angegebenen <Eigentümer-Kontonummer> sind. Sie erlaubt diesen Zugriff jedem Benutzer der <anderen Kontonummer>, wenn dieses Konto die Rolle übernommen hat, die in <Rollenname> angegeben ist:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CrossAccountRoleRead",
      "Effect": "Allow",
      "Action": ["mediastore:GetObject", "mediastore:DescribeObject"],
      "Principal":{
        "AWS": "arn:aws:iam::<other acct number>:role/<role name>"},
      "Resource": "arn:aws:mediastore:<region>:<owner acct number>:container/<container
 name>/*",
    }
  ]
}
```

Beispiel-Containerrichtlinie: Kontoübergreifender Vollzugriff für eine Rolle

Diese Beispielrichtlinie erlaubt kontoübergreifenden Zugriff zur Aktualisierung eines beliebigen Objekts im Konto, wenn der Benutzer über HTTP angemeldet ist. Außerdem erlaubt sie kontoübergreifenden Zugriff, um Objekte über HTTP oder HTTPS in einem Konto zu löschen, herunterzuladen und zu beschreiben, das die angegebene Rolle angenommen hat.

- Die erste Anweisung ist `CrossAccountRolePostOverHttps`. Sie erlaubt den Zugriff auf die Operation `PutObject` für ein beliebiges Objekt und erlaubt diesen Zugriff jedem beliebigen Benutzer des angegebenen Kontos, wenn dieses Konto die Rolle übernommen hat, die in <Rollenname> angegeben ist. Sie gibt an, dass dieser Zugriff die Bedingung hat, dass HTTPS für die Operation gefordert wird (diese Bedingung muss immer enthalten sein, wenn Zugriff auf `PutObject` erteilt wird).

Mit anderen Worten, jeder Prinzipal, der kontoübergreifenden Zugriff besitzt, kann auf `PutObject` zugreifen, aber nur über HTTPS.

- Die zweite Anweisung ist `CrossAccountFullAccessExceptPost`. Sie erlaubt Zugriff auf alle Operationen außer `PutObject` für jedes Objekt. Sie erlaubt diesen Zugriff jedem Benutzer des angegebenen Kontos, wenn dieses Konto die Rolle übernommen hat, die in <Rollenname> angegeben ist. Dieser Zugriff hat nicht die Bedingung hat, dass HTTPS für die Operationen gefordert wird.

  Mit anderen Worten, jedes Konto mit kontoübergreifendem Zugriff kann auf `DeleteObject`, `GetObject` usw. (aber nicht `PutObject`) zugreifen, und dies über HTTP oder HTTPS.

  Wenn Sie `PutObject` nicht von der zweiten Anweisung ausschließen, ist die Anweisung nicht gültig (weil Sie HTTPS explizit als Bedingung vorgeben müssen, wenn Sie `PutObject` aufnehmen).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CrossAccountRolePostOverHttps",
            "Effect": "Allow",
            "Action": "mediastore:PutObject",
            "Principal": {
                "AWS": "arn:aws:iam::333333333333:role/<role name>"
            },
            "Resource": "arn:aws:mediastore:us-
east-2:333333333333:container/<container name>/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "true"
                }
            }
        },
        {
            "Sid": "CrossAccountFullAccessExceptPost",
            "Effect": "Allow",
            "NotAction": "mediastore:PutObject",
```

```
            "Principal": {
                "AWS": "arn:aws:iam::333333333333:role/<role name>"
            },
            "Resource": "arn:aws:mediastore:us-
east-2:333333333333:container/<container name>/*"
        }
    ]
}
```

Beispiel-Containerrichtlinie: Zugriff auf bestimmte IP-Adressen beschränkt

Diese Beispielrichtlinie ermöglicht den Zugriff auf alle AWS Elemental MediaStore Elemental-Operationen an Objekten im angegebenen Container. Die Anfrage muss jedoch aus dem in der Bedingung angegebenen IP-Adressbereich stammen.

Die Bedingung in dieser Anweisung identifiziert den Bereich 198.51.100.* der zulässigen IP-Adressen der Internetprotokollversion 4 (IPv4), mit einer Ausnahme: 198.51.100.188.

Der `Condition`-Block verwendet die Bedingungen `IpAddress` und `NotIpAddress` und den Bedingungsschlüssel `aws:SourceIp`, wobei es sich um einen AWS-übergreifenden Bedingungsschlüssel handelt. Die Werte verwenden die standardmäßige CIDR-Notation. `aws:sourceIp` IPv4 Weitere Informationen finden Sie unter IP-Adressbedingungsoperatoren im IAM-Benutzerhandbuch.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AccessBySpecificIPAddress",
            "Effect": "Allow",
            "Action": [
                "mediastore:GetObject",
                "mediastore:DescribeObject"
            ],
            "Principal": "*",
            "Resource": "arn:aws:mediastore:us-
east-2:333333333333:container/<container name>/*",
            "Condition": {
```

```
                "IpAddress": {
                    "aws:SourceIp": [
                        "198.51.100.0/24"
                    ]
                },
                "NotIpAddress": {
                    "aws:SourceIp": "198.51.100.188/32"
                }
            }
        }
    ]
}
```

CORS-Richtlinien (Cross-Origin Resource Sharing) in AWS Elemental MediaStore

Cross-Origin Resource Sharing (CORS) bestimmt für Client-Webanwendungen, die in einer Domain geladen sind, eine Möglichkeit zur Interaktion mit Ressourcen in einer anderen Domain. Mit der CORS-Unterstützung in AWS Elemental MediaStore können Sie umfangreiche clientseitige Webanwendungen mit Ihren Ressourcen erstellen MediaStore und selektiv den ursprungsübergreifenden Zugriff auf Ihre Ressourcen zulassen. MediaStore

> **Note**
>
> Wenn Sie Amazon verwenden CloudFront , um Inhalte aus einem Container mit einer CORS-Richtlinie zu verteilen, müssen Sie die Verteilung für AWS Elemental konfigurieren MediaStore (einschließlich des Schritts zur Bearbeitung des Cache-Verhaltens zur Einrichtung von CORS).

Dieser Abschnitt bietet eine Übersicht über CORS. In den Unterthemen wird beschrieben, wie Sie CORS mithilfe der AWS Elemental MediaStore Elemental-Konsole oder programmgesteuert mithilfe der MediaStore REST-API und der AWS aktivieren können. SDKs

Themen

- CORS-Anwendungsfälle
- Hinzufügen einer CORS-Richtlinie zu einem Container

CORS-Anwendungsfälle

Es folgen typische Beispielszenarien für den Einsatz von CORS:

- Szenario 1: Angenommen, Sie verteilen Live-Streaming-Video in einem AWS Elemental MediaStore Elemental-Container mit dem Namen LiveVideo. Ihre Benutzer laden den Video-Manifest-Endpunkt `http://livevideo.mediastore.ap-southeast-2.amazonaws.com` von einem bestimmten Ursprungsserver wie beispielsweise `www.example.com`. Sie möchten einen JavaScript Videoplayer verwenden, um über unauthentifizierte `GET` Anfragen auf Videos zuzugreifen, die aus diesem Container stammen. `PUT` Ein Browser würde diese Anfragen normalerweise blockieren JavaScript, aber Sie können eine CORS-Richtlinie für Ihren Container festlegen, um diese Anfragen von explizit zu aktivieren. `www.example.com`
- Szenario 2: Angenommen, Sie möchten denselben Livestream wie in Szenario 1 von Ihrem MediaStore Container aus hosten, möchten aber Anfragen von beliebiger Herkunft zulassen. Sie können eine CORS-Richtlinie so konfigurieren, dass Wildcard-(*)-Ursprünge erlaubt sind, sodass Anfragen von jedem beliebigen Ursprung auf das Video zugreifen können.

Hinzufügen einer CORS-Richtlinie zu einem Container

In diesem Abschnitt wird erklärt, wie Sie einem AWS Elemental MediaStore Elemental-Container eine CORS-Konfiguration (Cross-Origin Resource Sharing) hinzufügen. CORS erlaubt Client-Webanwendungen, die in einer Domäne geladen sind, eine Möglichkeit zur Interaktion mit Ressourcen in einer anderen Domäne.

Um Ihren Container so zu konfigurieren, dass er ursprungsübergreifende Anfragen zulässt, fügen Sie dem Container eine CORS-Richtlinie hinzu. Eine CORS-Richtlinie definiert Regeln, die die Ursprünge identifizieren, die den Zugriff auf Ihren Container zulassen, die Operationen (HTTP-Methoden), die für jeden Ursprung unterstützt werden, sowie weitere operationsspezifische Informationen.

Wenn Sie dem Container eine CORS-Richtlinie hinzufügen, gelten die Container-Richtlinien (die die Zugriffsrechte auf den Container regeln) weiterhin.

Hinzufügen einer CORS-Richtlinie (Konsole)

1. Öffnen Sie die Konsole unter MediaStore . https://console.aws.amazon.com/mediastore/
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, für den Sie eine CORS-Richtlinie erstellen möchten.

   Die Seite mit den Containerdetails wird angezeigt.
3. Wählen Sie im Abschnitt Container CORS policy (Container-CORS-Richtlinie) die Option Create CORS policy (CORS-Richtlinie erstellen).
4. Fügen Sie die Richtlinie im JSON-Format ein und wählen Sie Save (Speichern).

Hinzufügen einer CORS-Richtlinie (AWS CLI)

1. Erstellen Sie eine Datei, mit der die CORS-Richtlinie definiert wird:

```
[
  {
    "AllowedHeaders": [
      "*"
    ],
    "AllowedMethods": [
      "GET",
      "HEAD"
    ],
    "AllowedOrigins": [
      "*"
    ],
    "MaxAgeSeconds": 3000
  }
]
```

2. Verwenden Sie im AWS CLI den `put-cors-policy` Befehl.

```
aws mediastore put-cors-policy --container-name ExampleContainer --cors-policy
 file://corsPolicy.json --region us-west-2
```

   Dieser Befehl hat keinen Rückgabewert.

Anzeigen einer CORS-Richtlinie

Cross-Origin Resource Sharing (CORS) bestimmt für Client-Webanwendungen, die in einer Domain geladen sind, eine Möglichkeit zur Interaktion mit Ressourcen in einer anderen Domain.

Anzeige einer CORS-Richtlinie (Konsole)

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, für den Sie eine CORS-Richtlinie anzeigen möchten.

   Die Container-Detailseite wird angezeigt, mit der CORS-Richtlinie im Abschnitt Container CORS Policy (Container--CORS-Richtlinie).

Anzeige einer CORS-Richtlinie (AWS CLI)

- Verwenden Sie im AWS CLI den folgenden `get-cors-policy` Befehl:

  ```
  aws mediastore get-cors-policy --container-name ExampleContainer --region us-west-2
  ```

  Im folgenden Beispiel finden Sie den Rückgabewert:

  ```
  {
      "CorsPolicy": [
          {
              "AllowedMethods": [
                  "GET",
                  "HEAD"
              ],
              "MaxAgeSeconds": 3000,
              "AllowedOrigins": [
                  "*"
              ],
              "AllowedHeaders": [
                  "*"
              ]
          }
      ]
  }
  ```

Bearbeiten einer CORS-Richtlinie

Cross-Origin Resource Sharing (CORS) bestimmt für Client-Webanwendungen, die in einer Domain geladen sind, eine Möglichkeit zur Interaktion mit Ressourcen in einer anderen Domain.

Bearbeiten einer CORS-Richtlinie (Konsole)

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, für den Sie eine CORS-Richtlinie bearbeiten möchten.

   Die Seite mit den Containerdetails wird angezeigt.
3. Wählen Sie im Abschnitt Container CORS policy (Container-CORS-Richtlinie) die Option Edit CORS policy (CORS-Richtlinie bearbeiten).
4. Nehmen Sie Ihre Änderungen an der Richtlinie vor und wählen Sie Save (Speichern).

So bearbeiten Sie eine CORS-Richtlinie (AWS CLI)

1. Erstellen Sie eine Datei, mit der die aktualisierte CORS-Richtlinie definiert wird:

```
[
  {
    "AllowedHeaders": [
      "*"
    ],
    "AllowedMethods": [
      "GET",
      "HEAD"
    ],
    "AllowedOrigins": [
      "https://www.example.com"
    ],
    "MaxAgeSeconds": 3000
  }
]
```

2. Verwenden Sie im AWS CLI den `put-cors-policy` Befehl.

```
aws mediastore put-cors-policy --container-name ExampleContainer --cors-policy
 file://corsPolicy2.json --region us-west-2
```

Dieser Befehl hat keinen Rückgabewert.

Löschen einer CORS-Richtlinie

Cross-Origin Resource Sharing (CORS) bestimmt für Client-Webanwendungen, die in einer Domain geladen sind, eine Möglichkeit zur Interaktion mit Ressourcen in einer anderen Domain. Das Löschen der CORS-Richtlinie für einen Container entfernt Berechtigungen für ursprungsübergreifende Anfragen.

Löschen einer CORS-Richtlinie (Konsole)

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, für den Sie eine CORS-Richtlinie löschen möchten.

   Die Seite mit den Containerdetails wird angezeigt.
3. Wählen Sie im Abschnitt Container CORS policy (Container-CORS-Richtlinie) die Option Delete CORS policy (CORS-Richtlinie löschen).
4. Wählen Sie zur Bestätigung Weiter und anschließend Speichern) aus.

Löschen einer CORS-Richtlinie (AWS CLI)

- Verwenden Sie im AWS CLI den folgenden `delete-cors-policy` Befehl:

  ```
  aws mediastore delete-cors-policy --container-name ExampleContainer --region us-west-2
  ```

  Dieser Befehl hat keinen Rückgabewert.

Fehlerbehebung bei CORS-Problemen

Wenn Sie beim Zugriff auf einen Container mit einer CORS-Richtlinie auf unerwartetes Verhalten stoßen, gehen Sie wie folgt vor, um das Problem zu beheben.

1. Vergewissern Sie sich, dass die CORS-Richtlinie dem Container zugewiesen ist.

   Detaillierte Anweisungen finden Sie unter the section called “Anzeigen einer CORS-Richtlinie”.

2. Erfassen Sie die vollständige Anfrage und die Antwort mit einem Tool Ihrer Wahl (z. B. der Entwicklerkonsole Ihres Browsers). Vergewissern Sie sich, dass die CORS-Richtlinie, die dem Container zugewiesen ist, mindestens eine CORS-Regel enthält, die mit Daten in Ihrer Anfrage übereinstimmt, wie folgt:

   a. Stellen Sie sicher, dass die Anfrage einen `Origin`-Header besitzt.

      Wenn der Header fehlt, behandelt AWS Elemental die Anfrage MediaStore nicht als ursprungsübergreifende Anfrage und sendet keine CORS-Antwortheader in der Antwort zurück.

   b. Stellen Sie sicher, dass der `Origin`-Header in Ihrer Anfrage mit mindestens einem der `AllowedOrigins`-Elemente in der betreffenden `CORSRule` übereinstimmt.

      Das Schema, der Host und die Port-Werte im `Origin`-Anfrageheader müssen mit den `AllowedOrigins` in der `CORSRule` übereinstimmen. Wenn Sie beispielsweise die `CORSRule` so eingerichtet haben, dass der Ursprung `http://www.example.com` zulässig ist, stimmen die Ursprünge `https://www.example.com` und `http://www.example.com:80` in Ihrer Anfrage nicht mit dem in Ihrer Konfiguration erlaubten Ursprung überein.

   c. Stellen Sie sicher, dass die Methode in Ihrer Anfrage (oder die in `Access-Control-Request-Method` spezifizierte Methode, falls es sich um eine Preflight-Anfrage handelt) eines der `AllowedMethods`-Elemente in derselben `CORSRule` ist.

   d. Wenn bei einer Preflight-Anfrage die anfrage einen `Access-Control-Request-Headers`-Header enthält, überprüfen Sie, ob die `CORSRule` die `AllowedHeaders`-Einträge für jeden Wert im `Access-Control-Request-Headers`-Header enthält.

CORS-Beispielrichtlinien

Die folgenden Beispiele zeigen Cross-Origin Resource Sharing (CORS)-Richtlinien.

Themen

CORS-Beispielrichtlinie: Lesezugriff für jede Domäne

Die folgende Richtlinie ermöglicht es einer Webseite aus einer beliebigen Domain, Inhalte aus Ihrem AWS Elemental MediaStore Container abzurufen. Die Anfrage enthält alle HTTP-Header der Ursprungsdomäne, und der Service reagiert nur auf HTTP-GET- und HTTP-HEAD-Anfragen aus der Ursprungsdomäne. Die Ergebnisse werden für 3.000 Sekunden zwischengespeichert, bevor eine neue Ergebnismenge bereitgestellt wird.

```
[
  {
    "AllowedHeaders": [
      "*"
    ],
    "AllowedMethods": [
      "GET",
      "HEAD"
    ],
    "AllowedOrigins": [
      "*"
    ],
    "MaxAgeSeconds": 3000
  }
]
```

CORS-Beispielrichtlinie: Lesezugriff für eine bestimmte Domäne

Die folgende Richtlinie ermöglicht es einer Webseite`https://www.example.com`, Inhalte aus Ihrem AWS Elemental MediaStore Container abzurufen. Die Anfrage enthält alle HTTP-Header aus `https://www.example.com`, und der Service reagiert nur auf HTTP-GET- und HTTP-HEAD-Anfragen aus `https://www.example.com`. Die Ergebnisse werden für 3.000 Sekunden zwischengespeichert, bevor eine neue Ergebnismenge bereitgestellt wird.

```
[
  {
    "AllowedHeaders": [
      "*"
    ],
    "AllowedMethods": [
      "GET",
      "HEAD"
    ],
```

```
    "AllowedOrigins": [
      "https://www.example.com"
    ],
    "MaxAgeSeconds": 3000
  }
]
```

Richtlinien für den Objektlebenszyklus in AWS Elemental MediaStore

Sie können für jeden Container eine Objektlebenszyklus-Richtlinie erstellen, die regelt, wie lange Objekte im Container gespeichert werden sollen. Wenn Objekte das von Ihnen angegebene Höchstalter erreichen, MediaStore löscht AWS Elemental die Objekte. Sie können Objekte löschen, nachdem sie nicht mehr benötigt werden, um Speicherkosten sparen.

Sie können auch angeben, dass Objekte nach Erreichen eines bestimmten Alters in die Speicherklasse für seltenen Zugriff (IA) verschoben werden MediaStore sollen. Objekte, die in der Speicherklasse für seltenen Zugriff gespeichert sind, weisen andere Speicher- und Abrufraten auf als Objekte, die in der Standardspeicherklasse gespeichert sind. Weitere Informationen finden Sie unter MediaStore -Preisgestaltung.

Eine Objektlebenszyklus-Richtlinie enthält Regeln, die die Lebensdauer von Objekten anhand von Unterordnern bestimmen. (Sie können eine Objektlebenszyklus-Richtlinie keinen einzelnen Objekten zuweisen). Sie können nur eine Objektlebenszyklus-Richtlinie an einen Container anhängen, aber Sie können bis zu 10 Regeln zu jeder Objektlebenszyklus-Richtlinie hinzufügen. Weitere Informationen finden Sie unter Komponenten einer Objektlebenszyklus-Richtlinie.

Themen

Komponenten einer Objektlebenszyklus-Richtlinie

Objektlebenszyklusrichtlinien regeln, wie lange Objekte in einem AWS Elemental MediaStore Container verbleiben. Jede Objektlebenszyklus-Richtlinie besteht aus mindestens einer Regel, die die Lebensdauer von Objekten bestimmt. Eine Regel kann für einen Ordner, mehrere Ordner oder den gesamten Container gelten.

Sie können eine Objektlebenszyklus-Richtlinie an einen Container anhängen und jede Objektlebenszyklus-Richtlinie kann bis zu 10 Regeln enthalten. Sie können eine Objektlebenszyklus-Richtlinie keinem einzelnen Objekt zuweisen.

Regeln in einer Objektlebenszyklus-Richtlinie

Sie können drei Arten von Regeln erstellen:

- Vorübergehende Daten
- Objekt löschen
- Lebenszyklusübergang

Vorübergehende Daten

Eine Regel für transiente Daten legt fest, dass Objekte innerhalb von Sekunden ablaufen. Dieser Regeltyp gilt nur für Objekte, die dem Container hinzugefügt werden, nachdem die Richtlinie wirksam wird. Es dauert bis zu 20 Minuten MediaStore , bis die neue Richtlinie auf den Container angewendet wird.

Hier sehen Sie ein Beispiel für eine Regel für transiente Daten:

```
        {
            "definition": {
                "path": [ {"wildcard": "Football/index*.m3u8"} ],
                "seconds_since_create": [
                    {"numeric": [">", 120]}
                ]
            },
            "action": "EXPIRE"
        },
```

Regeln für transiente Daten bestehen aus drei Teilen:

- `path`: Immer auf `wildcard` gesetzt. Mit diesem Teil definieren Sie, welche Objekte gelöscht werden sollen. Sie können einen oder mehrere Platzhalter verwenden, dargestellt durch ein Sternchen (*). Jeder Platzhalter steht für eine beliebige Kombination aus null oder mehr Zeichen. Beispielsweise gilt `"path": [ {"wildcard": "Football/index*.m3u8"} ],` für alle Dateien im Ordner `Football`, die dem Muster von `index*.m3u8` entsprechen (z. B. „index.m3u8“, „index1.m3us8“ und „index123456.m3u8“). Sie können bis zu 10 -Pfade in eine Regel aufnehmen.
- `seconds_since_create`: Immer auf `numeric` gesetzt. Sie können einen Wert von 1 bis 300 angeben. Sie können den Operator auch auf „größer als“ (**>**) oder „größer oder gleich“ (**>=**) festlegen.
- `action`: Immer auf `EXPIRE` gesetzt.

Bei Regeln für transiente Daten (Objekte laufen innerhalb von Sekunden ab) gibt es keine Verzögerung zwischen dem Ablauf eines Objekts und dem Löschen des Objekts.

> **Note**
>
> Objekte, die einer Regel für transiente Daten unterliegen, sind nicht in einer `list-items`-Antwort enthalten. Außerdem lösen Objekte, die aufgrund einer Regel für transiente Daten ablaufen, kein CloudWatch Ereignis aus, wenn sie ablaufen.

Objekt löschen

Eine Regel zum Löschen von Objekten legt fest, dass Objekte innerhalb von Tagen ablaufen. Dieser Regeltyp gilt für alle Objekte im Container, auch wenn sie dem Container hinzugefügt wurden, bevor die Richtlinie erstellt wurde. Es dauert bis zu 20 Minuten, MediaStore bis die neue Richtlinie angewendet wird, aber es kann bis zu 24 Stunden dauern, bis die Objekte aus dem Container gelöscht sind.

Ein Beispiel für zwei Regeln zum Löschen von Objekten sieht wie folgt aus:

```
        {
            "definition": {
                "path": [ { "prefix": "FolderName/" }  ],
                "days_since_create": [
                    {"numeric": [">" , 5]}
                ]
```

```
            },
            "action": "EXPIRE"
        },
        {
            "definition": {
                "path": [ { "wildcard": "Football/*.ts" }  ],
                "days_since_create": [
                    {"numeric": [">" , 5]}
                ]
            },
            "action": "EXPIRE"
        }
```

Regeln zum Löschen von Objekten bestehen aus drei Teilen:

- `path`: Festlegung entweder auf `prefix` oder `wildcard`. `prefix` und `wildcard` können in derselben Regel nicht zusammen verwendet werden. Wenn Sie beide verwenden möchten, müssen Sie, wie im obigen Beispiel gezeigt, eine Regel für `prefix` und eine separate Regel für `wildcard`erstellen.
  - `prefix` - Legen Sie den Pfad auf `prefix` fest, wenn Sie alle Objekte innerhalb eines bestimmten Ordners löschen möchten. Wenn der Parameter leer ist (`"path": [ { "prefix": "" } ],`), umfasst das Ziel alle Objekte, die an einer beliebigen Stelle innerhalb des aktuellen Containers gespeichert sind. Sie können bis zu 10 `prefix`-Pfade in eine Regel aufnehmen.
  - `wildcard` - Legen Sie den Pfad auf `wildcard` fest, wenn Sie bestimmte Objekte basierend auf Dateinamen und/oder Dateityp löschen möchten. Sie können einen oder mehrere Platzhalter verwenden, dargestellt durch ein Sternchen (*). Jeder Platzhalter steht für eine beliebige Kombination aus null oder mehr Zeichen. `"path": [ {"wildcard": "Football/*.ts"} ],` gilt z. B. für alle Dateien im `Football`-Ordner, die dem Muster von `*.ts` entsprechen (z. B. Dateiname.ts, Dateiname1.ts und Dateiname123456.ts). Sie können bis zu 10 `wildcard`-Pfade in eine Regel aufnehmen.
- `days_since_create`: Immer auf `numeric` gesetzt. Sie können einen Wert von 1 bis 36.500 Tage angeben. Sie können den Operator auch auf „größer als“ (**>**) oder „größer oder gleich“ (**>=**) festlegen.
- `action`: Immer auf `EXPIRE` gesetzt.

Bei Regeln zum Löschen von Objekten (Objekte laufen innerhalb von Tagen ab) gibt es u. U. eine geringfügige Verzögerung zwischen dem Ablauf eines Objekts und dem Löschen des Objekts.

Änderungen bei der Fakturierung erfolgen jedoch, sobald das Objekt abläuft. Wenn beispielsweise eine Lebenszyklusregel 10 angibt `days_since_create`, wird dem Konto das Objekt nicht in Rechnung gestellt, nachdem das Objekt 10 Tage alt ist, auch wenn das Objekt noch nicht gelöscht wurde.

Lebenszyklusübergang

Eine Lebenszyklus-Übergangsregel legt fest, dass Objekte in die Speicherklasse für den seltenen Zugriff verschoben werden, nachdem sie ein bestimmtes Alter (gemessen in Tagen) erreicht haben. Objekte, die in der Speicherklasse für seltenen Zugriff gespeichert sind, weisen andere Speicher- und Abrufraten auf als Objekte, die in der Standardspeicherklasse gespeichert sind. Weitere Informationen finden Sie unter MediaStore-Preisgestaltung.

Sobald ein Objekt in die Speicherklasse für seltenen Zugriff verschoben wurde, können Sie es nicht zurück in die Standardspeicherklasse verschieben.

Die Lebenszyklusübergangsregel gilt für alle Objekte im Container, selbst wenn sie dem Container hinzugefügt wurden, bevor die Richtlinie erstellt wurde. Es dauert bis zu 20 Minuten, MediaStore bis die neue Richtlinie angewendet wird, aber es kann bis zu 24 Stunden dauern, bis die Objekte aus dem Container gelöscht sind.

Ein Beispiel für eine Lebenszyklus-Übergangsregel sieht folgendermaßen aus:

```
{
    "definition": {
        "path": [
            {"prefix": "AwardsShow/"}
        ],
        "days_since_create": [
            {"numeric": [">=" , 30]}
        ]
    },
    "action": "ARCHIVE"
}
```

Lebenszyklus-Übergangsregeln haben drei Teile:

- `path`: Festlegung entweder auf `prefix` oder `wildcard`. `prefix` und `wildcard` können in derselben Regel nicht zusammen verwendet werden. Wenn Sie beide verwenden möchten, müssen Sie eine Regel für `prefix` und eine separate Regel für `wildcard` erstellen.

- `prefix`: Sie legen den Pfad auf `prefix` fest, wenn Sie alle Objekte in einem bestimmten Ordner in die Speicherklasse für seltenen Zugriff übertragen möchten. Wenn der Parameter leer ist (`"path": [ { "prefix": "" } ],`), umfasst das Ziel alle Objekte, die an einer beliebigen Stelle innerhalb des aktuellen Containers gespeichert sind. Sie können bis zu 10 `prefix`-Pfade in eine Regel aufnehmen.
- `wildcard`: Sie legen den Pfad auf `wildcard` fest, wenn Sie bestimmte Objekte basierend auf dem Dateinamen und/oder Dateityp in die Speicherklasse für seltenen Zugriff übertragen möchten. Sie können einen oder mehrere Platzhalter verwenden, dargestellt durch ein Sternchen (*). Jeder Platzhalter steht für eine beliebige Kombination aus null oder mehr Zeichen. `"path": [ {"wildcard": "Football/*.ts"} ],` gilt z. B. für alle Dateien im `Football`-Ordner, die dem Muster von `*.ts` entsprechen (z. B. Dateiname.ts, Dateiname1.ts und Dateiname123456.ts). Sie können bis zu 10 `wildcard`-Pfade in eine Regel aufnehmen.

- `days_since_create`: Immer auf `"numeric": [">=" , 30]` gesetzt.
- `action`: Immer auf `ARCHIVE` gesetzt.

Beispiel

Angenommen, ein Container mit dem Namen `LiveEvents` verfügt über vier Unterordner: `Football`, `Baseball`, `Basketball` und `AwardsShow`. Die dem `LiveEvents`-Ordner zugewiesene Objektlebenszyklus-Richtlinie kann wie folgt aussehen:

```
{
    "rules": [
         {
            "definition": {
                "path": [
                    {"prefix": "Football/"},
                    {"prefix": "Baseball/"}
                ],
                "days_since_create": [
                    {"numeric": [">" , 28]}
                ]
            },
            "action": "EXPIRE"
        },
        {
            "definition": {
                "path": [ { "prefix": "AwardsShow/" }  ],
                "days_since_create": [
```

```
                {"numeric": [">=" , 15]}
            ]
        },
        "action": "EXPIRE"
    },
    {
        "definition": {
            "path": [ { "prefix": "" }  ],
            "days_since_create": [
                {"numeric": [">" , 40]}
            ]
        },
        "action": "EXPIRE"
    },
    {
        "definition": {
            "path": [ { "wildcard": "Football/*.ts" }  ],
            "days_since_create": [
                {"numeric": [">" , 20]}
            ]
        },
        "action": "EXPIRE"
    },
    {
        "definition": {
            "path": [
                {"wildcard": "Football/index*.m3u8"}
            ],
            "seconds_since_create": [
                {"numeric": [">" , 15]}
            ]
        },
        "action": "EXPIRE"
    },
    {
            "definition": {
                "path": [
                    {"prefix": "Program/"}
                ],
                "days_since_create": [
                    {"numeric": [">=" , 30]}
                ]
            },
            "action": "ARCHIVE"
```

```
            }
    ]
}
```

Die obige Richtlinie legt Folgendes fest:

- Die erste Regel weist AWS Elemental MediaStore an, Objekte, die in dem `LiveEvents/Football` Ordner gespeichert sind, und den `LiveEvents/Baseball` Ordner zu löschen, nachdem sie älter als 28 Tage sind.
- Die zweite Regel weist den Service an, Objekte zu löschen, die im Ordner `LiveEvents/AwardsShow` gespeichert sind, nachdem sie mindestens 15 Tage alt sind.
- Die dritte Regel weist den Service an, Objekte zu löschen, die an einem beliebigen Speicherort im Container `LiveEvents` gespeichert sind, nachdem sie älter als 40 Tage sind. Diese Regel gilt für Objekte, die direkt im `LiveEvents`-Container gespeichert sind, sowie für gespeicherte Objekte in jedem der vier Unterordner des Containers.
- Die vierte Regel weist den Service an, Objekte im `Football`-Ordner zu löschen, die dem Muster `*.ts`-entsprechen, nachdem sie älter als 20 Tage sind.
- Die fünfte Regel weist den Service an, Objekte in dem `Football` Ordner, die `index*.m3u8` dem Muster entsprechen, zu löschen, wenn sie älter als 15 Sekunden sind. MediaStore löscht diese Dateien 16 Sekunden, nachdem sie im Container platziert wurden.
- Die sechste Regel weist den Service an, Objekte im `Program`-Ordner in die Speicherklasse für seltenen Zugriff zu verschieben, nachdem sie 30 Tage alt sind.

Weitere Beispiele für Objektlebenszyklus-Richtlinien finden Sie unter Beispiele für Objektlebenszyklus-Richtlinien.

Hinzufügen einer Objektlebenszyklus-Richtlinie zu einem Container

Mit einer Objektlebenszyklus-Richtlinie können Sie angeben, wie lange Ihre Objekte in einem Container gespeichert werden sollen. Sie legen ein Ablaufdatum fest, und nach dem Ablaufdatum MediaStore löscht AWS Elemental die Objekte. Es dauert bis zu 20 Minuten, bis der Service die neue Richtlinie auf den Container anwendet.

Weitere Informationen zum Erstellen einer Lebenszyklusrichtlinie finden Sie unter Komponenten einer Objektlebenszyklus-Richtlinie.

> **Note**
>
> Bei Regeln zum Löschen von Objekten (Objekte laufen innerhalb von Tagen ab) gibt es u. U. eine geringfügige Verzögerung zwischen dem Ablauf eines Objekts und dem Löschen des Objekts. Änderungen bei der Fakturierung erfolgen jedoch, sobald das Objekt abläuft. Wenn beispielsweise eine Lebenszyklusregel 10 angibt `days_since_create`, wird dem Konto das Objekt nicht in Rechnung gestellt, nachdem das Objekt 10 Tage alt ist, auch wenn das Objekt noch nicht gelöscht wurde.

So fügen Sie eine Objektlebenszyklus-Richtlinie hinzu (Konsole)

1. Öffnen Sie die MediaStore Konsole unter. https://console.aws.amazon.com/mediastore/
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, für den Sie eine Objektlebenszyklus-Richtlinie erstellen möchten.

   Die Seite mit den Containerdetails wird angezeigt.
3. Wählen Sie im Abschnitt Objektlebenszyklus-Richtlinie die Option zum Erstellen einer Objektlebenszyklus-Richtlinie aus.
4. Fügen Sie die Richtlinie im JSON-Format ein und wählen Sie Save (Speichern).

So fügen Sie eine Objektlebenszyklus-Richtlinie hinzu (AWS CLI)

1. Erstellen Sie eine Datei, die die Objektlebenszyklus-Richtlinie definiert:

```
{
    "rules": [
         {
            "definition": {
                "path": [
                    {"prefix": "Football/"},
                    {"prefix": "Baseball/"}
                ],
                "days_since_create": [
                    {"numeric": [">" , 28]}
                ]
            },
            "action": "EXPIRE"
        },
```

```
        {
            "definition": {
                "path": [
                    {"wildcard": "AwardsShow/index*.m3u8"}
                ],
                "seconds_since_create": [
                    {"numeric": [">" , 8]}
                ]
            },
            "action": "EXPIRE"
        }
    ]
}
```

2. Verwenden Sie im AWS CLI den folgenden `put-lifecycle-policy` Befehl:

```
aws mediastore put-lifecycle-policy --container-name LiveEvents --lifecycle-
policy file://LiveEventsLifecyclePolicy.json --region us-west-2
```

Dieser Befehl hat keinen Rückgabewert. Der Service fügt die angegebene Richtlinie an den Container an.

Anzeigen einer Objektlebenszyklus-Richtlinie

Eine Objektlebenszyklus-Richtlinie gibt an, wie lange Objekte in einem Container aufbewahrt werden sollen.

So zeigen Sie eine Objektlebenszyklus-Richtlinie an (Konsole)

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, für den Sie die Objektlebenszyklus-Richtlinie anzeigen möchten.

   Die Container-Detailseite wird geöffnet und die Objektlebenszyklus-Richtlinie wird im Abschnitt Objektlebenszyklus-Richtlinie angezeigt.

So zeigen Sie eine Objektlebenszyklus-Richtlinie an (AWS CLI)

- Verwenden Sie im AWS CLI den folgenden `get-lifecycle-policy` Befehl:

```
aws mediastore get-lifecycle-policy --container-name LiveEvents --region us-west-2
```

Im folgenden Beispiel finden Sie den Rückgabewert:

```
{
    "LifecyclePolicy": "{
        "rules": [
            {
                "definition": {
                    "path": [
                        {"prefix": "Football/"},
                        {"prefix": "Baseball/"}
                    ],
                    "days_since_create": [
                        {"numeric": [">" , 28]}
                    ]
                },
             "action": "EXPIRE"
            }
        ]
    }"
}
```

Bearbeiten einer Objektlebenszyklus-Richtlinie

Sie können keine vorhandene Objektlebenszyklus-Richtlinie bearbeiten. Sie können jedoch eine vorhandene Richtlinie ändern, indem Sie eine Ersatz-Richtlinie hochladen. Es dauert bis zu 20 Minuten, bis der Service die aktualisierte Richtlinie auf den Container anwendet.

So bearbeiten Sie eine Objektlebenszyklus-Richtlinie (Konsole)

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, für den Sie die Objektlebenszyklus-Richtlinie bearbeiten möchten.

   Die Seite mit den Containerdetails wird angezeigt.
3. Wählen Sie im Abschnitt Objektlebenszyklus-Richtlinie die Option zum Bearbeiten einer Objektlebenszyklus-Richtlinie aus.

4. Nehmen Sie Ihre Änderungen an der Richtlinie vor und wählen Sie Save (Speichern).

So bearbeiten Sie eine Objektlebenszyklus-Richtlinie (AWS CLI)

1. Erstellen Sie eine Datei, die die aktualisierte Objektlebenszyklus-Richtlinie definiert:

```
{
    "rules": [
         {
            "definition": {
                "path": [
                    {"prefix": "Football/"},
                    {"prefix": "Baseball/"}
                    {"prefix": "Basketball/"}
                ],
                "days_since_create": [
                    {"numeric": [">" , 28]}
                ]
            },
            "action": "EXPIRE"
        }
    ]
}
```

2. Verwenden Sie im AWS CLI den folgenden `put-lifecycle-policy` Befehl:

```
aws mediastore put-lifecycle-policy --container-name LiveEvents --lifecycle-
policy file://LiveEvents2LifecyclePolicy --region us-west-2
```

   Dieser Befehl hat keinen Rückgabewert. Der Service fügt die angegebene Richtlinie an den Container an und ersetzt damit die vorherige Richtlinie.

Löschen einer Objektlebenszyklus-Richtlinie

Wenn Sie eine Objekt-Lebenszyklus-Richtlinie löschen, dauert es bis zu 20 Minuten, bis der Service die Änderung auf den Container angewendet hat.

So löschen Sie eine Objektlebenszyklus-Richtlinie (Konsole)

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.

2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, für den Sie die Objektlebenszyklus-Richtlinie löschen möchten.

   Die Seite mit den Containerdetails wird angezeigt.
3. Wählen Sie im Abschnitt Objektlebenszyklus-Richtlinie die Option zum Löschen einer Lebenszyklusrichtlinie aus.
4. Wählen Sie zur Bestätigung Weiter und anschließend Speichern) aus.

So löschen Sie eine Objektlebenszyklus-Richtlinie (AWS CLI)

- Verwenden Sie im AWS CLI den folgenden `delete-lifecycle-policy` Befehl:

  ```
  aws mediastore delete-lifecycle-policy --container-name LiveEvents --region us-west-2
  ```

  Dieser Befehl hat keinen Rückgabewert.

Beispiele für Objektlebenszyklus-Richtlinien

Die folgenden Beispiele zeigen Objekt-Lebenszyklusrichtlinien.

Themen

Beispiel für Objektlebenszyklus-Richtlinie: Ablauf innerhalb von Sekunden

Die folgende Richtlinie legt fest, dass Objekte MediaStore gelöscht werden, die allen folgenden Kriterien entsprechen:

- Das Objekt wird dem Container hinzugefügt, nachdem die Richtlinie wirksam wurde.
- Das Objekt wird im `Football`-Ordner gespeichert.
- Das Objekt hat die Dateierweiterung `m3u8`.

- Das Objekt befindet sich seit mehr als 20 Sekunden im Container.

```
{
    "rules": [
        {
            "definition": {
                "path": [
                    {"wildcard": "Football/*.m3u8"}
                ],
                "seconds_since_create": [
                    {"numeric": [ ">", 20 ]}
                ]
            },
            "action": "EXPIRE"
        }
    ]
}
```

Beispiel für Objektlebenszyklus-Richtlinie: Ablauf innerhalb von Tagen

Die folgende Richtlinie legt fest, dass Objekte MediaStore gelöscht werden, die allen folgenden Kriterien entsprechen:

- Das Objekt wird im `Program`-Ordner gespeichert.
- Das Objekt hat die Dateierweiterung `ts`.
- Das Objekt befindet sich seit mehr als 5 Tagen im Container.

```
{
    "rules": [
        {
            "definition": {
                "path": [
                    {"wildcard": "Program/*.ts"}
                ],
                "days_since_create": [
                    {"numeric": [ ">", 5 ]}
                ]
            },
            "action": "EXPIRE"
        }
```

```
    ]
}
```

Beispiel für Objektlebenszyklus-Richtlinie: Übertragung in Speicherklasse mit seltenem Zugriff

Die folgende Richtlinie legt fest, dass Objekte in die Speicherklasse für seltenen Zugriff (IA) MediaStore verschoben werden, wenn sie 30 Tage alt sind. Objekte, die in der Speicherklasse für seltenen Zugriff gespeichert sind, weisen andere Speicher- und Abrufraten auf als Objekte, die in der Standardspeicherklasse gespeichert sind.

Das `days_since_create`-Feld muss auf `"numeric": [">=" ,30]` eingestellt sein.

```
{
    "rules": [
        {
            "definition": {
                "path": [
                    {"prefix": "Football/"},
                    {"prefix": "Baseball/"}
                ],
                "days_since_create": [
                    {"numeric": [">=" , 30]}
                ]
            },
            "action": "ARCHIVE"
        }
    ]
}
```

Beispiel für Objektlebenszyklus-Richtlinie: mehrere Regeln

Die folgende Richtlinie legt fest, dass MediaStore dies Folgendes bewirkt:

- Objekte, die im `AwardsShow`-Ordner gespeichert sind, nach 30 Tagen in die Speicherklasse für den seltenen Zugriff verschieben
- Objekte mit der Dateierweiterung `m3u8` nach 20 Sekunden im `Football`-Ordner löschen
- Objekte nach 10 Tagen im `April`-Ordner löschen
- Objekte mit der Dateierweiterung `ts` nach 5 Tagen im `Program`-Ordner löschen

```
{
    "rules": [
        {
            "definition": {
                "path": [
                    {"prefix": "AwardsShow/"}
                ],
                "days_since_create": [
                    {"numeric": [ ">=" , 30 ]}
                ]
            },
            "action": "ARCHIVE"
        },
        {
            "definition": {
                "path": [
                    {"wildcard": "Football/*.m3u8"}
                ],
                "seconds_since_create": [
                    {"numeric": [ ">", 20 ]}
                ]
            },
            "action": "EXPIRE"
        },
        {
            "definition": {
                "path": [
                    {"prefix": "April"}
                ],
                "days_since_create": [
                    {"numeric": [ ">", 10 ]}
                ]
            },
            "action": "EXPIRE"
        },
        {
            "definition": {
                "path": [
                    {"wildcard": "Program/*.ts"}
                ],
                "days_since_create": [
                    {"numeric": [ ">", 5 ]}
                ]
```

```
            },
            "action": "EXPIRE"
        }
    ]
}
```

Beispiel für Objektlebenszyklus-Richtlinie: Container leeren

Die folgende Objektlebenszyklus-Richtlinie legt fest, dass alle Objekte im Container, einschließlich Ordner und Unterordner, einen Tag nach dem Hinzufügen zum Container MediaStore gelöscht werden. Wenn der Container Objekte enthält, bevor diese Richtlinie angewendet wird, werden die Objekte 1 Tag nach Inkrafttreten der Richtlinie MediaStore gelöscht. Es dauert bis zu 20 Minuten, bis der Service die neue Richtlinie auf den Container anwendet.

```
{
    "rules": [
        {
            "definition": {
                "path": [
                    {"wildcard": "*"}
                ],
                "days_since_create": [
                    {"numeric": [ ">=", 1 ]}
                ]
            },
            "action": "EXPIRE"
        }
    ]
}
```

Metrikrichtlinien in AWS Elemental MediaStore

Für jeden Container können Sie eine Metrikrichtlinie hinzufügen, damit AWS Elemental Metriken MediaStore an Amazon CloudWatch senden kann. Es dauert bis zu 20 Minuten, bis die neue Richtlinie wirksam wird. Eine Beschreibung der einzelnen MediaStore Metriken finden Sie unterMediaStore Metriken.

Eine Metrikrichtlinie enthält Folgendes:

- Eine Einstellung zum Aktivieren oder Deaktivieren von Metriken auf Containerebene.

- Zwischen null und fünf Regeln, die Metriken auf Objektebene aktivieren. Wenn die Richtlinie Regeln enthält, muss jede Regel Folgendes umfassen:
  - Eine Objektgruppe, die definiert, welche Objekte in die Gruppe aufgenommen werden sollen. Die Definition kann ein Pfad oder ein Dateiname sein, darf jedoch nicht mehr als 900 Zeichen enthalten. Gültige Zeichen sind: a–z, A–Z, 0–9, _ (Unterstrich), = (gleich), : (Doppelpunkt), . (Punkt), - (Bindestrich), ~ (Tilde), / (Schrägstrich) und * (Sternchen). Platzhalter (*) sind zulässig.
  - Ein Objektgruppenname, mit dem Sie auf die Objektgruppe verweisen können. Der Name darf nicht mehr als 30 Zeichen enthalten. Gültige Zeichen sind a–z, A–Z, 0–9 und _ (Unterstrich).

Wenn ein Objekt mehreren Regeln entspricht, CloudWatch wird für jede übereinstimmende Regel ein Datenpunkt angezeigt. Wenn ein Objekt beispielsweise zwei Regeln mit dem Namen `rule1` und entsprichtrule2, CloudWatch werden zwei Datenpunkte für diese Regeln angezeigt. Die erste hat die Dimension `ObjectGroupName=rule1` und die zweite die Dimension `ObjectGroupName=rule2`.

Themen

Hinzufügen einer Metrikrichtlinie

Eine Metrikrichtlinie enthält Regeln, die festlegen, welche Metriken AWS Elemental MediaStore an Amazon sendet. CloudWatch Beispiele für Metrikrichtlinien finden Sie unter Beispiele für Metrikrichtlinien.

So fügen Sie eine Metrikrichtlinie hinzu (Konsole)

1. Öffnen Sie die MediaStore Konsole unter. https://console.aws.amazon.com/mediastore/
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, dem Sie eine Metrikrichtlinie hinzufügen möchten.

   Die Seite mit den Containerdetails wird angezeigt.
3. Wählen Sie im Abschnitt Metric policy (Metrikrichtlinie) die Option Create metric policy (Metrikrichtlinie erstellen) aus.

4. Fügen Sie die Richtlinie im JSON-Format ein und wählen Sie Save (Speichern).

Anzeigen einer Metrikrichtlinie

Sie können die Konsole oder die verwenden AWS CLI , um die Metrikrichtlinie eines Containers anzuzeigen.

So zeigen Sie eine Metrikrichtlinie an (Konsole)

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.
2. Wählen Sie auf der Seite Containers (Container) den Containernamen aus.

   Die Seite mit den Containerdetails wird angezeigt. Die Richtlinie wird im Abschnitt Metric policy (Metrikrichtlinie) angezeigt.

Bearbeiten einer Metrikrichtlinie

Eine Metrikrichtlinie enthält Regeln, die festlegen, welche Metriken AWS Elemental MediaStore an Amazon sendet. CloudWatch Wenn Sie eine vorhandene Metrikrichtlinie bearbeiten, dauert es bis zu 20 Minuten, bis die neue Richtlinie wirksam wird. Beispiele für Metrikrichtlinien finden Sie unter Beispiele für Metrikrichtlinien.

So bearbeiten Sie eine Metrikrichtlinie (Konsole)

1. Öffnen Sie die MediaStore Konsole unter. https://console.aws.amazon.com/mediastore/
2. Wählen Sie auf der Seite Containers (Container) den Containernamen aus.
3. Wählen Sie im Abschnitt Metric policy (Metrikrichtlinie) die Option Edit metric policy (Metrikrichtlinie bearbeiten) aus.
4. Nehmen Sie die entsprechenden Änderungen vor und wählen Sie dann Save (Speichern) aus.

Beispiele für Metrikrichtlinien

Die folgenden Beispiele veranschaulichen Metrikrichtlinien, die für verschiedene Anwendungsfälle erstellt wurden.

Themen

- Beispiel für Metrikrichtlinien: Metriken auf Containerebene

- Beispiel für Metrikrichtlinien: Metriken auf Pfadebene
- Beispiel für Metrikrichtlinien: Metriken auf Container- und Pfadebene
- Beispiel für Metrikrichtlinien: Metriken auf Pfadebene mit Platzhaltern
- Beispiel für Metrikrichtlinien: Metriken auf Pfadebene mit sich überschneidenden Regeln

Beispiel für Metrikrichtlinien: Metriken auf Containerebene

Diese Beispielrichtlinie gibt an, dass AWS Elemental Metriken auf Containerebene CloudWatch an Amazon senden MediaStore soll. Dies schließt beispielsweise die Metrik `RequestCount` ein, die die Anzahl der `Put`-Anforderungen an den Container zählt. Alternativ können Sie diese Richtlinie auf `DISABLED` einstellen.

Da diese Richtlinie keine Regeln MediaStore enthält, werden keine Metriken auf Pfadebene gesendet. Beispielsweise können Sie nicht sehen, wie viele `Put`-Anforderungen an einen bestimmten Ordner innerhalb dieses Containers gestellt wurden.

```
{
   "ContainerLevelMetrics": "ENABLED"
}
```

Beispiel für Metrikrichtlinien: Metriken auf Pfadebene

Diese Beispielrichtlinie gibt an, dass AWS Elemental keine Metriken auf Containerebene CloudWatch an Amazon senden MediaStore sollte. Darüber hinaus soll MediaStore Metriken für Objekte in zwei bestimmten Ordnern senden: `baseball/saturday` und `football/saturday`. Die Metriken für MediaStore -Anforderungen lauten wie folgt:

- Anfragen an den `baseball/saturday` Ordner haben eine CloudWatch Dimension von. `ObjectGroupName=baseballGroup`
- Anforderungen an den Ordner `football/saturday` haben die Dimension `ObjectGroupName=footballGroup`.

```
{
   "ContainerLevelMetrics": "DISABLED",
   "MetricPolicyRules": [
     {
       "ObjectGroup": "baseball/saturday",
```

```
        "ObjectGroupName": "baseballGroup"
      },
      {
        "ObjectGroup": "football/saturday",
        "ObjectGroupName": "footballGroup"
      }
    ]
}
```

Beispiel für Metrikrichtlinien: Metriken auf Container- und Pfadebene

Diese Beispielrichtlinie gibt an, dass AWS Elemental Metriken auf Containerebene CloudWatch an Amazon senden MediaStore soll. Darüber hinaus MediaStore sollte Metriken für Objekte in zwei bestimmten Ordnern gesendet werden: `baseball/saturday` und`football/saturday`. Die Metriken für MediaStore-Anforderungen lauten wie folgt:

- Anfragen an den `baseball/saturday` Ordner haben eine CloudWatch Dimension von`ObjectGroupName=baseballGroup`.
- Anfragen an den `football/saturday` Ordner haben eine CloudWatch Dimension`ObjectGroupName=footballGroup`.

```
{
   "ContainerLevelMetrics": "ENABLED",
   "MetricPolicyRules": [
     {
       "ObjectGroup": "baseball/saturday",
       "ObjectGroupName": "baseballGroup"
     },
     {
       "ObjectGroup": "football/saturday",
       "ObjectGroupName": "footballGroup"
     }
   ]
}
```

Beispiel für Metrikrichtlinien: Metriken auf Pfadebene mit Platzhaltern

Diese Beispielrichtlinie gibt an, dass AWS Elemental Metriken auf Containerebene CloudWatch an Amazon senden MediaStore soll. Darüber hinaus MediaStore sollte auch Metriken für Objekte auf der Grundlage ihres Dateinamens gesendet werden. Ein Platzhalter gibt an, dass die Objekte an einer

beliebigen Stelle im Container gespeichert werden und einen beliebigen Dateinamen haben können, solange dieser mit der Erweiterung `.m3u8` endet.

```
{
   "ContainerLevelMetrics": "ENABLED",
   "MetricPolicyRules": [
     {
       "ObjectGroup": "*.m3u8",
       "ObjectGroupName": "index"
     }
   ]
 }
```

Beispiel für Metrikrichtlinien: Metriken auf Pfadebene mit sich überschneidenden Regeln

Diese Beispielrichtlinie gibt an, dass AWS Elemental Metriken auf Containerebene CloudWatch an Amazon senden MediaStore soll. Darüber hinaus MediaStore sollte Metriken für zwei Ordner gesendet werden: `sports/football/saturday` und`sports/football`.

Die Metriken für MediaStore Anfragen an den `sports/football/saturday` Ordner haben eine CloudWatch Dimension von`ObjectGroupName=footballGroup1`. Da Objekte, die im Ordner `sports/football` gespeichert sind, beiden Regeln entsprechen, zeigt CloudWatch zwei Datenpunkte für diese Objekte an: einen mit der Dimension `ObjectGroupName=footballGroup1` und den zweiten mit der Dimension `ObjectGroupName=footballGroup2`.

```
{
   "ContainerLevelMetrics": "ENABLED",
   "MetricPolicyRules": [
     {
       "ObjectGroup": "sports/football/saturday",
       "ObjectGroupName": "footballGroup1"
     },
     {
       "ObjectGroup": "sports/football",
       "ObjectGroupName": "footballGroup2"
     }
   ]
 }
```

Ordner in AWS Elemental MediaStore

Ordner sind Teilbereiche in einem Container. Sie Verwenden Ordner, um Ihren Container so zu unterteilen, wie Sie Unterordner erstellen, um einen Ordner in einem Dateisystem zu unterteilen. Sie können bis zu 10 Ordnerebenen erstellen (ohne den Container selbst).

Ordner sind optional. Sie können auswählen, ob Sie die Objekte direkt in einen Container statt in einen Ordner hochladen wollen. Ordner bieten jedoch eine einfache Möglichkeit, Ihre Objekte zu organisieren.

Um ein Objekt in einen Ordner hochzuladen, geben Sie den Pfad zum Ordner an. Wenn der Ordner bereits existiert, MediaStore speichert AWS Elemental das Objekt in dem Ordner. Wenn der Ordner nicht vorhanden, legt der Service ihn an und speichert das Objekt in dem Ordner.

Nehmen wir zum Beispiel an, Sie haben einen Container mit dem Namen und laden eine Datei hochmovies, die `mlaw.ts` mit dem Pfad `premium/canada` benannt ist. AWS Elemental MediaStore speichert das Objekt im Unterordner Canada unter dem Ordner Premium. Wenn keine der Ordner vorhanden ist, erstellt der Service sowohl den Ordner `premium`, als auch den Unterordner `canada`, und speichert das Objekt im Unterordner `canada`. Wenn Sie nur den Container `movies` (ohne Pfad) angeben, speichert der Service das Objekt direkt im Container.

AWS Elemental löscht einen Ordner MediaStore automatisch, wenn Sie das letzte Objekt in diesem Ordner löschen. Der Service löscht auch alle leeren Ordner oberhalb dieses Ordners. Beispielsweise angenommen, Sie verfügen über einen Ordner mit dem Namen premium, der keine Dateien enthält, jedoch einen Unterordner mit dem Namen `canada`. Der Unterordner `canada` enthält eine mit dem Namen .`mlaw.ts` Wenn Sie die Datei `mlaw.ts` löschen, löscht der Service die Ordner `premium` und `canada`. Dieses automatische Löschen gilt nur für Ordner. Der Service löscht keine leeren Container.

Themen

Regeln für Ordnernamen

Wenn Sie einen Namen für Ihren Ordner wählen, beachten Sie Folgendes:

- Der Name darf nur die folgenden Zeichen enthalten: Großbuchstaben (A-Z), Kleinbuchstaben (a-z), Zahlen (0-9), Punkte (.), Bindestriche (-), Tilden (~), Unterstriche (_), Gleichheitszeichen (=) und Doppelpunkte (:).
- Der Name muss mindestens ein Zeichen lang sein. Leere Ordnernamen (z. B.`folder1//folder3/`) sind nicht zulässig.
- Bei den Namen muss die Groß- und Kleinschreibung beachtet werden. Sie können beispielsweise einen Ordner mit dem Namen `myFolder` und einen Ordner mit dem Namen `myfolder` im selben Container oder Ordner verwenden, weil diese Namen eindeutig sind.
- Der Name muss eindeutig innerhalb seines übergeordneten Containers oder Ordners sein. Sie können beispielsweise einen Ordner mit dem Namen `myfolder` in zwei verschiedenen Containern erstellen: `movies/myfolder` und `sports/myfolder`.
- Der Name kann denselben Namen wie der übergeordnete Container haben.
- Der Ordner kann nach dem Erstellen nicht mehr umbenannt werden.

Erstellen eines Ordners

Sie können Ordner erstellen, wenn Sie Objekte hochladen. Um ein Objekt in einen Ordner hochzuladen, geben Sie den Pfad zum Ordner an. Wenn der Ordner bereits existiert, MediaStore speichert AWS Elemental das Objekt in dem Ordner. Wenn der Ordner nicht vorhanden, legt der Service ihn an und speichert das Objekt in dem Ordner.

Weitere Informationen finden Sie unter the section called “Hochladen eines Objekts”.

Löschen eines Ordners

Sie können Ordner nur löschen, wenn der Ordner leer ist. Sie können keinen Ordner löschen, die Objekte enthalten.

AWS Elemental löscht einen Ordner MediaStore automatisch, wenn Sie das letzte Objekt in diesem Ordner löschen. Der Service löscht auch alle leeren Ordner oberhalb dieses Ordners. Beispielsweise angenommen, Sie verfügen über einen Ordner mit dem Namen `premium`, der keine Dateien enthält, jedoch einen Unterordner mit dem Namen `canada`. Der Unterordner `canada` enthält eine mit dem Namen .`mlaw.ts` Wenn Sie die Datei `mlaw.ts` löschen, löscht der Service die Ordner `premium` und `canada`. Dieses automatische Löschen gilt nur für Ordner. Der Service löscht keine leeren Container.

Weitere Informationen finden Sie unter Löschen eines Objekts.

Objekte in AWS Elemental MediaStore

AWS Elemental MediaStore Elemental-Assets werden Objekte genannt. Sie können ein Objekt in einen Container oder in einen Ordner innerhalb des Containers hochladen.

MediaStoreIn können Sie Objekte hochladen, herunterladen und löschen:

- Hochladen – Ein Objekt in einem Container oder Ordner hinzufügen. Dies ist nicht dasselbe wie das Erstellen eines Objekts. Sie müssen Ihre Objekte lokal erstellen, bevor Sie sie hochladen können MediaStore.
- Herunterladen — Kopiert ein Objekt von MediaStore einem anderen Ort. Dadurch wird das Objekt nicht entfernt MediaStore.
- Löschen – Ein Objekt aus MediaStore vollständig entfernen. Sie können Objekte einzeln löschen oder einer Objektlebenszyklus-Richtlinie hinzufügen, um Objekte in einem Container nach einer bestimmten Zeit automatisch zu löschen.

MediaStore akzeptiert alle Dateitypen.

Themen

Hochladen eines Objekts

Sie können Objekte in einen Container oder in einen Ordner innerhalb eines Containers hochladen. Um ein Objekt in einen Ordner hochzuladen, geben Sie den Pfad zum Ordner an. Wenn der Ordner bereits existiert, MediaStore speichert AWS Elemental das Objekt im Ordner. Wenn der Ordner nicht vorhanden, legt der Service ihn an und speichert das Objekt in dem Ordner. Weitere Informationen über Ordner finden Sie unter Ordner in AWS Elemental MediaStore.

Sie können die MediaStore Konsole oder die verwenden AWS CLI , um Objekte hochzuladen.

MediaStore unterstützt die stückweise Übertragung von Objekten, wodurch die Latenz reduziert wird, indem ein Objekt zum Herunterladen verfügbar gemacht wird, während es noch hochgeladen wird. Um diese Funktion zu verwenden, stellen Sie für die Upload-Verfügbarkeit des Objekts `streaming` ein. Sie können den Wert dieses Headers festlegen, wenn Sie das Objekt mithilfe der API hochladen. Wenn Sie diesen Header in Ihrer Anfrage nicht angeben, MediaStore weist er der Upload-Verfügbarkeit des Objekts `standard` den Standardwert von zu.

Objekte dürfen eine Größe von 25 MB für Standard-Upload-Verfügbarkeit und von 10 MB für Streaming-Upload-Verfügbarkeit nicht überschreiten.

> **Note**
>
> Objektdateinamen dürfen nur Buchstaben, Zahlen, Punkte (.), Unterstriche (_), Tilden (~), Bindestriche (-), Gleichheitszeichen (=) und Doppelpunkte (:) enthalten.

Hochladen eines Objekts (Konsole)

1. Öffnen Sie die Konsole unter. MediaStore https://console.aws.amazon.com/mediastore/
2. Wählen Sie auf der Seite Containers (Container) den Containernamen aus. Der Detailbereich für den Container wird angezeigt.
3. Wählen Sie Upload object (Objekt hochladen).
4. Geben Sie für Target path (Zielpfad) einen Pfad für die Ordner ein. Beispiel, `premium/canada`. Wenn einer der Ordner in dem von Ihnen angegebenen Pfad noch nicht vorhanden ist, legt der Service ihn automatisch an.
5. Wählen Sie im Bereich Object (Objekt) die Option Browse (Durchsuchen).
6. Navigieren Sie zum entsprechenden Ordner und wählen Sie ein Objekt zum Hochladen aus.
7. Wählen Sie Open (Öffnen) und anschließend Upload (Hochladen).

> **Note**
>
> Wenn im ausgewählten Ordner bereits eine Datei mit dem gleichen Namen vorhanden ist, ersetzt der Service die Originaldatei durch die hochgeladene Datei.

Ein Objekt hochladen (AWS CLI)

- Verwenden Sie im AWS CLI den `put-object` Befehl. Sie können auch einen der folgenden Parameter einschließen: `content-type`, `cache-control` (um dem Aufrufer zu erlauben, das Cache-Verhalten des Objekts zu steuern) und `path` (um das Objekt in einen Ordner innerhalb des Containers zu legen).

  > **Note**
  >
  > Nachdem Sie das Objekt hochgeladen haben, können Sie `content-type`, `cache-control` oder `path` nicht mehr bearbeiten.

```
aws mediastore-data put-object --endpoint https://
aaabbbcccddddee.data.mediastore.us-west-2.amazonaws.com --body README.md --path /
folder_name/README.md --cache-control "max-age=6, public" --content-type binary/
octet-stream --region us-west-2
```

Im folgenden Beispiel finden Sie den Rückgabewert:

```
{
    "ContentSHA256":
 "74b5fdb517f423ed750ef214c44adfe2be36e37d861eafe9c842cbe1bf387a9d",
    "StorageClass": "TEMPORAL",
    "ETag": "af3e4731af032167a106015d1f2fe934e68b32ed1aa297a9e325f5c64979277b"
}
```

Anzeigen einer Liste von Objekten

Sie können die AWS Elemental MediaStore Elemental-Konsole verwenden, um Elemente (Objekte und Ordner) anzuzeigen, die in der obersten Ebene eines Containers oder in einem Ordner gespeichert sind. Elemente in einem Unterordner des aktuellen Containers oder Ordners werden nicht angezeigt. Sie können die verwenden AWS CLI , um eine Liste von Objekten und Ordnern innerhalb eines Containers anzuzeigen, unabhängig davon, wie viele Ordner oder Unterordner sich innerhalb des Containers befinden.

Eine Liste der Objekte in einem bestimmten Container anzeigen (Konsole)

1. Öffnen Sie die MediaStore Konsole unter. https://console.aws.amazon.com/mediastore/
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, der den Ordner enthält, den Sie anzeigen möchten.
3. Wählen Sie den Namen des Ordners in der Liste aus.

   Eine Detailseite wird angezeigt. Sie enthält alle Ordner und Objekte, die im Ordner gespeichert sind.

Eine Liste der Objekte in einem bestimmten Ordner anzeigen (Konsole)

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, der den Ordner enthält, den Sie anzeigen möchten.

   Eine Detailseite wird angezeigt. Sie enthält alle Ordner und Objekte, die im Container gespeichert sind.

Eine Liste der Objekte und Ordner in einem bestimmten Container anzeigen (AWS CLI)

- Verwenden Sie im AWS CLI den folgenden `list-items` Befehl:

```
aws mediastore-data list-items --endpoint https://
aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com --region us-west-2
```

Im folgenden Beispiel finden Sie den Rückgabewert:

```
{
    "Items": [
        {
            "ContentType": "image/jpeg",
            "LastModified": 1563571859.379,
            "Name": "filename.jpg",
            "Type": "OBJECT",
            "ETag":
 "543ab21abcd1a234ab123456a1a2b12345ab12abc12a1234abc1a2bc12345a12",
            "ContentLength": 3784
        },
```

```
        {
            "Type": "FOLDER",
            "Name": "ExampleLiveDemo"
        }
    ]
}
```

> **Note**
>
> Objekte, die einer `list-items`-Regel unterliegen, sind nicht in einer `seconds_since_create`-Antwort enthalten.

Eine Liste der Objekte und Ordner in einem bestimmten Ordner anzeigen (AWS CLI)

- Verwenden Sie in der AWS CLI den `list-items` Befehl mit dem angegebenen Ordnernamen am Ende der Anfrage:

```
aws mediastore-data list-items --endpoint https://
aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com --path /folder_name --
region us-west-2
```

Im folgenden Beispiel finden Sie den Rückgabewert:

```
{
    "Items": [
        {
            "Type": "FOLDER",
            "Name": "folder_1"
        },
        {
            "LastModified": 1563571940.861,
            "ContentLength": 2307346,
            "Name": "file1234.jpg",
            "ETag":
 "111a1a22222a1a1a222abc333a444444b55ab1111ab2222222222ab333333a2b",
            "ContentType": "image/jpeg",
            "Type": "OBJECT"
        }
    ]
```

```
}
```

> **Note**
>
> Objekte, die einer `list-items`-Regel unterliegen, sind nicht in einer `seconds_since_create`-Antwort enthalten.

Anzeigen der Details eines Objekts

Nachdem Sie ein Objekt hochgeladen haben, MediaStore speichert AWS Elemental Details wie das Änderungsdatum, die Inhaltslänge ETag (Entitäts-Tag) und den Inhaltstyp. Informationen zur Verwendung der Metadaten eines Objekts finden Sie unter Interaktion von MediaStore mit HTTP-Caches.

Anzeigen der Details eines Objekts (Konsole)

1. Öffnen Sie die MediaStore Konsole unter. https://console.aws.amazon.com/mediastore/
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, der das Objekt enthält, das Sie anzeigen möchten.
3. Wenn sich das Objekt, das Sie anzeigen möchten, in einem Ordner befindet, wählen Sie so lange Ordnernamen aus, bis Sie das Objekt sehen.
4. Wählen Sie den Namen des Objekts.

   Eine Detailseite wird angezeigt. Sie enthält Informationen über das Objekt.

Anzeigen der Details eines Objekts (AWS CLI)

- Verwenden Sie im AWS CLI den folgenden `describe-object` Befehl:

  ```
  aws mediastore-data describe-object --endpoint https://
  aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com --path /folder_name/
  file1234.jpg --region us-west-2
  ```

  Im folgenden Beispiel finden Sie den Rückgabewert:

  ```
  {
      "ContentType": "image/jpeg",
  ```

```
    "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",
    "ContentLength": "2307346",
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3"
}
```

Herunterladen eines Objekts

Sie können die Konsole verwenden, um ein Objekt herunterzuladen. Sie können das verwenden AWS CLI , um ein Objekt oder nur einen Teil eines Objekts herunterzuladen.

Herunterladen eines Objekts (Konsole)

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, der das Objekt enthält, das Sie herunterladen möchten.
3. Wenn sich das Objekt, das Sie herunterladen möchten, in einem Ordner befindet, wählen Sie so lange Ordnernamen aus, bis Sie das Objekt sehen.
4. Wählen Sie den Namen des Objekts.
5. Wählen Sie auf der Detailseite Object (Objekt) die Option Download (Herunterladen).

Ein Objekt herunterladen (AWS CLI)

- Verwenden Sie im AWS CLI den folgenden `get-object` Befehl:

```
aws mediastore-data get-object --endpoint https://
aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com --path=/folder_name/
README.md README.md --region us-west-2
```

Im folgenden Beispiel finden Sie den Rückgabewert:

```
{
    "ContentLength": "2307346",
    "ContentType": "image/jpeg",
    "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3",
    "StatusCode": 200
}
```

Einen Teil eines Objekts herunterladen (AWS CLI)

- Verwenden Sie in der AWS CLI den `get-object` Befehl und geben Sie einen Bereich an.

```
aws mediastore-data get-object --endpoint https://
aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com --path /folder_name/
README.md --range="bytes=0-100" README2.md --region us-west-2
```

Im folgenden Beispiel finden Sie den Rückgabewert:

```
{
    "StatusCode": 206,
    "ContentRange": "bytes 0-100/2307346",
    "ContentLength": "101",
    "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",
    "ContentType": "image/jpeg",
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3"
}
```

Löschen von Objekten

AWS Elemental MediaStore bietet verschiedene Optionen zum Löschen von Objekten aus Containern:

- Löschen eines einzelnen Objekts. Es fallen keine Gebühren an.
- Leeren eines Containers, um alle Objekte innerhalb eines Containers gleichzeitig zu löschen. Da dieser Prozess API-Aufrufe verwendet, fallen normale API-Gebühren an.
- Hinzufügen einer Objektlebenszyklus-Richtlinie, um Objekte zu löschen, wenn sie ein bestimmtes Alter erreicht haben. Es fallen keine Gebühren an.

Löschen eines Objekts

Sie können die Objekte individuell löschen. Verwenden Sie hierzu die Konsole oder die AWS CLI. Alternativ können Sie eine Objektlebenszyklus-Richtlinie hinzufügen, um Objekte automatisch zu löschen, nachdem sie ein bestimmtes Alter in einem Container erreicht haben. Sie können auch einen Container leeren, um alle Objekte in diesem Container zu löschen.

> **Note**
>
> Wenn Sie das einzige Objekt in einem Ordner löschen, löscht AWS Elemental MediaStore automatisch den Ordner und alle leeren Ordner über diesem Ordner. Beispielsweise angenommen, Sie verfügen über einen Ordner mit dem Namen `premium`, der keine Dateien enthält, jedoch einen Unterordner mit dem Namen `canada`. Der Unterordner `canada` enthält eine mit dem Namen .`mlaw.ts` Wenn Sie die Datei `mlaw.ts` löschen, löscht der Service die Ordner `premium` und `canada`.

Löschen eines Objekts (Konsole)

1. Öffnen Sie die MediaStore Konsole unter. https://console.aws.amazon.com/mediastore/
2. Wählen Sie auf der Seite Containers (Container) den Namen des Containers, der das Objekt enthält, das Sie löschen möchten.
3. Wenn sich das Objekt, das Sie löschen möchten, in einem Ordner befindet, wählen Sie so lange Ordnernamen aus, bis Sie das Objekt sehen.
4. Wählen Sie die Option links neben dem Objektnamen.
5. Wählen Sie Löschen aus.

Ein Objekt löschen (AWS CLI)

- Verwenden Sie im AWS CLI den `delete-object` Befehl.

  Beispiel:

  ```
  aws mediastore-data --region us-west-2 delete-object --endpoint=https://
  aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com --path=/folder_name/
  README.md
  ```

  Dieser Befehl hat keinen Rückgabewert.

Leeren eines Containers

Sie können einen Container leeren, um alle Objekte zu löschen, die im Container gespeichert sind. Alternativ können Sie eine Objektlebenszyklus-Richtlinie hinzufügen, um Objekte automatisch zu

löschen, nachdem sie ein bestimmtes Alter in einem Container erreicht haben. Sie können auch Objekte einzeln löschen.

So leeren Sie einen Container (Konsole)

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.
2. Wählen Sie auf der Seite Containers (Container) die Option für den Container, den Sie leeren möchten.
3. Wählen Sie Empty container (Container leeren) aus. Es wird eine Bestätigungsmeldung angezeigt.
4. Bestätigen Sie, dass Sie den Container leeren möchten, indem Sie den Container-Namen in das Textfeld eingeben, und wählen Sie dann Leer.

Sicherheit in AWS Elemental MediaStore

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von Rechenzentren und Netzwerkarchitekturen, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame AWS Verantwortung von Ihnen und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud selbst und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, auf der AWS Dienste in der ausgeführt AWS Cloud werden. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Externe Prüfer testen und verifizieren regelmäßig die Wirksamkeit unserer Sicherheitsmaßnahmen im Rahmen der AWS . Weitere Informationen zu den Compliance-Programmen, die für AWS Elemental gelten MediaStore, finden Sie unter AWS Services in Scope by Compliance Program AWS .
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Service, den Sie nutzen. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Nutzung anwenden können MediaStore. In den folgenden Themen erfahren Sie, wie Sie die Konfiguration vornehmen MediaStore , um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere AWS Dienste nutzen können, die Sie bei der Überwachung und Sicherung Ihrer MediaStore Ressourcen unterstützen.

Topics

Datenschutz in AWS Elemental MediaStore

Das AWS Modell der gilt für den Datenschutz in AWS Elemental MediaStore. Wie in diesem Modell beschrieben, AWS ist verantwortlich für den Schutz der globalen Infrastruktur, auf der AWS Cloud alle Systeme laufen. Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Sie sind auch für die Sicherheitskonfiguration und die Verwaltungsaufgaben für die von Ihnen verwendeten AWS-Services verantwortlich. Weitere Informationen zum Datenschutz finden Sie unter Häufig gestellte Fragen zum Datenschutz. Informationen zum Datenschutz in Europa finden Sie im Blog-Beitrag AWS -Modell der geteilten Verantwortung und in der DSGVO im AWS -Sicherheitsblog.

Aus Datenschutzgründen empfehlen wir, dass Sie AWS-Konto Anmeldeinformationen schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einrichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem empfehlen wir, die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- Wird verwendet SSL/TLS , um mit AWS Ressourcen zu kommunizieren. Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein AWS CloudTrail. Informationen zur Verwendung von CloudTrail Pfaden zur Erfassung von AWS Aktivitäten finden Sie unter Arbeiten mit CloudTrail Pfaden im AWS CloudTrail Benutzerhandbuch.
- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen darin enthaltenen Standardsicherheitskontrollen AWS-Services.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu schützen.
- Wenn Sie für den Zugriff AWS über eine Befehlszeilenschnittstelle oder eine API FIPS 140-3-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-3.

Wir empfehlen dringend, in Freitextfeldern, z. B. im Feld Name, keine vertraulichen oder sensiblen Informationen wie die E-Mail-Adressen Ihrer Kunden einzugeben. Dies gilt auch, wenn Sie mit der Konsole, der MediaStore API oder auf andere AWS-Services Weise arbeiten oder diese verwenden. AWS CLI AWS SDKs Alle Daten, die Sie in Tags oder Freitextfelder eingeben, die für Namen verwendet werden, können für Abrechnungs- oder Diagnoseprotokolle verwendet

werden. Wenn Sie eine URL für einen externen Server bereitstellen, empfehlen wir dringend, keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL einzuschließen.

Datenverschlüsselung

MediaStore verschlüsselt Container und Objekte im Ruhezustand mithilfe des branchenüblichen AES-256-Algorithmus. Wir empfehlen Ihnen, Ihre Daten MediaStore auf folgende Weise zu sichern:

- Erstellen Sie eine Container-Richtlinie, um die Zugriffsrechte auf alle Ordner und Objekte in diesem Container zu kontrollieren. Weitere Informationen finden Sie unter the section called “Containerrichtlinien”.
- Erstellen Sie eine CORS-Richtlinie (Cross-Origin Resource Sharing), um den quellenübergreifenden Zugriff auf Ihre Ressourcen zu ermöglichen. MediaStore Mit CORS können Sie Client-Webanwendungen, die in einer Domain geladen sind, die Interaktion mit Ressourcen in einer anderen Domain erlauben. Weitere Informationen finden Sie unter the section called “CORS-Richtlinien”.

Identity and Access Management für AWS Elemental MediaStore

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf Ressourcen sicher zu AWS kontrollieren. IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um Ressourcen zu verwenden. MediaStore IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von Ihrer Rolle ab:

- Servicebenutzer – Fordern Sie von Ihrem Administrator Berechtigungen an, wenn Sie nicht auf Funktionen zugreifen können (siehe Fehlerbehebung bei AWS Elemental MediaStore Identity und Access).
- Serviceadministrator – Bestimmen Sie den Benutzerzugriff und stellen Sie Berechtigungsanfragen (siehe So MediaStore funktioniert AWS Elemental mit IAM).
- IAM-Administrator – Schreiben Sie Richtlinien zur Zugriffsverwaltung (siehe Beispiele für identitätsbasierte Richtlinien für AWS Elemental MediaStore).

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen sich als IAM-Benutzer authentifizieren oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich als föderierte Identität anmelden, indem Sie Anmeldeinformationen aus einer Identitätsquelle wie AWS IAM Identity Center (IAM Identity Center), Single Sign-On-Authentifizierung oder Anmeldeinformationen verwenden. Google/Facebook Weitere Informationen zum Anmelden finden Sie unter So melden Sie sich bei Ihrem AWS-Konto an im Benutzerhandbuch für AWS-Anmeldung .

AWS Bietet für den programmatischen Zugriff ein SDK und eine CLI zum kryptografischen Signieren von Anfragen. Weitere Informationen finden Sie unter AWS Signature Version 4 for API requests im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, dem sogenannten AWS-Konto Root-Benutzer, der vollständigen Zugriff auf alle AWS-Services Ressourcen hat. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Eine Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Tasks that require root user credentials im IAM-Benutzerhandbuch.

Verbundidentität

Als bewährte Methode sollten menschliche Benutzer für den Zugriff AWS-Services mithilfe temporärer Anmeldeinformationen einen Verbund mit einem Identitätsanbieter verwenden.

Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensverzeichnis, Ihrem Directory Service Web-Identitätsanbieter oder der AWS-Services mithilfe von Anmeldeinformationen aus einer Identitätsquelle zugreift. Verbundidentitäten übernehmen Rollen, die temporäre Anmeldeinformationen bereitstellen.

Für die zentrale Zugriffsverwaltung empfehlen wir AWS IAM Identity Center. Weitere Informationen finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center -Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität mit bestimmten Berechtigungen für eine einzelne Person oder Anwendung. Verwenden Sie möglichst temporäre Anmeldeinformationen statt IAM-Benutzer mit langfristigen Anmeldeinformationen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Erfordern, dass menschliche Benutzer den Verbund mit einem Identitätsanbieter verwenden müssen, um AWS mithilfe temporärer Anmeldeinformationen darauf zugreifen zu können.

Eine IAM-Gruppe gibt eine Sammlung von IAM-Benutzern an und vereinfacht die Verwaltung von Berechtigungen bei großer Benutzerzahl. Weitere Informationen finden Sie unter Use cases for IAM users im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität mit bestimmten Berechtigungen, die temporäre Anmeldeinformationen bereitstellt. Sie können eine Rolle übernehmen, indem Sie von einer Benutzer- zu einer IAM-Rolle (Konsole) wechseln AWS CLI oder einen AWS API-Vorgang aufrufen. Weitere Informationen finden Sie unter Methods to assume a role im IAM-Benutzerhandbuch.

IAM-Rollen sind nützlich für Verbundbenutzerzugriff, temporäre IAM-Benutzerberechtigungen, kontoübergreifenden Zugriff, dienstübergreifenden Zugriff und Anwendungen, die auf Amazon ausgeführt werden. EC2 Weitere Informationen finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an Identitäten oder Ressourcen anhängen. AWS Eine Richtlinie definiert Berechtigungen, wenn sie mit einer

Identität oder Ressource verknüpft sind. AWS bewertet diese Richtlinien, wenn ein Principal eine Anfrage stellt. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu JSON-Richtliniendokumenten finden Sie unter Overview of JSON policies im IAM-Benutzerhandbuch.

Mithilfe von Richtlinien legen Administratoren fest, wer auf was Zugriff hat, indem sie definieren, welcher Prinzipal Aktionen mit welchen Ressourcen und unter welchen Bedingungen ausführen kann.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator erstellt IAM-Richtlinien und fügt sie Rollen hinzu, die Benutzer dann übernehmen können. IAM-Richtlinien definieren Berechtigungen unabhängig von der Methode, die zur Ausführung der Operation verwendet wird.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität (Benutzer, Gruppe oder Rolle) anfügen können. Diese Richtlinien steuern, welche Aktionen Identitäten für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können Inline-Richtlinien (Richtlinien, die direkt in eine einzelne Identität eingebettet sind) oder verwaltete Richtlinien (eigenständige Richtlinien, die mehreren Identitäten zugeordnet sind) sein. Informationen dazu, wie Sie zwischen verwalteten und Inline-Richtlinien wählen, finden Sie unter Choose between managed policies and inline policies im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele hierfür sind Vertrauensrichtlinien für IAM-Rollen und Bucket-Richtlinien von Amazon S3. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben.

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Weitere Richtlinientypen

AWS unterstützt zusätzliche Richtlinientypen, mit denen die maximalen Berechtigungen festgelegt werden können, die durch gängigere Richtlinientypen gewährt werden:

- Berechtigungsgrenzen – Eine Berechtigungsgrenze legt die maximalen Berechtigungen fest, die eine identitätsbasierte Richtlinie einer IAM-Entität erteilen kann. Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im -IAM-Benutzerhandbuch.
- Richtlinien zur Dienstkontrolle (SCPs) — Geben Sie die maximalen Berechtigungen für eine Organisation oder Organisationseinheit in an AWS Organizations. Weitere Informationen finden Sie unter Service-Kontrollrichtlinien im AWS Organizations -Benutzerhandbuch.
- Richtlinien zur Ressourcenkontrolle (RCPs) — Legen Sie die maximal verfügbaren Berechtigungen für Ressourcen in Ihren Konten fest. Weitere Informationen finden Sie im AWS Organizations Benutzerhandbuch unter Richtlinien zur Ressourcenkontrolle (RCPs).
- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die als Parameter übergeben werden, wenn Sie eine temporäre Sitzung für eine Rolle oder einen Verbundbenutzer erstellen. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn für eine Anfrage mehrere Arten von Richtlinien gelten, sind die daraus resultierenden Berechtigungen schwieriger zu verstehen. Informationen darüber, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie unter Bewertungslogik für Richtlinien im IAM-Benutzerhandbuch.

So MediaStore funktioniert AWS Elemental mit IAM

Bevor Sie IAM zur Verwaltung des Zugriffs auf verwenden, sollten Sie sich darüber informieren MediaStore, mit welchen IAM-Funktionen Sie arbeiten können. MediaStore

IAM-Funktionen, die Sie mit AWS Elemental verwenden können MediaStore

IAM-Feature	MediaStore Unterstützung
Identitätsbasierte Richtlinien	Ja

IAM-Feature	MediaStore Unterstützung
Ressourcenbasierte Richtlinien	Ja
Richtlinienaktionen	Ja
Richtlinienressourcen	Ja
Richtlinienbedingungsschlüssel (services pezifisch)	Ja
ACLs	Nein
ABAC (Tags in Richtlinien)	Teilweise
Temporäre Anmeldeinformationen	Ja
Prinzipalberechtigungen	Ja
Servicerollen	Ja
Service-verknüpfte Rollen	Nein

Einen allgemeinen Überblick darüber, wie MediaStore und andere AWS Dienste mit den meisten IAM-Funktionen funktionieren, finden Sie im IAM-Benutzerhandbuch unter AWS Dienste, die mit IAM funktionieren.

Identitätsbasierte Richtlinien für MediaStore

Unterstützt Richtlinien auf Identitätsbasis: Ja

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien im IAM-Benutzerhandbuch.

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter

denen Aktionen zugelassen oder abgelehnt werden. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Beispiele für identitätsbasierte Richtlinien für MediaStore

Beispiele für MediaStore identitätsbasierte Richtlinien finden Sie unter. Beispiele für identitätsbasierte Richtlinien für AWS Elemental MediaStore

Ressourcenbasierte Richtlinien finden Sie in MediaStore

Unterstützt ressourcenbasierte Richtlinien: Ja

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Prinzipal in einer ressourcenbasierten Richtlinie angeben. Weitere Informationen finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

> **Note**
>
> MediaStore unterstützt auch Container-Richtlinien, die definieren, welche Prinzipalentitäten (Konten, Benutzer, Rollen und Verbundbenutzer) Aktionen für den Container ausführen können. Weitere Informationen finden Sie unter Containerrichtlinien.

Richtlinienaktionen für MediaStore

Unterstützt Richtlinienaktionen: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Nehmen Sie Aktionen in eine Richtlinie auf, um Berechtigungen zur Ausführung des zugehörigen Vorgangs zu erteilen.

Eine Liste der MediaStore Aktionen finden Sie unter Von AWS Elemental definierte Aktionen MediaStore in der Service Authorization Reference.

Bei Richtlinienaktionen wird vor der Aktion das folgende Präfix MediaStore verwendet:

```
mediastore
```

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Kommata:

```
"Action": [
      "mediastore:action1",
      "mediastore:action2"
         ]
```

Beispiele für MediaStore identitätsbasierte Richtlinien finden Sie unter. Beispiele für identitätsbasierte Richtlinien für AWS Elemental MediaStore

Politische Ressourcen für MediaStore

Unterstützt Richtlinienressourcen: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Eine Liste der MediaStore Ressourcentypen und ihrer ARNs Eigenschaften finden Sie unter Von AWS Elemental definierte Ressourcen MediaStore in der Service Authorization Reference. Informationen darüber, mit welchen Aktionen Sie den ARN jeder Ressource angeben können, finden Sie unter Von AWS Elemental MediaStore definierte Aktionen.

Die MediaStore Container-Ressource hat den folgenden ARN:

```
arn:${Partition}:mediastore:${Region}:${Account}:container/${containerName}
```

Weitere Informationen zum Format von ARNs finden Sie unter Amazon Resource Names (ARNs) und AWS Service Namespaces.

Um beispielsweise den Container `AwardsShow` in Ihrer Anweisung anzugeben, verwenden Sie den folgenden ARN:

```
"Resource": "arn:aws:mediastore:us-east-1:111122223333:container/AwardsShow"
```

Richtlinien-Bedingungsschlüssel für MediaStore

Unterstützt servicespezifische Richtlinienbedingungsschlüssel: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das `Condition`-Element legt fest, ob Anweisungen auf der Grundlage definierter Kriterien ausgeführt werden. Sie können bedingte Ausdrücke erstellen, die Bedingungsoperatoren verwenden, z. B. ist gleich oder kleiner als, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter Kontextschlüssel für AWS globale Bedingungen im IAM-Benutzerhandbuch.

Eine Liste der MediaStore Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für AWS Elemental MediaStore in der Service Authorization Reference. Informationen zu den Aktionen und Ressourcen, mit denen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter Von AWS Elemental MediaStore definierte Aktionen.

Beispiele für MediaStore identitätsbasierte Richtlinien finden Sie unter. Beispiele für identitätsbasierte Richtlinien für AWS Elemental MediaStore

ACLs in MediaStore

Unterstützt ACLs: Nein

Zugriffskontrolllisten (ACLs) steuern, welche Principals (Kontomitglieder, Benutzer oder Rollen) über Zugriffsberechtigungen für eine Ressource verfügen. ACLs ähneln ressourcenbasierten Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

ABAC mit MediaStore

Unterstützt ABAC (Tags in Richtlinien): Teilweise

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen, auch als Tags bezeichnet, definiert werden. Sie können Tags an IAM-Entitäten und AWS -Ressourcen anhängen und dann ABAC-Richtlinien entwerfen, die Operationen zulassen, wenn das Tag des Prinzipals mit dem Tag auf der Ressource übereinstimmt.

Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, oder Bedingung `aws:TagKeys` verwenden.

Wenn ein Service alle drei Bedingungsschlüssel für jeden Ressourcentyp unterstützt, lautet der Wert für den Service Ja. Wenn ein Service alle drei Bedingungsschlüssel für nur einige Ressourcentypen unterstützt, lautet der Wert Teilweise.

Weitere Informationen zu ABAC finden Sie unter Definieren von Berechtigungen mit ABAC-Autorisierung im IAM-Benutzerhandbuch. Um ein Tutorial mit Schritten zur Einstellung von ABAC anzuzeigen, siehe Attributbasierte Zugriffskontrolle (ABAC) verwenden im IAM-Benutzerhandbuch.

Verwenden temporärer Anmeldeinformationen mit MediaStore

Unterstützt temporäre Anmeldeinformationen: Ja

Temporäre Anmeldeinformationen ermöglichen kurzfristigen Zugriff auf AWS Ressourcen und werden automatisch erstellt, wenn Sie einen Verbund verwenden oder die Rollen wechseln. AWS empfiehlt, temporäre Anmeldeinformationen dynamisch zu generieren, anstatt langfristige Zugriffsschlüssel zu

verwenden. Weitere Informationen finden Sie unter Temporary security credentials in IAM und AWS-Services that work with IAM im IAM-Benutzerhandbuch.

Serviceübergreifende Prinzipalberechtigungen für MediaStore

Unterstützt Forward Access Sessions (FAS): Ja

Forward Access Sessions (FAS) verwenden die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anforderung, Anfragen an nachgelagerte Dienste AWS-Service zu stellen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.

Servicerollen für MediaStore

Unterstützt Servicerollen: Ja

Eine Servicerolle ist eine IAM-Rolle, die ein Service annimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

> **Warning**
>
> Durch das Ändern der Berechtigungen für eine Servicerolle kann die MediaStore Funktionalität beeinträchtigt werden. Bearbeiten Sie Servicerollen nur, MediaStore wenn Sie dazu eine Anleitung erhalten.

Dienstbezogene Rollen für MediaStore

Unterstützt serviceverknüpfte Rollen: Ja

Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Dienstbezogene Rollen werden in Ihrem Dienst angezeigt AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

Details zum Erstellen oder Verwalten von serviceverknüpften Rollen finden Sie unter AWS -Services, die mit IAM funktionieren. Suchen Sie in der Tabelle nach einem Service mit einem `Yes` in der Spalte Service-linked role (Serviceverknüpfte Rolle). Wählen Sie den Link Yes (Ja) aus, um die Dokumentation für die serviceverknüpfte Rolle für diesen Service anzuzeigen.

Beispiele für identitätsbasierte Richtlinien für AWS Elemental MediaStore

Benutzer und Rollen haben standardmäßig nicht die Berechtigung, MediaStore-Ressourcen zu erstellen oder zu ändern. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von IAM-Richtlinien (Konsole) im IAM-Benutzerhandbuch.

Einzelheiten zu Aktionen und Ressourcentypen, die von definiert wurden MediaStore, einschließlich des Formats von ARNs für jeden der Ressourcentypen, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Elemental MediaStore in der Service Authorization Reference.

Themen

Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand MediaStore Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder sie löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Beachten Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Richtlinien und Empfehlungen:

- Erste Schritte mit AWS verwalteten Richtlinien und Umstellung auf Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien oder Von AWS verwaltete Richtlinien für Auftragsfunktionen im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte

Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.

- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. CloudFormation B. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung mit IAM Access Analyzer im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Sicherer API-Zugriff mit MFA im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Verwenden der MediaStore-Konsole

Um auf die AWS Elemental MediaStore Console zugreifen zu können, benötigen Sie ein Minimum an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, die MediaStore Ressourcen in Ihrem AWS-Konto aufzulisten und Details zu diesen anzuzeigen. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (Benutzer oder Rollen) mit dieser Richtlinie.

Sie müssen Benutzern, die nur die API AWS CLI oder die AWS API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die der API-Operation entsprechen, die die Benutzer ausführen möchten.

Um sicherzustellen, dass Benutzer und Rollen die MediaStore Konsole weiterhin verwenden können, fügen Sie den Entitäten auch die MediaStore *ConsoleAccess* oder die *ReadOnly* AWS verwaltete Richtlinie hinzu. Weitere Informationen finden Sie unter Hinzufügen von Berechtigungen zu einem Benutzer im IAM-Benutzerhandbuch.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der AWS CLI AWS OR-API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
```

```
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Fehlerbehebung bei AWS Elemental MediaStore Identity und Access

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit MediaStore und IAM auftreten können.

Themen

- Ich bin nicht berechtigt, eine Aktion durchzuführen in MediaStore
- Ich bin nicht berechtigt, iam auszuführen: PassRole
- Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine MediaStore Ressourcen ermöglichen

Ich bin nicht berechtigt, eine Aktion durchzuführen in MediaStore

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht zur Durchführung einer Aktion berechtigt sind, müssen Ihre Richtlinien aktualisiert werden, damit Sie die Aktion durchführen können.

Der folgende Beispielfehler tritt auf, wenn der IAM-Benutzer `mateojackson` versucht, über die Konsole Details zu einer fiktiven *`my-example-widget`*-Ressource anzuzeigen, jedoch nicht über `mediastore:`*`GetWidget`*-Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 mediastore:GetWidget on resource: my-example-widget
```

In diesem Fall muss die Richtlinie für den Benutzer `mateojackson` aktualisiert werden, damit er mit der `mediastore:`*`GetWidget`*-Aktion auf die *`my-example-widget`*-Ressource zugreifen kann.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich bin nicht berechtigt, iam auszuführen: PassRole

Wenn Sie die Fehlermeldung erhalten, dass Sie nicht zum Durchführen der `iam:PassRole`-Aktion autorisiert sind, müssen Ihre Richtlinien aktualisiert werden, um eine Rolle an MediaStore übergeben zu können.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Dienst zu übergeben, anstatt eine neue Servicerolle oder eine dienstverknüpfte Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in MediaStore auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine MediaStore Ressourcen ermöglichen

Sie können eine Rolle erstellen, mit der Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation auf Ihre Ressourcen zugreifen können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Für Dienste, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (ACLs) unterstützen, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob diese Funktionen MediaStore unterstützt werden, finden Sie unter. So MediaStore funktioniert AWS Elemental mit IAM
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.

- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Anmeldung und Überwachung AWS Elemental MediaStore

Dieser Abschnitt bietet eine Übersicht über die Optionen zur Protokollierung und Überwachung in AWS Elemental MediaStore zu Sicherheitszwecken. Weitere Informationen zur Protokollierung und Überwachung MediaStore finden Sie unter Überwachung und Tagging in AWS Elemental MediaStore.

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit AWS Elemental MediaStore und Leistung Ihrer AWS Lösungen. Sie sollten Überwachungsdaten aus allen Teilen Ihrer AWS Lösung sammeln, damit Sie einen etwaigen Ausfall an mehreren Stellen leichter debuggen können. AWS bietet mehrere Tools zur Überwachung Ihrer MediaStore Ressourcen und zur Reaktion auf potenzielle Vorfälle.

CloudWatch Amazon-Alarme

Mithilfe von CloudWatch Alarmen beobachten Sie eine einzelne Metrik über einen von Ihnen festgelegten Zeitraum. Wenn die Metrik einen bestimmten Schwellenwert überschreitet, wird eine Benachrichtigung an ein Amazon SNS SNS-Thema oder eine AWS Auto Scaling Scaling-Richtlinie gesendet. CloudWatch Alarme lösen keine Aktionen aus, da sie sich in einem bestimmten Status befinden. Der Status muss sich stattdessen geändert haben und für eine festgelegte Anzahl an Zeiträumen aufrechterhalten worden sein. Weitere Informationen finden Sie unter Überwachung mit CloudWatch.

AWS CloudTrail Logs

CloudTrail bietet eine Aufzeichnung der Aktionen, die von einem Benutzer, einer Rolle oder einem AWS Dienst in ausgeführt wurden AWS Elemental MediaStore. Anhand der von gesammelten Informationen können Sie die Anfrage CloudTrail, an die die Anfrage gestellt wurde MediaStore, die

IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde und weitere Details ermitteln. Weitere Informationen finden Sie unter Protokollierung von API-Aufrufen mit CloudTrail.

AWS Trusted Advisor

Trusted Advisor stützt sich auf bewährte Verfahren, die wir bei der Betreuung von Hunderttausenden von AWS Kunden gelernt haben. Trusted Advisor untersucht Ihre AWS-Umgebung und gibt dann Empfehlungen, wenn Möglichkeiten bestehen, Geld zu sparen, die Systemverfügbarkeit und -leistung zu verbessern oder Sicherheitslücken zu schließen. Alle AWS Kunden haben Zugriff auf fünf Trusted Advisor Advisor-Checks. Kunden mit einem Business- oder Enterprise-Supportplan können alle Trusted Advisor Schecks einsehen.

Weitere Informationen finden Sie unter AWS Trusted Advisor.

Konformitätsvalidierung für AWS Elemental MediaStore

Informationen darüber, ob AWS-Service ein AWS-Services in den Geltungsbereich bestimmter Compliance-Programme fällt, finden Sie unter Umfang nach Compliance-Programm AWS-Services unter . Wählen Sie dort das Compliance-Programm aus, an dem Sie interessiert sind. Allgemeine Informationen finden Sie unter AWS Compliance-Programme AWS .

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte herunterladen unter .

Ihre Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. Weitere Informationen zu Ihrer Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services finden Sie in der AWS Sicherheitsdokumentation.

Resilienz in AWS Elemental MediaStore

Die AWS globale Infrastruktur basiert AWS-Regionen auf Availability Zones. AWS-Regionen bieten mehrere physisch getrennte und isolierte Availability Zones, die über Netzwerke mit niedriger Latenz, hohem Durchsatz und hoher Redundanz miteinander verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Zonen ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind

besser verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Weitere Informationen zu Availability Zones AWS-Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Zusätzlich zur AWS globalen Infrastruktur MediaStore bietet es mehrere Funktionen, die Sie bei Ihren Anforderungen an Datenstabilität und Datensicherung unterstützen.

Infrastruktursicherheit in AWS Elemental MediaStore

Als verwalteter Service MediaStore ist AWS Elemental durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsservices und zum AWS Schutz der Infrastruktur finden Sie unter AWS Cloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe für den Zugriff MediaStore über das Netzwerk. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Serviceübergreifende Confused-Deputy-Prävention

Das Confused-Deputy-Problem ist ein Sicherheitsproblem, bei dem eine juristische Stelle, die nicht über die Berechtigung zum Ausführen einer Aktion verfügt, eine privilegiertere juristische Stelle zwingen kann, die Aktion auszuführen. In AWS kann ein dienstübergreifendes Identitätswechsels zu dem Problem mit dem verwirrten Stellvertreter führen. Ein dienstübergreifender Identitätswechsel kann auftreten, wenn ein Dienst (der Anruf-Dienst) einen anderen Dienst anruft (den aufgerufenen Dienst). Der aufrufende Service kann manipuliert werden, um seine Berechtigungen zu verwenden, um Aktionen auf die Ressourcen eines anderen Kunden auszuführen, für die er sonst keine Zugriffsberechtigung haben sollte. Um dies zu verhindern, bietet AWS Tools, mit denen Sie Ihre Daten für alle Services mit Serviceprinzipalen schützen können, die Zugriff auf Ressourcen in Ihrem Konto erhalten haben.

Wir empfehlen, die Kontextschlüssel `aws:SourceArn` und die `aws:SourceAccount` globalen Bedingungsschlüssel in Ressourcenrichtlinien zu verwenden, um die Berechtigungen einzuschränken, die AWS Elemental einem anderen Service für die Ressource MediaStore erteilt. 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 Bedingungskontext-Schlüssels `aws:SourceArn` mit dem vollständigen ARN der Ressource. Wenn Sie den vollständigen ARN der Ressource nicht kennen oder wenn Sie mehrere Ressourcen angeben, verwenden Sie den globalen Kontextbedingungsschlüssel `aws:SourceArn` mit Platzhalterzeichen (*) für die unbekannten Teile des ARN. 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.

Der Wert von `aws:SourceArn` muss die Konfiguration sein, für die CloudWatch Protokolle in Ihrer Region und Ihrem Konto MediaStore veröffentlicht werden.

Das folgende Beispiel zeigt, wie Sie die Kontextschlüssel `aws:SourceArn` und die `aws:SourceAccount` globale Bedingung verwenden können, MediaStore um das Problem des verwirrten Stellvertreters zu vermeiden.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ConfusedDeputyPreventionExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "Service": "mediastore.amazonaws.com"
      },
      "Action": "mediastore:CreateContainer",
      "Resource": [
        "arn:aws:mediastore:us-east-2:333333333333:container/ResourceName/*"
```

```
      ],
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:mediastore:*:333333333333:*"
        },
        "StringEquals": {
          "aws:SourceAccount": "333333333333"
        }
      }
    }
  ]
}
```

Überwachung und Tagging in AWS Elemental MediaStore

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von AWS Elemental MediaStore und Ihren anderen AWS Lösungen. AWS bietet die folgenden Überwachungstools, mit denen Sie beobachten MediaStore, melden können, wenn etwas nicht stimmt, und gegebenenfalls automatische Maßnahmen ergreifen können:

- AWS CloudTrailerfasst API-Aufrufe und zugehörige Ereignisse, die von oder im Namen Ihres AWS Kontos getätigt wurden, und übermittelt die Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket. Sie können feststellen, welche Benutzer und Konten angerufen wurden AWS, von welcher Quell-IP-Adresse aus die Anrufe getätigt wurden und wann die Aufrufe erfolgten. Weitere Informationen finden Sie im AWS CloudTrail -Benutzerhandbuch.
- Amazon CloudWatch überwacht Ihre AWS Ressourcen und die Anwendungen, auf denen Sie laufen, AWS in Echtzeit. Sie können Kennzahlen erfassen und verfolgen, benutzerdefinierte Dashboards erstellen und Alarme festlegen, die Sie benachrichtigen oder Maßnahmen ergreifen, wenn eine bestimmte Metrik einen von Ihnen festgelegten Schwellenwert erreicht. Sie können beispielsweise die CPU-Auslastung oder andere Kennzahlen Ihrer EC2 Amazon-Instances CloudWatch verfolgen und bei Bedarf automatisch neue Instances starten. Weitere Informationen finden Sie im CloudWatch Amazon-Benutzerhandbuch.
- Amazon CloudWatch Events liefert eine Reihe von Systemereignissen, die Änderungen an AWS Ressourcen beschreiben. In der Regel liefern AWS Services Ereignisbenachrichtigungen für CloudWatch Ereignisse innerhalb von Sekunden, manchmal kann dies jedoch eine Minute oder länger dauern. CloudWatch Events ermöglicht automatisiertes ereignisgesteuertes Computing, da Sie Regeln schreiben können, die auf bestimmte Ereignisse achten und bei deren Eintreten automatisierte Aktionen in anderen AWS Diensten auslösen können. Weitere Informationen finden Sie im Amazon CloudWatch Events-Benutzerhandbuch.
- Mit Amazon CloudWatch Logs können Sie Ihre Protokolldateien von EC2 Amazon-Instances und anderen Quellen überwachen CloudTrail, speichern und darauf zugreifen. CloudWatch Logs kann Informationen in den Protokolldateien überwachen und Sie benachrichtigen, wenn bestimmte Schwellenwerte erreicht werden. Sie können Ihre Protokolldaten auch in einem sehr robusten Speicher archivieren. Weitere Informationen finden Sie im Amazon CloudWatch Logs-Benutzerhandbuch.

Sie können Ihren MediaStore Containern auch Metadaten in Form von Tags zuweisen. Jedes Tag ist eine Markierung, die aus einem von Ihnen definierten Schlüssel und Wert besteht. Tags können

die Verwaltung, Suche und Filterung von Ressourcen erleichtern. Sie können Tags verwenden, um Ihre AWS Ressourcen in der AWS Management Console zu organisieren, Nutzungs- und Abrechnungsberichte für all Ihre AWS Ressourcen zu erstellen und Ressourcen bei Aktivitäten zur Infrastrukturautomatisierung zu filtern.

Topics

Protokollieren von AWS Elemental MediaStore API-Aufrufen mit AWS CloudTrail

AWS Elemental MediaStore ist in einen Service integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen bereitstellt, die von einem Benutzer, einer Rolle oder einem AWS Service in MediaStore ausgeführt wurden. CloudTrail erfasst eine Teilmenge von API-Aufrufen MediaStore als Ereignisse, einschließlich Aufrufe von der MediaStore Konsole und von Codeaufrufen an die MediaStore API. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen an einen Amazon S3 S3-Bucket aktivieren, einschließlich Ereignissen für MediaStore. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse trotzdem in der CloudTrail Konsole im Ereignisverlauf anzeigen. Anhand der von gesammelten Informationen können Sie ermitteln CloudTrail, an welche Adresse die Anfrage gestellt wurde MediaStore, von welcher IP-Adresse aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde und vieles mehr.

Weitere Informationen darüber CloudTrail, einschließlich der Konfiguration und Aktivierung, finden Sie im AWS CloudTrail Benutzerhandbuch.

Topics

MediaStoreAWS-Elementarinformationen in CloudTrail

CloudTrail ist in Ihrem AWS Konto aktiviert, wenn Sie das Konto erstellen. Wenn unterstützte Ereignisaktivitäten in AWS Elemental auftreten MediaStore, wird diese Aktivität zusammen mit

anderen AWS Serviceereignissen im CloudTrail Ereignisverlauf in einem Ereignis aufgezeichnet. Sie können aktuelle Ereignisse in Ihrem AWS Konto anzeigen, suchen und herunterladen. Weitere Informationen finden Sie unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS Konto, einschließlich der Ereignisse für MediaStore, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser für alle AWS -Regionen. Der Trail protokolliert Ereignisse aus allen Regionen der AWS Partition und übermittelt die Protokolldateien an den von Ihnen angegebenen Amazon S3 S3-Bucket. Darüber hinaus können Sie andere AWS Dienste konfigurieren, um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren. Weitere Informationen finden Sie unter den folgenden Themen:

- Übersicht zum Erstellen eines Trails
- CloudTrail Unterstützte Dienste und Integrationen
- Konfiguration von Amazon SNS SNS-Benachrichtigungen für CloudTrail
- Empfangen von CloudTrail Protokolldateien aus mehreren Regionen und Empfangen von CloudTrail Protokolldateien von mehreren Konten

AWS Elemental MediaStore unterstützt die Protokollierung der folgenden Vorgänge als Ereignisse in CloudTrail Protokolldateien:

- `CreateContainer`
- `DeleteContainer`
- `DeleteContainerPolicy`
- `DeleteCorsPolicy`
- `DescribeContainer`
- `GetContainerPolicy`
- `GetCorsPolicy`
- `ListContainers`
- `PutContainerPolicy`
- `PutCorsPolicy`

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 Root-Benutzer- oder Benutzeranmeldedaten gestellt wurde
- Ob die Anfrage mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen föderierten Benutzer ausgeführt wurde
- Ob die Anfrage von einem anderen AWS Dienst gestellt wurde

Weitere Informationen finden Sie unter CloudTrail userIdentity-Element.

Beispiel: Einträge in der AWS Elemental MediaStore Elemental-Protokolldatei

Ein Trail ist eine Konfiguration, die die Übertragung von Ereignissen als Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket ermöglicht. CloudTrail Protokolldateien enthalten einen oder mehrere Protokolleinträge. Ein Ereignis stellt eine einzelne Anfrage aus einer beliebigen Quelle dar und enthält unter anderem Informationen über die angeforderte Aktion, das Datum und die Uhrzeit der Aktion sowie über die Anfrageparameter. CloudTrail -Protokolldateien sind kein geordnetes Stacktrace der öffentlichen API-Aufrufe und erscheinen daher nicht in einer bestimmten Reihenfolge.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der den `CreateContainer` Vorgang veranschaulicht:

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "ABCDEFGHIJKL123456789",
        "arn": "arn:aws:iam::111122223333:user/testUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "testUser",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-07-09T12:55:42Z"
            }
        },
```

```
        "invokedBy": "signin.amazonaws.com"
    },
    "eventTime": "2018-07-09T12:56:54Z",
    "eventSource": "mediastore.amazonaws.com",
    "eventName": "CreateContainer",
    "awsRegion": "ap-northeast-1",
    "sourceIPAddress": "54.239.119.16",
    "userAgent": "signin.amazonaws.com",
    "requestParameters": {
        "containerName": "TestContainer"
    },
    "responseElements": {
        "container": {
            "status": "CREATING",
            "creationTime": "Jul 9, 2018 12:56:54 PM",
            "name": " TestContainer ",
            "aRN": "arn:aws:mediastore:ap-northeast-1:111122223333:container/
TestContainer"
        }
    },
    "requestID":
 "MNCTGH4HRQJ27GRMBVDPIVHEP4LO2BN6MUVHBCPSHOAWNSOKSXCO24B2UEOBBND5DONRXTMFK3TOJ4G7AHWMESI",
    "eventID": "7085b140-fb2c-409b-a329-f567912d704c",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```


Überwachung von AWS Elemental MediaStore mit Amazon CloudWatch

Sie können AWS Elemental MediaStore mithilfe von AWS Elemental überwachen CloudWatch, das Rohdaten sammelt und zu lesbaren Metriken verarbeitet. CloudWatch speichert Statistiken für 15 Monate, sodass Sie auf historische Informationen zugreifen und sich einen besseren Überblick über die Leistung Ihrer Webanwendung oder Ihres Dienstes verschaffen können. Sie können auch Alarme einrichten, die auf bestimmte Grenzwerte achten und Benachrichtigungen senden oder Aktivitäten auslösen, wenn diese Grenzwerte erreicht werden. Weitere Informationen finden Sie im CloudWatch Amazon-Benutzerhandbuch.

AWS bietet die folgenden Überwachungstools, mit denen Sie beobachten MediaStore, melden können, wenn etwas nicht stimmt, und gegebenenfalls automatische Maßnahmen ergreifen können:

- Mit Amazon CloudWatch Logs können Sie Ihre Protokolldateien von AWS Diensten wie AWS Elemental MediaStore überwachen, speichern und darauf zugreifen. Sie können CloudWatch Logs verwenden, um Anwendungen und Systeme mithilfe von Protokolldaten zu überwachen. Mithilfe von CloudWatch Protokollen können Sie beispielsweise die Anzahl der in Ihren Anwendungsprotokollen aufgetretenen Fehler nachverfolgen und Ihnen eine Benachrichtigung senden, wenn die Fehlerrate einen von Ihnen angegebenen Schwellenwert überschreitet. CloudWatch Logs verwendet Ihre Protokolldaten zur Überwachung, sodass keine Codeänderungen erforderlich sind. Sie können beispielsweise Anwendungsprotokolle auf bestimmte wörtliche Begriffe (wie "ValidationException„) hin überwachen oder die Anzahl der `PutObject` Anfragen zählen, die in einem bestimmten Zeitraum gestellt wurden. Wenn der Begriff, nach dem Sie suchen, gefunden wird, CloudWatch meldet Logs die Daten anhand einer von Ihnen angegebenen CloudWatch Metrik. Die Protokolldaten werden während der Übermittlung und Speicherung verschlüsselt.
- Amazon CloudWatch Events liefert Systemereignisse, die Änderungen an AWS Ressourcen, wie MediaStore Objekten, beschreiben. In der Regel liefern AWS Services Ereignisbenachrichtigungen für CloudWatch Ereignisse innerhalb von Sekunden, manchmal kann dies jedoch eine Minute oder länger dauern. Sie können Regeln einrichten, um Ereignisse (z. B. eine `DeleteObject` Anfrage) zuzuordnen und sie an eine oder mehrere Zielfunktionen oder Streams weiterzuleiten. CloudWatch Events erkennt betriebliche Änderungen, sobald sie eintreten. Darüber hinaus reagiert CloudWatch Events auf diese betrieblichen Änderungen und ergreift bei Bedarf Korrekturmaßnahmen, indem es Nachrichten sendet, um auf die Umgebung zu reagieren, Funktionen aktiviert, Änderungen vornimmt und Statusinformationen erfasst.

CloudWatch Protokolle

Die Zugriffsprotokollierung stellt detaillierte Aufzeichnungen über die Anfragen bereit, die an Objekte in einem Container gestellt wurden. Zugriffsprotokolle sind für für viele Anwendungen nützlich, wie z. B. für Sicherheits- und Zugriffsüberprüfungen. Sie können Ihnen auch dabei helfen, mehr über Ihren Kundenstamm zu erfahren und Ihre MediaStore Rechnung zu verstehen. CloudWatch Protokolle werden wie folgt kategorisiert:

- Ein Protokollstream ist eine Abfolge von Protokollereignissen, die dieselbe Quelle nutzen.
- Eine Protokollgruppe ist eine Gruppe von Protokollstreams, die dieselben Einstellungen für die Aufbewahrung, Überwachung und Zugriffskontrolle besitzen. Wenn Sie die Zugriffsprotokollierung für einen Container aktivieren, MediaStore wird eine Protokollgruppe mit einem Namen wie erstellt/`aws/mediastore/MyContainerName`. Sie können Protokollgruppen definieren und angeben,

welche Streams in welche Gruppe geschickt werden sollen. Es gibt kein Kontingent dazu, wie viele Protokoll-Streams zu einer Protokollgruppe gehören können.

Standardmäßig werden Protokolle unbegrenzt aufbewahrt und laufen nicht ab. Sie können die Aufbewahrungsrichtlinie für jede Protokollgruppe anpassen und Protokolle entweder unbegrenzt speichern oder einen Aufbewahrungszeitraum zwischen einem Tag und 10 Jahren auswählen.

Berechtigungen für Amazon einrichten CloudWatch

Verwenden Sie AWS Identity and Access Management (IAM), um eine Rolle zu erstellen, die AWS Elemental MediaStore Zugriff auf Amazon gewährt. CloudWatch Sie müssen diese Schritte ausführen, damit CloudWatch Logs für Ihr Konto veröffentlicht werden können. CloudWatch veröffentlicht automatisch Metriken für Ihr Konto.

Um den MediaStore Zugriff zu ermöglichen CloudWatch

1. Öffnen Sie unter https://console.aws.amazon.com/iam/ die IAM-Konsole.
2. Wählen Sie in der IAM-Konsole im Navigationsbereich die Option Policies (Richtlinien) und dann Create policy (Richtlinie erstellen) aus.
3. Wählen Sie die Registerkarte JSON und fügen Sie dann die folgende Richtlinie ein:

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents"
```

```
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/mediastore/*"
        }
    ]
}
```

Diese Richtlinie ermöglicht MediaStore die Erstellung von Protokollgruppen und Protokollstreams für alle Container in jeder Region innerhalb Ihres AWS Kontos.

4. Wählen Sie Richtlinie prüfen.
5. Geben Sie auf der Seite Review policy (Richtlinie prüfen) für Name den Namen **`MediaStoreAccessLogsPolicy`** ein und wählen Sie dann Create policy (Richtlinie erstellen).
6. Klicken Sie im Navigationsbereich der IAM-Konsole auf Rollen, und wählen Sie dann Rolle erstellen.
7. Wählen Sie den Rollentyp Another AWS account (Anderes AWS-Konto) aus.
8. Geben Sie als Konto-ID Ihre AWS Konto-ID ein.
9. Wählen Sie Weiter: Berechtigungen aus.
10. Geben Sie in das Suchfeld **`MediaStoreAccessLogsPolicy`** ein.
11. Aktivieren Sie das Kontrollkästchen neben der neuen Richtlinie und klicken Sie dann auf Next: Tags (Weiter: Tags).
12. Wählen Sie Next: Review (Weiter: Prüfen) aus, um eine Vorschau Ihres neuen Benutzers anzuzeigen.
13. Geben Sie für Role name (Rollenname) den Namen **`MediaStoreAccessLogs`** ein und klicken Sie auf Create role (Rolle erstellen).
14. Wählen Sie in der Bestätigungsmeldung den Namen der Rolle, die Sie gerade erstellt haben (**`MediaStoreAccessLogs`**).
15. Wählen Sie auf der Seite Summary (Übersicht) der Rolle die Registerkarte Trust relationships (Vertrauensstellungen).
16. Wählen Sie Vertrauensstellung bearbeiten aus.
17. Ändern Sie im Richtliniendokument den Prinzipal auf den MediaStore-Service. Das sollte wie folgt aussehen:

```
"Principal": {
   "Service": "mediastore.amazonaws.com"
},
```

Die gesamte Richtlinie sollte folgendermaßen lauten:

JSON

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

18. Wählen Sie Update Trust Policy (Trust Policy aktualisieren).

Aktivieren der Zugriffsprotokollierung für einen Container

Standardmäßig sammelt AWS Elemental MediaStore keine Zugriffsprotokolle. Wenn Sie die Zugriffsprotokollierung für einen Container aktivieren, MediaStore werden Zugriffsprotokolle für Objekte, die in diesem Container gespeichert sind, an Amazon übermittelt CloudWatch. Die Zugriffsprotokolle enthalten detaillierte Datensätze für Anfragen, die an ein beliebiges im Container gespeichertes Objekt gestellt wurden. Diese Informationen können den Anfragetyp, die in der Anfrage angegebenen Ressourcen sowie die Uhrzeit und das Datum der Bearbeitung der Anfrage umfassen.

> **Important**
>
> Für die Aktivierung der Zugriffsprotokollierung auf einem MediaStore-Container fallen keine zusätzlichen Kosten an. Für die Protokolldateien, die der Service an Sie überträgt, fallen die normalen Gebühren für die Speicherung an. (Sie können die Protokolldateien jederzeit löschen.) AWS berechnet keine Datenübertragungskosten für die Übertragung der Protokolldateien. Für den Zugriff auf die Protokolldateien fällt aber die normale Gebühr für Datenübertragungen an.

So aktivieren Sie die Zugriffsprotokollierung (AWS CLI):

- Verwenden Sie in der AWS CLI den `start-access-logging` folgenden Befehl:

  ```
  aws mediastore start-access-logging --container-name LiveEvents --region us-west-2
  ```

  Dieser Befehl hat keinen Rückgabewert.

Deaktivieren der Zugriffsprotokollierung für einen Container

Wenn Sie die Zugriffsprotokollierung für einen Container deaktivieren, sendet AWS Elemental MediaStore keine Zugriffsprotokolle mehr an Amazon CloudWatch. Diese Zugriffsprotokolle werden nicht gespeichert und können nicht abgerufen werden.

Deaktivieren der Zugriffsprotokollierung (AWS CLI)

- Verwenden Sie in AWS CLI der den `stop-access-logging` folgenden Befehl:

  ```
  aws mediastore stop-access-logging --container-name LiveEvents --region us-west-2
  ```

  Dieser Befehl hat keinen Rückgabewert.

Fehlerbehebung bei der Zugriffsprotokollierung in AWS Elemental MediaStore

Wenn AWS Elemental MediaStore Access Logs nicht in Amazon erscheinen CloudWatch, finden Sie in der folgenden Tabelle mögliche Ursachen und Lösungen.

> **Note**
>
> Stellen Sie sicher, dass AWS CloudTrail Logs aktiviert ist, um Sie bei der Fehlerbehebung zu unterstützen.

Symptom	Das Problem ist möglicherweise...	Versuchen Sie...
Sie sehen keine CloudTrail Ereignisse, obwohl CloudTrail Protokolle aktiviert sind.	Die IAM-Rolle ist entweder nicht vorhanden oder hat einen falschen Namen, falsche Berechtigungen oder eine falsche Vertrauensrichtlinie.	Erstellen Sie eine Rolle mit dem richtigen Namen, den richtigen Berechtigungen und der richtigen Vertrauensrichtlinie. Siehe the section called “Berechtigungen einrichten für CloudWatch”.
Sie haben eine `DescribeContainer` -API-Anfrage gestellt, aber die Antwort zeigt, dass der `AccessLoggingEnabled` -Parameter den Wert `False` aufweist. Außerdem können Sie keine CloudTrail-Ereignisse für die `MediaStoreAccessLogs` -Rolle sehen, die einen erfolgreichen `DescribeLogGroup` -, `CreateLogGroup` -, `DescribeLogStream` - oder `CreateLogStream` -Anruf tätigt.	Die IAM-Rolle ist entweder nicht vorhanden oder hat einen falschen Namen, falsche Berechtigungen oder eine falsche Vertrauensrichtlinie.	Erstellen Sie eine Rolle mit dem richtigen Namen, den richtigen Berechtigungen und der richtigen Vertrauensrichtlinie. Siehe the section called “Berechtigungen einrichten für CloudWatch”.
	Die Zugriffsprotokollierung ist auf dem Container nicht aktiviert.	Aktivieren Sie die Zugriffsprotokolle für den Container . Siehe the section called “Aktivieren der Zugriffsprotokollierung”.
Auf der CloudTrail Konsole wird ein Ereignis mit dem Fehler „Zugriff verweigert“ angezeigt, das sich auf die `MediaStoreAccessLogs` Rolle bezieht. Das CloudTrail Ereignis kann Zeilen wie die folgenden enthalten: `"eventSource": "logs.amazonaws.com",`	Die IAM-Rolle hat nicht die richtigen Berechtigungen für AWS Elemental MediaStore.	Aktualisieren Sie die IAM-Rolle mit den richtigen Berechtigungen und Vertrauensrichtlinien. Siehe the section called “Berechtigungen einrichten für CloudWatch”.

Symptom	Das Problem ist möglicherweise...	Versuchen Sie...

`"errorCode": "AccessDenied",`
 	
`"errorMessage": "User: arn:aws:sts::111122223333:assumed-role/MediaStoreAccessLogs/MediaStoreAccessLogsSession is not authorized to perform: logs:DescribeLogGroups on resource: arn:aws:logs:us-west-2:111122223333:log-group::log-stream:",`		

Sie sehen keine Protokolle für einen ganzen Container oder mehrere Container.	Ihr Konto hat möglicherweise das CloudWatch Kontingent für Protokollgruppen pro Konto und Region überschritten. Die Kontingente für Protokollgruppen finden Sie im Amazon CloudWatch Logs-Benutzerhandbuch.	Stellen Sie auf der CloudWatch Konsole fest, ob Ihr Konto das CloudWatch Kontingent für Protokollgruppen erreicht hat. Bei Bedarf fordern Sie eine Kontingenterhöhung an.

Symptom	Das Problem ist möglicherweise...	Versuchen Sie...
Sie sehen einige Logs CloudWatch, aber nicht alle Logs, die Sie erwarten.	Ihr Konto hat möglicherweise das CloudWatch Kontingent für Transaktionen pro Sekunde, Konto und Region überschritten. Die Kontingente für finden Sie `PutLogEvents` im Amazon CloudWatch Logs-Benutzerhandbuch.	Beantragen Sie eine Erhöhung des Kontingents für CloudWatch Transaktionen pro Sekunde, Konto und Region.

Zugriffsprotokollformat

Die Zugriffsprotokolldateien bestehen aus einer Reihe von JSON-formatierten Protokolldatensätzen, wobei jeder Protokolldatensatz eine Anfrage darstellt. Die Reihenfolge der Felder innerhalb des Protokolls kann variieren. Im Folgenden finden Sie ein Beispielprotokoll, das aus zwei Protokolldatensätzen besteht:

```
{
  "Path": "/FootballMatch/West",
  "Requester": "arn:aws:iam::111122223333:user/maria-garcia",
  "AWSAccountId": "111122223333",
  "RequestID":
 "aaaAAA111bbbBBB222cccCCC333dddDDD444eeeEEE555fffFFF666gggGGG777hhhHHH888iiiIII999jjjJJJ",
  "ContainerName": "LiveEvents",
  "TotalTime": 147,
  "BytesReceived": 1572864,
  "BytesSent": 184,
  "ReceivedTime": "2018-12-13T12:22:06.245Z",
  "Operation": "PutObject",
  "ErrorCode": null,
  "Source": "192.0.2.3",
  "HTTPStatus": 200,
```

```
  "TurnAroundTime': 7,
  "ExpiresAt": "2018-12-13T12:22:36Z"
}
 {
  "Path": "/FootballMatch/West",
  "Requester": "arn:aws:iam::111122223333:user/maria-garcia",
  "AWSAccountId": "111122223333",
  "RequestID":
 "dddDDD444eeeEEE555fffFFF666gggGGG777hhhHHH888iiiIII999jjjJJJ000cccCCC333bbbBBB222aaaAAA",
  "ContainerName": "LiveEvents",
  "TotalTime": 3,
  "BytesReceived": 641354,
  "BytesSent": 163,
  "ReceivedTime": "2018-12-13T12:22:51.779Z",
  "Operation": "PutObject",
  "ErrorCode": "ValidationException",
  "Source": "198.51.100.15",
  "HTTPStatus": 400,
  "TurnAroundTime": 1,
  "ExpiresAt": null
}
```


In der folgenden Liste werden die wichtigsten Protokolldatensatzfelder beschrieben:

AWSAccountID

Die AWS Konto-ID des Kontos, das für die Anfrage verwendet wurde.

BytesReceived

Die Anzahl der Bytes im Anforderungstext, die der MediaStore-Server empfängt.

BytesSent

Die Anzahl der Bytes im Antworttext, die der MediaStore-Server sendet. Dieser Wert ist häufig identisch mit dem Wert des `Content-Length`-Headers, der in Serverantworten enthalten ist.

ContainerName

Der Name des Containers, der die Anfrage empfangen hat.

ErrorCode

Der MediaStore Fehlercode (z. B.`InternalServerError`). Wenn keine Fehler aufgetreten, wird das Zeichen `-` angezeigt. Ein Fehlercode kann sogar angezeigt werden, wenn der Statuscode 200

lautet (wird angezeigt, wenn eine geschlossene Verbindung oder eine Fehlermeldung angezeigt wird, nachdem der Server mit dem Streamen der Antwort begonnen hat).

ExpiresAt

Das Ablaufdatum und die Uhrzeit des Objekts. Dieser Wert basiert auf dem Ablaufalter, das transient data rule in einer Lebenszyklusrichtlinie festgelegt wurde, die auf den Container angewendet wird. Der Wert ist `ISO-8601`-Datum und -Uhrzeit und basiert auf der Systemuhr des Hosts, der die Anfrage verarbeitet. Wenn die Lebenszyklusrichtlinie keine Regel für vorübergehende Daten enthält, die für das Objekt gilt, oder wenn keine Lebenszyklusrichtlinie auf den Container angewendet wurde, lautet der Wert dieses Felds. `null` Dieses Feld gilt nur für die folgenden Operationen:`PutObject`,`GetObject`, `DescribeObject` und. `DeleteObject`

HTTPStatus

Der numerische HTTP-Statuscode der Antwort.

Operation

Die Operation, die durchgeführt wurde, z. B. `PutObject` oder `ListItems`.

Pfad

Der Pfad im Container, in dem das Objekt gespeichert ist. Wenn die Operation keinen Pfadparameter verwendet, wird das Zeichen `-` angezeigt.

ReceivedTime

Die Tageszeit, zu der die Anfrage empfangen wurde. Der Wert ist `ISO-8601`-Datum und -Uhrzeit und basiert auf der Systemuhr des Hosts, der die Anfrage verarbeitet.

Auftraggeber

Der Amazon Resource Name (ARN) des Benutzers des Kontos, das zum Erstellen der Anfrage verwendet wurde. Bei nicht authentifizierten Anfragen ist dieser Wert `anonymous`. Wenn die Anforderung fehlschlägt, bevor die Authentifizierung abgeschlossen ist, fehlt dieses Feld möglicherweise im Protokoll. Bei solchen Anforderungen ist möglicherweise am `ErrorCode` das Autorisierungsproblem zu erkennen.

RequestID

Eine Zeichenfolge, die von AWS Elemental generiert wird MediaStore , um jede Anfrage eindeutig zu identifizieren.

Quelle

Die offensichtliche Internetadresse des Auftraggebers oder des Service-Prinzipals des AWS-Service, der den Aufruf vornimmt. Wenn zwischengeschaltete Proxys und Firewalls die Adresse des Computers verschleiern, der die Anfrage stellt, wird der Wert auf Null gesetzt.

TotalTime

Die Anzahl der Millisekunden (ms), die die Anfrage aus Perspektive des Servers unterwegs war. Dieser Wert wird ab der Zeit gemessen, zu der Ihre Anfrage vom Service empfangen wird, und bis zu der Zeit, zu der das letzte Byte der Antwort gesendet wurde. Dieser Wert wird von der Perspektive des Servers aus gemessen, da Messungen aus der Perspektive des Clients von der Netzwerklatenz beeinträchtigt sind.

TurnAroundTime

Die Anzahl der Millisekunden, die für die MediaStore Bearbeitung Ihrer Anfrage aufgewendet wurden. Dieser Wert wird ab der Zeit gemessen, zu der das letzte Byte Ihrer Anforderung empfangen wurde, bis zu der Zeit, zu der das erste Byte der Antwort gesendet wurde.

Die Reihenfolge der Felder in der Protokolldatei kann variieren.

Protokollierungsstatusänderungen werden mit der Zeit wirksam.

Änderungen am Protokollierungsstatus eines Containers benötigen einige Zeit, bis sie sich auf die Bereitstellung von Protokolldateien auswirken. Wenn Sie beispielsweise die Protokollierung für einen Container A aktivieren, werden möglicherweise einige Anfragen, die in der darauffolgenden Stunde gestellt werden, protokolliert, andere hingegen nicht. Wenn Sie die Protokollierung für Container B deaktivieren, werden in der nächsten Stunde einige Protokolle möglicherweise weiter zugestellt, andere möglicherweise nicht. Die neuen Einstellungen werden letztendlich in allen Fällen ohne weiteres Eingreifen Ihrerseits wirksam.

Best-Effort-Protokollbereitstellung der Server

Zugriffsprotokoll-Datensätze werden auf Best-Effort-Basis bereitgestellt. Die meisten Anfragen nach einem Container, der für die Protokollierung richtig konfiguriert ist, führen zu einem ausgelieferten Protokollsatz. Die meisten Protokollsätze werden innerhalb weniger Stunden nach der Aufnahme geliefert, können aber häufiger geliefert werden.

Die Vollständigkeit und Aktualität der Zugriffsprotokollierung wird nicht garantiert. Der Protokolldatensatz für eine bestimmte Anforderung wird möglicherweise viel später bereitgestellt, als

die Anforderung tatsächlich verarbeitet wurde; es kann auch sein, dass er gar nicht bereitgestellt wird. Der Zweck der Zugriffsprotokolle besteht darin, Ihnen einen Überblick über die Art des Datenverkehrs zu und von Ihrem Container zu vermitteln. Es passiert selten, das Protokolldatensätze verloren gehen, aber die Zugriffsprotokollierung ist nicht als vollständige Auflistung aller Anfragen vorgesehen.

Aufgrund der Best-Effort-Natur der Zugriffsprotokollierungsfunktion können die im AWS-Portal verfügbaren Nutzungsberichte (Abrechnungs- und Kostenverwaltungsberichte auf der AWS-Managementkonsole) eine oder mehrere Zugriffsanfragen enthalten, die nicht in einem bereitgestellten Zugriffsprotokoll angezeigt werden.

Überlegungen bei der Programmierung des Zugriffsprotokollformats

Wir erweitern möglicherweise von Zeit zu Zeit das Zugriffsprotokollformat, indem wir neue Felder hinzufügen. Code, der Zugriffprotokolle analysiert, muss so geschrieben werden, dass er zusätzliche Felder verarbeiten kann, die er nicht versteht.

CloudWatch Events

Mit Amazon CloudWatch Events können Sie Ihre AWS Services automatisieren und automatisch auf Systemereignisse wie Probleme mit der Anwendungsverfügbarkeit oder Ressourcenänderungen reagieren. Sie können einfache Regeln schreiben, um anzugeben, welche Ereignisse für Sie interessant sind und welche automatisierten Aktionen durchgeführt werden sollen, wenn sich für ein Ereignis eine Übereinstimmung mit einer Regel ergibt.

> **Important**
>
> In der Regel liefern AWS Services innerhalb von Sekunden Ereignisbenachrichtigungen für CloudWatch Ereignisse, manchmal kann dies jedoch eine Minute oder länger dauern.

Wenn eine Datei in einen Container hochgeladen oder aus einem Container entfernt wird, werden im CloudWatch Service nacheinander zwei Ereignisse ausgelöst:

1. the section called “Object State Change Ereignis”
2. the section called “Container State Change Ereignis”

Informationen zum Abonnieren dieser Veranstaltungen finden Sie auf Amazon CloudWatch.

Die folgenden Aktionen können beispielsweise automatisch ausgelöst werden:

- Eine Funktion aufrufen AWS Lambda
- Amazon EC2 Run Command aufrufen
- Weiterleiten des Ereignisses an Amazon Kinesis Data Streams
- Aktivierung einer AWS Step Functions Zustandsmaschine
- Ein Amazon SNS SNS-Thema oder eine Warteschlange benachrichtigen AWS SMS

Einige Beispiele für die Verwendung von CloudWatch Events mit AWS Elemental MediaStore sind:

- Aktivierung einer Lambda-Funktion bei jeder Erstellung eines Containers
- Ein Amazon SNS SNS-Thema benachrichtigen, wenn ein Objekt gelöscht wird

Weitere Informationen finden Sie im Amazon CloudWatch Events-Benutzerhandbuch.

Topics

Ereignis zur Änderung des Status MediaStore eines AWS-Elementar-Objekts

Dieses Ereignis wird veröffentlicht, wenn sich der Status eines Objekts geändert hat (wenn das Objekt hochgeladen oder gelöscht wurde).

> **Note**
>
> Objekte, die aufgrund einer Regel für vorübergehende Daten ablaufen, geben kein CloudWatch Ereignis aus, wenn sie ablaufen.

Informationen zum Abonnieren dieser Veranstaltung finden Sie auf Amazon CloudWatch.

Objekt aktualisiert

```
{
  "version": "1",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
```

```
  "detail-type": "MediaStore Object State Change",
  "source": "aws.mediastore",
  "account": "111122223333",
  "time": "2017-02-22T18:43:48Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:mediastore:us-east-1:111122223333:MondayMornings/Episode1/
Introduction.avi"
  ],
  "detail": {
    "ContainerName": "Movies",
    "Operation": "UPDATE",
    "Path":"TVShow/Episode1/Pilot.avi",
    "ObjectSize":123456,
    "URL": "https://a832p1qeaznlp9.files.mediastore-us-west-2.com/Movies/
MondayMornings/Episode1/Introduction.avi"
  }
}
```

Objekt entfernt

```
{
  "version": "1",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "MediaStore Object State Change",
  "source": "aws.mediastore",
  "account": "111122223333",
  "time": "2017-02-22T18:43:48Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:mediastore:us-east-1:111122223333:Movies/MondayMornings/Episode1/
Introduction.avi"
  ],
  "detail": {
    "ContainerName": "Movies",
    "Operation": "REMOVE",
    "Path":"Movies/MondayMornings/Episode1/Introduction.avi",
    "URL": "https://a832p1qeaznlp9.files.mediastore-us-west-2.com/Movies/
MondayMornings/Episode1/Introduction.avi"
  }
}
```

Ereignis zur Änderung des Status von AWS Elemental MediaStore Containern

Dieses Ereignis wird veröffentlicht, wenn sich der Status eines Containers geändert hat (wenn ein Container hinzugefügt oder gelöscht wurde). Informationen zum Abonnieren dieser Veranstaltung finden Sie auf Amazon CloudWatch.

Container erstellt

```
{
  "version": "1",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "MediaStore Container State Change",
  "source": "aws.mediastore",
  "account": "111122223333",
  "time": "2017-02-22T18:43:48Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:mediastore:us-east-1:111122223333:container/Movies"
  ],
  "detail": {
    "ContainerName": "Movies",
    "Operation": "CREATE"
    "Endpoint": "https://a832p1qeaznlp9.mediastore-us-west-2.amazonaws.com"
  }
}
```

Container entfernt

```
{
  "version": "1",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "MediaStore Container State Change",
  "source": "aws.mediastore",
  "account": "111122223333",
  "time": "2017-02-22T18:43:48Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:mediastore:us-east-1:111122223333:container/Movies"
  ],
  "detail": {
    "ContainerName": "Movies",
    "Operation": "REMOVE"
```

```
  }
}
```

Überwachung von AWS Elemental MediaStore mit CloudWatch Amazon-Metriken

Sie können AWS Elemental MediaStore mithilfe von AWS Elemental überwachen CloudWatch, das Rohdaten sammelt und zu lesbaren Metriken verarbeitet. CloudWatchspeichert Statistiken für 15 Monate, sodass Sie auf historische Informationen zugreifen und sich einen besseren Überblick über die Leistung Ihrer Webanwendung oder Ihres Dienstes verschaffen können. Sie können auch Alarme einrichten, die auf bestimmte Grenzwerte achten und Benachrichtigungen senden oder Aktivitäten auslösen, wenn diese Grenzwerte erreicht werden. Weitere Informationen finden Sie im CloudWatch Amazon-Benutzerhandbuch.

Bei AWS Elemental MediaStore möchten Sie vielleicht zuschauen `BytesDownloaded` und eine E-Mail an sich selbst senden, wenn diese Metrik einen bestimmten Schwellenwert erreicht.

Um Metriken mit der CloudWatch Konsole anzuzeigen

Metriken werden zunächst nach dem Service-Namespace und anschließend nach den verschiedenen Dimensionskombinationen in den einzelnen Namespaces gruppiert.

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Metriken aus.
3. Wählen Sie unter Alle Metriken den AWS/ MediaStore Namespace aus.
4. Wählen Sie die Metrikdimension aus, um die Metriken anzuzeigen. Wählen Sie beispielsweise `Request metrics by container` aus, um Metriken für die verschiedenen Arten von Anforderungen anzuzeigen, die an den Container gesendet wurden.

Um Metriken anzuzeigen, verwenden Sie AWS CLI

- Geben Sie in einer Eingabeaufforderung den folgenden Befehl ein:

```
aws cloudwatch list-metrics --namespace "AWS/MediaStore"
```

AWS Elemental Elemental-Metriken MediaStore

In der folgenden Tabelle sind Metriken aufgeführt, an die AWS Elemental MediaStore sendet. CloudWatch

> **Note**
>
> Um Metriken anzuzeigen, müssen Sie dem Container eine Metrikrichtlinie hinzufügen, damit Metriken MediaStore an Amazon gesendet werden können CloudWatch.

Metrik	Description
`RequestCount`	Die Gesamtzahl der an einen MediaStore-Container gestellten HTTP-Anforderungen, getrennt durch den Vorgangstyp (`Put`, `Get`, `Delete`, `Describe`, `List`). Einheiten: Anzahl Gültige Dimensionen: • Container-Name • Objektgruppenname • Typ der Anforderung Gültige Statistiken: Summe
`4xxErrorCount`	Die Anzahl der HTTP-Anfragen, die an MediaStore diese Stelle gestellt wurden, führte zu einem 4xx-Fehler. Einheiten: Anzahl Gültige Dimensionen: • Container-Name • Objektgruppenname • Typ der Anforderung

Metrik	Description
	Gültige Statistiken: Summe
`5xxErrorCount`	Die Anzahl der HTTP-Anfragen, die an diese Stelle gestellt wurden MediaStore , führte zu einem 5xx-Fehler. Einheiten: Anzahl Gültige Dimensionen: • Container-Name • Objektgruppenname • Typ der Anforderung Gültige Statistiken: Summe
`BytesUploaded`	Die Anzahl der Bytes, die für Anforderungen an einen MediaStore -Container hochgeladen wurden, wobei die Anforderung einen Text enthält. Einheiten: Byte Gültige Dimensionen: • Container-Name • Objektgruppenname Gültige Statistiken: Durchschnitt (Byte pro Anforderung), Summe (Byte pro Zeitraum), Stichprobenanzahl, Min (entspricht P0,0), Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p99,9

Metrik	Description
`BytesDownloaded`	Anzahl der heruntergeladenen Bytes für Anforderungen an einen MediaStore -Container, wobei die Antwort einen Text enthält. Einheiten: Byte Gültige Dimensionen: • Container-Name • Objektgruppenname Gültige Statistiken: Durchschnitt (Byte pro Anforderung), Summe (Byte pro Zeitraum), Stichprobenanzahl, Min (entspricht P0,0), Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p99,9
`TotalTime`	Die Anzahl der Millisekunden (ms), die die Anforderung aus Perspektive des Servers unterwegs war. Dieser Wert wird vom Zeitpunkt des MediaStore Eingangs Ihrer Anfrage bis zum Senden des letzten Byte der Antwort gemessen. Dieser Wert wird von der Perspektive des Servers aus gemessen, da Messungen aus der Perspektive des Clients von der Netzwerklatenz beeinträchtigt sind. Einheiten: Millisekunden Gültige Dimensionen: • Container-Name • Objektgruppenname • Typ der Anforderung Gültige Statistiken: Durchschnitt, Min (entspricht P0,0), Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p100

Metrik	Description
`TurnaroundTime`	Die Anzahl der Millisekunden, die für die Bearbeitung Ihrer Anfrage MediaStore aufgewendet wurden. Dieser Wert wird ab dem Zeitpunkt gemessen, zu dem das letzte Byte Ihrer Anfrage MediaStore empfangen wird, bis zu dem Zeitpunkt, zu dem das erste Byte der Antwort gesendet wird. Einheiten: Millisekunden Gültige Dimensionen: • Container-Name • Objektgruppenname • Typ der Anforderung Gültige Statistiken: Durchschnitt, Min (entspricht P0,0), Max (entspricht p100), beliebiges Perzentil zwischen p0,0 und p100
`ThrottleCount`	Die Anzahl der HTTP-Anfragen, die an diese gestellt wurden MediaStore , wurde gedrosselt. Einheiten: Anzahl Gültige Dimensionen: • Container-Name • Objektgruppenname • Typ der Anforderung Gültige Statistiken: Summe

Taggen von AWS Elemental Elemental-Ressourcen MediaStore

Ein Tag ist eine benutzerdefinierte Attributbezeichnung, die Sie zuweisen oder die einer AWS AWS Ressource zugewiesen wird. Jedes -Tag besteht aus zwei Teilen:

- einem Tag-Schlüssel (z. B. `CostCenter`, `Environment` oder `Project`). Bei Tag-Schlüsseln wird zwischen Groß- und Kleinschreibung unterschieden.
- einem optionalen Feld, dem sogenannten Tag-Wert (z. B. `111122223333` oder `Production`). Ein nicht angegebener Tag-Wert entspricht einer leeren Zeichenfolge. Wie bei Tag-Schlüsseln wird auch bei Tag-Werten zwischen Groß- und Kleinschreibung unterschieden.

Tags sind für folgende Aktivitäten nützlich:

- Identifizieren und organisieren Sie Ihre AWS Ressourcen. Viele AWS-Services unterstützen das Markieren mit Tags (kurz: Tagging). So können Ressourcen aus verschiedenen Services denselben Tag zuweisen, um anzugeben, dass die Ressourcen verbunden sind. Sie könnten beispielsweise einem AWS Elemental dasselbe Tag zuweisen MediaStore *`container`*, das Sie einer AWS Elemental MediaLive Eingabe zuweisen.
- Überwachen Ihrer AWS-Kosten. Sie aktivieren diese Tags auf dem AWS Fakturierung und Kostenmanagement Dashboard. AWS verwendet die Tags zur Kategorisierung Ihrer Kosten und zur Bereitstellung eines monatlichen Kostenzuordnungsberichts für Sie. Weitere Informationen finden Sie unter Verwendung von Kostenzuordnungs-Tags im AWS Billing -Benutzerhandbuch.

In den folgenden Abschnitten finden Sie weitere Informationen zu Tags für AWS Elemental MediaStore.

Unterstützte Ressourcen in AWS Elemental MediaStore

Die folgenden Ressourcen in AWS Elemental MediaStore unterstützen Tagging:

- *`container`*

Weitere Informationen zum Hinzufügen und Verwalten von Tags finden Sie unter Tags verwalten.

AWS Elemental unterstützt die tagbasierte Zugriffskontrollfunktion von AWS Identity and Access Management (IAM) MediaStore nicht.

Konventionen für die Tag-Benennung und -Verwendung

Die folgenden grundlegenden Benennungs- und Verwendungskonventionen gelten für die Verwendung von Tags mit AWS Elemental MediaStore Elemental-Ressourcen:

- Jede Ressource kann maximal 50 Tags haben.

- Jeder Tag muss für jede Ressource eindeutig sein. Jeder Tag kann nur einen Wert haben.
- Die maximale Länge des Tag-Schlüssels beträgt 128 Unicode-Zeichen in UTF-8.
- 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). EC2 Amazon-Ressourcen erlauben beliebige Zeichen.
- Bei Tag-Schlüsseln und -Werten muss die Groß- und Kleinschreibung beachtet werden. 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 diese Konvention für alle Tags. Vermeiden Sie die Verwendung von ähnlichen Tags mit uneinheitlicher Fallunterscheidung.
- Das `aws:` Präfix ist für Tags verboten; es ist für die AWS Verwendung reserviert. Sie können keine Tag-Schlüssel oder -Werte mit diesem Präfix bearbeiten oder löschen. Tags mit diesem Präfix werden nicht auf Ihre Tags pro Ressourcenkontingent angerechnet.

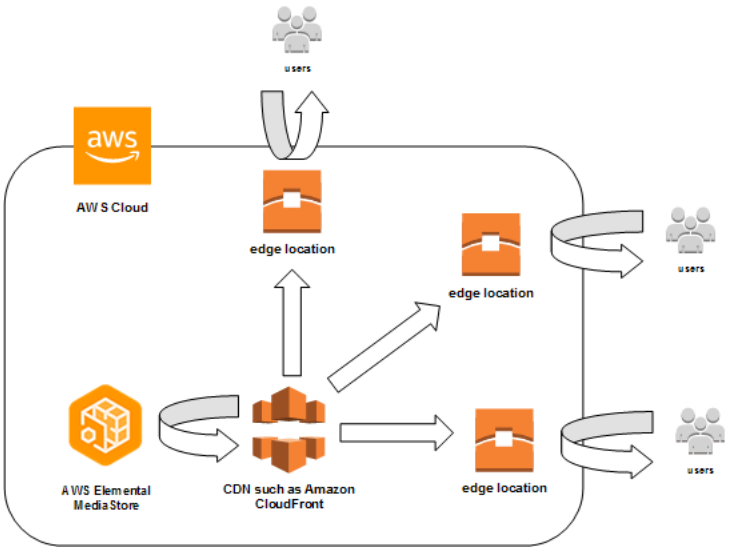
Tags verwalten

Tags bestehen aus den Eigenschaften `Key` und `Value` für eine Ressource. Sie können die AWS CLI oder die MediaStore API verwenden, um die Werte für diese Eigenschaften hinzuzufügen, zu bearbeiten oder zu löschen. Informationen zur Arbeit mit Tags finden Sie in den folgenden Abschnitten der AWS Elemental MediaStore API-Referenz:

- CreateContainer
- ListTagsForResource
- Ressourcen
- TagResource
- UntagResource

Arbeiten mit Content Delivery Networks (CDNs)

Sie können ein Content Delivery Network (CDN) wie Amazon verwenden CloudFront, um die Inhalte bereitzustellen, die Sie in AWS Elemental MediaStore speichern. Ein CDN ist eine weltweit verteilte Gruppe von Servern, die Inhalte wie Videos zwischenspeichern. Wenn ein Benutzer Ihren Inhalt anfordert, leitet das CDN die Anfrage an den Edge-Standort weiter, der die niedrigste Latenz bietet. Wenn Ihr Inhalt bereits an diesem Edge-Standort zwischengespeichert ist, stellt das CDN ihn unmittelbar bereit. Wenn sich Ihre Inhalte derzeit nicht an diesem Edge-Standort befinden, ruft das CDN sie von Ihrem Ursprung ab (z. B. Ihrem MediaStore Container) und verteilt sie an den Benutzer.



Topics

- Amazon CloudFront den Zugriff auf Ihren AWS Elemental Container MediaStore erlauben
- Die Interaktion MediaStore von AWS Elemental mit HTTP-Caches

Amazon CloudFront den Zugriff auf Ihren AWS Elemental Container MediaStore erlauben

Sie können Amazon verwenden CloudFront , um die Inhalte bereitzustellen, die Sie in einem Container in AWS Elemental MediaStore speichern. Sie können dies auf eine der folgenden Arten tun:

- Verwenden Sie Origin Access Control (OAC)- (Empfohlen) Verwenden Sie diese Option, wenn Sie die OAC-Funktion von AWS-Region CloudFront unterstützen.

- Shared Secrets verwenden- Verwenden Sie diese Option, wenn Sie die OAC-Funktion von AWS-Region nicht unterstützen. CloudFront

Verwenden Sie Origin Access Control (OAC)

Sie können die Origin Access Control (OAC) -Funktion von Amazon verwenden, CloudFront um AWS Elemental MediaStore Origins mit verbesserter Sicherheit zu sichern. Sie können AWS Signature Version 4 (Sigv4) für CloudFront Anfragen nach MediaStore Ursprüngen aktivieren und festlegen, wann und ob die Anfragen signiert CloudFront werden sollen. Sie können CloudFront über die Konsole, APIs das SDK oder die CLI auf die OAC-Funktion von zugreifen, und für deren Nutzung fallen keine zusätzlichen Gebühren an.

Weitere Informationen zur Verwendung der OAC-Funktion mit MediaStore finden Sie unter Beschränken des Zugriffs auf einen MediaStore Ursprung im Amazon CloudFront Developer Guide.

Shared Secrets verwenden

Wenn Sie die OAC-Funktion von Amazon AWS-Region nicht unterstützen CloudFront, können Sie Ihrem AWS Elemental MediaStore Elemental-Container eine Richtlinie hinzufügen, die Lesezugriff oder mehr gewährt für. CloudFront

> **Note**
>
> Wir empfehlen die Verwendung der OAC-Funktion, sofern Sie AWS-Region sie unterstützen. Bei den folgenden Verfahren müssen Sie Shared Secrets konfigurieren MediaStore , um den Zugriff auf MediaStore Container einzuschränken. CloudFront Um den besten Sicherheitspraktiken zu folgen, erfordert diese manuelle Konfiguration eine regelmäßige Rotation von Geheimnissen. Wenn OAC auf MediaStore Origins gesetzt ist, können Sie anweisen, Anfragen mit SigV4 CloudFront zu signieren und sie MediaStore für den Signaturabgleich an diese weiterzuleiten. Dadurch entfällt die Notwendigkeit, geheime Daten zu verwenden und zu rotieren. Dadurch wird sichergestellt, dass Anfragen vor der Bereitstellung von Medieninhalten automatisch verifiziert werden, wodurch die Übermittlung von Medieninhalten CloudFront einfacher MediaStore und sicherer wird.

Um den CloudFront Zugriff auf Ihren Container (Konsole) zu ermöglichen

1. Öffnen Sie die MediaStore Konsole unter https://console.aws.amazon.com/mediastore/.

2. Wählen Sie auf der Seite Containers (Container) den Containernamen aus.

   Die Seite mit den Containerdetails wird angezeigt.

3. Fügen Sie im Abschnitt Container-Richtlinie eine Richtlinie hinzu, die Amazon Lesezugriff oder mehr gewährt CloudFront.

   Example

   Die folgende Beispielrichtlinie, die der Beispielrichtlinie für öffentlichen Lesezugriff über HTTPS ähnelt, erfüllt diese Anforderungen, da sie jedem, der Anfragen über HTTPS an Ihre Domain sendet, erlaubt `GetObject` und `DescribeObject` Befehle erteilt. Darüber hinaus schützt die folgende Beispielrichtlinie Ihren Workflow besser, da sie den CloudFront Zugriff auf MediaStore Objekte nur ermöglicht, wenn die Anfrage über eine HTTPS-Verbindung erfolgt und den richtigen Referer-Header enthält.

4. Weisen Sie im Abschnitt Container CORS policy (Container CORS-Richtlinie) eine Richtlinie zu, die die entsprechende Zugriffsebene gewährt.

   > **Note**
   >
   > Eine CORS-Richtlinie ist nur dann erforderlich, wenn Sie Zugriff auf einen browserbasierten Player gewähren wollen.

5. Notieren Sie dabei die folgenden Details:

   - Den Datenendpunkt, der Ihrem Container zugeordnet ist. Sie finden diese Informationen im Abschnitt Info auf der Seite Containers (Container). In CloudFront wird der Datenendpunkt als Ursprungsdomänenname bezeichnet.
   - Die Ordnerstruktur im Container, in dem die Objekte gespeichert werden. CloudFrontIn wird dies als Ursprungspfad bezeichnet. Beachten Sie, dass diese Einstellung optional ist. Weitere Informationen zu Herkunftspfaden finden Sie im Amazon CloudFront Developer Guide.

6. Erstellen Sie in eine Distribution CloudFront, die für die Bereitstellung von Inhalten aus AWS Elemental MediaStore konfiguriert ist. Sie benötigen dazu die Informationen, die Sie im vorigen Schritt aufgezeichnet haben.

Nachdem Sie die Richtlinie an Ihre MediaStore Container angehängt haben, müssen Sie sie so konfigurieren, CloudFront dass nur HTTPS-Verbindungen für ursprüngliche Anfragen verwendet

werden. Außerdem müssen Sie einen benutzerdefinierten Header mit dem richtigen geheimen Wert hinzufügen.

Um den Zugriff auf Ihren Container über eine HTTPS-Verbindung mit einem geheimen Wert für den Referer-Header (Konsole) zu konfigurieren CloudFront

1. Öffnen Sie die CloudFront Konsole.
2. Wähle auf der Origins-Seite deinen MediaStore Ursprung aus.
3. Wählen Sie Bearbeiten aus.
4. Wählen Sie nur HTTPS für das Protokoll.
5. Wählen Sie im Abschnitt Benutzerdefinierte Kopfzeile hinzufügen die Option Kopfzeile hinzufügen aus.
6. Wählen Sie als Namen die Option Referer aus. Verwenden Sie für den Wert dieselbe *<secretValue>* Zeichenfolge, die Sie in Ihrer Container-Richtlinie verwendet haben.
7. Wählen Sie Speichern und lassen Sie die Änderungen bereitstellen.

Die Interaktion MediaStore von AWS Elemental mit HTTP-Caches

AWS Elemental MediaStore speichert Objekte, sodass sie von Content Delivery Networks (CDNs) wie Amazon korrekt und effizient zwischengespeichert werden können. CloudFront Wenn ein Endbenutzer oder ein CDN ein Objekt von abruft MediaStore, gibt der Service HTTP-Header zurück, die das Caching-Verhalten des Objekts beeinflussen. (Die Standards für das HTTP 1.1-Caching-Verhalten finden Sie in Abschnitt 13.) RFC2616 Diese Header sind:

- **`ETag`** (nicht anpassbar) – Der Header für das Entitäts-Tag ist eine eindeutige Kennung für die Antwort, die MediaStore sendet. Standardkonforme Browser CDNs und Webbrowser verwenden dieses Tag als Schlüssel, mit dem das Objekt zwischengespeichert wird. MediaStore generiert automatisch eine `ETag` für jedes Objekt, wenn es hochgeladen wird. Sie können sich die Details eines Objekts ansehen, um seinen ETag Wert zu ermitteln.
- **`Last-Modified`**(nicht anpassbar) — Der Wert dieses Headers gibt das Datum und die Uhrzeit der Änderung des Objekts an. MediaStore generiert diesen Wert automatisch, wenn das Objekt hochgeladen wird.
- **`Cache-Control`** (anpassbar) – Der Wert dieses Headers steuert, wie lange ein Objekt zwischengespeichert werden soll, bevor das CDN überprüft, ob es geändert wurde. Sie können diesen Header auf einen beliebigen Wert setzen, wenn Sie ein Objekt mithilfe der CLI oder API in

einen MediaStore Container hochladen. Eine Übersicht über alle gültigen Werte finden Sie in der HTTP/1.1-Dokumentation. Wenn Sie diesen Wert beim Hochladen eines Objekts nicht festlegen, MediaStore wird dieser Header nicht zurückgegeben, wenn das Objekt abgerufen wird.

Der Cache-Control-Header wird meist dazu verwendet, um die Zwischenspeicherdauer für ein Objekt anzugeben. Angenommen, Ihre Videomanifestdatei wird häufig von einem Encoder überschrieben. Sie können den Wert für `max-age` auf 10 setzen, damit das Objekt nur 10 Sekunden lang zwischengespeichert wird. Ein weiteres Beispiel ist ein gespeichertes Videosegment, das nie überschrieben wird. Sie können den Wert für `max-age` auf 31 536 000 festlegen, damit das Objekt etwa 1 Jahr lang zwischengespeichert wird.

Bedingte Anforderungen

Bedingte Anfragen an MediaStore

MediaStore reagiert identisch auf bedingte Anfragen (unter Verwendung von Anforderungsheadern wie `If-Modified-Since` und`If-None-Match`, wie unter beschrieben RFC7232) und auf bedingungslose Anfragen. Das bedeutet, dass der Dienst, wenn MediaStore er eine gültige `GetObject` Anfrage erhält, das Objekt immer zurückgibt, auch wenn der Client das Objekt bereits hat.

Bedingte Anfragen an CDNs

CDNs die Inhalte im Namen von bereitstellen, MediaStore können bedingte Anfragen bearbeiten, indem sie zurücksenden`304 Not Modified`, wie in RFC7232 Abschnitt 4.1 beschrieben. Es muss also nicht der gesamte Objektinhalt übertragen werden, weil der Anforderer bereits ein Objekt hat, das zur bedingten Anforderung passt.

CDNs (und andere Caches, die HTTP/1.1-konform sind) basieren bei diesen Entscheidungen auf den `Cache-Control` Headern `ETag` und, die von den Ursprungsservern weitergeleitet werden. Um zu steuern, wie oft CDNs Originalserver nach Aktualisierungen von wiederholt abgerufenen Objekten abgefragt werden, legen Sie die `Cache-Control` Header für diese Objekte fest, wenn Sie sie hochladen. MediaStore MediaStore

Verwenden Sie diesen Service mit einem SDK AWS

AWS Software Development Kits (SDKs) sind für viele beliebte Programmiersprachen verfügbar. Jedes SDK bietet eine API, Codebeispiele und Dokumentation, die es Entwicklern erleichtern, Anwendungen in ihrer bevorzugten Sprache zu erstellen.

SDK-Dokumentation	Codebeispiele
AWS SDK für C++	AWS SDK für C++ Codebeispiele
AWS CLI	AWS CLI Code-Beispiele
AWS SDK für Go	AWS SDK für Go Code-Beispiele
AWS SDK für Java	AWS SDK für Java Code-Beispiele
AWS SDK für JavaScript	AWS SDK für JavaScript Code-Beispiele
AWS SDK für Kotlin	AWS SDK für Kotlin Code-Beispiele
AWS SDK für .NET	AWS SDK für .NET Code-Beispiele
AWS SDK für PHP	AWS SDK für PHP Code-Beispiele
AWS -Tools für PowerShell	AWS -Tools für PowerShell Code-Beispiele
AWS SDK für Python (Boto3)	AWS SDK für Python (Boto3) Code-Beispiele
AWS SDK für Ruby	AWS SDK für Ruby Code-Beispiele
AWS SDK für Rust	AWS SDK für Rust Code-Beispiele
AWS SDK für SAP ABAP	AWS SDK für SAP ABAP Code-Beispiele
AWS SDK für Swift	AWS SDK für Swift Code-Beispiele

Weitere Beispiele speziell für diesen Service finden Sie unter Codebeispiele für die MediaStore Verwendung AWS SDKs.

Beispiel für die Verfügbarkeit

Sie können nicht finden, was Sie brauchen? Fordern Sie ein Codebeispiel an, indem Sie unten den Link Provide feedback (Feedback geben) auswählen.

Codebeispiele für die MediaStore Verwendung AWS SDKs

Die folgenden Codebeispiele zeigen die Verwendung MediaStore mit einem AWS Software Development Kit (SDK).

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Service-Funktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarios anzeigen.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele

- Grundlegende Beispiele für die MediaStore Verwendung AWS SDKs
  - Aktionen zur MediaStore Verwendung AWS SDKs
    - Verwendung CreateContainer mit einem AWS SDK oder CLI
    - Verwendung DeleteContainer mit einem AWS SDK oder CLI
    - DeleteObjectMit einem AWS SDK verwenden
    - Verwendung DescribeContainer mit einem AWS SDK oder CLI
    - Verwendung GetObject mit einem AWS SDK oder CLI
    - Verwendung ListContainers mit einem AWS SDK oder CLI
    - Verwendung PutObject mit einem AWS SDK oder CLI

Grundlegende Beispiele für die MediaStore Verwendung AWS SDKs

Die folgenden Codebeispiele zeigen, wie die Grundlagen von AWS Elemental MediaStore with verwendet AWS SDKs werden.

Beispiele

- Aktionen zur MediaStore Verwendung AWS SDKs
  - Verwendung CreateContainer mit einem AWS SDK oder CLI
  - Verwendung DeleteContainer mit einem AWS SDK oder CLI

- DeleteObjectMit einem AWS SDK verwenden
- Verwendung DescribeContainer mit einem AWS SDK oder CLI
- Verwendung GetObject mit einem AWS SDK oder CLI
- Verwendung ListContainers mit einem AWS SDK oder CLI
- Verwendung PutObject mit einem AWS SDK oder CLI

Aktionen zur MediaStore Verwendung AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie einzelne MediaStore Aktionen mit ausführen AWS SDKs. 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 AWS Elemental MediaStore -API-Referenz.

Beispiele

- Verwendung CreateContainer mit einem AWS SDK oder CLI
- Verwendung DeleteContainer mit einem AWS SDK oder CLI
- DeleteObjectMit einem AWS SDK verwenden
- Verwendung DescribeContainer mit einem AWS SDK oder CLI
- Verwendung GetObject mit einem AWS SDK oder CLI
- Verwendung ListContainers mit einem AWS SDK oder CLI
- Verwendung PutObject mit einem AWS SDK oder CLI

Verwendung **`CreateContainer`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `CreateContainer` verwendet wird.

CLI

AWS CLI

Um einen Container zu erstellen

Im folgenden `create-container` Beispiel wird ein neuer, leerer Container erstellt.

```
aws mediastore create-container --container-name ExampleContainer
```

Ausgabe:

```
{
    "Container": {
        "AccessLoggingEnabled": false,
        "CreationTime": 1563557265,
        "Name": "ExampleContainer",
        "Status": "CREATING",
        "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleContainer"
    }
}
```

Weitere Informationen finden Sie unter Erstellen eines Containers im AWS Elemental MediaStore User Guide.

- Einzelheiten zur API finden Sie unter CreateContainer AWS CLIBefehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.services.mediastore.MediaStoreClient;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.mediastore.model.CreateContainerRequest;
import software.amazon.awssdk.services.mediastore.model.CreateContainerResponse;
import software.amazon.awssdk.services.mediastore.model.MediaStoreException;

/**
 * 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 CreateContainer {
    public static long sleepTime = 10;

    public static void main(String[] args) {
        final String usage = """

                Usage:    <containerName>

                Where:
                   containerName - The name of the container to create.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String containerName = args[0];
        Region region = Region.US_EAST_1;
        MediaStoreClient mediaStoreClient = MediaStoreClient.builder()
                .region(region)
                .build();

        createMediaContainer(mediaStoreClient, containerName);
        mediaStoreClient.close();
    }


    public static void createMediaContainer(MediaStoreClient mediaStoreClient,
 String containerName) {
        try {
            CreateContainerRequest containerRequest =
 CreateContainerRequest.builder()
                    .containerName(containerName)
                    .build();

            CreateContainerResponse containerResponse =
 mediaStoreClient.createContainer(containerRequest);
            String status = containerResponse.container().status().toString();
            while (!status.equalsIgnoreCase("Active")) {
```

```
                status = DescribeContainer.checkContainer(mediaStoreClient,
 containerName);
                System.out.println("Status - " + status);
                Thread.sleep(sleepTime * 1000);
            }

            System.out.println("The container ARN value is " +
 containerResponse.container().arn());
            System.out.println("Finished ");

        } catch (MediaStoreException | InterruptedException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie CreateContainerin der AWS SDK for Java 2.x API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteContainer`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `DeleteContainer` verwendet wird.

CLI

AWS CLI

Um einen Container zu löschen

Im folgenden `delete-container` Beispiel wird der angegebene Container gelöscht. Sie können einen Container nur löschen, wenn er keine Objekte enthält.

```
aws mediastore delete-container \
    --container-name=ExampleLiveDemo
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Löschen eines Containers im AWS Elemental MediaStore User Guide.

- Einzelheiten zur API finden Sie unter DeleteContainer AWS CLIBefehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.services.mediastore.MediaStoreClient;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.mediastore.model.CreateContainerRequest;
import software.amazon.awssdk.services.mediastore.model.CreateContainerResponse;
import software.amazon.awssdk.services.mediastore.model.MediaStoreException;

/**
 * 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 CreateContainer {
    public static long sleepTime = 10;

    public static void main(String[] args) {
        final String usage = """

                Usage:    <containerName>

                Where:
                   containerName - The name of the container to create.
                """;
```

```
        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String containerName = args[0];
        Region region = Region.US_EAST_1;
        MediaStoreClient mediaStoreClient = MediaStoreClient.builder()
                .region(region)
                .build();

        createMediaContainer(mediaStoreClient, containerName);
        mediaStoreClient.close();
    }


    public static void createMediaContainer(MediaStoreClient mediaStoreClient,
 String containerName) {
        try {
            CreateContainerRequest containerRequest =
 CreateContainerRequest.builder()
                    .containerName(containerName)
                    .build();

            CreateContainerResponse containerResponse =
 mediaStoreClient.createContainer(containerRequest);
            String status = containerResponse.container().status().toString();
            while (!status.equalsIgnoreCase("Active")) {
                status = DescribeContainer.checkContainer(mediaStoreClient,
 containerName);
                System.out.println("Status - " + status);
                Thread.sleep(sleepTime * 1000);
            }

            System.out.println("The container ARN value is " +
 containerResponse.container().arn());
            System.out.println("Finished ");

        } catch (MediaStoreException | InterruptedException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie DeleteContainerin der AWS SDK for Java 2.x API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

`DeleteObject`Mit einem AWS SDK verwenden

Das folgende Codebeispiel zeigt, wie es verwendet wird`DeleteObject`.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.mediastore.MediaStoreClient;
import software.amazon.awssdk.services.mediastore.model.DescribeContainerRequest;
import
 software.amazon.awssdk.services.mediastore.model.DescribeContainerResponse;
import software.amazon.awssdk.services.mediastoredata.MediaStoreDataClient;
import software.amazon.awssdk.services.mediastoredata.model.DeleteObjectRequest;
import
 software.amazon.awssdk.services.mediastoredata.model.MediaStoreDataException;
import java.net.URI;
import java.net.URISyntaxException;

/**
 * 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 DeleteObject {
    public static void main(String[] args) throws URISyntaxException {
        final String usage = """

                Usage:    <completePath> <containerName>

                Where:
                   completePath - The path (including the container) of the item
 to delete.
                   containerName - The name of the container.
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String completePath = args[0];
        String containerName = args[1];
        Region region = Region.US_EAST_1;
        URI uri = new URI(getEndpoint(containerName));

        MediaStoreDataClient mediaStoreData = MediaStoreDataClient.builder()
                .endpointOverride(uri)
                .region(region)
                .build();

        deleteMediaObject(mediaStoreData, completePath);
        mediaStoreData.close();
    }

    public static void deleteMediaObject(MediaStoreDataClient mediaStoreData,
 String completePath) {
        try {
            DeleteObjectRequest deleteObjectRequest =
 DeleteObjectRequest.builder()
                    .path(completePath)
                    .build();

            mediaStoreData.deleteObject(deleteObjectRequest);
```

```
        } catch (MediaStoreDataException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    private static String getEndpoint(String containerName) {
        Region region = Region.US_EAST_1;
        MediaStoreClient mediaStoreClient = MediaStoreClient.builder()
                .region(region)
                .build();

        DescribeContainerRequest containerRequest =
 DescribeContainerRequest.builder()
                .containerName(containerName)
                .build();

        DescribeContainerResponse response =
 mediaStoreClient.describeContainer(containerRequest);
        mediaStoreClient.close();
        return response.container().endpoint();
    }
}
```

- Einzelheiten zur API finden Sie DeleteObjectin der AWS SDK for Java 2.x API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DescribeContainer`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `DescribeContainer` verwendet wird.

CLI

AWS CLI

Um die Details eines Containers anzuzeigen

Im folgenden `describe-container` Beispiel werden die Details des angegebenen Containers angezeigt.

```
aws mediastore describe-container \
    --container-name ExampleContainer
```

Ausgabe:

```
{
    "Container": {
        "CreationTime": 1563558086,
        "AccessLoggingEnabled": false,
        "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleContainer",
        "Status": "ACTIVE",
        "Name": "ExampleContainer",
        "Endpoint": "https://aaabbbcccdddee.data.mediastore.us-
west-2.amazonaws.com"
    }
}
```

Weitere Informationen finden Sie unter Anzeigen der Details für einen Container im AWS Elemental MediaStore User Guide.

- Einzelheiten zur API finden Sie unter DescribeContainer AWS CLIBefehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.mediastore.MediaStoreClient;
import software.amazon.awssdk.services.mediastore.model.DescribeContainerRequest;
import
 software.amazon.awssdk.services.mediastore.model.DescribeContainerResponse;
import software.amazon.awssdk.services.mediastore.model.MediaStoreException;

/**
```

```
 * 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 DescribeContainer {

    public static void main(String[] args) {
        final String usage = """

                Usage:    <containerName>

                Where:
                   containerName - The name of the container to describe.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String containerName = args[0];
        Region region = Region.US_EAST_1;
        MediaStoreClient mediaStoreClient = MediaStoreClient.builder()
                .region(region)
                .build();

        System.out.println("Status is " + checkContainer(mediaStoreClient,
 containerName));
        mediaStoreClient.close();
    }

    public static String checkContainer(MediaStoreClient mediaStoreClient, String
 containerName) {
        try {
            DescribeContainerRequest describeContainerRequest =
 DescribeContainerRequest.builder()
                    .containerName(containerName)
                    .build();
```

```
            DescribeContainerResponse containerResponse =
 mediaStoreClient.describeContainer(describeContainerRequest);
            System.out.println("The container name is " +
 containerResponse.container().name());
            System.out.println("The container ARN is " +
 containerResponse.container().arn());
            return containerResponse.container().status().toString();

        } catch (MediaStoreException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Einzelheiten zur API finden Sie DescribeContainerin der AWS SDK for Java 2.x API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetObject`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `GetObject` verwendet wird.

CLI

AWS CLI

Um ein Objekt herunterzuladen

Im folgenden `get-object` Beispiel wird ein Objekt auf den angegebenen Endpunkt heruntergeladen.

```
aws mediastore-data get-object \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --path=/folder_name/README.md README.md
```

Ausgabe:

```
{
    "ContentLength": "2307346",
    "ContentType": "image/jpeg",
    "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3",
    "StatusCode": 200
}
```

Um einen Teil eines Objekts herunterzuladen

Im folgenden `get-object` Beispiel wird ein Teil eines Objekts auf den angegebenen Endpunkt heruntergeladen.

```
aws mediastore-data get-object \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --path /folder_name/README.md \
    --range="bytes=0-100" README2.md
```

Ausgabe:

```
{
    "StatusCode": 206,
    "ContentRange": "bytes 0-100/2307346",
    "ContentLength": "101",
    "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",
    "ContentType": "image/jpeg",
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3"
}
```

Weitere Informationen finden Sie im AWS Elemental MediaStore User Guide unter Ein Objekt herunterladen.

- Einzelheiten zur API finden Sie GetObjectin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.core.ResponseInputStream;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.mediastore.MediaStoreClient;
import software.amazon.awssdk.services.mediastore.model.DescribeContainerRequest;
import
 software.amazon.awssdk.services.mediastore.model.DescribeContainerResponse;
import software.amazon.awssdk.services.mediastoredata.MediaStoreDataClient;
import software.amazon.awssdk.services.mediastoredata.model.GetObjectRequest;
import software.amazon.awssdk.services.mediastoredata.model.GetObjectResponse;
import
 software.amazon.awssdk.services.mediastoredata.model.MediaStoreDataException;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.net.URI;
import java.net.URISyntaxException;

/**
 * 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 GetObject {
    public static void main(String[] args) throws URISyntaxException {
        final String usage = """

                Usage:    <completePath> <containerName> <savePath>
```

```
                Where:
                   completePath - The path of the object in the container (for
example, Videos5/sampleVideo.mp4).
                   containerName - The name of the container.
                   savePath - The path on the local drive where the file is
saved, including the file name (for example, C:/AWS/myvid.mp4).
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String completePath = args[0];
        String containerName = args[1];
        String savePath = args[2];

        Region region = Region.US_EAST_1;
        URI uri = new URI(getEndpoint(containerName));
        MediaStoreDataClient mediaStoreData = MediaStoreDataClient.builder()
                .endpointOverride(uri)
                .region(region)
                .build();

        getMediaObject(mediaStoreData, completePath, savePath);
        mediaStoreData.close();
    }

    public static void getMediaObject(MediaStoreDataClient mediaStoreData, String
completePath, String savePath) {

        try {
            GetObjectRequest objectRequest = GetObjectRequest.builder()
                    .path(completePath)
                    .build();

            // Write out the data to a file.
            ResponseInputStream<GetObjectResponse> data =
mediaStoreData.getObject(objectRequest);
            byte[] buffer = new byte[data.available()];
            data.read(buffer);

            File targetFile = new File(savePath);
```

```
            OutputStream outStream = new FileOutputStream(targetFile);
            outStream.write(buffer);
            System.out.println("The data was written to " + savePath);

        } catch (MediaStoreDataException | IOException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    private static String getEndpoint(String containerName) {
        Region region = Region.US_EAST_1;
        MediaStoreClient mediaStoreClient = MediaStoreClient.builder()
                .region(region)
                .build();

        DescribeContainerRequest containerRequest =
 DescribeContainerRequest.builder()
                .containerName(containerName)
                .build();

        DescribeContainerResponse response =
 mediaStoreClient.describeContainer(containerRequest);
        return response.container().endpoint();
    }
}
```

- Einzelheiten zur API finden Sie GetObjectin der AWS SDK for Java 2.x API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListContainers`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `ListContainers` verwendet wird.

CLI

AWS CLI

Um eine Liste von Containern anzuzeigen

Im folgenden `list-containers` Beispiel wird eine Liste aller Container angezeigt, die Ihrem Konto zugeordnet sind.

```
aws mediastore list-containers
```

Ausgabe:

```
{
    "Containers": [
        {
            "CreationTime": 1505317931,
            "Endpoint": "https://aaabbbcccdddee.data.mediastore.us-
west-2.amazonaws.com",
            "Status": "ACTIVE",
            "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleLiveDemo",
            "AccessLoggingEnabled": false,
            "Name": "ExampleLiveDemo"
        },
        {
            "CreationTime": 1506528818,
            "Endpoint": "https://fffggghhhiiijj.data.mediastore.us-
west-2.amazonaws.com",
            "Status": "ACTIVE",
            "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleContainer",
            "AccessLoggingEnabled": false,
            "Name": "ExampleContainer"
        }
    ]
}
```

Weitere Informationen finden Sie im AWS Elemental MediaStore User Guide unter Eine Liste von Containern anzeigen.

- Einzelheiten zur API finden Sie unter ListContainers AWS CLIBefehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.mediastore.MediaStoreClient;
import software.amazon.awssdk.services.mediastore.model.Container;
import software.amazon.awssdk.services.mediastore.model.ListContainersResponse;
import software.amazon.awssdk.services.mediastore.model.MediaStoreException;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListContainers {

    public static void main(String[] args) {

        Region region = Region.US_EAST_1;
        MediaStoreClient mediaStoreClient = MediaStoreClient.builder()
                .region(region)
                .build();

        listAllContainers(mediaStoreClient);
        mediaStoreClient.close();
    }

    public static void listAllContainers(MediaStoreClient mediaStoreClient) {
        try {
```

```
            ListContainersResponse containersResponse =
 mediaStoreClient.listContainers();
            List<Container> containers = containersResponse.containers();
            for (Container container : containers) {
                System.out.println("Container name is " + container.name());
            }

        } catch (MediaStoreException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie ListContainersin der AWS SDK for Java 2.x API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`PutObject`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `PutObject` verwendet wird.

CLI

AWS CLI

Um ein Objekt hochzuladen

Im folgenden `put-object` Beispiel wird ein Objekt in den angegebenen Container hochgeladen. Sie können einen Ordnerpfad angeben, in dem das Objekt innerhalb des Containers gespeichert wird. Wenn der Ordner bereits existiert, MediaStore speichert AWS Elemental das Objekt im Ordner. Wenn der Ordner nicht existiert, erstellt der Dienst ihn und speichert das Objekt dann im Ordner.

```
aws mediastore-data put-object \
    --endpoint https://aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com \
    --body README.md \
    --path /folder_name/README.md \
    --cache-control "max-age=6, public" \
```

```
    --content-type binary/octet-stream
```

Ausgabe:

```
{
    "ContentSHA256":
 "74b5fdb517f423ed750ef214c44adfe2be36e37d861eafe9c842cbe1bf387a9d",
    "StorageClass": "TEMPORAL",
    "ETag": "af3e4731af032167a106015d1f2fe934e68b32ed1aa297a9e325f5c64979277b"
}
```

Weitere Informationen finden Sie unter Hochladen eines Objekts im AWS Elemental MediaStore User Guide.

- Einzelheiten zur API finden Sie PutObjectin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.mediastore.MediaStoreClient;
import software.amazon.awssdk.services.mediastoredata.MediaStoreDataClient;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.mediastoredata.model.PutObjectRequest;
import
 software.amazon.awssdk.services.mediastoredata.model.MediaStoreDataException;
import software.amazon.awssdk.services.mediastoredata.model.PutObjectResponse;
import software.amazon.awssdk.services.mediastore.model.DescribeContainerRequest;
import
 software.amazon.awssdk.services.mediastore.model.DescribeContainerResponse;
import java.io.File;
import java.net.URI;
import java.net.URISyntaxException;
```

```
/**
 * 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 PutObject {
    public static void main(String[] args) throws URISyntaxException {
        final String USAGE = """

                To run this example, supply the name of a container, a file
 location to use, and path in the container\s

                Ex: <containerName> <filePath> <completePath>
                """;

        if (args.length < 3) {
            System.out.println(USAGE);
            System.exit(1);
        }

        String containerName = args[0];
        String filePath = args[1];
        String completePath = args[2];

        Region region = Region.US_EAST_1;
        URI uri = new URI(getEndpoint(containerName));
        MediaStoreDataClient mediaStoreData = MediaStoreDataClient.builder()
                .endpointOverride(uri)
                .region(region)
                .build();

        putMediaObject(mediaStoreData, filePath, completePath);
        mediaStoreData.close();
    }

    public static void putMediaObject(MediaStoreDataClient mediaStoreData, String
 filePath, String completePath) {
        try {
            File myFile = new File(filePath);
            RequestBody requestBody = RequestBody.fromFile(myFile);
```

```
            PutObjectRequest objectRequest = PutObjectRequest.builder()
                    .path(completePath)
                    .contentType("video/mp4")
                    .build();

            PutObjectResponse response = mediaStoreData.putObject(objectRequest,
 requestBody);
            System.out.println("The saved object is " +
 response.storageClass().toString());

        } catch (MediaStoreDataException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static String getEndpoint(String containerName) {

        Region region = Region.US_EAST_1;
        MediaStoreClient mediaStoreClient = MediaStoreClient.builder()
                .region(region)
                .build();

        DescribeContainerRequest containerRequest =
 DescribeContainerRequest.builder()
                .containerName(containerName)
                .build();

        DescribeContainerResponse response =
 mediaStoreClient.describeContainer(containerRequest);
        return response.container().endpoint();
    }
}
```

- Einzelheiten zur API finden Sie PutObjectin der AWS SDK for Java 2.x API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Service mit einem SDK AWS. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Kontingente in AWS Elemental MediaStore

Die Service Quotas Quotas-Konsole bietet Informationen zu AWS Elemental MediaStore Elemental-Kontingenten. Neben der Anzeige der Standardkontingente können Sie die Servicekontingentkonsole verwenden, um Kontingenterhöhungen für einstellbare Kontingente anzufordern.

In der folgenden Tabelle werden Kontingente, früher als Limits bezeichnet, in AWS Elemental MediaStore beschrieben. Kontingente sind die maximale Anzahl von Serviceressourcen oder Vorgängen für Ihr AWS-Konto.

> **Note**
>
> Um einzelnen Containern in Ihrem Konto Kontingente zuzuweisen, wenden Sie sich an den AWS-Support oder Ihren Kundenbetreuer. Diese Option kann Ihnen helfen, die Limits auf Kontoebene auf Ihre Container aufzuteilen, um zu verhindern, dass ein Container Ihr gesamtes Kontingent verbraucht.

Ressource oder Operation	Standardkontingent	Kommentare
Container	100	Die maximale Anzahl der Container, die Sie in diesem Konto erstellen können.
Ordnerebenen	10	Die maximale Anzahl der Ordnerebenen, die Sie in einem Container erstellen können. Sie können eine beliebige Anzahl von Ordnern erstellen, solange sie nicht mehr als 10 Ebenen innerhalb eines Containers verschachtelt sind.
Ordner	Unbegrenzt	Sie können eine beliebige Anzahl von Ordnern erstellen, solange sie nicht mehr als 10 Ebenen innerhalb eines Containers verschachtelt sind.
Objektgröße	25 MB	Die maximale Dateigröße eines einzelnen Objekts.

Ressource oder Operation	Standardkontingent	Kommentare
Objekte	Unbegrenzt	Sie können beliebig viele Objekte in einen Ordner oder Container in Ihrem Konto hochladen.
Rate of DeleteObject API requests (Rate der API-Anforderungen)	100	Die maximale Anzahl von Operationsanforderungen, die Sie pro Sekunde machen können. Darüber hinausgehende Anfragen werden gedrosselt. Sie können eine Kontingenterhöhung beantragen.
Rate of DescribeObject API requests (Rate der API-Anforderungen)	1.000	Die maximale Anzahl von Operationsanforderungen, die Sie pro Sekunde machen können. Darüber hinausgehende Anfragen werden gedrosselt. Sie können eine Kontingenterhöhung beantragen.
Rate der GetObject API-Anfragen für die Verfügbarkeit von Standard-Uploads	1.000	Die maximale Anzahl von Operationsanforderungen, die Sie pro Sekunde machen können. Darüber hinausgehende Anfragen werden gedrosselt. Sie können eine Kontingenterhöhung beantragen.
Rate der GetObject API-Anfragen für die Verfügbarkeit von Streaming-Uploads	25	Die maximale Anzahl von Operationsanforderungen, die Sie pro Sekunde machen können. Darüber hinausgehende Anfragen werden gedrosselt. Sie können eine Kontingenterhöhung beantragen.

Ressource oder Operation	Standardkontingent	Kommentare
Rate of ListItems API requests (Rate der API-Anfor derungen)	5	Die maximale Anzahl von Operationsanforder ungen, die Sie pro Sekunde machen können. Darüber hinausgehende Anfragen werden gedrosselt. Sie können eine Kontingenterhöhung beantragen.
Rate der PutObject API-Anfragen für die Verschlüsselung von Chunked-T ransfers (auch bekannt als Verfügbarkeit von Streaming-Uploads)	10	Die maximale Anzahl von Operationsanforder ungen, die Sie pro Sekunde machen können. Darüber hinausgehende Anfragen werden gedrosselt. Sie können eine Kontingenterhöhung beantrage n. Geben Sie in der Anforderung die angeforderte TPS und die durchschnittliche Objektgröße an.
Rate der PutObject API-Anfragen für die Verfügbarkeit von Standard-Uploads	100	Die maximale Anzahl von Operationsanforder ungen, die Sie pro Sekunde machen können. Darüber hinausgehende Anfragen werden gedrosselt. Sie können eine Kontingenterhöhung beantrage n. Geben Sie in der Anforderung die angeforderte TPS und die durchschnittliche Objektgröße an.
Regeln in einer Metrikrichtlinie	10	Die maximale Anzahl von Regeln, die Sie in eine Metrikrichtlinie aufnehmen können.
Regeln in einer Objektlebenszyklus -Richtlinie	10	Die maximale Anzahl der Regeln, die in einer Objektlebenszyklus-Richtlinie enthalten sein können.

MediaStore Verwandte Informationen zu AWS Elemental

In der folgenden Tabelle sind verwandte Ressourcen aufgeführt, die Ihnen bei der Arbeit mit AWS Elemental MediaStore nützlich sein werden.

- Kurse und Workshops — Links zu rollen- und Spezialkursen sowie zu Übungen zum Selbststudium, mit denen Sie Ihre AWS Fähigkeiten verbessern und praktische Erfahrungen sammeln können.
- AWS Developer Center — Erkunden Sie Tutorials, laden Sie Tools herunter und erfahren Sie mehr über Veranstaltungen für Entwickler. AWS
- AWS Entwicklertools — Links zu Entwicklertools SDKs, IDE-Toolkits und Befehlszeilentools für die Entwicklung und Verwaltung von AWS Anwendungen.
- Ressourcencenter für die ersten Schritte — Erfahren Sie AWS-Konto, wie Sie Ihre erste Anwendung einrichten, der AWS Community beitreten und sie starten.
- Praktische Tutorials — Folgen Sie den step-by-step Tutorials, um Ihre erste Anwendung zu starten. AWS
- AWS Whitepapers — Links zu einer umfassenden Liste von technischen AWS Whitepapers zu Themen wie Architektur, Sicherheit und Wirtschaft, die von Solutions Architects oder anderen technischen Experten verfasst wurden. AWS
- AWS Support Center — Die zentrale Anlaufstelle für die Erstellung und Verwaltung Ihrer Fälle. AWS Support Enthält auch Links zu anderen hilfreichen Ressourcen wie Foren, technischen FAQs Informationen, Servicestatus und AWS Trusted Advisor.
- Support— Die wichtigste Webseite mit Informationen über Support einen Support-Kanal mit schnellen Reaktionszeiten one-on-one, der Sie bei der Entwicklung und Ausführung von Anwendungen in der Cloud unterstützt.
- Kontakt – Zentraler Kontaktpunkt für Fragen zu AWS -Abrechnung, Konten, Ereignissen Missbrauch und anderen Problemen.
- AWS Nutzungsbedingungen der Website — Detaillierte Informationen zu unseren Urheberrechten und Marken, zu Ihrem Konto, Ihrer Lizenz und Ihrem Zugriff auf die Website sowie zu anderen Themen.

Dokumentverlauf für das Benutzerhandbuch

In der folgenden Tabelle wird die Dokumentation für diese Version von AWS Elemental MediaStore beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

Änderung	Beschreibung	Datum
Ende des Support-Hinweises	Hinweis zum Ende des Supports: Am 13. November 2025 AWS wird der Support für AWS Elemental MediaStore eingestellt. Nach dem 13. November 2025 können Sie nicht mehr auf die MediaStore Konsole oder MediaStore die Ressourcen zugreifen. Weitere Informationen finden Sie in diesem Blogbeitrag.	12. November 2024
Verbesserung von Origin Access Control (OAC)	Es wurden Informationen zur Verwendung von OAC mit AWS Elemental MediaStore hinzugefügt.	17. April 2023
Aktualisierungen der Kontingente	Der Kontingentwert und die Beschreibung für wurden korrigiert`Rules in a Metric Policy`.	25. Oktober 2022
ExpiresAt field	Die Zugriffsprotokolle enthalten jetzt ein `ExpiresAt` Feld, das das Ablaufdatum und die Uhrzeit des Objekts angibt, basierend auf den Regeln für transiente Daten in	16. Juli 2020

	der Lebenszyklusrichtlinie des Containers.	
Regeln für den Übergang zum Lebenszyklus	Sie können nun eine Lebenszyklus-Übergangsregel zu Ihrer Objektlebenszyklus -Richtlinie hinzufügen, die festlegt, dass Objekte nach Erreichen eines bestimmten Alters in die Speicherklasse für seltenen Zugriff verschoben werden.	20. April 2020
Leerer Behälter	Sie können nun alle Objekte innerhalb eines Containers gleichzeitig löschen.	7. April 2020
Support für CloudWatch Amazon-Metriken	Sie können eine Metrikric htlinie festlegen, um festzuleg en, an welche Metriken MediaStore gesendet werden CloudWatch.	30. März 2020
Platzhalter in den Regeln zum Löschen von Objekten	In einer Richtlinie zum Objektlebenszyklus können Sie nun einen Platzhalter in einer Regel zum Löschen von Objekten verwenden. Damit können Sie Dateien anhand ihres Dateinamens oder ihrer Erweiterung angeben, die der Service nach einer bestimmte n Anzahl von Tagen löschen soll.	20. Dezember 2019

Richtlinien für den Lebenszyklus von Objekten	Sie können Ihrer Objektlebenszyklus-Richtlinie jetzt eine Regel hinzufügen, die einen Ablauf nach Alter in Sekunden anzeigt.	13. September 2019
CloudFormation Unterstützung	Sie können jetzt eine CloudFormation Vorlage verwenden, um einen Container automatisch zu erstellen. Die CloudFormation -Vorlage verwaltet Daten für fünf API-Aktionen: Erstellung eines Containers, Einstellung der Zugriffsprotokollierung, Aktualisierung der Standard-Containerrichtlinie, Hinzufügen einer CORS-Richtlinie (Cross-Origin Resource Sharing) und Hinzufügen einer Objektlebenszyklusrichtlinie.	17. Mai 2019
Kontingente für die Verfügbarkeit von Streaming-Uploads	Bei Objekten mit Streaming -Upload-Verfügbarkeit (gestückelte Übertragung von Objekten) darf die `PutObject` -Operation 10 TPS und die `GetObject` -Operation 25 TPS nicht überschreiten.	8. April 2019

Geteilte Übertragung von Objekten	Zusätzliche Unterstützung für das Aufteilen der Übertragung von Objekten. Mit dieser Funktion können Sie angeben, dass ein Objekt zum Herunterladen verfügbar ist, bevor das Objekt vollständig hochgeladen ist.	5. April 2019
Protokollierung des Zugriffs	AWS Elemental unterstützt MediaStore jetzt die Zugriffsprotokollierung, die detaillierte Aufzeichnungen für die Anfragen bereitstellt, die an Objekte in einem Container gestellt werden.	25. Februar 2019
Richtlinien für den Lebenszyklus von Objekten	Zusätzliche Unterstützung für Objektlebenszyklus-Richtlinien, die das Ablaufdatum von Objekten innerhalb des aktuellen Containers regeln.	12. Dezember 2018
Höheres Kontingent für Objektgrößen	Das Kontingent für die Größe eines Objekts beträgt nun 25 MB.	10. Oktober 2018
Höheres Objektgrößenkontingent	Das Kontingent für die Größe eines Objekts beträgt nun 20 MB.	6. September 2018
AWS CloudTrail Integration	Der Inhalt der CloudTrail Integration wurde aktualisiert, um ihn den jüngsten Änderungen am CloudTrail Service anzupassen.	12. Juli 2018

CDN-Zusammenarbeit	Es wurden Informationen zur Verwendung von AWS Elemental MediaStore mit einem Content Delivery Network (CDN) wie Amazon hinzugefügt. CloudFront	14. April 2018
CORS-Konfigurationen	AWS Elemental unterstützt MediaStore jetzt Cross-Origin Resource Sharing (CORS), wodurch Client-Webanwendungen, die in einer Domain geladen sind, mit Ressourcen in einer anderen Domain interagieren können.	7. Februar 2018
Neuer Dienst mit dazugehörigem Handbuch	Dies ist die erste Version des Videoerstellungs- und Speicherservices AWS Elemental MediaStore und des AWS Elemental User Guide MediaStore .	27. November 2017

Note

- Die AWS Media Services sind nicht für die Verwendung mit Anwendungen oder in Situationen konzipiert oder vorgesehen, in denen eine ausfallsichere Leistung erforderlich ist, wie z. B. für den Betrieb von Menschen, Navigations- oder Kommunikationssystemen, Flugsicherungs- oder lebenserhaltenden Maschinen, in denen die Nichtverfügbarkeit, Unterbrechung oder der Ausfall der Dienste zu Tod, Personen-, Sach- oder Umweltschäden führen könnte.

AWS Glossar

Die neueste AWS Terminologie finden Sie im AWS Glossar in der AWS-Glossar Referenz.