aws

Panduan Pengguna

# AWS Elemental MediaStore

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

# AWS Elemental MediaStore: Panduan Pengguna

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara pelanggan, atau dengan cara apa pun yang merendahkan atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan hak milik masing-masing pemiliknya, yang mungkin atau tidak terafiliasi, terkait dengan, atau disponsori oleh Amazon.

# Table of Contents


..... vi
Apa itu MediaStore? ..... 1
Konsep dan terminologi ..... 1
Layanan terkait ..... 3
Mengakses MediaStore ..... 3
Harga ..... 4
Wilayah dan titik akhir ..... 4
Menyiapkan AWS Elemental MediaStore ..... 6
Mendaftar untuk Akun AWS ..... 6
Buat pengguna dengan akses administratif ..... 7
Memulai ..... 9
Langkah 1: Akses AWS Elemental MediaStore ..... 9
Langkah 2: Buat wadah ..... 9
Langkah 3: Unggah objek ..... 10
Langkah 4: Akses objek ..... 10
Kontainer ..... 12
Aturan untuk nama kontainer ..... 12
Membuat wadah ..... 12
Melihat detail kontainer ..... 14
Melihat daftar kontainer ..... 15
Menghapus wadah ..... 16
Kebijakan ..... 17
Kebijakan kontainer ..... 17
Melihat kebijakan kontainer ..... 18
Mengedit kebijakan kontainer ..... 19
Contoh kebijakan kontainer ..... 20
Kebijakan CORS ..... 27
Skenario kasus penggunaan ..... 27
Menambahkan kebijakan CORS ..... 28
Melihat kebijakan CORS ..... 29
Mengedit kebijakan CORS ..... 30
Menghapus kebijakan CORS ..... 31
Pemecahan Masalah ..... 32
Contoh kebijakan CORS ..... 33

Kebijakan siklus hidup objek ..... 34
Komponen kebijakan siklus hidup objek ..... 35
Menambahkan kebijakan siklus hidup objek ..... 41
Melihat kebijakan siklus hidup objek ..... 43
Mengedit kebijakan siklus hidup objek ..... 44
Menghapus kebijakan siklus hidup objek ..... 45
Contoh kebijakan siklus hidup objek ..... 46
Kebijakan metrik ..... 50
Menambahkan kebijakan metrik ..... 51
Melihat kebijakan metrik ..... 51
Mengedit kebijakan metrik ..... 52
Contoh kebijakan metrik ..... 52
Folder ..... 56
Aturan untuk nama folder ..... 56
Membuat folder ..... 57
Menghapus folder ..... 57
Objek ..... 58
Mengunggah Objek ..... 58
Melihat daftar ..... 60
Melihat detail objek ..... 62
Mengunduh objek ..... 63
Menghapus objek ..... 65
Menghapus satu objek ..... 65
Mengosongkan wadah ..... 66
Keamanan ..... 67
Perlindungan data ..... 68
Enkripsi data ..... 69
Identity and Access Management ..... 69
Audiens ..... 70
Mengautentikasi dengan identitas ..... 70
Mengelola akses menggunakan kebijakan ..... 74
Bagaimana AWS Elemental MediaStore bekerja dengan IAM ..... 77
Contoh kebijakan berbasis identitas ..... 84
Pemecahan Masalah ..... 87
Pencatatan log dan pemantauan ..... 89
CloudWatch Alarm Amazon ..... 89

AWS CloudTrail log ..... 90
AWS Trusted Advisor ..... 90
Validasi kepatuhan ..... 90
Ketahanan ..... 91
Keamanan Infrastruktur ..... 92
Pencegahan "confused deputy" lintas layanan ..... 92
Pemantauan dan penandaan ..... 95
Membuat log panggilan API dengan CloudTrail ..... 96
MediaStoreInformasi di CloudTrail ..... 96
Contoh: Entri berkas log ..... 98
Pemantauan CloudWatch dengan ..... 99
CloudWatch Log ..... 100
CloudWatch Acara ..... 110
Metrik-metrik CloudWatch ..... 113
Penandaan ..... 118
Sumber daya yang didukung di AWS Elemental MediaStore ..... 119
Kesepakatan penamaan dan penggunaan tag ..... 119
Mengelola tag ..... 119
Bekerja dengan CDNs ..... 121
Memungkinkan CloudFront untuk mengakses wadah Anda ..... 121
Menggunakan Origin Access Control (OAC) ..... 122
Menggunakan Rahasia Bersama ..... 122
MediaStoreInteraksi dengan cache HTTP ..... 125
Permintaan bersyarat ..... 125
Bekerja dengan AWS SDKs ..... 127
Contoh kode ..... 129
Hal-hal mendasar ..... 129
Tindakan ..... 130
Kuota ..... 152
Informasi terkait ..... 155
Riwayat dokumen ..... 156
AWS Glosarium ..... 161

Pemberitahuan akhir dukungan: Pada 13 November 2025, AWS akan menghentikan dukungan untuk AWS Elemental. MediaStore Setelah 13 November 2025, Anda tidak akan lagi dapat mengakses MediaStore konsol atau MediaStore sumber daya. Untuk informasi lebih lanjut, kunjungi posting blog ini.

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.

# Apa itu AWS Elemental? MediaStore

AWS Elemental MediaStore adalah layanan originasi dan penyimpanan video yang menawarkan kinerja tinggi dan konsistensi langsung yang diperlukan untuk originasi langsung. Dengan MediaStore, Anda dapat mengelola aset video sebagai objek dalam wadah untuk membangun alur kerja media berbasis cloud yang dapat diandalkan.

Untuk menggunakan layanan, Anda mengunggah objek dari sumber, seperti encoder atau umpan data, ke wadah yang Anda buat. MediaStore

MediaStore adalah pilihan tepat untuk menyimpan file video yang terfragmentasi saat Anda membutuhkan konsistensi yang kuat, membaca dan menulis latensi rendah, dan kemampuan untuk menangani permintaan bersamaan dengan volume tinggi. Jika Anda tidak mengirimkan video streaming langsung, pertimbangkan untuk menggunakan Amazon Simple Storage Service (Amazon S3).

Topik


- MediaStore Konsep dan terminologi AWS
- Layanan terkait
- Mengakses AWS Elemental MediaStore
- Harga untuk AWS Elemental MediaStore
- Wilayah dan titik akhir untuk AWS Elemental MediaStore


# MediaStore Konsep dan terminologi AWS

ARN

Nama Sumber Daya Amazon.

Tubuh

Data yang akan diunggah ke objek.

Rentang (Byte)

Sebuah subset dari data objek yang akan ditangani. Untuk informasi selengkapnya, lihat rentang dari spesifikasi HTTP.

Kontainer

Namespace yang menyimpan objek. Container memiliki titik akhir yang dapat Anda gunakan untuk menulis dan mengambil objek dan melampirkan kebijakan akses.

Titik Akhir

Titik masuk ke MediaStore layanan, diberikan sebagai URL root HTTPS.

ETag

Tag entitas, yang merupakan hash dari data objek.

Folder

Pembagian wadah. Folder dapat menyimpan objek dan folder lainnya.

Item

Istilah yang digunakan untuk merujuk ke objek dan folder.

Objek

Aset, mirip dengan objek Amazon S3. Objek adalah entitas dasar yang disimpan di MediaStore. Layanan menerima semua jenis file.

Layanan Asal

MediaStore dianggap sebagai layanan originasi karena merupakan titik distribusi untuk pengiriman konten media.

Jalur

Pengidentifikasi unik untuk objek atau folder, yang menunjukkan lokasinya di wadah.

Bagian

Sebuah subset dari data (chunk) dari suatu objek.

Kebijakan

Kebijakan IAM.

Sumber Daya

Sebuah entitas di AWS yang dapat Anda kerjakan. Setiap sumber daya AWS diberikan Amazon Resource Name (ARN) yang bertindak sebagai pengidentifikasi unik. Di MediaStore, ini adalah sumber daya dan format ARN nya:

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

## Layanan terkait

- Amazon CloudFront adalah layanan jaringan pengiriman konten (CDN) global yang mengirimkan data dan video secara aman ke pemirsa Anda. Gunakan CloudFront untuk menyampaikan konten dengan kinerja terbaik. Untuk informasi selengkapnya, lihat Panduan CloudFront Developer Amazon.
- AWS CloudFormationadalah layanan yang membantu Anda membuat model dan pengaturan AWS sumber daya Anda. Anda membuat templat yang menjelaskan semua AWS sumber daya yang Anda inginkan (seperti MediaStore kontainer), dan AWS CloudFormation mengurus penyediaan dan konfigurasi sumber daya tersebut untuk Anda. Anda tidak perlu membuat dan mengonfigurasi AWS sumber daya secara individual dan mencari tahu apa yang tergantung pada apa; AWS CloudFormation menangani semua itu. Untuk informasi selengkapnya, silakan lihat Panduan Pengguna AWS CloudFormation.
- AWS CloudTrailadalah layanan yang memungkinkan Anda memantau panggilan yang dilakukan ke CloudTrail API untuk akun Anda, termasuk panggilan yang dilakukan oleh Konsol Manajemen AWS AWS CLI, dan layanan lainnya. Untuk informasi selengkapnya, silakan lihat Panduan Pengguna AWS CloudTrail.
- Amazon CloudWatch adalah layanan pemantauan untuk sumber daya AWS Cloud dan aplikasi yang Anda jalankan AWS. Gunakan CloudWatch Peristiwa untuk melacak perubahan status kontainer dan objek di MediaStore. Untuk informasi selengkapnya, lihat CloudWatch dokumentasi Amazon.
- AWS Identity and Access Management (IAM) adalah layanan web yang membantu Anda mengontrol akses ke AWS sumber daya secara aman bagi pengguna Anda. Gunakan IAM untuk mengontrol orang yang dapat menggunakan AWS sumber daya Anda (autentikasi) dan sumber daya apa yang dapat digunakan pengguna dengan cara (otorisasi). Untuk informasi selengkapnya, lihat Menyiapkan AWS Elemental MediaStore.
- Amazon Simple Storage Service (Amazon S3) adalah penyimpanan objek yang dibangun untuk menyimpan dan mengambil berapa pun jumlah data dari mana saja. Untuk informasi lebih lanjut, lihat Dokumentasi Amazon S3.

## Mengakses AWS Elemental MediaStore

Anda dapat mengakses MediaStore menggunakan salah satu metode berikut:

- Konsol Manajemen AWS Prosedur di seluruh panduan ini menjelaskan cara menggunakan Konsol Manajemen AWS untuk melakukan tugas MediaStore. Untuk mengakses MediaStore menggunakan konsol:

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

- AWS Command Line Interface— Untuk informasi selengkapnya, lihat Panduan AWS Command Line Interface Pengguna. Untuk mengakses MediaStore menggunakan titik akhir CLI:

```
aws mediastore
```

- MediaStore API — Jika Anda menggunakan bahasa pemrograman yang tidak tersedia untuk SDK, lihat Referensi AWS Elemental MediaStore API untuk informasi tentang tindakan API dan cara membuat permintaan API. Untuk mengakses MediaStore menggunakan titik akhir REST API:

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

- AWS SDKs — Jika Anda menggunakan bahasa pemrograman yang disediakan AWS SDK, Anda dapat menggunakan SDK untuk mengakses. MediaStore SDKs menyederhanakan autentikasi, integrasikan dengan mudah dengan lingkungan pengembangan Anda, dan berikan akses mudah ke MediaStore perintah. Untuk informasi lebih lanjut, lihat Alat untuk Amazon Web Services.
- AWS Tools untuk Windows PowerShell — Untuk informasi selengkapnya, lihat Panduan AWS Tools for Windows PowerShell Pengguna.

## Harga untuk AWS Elemental MediaStore

Seperti AWS produk lainnya, tidak ada kontrak atau komitmen minimum untuk menggunakan MediaStore. Anda dikenakan biaya konsumsi per GB ketika konten masuk ke layanan dan biaya per GB bulanan untuk konten yang Anda simpan di layanan. Untuk informasi selengkapnya, lihat Harga Elemen MediaStore AWS.

## Wilayah dan titik akhir untuk AWS Elemental MediaStore

Untuk mengurangi latensi data dalam aplikasi Anda, MediaStore tawarkan titik akhir regional untuk membuat permintaan Anda:

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

Untuk melihat daftar lengkap Wilayah AWS jika MediaStore tersedia, lihat MediaStore titik akhir dan kuota AWS Elemental di Referensi Umum AWS.

# Menyiapkan AWS Elemental MediaStore

Bagian ini memandu Anda melalui langkah-langkah yang diperlukan untuk mengonfigurasi pengguna agar mengakses AWS Elemental MediaStore. Untuk latar belakang dan informasi tambahan tentang identitas dan manajemen akses MediaStore, lihatIdentity and Access Management untuk AWS Elemental MediaStore.

Untuk mulai menggunakan AWS Elemental MediaStore, selesaikan langkah-langkah berikut.

Topik

- Mendaftar untuk Akun AWS
- Buat pengguna dengan akses administratif

## Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

# Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

  Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

# Memulai AWS Elemental MediaStore

Tutorial Memulai ini menunjukkan kepada Anda cara menggunakan AWS Elemental MediaStore untuk membuat wadah dan mengunggah objek.

Topik


- Langkah 1: Akses AWS Elemental MediaStore
- Langkah 2: Buat wadah
- Langkah 3: Unggah objek
- Langkah 4: Akses objek


## Langkah 1: Akses AWS Elemental MediaStore

Setelah menyiapkan akun AWS dan membuat pengguna dan peran, Anda masuk ke konsol untuk AWS Elemental MediaStore.

Untuk mengakses AWS Elemental MediaStore

- Masuk ke AWS Management Console dan buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.

> **Note**
>
> Anda dapat login menggunakan salah satu kredensi IAM yang telah Anda buat untuk akun ini. Untuk informasi tentang membuat kredenal IAM, lihat. Menyiapkan AWS Elemental MediaStore

## Langkah 2: Buat wadah

Anda menggunakan kontainer di AWS Elemental MediaStore untuk menyimpan folder dan objek Anda. Anda dapat menggunakan kontainer untuk mengelompokkan objek terkait dengan cara yang sama seperti Anda menggunakan direktori untuk mengelompokkan file dalam sistem file. Anda tidak dikenakan biaya saat membuat kontainer; Anda hanya dikenakan biaya saat mengunggah objek ke kontainer.

Untuk membuat wadah

1. Pada halaman Containers, pilih Create container.
2. Untuk nama Container, ketikkan nama untuk kontainer Anda. Untuk informasi selengkapnya, lihat Aturan untuk nama kontainer.
3. Pilih Buat wadah. AWS Elemental MediaStore menambahkan kontainer baru ke daftar kontainer. Awalnya, status wadah adalah Membuat, dan kemudian berubah menjadi Aktif.

# Langkah 3: Unggah objek

Anda dapat mengunggah objek (masing-masing hingga 25 MB) ke wadah atau ke folder di dalam wadah. Untuk mengunggah objek ke folder, Anda menentukan jalur ke folder. Jika folder sudah ada, AWS Elemental MediaStore menyimpan objek di folder. Jika folder tidak ada, layanan membuatnya, dan kemudian menyimpan objek di folder.

> **Note**
>
> Nama file objek hanya dapat berisi huruf, angka, titik (.), garis bawah (_), tildes (~), dan tanda hubung (-).

Untuk mengunggah objek

1. Pada halaman Kontainer, pilih nama wadah yang baru saja Anda buat. Halaman detail untuk wadah muncul.
2. Pilih Unggah objek.
3. Untuk jalur Target, ketik jalur untuk folder. Misalnya, `premium/canada`. Jika salah satu folder di jalur belum ada, AWS Elemental membuatnya secara MediaStore otomatis.
4. Untuk Object, pilih Browse.
5. Arahkan ke folder yang sesuai, dan pilih satu objek untuk diunggah.
6. Pilih Buka, lalu pilih Unggah.

# Langkah 4: Akses objek

Anda dapat mengunduh objek Anda ke titik akhir yang ditentukan.

1. Pada halaman Kontainer, pilih nama wadah yang memiliki objek yang ingin Anda unduh.
2. Jika objek yang ingin Anda unduh ada di subfolder, lanjutkan memilih nama folder sampai Anda melihat objek.
3. Pilih nama objek.
4. Pada halaman detail untuk objek, pilih Unduh.

# Kontainer di AWS Elemental MediaStore

Anda menggunakan wadah MediaStore untuk menyimpan folder dan objek Anda. Objek terkait dapat dikelompokkan dalam wadah dengan cara yang sama seperti Anda menggunakan direktori untuk mengelompokkan file dalam sistem file. Anda tidak dikenakan biaya saat membuat kontainer; Anda hanya dikenakan biaya saat mengunggah objek ke kontainer. Untuk informasi selengkapnya tentang biaya, lihat Harga AWS Elemental MediaStore.

Topik


- Aturan untuk nama kontainer
- Membuat wadah
- Melihat detail untuk wadah
- Melihat daftar kontainer
- Menghapus wadah


## Aturan untuk nama kontainer

Saat Anda memilih nama untuk wadah Anda, ingatlah hal berikut:

- Nama harus unik dalam akun saat ini untuk Wilayah AWS saat ini.
- Nama dapat berisi huruf besar, huruf kecil, angka, dan garis bawah (_).
- Nama harus dari 1 hingga 255 karakter.
- Nilai peka huruf besar/kecil. Misalnya, Anda dapat memiliki wadah bernama `myContainer` dan folder bernama `mycontainer` karena nama-nama itu unik.
- Sebuah wadah tidak dapat diganti namanya setelah dibuat.

## Membuat wadah

Anda dapat membuat hingga 100 kontainer untuk setiap akun AWS. Anda dapat membuat folder sebanyak yang Anda inginkan, asalkan tidak bersarang lebih dari 10 level dalam wadah. Selain itu, Anda dapat mengunggah objek sebanyak yang Anda inginkan ke setiap wadah.

> **Tip**
>
> Anda juga dapat membuat wadah secara otomatis dengan menggunakan AWS CloudFormation template. AWS CloudFormation Template mengelola data untuk lima tindakan API: membuat container, menyetel akses logging, memperbarui kebijakan container default, menambahkan kebijakan cross-origin resource sharing (CORS), dan menambahkan kebijakan siklus hidup objek. Untuk informasi selengkapnya, silakan lihat Panduan Pengguna AWS CloudFormation.

Untuk membuat wadah (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Containers, pilih Create container.
3. Untuk nama Container, masukkan nama untuk wadah. Untuk informasi selengkapnya, lihat Aturan untuk nama kontainer.
4. Pilih Buat wadah. AWS Elemental MediaStore menambahkan kontainer baru ke daftar kontainer. Awalnya, status wadah adalah Membuat, dan kemudian berubah menjadi Aktif.

Untuk membuat wadah (AWS CLI)

- Di AWS CLI, gunakan `create-container` perintah:

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

Contoh berikut menunjukkan nilai kembali:

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

# Melihat detail untuk wadah

Detail untuk kontainer mencakup kebijakan kontainer, titik akhir, ARN, dan waktu pembuatan.

Untuk melihat detail untuk kontainer (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama wadah.

   Halaman detail kontainer muncul. Halaman ini dibagi menjadi dua bagian:

   - Bagian Objek, yang mencantumkan objek dan folder dalam wadah.
   - Bagian kebijakan Container, yang menampilkan kebijakan berbasis sumber daya yang terkait dengan container ini. Untuk informasi tentang kebijakan sumber daya, lihatKebijakan kontainer.

Untuk melihat detail kontainer (AWS CLI)

- Di AWS CLI, gunakan `describe-container` perintah:

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

Contoh berikut menunjukkan nilai kembali:

```
{
    "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"
    }
}
```

# Melihat daftar kontainer

Anda dapat melihat daftar semua kontainer yang terkait dengan akun Anda.

Untuk melihat daftar kontainer (konsol)

- Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.

  Halaman Kontainer muncul, mencantumkan semua kontainer yang terkait dengan akun Anda.

Untuk melihat daftar kontainer (AWS CLI)

- Di AWS CLI, gunakan `list-containers` perintah.

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

Contoh berikut menunjukkan nilai kembali:

```
{
    "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"
        }
    ]
```

```
}
```

# Menghapus wadah

Anda dapat menghapus wadah hanya jika tidak memiliki objek.

Untuk menghapus wadah (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih opsi di sebelah kiri nama kontainer.
3. Pilih Hapus.

Untuk menghapus kontainer (AWS CLI)

- Di AWS CLI, gunakan `delete-container` perintah:

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

  Perintah ini tidak memiliki nilai kembali.

# Kebijakan di AWS Elemental MediaStore

Anda dapat menerapkan satu atau beberapa kebijakan ini ke wadah AWS Elemental MediaStore Anda:

- Kebijakan kontainer - Menetapkan hak akses ke semua folder dan objek di dalam wadah. MediaStore menetapkan kebijakan default yang memungkinkan pengguna untuk melakukan semua MediaStore operasi pada container. Kebijakan ini menetapkan bahwa semua operasi harus dilakukan melalui HTTPS. Setelah membuat kontainer, Anda dapat mengedit kebijakan penampung.
- Kebijakan Cross-Origin Resource Sharing (CORS) - Memungkinkan aplikasi web klien dari satu domain berinteraksi dengan sumber daya di domain yang berbeda. MediaStore tidak menetapkan kebijakan CORS default.
- Kebijakan metrik - Memungkinkan MediaStore untuk mengirim metrik ke Amazon. CloudWatch MediaStore tidak menetapkan kebijakan metrik default.
- Kebijakan siklus hidup objek - Mengontrol berapa lama objek tetap berada dalam wadah MediaStore . MediaStoretidak menetapkan kebijakan siklus hidup objek default.

# Kebijakan kontainer di AWS Elemental MediaStore

Setiap kontainer memiliki kebijakan berbasis sumber daya yang mengatur hak akses ke semua folder dan objek dalam wadah tersebut. Kebijakan default, yang secara otomatis dilampirkan ke semua kontainer baru, memungkinkan akses ke semua MediaStore operasi AWS Elemental pada penampung. Ini menentukan bahwa akses ini memiliki kondisi yang membutuhkan HTTPS untuk operasi. Setelah membuat kontainer, Anda dapat mengedit kebijakan yang dilampirkan ke wadah tersebut.

Anda juga dapat menentukan kebijakan siklus hidup objek yang mengatur tanggal kedaluwarsa objek dalam wadah. Setelah objek mencapai usia maksimum yang Anda tentukan, layanan menghapus objek dari wadah.

Topik

- Melihat kebijakan kontainer
- Mengedit kebijakan kontainer
- Contoh kebijakan kontainer

# Melihat kebijakan kontainer

Anda dapat menggunakan konsol atau AWS CLI untuk melihat kebijakan penampung berbasis sumber daya.

Untuk melihat kebijakan kontainer (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama kontainer.

   Halaman detail kontainer muncul. Kebijakan ditampilkan di bagian Kebijakan kontainer.

Untuk melihat kebijakan kontainer (AWS CLI)

- Di AWS CLI, gunakan `get-container-policy` perintah:

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

  Contoh berikut menunjukkan nilai kembali:

  ```
  {
    "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"
            }
  ```

```
        }
      }
    ]
  }
}
```

## Mengedit kebijakan kontainer

Anda dapat mengedit izin dalam kebijakan kontainer default, atau Anda dapat membuat kebijakan baru yang menggantikan kebijakan default. Butuh waktu hingga lima menit agar kebijakan baru berlaku.

Untuk mengedit kebijakan kontainer (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama kontainer.
3. Pilih Sunting kebijakan. Untuk contoh yang menunjukkan cara menyetel izin yang berbeda, lihatthe section called “Contoh kebijakan kontainer”.
4. Buat perubahan yang sesuai, lalu pilih Simpan.

Untuk mengedit kebijakan kontainer (AWS CLI)

1. Buat file yang mendefinisikan kebijakan kontainer:

```
{
  "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. Di AWS CLI, gunakan `put-container-policy` perintah:

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

Perintah ini tidak memiliki nilai kembali.

## Contoh kebijakan kontainer

Contoh berikut menunjukkan kebijakan kontainer yang dibangun untuk grup pengguna yang berbeda.

Topik


- Contoh kebijakan wadah: Default
- Contoh kebijakan wadah: Akses baca publik melalui HTTPS
- Contoh kebijakan wadah: Akses baca publik melalui HTTP atau HTTPS
- Contoh kebijakan wadah: Akses baca lintas akun-HTTP diaktifkan
- Contoh kebijakan wadah: Akses baca lintas akun melalui HTTPS
- Contoh kebijakan kontainer: Akses baca lintas akun ke peran
- Contoh kebijakan kontainer: Akses penuh lintas akun ke peran
- Contoh kebijakan kontainer: Akses dibatasi ke alamat IP tertentu


### Contoh kebijakan wadah: Default

Saat Anda membuat container, AWS Elemental MediaStore secara otomatis melampirkan kebijakan berbasis sumber daya berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "MediaStoreFullAccess",
      "Action": [ "mediastore:*" ],
```

```
      "Principal":{
          "AWS" : "arn:aws:iam::<aws_account_number>:root"},
      "Effect": "Allow",
      "Resource": "arn:aws:mediastore:<region>:<owner acct number>:container/<container
 name>/*",
      "Condition": {
        "Bool": { "aws:SecureTransport": "true" }
      }
    }
  ]
}
```

Kebijakan ini dibangun ke dalam layanan, jadi Anda tidak perlu membuatnya. Namun, Anda dapat mengedit kebijakan pada penampung jika izin dalam kebijakan default tidak sejajar dengan izin yang ingin Anda gunakan untuk penampung.

Kebijakan default yang ditetapkan ke semua kontainer baru memungkinkan akses ke semua MediaStore operasi pada kontainer. Ini menentukan bahwa akses ini memiliki kondisi yang membutuhkan HTTPS untuk operasi.

## Contoh kebijakan wadah: Akses baca publik melalui HTTPS

Kebijakan contoh ini memungkinkan pengguna untuk mengambil objek melalui permintaan HTTPS. Ini memungkinkan akses baca ke siapa pun melalui koneksi SSL/TLS yang aman: pengguna yang diautentikasi dan pengguna anonim (pengguna yang tidak masuk). Pernyataan itu memiliki nama`PublicReadOverHttps`. Hal ini memungkinkan akses ke `GetObject` dan `DescribeObject` operasi pada objek apapun (seperti yang ditentukan oleh * di akhir jalur sumber daya). Ini menentukan bahwa akses ini memiliki kondisi yang memerlukan HTTPS untuk operasi:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadOverHttps",
      "Effect": "Allow",
      "Action": ["mediastore:GetObject", "mediastore:DescribeObject"],
      "Principal": "*",
      "Resource": "arn:aws:mediastore:<region>:<owner acct number>:container/<container
 name>/*",
      "Condition": {
        "Bool": {
```

```
            "aws:SecureTransport": "true"
        }
      }
    }
  ]
}
```

## Contoh kebijakan wadah: Akses baca publik melalui HTTP atau HTTPS

Kebijakan contoh ini memungkinkan akses ke `GetObject` dan `DescribeObject` operasi pada objek apa pun (seperti yang ditentukan oleh* di akhir jalur sumber daya). Ini memungkinkan akses baca ke siapa pun, termasuk semua pengguna yang diautentikasi dan pengguna anonim (pengguna yang tidak masuk):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadOverHttpOrHttps",
      "Effect": "Allow",
      "Action": ["mediastore:GetObject", "mediastore:DescribeObject"],
      "Principal": "*",
      "Resource": "arn:aws:mediastore:<region>:<owner acct number>:container/<container
 name>/*",
      "Condition": {
        "Bool": { "aws:SecureTransport": ["true", "false"] }
      }
    }
  ]
}
```

## Contoh kebijakan wadah: Akses baca lintas akun-HTTP diaktifkan

Kebijakan contoh ini memungkinkan pengguna untuk mengambil objek melalui permintaan HTTP. Ini memungkinkan akses ini ke pengguna yang diautentikasi dengan akses lintas akun. Objek tidak diperlukan untuk di-host di server dengan sertifikat SSL/TLS:

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

```
    "Effect" : "Allow",
    "Principal" : {
      "AWS" : "arn:aws:iam::<other acct number>:root"
    },
    "Action" : [ "mediastore:GetObject", "mediastore:DescribeObject" ],
    "Resource" : "arn:aws:mediastore:<region>:<owner acct number>:container/<container
 name>/*",
    "Condition" : {
      "Bool" : {
        "aws:SecureTransport" : [ "true", "false" ]
      }
    }
  } ]
}
```

## Contoh kebijakan wadah: Akses baca lintas akun melalui HTTPS

Kebijakan contoh ini memungkinkan akses ke `GetObject` dan `DescribeObject` operasi pada objek apa pun (sebagaimana ditentukan oleh* di akhir jalur sumber daya) yang dimiliki oleh pengguna root pengguna yang ditentukan<other acct number>. Ini menentukan bahwa akses ini memiliki kondisi yang memerlukan HTTPS untuk operasi:

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

## Contoh kebijakan kontainer: Akses baca lintas akun ke peran

Kebijakan contoh memungkinkan akses ke `GetObject` dan `DescribeObject` operasi pada objek apa pun (seperti yang ditentukan oleh* di akhir jalur sumber daya) yang dimiliki oleh<owner acct number>. Ini memungkinkan akses ini ke setiap pengguna <other acct number>jika akun tersebut telah mengambil peran yang ditentukan dalam<role name>:

```
{
  "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>/*",
    }
  ]
}
```

## Contoh kebijakan kontainer: Akses penuh lintas akun ke peran

Kebijakan contoh ini memungkinkan akses lintas akun untuk memperbarui objek apa pun di akun, selama pengguna masuk melalui HTTP. Ini juga memungkinkan akses lintas akun untuk menghapus, mengunduh, dan mendeskripsikan objek melalui HTTP atau HTTPS ke akun yang telah mengambil peran yang ditentukan:

- Pernyataan pertama adalah`CrossAccountRolePostOverHttps`. Ini memungkinkan akses ke `PutObject` operasi pada objek apa pun dan memungkinkan akses ini ke setiap pengguna dari akun yang ditentukan jika akun tersebut telah mengambil peran yang ditentukan dalam<role name>. Ini menentukan bahwa akses ini memiliki kondisi yang memerlukan HTTPS untuk operasi (kondisi ini harus selalu disertakan ketika menyediakan akses ke`PutObject`).

  Dengan kata lain, prinsipal apa pun yang memiliki akses lintas akun dapat mengakses`PutObject`, tetapi hanya melalui HTTPS.
- Pernyataan kedua adalah`CrossAccountFullAccessExceptPost`. Ini memungkinkan akses ke semua operasi kecuali `PutObject` pada objek apa pun. Ini memungkinkan akses ini ke setiap

pengguna dari akun yang ditentukan jika akun tersebut telah mengambil peran yang ditentukan dalam<role name>. Akses ini tidak memiliki kondisi memerlukan HTTPS untuk operasi.

Dengan kata lain, setiap akun yang memiliki akses lintas akun dapat mengaksesDeleteObject,GetObject, dan seterusnya (tetapi tidakPutObject), dan dapat melakukan ini melalui HTTP atau HTTPS.

Jika Anda tidak mengecualikan PutObject dari pernyataan kedua, pernyataan tidak akan valid (karena jika Anda menyertakan PutObject Anda harus secara eksplisit menetapkan HTTPS sebagai kondisi).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CrossAccountRolePostOverHttps",
      "Effect": "Allow",
      "Action": "mediastore:PutObject",
      "Principal":{
        "AWS": "arn:aws:iam::<other acct number>:role/<role name>"},
      "Resource": "arn:aws:mediastore:<region>:<owner acct number>:container/<container
 name>/*",
      "Condition": {
        "Bool": {
            "aws:SecureTransport": "true"
        }
      }
    },
    {
      "Sid": "CrossAccountFullAccessExceptPost",
      "Effect": "Allow",
      "NotAction": "mediastore:PutObject",
      "Principal":{
        "AWS": "arn:aws:iam::<other acct number>:role/<role name>"},
      "Resource": "arn:aws:mediastore:<region>:<owner acct number>:container/<container
 name>/*"
    }
  ]
}
```

## Contoh kebijakan kontainer: Akses dibatasi ke alamat IP tertentu

Kebijakan contoh ini memungkinkan akses ke semua MediaStore operasi AWS Elemental pada objek dalam wadah yang ditentukan. Namun, permintaan harus berasal dari rentang alamat IP yang ditentukan dalam syarat.

Kondisi dalam pernyataan ini mengidentifikasi kisaran 198.51.100.* alamat IP Internet Protocol versi 4 (IPv4) yang diizinkan, dengan satu pengecualian: 198.51.100.188.

`Condition`Blok menggunakan `IpAddress` dan `NotIpAddress` kondisi dan kunci kondisi, yang merupakan kunci `aws:SourceIp` kondisi di seluruh AWS. `aws:sourceIp` IPv4 Nilai-nilai menggunakan notasi CIDR standar. Untuk informasi selengkapnya, lihat Operator Kondisi Alamat IP di Panduan Pengguna IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AccessBySpecificIPAddress",
            "Effect": "Allow",
            "Action": [
                "mediastore:GetObject",
                "mediastore:DescribeObject"
            ],
            "Principal": "*",
            "Resource": "arn:aws:mediastore:<region>:<owner acct number>:container/
<container name>/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "198.51.100.0/24"
                    ]
                },
                "NotIpAddress": {
                    "aws:SourceIp": "198.51.100.188/32"
                }
            }
        }
    ]
}
```

# Kebijakan berbagi sumber daya lintas asal (CORS) di AWS Elemental MediaStore

Cross-origin resource sharing (CORS) menentukan cara aplikasi web klien yang dimuat di dalam satu domain untuk berinteraksi dengan sumber daya di domain yang berbeda. Dengan dukungan CORS di AWS MediaStore Elemental, Anda dapat membangun aplikasi web sisi klien yang kaya MediaStore dengan dan secara selektif mengizinkan akses lintas asal ke sumber daya Anda. MediaStore

> **Note**
>
> Jika Anda menggunakan Amazon CloudFront untuk mendistribusikan konten dari container yang memiliki kebijakan CORS, pastikan untuk mengonfigurasi distribusi AWS MediaStore Elemental (termasuk langkah untuk mengedit perilaku cache untuk menyiapkan CORS).

Bagian ini memberikan ikhtisar tentang CORS. Subtopik menjelaskan bagaimana Anda dapat mengaktifkan CORS menggunakan konsol AWS MediaStore Elemental, atau secara terprogram menggunakan MediaStore REST API dan AWS. SDKs

Topik


- Skenario kasus penggunaan CORS
- Menambahkan kebijakan CORS ke wadah
- Melihat kebijakan CORS
- Mengedit kebijakan CORS
- Menghapus kebijakan CORS
- Penyelesaian masalah isu CORS
- Contoh kebijakan CORS


## Skenario kasus penggunaan CORS

Berikut ini adalah contoh skenario untuk menggunakan CORS:

- Skenario 1: Misalkan Anda mendistribusikan video streaming langsung dalam wadah AWS MediaStore Elemental bernama. LiveVideo Pengguna Anda memuat titik akhir manifes video `http://livevideo.mediastore.ap-southeast-2.amazonaws.com` dari asal tertentu seperti`www.example.com`. Anda ingin menggunakan pemutar JavaScript video untuk mengakses

video yang berasal dari wadah ini melalui permintaan dan tidak diautentikasi`GET`. `PUT` Browser biasanya akan memblokir JavaScript agar tidak mengizinkan permintaan tersebut, tetapi Anda dapat menetapkan kebijakan CORS pada penampung Anda untuk mengaktifkan permintaan ini secara eksplisit. `www.example.com`

- Skenario 2: Misalkan Anda ingin meng-host streaming langsung yang sama seperti di Skenario 1 dari MediaStore penampung Anda, tetapi ingin mengizinkan permintaan dari asal mana pun. Anda dapat mengonfigurasi kebijakan CORS untuk mengizinkan asal wildcard (*), sehingga permintaan dari asal mana pun dapat mengakses video.

## Menambahkan kebijakan CORS ke wadah

Bagian ini menjelaskan cara menambahkan konfigurasi cross-origin resource sharing (CORS) ke container MediaStore AWS Elemental. CORS memungkinkan aplikasi web klien yang dimuat di satu domain untuk berinteraksi dengan sumber daya di domain lain.

Untuk mengonfigurasi penampung agar mengizinkan permintaan lintas asal, Anda menambahkan kebijakan CORS ke penampung. Kebijakan CORS mendefinisikan aturan yang mengidentifikasi asal yang Anda izinkan untuk mengakses penampung, operasi (metode HTTP) yang didukung untuk setiap asal, dan informasi spesifik operasi lainnya.

Saat Anda menambahkan kebijakan CORS ke penampung, kebijakan penampung (yang mengatur hak akses ke penampung) terus berlaku.

Untuk menambahkan kebijakan CORS (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama wadah yang ingin Anda buat kebijakan CORS.

   Halaman detail kontainer muncul.
3. Di bagian kebijakan Container CORS, pilih Buat kebijakan CORS.
4. Masukkan kebijakan dalam format JSON, lalu pilih Simpan.

Untuk menambahkan kebijakan CORS ()AWS CLI

1. Buat file yang mendefinisikan kebijakan CORS:

```
[
  {
```

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

2. Di AWS CLI, gunakan `put-cors-policy` perintah.

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

Perintah ini tidak memiliki nilai kembali.

## Melihat kebijakan CORS

Cross-origin resource sharing (CORS) menentukan cara aplikasi web klien yang dimuat di dalam satu domain untuk berinteraksi dengan sumber daya di domain yang berbeda.

Untuk melihat kebijakan CORS (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama penampung yang ingin Anda lihat kebijakan CORS.

   Halaman detail kontainer muncul, dengan kebijakan CORS di bagian kebijakan Container CORS.

Untuk melihat kebijakan CORS ()AWS CLI

- Di AWS CLI, gunakan `get-cors-policy` perintah:

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

Contoh berikut menunjukkan nilai kembali:

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

## Mengedit kebijakan CORS

Cross-origin resource sharing (CORS) menentukan cara aplikasi web klien yang dimuat di dalam satu domain untuk berinteraksi dengan sumber daya di domain yang berbeda.

Untuk mengedit kebijakan CORS (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama penampung yang ingin Anda edit kebijakan CORS.

   Halaman detail kontainer muncul.
3. Di bagian kebijakan Container CORS, pilih Edit kebijakan CORS.
4. Buat perubahan pada kebijakan, lalu pilih Simpan.

Untuk mengedit kebijakan CORS ()AWS CLI

1. Buat file yang mendefinisikan kebijakan CORS yang diperbarui:

```
[
  {
    "AllowedHeaders": [
```

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

2. Di AWS CLI, gunakan `put-cors-policy` perintah.

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

Perintah ini tidak memiliki nilai kembali.

## Menghapus kebijakan CORS

Cross-origin resource sharing (CORS) menentukan cara aplikasi web klien yang dimuat di dalam satu domain untuk berinteraksi dengan sumber daya di domain yang berbeda. Menghapus kebijakan CORS dari wadah akan menghapus izin untuk permintaan lintas asal.

Untuk menghapus kebijakan CORS (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama penampung yang ingin Anda hapus kebijakan CORS.

   Halaman detail kontainer muncul.
3. Di bagian kebijakan Container CORS, pilih Hapus kebijakan CORS.
4. Pilih Lanjutkan untuk mengonfirmasi, lalu pilih Simpan.

Untuk menghapus kebijakan CORS ()AWS CLI

- Di AWS CLI, gunakan `delete-cors-policy` perintah:

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

Perintah ini tidak memiliki nilai kembali.

## Penyelesaian masalah isu CORS

Jika Anda mengalami perilaku tak terduga saat mengakses container yang memiliki kebijakan CORS, ikuti langkah-langkah berikut untuk memecahkan masalah tersebut.

1. Verifikasi bahwa kebijakan CORS dilampirkan ke wadah.

   Untuk petunjuk, silakan lihat the section called “Melihat kebijakan CORS”.

2. Tangkap permintaan dan respons lengkap menggunakan alat pilihan Anda (seperti konsol pengembang browser Anda). Verifikasi bahwa kebijakan CORS yang dilampirkan ke penampung mencakup setidaknya satu aturan CORS yang cocok dengan data dalam permintaan Anda, sebagai berikut:

   a. Verifikasi bahwa permintaan memiliki `Origin` header.

      Jika header tidak ada, AWS Elemental MediaStore tidak memperlakukan permintaan sebagai permintaan lintas asal dan tidak mengirim header respons CORS kembali dalam respons.

   b. Verifikasi bahwa `Origin` header dalam permintaan Anda cocok dengan setidaknya salah satu `AllowedOrigins` elemen dalam spesifik`CORSRule`.

      Skema, host, dan nilai port di header `Origin` permintaan harus cocok dengan `AllowedOrigins` di`CORSRule`. Misalnya, jika Anda menyetel `CORSRule` untuk mengizinkan asal`http://www.example.com`, maka keduanya `https://www.example.com` dan `http://www.example.com:80` asal dalam permintaan Anda tidak cocok dengan asal yang diizinkan dalam konfigurasi Anda.

   c. Verifikasi bahwa metode dalam permintaan Anda (atau metode yang ditentukan `Access-Control-Request-Method` dalam kasus permintaan preflight) adalah salah satu `AllowedMethods` elemen yang sama`CORSRule`.

d. Untuk permintaan preflight, jika permintaan mencakup header `Access-Control-Request-Headers`, verifikasikan bahwa `CORSRule` termasuk entri `AllowedHeaders` untuk setiap nilai dalam header `Access-Control-Request-Headers`.

## Contoh kebijakan CORS

Contoh berikut menunjukkan kebijakan cross-origin resource sharing (CORS).

Topik


- Contoh kebijakan CORS: Baca akses untuk domain apa pun
- Contoh kebijakan CORS: Baca akses untuk domain tertentu


### Contoh kebijakan CORS: Baca akses untuk domain apa pun

Kebijakan berikut memungkinkan halaman web dari domain apa pun untuk mengambil konten dari penampung AWS MediaStore Elemental Anda. Permintaan mencakup semua header HTTP dari domain asal, dan layanan hanya merespons permintaan HTTP GET dan HTTP HEAD dari domain asal. Hasilnya di-cache selama 3.000 detik sebelum serangkaian hasil baru dikirimkan.

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

### Contoh kebijakan CORS: Baca akses untuk domain tertentu

Kebijakan berikut memungkinkan halaman web `https://www.example.com` untuk mengambil konten dari penampung AWS MediaStore Elemental Anda. Permintaan mencakup semua header

HTTP dari`https://www.example.com`, dan layanan hanya merespons permintaan HTTP GET dan HTTP HEAD dari. `https://www.example.com` Hasilnya di-cache selama 3.000 detik sebelum serangkaian hasil baru dikirimkan.

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

# Kebijakan siklus hidup objek di AWS Elemental MediaStore

Untuk setiap kontainer, Anda dapat membuat kebijakan siklus hidup objek yang mengatur berapa lama objek harus disimpan dalam wadah. Saat objek mencapai usia maksimum yang Anda tentukan, AWS Elemental MediaStore menghapus objek. Anda dapat menghapus objek setelah mereka tidak lagi diperlukan untuk menghemat biaya penyimpanan.

Anda juga dapat menentukan yang MediaStore harus memindahkan objek ke kelas penyimpanan akses jarang (IA) setelah mereka mencapai usia tertentu. Objek yang disimpan di kelas penyimpanan IA memiliki tingkat penyimpanan dan pengambilan yang berbeda dari objek yang disimpan di kelas penyimpanan standar. Untuk informasi selengkapnya, silakan lihat Harga MediaStore .

Kebijakan siklus hidup objek berisi aturan, yang menentukan umur objek berdasarkan subfolder. (Anda tidak dapat menetapkan kebijakan siklus hidup objek ke objek individual). Anda hanya dapat melampirkan satu kebijakan siklus hidup objek ke wadah, tetapi Anda dapat menambahkan hingga 10 aturan ke setiap kebijakan siklus hidup objek. Untuk informasi selengkapnya, lihat Komponen kebijakan siklus hidup objek.

Topik

- Komponen kebijakan siklus hidup objek

- Menambahkan kebijakan siklus hidup objek ke wadah
- Melihat kebijakan siklus hidup objek
- Mengedit kebijakan siklus hidup objek
- Menghapus kebijakan siklus hidup objek
- Contoh kebijakan siklus hidup objek

## Komponen kebijakan siklus hidup objek

Kebijakan siklus hidup objek mengatur berapa lama objek tetap berada dalam wadah AWS Elemental. MediaStore Setiap kebijakan siklus hidup objek terdiri dari satu atau lebih aturan, yang menentukan umur objek. Aturan dapat diterapkan ke satu folder, beberapa folder, atau seluruh wadah.

Anda dapat melampirkan kebijakan siklus hidup satu objek ke wadah, dan setiap kebijakan siklus hidup objek dapat berisi hingga 10 aturan. Anda tidak dapat menetapkan kebijakan siklus hidup objek ke objek individual.

### Aturan dalam kebijakan siklus hidup objek

Anda dapat membuat tiga jenis aturan:

- Data sementara
- Hapus objek
- Transisi siklus hidup

#### Data sementara

Aturan data sementara menetapkan objek untuk kedaluwarsa dalam hitungan detik. Jenis aturan ini hanya berlaku untuk objek yang ditambahkan ke wadah setelah kebijakan menjadi efektif. Diperlukan waktu hingga 20 menit MediaStore untuk menerapkan kebijakan baru ke wadah.

Contoh aturan untuk data transien terlihat seperti ini:

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

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

Aturan data transien memiliki tiga bagian:

- `path`: Selalu atur ke`wildcard`. Anda menggunakan bagian ini untuk menentukan objek mana yang ingin Anda hapus. Anda dapat menggunakan satu atau lebih wildcard, diwakili oleh tanda bintang (*). Setiap wildcard mewakili kombinasi nol atau lebih karakter. Misalnya, `"path": [ {"wildcard": "Football/index*.m3u8"} ],` berlaku untuk semua file dalam `Football` folder yang cocok dengan pola `index*.m3u8` (seperti index.m3u8, index1.m3us8, dan index123456.m3u8). Anda dapat menyertakan hingga 10 jalur dalam satu aturan.
- `seconds_since_create`: Selalu atur ke`numeric`. Anda dapat menentukan nilai dari 1-300 detik. Anda juga dapat mengatur operator menjadi lebih besar dari (**>**) atau lebih besar dari atau sama dengan (**>=**).
- `action`: Selalu atur ke`EXPIRE`.

Untuk aturan data transien (objek kedaluwarsa dalam hitungan detik), tidak ada jeda antara kedaluwarsa suatu objek dan penghapusan objek.

> **Note**
>
> Objek yang tunduk pada aturan data sementara tidak termasuk dalam respons. `list-items` Selain itu, objek yang kedaluwarsa karena aturan data sementara tidak memancarkan CloudWatch peristiwa ketika mereka kedaluwarsa.

Hapus objek

Aturan objek hapus menetapkan objek untuk kedaluwarsa dalam beberapa hari. Jenis aturan ini berlaku untuk semua objek dalam wadah, bahkan jika mereka ditambahkan ke wadah sebelum kebijakan dibuat. Diperlukan waktu hingga 20 menit MediaStore untuk menerapkan kebijakan baru, tetapi dapat memakan waktu hingga 24 jam agar objek dapat dibersihkan dari wadah.

Contoh dua aturan untuk menghapus objek terlihat seperti ini:

```
    {
```

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

Hapus aturan objek memiliki tiga bagian:

- `path`: Setel ke salah satu `prefix` atau`wildcard`. Anda tidak dapat mencampur `prefix` dan `wildcard` dalam aturan yang sama. Jika Anda ingin menggunakan keduanya, Anda harus membuat satu aturan untuk `prefix` dan aturan terpisah untuk`wildcard`, seperti yang ditunjukkan pada contoh di atas.
  - `prefix`- Anda mengatur jalur ke `prefix` jika Anda ingin menghapus semua objek dalam folder tertentu. Jika parameternya kosong (`"path": [ { "prefix": "" } ],`), targetnya adalah semua objek yang disimpan di mana saja dalam wadah saat ini. Anda dapat menyertakan hingga 10 `prefix` jalur dalam satu aturan.
  - `wildcard`- Anda mengatur jalur ke `wildcard` jika Anda ingin menghapus objek tertentu berdasarkan nama file dan/atau jenis file. Anda dapat menggunakan satu atau lebih wildcard, diwakili oleh tanda bintang (*). Setiap wildcard mewakili kombinasi nol atau lebih karakter. Misalnya, `"path": [ {"wildcard": "Football/*.ts"} ],` berlaku untuk semua file dalam `Football` folder yang cocok dengan pola `*.ts` (seperti filename.ts, filename1.ts, dan filename123456.ts). Anda dapat menyertakan hingga 10 `wildcard` jalur dalam satu aturan.
- `days_since_create`: Selalu atur ke`numeric`. Anda dapat menentukan nilai dari 1-36.500 hari. Anda juga dapat mengatur operator menjadi lebih besar dari (**>**) atau lebih besar dari atau sama dengan (**>=**).
- `action`: Selalu atur ke`EXPIRE`.

Untuk menghapus aturan objek (objek kedaluwarsa dalam beberapa hari), mungkin ada sedikit jeda antara kedaluwarsa objek dan penghapusan objek. Namun, perubahan penagihan terjadi segera setelah objek kedaluwarsa. Misalnya, jika aturan siklus hidup menentukan 10`days_since_create`, akun tidak ditagih untuk objek setelah objek berusia 10 hari, bahkan jika objek belum dihapus.

Transisi siklus hidup

Aturan transisi siklus hidup menetapkan objek yang akan dipindahkan ke kelas penyimpanan akses jarang (IA) setelah mencapai usia tertentu, diukur dalam beberapa hari. Objek yang disimpan di kelas penyimpanan IA memiliki tingkat penyimpanan dan pengambilan yang berbeda dari objek yang disimpan di kelas penyimpanan standar. Untuk informasi selengkapnya, silakan lihat Harga MediaStore.

Setelah objek pindah ke kelas penyimpanan IA, Anda tidak dapat memindahkannya kembali ke kelas penyimpanan standar.

Aturan transisi siklus hidup berlaku untuk semua objek dalam wadah, meskipun ditambahkan ke wadah sebelum kebijakan dibuat. Diperlukan waktu hingga 20 menit MediaStore untuk menerapkan kebijakan baru, tetapi dapat memakan waktu hingga 24 jam agar objek dapat dibersihkan dari wadah.

Contoh aturan transisi siklus hidup terlihat seperti ini:

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

Aturan transisi siklus hidup memiliki tiga bagian:

- `path`: Setel ke salah satu `prefix` atau`wildcard`. Anda tidak dapat mencampur `prefix` dan `wildcard` dalam aturan yang sama. Jika Anda ingin menggunakan keduanya, Anda harus membuat satu aturan untuk `prefix` dan aturan terpisah untuk`wildcard`.
  - `prefix`- Anda mengatur jalur ke `prefix` jika Anda ingin transisi semua objek dalam folder tertentu ke kelas penyimpanan IA. Jika parameternya kosong (`"path": [ { "prefix":`

`"" } ],`), targetnya adalah semua objek yang disimpan di mana saja dalam wadah saat ini. Anda dapat menyertakan hingga 10 `prefix` jalur dalam satu aturan.

- `wildcard`- Anda mengatur jalur ke `wildcard` jika Anda ingin transisi objek tertentu ke kelas penyimpanan IA berdasarkan nama file dan/atau jenis file. Anda dapat menggunakan satu atau lebih wildcard, diwakili oleh tanda bintang (*). Setiap wildcard mewakili kombinasi nol atau lebih karakter. Misalnya, `"path": [ {"wildcard": "Football/*.ts"} ],` berlaku untuk semua file dalam `Football` folder yang cocok dengan pola `*.ts` (seperti filename.ts, filename1.ts, dan filename123456.ts). Anda dapat menyertakan hingga 10 `wildcard` jalur dalam satu aturan.

- `days_since_create`: Selalu atur ke`"numeric": [">=" , 30]`.
- `action`: Selalu atur ke`ARCHIVE`.

## Contoh

Misalkan sebuah wadah bernama `LiveEvents` memiliki empat subfolder:`Footbal`l,, `BaseballBasketball`, dan`AwardsShow`. Kebijakan siklus hidup objek yang ditetapkan ke `LiveEvents` folder mungkin terlihat seperti ini:

```
{
    "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"
            }
    ]
}
```

Kebijakan sebelumnya menentukan hal berikut:

- Aturan pertama menginstruksikan AWS MediaStore Elemental untuk menghapus objek yang disimpan dalam `LiveEvents/Football` folder dan `LiveEvents/Baseball` folder setelah mereka lebih dari 28 hari.
- Aturan kedua menginstruksikan layanan untuk menghapus objek yang disimpan dalam `LiveEvents/AwardsShow` folder ketika mereka berusia 15 hari atau lebih.
- Aturan ketiga menginstruksikan layanan untuk menghapus objek yang disimpan di mana saja dalam `LiveEvents` wadah setelah mereka lebih dari 40 hari. Aturan ini berlaku untuk objek yang disimpan langsung dalam `LiveEvents` wadah, serta objek yang disimpan di salah satu dari empat subfolder wadah.
- Aturan keempat menginstruksikan layanan untuk menghapus objek dalam `Football` folder yang cocok dengan pola `*.ts` setelah mereka lebih dari 20 hari.
- Aturan kelima menginstruksikan layanan untuk menghapus objek dalam `Football` folder yang cocok dengan pola `index*.m3u8` setelah mereka lebih tua dari 15 detik. MediaStore menghapus file-file ini 16 detik setelah mereka ditempatkan dalam wadah.
- Aturan keenam menginstruksikan layanan untuk memindahkan objek dalam `Program` folder ke kelas penyimpanan IA setelah mereka berusia 30 hari.

Untuk lebih banyak contoh kebijakan siklus hidup objek, lihat. Contoh kebijakan siklus hidup objek

## Menambahkan kebijakan siklus hidup objek ke wadah

Kebijakan siklus hidup objek memungkinkan Anda menentukan berapa lama untuk menyimpan objek dalam wadah. Anda menetapkan tanggal kedaluwarsa, dan setelah tanggal kedaluwarsa AWS Elemental menghapus objek. MediaStore Diperlukan waktu hingga 20 menit bagi layanan untuk menerapkan kebijakan baru ke wadah.

Untuk informasi tentang cara membuat kebijakan siklus hidup, lihat. Komponen kebijakan siklus hidup objek

> **Note**
>
> Untuk menghapus aturan objek (objek kedaluwarsa dalam beberapa hari), mungkin ada sedikit jeda antara kedaluwarsa objek dan penghapusan objek. Namun, perubahan penagihan terjadi segera setelah objek kedaluwarsa. Misalnya, jika aturan siklus hidup

menentukan 10`days_since_create`, akun tidak ditagih untuk objek setelah objek berusia 10 hari, bahkan jika objek belum dihapus.

Untuk menambahkan kebijakan siklus hidup objek (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Container, pilih nama kontainer yang ingin Anda buat kebijakan siklus hidup objek.

   Halaman detail kontainer muncul.
3. Di bagian Kebijakan siklus hidup objek, pilih Buat kebijakan siklus hidup objek.
4. Masukkan kebijakan dalam format JSON, lalu pilih Simpan.

Untuk menambahkan kebijakan siklus hidup objek ()AWS CLI

1. Buat file yang mendefinisikan kebijakan siklus hidup objek:

```
{
    "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. Di AWS CLI, gunakan `put-lifecycle-policy` perintah:

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

Perintah ini tidak memiliki nilai kembali. Layanan melampirkan kebijakan yang ditentukan ke wadah.

## Melihat kebijakan siklus hidup objek

Kebijakan siklus hidup objek menentukan berapa lama objek harus disimpan dalam wadah.

Untuk melihat kebijakan siklus hidup objek (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Container, pilih nama kontainer yang ingin Anda lihat kebijakan siklus hidup objek.

   Halaman detail kontainer akan muncul, dengan kebijakan siklus hidup objek di bagian Kebijakan siklus hidup Object.

Untuk melihat kebijakan siklus hidup objek ()AWS CLI

- Di AWS CLI, gunakan `get-lifecycle-policy` perintah:

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

Contoh berikut menunjukkan nilai kembali:

```
{
    "LifecyclePolicy": "{
        "rules": [
            {
```

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

## Mengedit kebijakan siklus hidup objek

Anda tidak dapat mengedit kebijakan siklus hidup objek yang ada. Namun, Anda dapat mengubah kebijakan yang ada dengan mengunggah kebijakan penggantian. Diperlukan waktu hingga 20 menit bagi layanan untuk menerapkan kebijakan yang diperbarui ke wadah.

Untuk mengedit kebijakan siklus hidup objek (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama wadah yang ingin Anda edit kebijakan siklus hidup objek.

   Halaman detail kontainer muncul.
3. Di bagian Kebijakan siklus hidup objek, pilih Edit kebijakan siklus hidup objek.
4. Buat perubahan pada kebijakan, lalu pilih Simpan.

Untuk mengedit kebijakan siklus hidup objek ()AWS CLI

1. Buat file yang mendefinisikan kebijakan siklus hidup objek yang diperbarui:

```
{
    "rules": [
         {
            "definition": {
                "path": [
                    {"prefix": "Football/"},
```

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

2. Di AWS CLI, gunakan `put-lifecycle-policy` perintah:

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

Perintah ini tidak memiliki nilai kembali. Layanan melampirkan kebijakan yang ditentukan ke wadah, menggantikan kebijakan sebelumnya.

## Menghapus kebijakan siklus hidup objek

Saat Anda menghapus kebijakan siklus hidup objek, layanan membutuhkan waktu hingga 20 menit untuk menerapkan perubahan ke wadah.

Untuk menghapus kebijakan siklus hidup objek (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama penampung yang ingin Anda hapus kebijakan siklus hidup objek.

   Halaman detail kontainer muncul.
3. Di bagian Kebijakan siklus hidup objek, pilih Hapus kebijakan siklus hidup.
4. Pilih Lanjutkan untuk mengonfirmasi, lalu pilih Simpan.

Untuk menghapus kebijakan siklus hidup objek ()AWS CLI

- Di AWS CLI, gunakan `delete-lifecycle-policy` perintah:

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

Perintah ini tidak memiliki nilai kembali.

## Contoh kebijakan siklus hidup objek

Contoh berikut menunjukkan kebijakan siklus hidup objek.

Topik

- Contoh kebijakan siklus hidup objek: Kedaluwarsa dalam hitungan detik
- Contoh kebijakan siklus hidup objek: Kedaluwarsa dalam beberapa hari
- Contoh kebijakan siklus hidup objek: Transisi ke kelas penyimpanan akses jarang
- Contoh kebijakan siklus hidup objek: Beberapa aturan
- Contoh kebijakan siklus hidup objek: Kontainer kosong

### Contoh kebijakan siklus hidup objek: Kedaluwarsa dalam hitungan detik

Kebijakan berikut menetapkan bahwa MediaStore menghapus objek yang cocok dengan semua kriteria berikut:

- Objek ditambahkan ke wadah setelah kebijakan menjadi efektif.
- Objek disimpan dalam `Football` folder.
- Objek memiliki ekstensi file`m3u8`.
- Objek telah berada di wadah selama lebih dari 20 detik.

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

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

## Contoh kebijakan siklus hidup objek: Kedaluwarsa dalam beberapa hari

Kebijakan berikut menetapkan bahwa MediaStore menghapus objek yang cocok dengan semua kriteria berikut:

- Objek disimpan dalam `Program` folder
- Objek memiliki ekstensi file `ts`
- Objek telah berada di wadah selama lebih dari 5 hari

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

## Contoh kebijakan siklus hidup objek: Transisi ke kelas penyimpanan akses jarang

Kebijakan berikut menetapkan bahwa MediaStore memindahkan objek ke kelas penyimpanan akses jarang (IA) ketika mereka berusia 30 hari. Objek yang disimpan di kelas penyimpanan IA memiliki tingkat penyimpanan dan pengambilan yang berbeda dari objek yang disimpan di kelas penyimpanan standar.

`days_since_create`Bidang harus diatur ke`"numeric": [">=" ,30]`.

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

## Contoh kebijakan siklus hidup objek: Beberapa aturan

Kebijakan berikut menetapkan bahwa MediaStore melakukan hal berikut:

- Pindahkan objek yang disimpan dalam `AwardsShow` folder ke kelas penyimpanan akses jarang (IA) setelah 30 hari
- Hapus objek yang memiliki ekstensi file `m3u8` dan disimpan dalam `Football` folder setelah 20 detik
- Hapus objek yang disimpan dalam `April` folder setelah 10 hari
- Hapus objek yang memiliki ekstensi file `ts` dan disimpan dalam `Program` folder setelah 5 hari

```
{
    "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"
        }
    ]
}
```

## Contoh kebijakan siklus hidup objek: Kontainer kosong

Kebijakan siklus hidup objek berikut menetapkan bahwa MediaStore menghapus semua objek dalam wadah, termasuk folder dan subfolder, 1 hari setelah mereka ditambahkan ke wadah. Jika penampung menyimpan objek apa pun sebelum kebijakan ini diterapkan, MediaStore hapus objek

1 hari setelah kebijakan berlaku. Diperlukan waktu hingga 20 menit bagi layanan untuk menerapkan kebijakan baru ke wadah.

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

# Kebijakan metrik di AWS Elemental MediaStore

Untuk setiap kontainer, Anda dapat menambahkan kebijakan metrik untuk mengizinkan AWS Elemental mengirim metrik MediaStore ke Amazon. CloudWatch Diperlukan waktu hingga 20 menit agar kebijakan baru berlaku. Untuk deskripsi setiap MediaStore metrik, lihatMediaStore metrik.

Kebijakan metrik berisi hal-hal berikut:

- Pengaturan untuk mengaktifkan atau menonaktifkan metrik di tingkat kontainer.
- Di mana saja dari nol hingga lima aturan yang memungkinkan metrik di tingkat objek. Jika kebijakan berisi aturan, setiap aturan harus mencakup kedua hal berikut:
  - Grup objek yang mendefinisikan objek mana yang akan dimasukkan dalam grup. Definisi dapat berupa jalur atau nama file, tetapi tidak dapat memiliki lebih dari 900 karakter. Karakter yang valid adalah: a-z, A-Z, 0-9, _ (garis bawah), = (sama),: (titik dua),. (periode), - (tanda hubung), ~ (tilde),/(garis miring), dan * (tanda bintang). Wildcard (*) dapat diterima.
  - Nama grup objek yang memungkinkan Anda merujuk ke grup objek. Nama tidak boleh memiliki lebih dari 30 karakter. Karakter yang valid adalah: a-z, A-Z, 0-9, dan _ (garis bawah).

Jika objek cocok dengan beberapa aturan, CloudWatch menampilkan titik data untuk setiap aturan yang cocok. Misalnya, jika objek cocok dengan dua aturan bernama `rule1` dan`rule2`,

CloudWatch menampilkan dua titik data untuk aturan ini. Yang pertama memiliki dimensi `ObjectGroupName=rule1` dan yang kedua memiliki dimensi`ObjectGroupName=rule2`.

Topik


- Menambahkan kebijakan metrik
- Melihat kebijakan metrik
- Mengedit kebijakan metrik
- Contoh kebijakan metrik


## Menambahkan kebijakan metrik

Kebijakan metrik berisi aturan yang menentukan metrik mana yang MediaStore dikirim AWS Elemental ke Amazon. CloudWatch Untuk contoh kebijakan metrik, lihatContoh kebijakan metrik.

Untuk menambahkan kebijakan metrik (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama penampung yang ingin Anda tambahkan kebijakan metrik.

   Halaman detail kontainer muncul.
3. Di bagian Kebijakan metrik, pilih Buat kebijakan metrik.
4. Masukkan kebijakan dalam format JSON, lalu pilih Simpan.

## Melihat kebijakan metrik

Anda dapat menggunakan konsol atau AWS CLI untuk melihat kebijakan metrik kontainer.

Untuk melihat kebijakan metrik (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama kontainer.

   Halaman detail kontainer muncul. Kebijakan ditampilkan di bagian Kebijakan metrik.

# Mengedit kebijakan metrik

Kebijakan metrik berisi aturan yang menentukan metrik mana yang MediaStore dikirim AWS Elemental ke Amazon. CloudWatch Saat Anda mengedit kebijakan metrik yang sudah ada, kebijakan baru akan diberlakukan hingga 20 menit. Untuk contoh kebijakan metrik, lihatContoh kebijakan metrik.

Untuk mengedit kebijakan metrik (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama kontainer.
3. Di bagian Kebijakan metrik, pilih Edit kebijakan metrik.
4. Buat perubahan yang sesuai, lalu pilih Simpan.

# Contoh kebijakan metrik

Contoh berikut menunjukkan kebijakan metrik yang dibangun untuk kasus penggunaan yang berbeda.

Topik

- Contoh kebijakan metrik: Metrik tingkat kontainer
- Contoh kebijakan metrik: Metrik tingkat jalur
- Contoh kebijakan metrik: Metrik tingkat kontainer dan tingkat jalur
- Contoh kebijakan metrik: Metrik tingkat jalur menggunakan wildcard
- Contoh kebijakan metrik: Metrik tingkat jalur dengan aturan yang tumpang tindih

## Contoh kebijakan metrik: Metrik tingkat kontainer

Kebijakan contoh ini menunjukkan bahwa AWS Elemental MediaStore harus mengirim metrik ke Amazon CloudWatch di tingkat penampung. Misalnya, ini termasuk `RequestCount` metrik yang menghitung jumlah `Put` permintaan yang dibuat ke wadah. Atau, Anda dapat mengatur ini ke`DISABLED`.

Karena tidak ada aturan dalam kebijakan ini, MediaStore tidak mengirim metrik di tingkat jalur. Misalnya, Anda tidak dapat melihat berapa banyak `Put` permintaan yang dibuat ke folder tertentu dalam wadah ini.

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

## Contoh kebijakan metrik: Metrik tingkat jalur

Kebijakan contoh ini menunjukkan bahwa AWS Elemental tidak MediaStore boleh mengirim metrik ke Amazon CloudWatch di tingkat penampung. Selain itu, MediaStore harus mengirim metrik untuk objek dalam dua folder tertentu: `baseball/saturday` dan`football/saturday`. Metrik untuk MediaStore permintaan adalah sebagai berikut:

- Permintaan ke `baseball/saturday` folder memiliki CloudWatch dimensi`ObjectGroupName=baseballGroup`.
- Permintaan ke `football/saturday` folder memiliki dimensi`ObjectGroupName=footballGroup`.

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

## Contoh kebijakan metrik: Metrik tingkat kontainer dan tingkat jalur

Kebijakan contoh ini menunjukkan bahwa AWS Elemental MediaStore harus mengirim metrik ke Amazon CloudWatch di tingkat penampung. Selain itu, MediaStore harus mengirim metrik untuk objek dalam dua folder tertentu: `baseball/saturday` dan`football/saturday`. Metrik untuk MediaStore permintaan adalah sebagai berikut:

- Permintaan ke `baseball/saturday` folder memiliki CloudWatch dimensi`ObjectGroupName=baseballGroup`.

- Permintaan ke `football/saturday` folder memiliki CloudWatch dimensi`ObjectGroupName=footballGroup`.

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

## Contoh kebijakan metrik: Metrik tingkat jalur menggunakan wildcard

Kebijakan contoh ini menunjukkan bahwa AWS Elemental MediaStore harus mengirim metrik ke Amazon CloudWatch di tingkat penampung. Selain itu, juga MediaStore harus mengirim metrik untuk objek berdasarkan nama file mereka. Wildcard menunjukkan bahwa objek dapat disimpan di mana saja dalam wadah dan mereka dapat memiliki nama file apa pun, asalkan diakhiri dengan `.m3u8` ekstensi.

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

## Contoh kebijakan metrik: Metrik tingkat jalur dengan aturan yang tumpang tindih

Kebijakan contoh ini menunjukkan bahwa AWS Elemental MediaStore harus mengirim metrik ke Amazon CloudWatch di tingkat penampung. Selain itu, MediaStore harus mengirim metrik untuk dua folder: `sports/football/saturday` dan`sports/football`.

Metrik untuk MediaStore permintaan ke `sports/football/saturday` folder memiliki CloudWatch dimensi. `ObjectGroupName=footballGroup1` Karena objek yang disimpan dalam `sports/football` folder cocok dengan kedua aturan, CloudWatch menampilkan dua titik data untuk objek ini: satu dengan dimensi `ObjectGroupName=footballGroup1` dan yang kedua dengan dimensi`ObjectGroupName=footballGroup2`.

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

# Folder di AWS Elemental MediaStore

Folder adalah divisi dalam wadah. Anda menggunakan folder untuk membagi wadah Anda dengan cara yang sama seperti Anda membuat subfolder untuk membagi folder dalam sistem file. Anda dapat membuat hingga 10 tingkat folder (tidak termasuk wadah itu sendiri).

Folder bersifat opsional; Anda dapat memilih untuk mengunggah objek Anda langsung ke wadah alih-alih folder. Namun, folder adalah cara mudah untuk mengatur objek Anda.

Untuk mengunggah objek ke folder, Anda menentukan jalur ke folder. Jika folder sudah ada, AWS Elemental MediaStore menyimpan objek di folder. Jika folder tidak ada, layanan membuatnya, dan kemudian menyimpan objek di folder.

Misalnya, Anda memiliki wadah bernama`movies`, dan Anda mengunggah file bernama `mlaw.ts` dengan jalur`premium/canada`. AWS Elemental MediaStore menyimpan objek di subfolder canada di bawah folder premium. Jika tidak ada folder, layanan membuat `premium` folder dan `canada` subfolder, dan kemudian menyimpan objek Anda di `canada` subfolder. Jika Anda hanya menentukan wadah `movies` (tanpa jalur), layanan menyimpan objek langsung di wadah.

AWS Elemental MediaStore secara otomatis menghapus folder saat Anda menghapus objek terakhir di folder itu. Layanan ini juga menghapus folder kosong di atas folder itu. Misalnya, Anda memiliki folder bernama premium yang tidak berisi file apa pun tetapi berisi satu subfolder bernama`canada`. `canada`Subfolder berisi satu file bernama`mlaw.ts`. Jika Anda menghapus file`mlaw.ts`, layanan menghapus `canada` folder `premium` dan folder. Penghapusan otomatis ini hanya berlaku untuk folder. Layanan tidak menghapus wadah kosong.

Topik

- Aturan untuk nama folder
- Membuat folder
- Menghapus folder

## Aturan untuk nama folder

Ketika Anda memilih nama untuk folder Anda, ingat yang berikut:

- Nama hanya dapat berisi karakter berikut: huruf besar (A-Z), huruf kecil (a-z), angka (0-9), titik (.), tanda hubung (-), tildes (~), garis bawah (_), tanda sama (=), dan titik dua (:).

- Nama harus setidaknya satu karakter panjang. Nama folder kosong (sepertifolder1//folder3/) tidak diperbolehkan.
- Nilai peka huruf besar/kecil. Misalnya, Anda dapat memiliki folder bernama myFolder dan folder bernama myfolder dalam wadah atau folder yang sama karena nama-nama itu unik.
- Nama harus unik hanya dalam wadah atau folder induknya. Misalnya, Anda dapat membuat folder bernama myfolder dalam dua wadah yang berbeda: movies/myfolder dansports/myfolder.
- Nama dapat memiliki nama yang sama dengan wadah induknya.
- Folder tidak dapat diganti namanya setelah dibuat.

## Membuat folder

Anda dapat membuat folder saat mengunggah objek. Untuk mengunggah objek ke folder, Anda menentukan jalur ke folder. Jika folder sudah ada, AWS Elemental MediaStore menyimpan objek di folder. Jika folder tidak ada, layanan membuatnya, dan kemudian menyimpan objek di folder.

Untuk informasi selengkapnya, lihat the section called “Mengunggah Objek”.

## Menghapus folder

Anda dapat menghapus folder hanya jika folder kosong; Anda tidak dapat menghapus folder yang berisi objek.

AWS Elemental MediaStore secara otomatis menghapus folder saat Anda menghapus objek terakhir di folder itu. Layanan ini juga menghapus folder kosong di atas folder itu. Misalnya, Anda memiliki folder bernama premium yang tidak berisi file apa pun tetapi berisi satu subfolder bernamacanada. canadaSubfolder berisi satu file bernamamlaw.ts. Jika Anda menghapus filemlaw.ts, layanan menghapus canada folder premium dan folder. Penghapusan otomatis ini hanya berlaku untuk folder. Layanan tidak menghapus wadah kosong.

Lihat informasi yang lebih lengkap di Menghapus objek.

# Objek di AWS Elemental MediaStore

MediaStore Aset AWS Elemental disebut objek. Anda dapat mengunggah objek ke wadah atau ke folder di dalam wadah.

Di MediaStore, Anda dapat mengunggah, mengunduh, dan menghapus objek:

- Unggah - Tambahkan objek ke wadah atau folder. Ini tidak sama dengan membuat objek. Anda harus membuat objek Anda secara lokal sebelum Anda dapat mengunggahnya. MediaStore
- Download — Salin objek dari MediaStore lokasi lain. Ini tidak menghapus objek dari MediaStore.
- Hapus — Hapus objek dari MediaStore sepenuhnya. Anda dapat menghapus objek satu per satu, atau Anda dapat menambahkan kebijakan siklus hidup objek untuk secara otomatis menghapus objek dalam wadah setelah durasi yang ditentukan.

MediaStore menerima semua jenis file.

Topik


- Mengunggah Objek
- Melihat daftar objek
- Melihat detail suatu objek
- Mengunduh objek
- Menghapus objek


## Mengunggah Objek

Anda dapat mengunggah objek ke wadah atau ke folder dalam wadah. Untuk mengunggah objek ke folder, Anda menentukan jalur ke folder. Jika folder sudah ada, AWS Elemental MediaStore menyimpan objek di folder. Jika folder tidak ada, layanan membuatnya, dan kemudian menyimpan objek di folder. Untuk informasi selengkapnya tentang folder, lihatFolder di AWS Elemental MediaStore.

Anda dapat menggunakan MediaStore konsol atau AWS CLI untuk mengunggah objek.

MediaStore mendukung transfer objek yang terpotong, yang mengurangi latensi dengan membuat objek tersedia untuk diunduh saat masih diunggah. Untuk menggunakan kemampuan ini, atur

ketersediaan unggahan objek `kestreaming`. Anda dapat mengatur nilai header ini saat Anda mengunggah objek menggunakan API. Jika Anda tidak menentukan header ini dalam permintaan Anda, MediaStore tetapkan nilai default `standard` untuk ketersediaan unggahan objek.

Ukuran objek tidak boleh melebihi 25 MB untuk ketersediaan unggahan standar dan 10 MB untuk ketersediaan upload streaming.

> **Note**
>
> Nama file objek hanya dapat berisi huruf, angka, titik (.), garis bawah (_), tildes (~), tanda hubung (-), tanda sama dengan (=), dan titik dua (:).

Untuk mengunggah objek (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama wadah. Panel detail untuk wadah muncul.
3. Pilih Unggah objek.
4. Untuk jalur Target, ketik jalur untuk folder. Misalnya, `premium/canada`. Jika salah satu folder di jalur yang Anda tentukan belum ada, layanan akan membuatnya secara otomatis.
5. Di bagian Object, pilih Browse.
6. Arahkan ke folder yang sesuai, dan pilih satu objek untuk diunggah.
7. Pilih Buka, lalu pilih Unggah.

> **Note**
>
> Jika file dengan nama yang sama sudah ada di folder yang dipilih, layanan menggantikan file asli dengan file yang diunggah.

Untuk mengunggah objek (AWS CLI)

- Di AWS CLI, gunakan `put-object` perintah. Anda juga dapat menyertakan salah satu parameter berikut:`content-type`, `cache-control` (untuk memungkinkan pemanggil mengontrol perilaku cache objek), dan `path` (untuk menempatkan objek dalam folder dalam wadah).

> **Note**
>
> Setelah Anda mengunggah objek, Anda tidak dapat mengedit`content-type`,`cache-control`, atau`path`.

```
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 --region us-west-2
```

Contoh berikut menunjukkan nilai kembali:

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

## Melihat daftar objek

Anda dapat menggunakan MediaStore konsol AWS Elemental untuk melihat item (objek dan folder) yang disimpan di tingkat paling atas wadah atau dalam folder. Item yang disimpan dalam subfolder dari wadah atau folder saat ini tidak akan ditampilkan. Anda dapat menggunakan AWS CLI untuk melihat daftar objek dan folder dalam wadah, terlepas dari berapa banyak folder atau subfolder dalam wadah.

Untuk melihat daftar objek dalam wadah tertentu (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama wadah yang memiliki folder yang ingin Anda lihat.
3. Pilih nama folder dari daftar.

   Halaman detail muncul, menampilkan semua folder dan objek yang disimpan dalam folder.

Untuk melihat daftar objek dalam folder tertentu (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama wadah yang memiliki folder yang ingin Anda lihat.

   Halaman detail muncul, menampilkan semua folder dan objek yang disimpan dalam wadah.

Untuk melihat daftar objek dan folder dalam wadah tertentu (AWS CLI)

- Di AWS CLI, gunakan `list-items` perintah:

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

Contoh berikut menunjukkan nilai kembali:

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

> **Note**
>
> Objek yang tunduk pada `seconds_since_create` aturan tidak termasuk dalam `list-items` respons.

Untuk melihat daftar objek dan folder dalam folder tertentu (AWS CLI)

- Di AWS CLI, gunakan `list-items` perintah, dengan nama folder yang ditentukan di akhir permintaan:

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

Contoh berikut menunjukkan nilai kembali:

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

> **Note**
>
> Objek yang tunduk pada `seconds_since_create` aturan tidak termasuk dalam `list-items` respons.

## Melihat detail suatu objek

Setelah Anda mengunggah objek, AWS Elemental MediaStore menyimpan detail seperti tanggal modifikasi, panjang konten, ETag (tag entitas), dan jenis konten. Untuk mempelajari bagaimana metadata objek digunakan, lihat. MediaStoreInteraksi dengan cache HTTP

Untuk melihat detail objek (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Containers, pilih nama kontainer yang memiliki objek yang ingin Anda lihat.
3. Jika objek yang ingin Anda lihat ada di folder, lanjutkan memilih nama folder sampai Anda melihat objek.
4. Pilih nama objek.

   Halaman detail muncul, menampilkan informasi tentang objek.

Untuk melihat detail objek (AWS CLI)

- Di AWS CLI, gunakan `describe-object` perintah:

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

  Contoh berikut menunjukkan nilai kembali:

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

# Mengunduh objek

Anda dapat menggunakan konsol untuk mengunduh objek. Anda dapat menggunakan AWS CLI untuk mengunduh objek atau hanya sebagian dari objek.

Untuk mengunduh objek (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama wadah yang memiliki objek yang ingin Anda unduh.

3. Jika objek yang ingin Anda unduh ada di folder, lanjutkan memilih nama folder sampai Anda melihat objek.
4. Pilih nama objek.
5. Pada halaman Detail objek, pilih Unduh.

Untuk mengunduh objek (AWS CLI)

- Di AWS CLI, gunakan `get-object` perintah:

```
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
```

Contoh berikut menunjukkan nilai kembali:

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

Untuk men-download bagian dari objek (AWS CLI)

- Dalam AWS CLI, gunakan `get-object` perintah, dan tentukan rentang.

```
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
```

Contoh berikut menunjukkan nilai kembali:

```
{
    "StatusCode": 206,
    "ContentRange": "bytes 0-100/2307346",
    "ContentLength": "101",
    "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",
```

```
    "ContentType": "image/jpeg",
    "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f555555555555da6d3"
}
```

# Menghapus objek

AWS Elemental MediaStore menawarkan opsi berbeda untuk menghapus objek dari wadah:

- Hapus objek individual. Tidak ada biaya yang berlaku.
- Kosongkan wadah untuk menghapus semua objek dalam wadah sekaligus. Karena proses ini menggunakan panggilan API, biaya API normal berlaku.
- Tambahkan kebijakan siklus hidup objek untuk menghapus objek ketika mereka mencapai usia tertentu. Tidak ada biaya yang berlaku.

## Menghapus objek

Anda dapat menghapus objek satu per satu menggunakan konsol atau AWS CLI. Atau, Anda dapat menambahkan kebijakan siklus hidup objek untuk menghapus objek secara otomatis setelah mereka mencapai usia tertentu dalam wadah, atau Anda dapat mengosongkan wadah untuk menghapus semua objek dalam wadah tersebut.

> **Note**
>
> Saat Anda menghapus satu-satunya objek dalam folder, AWS Elemental MediaStore secara otomatis menghapus folder dan folder kosong apa pun di atas folder itu. Misalnya, Anda memiliki folder bernama `premium` yang tidak berisi file apa pun tetapi berisi satu subfolder bernama`canada`. `canada`Subfolder berisi satu file bernama`law.ts`. Jika Anda menghapus file`law.ts`, layanan menghapus `canada` folder `premium` dan folder.

Untuk menghapus objek (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih nama wadah yang memiliki objek yang ingin Anda hapus.
3. Jika objek yang ingin Anda hapus ada di folder, lanjutkan memilih nama folder sampai Anda melihat objek.

4. Pilih opsi di sebelah kiri nama objek.
5. Pilih Hapus.

Untuk menghapus objek (AWS CLI)

- Di AWS CLI, gunakan `delete-object` perintah.

  Contoh:

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

  Perintah ini tidak memiliki nilai kembali.

## Mengosongkan wadah

Anda dapat mengosongkan wadah untuk menghapus semua objek yang disimpan di dalam wadah. Atau, Anda dapat menambahkan kebijakan siklus hidup objek untuk menghapus objek secara otomatis setelah mencapai usia tertentu dalam wadah, atau Anda dapat menghapus objek satu per satu.

Untuk mengosongkan wadah (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.
2. Pada halaman Kontainer, pilih opsi untuk wadah yang ingin Anda kosongkan.
3. Pilih wadah kosong. Sebuah pesan konfirmasi akan muncul.
4. Konfirmasikan bahwa Anda ingin mengosongkan wadah dengan memasukkan nama kontainer ke dalam bidang teks, lalu pilih Kosong.

# Keamanan di AWS Elemental MediaStore

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. Model tanggung jawab bersama menjelaskan hal ini sebagai keamanan dari cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari Program AWS Kepatuhan Program AWS Kepatuhan . Untuk mempelajari tentang program kepatuhan yang berlaku untuk AWS Elemental MediaStore, lihat AWS Layanan dalam Lingkup oleh AWS Layanan Program Kepatuhan dalam Lingkup oleh Program .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, yang mencakup sensitivitas data Anda, persyaratan perusahaan Anda, serta undang-undang dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan MediaStore. Topik berikut menunjukkan cara mengonfigurasi MediaStore untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga belajar cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan MediaStore sumber daya Anda.

Topik


- Perlindungan data di AWS Elemental MediaStore
- Identity and Access Management untuk AWS Elemental MediaStore
- Penebangan dan pemantauan di AWS Elemental MediaStore
- Validasi kepatuhan untuk AWS Elemental MediaStore
- Ketahanan dalam AWS Elemental MediaStore
- Keamanan Infrastruktur di AWS Elemental MediaStore
- Pencegahan "confused deputy" lintas layanan

# Perlindungan data di AWS Elemental MediaStore

Model tanggung jawab AWS bersama model tanggung berlaku untuk perlindungan data di AWS Elemental MediaStore. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Lihat informasi yang lebih lengkap tentang privasi data dalam Pertanyaan Umum Privasi Data. Lihat informasi tentang perlindungan data di Eropa di pos blog Model Tanggung Jawab Bersama dan GDPR AWS di Blog Keamanan AWS .

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensil dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan AWS sumber daya. Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan logging aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang penggunaan CloudTrail jejak untuk menangkap AWS aktivitas, lihat Bekerja dengan CloudTrail jejak di AWS CloudTrail Panduan Pengguna.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola tingkat lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-3 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi selengkapnya tentang titik akhir FIPS yang tersedia di Standar Pemrosesan Informasi Federal (FIPS) 140-3.

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan MediaStore atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau AWS SDKs. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan

atau log diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

## Enkripsi data

MediaStore mengenkripsi wadah dan objek saat istirahat menggunakan algoritma AES-256 standar industri. Kami menyarankan Anda menggunakan MediaStore untuk mengamankan data Anda dengan cara berikut:

- Buat kebijakan kontainer untuk mengontrol hak akses ke semua folder dan objek dalam wadah itu. Untuk informasi selengkapnya, lihat the section called “Kebijakan kontainer”.
- Buat kebijakan berbagi sumber daya lintas asal (CORS) untuk memungkinkan akses lintas asal secara selektif ke sumber daya Anda. MediaStore Dengan CORS, Anda dapat mengizinkan aplikasi web klien yang dimuat dalam satu domain untuk berinteraksi dengan sumber daya di domain yang berbeda. Untuk informasi selengkapnya, lihat the section called “Kebijakan CORS”.

# Identity and Access Management untuk AWS Elemental MediaStore

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan sumber daya. MediaStore IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Topik

- Audiens
- Mengautentikasi dengan identitas
- Mengelola akses menggunakan kebijakan
- Bagaimana AWS Elemental MediaStore bekerja dengan IAM
- Contoh kebijakan berbasis identitas untuk AWS Elemental MediaStore
- Memecahkan masalah identitas dan akses AWS MediaStore Elemental

## Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan. MediaStore

Pengguna layanan — Jika Anda menggunakan MediaStore layanan untuk melakukan pekerjaan Anda, maka administrator Anda memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak MediaStore fitur untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara mengelola akses dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur di MediaStore, lihat Memecahkan masalah identitas dan akses AWS MediaStore Elemental.

Administrator layanan — Jika Anda bertanggung jawab atas MediaStore sumber daya di perusahaan Anda, Anda mungkin memiliki akses penuh ke MediaStore. Tugas Anda adalah menentukan MediaStore fitur dan sumber daya mana yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep dasar IAM. Untuk mempelajari lebih lanjut tentang bagaimana perusahaan Anda dapat menggunakan IAM MediaStore, lihatBagaimana AWS Elemental MediaStore bekerja dengan IAM.

Administrator IAM – Jika Anda adalah administrator IAM, Anda mungkin ingin belajar dengan lebih detail tentang cara Anda menulis kebijakan untuk mengelola akses ke MediaStore. Untuk melihat contoh kebijakan MediaStore berbasis identitas yang dapat Anda gunakan di IAM, lihat. Contoh kebijakan berbasis identitas untuk AWS Elemental MediaStore

## Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensi identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk AWS sebagai identitas federasi dengan menggunakan kredensi yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (IAM Identity Center), autentikasi masuk tunggal perusahaan Anda, dan kredensi Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas terfederasi, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal AWS akses. Untuk informasi selengkapnya tentang masuk AWS, lihat Cara masuk ke Panduan AWS Sign-In Pengguna Anda Akun AWS.

Jika Anda mengakses AWS secara terprogram, AWS sediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensil Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Guna mengetahui informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, lihat AWS Signature Version 4 untuk permintaan API dalam Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat Autentikasi multi-faktor dalam Panduan Pengguna AWS IAM Identity Center dan Autentikasi multi-faktor AWS di IAM dalam Panduan Pengguna IAM.

### Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihat Tugas yang memerlukan kredensial pengguna root dalam Panduan Pengguna IAM.

### Identitas gabungan

Sebagai praktik terbaik, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensi sementara.

Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, direktori Pusat Identitas AWS Directory Service, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensil yang disediakan melalui sumber identitas. Ketika identitas federasi mengakses Akun AWS, mereka mengambil peran, dan peran memberikan kredensi sementara.

Untuk manajemen akses terpusat, kami sarankan Anda menggunakan AWS IAM Identity Center. Anda dapat membuat pengguna dan grup di Pusat Identitas IAM, atau Anda dapat menghubungkan dan menyinkronkan ke sekumpulan pengguna dan grup di sumber identitas Anda sendiri untuk digunakan di semua aplikasi Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, lihat Apakah itu Pusat Identitas IAM? dalam Panduan Pengguna AWS IAM Identity Center .

## Pengguna dan grup IAM

Pengguna IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, kami merekomendasikan untuk mengandalkan kredensial sementara, bukan membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan tertentu yang memerlukan kredensial jangka panjang dengan pengguna IAM, kami merekomendasikan Anda merotasi kunci akses. Untuk informasi selengkapnya, lihat Merotasi kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensial jangka panjang dalam Panduan Pengguna IAM.

Grup IAM adalah identitas yang menentukan sekumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin bagi beberapa pengguna sekaligus. Grup mempermudah manajemen izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat meminta kelompok untuk menyebutkan IAMAdmins dan memberikan izin kepada grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari selengkapnya, lihat Kasus penggunaan untuk pengguna IAM dalam Panduan Pengguna IAM.

## Peran IAM

Peran IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Untuk mengambil peran IAM sementara AWS Management Console, Anda dapat beralih dari pengguna ke peran IAM (konsol). Anda dapat mengambil peran dengan memanggil operasi AWS CLI atau AWS API atau dengan menggunakan URL kustom. Untuk informasi selengkapnya tentang cara menggunakan peran, lihat Metode untuk mengambil peran dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Ketika identitas terfederasi mengautentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat Buat peran untuk penyedia identitas pihak ketiga dalam Panduan Pengguna IAM. Jika menggunakan Pusat Identitas IAM, Anda harus mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM akan mengorelasikan set izin ke peran dalam IAM. Untuk informasi tentang set izin, lihat Set izin dalam Panduan Pengguna AWS IAM Identity Center .
- Izin pengguna IAM sementara – Pengguna atau peran IAM dapat mengambil peran IAM guna mendapatkan berbagai izin secara sementara untuk tugas tertentu.
- Akses lintas akun – Anda dapat menggunakan peran IAM untuk mengizinkan seseorang (prinsipal tepercaya) di akun lain untuk mengakses sumber daya di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM dalam Panduan Pengguna IAM.
- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Misalnya, saat Anda melakukan panggilan dalam suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.
  - Sesi akses teruskan (FAS) — Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.
  - Peran layanan – Peran layanan adalah peran IAM yang dijalankan oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

  - Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke peran layanan. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.
- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan peran IAM untuk mengelola kredensi sementara untuk aplikasi yang berjalan pada EC2 instance dan membuat AWS CLI atau AWS permintaan API. Ini lebih baik untuk menyimpan kunci akses dalam EC2 instance. Untuk menetapkan AWS peran ke EC2 instance dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instance yang dilampirkan ke instance. Profil instance berisi peran dan memungkinkan program yang berjalan pada EC2 instance untuk mendapatkan kredensi sementara. Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di EC2 instans Amazon di Panduan Pengguna IAM.

## Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal (pengguna, pengguna root, atau sesi peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, lihat Gambaran umum kebijakan JSON dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasinya. Misalnya, anggaplah Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut bisa mendapatkan informasi peran dari AWS Management Console, API AWS CLI, atau AWS API.

## Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Tentukan izin IAM kustom dengan kebijakan terkelola pelanggan dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan yang dikelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan terkelola adalah kebijakan mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran dalam. Akun AWS Kebijakan AWS terkelola mencakup kebijakan terkelola dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan yang dikelola atau kebijakan inline, lihat Pilih antara kebijakan yang dikelola dan kebijakan inline dalam Panduan Pengguna IAM.

## Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola dari IAM dalam kebijakan berbasis sumber daya.

## Daftar kontrol akses (ACLs)

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung. ACLs Untuk mempelajari selengkapnya ACLs, lihat Ringkasan daftar kontrol akses (ACL) di Panduan Pengembang Layanan Penyimpanan Sederhana Amazon.

## Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Jenis-jenis kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh jenis kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan tempat Anda mengatur izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke entitas IAM (pengguna IAM atau peran IAM). Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh batasan izin. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya tentang batasan izin, lihat Batasan izin untuk entitas IAM dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCPs) — SCPs adalah kebijakan JSON yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU) di. AWS Organizations AWS Organizations adalah layanan untuk mengelompokkan dan mengelola secara terpusat beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur dalam suatu organisasi, maka Anda dapat menerapkan kebijakan kontrol layanan (SCPs) ke salah satu atau semua akun Anda. SCP membatasi izin untuk entitas di akun anggota, termasuk masing-masing. Pengguna root akun AWS Untuk informasi selengkapnya tentang Organizations dan SCPs, lihat Kebijakan kontrol layanan di Panduan AWS Organizations Pengguna.
- Kebijakan kontrol sumber daya (RCPs) — RCPs adalah kebijakan JSON yang dapat Anda gunakan untuk menetapkan izin maksimum yang tersedia untuk sumber daya di akun Anda tanpa memperbarui kebijakan IAM yang dilampirkan ke setiap sumber daya yang Anda miliki. RCP membatasi izin untuk sumber daya di akun anggota dan dapat memengaruhi izin efektif untuk identitas, termasuk Pengguna root akun AWS, terlepas dari apakah itu milik organisasi Anda. Untuk informasi selengkapnya tentang Organizations dan RCPs, termasuk daftar dukungan Layanan AWS tersebut RCPs, lihat Kebijakan kontrol sumber daya (RCPs) di Panduan AWS Organizations Pengguna.
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara programatis untuk peran atau pengguna terfederasi. Izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya, lihat Kebijakan sesi dalam Panduan Pengguna IAM.

### Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat Logika evaluasi kebijakan di Panduan Pengguna IAM.

## Bagaimana AWS Elemental MediaStore bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses MediaStore, pelajari fitur IAM yang tersedia untuk digunakan. MediaStore

Fitur IAM yang dapat Anda gunakan dengan AWS Elemental MediaStore

| Fitur IAM | MediaStore dukungan |
|---|---|
| Kebijakan berbasis identitas | Ya |
| Kebijakan berbasis sumber daya | Ya |
| Tindakan kebijakan | Ya |
| Sumber daya kebijakan | Ya |
| kunci-kunci persyaratan kebijakan (spesifik layanan) | Ya |
| ACLs | Tidak |
| ABAC (tanda dalam kebijakan) | Parsial |
| Kredensial sementara | Ya |
| Izin principal | Ya |
| Peran layanan | Ya |
| Peran terkait layanan | Tidak |

Untuk mendapatkan tampilan tingkat tinggi tentang cara MediaStore dan AWS layanan lain bekerja dengan sebagian besar fitur IAM, lihat AWS layanan yang bekerja dengan IAM di Panduan Pengguna IAM.

## Kebijakan berbasis identitas untuk MediaStore

Mendukung kebijakan berbasis identitas: Ya

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Tentukan izin IAM kustom dengan kebijakan terkelola pelanggan dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta kondisi yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Anda tidak dapat menentukan secara spesifik prinsipal dalam sebuah kebijakan berbasis identitas karena prinsipal berlaku bagi pengguna atau peran yang melekat kepadanya. Untuk mempelajari semua elemen yang dapat Anda gunakan dalam kebijakan JSON, lihat Referensi elemen kebijakan JSON IAM dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk MediaStore

Untuk melihat contoh kebijakan MediaStore berbasis identitas, lihat. Contoh kebijakan berbasis identitas untuk AWS Elemental MediaStore

## Kebijakan berbasis sumber daya dalam MediaStore

Mendukung kebijakan berbasis sumber daya: Ya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan secara spesifik seluruh akun atau entitas IAM di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya. Menambahkan prinsipal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsipal dan sumber daya berbeda Akun AWS, administrator IAM di akun tepercaya juga harus memberikan izin entitas utama (pengguna atau peran) untuk mengakses sumber daya. Mereka memberikan izin dengan melampirkan kebijakan berbasis identitas kepada entitas. Namun, jika kebijakan berbasis sumber daya memberikan akses ke principal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk informasi selengkapnya, lihat Akses sumber daya lintas akun di IAM dalam Panduan Pengguna IAM.

> **Note**
>
> MediaStore juga mendukung kebijakan kontainer yang menentukan entitas utama mana (akun, pengguna, peran, dan pengguna federasi) yang dapat melakukan tindakan pada penampung. Untuk informasi selengkapnya, lihat Kebijakan kontainer.

## Tindakan kebijakan untuk MediaStore

Mendukung tindakan kebijakan: Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan operasi AWS API terkait. Ada beberapa pengecualian, misalnya tindakan hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Sertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Untuk melihat daftar MediaStore tindakan, lihat Tindakan yang ditentukan oleh AWS Elemental MediaStore dalam Referensi Otorisasi Layanan.

Tindakan kebijakan MediaStore menggunakan awalan berikut sebelum tindakan:

```
mediastore
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan tersebut dengan koma.

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

Untuk melihat contoh kebijakan MediaStore berbasis identitas, lihat. Contoh kebijakan berbasis identitas untuk AWS Elemental MediaStore

## Sumber daya kebijakan untuk MediaStore

Mendukung sumber daya kebijakan: Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek yang menjadi target penerapan tindakan. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`. Praktik terbaiknya, tentukan sumber daya menggunakan Amazon Resource Name (ARN). Anda dapat melakukan ini untuk tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

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

Untuk melihat daftar jenis MediaStore sumber daya dan jenisnya ARNs, lihat Sumber Daya yang ditentukan oleh AWS Elemental MediaStore dalam Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat Tindakan yang ditentukan oleh AWS Elemental. MediaStore

Sumber daya MediaStore kontainer memiliki ARN berikut:

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

Untuk informasi selengkapnya tentang format ARNs, lihat Amazon Resource Names (ARNs) dan Ruang Nama AWS Layanan.

Misalnya, untuk menentukan `AwardsShow` wadah dalam pernyataan Anda, gunakan ARN berikut:

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

## Kunci kondisi kebijakan untuk MediaStore

Mendukung kunci kondisi kebijakan khusus layanan: Yes

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan operator kondisi, misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam sebuah pernyataan, atau beberapa kunci dalam elemen `Condition` tunggal, maka AWS akan mengevaluasinya menggunakan operasi `AND` logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan `OR` operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Sebagai contoh, Anda dapat memberikan izin kepada pengguna IAM untuk mengakses sumber daya hanya jika izin tersebut mempunyai tanda yang sesuai dengan nama pengguna IAM mereka. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM: variabel dan tanda dalam Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi AWS global, lihat kunci konteks kondisi AWS global di Panduan Pengguna IAM.

Untuk melihat daftar kunci kondisi, lihat Kunci MediaStore kondisi untuk AWS Elemental MediaStore di Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat Tindakan yang ditentukan oleh AWS Elemental MediaStore.

Untuk melihat contoh kebijakan MediaStore berbasis identitas, lihat. Contoh kebijakan berbasis identitas untuk AWS Elemental MediaStore

## ACLs di MediaStore

Mendukung ACLs: Tidak

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

## ABAC dengan MediaStore

Mendukung ABAC (tag dalam kebijakan): Sebagian

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke entitas IAM (pengguna atau peran) dan ke banyak AWS sumber daya. Penandaan ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi ketika tanda milik prinsipal cocok dengan tanda yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna di situasi saat manajemen kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tanda, berikan informasi tentang tanda di elemen kondisi dari kebijakan menggunakan kunci kondisi `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, atau `aws:TagKeys`.

Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya tentang ABAC, lihat Tentukan izin dengan otorisasi ABAC dalam Panduan Pengguna IAM. Untuk melihat tutorial yang menguraikan langkah-langkah pengaturan ABAC, lihat Menggunakan kontrol akses berbasis atribut (ABAC) dalam Panduan Pengguna IAM.

## Menggunakan kredensi sementara dengan MediaStore

Mendukung kredensial sementara: Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensil sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensi sementara, lihat Layanan AWS yang bekerja dengan IAM di Panduan Pengguna IAM.

Anda menggunakan kredensi sementara jika Anda masuk AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS menggunakan tautan masuk tunggal (SSO) perusahaan Anda, proses tersebut secara otomatis membuat kredensil sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna lalu beralih peran. Untuk informasi selengkapnya tentang peralihan peran, lihat Beralih dari pengguna ke peran IAM (konsol) dalam Panduan Pengguna IAM.

Anda dapat membuat kredensil sementara secara manual menggunakan API AWS CLI atau AWS . Anda kemudian dapat menggunakan kredensil sementara tersebut untuk mengakses. AWS AWS merekomendasikan agar Anda secara dinamis menghasilkan kredensi sementara alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat Kredensial keamanan sementara di IAM.

## Izin utama lintas layanan untuk MediaStore

Mendukung sesi akses maju (FAS): Ya

Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.

## Peran layanan untuk MediaStore

Mendukung peran layanan: Ya

Peran layanan adalah peran IAM yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

> **Warning**
>
> Mengubah izin untuk peran layanan dapat merusak MediaStore fungsionalitas. Edit peran layanan hanya jika MediaStore memberikan panduan untuk melakukannya.

### Peran terkait layanan untuk MediaStore

Mendukung peran terkait layanan: Tidak

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Untuk detail tentang pembuatan atau manajemen peran terkait layanan, lihat Layanan AWS yang berfungsi dengan IAM. Cari layanan dalam tabel yang memiliki `Yes` di kolom Peran terkait layanan. Pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

## Contoh kebijakan berbasis identitas untuk AWS Elemental MediaStore

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau mengubah sumber daya MediaStore. Mereka juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM dengan menggunakan contoh dokumen kebijakan JSON ini, lihat Membuat kebijakan IAM (konsol) di Panduan Pengguna IAM.

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh MediaStore, termasuk format ARNs untuk setiap jenis sumber daya, lihat Kunci tindakan, sumber daya, dan kondisi untuk AWS Elemental MediaStore dalam Referensi Otorisasi Layanan.

Topik

- Praktik terbaik kebijakan
- Menggunakan konsol MediaStore
- Mengizinkan pengguna melihat izin mereka sendiri

## Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus MediaStore sumber daya di akun Anda. Tindakan ini membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat Kebijakan yang dikelola AWS atau Kebijakan yang dikelola AWS untuk fungsi tugas dalam Panduan Pengguna IAM.
- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat Kebijakan dan izin dalam IAM dalam Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Kondisi dalam Panduan Pengguna IAM.
- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat Validasi kebijakan dengan IAM Access Analyzer dalam Panduan Pengguna IAM.
- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan.

Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Amankan akses API dengan MFA dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat Praktik terbaik keamanan di IAM dalam Panduan Pengguna IAM.

### Menggunakan konsol MediaStore

Untuk mengakses MediaStore konsol AWS Elemental, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang MediaStore sumber daya di Anda Akun AWS. Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna atau peran) dengan kebijakan tersebut.

Anda tidak perlu mengizinkan izin konsol minimum untuk pengguna yang melakukan panggilan hanya ke AWS CLI atau AWS API. Sebagai gantinya, izinkan akses hanya ke tindakan yang sesuai dengan operasi API yang coba mereka lakukan.

Untuk memastikan bahwa pengguna dan peran masih dapat menggunakan MediaStore konsol, lampirkan juga kebijakan MediaStore *ConsoleAccess* atau *ReadOnly* AWS terkelola ke entitas. Untuk informasi selengkapnya, lihat Menambah izin untuk pengguna dalam Panduan Pengguna IAM.

### Mengizinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan ini di konsol atau menggunakan API atau secara terprogram. AWS CLI AWS

```
{
    "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": "*"
        }
    ]
}
```

## Memecahkan masalah identitas dan akses AWS MediaStore Elemental

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan MediaStore dan IAM.

Topik

- Saya tidak berwenang untuk melakukan tindakan di MediaStore
- Saya tidak berwenang untuk melakukan iam: PassRole
- Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses MediaStore sumber daya saya

### Saya tidak berwenang untuk melakukan tindakan di MediaStore

Jika Anda menerima pesan kesalahan bahwa Anda tidak memiliki otorisasi untuk melakukan tindakan, kebijakan Anda harus diperbarui agar Anda dapat melakukan tindakan tersebut.

Contoh kesalahan berikut terjadi ketika pengguna IAM `mateojackson` mencoba menggunakan konsol untuk melihat detail tentang suatu sumber daya *`my-example-widget`* rekaan, tetapi tidak memiliki izin `mediastore:`*`GetWidget`* rekaan.

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

Dalam hal ini, kebijakan untuk pengguna `mateojackson` harus diperbarui untuk mengizinkan akses ke sumber daya *`my-example-widget`* dengan menggunakan tindakan `mediastore:`*`GetWidget`*.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

### Saya tidak berwenang untuk melakukan iam: PassRole

Jika Anda menerima kesalahan yang tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran MediaStore.

Beberapa Layanan AWS memungkinkan Anda untuk meneruskan peran yang ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait layanan. Untuk melakukannya, Anda harus memiliki izin untuk meneruskan peran ke layanan.

Contoh kesalahan berikut terjadi ketika pengguna IAM bernama `marymajor` mencoba menggunakan konsol tersebut untuk melakukan tindakan di MediaStore. Namun, tindakan tersebut memerlukan layanan untuk mendapatkan izin yang diberikan oleh peran layanan. Mary tidak memiliki izin untuk meneruskan peran tersebut pada layanan.

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

Dalam kasus ini, kebijakan Mary harus diperbarui agar dia mendapatkan izin untuk melakukan tindakan `iam:PassRole` tersebut.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

### Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses MediaStore sumber daya saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang

dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACLs), Anda dapat menggunakan kebijakan tersebut untuk memberi orang akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa referensi berikut:

- Untuk mempelajari apakah MediaStore mendukung fitur-fitur ini, lihatBagaimana AWS Elemental MediaStore bekerja dengan IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda di seluruh sumber daya Akun AWS yang Anda miliki, lihat Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki di Panduan Pengguna IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda kepada pihak ketiga Akun AWS, lihat Menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat Menyediakan akses ke pengguna terautentikasi eksternal (federasi identitas) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara menggunakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM.

# Penebangan dan pemantauan di AWS Elemental MediaStore

Bagian ini memberikan gambaran umum tentang opsi untuk masuk dan memantau AWS Elemental MediaStore untuk tujuan keamanan. Untuk informasi selengkapnya tentang pencatatan dan pemantauan MediaStore, lihatPemantauan dan penandaan di AWS Elemental MediaStore.

Pemantauan adalah bagian penting dari menjaga keandalan, ketersediaan, dan kinerja AWS Elemental MediaStore dan AWS solusi Anda. Anda harus mengumpulkan data pemantauan dari semua bagian AWS solusi Anda sehingga Anda dapat lebih mudah men-debug kegagalan multi-titik jika terjadi. AWS menyediakan beberapa alat untuk memantau MediaStore sumber daya Anda dan menanggapi potensi insiden.

## CloudWatch Alarm Amazon

Menggunakan CloudWatch alarm, Anda menonton satu metrik selama periode waktu yang Anda tentukan. Jika metrik melebihi ambang batas tertentu, notifikasi akan dikirim ke topik Amazon SNS atau kebijakan AWS Auto Scaling. CloudWatch alarm tidak memanggil tindakan karena mereka

berada dalam keadaan tertentu. Sebaliknya, negara harus telah berubah dan dipertahankan untuk sejumlah periode tertentu. Untuk informasi selengkapnya, lihat Pemantauan CloudWatch dengan.

### AWS CloudTrail log

CloudTrail menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di AWS Elemental MediaStore. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat MediaStore, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan. Untuk informasi selengkapnya, lihat Membuat log panggilan API dengan CloudTrail.

### AWS Trusted Advisor

Trusted Advisor mengacu pada praktik terbaik yang dipelajari dari melayani ratusan ribu AWS pelanggan. Trusted Advisor memeriksa lingkungan AWS Anda dan kemudian membuat rekomendasi ketika ada peluang untuk menghemat uang, meningkatkan ketersediaan dan kinerja sistem, atau membantu menutup celah keamanan. Semua AWS pelanggan memiliki akses ke lima cek Trusted Advisor. Pelanggan dengan paket dukungan Bisnis atau Perusahaan dapat melihat semua Trusted Advisor pemeriksaan.

Untuk informasi selengkapnya, lihat AWS Trusted Advisor.

## Validasi kepatuhan untuk AWS Elemental MediaStore

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat Program AWS Kepatuhan Program AWS .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Kepatuhan dan Tata Kelola Keamanan – Panduan implementasi solusi ini membahas pertimbangan arsitektur serta memberikan langkah-langkah untuk menerapkan fitur keamanan dan kepatuhan.

- Referensi Layanan yang Memenuhi Syarat HIPAA — Daftar layanan yang memenuhi syarat HIPAA. Tidak semua memenuhi Layanan AWS syarat HIPAA.
- AWS Sumber Daya AWS — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- AWS Panduan Kepatuhan Pelanggan - Memahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan untuk kontrol keamanan di berbagai kerangka kerja (termasuk Institut Standar dan Teknologi Nasional (NIST), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi (ISO)).
- Mengevaluasi Sumber Daya dengan Aturan dalam Panduan AWS Config Pengembang — AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- AWS Security Hub— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS. Security Hub menggunakan kontrol keamanan untuk sumber daya AWS Anda serta untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat Referensi kontrol Security Hub.
- Amazon GuardDuty — Ini Layanan AWS mendeteksi potensi ancaman terhadap beban kerja Akun AWS, kontainer, dan data Anda dengan memantau lingkungan Anda untuk aktivitas mencurigakan dan berbahaya. GuardDuty dapat membantu Anda mengatasi berbagai persyaratan kepatuhan, seperti PCI DSS, dengan memenuhi persyaratan deteksi intrusi yang diamanatkan oleh kerangka kerja kepatuhan tertentu.
- AWS Audit ManagerIni Layanan AWS membantu Anda terus mengaudit AWS penggunaan Anda untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap peraturan dan standar industri.

# Ketahanan dalam AWS Elemental MediaStore

Infrastruktur AWS global dibangun di sekitar Wilayah AWS dan Availability Zones. Wilayah AWS menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan. Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan fail over di antara zona tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur pusat data tunggal atau multi tradisional.

Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat Infrastruktur AWS Global.

Selain infrastruktur AWS global, MediaStore menawarkan beberapa fitur untuk membantu mendukung ketahanan data dan kebutuhan cadangan Anda.

## Keamanan Infrastruktur di AWS Elemental MediaStore

Sebagai layanan terkelola, AWS Elemental MediaStore dilindungi oleh keamanan jaringan AWS global. Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat Keamanan AWS Cloud. Untuk mendesain AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat Perlindungan Infrastruktur dalam Kerangka Kerja yang AWS Diarsiteksikan dengan Baik Pilar Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses MediaStore melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Pengangkutan (TLS). Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Sandi cocok dengan sistem kerahasiaan maju sempurna (perfect forward secrecy, PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan prinsipal IAM. Atau Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk menghasilkan kredensial keamanan sementara untuk menandatangani permintaan.

## Pencegahan "confused deputy" lintas layanan

Masalah "confused deputy" adalah masalah keamanan saat entitas yang tidak memiliki izin untuk melakukan suatu tindakan dapat memaksa entitas yang memilik hak akses lebih tinggi untuk melakukan tindakan tersebut. Pada tahun AWS, peniruan lintas layanan dapat mengakibatkan masalah wakil yang membingungkan. Peniruan identitas lintas layanan dapat terjadi ketika satu layanan (layanan yang dipanggil) memanggil layanan lain (layanan yang dipanggil). Layanan pemanggilan dapat dimanipulasi menggunakan izinnya untuk bertindak pada sumber daya pelanggan lain dengan cara yang seharusnya tidak dilakukannya kecuali bila memiliki izin untuk mengakses. Untuk mencegah hal ini, AWS menyediakan alat yang membantu Anda melindungi data untuk semua layanan dengan principal layanan yang telah diberi akses ke sumber daya di akun Anda.

Sebaiknya gunakan kunci konteks kondisi `aws:SourceAccount`global `aws:SourceArn`dan global dalam kebijakan sumber daya untuk membatasi izin yang diberikan AWS MediaStore Elemental layanan lain ke sumber daya. Gunakan `aws:SourceArn` jika Anda ingin hanya satu sumber daya yang akan dikaitkan dengan akses lintas layanan. Gunakan `aws:SourceAccount` jika Anda ingin mengizinkan sumber daya apa pun di akun tersebut dikaitkan dengan penggunaan lintas layanan.

Cara paling efektif untuk melindungi dari masalah "confused deputy" adalah dengan menggunakan kunci konteks kondisi global `aws:SourceArn` dengan ARN lengkap sumber daya. Jika Anda tidak mengetahui ARN lengkap sumber daya atau jika Anda menentukan beberapa sumber daya, gunakan kunci kondisi konteks global `aws:SourceArn` dengan karakter wildcard (*) untuk bagian ARN yang tidak diketahui. Misalnya, `arn:aws:`*`servicename`*`:*:`*`123456789012`*`:*`.

Jika nilai `aws:SourceArn` tidak berisi ID akun, seperti ARN bucket Amazon S3, Anda harus menggunakan kedua kunci konteks kondisi global tersebut untuk membatasi izin.

Nilai `aws:SourceArn` harus berupa konfigurasi yang MediaStore menerbitkan CloudWatch log di Wilayah dan akun Anda.

Contoh berikut menunjukkan bagaimana Anda dapat menggunakan kunci konteks kondisi `aws:SourceAccount` global `aws:SourceArn` dan MediaStore untuk mencegah masalah wakil yang membingungkan.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "servicename.amazonaws.com"
    },
    "Action": "servicename:ActionName",
    "Resource": [
      "arn:aws:servicename:::ResourceName/*"
    ],
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:servicename:*:123456789012:*"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
```

```
      }
    }
}
```

# Pemantauan dan penandaan di AWS Elemental MediaStore

Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja AWS Elemental MediaStore dan solusi Anda yang lain AWS . AWS menyediakan alat pemantauan berikut untuk menonton MediaStore, melaporkan ketika ada sesuatu yang salah, dan mengambil tindakan otomatis bila perlu:

- AWS CloudTrailmenangkap panggilan API dan peristiwa terkait yang dibuat oleh atau atas nama AWS akun Anda dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Anda dapat mengidentifikasi pengguna dan akun mana yang dipanggil AWS, alamat IP sumber dari mana panggilan dilakukan, dan kapan panggilan terjadi. Untuk informasi selengkapnya, silakan lihat Panduan Pengguna AWS CloudTrail.
- Amazon CloudWatch memantau AWS sumber daya Anda dan aplikasi yang Anda jalankan AWS secara real time. Anda dapat mengumpulkan dan melacak metrik, membuat dasbor yang disesuaikan, dan mengatur alarm yang memberi tahu Anda atau mengambil tindakan saat metrik tertentu mencapai ambang batas yang ditentukan. Misalnya, Anda dapat CloudWatch melacak penggunaan CPU atau metrik lain dari EC2 instans Amazon Anda dan secara otomatis meluncurkan instans baru bila diperlukan. Untuk informasi selengkapnya, lihat Panduan CloudWatch Pengguna Amazon.
- Amazon CloudWatch Events memberikan aliran peristiwa sistem yang menjelaskan perubahan AWS sumber daya. Biasanya, AWS layanan mengirimkan pemberitahuan acara ke CloudWatch Acara dalam hitungan detik tetapi terkadang dapat memakan waktu satu menit atau lebih lama. CloudWatch Peristiwa memungkinkan komputasi berbasis peristiwa otomatis, karena Anda dapat menulis aturan yang mengawasi peristiwa tertentu dan memicu tindakan otomatis di AWS layanan lain saat peristiwa ini terjadi. Untuk informasi selengkapnya, lihat Panduan Pengguna CloudWatch Acara Amazon.
- Amazon CloudWatch Logs memungkinkan Anda memantau, menyimpan, dan mengakses file log Anda dari EC2 instans Amazon CloudTrail, dan sumber lainnya. CloudWatch Log dapat memantau informasi dalam file log dan memberi tahu Anda ketika ambang batas tertentu terpenuhi. Anda juga dapat mengarsipkan data log dalam penyimpanan yang sangat durabel. Untuk informasi selengkapnya, lihat Panduan Pengguna Amazon CloudWatch Logs.

Anda juga dapat menetapkan metadata ke MediaStore kontainer Anda dalam bentuk tag. Setiap tag adalah label yang terdiri dari kunci dan nilai yang Anda tentukan. Tag dapat membuatnya lebih mudah untuk mengelola, mencari, dan memfilter sumber daya. Anda dapat menggunakan tag

untuk mengatur AWS sumber daya di Konsol AWS Manajemen, membuat laporan penggunaan dan penagihan di semua AWS sumber daya, dan memfilter sumber daya selama aktivitas otomatisasi infrastruktur.

Topik


- Mencatat panggilan AWS Elemental MediaStore API dengan AWS CloudTrail
- Memantau AWS Elemental MediaStore dengan Amazon CloudWatch
- Menandai sumber daya MediaStore AWS Elemental


# Mencatat panggilan AWS Elemental MediaStore API dengan AWS CloudTrail

AWS Elemental MediaStore terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di dalamnya. MediaStore CloudTrail menangkap subset panggilan API untuk MediaStore sebagai peristiwa, termasuk panggilan dari MediaStore konsol dan dari panggilan kode ke API. MediaStore Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara berkelanjutan ke bucket Amazon S3, termasuk acara untuk. MediaStore Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru di CloudTrail konsol dalam Riwayat acara. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat MediaStore, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan banyak lagi.

Untuk mempelajari selengkapnya CloudTrail, termasuk cara mengonfigurasi dan mengaktifkannya, lihat Panduan AWS CloudTrail Pengguna.

Topik


- Informasi AWS Elemental MediaStore di CloudTrail
- Contoh: entri file MediaStore log AWS Elemental


## Informasi AWS Elemental MediaStore di CloudTrail

CloudTrail diaktifkan di AWS akun Anda saat Anda membuat akun. Ketika aktivitas peristiwa yang didukung terjadi di AWS Elemental MediaStore, aktivitas tersebut direkam dalam suatu CloudTrail peristiwa bersama dengan peristiwa AWS layanan lainnya dalam riwayat Acara. Anda dapat melihat, mencari, dan mengunduh acara terbaru di AWS akun Anda. Untuk informasi selengkapnya, lihat Melihat Acara dengan Riwayat CloudTrail Acara.

Untuk catatan peristiwa yang sedang berlangsung di AWS akun Anda, termasuk acara untuk MediaStore, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Secara default, saat Anda membuat jejak di konsol tersebut, jejak diterapkan ke semua Wilayah AWS . Jejak mencatat peristiwa dari semua Wilayah di AWS partisi dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat topik berikut:

- Gambaran Umum untuk Membuat Jejak
- CloudTrail Layanan dan Integrasi yang Didukung
- Mengonfigurasi Notifikasi Amazon SNS untuk CloudTrail
- Menerima File CloudTrail Log dari Beberapa Wilayah dan Menerima File CloudTrail Log dari Beberapa Akun

AWS Elemental MediaStore mendukung pencatatan operasi berikut sebagai peristiwa dalam file CloudTrail log:

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

Setiap entri peristiwa atau log berisi informasi tentang entitas yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut ini:

- Apakah permintaan dibuat dengan pengguna root atau kredensi pengguna
- Baik permintaan tersebut dibuat dengan kredensial keamanan sementara untuk peran atau pengguna gabungan

- Apakah permintaan itu dibuat oleh AWS layanan lain

Untuk informasi lain, lihat Elemen userIdentity CloudTrail.

## Contoh: entri file MediaStore log AWS Elemental

Trail adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket Amazon S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber apa pun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan seterusnya. File log CloudTrail bukan merupakan jejak tumpukan terurut dari panggilan API publik, sehingga file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `CreateContainer` operasi:


```
{
    "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"
}
```

# Memantau AWS Elemental MediaStore dengan Amazon CloudWatch

Anda dapat memantau MediaStore penggunaan AWS Elemental CloudWatch, yang mengumpulkan data mentah dan memprosesnya menjadi metrik yang dapat dibaca. CloudWatch menyimpan statistik selama 15 bulan sehingga Anda dapat mengakses informasi historis dan mendapatkan perspektif yang lebih baik tentang kinerja aplikasi atau layanan web Anda. Anda juga dapat mengatur alarm yang memperhatikan ambang batas tertentu dan mengirim notifikasi atau mengambil tindakan saat ambang batas tersebut terpenuhi. Untuk informasi selengkapnya, lihat Panduan CloudWatch Pengguna Amazon.

AWS menyediakan alat pemantauan berikut untuk menonton MediaStore, melaporkan ketika ada sesuatu yang salah, dan mengambil tindakan otomatis bila perlu:

- Amazon CloudWatch Logs memungkinkan Anda memantau, menyimpan, dan mengakses file log Anda dari AWS layanan seperti AWS Elemental MediaStore. Anda dapat menggunakan CloudWatch Log untuk memantau aplikasi dan sistem menggunakan data log. Misalnya, CloudWatch Log dapat melacak jumlah kesalahan yang terjadi di log aplikasi Anda dan mengirimi Anda pemberitahuan setiap kali tingkat kesalahan melebihi ambang batas yang Anda tentukan. CloudWatch Log menggunakan data log Anda untuk pemantauan, sehingga tidak diperlukan perubahan kode. Misalnya, Anda dapat memantau log aplikasi untuk istilah literal tertentu (seperti "ValidationException“) atau menghitung jumlah `PutObject` permintaan yang dibuat selama periode waktu tertentu. Ketika istilah yang Anda cari ditemukan, CloudWatch Log melaporkan data ke CloudWatch metrik yang Anda tentukan. Data log dienkripsi saat transit dan saat diam.

- Amazon CloudWatch Events memberikan peristiwa sistem yang menjelaskan perubahan AWS sumber daya, seperti MediaStore objek. Biasanya, AWS layanan mengirimkan pemberitahuan acara ke CloudWatch Acara dalam hitungan detik tetapi terkadang dapat memakan waktu satu menit atau lebih lama. Anda dapat mengatur aturan untuk mencocokkan peristiwa (seperti `DeleteObject` permintaan) dan merutekkannya ke satu atau beberapa fungsi atau aliran target. CloudWatch Peristiwa menjadi sadar akan perubahan operasional saat terjadi. Selain itu, CloudWatch Peristiwa merespons perubahan operasional ini dan mengambil tindakan korektif seperlunya, dengan mengirim pesan untuk merespons lingkungan, mengaktifkan fungsi, membuat perubahan, dan menangkap informasi negara.

## CloudWatch Log

Access logging menyediakan catatan rinci untuk permintaan yang dibuat ke objek dalam wadah. Log akses berguna untuk banyak aplikasi, seperti audit keamanan dan akses. Mereka juga dapat membantu Anda mempelajari basis pelanggan Anda dan memahami MediaStore tagihan Anda. CloudWatch Log dikategorikan sebagai berikut:

- Pengaliran log adalah urutan log acara yang berbagi sumber yang sama.
- Grup log adalah grup log stream yang berbagi pengaturan retensi, pemantauan, dan kontrol akses yang sama. Saat Anda mengaktifkan akses masuk pada wadah, MediaStore buat grup log dengan nama seperti`/aws/mediastore/MyContainerName`. Anda dapat menentukan grup log dan menentukan pengaliran untuk dimasukkan ke dalam setiap grup. Tidak ada kuota pada jumlah pengaliran log yang dapat menjadi milik satu grup log.

Secara default, log disimpan tanpa batas waktu dan tidak pernah kedaluwarsa. Anda dapat menyesuaikan kebijakan penyimpanan untuk setiap grup log, menjaga retensi tidak terbatas, atau memilih periode retensi dari satu hari hingga 10 tahun.

### Menyiapkan izin untuk Amazon CloudWatch

Gunakan AWS Identity and Access Management (IAM) untuk membuat peran yang memberi AWS MediaStore Elemental akses ke Amazon. CloudWatch Anda harus melakukan langkah-langkah ini agar CloudWatch Log dipublikasikan untuk akun Anda. CloudWatch secara otomatis menerbitkan metrik untuk akun Anda.

Untuk memungkinkan MediaStore akses ke CloudWatch

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol IAM, pilih Kebijakan, lalu pilih Buat kebijakan.
3. Pilih tab JSON dan tempel kebijakan berikut:

```
{
    "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/*"
        }
    ]
}
```

   Kebijakan ini memungkinkan MediaStore untuk membuat grup log dan aliran log untuk setiap kontainer di Wilayah mana pun dalam AWS akun Anda.
4. Pilih Tinjau kebijakan.
5. Pada halaman Kebijakan tinjauan, untuk Nama, masukkan**`MediaStoreAccessLogsPolicy`**, lalu pilih Buat kebijakan.
6. Di panel navigasi konsol IAM, pilih Peran, dan lalu pilih Buat peran.
7. Pilih jenis peran Akun AWS lainnya.
8. Untuk ID Akun, masukkan ID AWS akun Anda.
9. Pilih Berikutnya: Izin.
10. Dalam kotak pencarian, masukkan **`MediaStoreAccessLogsPolicy`**.

11. Pilih kotak centang di samping kebijakan baru Anda, lalu pilih Berikutnya: Tag.
12. Pilih Berikutnya: Tinjau untuk melihat pratinjau pengguna baru Anda.
13. Untuk nama Peran**`MediaStoreAccessLogs`**, masukkan, lalu pilih Buat peran.
14. Dalam pesan konfirmasi, pilih nama peran yang baru saja Anda buat (**`MediaStoreAccessLogs`**).
15. Pada halaman Ringkasan peran, pilih tab Trust relationship.
16. Pilih Edit trust relationship (Edit Hubungan Kepercayaan).
17. Dalam dokumen kebijakan, ubah kepala sekolah ke MediaStore layanan. Seharusnya terlihat seperti ini:

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

    Seluruh kebijakan harus berbunyi sebagai berikut:

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

18. Pilih Perbarui Kebijakan Kepercayaan.

## Mengaktifkan pencatatan akses untuk kontainer

Secara default, AWS Elemental MediaStore tidak mengumpulkan log akses. Saat Anda mengaktifkan log akses pada kontainer, MediaStore mengirimkan log akses untuk objek yang disimpan dalam wadah tersebut ke Amazon CloudWatch. Log akses menyediakan catatan rinci untuk permintaan yang dibuat ke objek apa pun yang disimpan dalam wadah. Informasi ini dapat mencakup jenis

permintaan, sumber daya yang ditentukan dalam permintaan, dan waktu serta tanggal pemrosesan permintaan.

> **Important**
>
> Tidak ada biaya tambahan untuk mengaktifkan akses masuk pada MediaStore kontainer. Namun, file log apa pun yang diberikan layanan kepada Anda akan dikenakan biaya penyimpanan yang biasa. (Anda dapat menghapus file log kapan saja.) AWS tidak menilai biaya transfer data untuk pengiriman file log, tetapi membebankan kecepatan transfer data normal untuk mengakses file log.

Untuk mengaktifkan akses logging (AWS CLI)

- Di AWS CLI, gunakan `start-access-logging` perintah:

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

  Perintah ini tidak memiliki nilai kembali.

## Menonaktifkan pencatatan akses untuk kontainer

Saat Anda menonaktifkan log akses pada penampung, AWS Elemental MediaStore berhenti mengirim log akses ke Amazon. CloudWatch Log akses ini tidak disimpan dan tidak dapat diambil kembali.

Untuk menonaktifkan akses logging (AWS CLI)

- Di AWS CLI, gunakan `stop-access-logging` perintah:

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

  Perintah ini tidak memiliki nilai kembali.

## Memecahkan masalah pencatatan akses di AWS Elemental MediaStore

Jika log MediaStore akses AWS Elemental tidak muncul di Amazon CloudWatch, lihat tabel berikut untuk mengetahui kemungkinan penyebab dan resolusi.

> **Note**
>
> Pastikan untuk mengaktifkan AWS CloudTrail Log untuk membantu proses pemecahan masalah.

| Gejala | Masalahnya mungkin... | Coba ini... |
| --- | --- | --- |
| Anda tidak melihat CloudTrail peristiwa apa pun, meskipun CloudTrail log diaktifkan. | Peran IAM tidak ada atau memiliki nama, izin, atau kebijakan kepercayaan yang salah. | Buat peran dengan nama, izin, dan kebijakan kepercayaan yang benar. Lihat the section called “Menyiapkan izin untuk CloudWatch”. |
| Anda mengirimkan permintaan `DescribeContainer` API, tetapi respons menunjukkan bahwa `AccessLoggingEnabled` parameter memiliki nilai`False`. Selain itu, Anda tidak melihat CloudTrail peristiwa apa pun untuk `MediaStoreAccessLogs` peran yang membuat sukses`DescribeLogGroup` ,`CreateLogGroup` ,`DescribeLogStream` , atau `CreateLogStream` panggilan. | Peran IAM tidak ada atau memiliki nama, izin, atau kebijakan kepercayaan yang salah. | Buat peran dengan nama, izin, dan kebijakan kepercayaan yang benar. Lihat the section called “Menyiapkan izin untuk CloudWatch”. |
| | Pencatatan akses tidak diaktifkan pada penampung. | Aktifkan log akses untuk wadah. Lihat the section called “Mengaktifkan pencatatan akses”. |
| Di CloudTrail konsol, Anda melihat peristiwa dengan kesalahan akses ditolak yang terkait dengan `MediaStoreAccessLogs` peran tersebut. CloudTrail Acara ini mungkin mencakup garis-garis seperti berikut:<br><br>`"eventSource": "logs.amazonaws.com",` | Peran IAM tidak memiliki izin yang benar untuk AWS Elemental. MediaStore | Perbarui peran IAM agar memiliki izin dan kebijakan kepercayaan yang benar. Lihat the section called “Menyiapkan izin untuk CloudWatch”. |

| Gejala | Masalahnya mungkin... | Coba ini... |
| --- | --- | --- |
| `"errorCode": "AccessDenied",`<br><br>`"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:",` | | |
| Anda tidak melihat log apa pun untuk seluruh wadah atau wadah. | Akun Anda mungkin telah melebihi CloudWatch kuota untuk grup log per akun per Wilayah. Lihat kuota untuk grup log di Panduan Pengguna Amazon CloudWatch Logs. | Di CloudWatch konsol, tentukan apakah akun Anda telah memenuhi CloudWatch kuota untuk grup log. Jika perlu, minta kenaikan kuota. |

| Gejala | Masalahnya mungkin... | Coba ini... |
| --- | --- | --- |
| Anda melihat beberapa log masuk CloudWatch, tetapi tidak semua log yang Anda harapkan untuk dilihat. | Akun Anda mungkin telah melebihi CloudWatch kuota untuk transaksi per detik per akun per Wilayah. Lihat kuota untuk `PutLogEvents` di Panduan Pengguna Amazon CloudWatch Logs. | Minta kenaikan kuota untuk CloudWatch transaksi per detik per akun per Wilayah. |

## Akses format log

File log akses terdiri dari urutan catatan log berformat JSON, di mana setiap catatan log mewakili satu permintaan. Urutan bidang dalam log dapat bervariasi. Berikut ini adalah contoh log yang terdiri dari dua catatan log:

```
{
  "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
}
```


Daftar berikut menjelaskan bidang catatan log:

AWSAccountId

ID AWS akun akun yang digunakan untuk membuat permintaan.

BytesReceived

Jumlah byte dalam badan permintaan yang diterima MediaStore server.

BytesSent

Jumlah byte dalam badan respons yang dikirim MediaStore server. Nilai ini sering sama dengan nilai `Content-Length` header yang disertakan dengan respons server.

ContainerName

Nama wadah yang menerima permintaan.

ErrorCode

Kode MediaStore kesalahan (seperti`InternalServerError`). Jika tidak terjadi kesalahan, `-` karakter akan muncul. Kode kesalahan mungkin muncul bahkan jika kode statusnya 200 (menunjukkan koneksi tertutup atau kesalahan setelah server mulai mengalirkan respons).

ExpiresAt

Tanggal dan waktu kedaluwarsa objek. Nilai ini didasarkan pada usia kedaluwarsa yang ditetapkan oleh kebijakan transient data rule dalam siklus hidup yang diterapkan ke wadah. Nilainya adalah waktu `ISO-8601` tanggal dan didasarkan pada jam sistem host yang melayani permintaan. Jika kebijakan siklus hidup tidak memiliki aturan data sementara yang berlaku untuk objek, atau jika tidak ada kebijakan siklus hidup yang diterapkan ke wadah, nilai bidang ini adalah. `null` Bidang ini hanya berlaku untuk operasi berikut:`PutObject`,`GetObject`,`DescribeObject`, dan`DeleteObject`.

HTTPStatus

Kode status HTTP numerik dari respons.

Operasi

Operasi yang dilakukan, seperti `PutObject` atau`ListItems`.

Jalur

Jalur di dalam wadah tempat objek disimpan. Jika operasi tidak mengambil parameter jalur, `-` karakter muncul.

ReceivedTime

Waktu hari ketika permintaan diterima. Nilainya adalah waktu `ISO-8601` tanggal dan didasarkan pada jam sistem host yang melayani permintaan.

Peminta

Pengguna Amazon Resource Name (ARN) dari akun yang digunakan untuk membuat permintaan. Untuk permintaan yang tidak diautentikasi, nilai ini adalah. `anonymous` Jika permintaan gagal sebelum otentikasi selesai, bidang ini mungkin hilang dari log. Untuk permintaan tersebut, `ErrorCode` mungkin mengidentifikasi masalah otorisasi.

RequestID

String yang dihasilkan oleh AWS Elemental MediaStore untuk mengidentifikasi setiap permintaan secara unik.

Sumber

Alamat internet yang jelas dari pemohon atau kepala layanan layanan AWS yang melakukan panggilan. Jika proxy perantara dan firewall mengaburkan alamat mesin yang membuat permintaan, nilainya disetel ke null.

TotalTime

Jumlah milidetik (ms) bahwa permintaan berada dalam penerbangan dari perspektif server. Nilai ini diukur dimulai dengan waktu permintaan Anda diterima oleh layanan dan diakhiri dengan waktu byte terakhir dari respons dikirim. Nilai ini diukur dari perspektif server karena pengukuran yang dilakukan dari perspektif klien dipengaruhi oleh latensi jaringan.

TurnAroundTime

Jumlah milidetik yang MediaStore dihabiskan untuk memproses permintaan Anda. Nilai ini diukur dari waktu byte terakhir permintaan Anda diterima hingga saat byte pertama respons dikirim.

Urutan bidang di log dapat bervariasi.

## Perubahan status logging mulai berlaku dari waktu ke waktu

Perubahan status pencatatan kontainer membutuhkan waktu untuk benar-benar memengaruhi pengiriman file log. Misalnya, jika Anda mengaktifkan pencatatan untuk penampung A, beberapa permintaan yang dibuat pada jam berikutnya mungkin dicatat, sementara yang lain mungkin tidak. Jika Anda menonaktifkan pencatatan untuk penampung B, beberapa log untuk satu jam berikutnya mungkin terus dikirimkan, sementara yang lain mungkin tidak. Dalam semua kasus, pengaturan baru akhirnya berlaku tanpa tindakan lebih lanjut dari pihak Anda.

## Pengiriman log server dengan upaya terbaik

Catatan log akses dikirimkan atas dasar upaya terbaik. Sebagian besar permintaan untuk wadah yang dikonfigurasi dengan benar untuk pencatatan menghasilkan catatan log yang dikirimkan. Sebagian besar catatan log dikirim dalam beberapa jam setelah log dicatat, tetapi dapat dikirimkan lebih sering.

Kelengkapan dan ketepatan waktu akses logging tidak dijamin. Catatan log untuk permintaan tertentu mungkin dikirim dalam waktu lama setelah permintaan diproses, atau mungkin tidak dikirimkan sama sekali. Tujuan dari log akses adalah untuk memberi Anda gambaran tentang sifat lalu lintas terhadap wadah Anda. Sangat jarang kehilangan catatan log, tetapi pencatatan akses tidak dimaksudkan untuk menjadi akuntansi lengkap dari semua permintaan.

Ini mengikuti sifat upaya terbaik dari fitur pencatatan akses bahwa laporan penggunaan yang tersedia di portal AWS (laporan Billing and Cost Management on AWS Management Consolethe) mungkin menyertakan satu atau beberapa permintaan akses yang tidak muncul dalam log akses terkirim.

### Pertimbangan pemrograman untuk format log akses

Dari waktu ke waktu, kami mungkin memperluas format log akses dengan menambahkan bidang baru. Kode yang mem-parsing log akses harus ditulis untuk menangani bidang tambahan yang tidak dimengerti.

## CloudWatch Acara

Amazon CloudWatch Events memungkinkan Anda mengotomatiskan AWS layanan dan merespons secara otomatis peristiwa sistem seperti masalah ketersediaan aplikasi atau perubahan sumber daya. Anda dapat menulis aturan sederhana untuk menunjukkan kejadian mana yang sesuai kepentingan Anda, dan tindakan otomatis apa yang diambil ketika suatu kejadian sesuai dengan suatu aturan.

> **Important**
>
> Biasanya, AWS layanan mengirimkan pemberitahuan acara ke CloudWatch Acara dalam hitungan detik tetapi terkadang dapat memakan waktu satu menit atau lebih lama.

Ketika file diunggah ke wadah atau dihapus dari kontainer, dua peristiwa dijalankan secara berurutan dalam layanan: CloudWatch

1. the section called “Acara perubahan status objek”
2. the section called “Acara perubahan status kontainer”

Untuk informasi tentang berlangganan acara ini, lihat Amazon CloudWatch.

Tindakan yang dapat dipicu secara otomatis meliputi hal-hal berikut:

- Memanggil fungsi AWS Lambda
- Memanggil Perintah Amazon EC2 Run
- Mengirim peristiwa ke Amazon Kinesis Data Streams
- Mengaktifkan mesin AWS Step Functions negara
- Memberi tahu topik Amazon SNS atau antrian AWS SMS

Beberapa contoh penggunaan CloudWatch Events dengan AWS Elemental MediaStore meliputi:

- Mengaktifkan fungsi Lambda setiap kali wadah dibuat
- Memberi tahu topik Amazon SNS saat objek dihapus

Untuk informasi selengkapnya, lihat Panduan Pengguna CloudWatch Acara Amazon.

Topik

- Acara perubahan status MediaStore objek AWS Elemental
- Acara perubahan status MediaStore kontainer AWS Elemental

## Acara perubahan status MediaStore objek AWS Elemental

Acara ini diterbitkan ketika status objek telah berubah (ketika objek telah diunggah atau dihapus).

> **Note**
>
> Objek yang kedaluwarsa karena aturan data sementara tidak memancarkan CloudWatch peristiwa ketika mereka kedaluwarsa.

Untuk informasi tentang berlangganan acara ini, lihat Amazon CloudWatch.

Objek diperbarui

```
{
  "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"
  }
}
```

Objek dihapus

```
{
  "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"
  }
}
```

## Acara perubahan status MediaStore kontainer AWS Elemental

Peristiwa ini diterbitkan ketika status kontainer telah berubah (ketika wadah telah ditambahkan atau dihapus). Untuk informasi tentang berlangganan acara ini, lihat Amazon CloudWatch.

Kontainer dibuat

```
{
  "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"
  }
}
```

Wadah dihapus

```
{
  "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"
  }
}
```

## Memantau AWS Elemental dengan metrik MediaStore Amazon CloudWatch

Anda dapat memantau MediaStore penggunaan AWS Elemental CloudWatch, yang mengumpulkan data mentah dan memprosesnya menjadi metrik yang dapat dibaca. CloudWatchmenyimpan statistik disimpan selama 15 bulan sehingga Anda dapat mengakses informasi historis dan mendapatkan perspektif yang lebih baik tentang kinerja aplikasi atau layanan web Anda. Anda juga dapat mengatur alarm yang memperhatikan ambang batas tertentu dan mengirim notifikasi atau mengambil tindakan saat ambang batas tersebut terpenuhi. Untuk informasi selengkapnya, lihat Panduan CloudWatch Pengguna Amazon.

Untuk AWS Elemental MediaStore, Anda mungkin ingin menonton `BytesDownloaded` dan mengirim email ke diri Anda sendiri ketika metrik tersebut mencapai ambang batas tertentu.

Untuk melihat metrik menggunakan konsol CloudWatch

Metrik dikelompokkan terlebih dahulu berdasarkan namespace layanan, lalu berdasarkan berbagai kombinasi dimensi dalam setiap namespace.

1. Masuk ke AWS Management Console dan buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Pada panel navigasi, silakan pilih Metrik.
3. Di bawah Semua metrik, pilih MediaStoreAWS/namespace.
4. Pilih dimensi metrik untuk melihat metrik. Misalnya, pilih `Request metrics by container` untuk melihat metrik untuk berbagai jenis permintaan yang telah dikirim ke penampung.

Untuk melihat metrik menggunakan AWS CLI

- Pada prompt perintah, gunakan perintah berikut:

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

## Metrik AWS Elemental MediaStore

Tabel berikut mencantumkan metrik yang dikirimkan AWS MediaStore Elemental. CloudWatch

> **Note**
>
> Untuk melihat metrik, Anda harus menambahkan kebijakan metrik ke penampung agar dapat mengirim metrik MediaStore ke Amazon. CloudWatch

| Metrik | Deskripsi |
| --- | --- |
| `RequestCount` | Jumlah total permintaan HTTP yang dibuat ke MediaStore wadah, dipisahkan oleh jenis operasi (`Put`, `GetDelete`,`Describe`,`List`).<br><br>Unit: Hitungan<br><br>Dimensi yang valid:<br><br>• Nama kontainer |

| Metrik | Deskripsi |
| --- | --- |
| | • Nama grup objek<br>• Jenis permintaan<br><br>Statistik yang valid: Jumlah |
| `4xxErrorCount` | Jumlah permintaan HTTP yang dibuat untuk MediaStore itu menghasilkan kesalahan 4xx.<br><br>Unit: Hitungan<br><br>Dimensi yang valid:<br><br>• Nama kontainer<br>• Nama grup objek<br>• Jenis permintaan<br><br>Statistik yang valid: Jumlah |
| `5xxErrorCount` | Jumlah permintaan HTTP yang dibuat untuk MediaStore itu menghasilkan kesalahan 5xx.<br><br>Unit: Hitungan<br><br>Dimensi yang valid:<br><br>• Nama kontainer<br>• Nama grup objek<br>• Jenis permintaan<br><br>Statistik yang valid: Jumlah |

| Metrik | Deskripsi |
| --- | --- |
| `BytesUploaded` | Jumlah byte yang diunggah untuk permintaan yang dibuat ke MediaStore wadah, di mana permintaan menyertakan badan.<br><br>Unit: Byte<br><br>Dimensi yang valid:<br><br>• Nama kontainer<br>• Nama grup objek<br><br>Statistik yang valid: Rata-rata (byte per permintaan), Jumlah (byte per periode), Jumlah Sampel, Min (sama dengan P0.0), Maks (sama dengan p100), setiap persentil antara p0.0 dan p99.9 |
| `BytesDownloaded` | Jumlah byte yang diunduh untuk permintaan yang dibuat ke MediaStore wadah, di mana respons menyertakan badan.<br><br>Unit: Byte<br><br>Dimensi yang valid:<br><br>• Nama kontainer<br>• Nama grup objek<br><br>Statistik yang valid: Rata-rata (byte per permintaan), Jumlah (byte per periode), Jumlah Sampel, Min (sama dengan P0.0), Maks (sama dengan p100), setiap persentil antara p0.0 dan p99.9 |

| Metrik | Deskripsi |
| --- | --- |
| `TotalTime` | Jumlah milidetik permintaan dalam proses pengiriman dari perspektif server. Nilai ini diukur dari waktu yang MediaStore menerima permintaan Anda, hingga saat ia mengirimkan byte terakhir dari respons. Nilai ini diukur dari perspektif server karena pengukuran yang dilakukan dari perspektif klien dipengaruhi oleh latensi jaringan.<br><br>Unit: Milidetik<br><br>Dimensi yang valid:<br><br>• Nama kontainer<br>• Nama grup objek<br>• Jenis permintaan<br><br>Statistik yang valid: Rata-rata, Min (sama dengan P0.0), Max (sama dengan p100), setiap persentil antara p0.0 dan p100 |
| `TurnaroundTime` | Jumlah milidetik yang MediaStore dihabiskan untuk memproses permintaan Anda. Nilai ini diukur dari waktu yang MediaStore menerima byte terakhir dari permintaan Anda, hingga saat ia mengirimkan byte pertama dari respons.<br><br>Unit: Milidetik<br><br>Dimensi yang valid:<br><br>• Nama kontainer<br>• Nama grup objek<br>• Jenis permintaan<br><br>Statistik yang valid: Rata-rata, Min (sama dengan P0.0), Max (sama dengan p100), setiap persentil antara p0.0 dan p100 |

| Metrik | Deskripsi |
| --- | --- |
| `ThrottleCount` | Jumlah permintaan HTTP yang dibuat untuk MediaStore itu dibatasi.<br><br>Unit: Hitungan<br><br>Dimensi yang valid:<br><br>• Nama kontainer<br>• Nama grup objek<br>• Jenis permintaan<br><br>Statistik yang valid: Jumlah |

# Menandai sumber daya MediaStore AWS Elemental

Tag adalah label atribut kustom yang Anda tetapkan atau yang ditetapkan ke AWS sumber daya. AWS Setiap tag memiliki dua bagian:

- Sebuah kunci tag (misalnya, `CostCenter`, `Environment`, atau `Project`). Kunci tag peka huruf besar dan kecil.
- Bidang opsional yang dikenal sebagai nilai tag (misalnya, `111122223333` atau `Production`). Mengabaikan nilai tag sama dengan menggunakan sebuah string kosong. Seperti kunci tag, nilai tag peka huruf besar dan kecil.

Tag membantu Anda melakukan hal berikut:

- Identifikasi dan atur AWS sumber daya Anda. Banyak tag memberikan support pada layanan AWS, sehingga Anda dapat menetapkan tag yang sama ke sumber daya dari berbagai layanan untuk menunjukkan bahwa sumber daya tersebut terkait. Misalnya, Anda dapat menetapkan tag yang sama ke AWS MediaStore *`container`* Elemental yang Anda tetapkan ke input. AWS Elemental MediaLive
- Telusuri biaya AWS Anda. Anda mengaktifkan tag ini di AWS Manajemen Penagihan dan Biaya dasbor. AWS menggunakan tag untuk mengategorikan biaya Anda dan mengirimkan laporan alokasi biaya bulanan kepada Anda. Untuk informasi selengkapnya, lihat Menggunakan Tag Alokasi Biaya dalam Panduan Pengguna AWS Billing.

Bagian berikut memberikan informasi selengkapnya tentang tag untuk AWS Elemental MediaStore.

## Sumber daya yang didukung di AWS Elemental MediaStore

Sumber daya berikut dalam AWS Elemental MediaStore mendukung penandaan:

- *container*

Untuk informasi tentang menambahkan dan mengelola tag, silakan lihat Mengelola tag.

AWS Elemental MediaStore tidak mendukung fitur kontrol akses berbasis tag AWS Identity and Access Management (IAM).

## Kesepakatan penamaan dan penggunaan tag

Konvensi penamaan dan penggunaan dasar berikut berlaku untuk menggunakan tag dengan sumber daya AWS MediaStore Elemental:

- Setiap sumber daya dapat memiliki maksimum 50 tag.
- Untuk setiap sumber daya, setiap kunci tag harus unik, dan setiap kunci tag hanya dapat memiliki satu nilai.
- Panjang kunci tag maksimum adalah 128 karakter Unicode dalam UTF-8.
- Panjang nilai tag maksimum adalah 256 karakter Unicode dalam UTF-8.
- Karakter yang diperbolehkan adalah huruf, angka, spasi yang dapat ditampilkan di UTF-8, serta karakter berikut: . : + = @ _ / - (tanda hubung). EC2 Sumber daya Amazon memungkinkan karakter apa pun.
- Kunci dan nilai tanda peka huruf besar-kecil. Sebagai praktik terbaik, putuskan strategi untuk memanfaatkan tag dan terapkan strategi tersebut secara konsisten di semua jenis sumber daya. Misalnya, putuskan apakah akan menggunakan `Costcenter`, `costcenter`, atau `CostCenter` dan menggunakan kesepakatan yang sama untuk semua tag. Hindari penggunaan tag yang serupa dengan perlakuan kasus yang tidak konsisten.
- `aws:`Awalan dilarang untuk tag; itu dicadangkan untuk AWS digunakan. Anda tidak dapat menyunting atau menghapus kunci atau nilai tag dengan awalan ini. Tanda dengan prefiks ini tidak memengaruhi tanda Anda per kuota sumber daya.

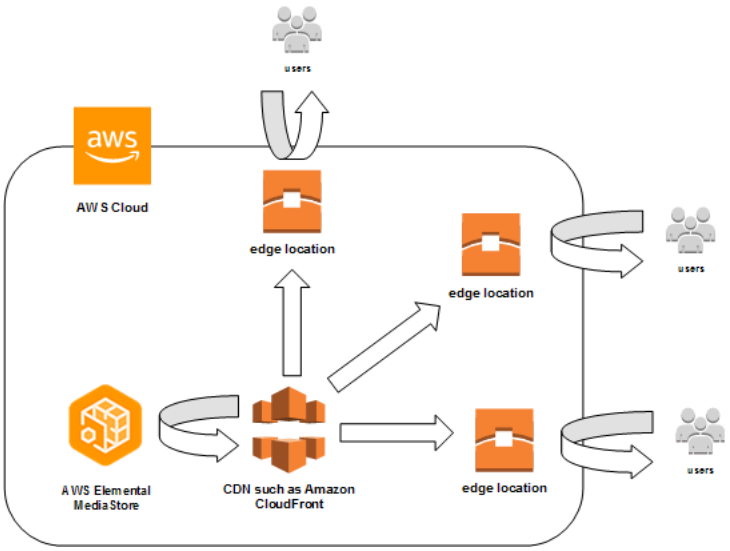
## Mengelola tag

Tag terdiri dari `Key` dan `Value` properti pada sumber daya. Anda dapat menggunakan AWS CLI atau MediaStore API untuk menambahkan, mengedit, atau menghapus nilai untuk properti ini. Untuk informasi tentang bekerja dengan tag, lihat bagian berikut di Referensi AWS Elemental MediaStore API:

- CreateContainer
- ListTagsForResource
- Sumber Daya
- TagResource
- UntagResource

# Bekerja dengan jaringan pengiriman konten (CDNs)

Anda dapat menggunakan jaringan pengiriman konten (CDN) seperti Amazon CloudFront untuk menyajikan konten yang Anda simpan di MediaStore AWS Elemental. CDN adalah kumpulan server yang didistribusikan secara global yang menyimpan konten seperti video. Saat pengguna meminta konten Anda, CDN merutekan permintaan ke lokasi tepi yang memberikan latensi terendah. Jika konten Anda sudah di-cache di lokasi tepi itu, CDN segera mengirimkannya. Jika konten Anda saat ini tidak berada di lokasi tepi tersebut, CDN mengambilnya dari asal Anda (seperti MediaStore penampung Anda) dan mendistribusikannya ke pengguna.




Topik

- Mengizinkan Amazon CloudFront mengakses wadah AWS Elemental MediaStore Anda
- Interaksi AWS Elemental MediaStore dengan cache HTTP

# Mengizinkan Amazon CloudFront mengakses wadah AWS Elemental MediaStore Anda

Anda dapat menggunakan Amazon CloudFront untuk menyajikan konten yang Anda simpan dalam wadah di AWS Elemental MediaStore. Anda dapat melakukannya dengan salah satu cara berikut:

- Menggunakan Origin Access Control (OAC)- (Disarankan) Gunakan opsi ini jika Anda Wilayah AWS mendukung fitur OAC. CloudFront

- Menggunakan Rahasia Bersama- Gunakan opsi ini jika Anda Wilayah AWS tidak mendukung fitur OAC. CloudFront

## Menggunakan Origin Access Control (OAC)

Anda dapat menggunakan fitur Origin Access Control (OAC) Amazon CloudFront untuk mengamankan asal AWS MediaStore Elemental dengan keamanan yang ditingkatkan. Anda dapat mengaktifkan AWS Signature Version 4 (SigV4) pada CloudFront permintaan untuk MediaStore asal dan mengatur kapan dan jika CloudFront harus menandatangani permintaan. Anda dapat mengakses fitur OAC CloudFront melalui konsol, SDK APIs, atau CLI, dan tidak ada biaya tambahan untuk penggunaannya.

Untuk informasi selengkapnya tentang menggunakan fitur OAC dengan MediaStore, lihat Membatasi akses ke MediaStore asal dalam Panduan CloudFront Pengembang Amazon.

## Menggunakan Rahasia Bersama

Jika Anda Wilayah AWS tidak mendukung fitur OAC Amazon CloudFront, Anda dapat melampirkan kebijakan ke kontainer AWS MediaStore Elemental yang memberikan akses baca atau yang lebih besar. CloudFront

> **Note**
>
> Sebaiknya gunakan fitur OAC jika Anda Wilayah AWS mendukungnya. Prosedur berikut mengharuskan Anda untuk mengkonfigurasi MediaStore dan CloudFront dengan rahasia bersama untuk membatasi akses ke MediaStore kontainer. Untuk mengikuti praktik keamanan terbaik, konfigurasi manual ini memerlukan rotasi rahasia secara berkala. Dengan OAC on MediaStore origin, Anda dapat menginstruksikan CloudFront untuk menandatangani permintaan menggunakan SigV4 dan meneruskannya ke MediaStore pencocokan tanda tangan, menghilangkan kebutuhan untuk menggunakan dan memutar rahasia. Ini memastikan bahwa permintaan diverifikasi secara otomatis sebelum konten media disajikan, membuat pengiriman konten media melalui MediaStore dan CloudFront lebih sederhana dan lebih aman.

Untuk memungkinkan CloudFront mengakses penampung Anda (konsol)

1. Buka MediaStore konsol di https://console.aws.amazon.com/mediastore/.

2. Pada halaman Kontainer, pilih nama kontainer.

   Halaman detail kontainer muncul.

3. Di bagian Kebijakan penampung, lampirkan kebijakan yang memberikan akses baca atau yang lebih besar ke Amazon CloudFront.

   Example

   Contoh kebijakan berikut, yang mirip dengan kebijakan contoh untuk Akses Baca Publik melalui HTTPS, cocok dengan persyaratan ini karena memungkinkan `GetObject` dan `DescribeObject` perintah dari siapa saja yang mengirimkan permintaan ke domain Anda melalui HTTPS. Selanjutnya, contoh kebijakan berikut lebih baik mengamankan alur kerja Anda karena memungkinkan CloudFront akses ke MediaStore objek hanya ketika permintaan terjadi melalui koneksi HTTPS dan berisi header Referer yang benar.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CloudFrontRead",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "mediastore:GetObject",
        "mediastore:DescribeObject"
      ],
      "Resource": "arn:aws:mediastore:<region>:<owner acct
 number>:container/<container name>/*",
      "Condition": {
        "StringEquals": {
          "aws:Referer": "<secretValue>"
        },
        "Bool": {
          "aws:SecureTransport": "true"
        }
      }
    }
 ]}
```

4. Di bagian kebijakan Container CORS, tetapkan kebijakan yang memungkinkan tingkat akses yang sesuai.

> **Note**
>
> Kebijakan CORS diperlukan hanya jika Anda ingin memberikan akses ke pemutar berbasis browser.

5. Catat detail berikut:

   - Titik akhir data yang ditetapkan untuk kontainer Anda. Anda dapat menemukan informasi ini di bagian Info pada halaman Kontainer. Pada tahun CloudFront, titik akhir data disebut sebagai nama domain asal.
   - Struktur folder dalam wadah tempat objek disimpan. Dalam CloudFront, ini disebut sebagai jalur asal. Perhatikan bahwa pengaturan ini opsional. Untuk informasi selengkapnya tentang jalur asal, lihat Panduan CloudFront Pengembang Amazon.

6. Di CloudFront, buat distribusi yang dikonfigurasi untuk menyajikan konten dari AWS Elemental MediaStore. Anda akan membutuhkan informasi yang Anda kumpulkan pada langkah sebelumnya.

Setelah melampirkan kebijakan ke MediaStore container, Anda harus mengonfigurasi CloudFront agar hanya menggunakan koneksi HTTPS untuk permintaan asal, dan juga menambahkan header kustom dengan nilai rahasia yang benar.

Untuk mengonfigurasi CloudFront untuk mengakses penampung Anda melalui koneksi HTTPS dengan nilai rahasia untuk header Referer (konsol)

1. Buka CloudFront konsol.
2. Pada halaman Origins, pilih MediaStore asal Anda.
3. Pilih Edit.
4. Pilih HTTPS hanya untuk protokol.
5. Di bagian Tambahkan header khusus, pilih Tambahkan header.
6. Untuk Nama, pilih Referer. Untuk nilainya, gunakan `<secretValue>` string yang sama dengan yang Anda gunakan dalam kebijakan kontainer Anda.
7. Pilih Simpan dan biarkan perubahan diterapkan.

# Interaksi AWS Elemental MediaStore dengan cache HTTP

AWS Elemental MediaStore menyimpan objek sehingga dapat di-cache dengan benar dan efisien oleh jaringan pengiriman konten () CDNs seperti Amazon. CloudFront Ketika pengguna akhir atau CDN mengambil objek dari MediaStore, layanan mengembalikan header HTTP yang mempengaruhi perilaku caching objek. (Standar untuk perilaku caching HTTP 1.1 ditemukan di RFC2616 bagian 13.) Header ini adalah:

- **`ETag`**(tidak dapat disesuaikan) - Header tag entitas adalah pengidentifikasi unik untuk respons yang mengirim. MediaStore Sesuai standar CDNs dan browser web menggunakan tag ini sebagai kunci untuk menyimpan objek dengan cache. MediaStore secara otomatis menghasilkan `ETag` untuk setiap objek ketika diunggah. Anda dapat melihat detail objek untuk menentukan ETag nilainya.
- **`Last-Modified`**(tidak dapat disesuaikan) - Nilai header ini menunjukkan tanggal dan waktu objek diubah. MediaStore secara otomatis menghasilkan nilai ini ketika objek diunggah.
- **`Cache-Control`**(dapat disesuaikan) - Nilai header ini mengontrol berapa lama suatu objek harus di-cache sebelum CDN memeriksa untuk melihat apakah telah dimodifikasi. Anda dapat mengatur header ini ke nilai apa pun saat Anda mengunggah objek ke MediaStore wadah menggunakan CLI atau API. Set lengkap nilai valid dijelaskan dalam dokumentasi HTTP/1.1. Jika Anda tidak menetapkan nilai ini ketika Anda meng-upload objek, MediaStore tidak akan mengembalikan header ini ketika objek diambil.

  Kasus penggunaan umum untuk header Cache-Control adalah untuk menentukan durasi untuk cache objek. Misalnya, Anda memiliki file manifes video yang sering ditimpa oleh encoder. Anda dapat mengatur `max-age` ke 10 untuk menunjukkan bahwa objek harus di-cache hanya selama 10 detik. Atau misalkan Anda memiliki segmen video tersimpan yang tidak akan pernah ditimpa. Anda dapat mengatur `max-age` untuk objek ini ke 31536000 ke cache selama kurang lebih 1 tahun.

## Permintaan bersyarat

### Permintaan bersyarat untuk MediaStore

MediaStore merespons secara identik dengan permintaan bersyarat (menggunakan header permintaan seperti `If-Modified-Since` dan`If-None-Match`, seperti yang dijelaskan dalam RFC7232) dan permintaan tanpa syarat. Ini berarti bahwa ketika MediaStore menerima `GetObject` permintaan yang valid, layanan selalu mengembalikan objek bahkan jika klien sudah memiliki objek.

## Permintaan bersyarat untuk CDNs

CDNs yang melayani konten atas nama MediaStore dapat memproses permintaan bersyarat dengan mengembalikan`304 Not Modified`, seperti yang dijelaskan dalam RFC7232 bagian 4.1. Ini menunjukkan bahwa tidak perlu mentransfer isi objek lengkap, karena pemohon sudah memiliki objek yang cocok dengan permintaan bersyarat.

CDNs (dan cache lain yang sesuai dengan HTTP/1.1) mendasarkan keputusan ini pada `ETag` dan `Cache-Control` header yang diteruskan oleh server asal. Untuk mengontrol seberapa sering server MediaStore asal CDNs kueri untuk pembaruan ke objek yang diambil berulang kali, setel `Cache-Control` header untuk objek tersebut saat Anda mengunggahnya. MediaStore

# Menggunakan layanan ini dengan AWS SDK

AWS kit pengembangan perangkat lunak (SDKs) tersedia untuk banyak bahasa pemrograman populer. Setiap SDK menyediakan API, contoh kode, dan dokumentasi yang memudahkan developer untuk membangun aplikasi dalam bahasa pilihan mereka.

| Dokumentasi SDK | Contoh kode |
| --- | --- |
| AWS SDK untuk C++ | AWS SDK untuk C++ contoh kode |
| AWS CLI | AWS CLI contoh kode |
| AWS SDK untuk Go | AWS SDK untuk Go contoh kode |
| AWS SDK untuk Java | AWS SDK untuk Java contoh kode |
| AWS SDK untuk JavaScript | AWS SDK untuk JavaScript contoh kode |
| AWS SDK untuk Kotlin | AWS SDK untuk Kotlin contoh kode |
| AWS SDK untuk .NET | AWS SDK untuk .NET contoh kode |
| AWS SDK untuk PHP | AWS SDK untuk PHP contoh kode |
| Alat AWS untuk PowerShell | Alat untuk contoh PowerShell kode |
| AWS SDK untuk Python (Boto3) | AWS SDK untuk Python (Boto3) contoh kode |
| AWS SDK untuk Ruby | AWS SDK untuk Ruby contoh kode |
| AWS SDK for Rust | AWS SDK for Rust contoh kode |
| AWS SDK untuk SAP ABAP | AWS SDK untuk SAP ABAP contoh kode |
| AWS SDK for Swift | AWS SDK for Swift contoh kode |

Untuk contoh khusus untuk layanan ini, lihat Contoh kode untuk MediaStore menggunakan AWS SDKs.

Ketersediaan contoh

Tidak dapat menemukan apa yang Anda butuhkan? Minta contoh kode menggunakan tautan Berikan umpan balik di bagian bawah halaman ini.

# Contoh kode untuk MediaStore menggunakan AWS SDKs

Contoh kode berikut menunjukkan cara menggunakan MediaStore kit pengembangan AWS perangkat lunak (SDK).

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Contoh kode

- Contoh dasar untuk MediaStore menggunakan AWS SDKs
  - Tindakan untuk MediaStore menggunakan AWS SDKs
    - Gunakan CreateContainer dengan AWS SDK atau CLI
    - Gunakan DeleteContainer dengan AWS SDK atau CLI
    - Gunakan DeleteObject dengan AWS SDK
    - Gunakan DescribeContainer dengan AWS SDK atau CLI
    - Gunakan GetObject dengan AWS SDK atau CLI
    - Gunakan ListContainers dengan AWS SDK atau CLI
    - Gunakan PutObject dengan AWS SDK atau CLI

# Contoh dasar untuk MediaStore menggunakan AWS SDKs

Contoh kode berikut menunjukkan cara menggunakan dasar-dasar AWS Elemental MediaStore dengan AWS SDKs.

Contoh

- Tindakan untuk MediaStore menggunakan AWS SDKs
  - Gunakan CreateContainer dengan AWS SDK atau CLI
  - Gunakan DeleteContainer dengan AWS SDK atau CLI
  - Gunakan DeleteObject dengan AWS SDK

- Gunakan DescribeContainer dengan AWS SDK atau CLI
- Gunakan GetObject dengan AWS SDK atau CLI
- Gunakan ListContainers dengan AWS SDK atau CLI
- Gunakan PutObject dengan AWS SDK atau CLI

## Tindakan untuk MediaStore menggunakan AWS SDKs

Contoh kode berikut menunjukkan bagaimana melakukan MediaStore tindakan individu dengan AWS SDKs. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkapnya, lihat Referensi AWS Elemental MediaStore API.

Contoh

- Gunakan CreateContainer dengan AWS SDK atau CLI
- Gunakan DeleteContainer dengan AWS SDK atau CLI
- Gunakan DeleteObject dengan AWS SDK
- Gunakan DescribeContainer dengan AWS SDK atau CLI
- Gunakan GetObject dengan AWS SDK atau CLI
- Gunakan ListContainers dengan AWS SDK atau CLI
- Gunakan PutObject dengan AWS SDK atau CLI

### Gunakan **`CreateContainer`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateContainer`.

CLI

AWS CLI

Untuk membuat wadah

`create-container`Contoh berikut membuat wadah baru yang kosong.

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

Output:

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

Untuk informasi selengkapnya, lihat Membuat Kontainer di Panduan MediaStore Pengguna AWS Elemental.

- Untuk detail API, lihat CreateContainerdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
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);
        }
    }
}
```

- Untuk detail API, lihat CreateContainerdi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **`DeleteContainer`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteContainer`.

CLI

AWS CLI

Untuk menghapus wadah

`delete-container`Contoh berikut menghapus wadah yang ditentukan. Anda dapat menghapus wadah hanya jika tidak memiliki objek.

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

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menghapus Kontainer di  MediaStore Panduan Pengguna AWS Elemental.

- Untuk detail API, lihat DeleteContainerdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
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);
        }
    }
}
```

- Untuk detail API, lihat DeleteContainerdi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **`DeleteObject`** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan`DeleteObject`.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
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();
    }
}
```

- Untuk detail API, lihat DeleteObjectdi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **`DescribeContainer`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DescribeContainer`.

CLI

AWS CLI

Untuk melihat detail kontainer

`describe-container`Contoh berikut menampilkan rincian wadah yang ditentukan.

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

Output:

```
{
    "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"
    }
}
```

Untuk informasi selengkapnya, lihat Melihat Detail untuk Kontainer di Panduan MediaStore Pengguna AWS Elemental.

- Untuk detail API, lihat DescribeContainerdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
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 "";
    }
}
```

- Untuk detail API, lihat DescribeContainerdi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **`GetObject`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetObject`.

CLI

AWS CLI

Untuk mengunduh objek

`get-object`Contoh berikut men-download objek ke endpoint yang ditentukan.

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

Output:

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

Untuk mengunduh bagian dari suatu objek

`get-object`Contoh berikut mendownload sebagian objek ke titik akhir yang ditentukan.

```
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
```

Output:

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

Untuk informasi selengkapnya, lihat Mengunduh Objek di Panduan MediaStore Pengguna AWS Elemental.

- Untuk detail API, lihat GetObjectdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
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();
    }
}
```

- Untuk detail API, lihat GetObjectdi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **`ListContainers`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListContainers`.

CLI

AWS CLI

Untuk melihat daftar kontainer

`list-containers`Contoh berikut menampilkan daftar semua kontainer yang terkait dengan akun Anda.

```
aws mediastore list-containers
```

Output:

```
{
    "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://fffgggghhhiiijj.data.mediastore.us-
west-2.amazonaws.com",
            "Status": "ACTIVE",
            "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/
ExampleContainer",
            "AccessLoggingEnabled": false,
            "Name": "ExampleContainer"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Melihat Daftar Kontainer di Panduan MediaStore Pengguna AWS Elemental.

- Untuk detail API, lihat ListContainersdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
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);
        }
    }
}
```

- Untuk detail API, lihat ListContainersdi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **`PutObject`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutObject`.

CLI

AWS CLI

Untuk mengunggah objek

`put-object`Contoh berikut mengunggah objek ke wadah yang ditentukan. Anda dapat menentukan jalur folder tempat objek akan disimpan di dalam wadah. Jika folder sudah ada, AWS Elemental MediaStore menyimpan objek di folder. Jika folder tidak ada, layanan membuatnya, dan kemudian menyimpan objek di folder.

```
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
```

Output:

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

Untuk informasi selengkapnya, lihat Mengunggah Objek di  MediaStore Panduan Pengguna AWS Elemental.

- Untuk detail API, lihat PutObjectdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
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();
    }
}
```

- Untuk detail API, lihat PutObjectdi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

# Kuota dalam AWS Elemental MediaStore

Konsol Service Quotas menyediakan informasi tentang kuota AWS MediaStore Elemental. Sembari melihat kuota default, Anda dapat menggunakan konsol Service Quotas guna mengajukan penambahan kuota untuk kuota yang dapat disesuaikan.

Tabel berikut menjelaskan kuota, sebelumnya disebut sebagai batas, di AWS Elemental. MediaStore Kuota adalah jumlah maksimum sumber daya layanan atau operasi untuk akun AWS Anda.

> **Note**
>
> Untuk menetapkan kuota ke setiap kontainer dalam akun Anda, hubungi AWS Support atau manajer akun Anda. Opsi ini dapat membantu Anda membagi batas tingkat akun di antara kontainer Anda, untuk mencegah satu kontainer menggunakan seluruh kuota Anda.

| Sumber Daya atau Operasi | Kuota Default | Komentar |
|---|---|---|
| Kontainer | 100 | Jumlah maksimum kontainer yang dapat Anda buat di akun ini. |
| Tingkat Folder | 10 | Jumlah maksimum level folder yang dapat Anda buat dalam wadah. Anda dapat membuat folder sebanyak yang Anda inginkan, asalkan tidak bersarang lebih dari 10 level dalam wadah. |
| Folder | Tidak terbatas. | Anda dapat membuat folder sebanyak yang Anda inginkan, asalkan tidak bersarang lebih dari 10 level dalam wadah. |
| Ukuran Objek | 25 MB | Ukuran file maksimum dari satu objek. |
| Objek | Tidak terbatas. | Anda dapat mengunggah objek sebanyak yang Anda inginkan ke folder atau wadah di akun Anda. |

| Sumber Daya atau Operasi | Kuota Default | Komentar |
| --- | --- | --- |
| Tingkat Permintaan API DeleteObject | 100 | Jumlah maksimum permintaan operasi yang dapat Anda buat per detik. Permintaan tambahan diperlambat.<br><br>Anda dapat meminta penambahan kuota. |
| Tingkat Permintaan API DescribeObject | 1.000 | Jumlah maksimum permintaan operasi yang dapat Anda buat per detik. Permintaan tambahan diperlambat.<br><br>Anda dapat meminta penambahan kuota. |
| Tingkat permintaan GetObjectAPI untuk ketersediaan unggahan standar | 1.000 | Jumlah maksimum permintaan operasi yang dapat Anda buat per detik. Permintaan tambahan diperlambat.<br><br>Anda dapat meminta penambahan kuota. |
| Tingkat permintaan GetObjectAPI untuk ketersediaan upload streaming | 25 | Jumlah maksimum permintaan operasi yang dapat Anda buat per detik. Permintaan tambahan diperlambat.<br><br>Anda dapat meminta penambahan kuota. |
| Tingkat Permintaan API ListItems | 5 | Jumlah maksimum permintaan operasi yang dapat Anda buat per detik. Permintaan tambahan diperlambat.<br><br>Anda dapat meminta penambahan kuota. |

| Sumber Daya atau Operasi | Kuota Default | Komentar |
| --- | --- | --- |
| Tingkat permintaan PutObjectAPI untuk encoding transfer chunked (juga dikenal sebagai ketersediaan upload streaming) | 10 | Jumlah maksimum permintaan operasi yang dapat Anda buat per detik. Permintaan tambahan diperlambat.<br><br>Anda dapat meminta penambahan kuota. Dalam permintaan, tentukan TPS yang diminta dan ukuran objek rata-rata. |
| Tingkat permintaan PutObjectAPI untuk ketersediaan unggahan standar | 100 | Jumlah maksimum permintaan operasi yang dapat Anda buat per detik. Permintaan tambahan diperlambat.<br><br>Anda dapat meminta penambahan kuota. Dalam permintaan, tentukan TPS yang diminta dan ukuran objek rata-rata. |
| Aturan dalam Kebijakan Metrik | 10 | Jumlah maksimum aturan yang dapat Anda sertakan dalam kebijakan metrik. |
| Aturan dalam Kebijakan Siklus Hidup Objek | 10 | Jumlah maksimum aturan yang dapat Anda sertakan dalam kebijakan siklus hidup objek. |

# Informasi terkait AWS Elemental MediaStore

Tabel berikut mencantumkan sumber daya terkait yang menurut Anda berguna saat Anda bekerja dengan AWS Elemental MediaStore.

- Kelas & Lokakarya - Tautan ke kursus berbasis peran dan khusus, selain laboratorium mandiri untuk membantu mempertajam keterampilan Anda AWS dan mendapatkan pengalaman praktis.
- AWS Pusat Pengembang — Jelajahi tutorial, unduh alat, dan pelajari tentang acara AWS pengembang.
- AWS Alat Pengembang — Tautan ke alat pengembang SDKs, toolkit IDE, dan alat baris perintah untuk mengembangkan dan mengelola AWS aplikasi.
- Memulai Pusat Sumber Daya — Pelajari cara menyiapkan Akun AWS, bergabung dengan AWS komunitas, dan meluncurkan aplikasi pertama Anda.
- Hands-On Tutorial - Ikuti step-by-step tutorial untuk meluncurkan aplikasi pertama Anda. AWS
- AWS Whitepaper — Tautan ke daftar lengkap AWS whitepaper teknis, yang mencakup topik-topik seperti arsitektur, keamanan, dan ekonomi dan ditulis oleh AWS Solutions Architects atau pakar teknis lainnya.
- AWS Dukungan Pusat — Hub untuk membuat dan mengelola AWS Dukungan kasus Anda. Juga termasuk tautan ke sumber daya bermanfaat lainnya, seperti forum, teknis FAQs, status kesehatan layanan, dan AWS Trusted Advisor.
- Dukungan— Halaman web utama untuk informasi tentang Dukungan, saluran dukungan respons cepat untuk membantu Anda membangun dan menjalankan aplikasi di cloud. one-on-one
- Hubungi Kami – Titik kontak pusat untuk pertanyaan tentang tandaihan AWS , akun, peristiwa, penyalahgunaan, dan masalah lainnya.
- AWS Ketentuan Situs — Informasi terperinci tentang hak cipta dan merek dagang kami; akun, lisensi, dan akses situs Anda; dan topik lainnya.

# Riwayat dokumen untuk panduan pengguna

Tabel berikut menjelaskan dokumentasi untuk rilis AWS Elemental MediaStore ini. Untuk notifikasi tentang pembaruan dokumentasi ini, Anda dapat berlangganan ke umpan RSS.

| Perubahan | Deskripsi | Tanggal |
| --- | --- | --- |
| Pemberitahuan akhir dukungan | Pemberitahuan akhir dukungan: Pada 13 November 2025, AWS akan menghenti kan dukungan untuk AWS Elemental. MediaStor e Setelah 13 November 2025, Anda tidak lagi dapat mengakses MediaStore konsol atau MediaStore sumber daya. Untuk informasi lebih lanjut, kunjungi posting blog ini. | November 12, 2024 |
| Peningkatan Kontrol Akses Asal (OAC) | Menambahkan informasi tentang cara menggunakan OAC dengan MediaStore AWS Elemental. | 17 April 2023 |
| Pembaruan kuota | Nilai kuota yang dikoreksi dan deskripsi untuk. `Rules in a Metric Policy` | 25 Oktober 2022 |
| ExpiresAt lapangan | Log akses sekarang menyertakan `ExpiresAt` bidang yang menunjukk an tanggal dan waktu kedaluwarsa objek berdasark an aturan data sementara dalam kebijakan siklus hidup container. | Juli 16, 2020 |

| | | |
|---|---|---|
| Aturan transisi siklus hidup | Sekarang Anda dapat menambahkan aturan transisi siklus hidup ke kebijakan siklus hidup objek yang menetapkan objek untuk dipindahkan ke kelas penyimpanan akses jarang (IA) setelah mencapai usia tertentu. | 20 April 2020 |
| Wadah kosong | Anda sekarang dapat menghapus semua objek dalam wadah sekaligus. | 7 April 2020 |
| Dukungan untuk CloudWatch metrik Amazon | Anda dapat menetapkan kebijakan metrik untuk menentukan metrik mana yang MediaStore dikirimkan. CloudWatch | 30 Maret 2020 |
| Wildcard dalam menghapus aturan objek | Dalam kebijakan siklus hidup objek, Anda sekarang dapat menggunakan wildcard dalam aturan objek hapus. Ini memungkinkan Anda untuk menentukan file berdasarkan nama file atau ekstensi mereka yang Anda ingin layanan untuk menghapus setelah beberapa hari. | 20 Desember 2019 |
| Kebijakan siklus hidup objek | Sekarang Anda dapat menambahkan aturan ke kebijakan siklus hidup objek yang menunjukkan kedaluwarsa berdasarkan usia dalam hitungan detik. | September 13, 2019 |

| | | |
|---|---|---|
| AWS CloudFormation dukungan | Anda sekarang dapat menggunakan AWS CloudFormation template untuk membuat wadah secara otomatis. AWS CloudFormation Template mengelola data untuk lima tindakan API: membuat container, menyetel akses logging, memperbarui kebijakan container default, menambahkan kebijakan cross-origin resource sharing (CORS), dan menambahkan kebijakan siklus hidup objek. | 17 Mei 2019 |
| Kuota untuk ketersediaan upload streaming | Untuk objek dengan ketersediaan unggahan streaming (transfer objek terpotongan), `PutObject` operasi tidak dapat melebihi 10 TPS dan `GetObject` operasi tidak dapat melebihi 25 TPS. | 8 April 2019 |
| Transfer objek yang terbongkar | Ditambahkan dukungan untuk transfer chunked objek. Kemampuan ini memungkinkan Anda untuk menentukan bahwa objek tersedia untuk diunduh sebelum objek diunggah sepenuhnya. | 5 April 2019 |
| Akses logging | AWS Elemental MediaStore sekarang mendukung pencatatan akses, yang menyediakan catatan terperinci untuk permintaan yang dibuat ke objek dalam wadah. | 25 Februari 2019 |

| | | |
|---|---|---|
| Kebijakan siklus hidup objek | Menambahkan dukungan untuk kebijakan siklus hidup objek, yang mengatur tanggal kedaluwarsa objek dalam wadah saat ini. | 12 Desember 2018 |
| Peningkatan kuota ukuran objek | Kuota untuk ukuran objek sekarang 25 MB. | 10 Oktober 2018 |
| Peningkatan kuota ukuran objek | Kuota untuk ukuran objek sekarang 20 MB. | 6 September 2018 |
| AWS CloudTrail integrasi | Konten CloudTrail integrasi telah diperbarui agar selaras dengan perubahan terbaru pada CloudTrail layanan. | 12 Juli 2018 |
| Kolaborasi CDN | Menambahkan informasi tentang cara menggunakan AWS Elemental MediaStore dengan jaringan pengiriman konten (CDN) seperti Amazon. CloudFront | April 14, 2018 |
| Konfigurasi CORS | AWS Elemental MediaStore sekarang mendukung berbagi sumber daya lintas asal (CORS), yang memungkin kan aplikasi web klien yang dimuat dalam satu domain untuk berinteraksi dengan sumber daya di domain yang berbeda. | Selasa, 07 Februari 2018 |

| | | |
|---|---|---|
| Layanan dan panduan baru | Ini adalah rilis awal dari layanan originasi dan penyimpanan video, AWS MediaStore Elemental, dan Panduan Pengguna AWS MediaStore Elemental. | 27 November 2017 |

> **Note**
>
> - Layanan AWS Media tidak dirancang atau dimaksudkan untuk digunakan dengan aplikasi atau dalam situasi yang memerlukan kinerja yang gagal, seperti operasi keselamatan jiwa, sistem navigasi atau komunikasi, kontrol lalu lintas udara, atau mesin pendukung kehidupan di mana tidak tersedianya, gangguan atau kegagalan layanan dapat menyebabkan kematian, cedera pribadi, kerusakan properti atau kerusakan lingkungan.

# AWS Glosarium

Untuk AWS terminologi terbaru, lihat AWS glosarium di Referensi.Glosarium AWS