aws

사용자 가이드

AWS 엘레멘탈 MediaStore

AWS 엘레멘탈 MediaStore: 사용자 가이드

Table of Contents

AWS Elemental MediaStore란 무엇인가요?

AWS Elemental MediaStore는 라이브 제작에 필요한 우수한 성능과 즉각적 일관성을 제공하는 비디오 제작 및 스토리지 서비스입니다. MediaStore를 사용하여 비디오 자산을 컨테이너에서 객체로 관리함으로써 안전적인 클라우드 기반 미디어 워크플로를 구축할 수 있습니다.

서비스를 사용하려면 MediaStore에서 인코더나 데이터 피드와 같은 소스의 객체를 만든 컨테이너에 업로드합니다.

MediaStore는 완벽한 일관성, 최소 지연 시간의 읽기 및 쓰기, 많은 양의 동시 요청을 처리할 수 있는 능력이 요구될 때 조각화된 비디오 파일을 저장할 수 있는 훌륭한 수단입니다. 라이브 스트리밍 비디오를 전송하지 않을 경우 Amazon Simple Storage Service(Amazon S3) 를 대신 사용해 보십시오.

주제

AWS Elemental MediaStore 개념 및 용어

ARN

Amazon 리소스 이름.

본문

객체에 업로드할 데이터.

(바이트) 범위

주소 지정할 객체 데이터 하위 집합. 자세한 내용은 HTTP 사양의 범위를 참조하세요.

컨테이너

객체를 포함하는 네임스페이스. 컨테이너에는 객체를 쓰고 검색하고, 액세스 정책을 연결하기 위해 사용할 수 있는 엔드포인트가 있습니다.

엔드포인트

MediaStore 서비스에 대한 진입점(HTTPS 루트 URL로 지정)

ETag

객체 데이터의 해시인 개체 태그.

폴더

컨테이너의 분할 요소. 폴더는 객체와 다른 폴더를 포함할 수 있습니다.

항목

객체와 폴더를 나타내는 데 사용하는 용어.

객체

Amazon S3 객체와 비슷한 자산 객체는 MediaStore에 저장되는 기본 객체입니다. 이 서비스는 모든 유형의 파일을 허용합니다.

제작 서비스

MediaStore는 미디어 콘텐츠 전송을 위한 배포 지점이라는 점에서 제작 서비스로 간주됩니다.

경로

객체나 폴더에 대한 고유 식별자로서, 컨테이너에서의 위치를 나타냅니다.

섹션

객체 데이터의 하위 집합(청크)

Policy

IAM 정책.

리소스

작업에 사용할 수 있는 AWS의 엔터티. 각 AWS 리소스에는 고유한 식별자인 Amazon 리소스 이름(ARN)이 할당됩니다. MediaStore에서 리소스 및 해당 ARN 형식은 다음과 같습니다.

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

관련 서비스

- Amazon CloudFront는 최종 사용자에게 데이터와 비디오를 안전하게 전송하는 글로벌 콘텐츠 전송 네트워크(CDN) 서비스입니다. CloudFront를 사용하여 최고의 성능으로 콘텐츠가 제공됩니다. 자세한 내용은 Amazon CloudFront 개발자 안내서를 참조하세요.

- AWS CloudFormation은 AWS 리소스를 모델링 및 설정할 수 있는 서비스입니다. 필요한 모든 AWS 리소스(예: MediaStore 컨테이너)를 설명하는 템플릿을 생성하면 AWS CloudFormation에서 해당 리소스의 프로비저닝과 구성을 담당합니다. AWS 리소스를 개별적으로 생성하고 구성할 필요가 없으며 어떤 것이 무엇에 의존하는지 파악할 필요도 없습니다. AWS CloudFormation에서 모든 것을 처리합니다. 자세한 내용은 AWS CloudFormation 사용 설명서를 참조하세요.
- AWS CloudTrail란 AWS Management Console, AWS CLI 및 기타 서비스에서 요청한 호출을 포함하여 계정에서 CloudTrail API에 대한 호출을 모니터링할 수 있는 서비스입니다. 자세한 내용은 AWS CloudTrail 사용 설명서를 참조하세요.
- Amazon CloudWatch는 AWS 클라우드 리소스 및 AWS에서 실행하는 애플리케이션을 모니터링하는 서비스입니다. CloudWatch Events를 사용하여 MediaStore의 컨테이너와 객체 상태에 대한 변화를 추적할 수 있습니다. 자세한 내용은 Amazon CloudWatch 설명서를 참조하세요.
- AWS Identity and Access Management(IAM)는 사용자를 위해 AWS 리소스에 대한 액세스를 안전하게 제어할 수 있도록 지원하는 웹 서비스입니다. IAM을 사용하여 AWS 리소스를 사용할 수 있는 사람을 제어(인증)하고 사용자가 사용할 수 있는 리소스 및 사용 방법을 제어(권한 부여)합니다. 자세한 내용은 AWS Elemental 설정 MediaStore 섹션을 참조하세요.
- Amazon Simple Storage Service(Amazon S3)는 어디서든 원하는 양의 데이터를 저장하고 검색할 수 있도록 구축된 객체 스토리지입니다. 자세한 내용은 Amazon S3 설명서를 참조하세요.

AWS Elemental MediaStore에 액세스

다음 방법 중 하나를 사용하여 MediaStore에 액세스할 수 있습니다.

- AWS Management Console - 이 설명서의 절차에서는 AWS Management Console을 사용하여 MediaStore에 대한 작업을 수행하는 방법을 설명합니다. 콘솔을 사용하여 MediaStore에 액세스하기

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

- AWS Command Line Interface – 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요. CLI 엔드포인트를 사용하여 MediaStore에 액세스하기

```
aws mediastore
```

- MediaStore API - SDK가 제공되지 않는 프로그래밍 언어를 사용하는 경우, AWS Elemental MediaStore API 참조에서 API 작업에 대한 정보와 API 요청을 수행하는 방법을 참조하세요. REST API 엔드포인트를 사용하여 MediaStore에 액세스하기

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

- AWS SDK - AWS가 SDK를 제공하는 프로그래밍 언어를 사용하는 경우, SDK를 사용하여 MediaStore에 액세스할 수 있습니다. SDK는 인증을 간편하게 만들고, 개발 환경에 쉽게 통합되며, MediaStore 명령어에 쉽게 액세스할 수 있게 해줍니다. 자세한 내용은 Amazon Web Services용 도구를 참조하세요.
- AWS Tools for Windows PowerShell - 자세한 내용은 AWS Tools for Windows PowerShell 사용 설명서를 참조하세요.

AWS Elemental MediaStore 요금

다른 AWS 제품과 마찬가지로 MediaStore에도 사용할 계약이나 최소 약정이 없습니다. 서비스에 콘텐츠가 제공될 때 GB당 수집 요금이 청구되고, 서비스에 저장하는 콘텐츠에 대해 GB당 월 요금이 청구됩니다. 자세한 내용은 AWS Elemental MediaStore 요금을 참조하세요.

AWS Elemental MediaStore에 사용되는 리전 및 엔드포인트

애플리케이션에서 데이터 지연 시간을 줄이기 위해 MediaStore는 요청을 수행하는 리전 엔드포인트를 제공합니다.

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

MediaStore를 사용할 수 있는 AWS 리전 전체 목록은 AWS 일반 참조에서 AWS Elemental MediaStore 엔드포인트 및 할당량을 참조하세요.

AWS Elemental 설정 MediaStore

이 섹션에서는 사용자가 AWS MediaStore Elemental에 액세스하도록 구성하는 데 필요한 단계를 안내합니다. 자격 증명 및 액세스 관리에 대한 MediaStore 배경 및 추가 정보는 을 참조하십시오AWS Elemental용 자격 증명 및 액세스 관리 MediaStore.

AWS Elemental MediaStore 사용을 시작하려면 다음 단계를 완료하십시오.

주제

가입하십시오. AWS 계정

계정이 없는 경우 다음 단계를 완료하여 계정을 만드세요. AWS 계정

가입하려면 AWS 계정

1. https://portal.aws.amazon.com/billing/signup을 여세요.
2. 온라인 지시 사항을 따르세요.

   등록 절차 중에는 전화를 받고 키패드로 인증 코드를 입력하는 과정이 있습니다.

   에 AWS 계정가입하면 AWS 계정 루트 사용자a가 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스 액세스 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 루트 사용자 액세스 권한이 필요한 작업을 수행하는 것입니다.

AWS 가입 절차가 완료된 후 확인 이메일을 보냅니다. 언제든지 https://aws.amazon.com/으로 가서 내 계정(My Account)을 선택하여 현재 계정 활동을 보고 계정을 관리할 수 있습니다.

관리자 액세스 권한이 있는 사용자 생성

등록한 AWS 계정후에는 일상적인 작업에 루트 사용자를 사용하지 않도록 관리 사용자를 보호하고 AWS IAM Identity Center활성화하고 생성하십시오 AWS 계정 루트 사용자.

보안을 유지하세요. AWS 계정 루트 사용자

1. 루트 사용자를 선택하고 AWS 계정 이메일 주소를 입력하여 계정 AWS Management Console소유자로 로그인합니다. 다음 페이지에서 비밀번호를 입력합니다.

   루트 사용자를 사용하여 로그인하는 데 도움이 필요하면AWS 로그인 사용 설명서의 루트 사용자로 로그인을 참조하세요.
2. 루트 사용자의 다중 인증(MFA)을 활성화합니다.

   지침은 IAM 사용 설명서의 AWS 계정 루트 사용자 (콘솔) 에 대한 가상 MFA 디바이스 활성화를 참조하십시오.

관리자 액세스 권한이 있는 사용자 생성

1. IAM Identity Center를 활성화합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 AWS IAM Identity Center설정을 참조하세요.
2. IAM Identity Center에서 사용자에게 관리 액세스 권한을 부여합니다.

   를 ID 소스로 사용하는 방법에 대한 자습서는 사용 설명서의 기본값으로 IAM Identity Center 디렉터리사용자 액세스 구성을 참조하십시오. IAM Identity Center 디렉터리 AWS IAM Identity Center

관리 액세스 권한이 있는 사용자로 로그인

- IAM IDentity Center 사용자로 로그인하려면 IAM IDentity Center 사용자를 생성할 때 이메일 주소로 전송된 로그인 URL을 사용합니다.

  IAM Identity Center 사용자를 사용하여 로그인하는 데 도움이 필요하면 사용 설명서의 AWS 액세스 포털에 로그인을 참조하십시오.AWS 로그인

추가 사용자에게 액세스 권한 할당

1. IAM Identity Center에서 최소 권한 적용 모범 사례를 따르는 권한 세트를 생성합니다.

   지침은AWS IAM Identity Center 사용 설명서의 Create a permission set를 참조하세요.
2. 사용자를 그룹에 할당하고, 그룹에 Single Sign-On 액세스 권한을 할당합니다.

   지침은AWS IAM Identity Center 사용 설명서의 Add groups를 참조하세요.

AWS Elemental MediaStore 시작하기

이 시작하기 튜토리얼에서는 AWS Elemental MediaStore를 사용하여 컨테이너를 만들고 객체를 업로드하는 방법을 보여 줍니다.

주제

1단계: AWS Elemental MediaStore에 액세스

AWS 계정을 설정하고 사용자와 역할을 만들었으면 AWS Elemental MediaStore용 콘솔에 로그인합니다.

AWS Elemental MediaStore에 액세스하기

- AWS Management Console에 로그인하고 https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.

> **Note**
>
> 이 계정에 대해 만든 IAM 자격 증명을 사용하여 로그인할 수 있습니다. IAM 자격 증명 만들기에 대한 자세한 내용은 AWS Elemental 설정 MediaStore 섹션을 참조하세요.

2단계: 컨테이너 생성

AWS Elemental MediaStore의 컨테이너를 사용하여 폴더와 객체를 저장합니다. 디렉터리로 파일 시스템의 파일을 그룹화하는 것처럼 컨테이너를 사용하여 관련 객체를 그룹화할 수 있습니다. 컨테이너를 만들 때에는 요금이 청구되지 않습니다. 컨테이너에 객체를 업로드해야만 요금이 청구됩니다.

컨테이너를 만들려면

1. 컨테이너 페이지에서 컨테이너 생성을 선택합니다.

2. 컨테이너 이름에 컨테이너 이름을 입력합니다. 자세한 내용은 컨테이너 이름에 대한 규칙 섹션을 참조하세요.
3. 컨테이너 생성을 선택합니다. AWS Elemental MediaStore는 새 컨테이너를 컨테이너 목록에 추가합니다. 처음에 컨테이너의 상태는 생성 중이고 그 다음에 활성으로 변경됩니다.

3단계: 객체 업로드

컨테이너 또는 컨테이너의 폴더에 객체(각각 최대 25MB)를 업로드할 수 있습니다. 폴더에 객체를 업로드하려면 폴더 경로를 지정합니다. 폴더가 이미 있을 경우 AWS Elemental MediaStore는 폴더에 객체를 저장합니다. 폴더가 없으면 폴더를 만든 후 그 폴더에 객체를 저장합니다.

> **Note**
>
> 객체 파일 이름에는 문자, 숫자, 마침표(.), 밑줄(_), 물결 기호(~), 하이픈(-)만 사용할 수 있습니다.

객체를 업로드하려면

1. 컨테이너 페이지에서 방금 만든 컨테이너 이름을 선택합니다. 컨테이너의 세부 정보 페이지가 나타납니다.
2. 객체 업로드를 선택합니다.
3. 대상 경로에 폴더 경로를 입력합니다. 예: `premium/canada` 이 경로에 폴더가 존재하지 않으면 AWS Elemental MediaStore는 해당 폴더를 자동으로 만듭니다.
4. 객체에서 찾아보기를 선택합니다.
5. 해당 폴더로 이동한 후 업로드할 객체를 하나 선택합니다.
6. 열기를 선택한 후 업로드를 선택합니다.

4단계: 객체 액세스

지정한 엔드포인트로 객체를 다운로드할 수 있습니다.

1. 컨테이너 페이지에서, 다운로드하려는 객체가 들어 있는 컨테이너 이름을 선택합니다.
2. 다운로드하려는 객체가 하위 폴더에 있으면 해당 객체가 보일 때까지 폴더 이름을 계속 선택합니다.

3. 객체 이름을 선택합니다.
4. 객체에 대한 세부 정보 페이지에서 다운로드를 선택합니다.

AWS Elemental MediaStore의 컨테이너

MediaStore의 컨테이너를 사용하여 폴더와 객체를 저장합니다. 디렉터리를 사용하여 파일 시스템의 파일을 그룹화하듯이 관련 객체를 컨테이너에 그룹화할 수 있습니다. 컨테이너를 만들 때에는 요금이 청구되지 않습니다. 컨테이너에 객체를 업로드해야만 요금이 청구됩니다. 요금에 대한 자세한 내용은 AWS Elemental MediaStore 요금을 참조하세요.

주제

컨테이너 이름에 대한 규칙

컨테이너의 이름을 선택할 때 다음에 유의하세요.

- 이름은 현재 AWS 리전의 현재 계정에서 고유해야 합니다.
- 이름은 대문자, 소문자, 숫자, 밑줄(_)을 포함할 수 있습니다.
- 이름은 길이가 1자~255자여야 합니다.
- 이름은 대/소문자를 구분합니다. 예를 들어 `myContainer`라는 컨테이너와 `mycontainer`라는 폴더는 이름이 고유하므로 둘 다 둘 수 있습니다.
- 컨테이너를 만든 후 이름을 변경할 수 없습니다.

컨테이너 생성

AWS 계정당 최대 100개의 컨테이너를 만들 수 있습니다. 폴더가 컨테이너 내에서 10개를 초과하는 수준으로 중첩되지 않는 한 원하는 개수만큼 폴더를 생성할 수 있습니다. 또한 각 컨테이너에 원하는 개수만큼 객체를 업로드할 수 있습니다.

> **Tip**
>
> 또한 AWS CloudFormation 템플릿을 사용하여 자동으로 컨테이너를 만들 수도 있습니다. AWS CloudFormation 템플릿은 컨테이너 생성, 액세스 로깅 설정, 기본 컨테이너 정책 업데이트, CORS(교차 원본 리소스 공유) 정책 추가, 객체 수명 주기 정책 추가 등의 5가지 API 작업에 대한 데이터를 관리합니다. 자세한 내용은 AWS CloudFormation 사용 설명서를 참조하세요.

컨테이너를 만들려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서 컨테이너 생성을 선택합니다.
3. 컨테이너 이름에 컨테이너 이름을 입력합니다. 자세한 내용은 컨테이너 이름에 대한 규칙 섹션을 참조하세요.
4. 컨테이너 생성을 선택합니다. AWS Elemental MediaStore는 새 컨테이너를 컨테이너 목록에 추가합니다. 처음에 컨테이너의 상태는 생성 중이고 그 다음에 활성으로 변경됩니다.

컨테이너를 만들려면(AWS CLI)

- AWS CLI에서 `create-container` 명령을 사용합니다.

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

다음 예제는 반환 값을 보여줍니다.

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

컨테이너에 대한 세부 정보 보기

컨테이너 세부 정보에는 컨테이너 정책, 엔드포인트, ARN, 생성 시간이 포함됩니다.

컨테이너에 대한 세부 정보를 보려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서 컨테이너 이름을 선택합니다.

   컨테이너 세부 정보 페이지가 나타납니다. 이 페이지는 두 섹션으로 나뉘어 있습니다.

   - 객체 섹션에는 컨테이너의 객체와 폴더가 나열됩니다.
   - 컨테이너 정책 섹션에는 해당 컨테이너와 연결된 리소스 기반 정책이 표시됩니다. 리소스 정책에 대한 자세한 내용은 컨테이너 정책 섹션을 참조하세요.

컨테이너에 대한 세부 정보를 보려면(AWS CLI)

- AWS CLI에서 `describe-container` 명령을 사용합니다.

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

다음 예제는 반환 값을 보여줍니다.

```
{
    "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"
    }
}
```

컨테이너 목록 보기

계정과 연결된 모든 컨테이너의 목록을 볼 수 있습니다.

컨테이너 목록을 보려면(콘솔)

- https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.

  해당 계정과 연결된 모든 컨테이너가 나열된 컨테이너 페이지가 나타납니다.

컨테이너 목록을 보려면(AWS CLI)

- AWS CLI에서 `list-containers` 명령을 사용합니다.

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

다음 예제는 반환 값을 보여줍니다.

```
{
    "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"
        }
    ]
```

```
}
```

컨테이너 삭제

객체가 들어 있지 않은 컨테이너만 삭제할 수 있습니다.

컨테이너를 삭제하려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서 해당 컨테이너 이름 왼쪽의 옵션을 선택합니다.
3. 삭제를 선택합니다.

컨테이너를 삭제하려면(AWS CLI)

- AWS CLI에서 `delete-container` 명령을 사용합니다.

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

  이 명령은 반환 값이 없습니다.

AWS Elemental MediaStore 정책

AWS Elemental MediaStore 컨테이너에 다음 정책을 하나 이상 적용할 수 있습니다.

- 컨테이너 정책 - 컨테이너 내의 모든 폴더와 객체에 대한 액세스 권한을 설정합니다. MediaStore는 사용자가 컨테이너에서 모든 MediaStore 작업을 수행할 수 있도록 허용하는 기본 정책을 설정합니다. 이 정책은 모든 작업이 HTTPS를 통해 수행되도록 지정합니다. 컨테이너를 생성한 후 컨테이너 정책을 편집할 수 있습니다.
- 교차 오리진 리소스 공유(CORS) 정책 - 한 도메인의 클라이언트 웹 애플리케이션이 다른 도메인의 리소스와 상호 작용할 수 있도록 허용합니다. MediaStore는 기본 CORS 정책을 설정하지 않습니다.
- 지표 정책 - MediaStore가 Amazon CloudWatch에 지표를 전송하도록 허용합니다. MediaStore는 기본 지표 정책을 설정하지 않습니다.
- 객체 수명 주기 정책 - 객체가 MediaStore 컨테이너에 남아 있는 기간을 제어합니다. MediaStore는 기본 객체 수명 주기 정책을 설정하지 않습니다.

AWS Elemental MediaStore의 컨테이너 정책

각 컨테이너에는 해당 컨테이너의 모든 폴더와 객체에 대한 액세스를 관리하는 리소스 기반 정책이 있습니다. 모든 새 컨테이너에 자동으로 연결되는 기본 정책은 컨테이너에서 모든 AWS Elemental MediaStore 작업에 대한 액세스를 허용합니다. 이러한 액세스는 작업에 대해 HTTPS를 요구하는 조건을 갖도록 지정됩니다. 컨테이너를 만든 후 해당 컨테이너에 연결되는 정책을 편집할 수 있습니다.

또한 컨테이너에서 객체의 만료 날짜를 관리하는 객체 수명 주기 정책을 지정할 수 있습니다. 객체가 지정된 최대 수명에 도달하면 서비스가 해당 객체를 컨테이너에서 삭제합니다.

주제

- 컨테이너 정책 보기
- 컨테이너 정책 편집
- 컨테이너 정책 예제

컨테이너 정책 보기

콘솔이나 AWS CLI를 사용하여 컨테이너의 리소스 기반 정책을 볼 수 있습니다.

컨테이너 정책을 보려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서 컨테이너 이름을 선택합니다.

   컨테이너 세부 정보 페이지가 나타납니다. 컨테이너 정책 섹션에 정책이 표시됩니다.

컨테이너 정책을 보려면(AWS CLI)

- AWS CLI에서 `get-container-policy` 명령을 사용합니다.

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

다음 예제는 반환 값을 보여줍니다.

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

컨테이너 정책 편집

기본 컨테이너 정책의 권한을 편집하거나, 기본 정책을 대체할 새 정책을 만들 수 있습니다. 새 정책이 적용되려면 최대 5분이 걸립니다.

컨테이너 정책을 편집하려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서 컨테이너 이름을 선택합니다.
3. 정책 편집을 선택합니다. 다양한 권한을 설정하는 방법을 보여 주는 예제는 the section called “컨테이너 정책 예제” 단원을 참조하십시오.
4. 정책을 적절히 변경하고 저장을 선택합니다.

컨테이너 정책을 편집하려면(AWS CLI)

1. 다음과 같이 컨테이너 정책을 정의하는 파일을 만듭니다.

```
{
  "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. AWS CLI에서 `put-container-policy` 명령을 사용합니다.

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

이 명령은 반환 값이 없습니다.

컨테이너 정책 예제

다음 예제는 여러 사용자 그룹에 대해 구성된 컨테이너 정책을 보여 줍니다.

주제

컨테이너 정책 예제: 기본

컨테이너를 만들면 AWS Elemental MediaStore에서 다음 리소스 기반 정책을 자동으로 연결합니다.

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

```
}
```

이 정책은 서비스에 포함되므로 따로 만들 필요가 없습니다. 그러나 기본 정책의 권한이 컨테이너에 사용하려는 권한과 일치하지 않는 경우 컨테이너에서 정책을 편집할 수 있습니다.

모든 새 컨테이너에 할당되는 기본 정책은 컨테이너에서 모든 MediaStore 작업에 대한 액세스를 허용합니다. 이러한 액세스는 작업에 대해 HTTPS를 요구하는 조건을 갖도록 지정됩니다.

컨테이너 정책 예제: HTTPS를 통해 퍼블릭 읽기 액세스

이 예제 정책은 사용자가 HTTPS 요청을 통해 객체를 검색할 수 있도록 허용합니다. 이 정책은 모든 사용자, 즉 인증된 사용자와 익명 사용자(로그인하지 않은 사용자)에게 보안 SSL/TLS 연결을 통한 읽기 액세스를 허용합니다. 구문에 `PublicReadOverHttps` 이름이 있습니다. 모든 객체(리소스 경로 끝에 * 기호로 지정)에 대해 `GetObject` 및 `DescribeObject` 작업에 대한 액세스를 허용합니다. 이러한 액세스는 작업에 대해 HTTPS를 요구하는 조건을 갖도록 지정됩니다.

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

컨테이너 정책 예제: HTTP 또는 HTTPS를 통한 퍼블릭 읽기 액세스

이 예제 정책은 모든 객체(리소스 경로 끝에 * 기호로 지정)에 대해 `GetObject` 및 `DescribeObject` 작업에 대한 액세스를 허용합니다. 모든 인증된 사용자와 익명 사용자(로그인하지 않은 사용자)를 포함하여 모든 사용자에게 읽기 액세스를 허용합니다.

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

컨테이너 정책 예제: 교차 계정 읽기 액세스-HTTP 활성화

이 예제 정책은 사용자가 HTTP 요청을 통해 객체를 검색할 수 있도록 허용합니다. 교차 계정 액세스 권한을 가진 인증된 사용자에 대해 이 액세스를 허용합니다. 객체가 SSL/TLS 인증서를 사용하여 서버에서 호스트될 필요는 없습니다.

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

컨테이너 정책 예제: HTTPS를 통한 교차 계정 읽기 액세스

이 예제 정책은 지정한 <다른 계정 번호>의 루트 사용자가 소유한 모든 객체(리소스 경로 끝에 * 기호로 지정)에 대해 GetObject 및 DescribeObject 작업에 대한 액세스를 허용합니다. 이러한 액세스는 작업에 대해 HTTPS를 요구하는 조건을 갖도록 지정됩니다.

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

컨테이너 정책 예제: 역할에 대한 교차 계정 읽기 액세스

이 예제 정책은 <소유자 계정 번호>가 소유한 모든 객체(리소스 경로 끝에 * 기호로 지정)에 대해 GetObject 및 DescribeObject 작업에 대한 액세스를 허용합니다. <역할 이름>에 지정된 역할을 해당 계정이 가지고 있을 경우 <다른 계정 번호>의 모든 사용자에 대해 이 액세스를 허용합니다.

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

컨테이너 정책 예제: 역할에 대한 교차 계정 전체 액세스

이 예제 정책은 계정의 객체를 업데이트할 수 있는 교차 계정 액세스 권한을 허용합니다. 단, 사용자가 HTTP를 통해 로그인해야 합니다. 또한 지정된 역할을 맡은 계정에 HTTP 또는 HTTPS를 통해 객체를 삭제, 다운로드 및 설명할 수 있는 교차 계정 액세스 권한을 허용합니다.

- 첫째 구문은 CrossAccountRolePostOverHttps입니다. 이 구문은 모든 객체에 대해 PutObject 작업에 대한 액세스를 허용하고, 지정한 계정이 <역할 이름>에 지정된 역할을 가지고 있을 경우 해당 계정의 사용자에 대해 이 액세스를 허용합니다. 이러한 액세스는 작업에 대해 HTTPS를 요구하는 조건을 갖도록 지정됩니다. PutObject에 대한 액세스를 제공할 때 이 조건이 항상 포함되어야 합니다.

  즉, 교차 계정 액세스 권한을 가진 보안 주체는 PutObject에 액세스할 수 있지만 HTTPS를 통해야만 합니다.
- 두 번째 구문은 CrossAccountFullAccessExceptPost입니다. 이 구문은 객체에 대해 PutObject를 제외한 모든 작업에 대한 액세스를 허용합니다. <역할 이름>에 지정된 역할을 해당 계정이 가지고 있을 경우 해당 계정의 사용자에 대해 이 액세스를 허용합니다. 이러한 액세스는 작업에 대해 HTTPS를 요구하는 조건을 갖지 않습니다.

  즉, 교차 계정 액세스 권한을 가진 계정은 DeleteObject, GetObject 등(PutObject 제외)에 액세스할 수 있고, HTTP 또는 HTTPS를 통해 이 작업을 수행할 수 있습니다.

  두 번째 구문에서 PutObject를 제외시키지 않으면 구문이 유효하지 않게 됩니다. PutObject를 포함시킬 경우 조건으로 HTTPS를 명시적으로 설정해야 하기 때문입니다.

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

컨테이너 정책 예제: 특정 IP 주소로 제한된 액세스

이 예제 정책은 지정된 컨테이너의 객체에 대한 모든 AWS Elemental MediaStore 작업에 액세스하도록 허용합니다. 하지만 조건에 지정된 IP 주소 범위에서만 요청을 허용해야 합니다.

이 문의 조건은 허용되는 IPv4(인터넷 프로토콜 버전 4) IP 주소인 198.51.100.* 범위를 식별합니다(한 가지 예외: 198.51.100.188).

`Condition` 블록은 `IpAddress` 및 `NotIpAddress` 조건과 AWS 전체 범위 조건 키인 `aws:SourceIp` 조건 키를 사용합니다. `aws:sourceIp` IPv4 값은 표준 CIDR 표기법을 사용합니다. 자세한 내용은 IAM 사용 설명서의 IP 주소 조건 연산자를 참조하세요.

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

AWS Elemental MediaStore의 교차 오리진 리소스 공유(CORS) 정책

CORS(Cross-origin 리소스 공유)는 한 도메인에서 로드되어 다른 도메인에 있는 리소스와 상호 작용하는 클라이언트 웹 애플리케이션에 대한 방법을 정의합니다. AWS Elemental MediaStore의 CORS 지원을 통해 MediaStore로 다양한 기능의 클라이언트 측 웹 애플리케이션을 구축하고, MediaStore 리소스에 대한 cross-origin 액세스를 선택적으로 허용할 수 있습니다.

> **Note**
>
> Amazon CloudFront를 사용하여 CORS 정책을 포함하는 컨테이너에서 콘텐츠를 배포하는 경우 반드시 AWS Elemental MediaStore에 대해 배포를 구성해야 합니다(CORS 설정을 위한 캐시 동작을 편집하는 단계 포함).

이 섹션에서는 CORS 개요를 다룹니다. 하위 주제로 AWS Elemental MediaStore 콘솔을 사용하여 CORS를 활성화하거나, 프로그래밍 방식으로 MediaStor REST API 및 AWS SDK를 사용하는 방법을 설명합니다.

주제

- CORS 사용 사례 시나리오
- 컨테이너에 CORS 정책 추가

- CORS 정책 보기
- CORS 정책 편집
- CORS 정책 삭제
- CORS 문제 해결
- CORS 정책 예제

CORS 사용 사례 시나리오

다음은 CORS 사용에 대한 예제 시나리오입니다.

- 시나리오 1: LiveVideo라는 AWS Elemental MediaStore 컨테이너에서 라이브 스트리밍 비디오를 배포한다고 가정해 보겠습니다. 사용자가 `http://livevideo.mediastore.ap-southeast-2.amazonaws.com`과 같은 특정 오리진에서 비디오 매니페스트 엔드포인트 `www.example.com`을 로드합니다. JavaScript 비디오 플레이어를 사용하여 이 컨테이너에서 제공하는 비디오를, 인증되지 않은 GET 요청과 PUT 요청을 통해 액세스하려고 합니다. 일반적으로 브라우저에서 이러한 요청을 허용하지 않도록 JavaScript를 차단하지만, 컨테이너에 CORS 정책을 설정하여 `www.example.com`으로부터의 이 요청을 명시적으로 허용할 수 있습니다.
- 시나리오 2: 시나리오 1과 같은 라이브 스트림을 MediaStore 컨테이너에서 호스팅하려 하며, 오리진으로부터의 요청을 허용하려 한다고 가정해 보겠습니다. 와일드카드(*) 오리진을 허용하도록 CORS 정책을 구성하여 모든 오리진으로부터의 요청이 비디오를 액세스하게 할 수 있습니다.

컨테이너에 CORS 정책 추가

이 섹션에서는 AWS Elemental MediaStore 컨테이너에 교차 오리진 리소스 공유(CORS) 구성을 추가하는 방법을 설명합니다. CORS는 한 도메인에서 로드되어 다른 도메인에 있는 리소스와 상호 작용하는 클라이언트 웹 애플리케이션을 허용합니다.

cross-origin 요청을 허용하도록 컨테이너를 구성하려면 컨테이너에 CORS 정책을 추가합니다. CORS 정책은 컨테이너에 대한 액세스를 허용할 오리진과 각 오리진에 대해 지원되는 작업(HTTP 메서드)을 식별하는 규칙과 기타 작업별 정보를 정의합니다.

컨테이너에 CORS 정책을 추가하면 컨테이너 정책(컨테이너에 대한 액세스 권한을 관리하는 정책)이 계속 적용됩니다.

CORS 정책을 추가하려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.

2. 컨테이너 페이지에서, CORS 정책을 만들려는 컨테이너의 이름을 선택합니다.

   컨테이너 세부 정보 페이지가 나타납니다.

3. 컨테이너 CORS 정책 섹션에서 CORS 정책 생성을 선택합니다.

4. JSON 형식으로 정책을 삽입한 후 저장을 선택합니다.

CORS 정책을 추가하려면(AWS CLI)

1. 다음과 같이 CORS 정책을 정의하는 파일을 만듭니다.

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

2. AWS CLI에서 `put-cors-policy` 명령을 사용합니다.

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

   이 명령은 반환 값이 없습니다.

CORS 정책 보기

교차 오리진 리소스 공유(CORS) 한 도메인에서 로드되어 다른 도메인에 있는 리소스와 상호 작용하는 클라이언트 웹 애플리케이션에 대한 방법을 정의합니다.

CORS 정책을 보려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서, CORS 정책을 보려는 컨테이너의 이름을 선택합니다.

   컨테이너 세부 정보 페이지가 나타나고 컨테이너 CORS 정책 섹션에 CORS 정책이 표시됩니다.

CORS 정책을 보려면(AWS CLI)

- AWS CLI에서 `get-cors-policy` 명령을 사용합니다.

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

다음 예제는 반환 값을 보여줍니다.

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

CORS 정책 편집

CORS(Cross-origin 리소스 공유)는 한 도메인에서 로드되어 다른 도메인에 있는 리소스와 상호 작용하는 클라이언트 웹 애플리케이션에 대한 방법을 정의합니다.

CORS 정책을 편집하려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서, CORS 정책을 편집하려는 컨테이너의 이름을 선택합니다.

   컨테이너 세부 정보 페이지가 나타납니다.
3. 컨테이너 CORS 정책 섹션에서 CORS 정책 편집을 선택합니다.
4. 정책을 변경하고 저장을 선택합니다.

CORS 정책을 편집하려면(AWS CLI)

1. 다음과 같이 업데이트된 CORS 정책을 정의하는 파일을 만듭니다.

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

2. AWS CLI에서 `put-cors-policy` 명령을 사용합니다.

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

   이 명령은 반환 값이 없습니다.

CORS 정책 삭제

CORS(Cross-origin 리소스 공유)는 한 도메인에서 로드되어 다른 도메인에 있는 리소스와 상호 작용하는 클라이언트 웹 애플리케이션에 대한 방법을 정의합니다. 컨테이너에서 CORS 정책을 삭제하면 cross-origin 요청에 대한 권한이 제거됩니다.

CORS 정책을 삭제하려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서, CORS 정책을 삭제하려는 컨테이너의 이름을 선택합니다.

   컨테이너 세부 정보 페이지가 나타납니다.
3. 컨테이너 CORS 정책 섹션에서 CORS 정책 삭제를 선택합니다.
4. 계속을 선택하여 확인한 다음 저장을 선택하세요.

CORS 정책을 삭제하려면(AWS CLI)

- AWS CLI에서 `delete-cors-policy` 명령을 사용합니다.

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

  이 명령은 반환 값이 없습니다.

CORS 문제 해결

CORS 정책이 있는 컨테이너에 액세스할 때 예기치 않은 동작이 발생할 경우 다음 단계에 따라 문제를 해결하세요.

1. 버킷에 CORS 정책이 연결되어 있는지 확인합니다.

   지침은 the section called “CORS 정책 보기” 섹션을 참조하세요.
2. 원하는 도구(예: 브라우저의 개발자 콘솔)를 사용하여 완료 요청 및 응답을 캡처합니다. 컨테이너에 연결된 CORS 정책에 해당 요청의 데이터와 일치하는 CORS 규칙이 한 개 이상 포함되어 있는지를 다음과 같이 확인합니다.

   a. 요청에 `Origin` 헤더가 있는지 확인합니다.

헤더가 없으면 AWS Elemental MediaStore는 요청을 cross-origin 요청으로 처리하지 않고, 응답에 CORS 응답 헤더를 돌려 보내지 않습니다.

b. 요청의 `Origin` 헤더가 해당 `AllowedOrigins`의 `CORSRule` 요소 중 최소 하나와 일치하는지 확인합니다.

`Origin` 요청 헤더의 체계, 호스트, 포트 값이 `AllowedOrigins`의 `CORSRule`와 일치해야 합니다. 예를 들어 `CORSRule`을 설정하여 `http://www.example.com` 오리진을 허용한 경우, 요청의 `https://www.example.com` 오리진과 `http://www.example.com:80` 오리진은 해당 구성의 허용되는 오리진과 일치하지 않습니다.

c. 요청의 메서드(preflight 요청의 경우 `Access-Control-Request-Method`에 지정된 메서드)가 동일한 `AllowedMethods`의 `CORSRule` 요소의 메서드 중 하나인지 확인합니다.

d. preflight 요청의 경우 요청에 `Access-Control-Request-Headers` 헤더가 있을 경우, `CORSRule` 헤더에 각 값에 대한 `AllowedHeaders` 항목이 `Access-Control-Request-Headers`에 포함되었는지 확인합니다.

CORS 정책 예제

다음 예제는 CORS(Cross-Origin Resource Sharing) 정책을 보여 줍니다.

주제

- CORS 정책 예제: 모든 도메인에 대한 읽기 액세스 권한
- CORS 정책 예제: 특정 도메인에 대한 읽기 액세스 권한

CORS 정책 예제: 모든 도메인에 대한 읽기 액세스 권한

다음 정책은 모든 도메인의 웹 페이지에서 AWS Elemental MediaStore 컨테이너의 콘텐츠를 가져올 수 있도록 허용합니다. 이 요청은 발신 도메인의 모든 HTTP 헤더를 포함하고, 서비스는 발신 도메인의 HTTP GET 요청과 HTTP HEAD 요청에만 응답합니다. 요청은 3,000초 동안 캐싱되며, 그 후에는 새로운 결과 집합이 전송됩니다.

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

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

CORS 정책 예제: 특정 도메인에 대한 읽기 액세스 권한

다음 정책은 `https://www.example.com`의 웹 페이지에서 AWS Elemental MediaStore 컨테이너의 콘텐츠를 가져올 수 있도록 허용합니다. 이 요청은 `https://www.example.com`의 모든 HTTP 헤더를 포함하고, 서비스는 `https://www.example.com`의 HTTP GET 요청과 HTTP HEAD 요청에만 응답합니다. 요청은 3,000초 동안 캐싱되며, 그 후에는 새로운 결과 집합이 전송됩니다.

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

AWS Elemental MediaStore의 객체 수명 주기 정책

각 컨테이너에 대해 해당 컨테이너에 저장된 객체가 유지되는 기간을 관리하는 객체 수명 주기 정책을 만들 수 있습니다. 객체가 지정된 최대 수명에 도달하면 AWS Elemental MediaStore가 해당 객체를 삭제합니다. 스토리지 비용을 절약하기 위해 더 이상 필요하지 않은 객체를 삭제할 수 있습니다.

특정 기간이 지나면 MediaStore는 객체를 IA(액세스 빈도 낮음) 스토리지 클래스로 이동하도록 지정할 수도 있습니다. IA 스토리지 클래스에 저장된 객체는 저장 및 검색 속도가 표준 스토리지 클래스에 저장된 객체와 다릅니다. 자세한 내용은 MediaStore 요금을 참조하세요.

객체 수명 주기 정책은 하위 폴더별로 객체의 수명을 지정하는 규칙을 포함합니다. (개별 객체에 객체 수명 주기 정책을 할당할 수는 없습니다.) 각 컨테이너에 하나의 객체 수명 주기 정책만 연결할 수 있지만, 각 객체 수명 주기 정책에 최대 10개의 규칙을 추가할 수 있습니다. 자세한 내용은 객체 수명 주기 정책의 구성 요소 섹션을 참조하세요.

주제

객체 수명 주기 정책의 구성 요소

객체 수명 주기 정책은 AWS Elemental MediaStore 컨테이너에서 객체가 유지되는 기간을 관리합니다. 각 객체 수명 주기 정책은 객체의 수명을 지정하는 하나 이상의 규칙으로 구성됩니다. 각 규칙은 한 폴더, 여러 폴더 또는 전체 컨테이너에 적용될 수 있습니다.

각 객체 수명 주기 정책을 한 컨테이너에 연결할 수 있고 각 객체 수명 주기 정책은 최대 10개의 규칙을 포함할 수 있습니다. 개별 객체에 객체 수명 주기 정책을 할당할 수는 없습니다.

객체 수명 주기 정책의 규칙

세 가지 유형의 규칙을 만들 수 있습니다.

- 임시 데이터
- 객체 삭제
- 수명 주기 전환

임시 데이터

임시 데이터 규칙은 객체가 수 초 내에 만료되도록 설정합니다. 이 유형의 규칙은 정책 실행 후 컨테이너에 추가된 객체에만 적용됩니다. MediaStore가 새 정책을 컨테이너에 적용하는 데 최대 20분이 걸립니다.

일시적인 데이터 규칙의 예는 다음과 같습니다:

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

임시 데이터 규칙에는 다음 세 부분이 있습니다.

- `path`: 항상 `wildcard`로 설정 이 부분을 사용하여 삭제할 객체를 정의합니다. 별표(*)로 표시되는 하나 이상의 와일드카드를 사용할 수 있습니다. 각 와일드카드는 0개 이상의 문자 조합을 나타냅니다. 예를 들어, `"path": [ {"wildcard": "Football/index*.m3u8"} ],`은 `index*.m3u8`의 패턴(예: index.m3u8, index1.m3us8, index123456.m3u8)과 일치하는 `Football` 폴더의 모든 파일에 적용됩니다. 단일 규칙에 최대 10개의 경로를 포함할 수 있습니다.
- `seconds_since_create`: 항상 `numeric`로 설정 1초~300초의 값을 지정할 수 있습니다. 연산자를 초과(>) 또는 이상(>=)으로 설정할 수도 있습니다.
- `action`: 항상 `EXPIRE`로 설정

임시 데이터 규칙(객체가 몇 초 내에 만료됨)의 경우 객체 만료와 객체 삭제 사이에 지연이 없습니다.

> **Note**
>
> 임시 규칙이 적용되는 객체는 `list-items` 응답에 포함되지 않습니다. 또한 임시 데이터 규칙으로 인해 만료되는 객체는 만료 시 CloudWatch 이벤트를 생성하지 않습니다.

객체 삭제

객체 삭제 규칙은 객체가 며칠 내에 만료되도록 설정합니다. 이러한 유형의 규칙은 정책이 생성되기 전에 컨테이너에 추가된 경우에도 컨테이너의 모든 객체에 적용됩니다. MediaStore가 새 정책을 적용하는 데 최대 20분이 걸리되지만 컨테이너에서 객체를 지우려면 최대 24시간이 걸릴 수 있습니다.

두 가지 객체 삭제 규칙의 예는 다음과 같습니다.

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

객체 삭제 규칙에는 다음 세 부분이 있습니다.

- path: prefix 또는 wildcard로 설정합니다. 같은 규칙에서 prefix와 wildcard를 함께 사용할 수 없습니다. 두 가지 모두를 사용하려면 위의 예와 같이 prefix에 대한 규칙과 wildcard에 대한 규칙을 별도로 생성해야 합니다.
  - prefix - 특정 폴더 내의 모든 객체를 삭제하려면 경로를 prefix로 설정합니다. 매개 변수가 비어있는 경우("path": [{ "prefix": "" }],) 대상은 현재 컨테이너 내의 아무 곳에나 저장된 모든 객체입니다. 단일 규칙에 최대 10개의 prefix 경로를 포함할 수 있습니다.
  - wildcard - 파일 이름 및/또는 파일 형식에 따라 특정 객체를 삭제하려면 경로를 wildcard로 설정합니다. 별표(*)로 표시되는 하나 이상의 와일드카드를 사용할 수 있습니다. 각 와일드카드는 0개 이상의 문자 조합을 나타냅니다. 예를 들어, "path": [{"wildcard": "Football/*.ts"}],는 *.ts의 패턴(예: filename.ts, filename1.ts, filename123456.ts)과 일치하는

Football 폴더의 모든 파일에 적용됩니다. 단일 규칙에 최대 10개의 wildcard 경로를 포함할 수 있습니다.

- days_since_create: 항상 numeric로 설정 1일~36,500일의 값을 지정할 수 있습니다. 연산자를 초과(>) 또는 이상(>=)으로 설정할 수도 있습니다.
- action: 항상 EXPIRE로 설정

객체 삭제 규칙(객체가 며칠 내에 만료됨)의 경우 객체 만료와 객체 삭제 사이에 약간의 지연이 있을 수 있습니다. 그러나 객체가 만료되자마자 결제에 변경 사항이 발생합니다. 예를 들어 수명 주기 규칙에서 days_since_create를 10으로 지정하는 경우 계정은 객체가 아직 삭제되지 않았어도 객체 생성 후 10일이 지나면 객체에 대한 요금이 부과되지 않습니다.

수명 주기 전환

수명 주기 전환 규칙은 며칠 단위로 측정하여 일정 기간이 지난 객체를 IA(액세스 빈도 낮음) 스토리지 클래스로 이동하도록 설정합니다. IA 스토리지 클래스에 저장된 객체는 저장 및 검색 속도가 표준 스토리지 클래스에 저장된 객체와 다릅니다. 자세한 내용은 MediaStore 요금을 참조하세요.

객체를 IA 스토리지 클래스로 이동한 후에는 표준 스토리지 클래스로 다시 이동할 수 없습니다.

수명 주기 전환 규칙은 정책이 생성되기 전에 컨테이너에 추가되었더라도 컨테이너의 모든 객체에 적용됩니다. MediaStore가 새 정책을 적용하는 데 최대 20분이 걸리지만 컨테이너에서 객체를 지우려면 최대 24시간이 걸릴 수 있습니다.

수명 주기 전환 규칙의 예는 다음과 같습니다.

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

수명 주기 전환 규칙은 다음 세 부분으로 이루어집니다.

- path: prefix 또는 wildcard로 설정합니다. 같은 규칙에서 prefix와 wildcard를 함께 사용할 수 없습니다. 둘 다 사용하려면 prefix의 규칙 하나와 wildcard의 규칙 하나를 별도로 만들어야 합니다.
  - prefix - 특정 폴더 내의 모든 객체를 IA 스토리지 클래스로 전환하려면 경로를 prefix로 설정합니다. 이 파라미터가 비어 있는 경우("path": [{ "prefix": "" }],), 위치와 관계없이 현재 컨테이너에 저장되어 있는 모든 객체가 대상이 됩니다. 단일 규칙에 최대 10개의 prefix 경로를 포함할 수 있습니다.
  - wildcard - 파일 이름 및/또는 파일 형식에 따라 특정 객체를 IA 스토리지 클래스로 전환하려면 경로를 wildcard로 설정합니다. 별표(*)로 표시되는 하나 이상의 와일드카드를 사용할 수 있습니다. 각 와일드카드는 0개 이상의 문자 조합을 나타냅니다. 예를 들어, "path": [{"wildcard": "Football/*.ts"}],는 *.ts의 패턴(예: filename.ts, filename1.ts, filename123456.ts)과 일치하는 Football 폴더의 모든 파일에 적용됩니다. 단일 규칙에 최대 10개의 wildcard 경로를 포함할 수 있습니다.
- days_since_create: 항상 "numeric": [">=" , 30]로 설정
- action: 항상 ARCHIVE로 설정

예

예를 들어 컨테이너 LiveEvents에 4개의 하위 폴더 Football, Baseball, Basketball, AwardsShow가 있습니다. LiveEvents 폴더에 할당된 객체 수명 주기 정책은 다음과 같을 수 있습니다.

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

위 정책은 다음을 지정합니다.

- 첫 번째 규칙은 `LiveEvents/Football` 폴더 및 `LiveEvents/Baseball` 폴더에 저장된 객체가 28일을 경과하면 AWS Elemental MediaStore가 해당 객체를 삭제하도록 지시합니다.
- 두 번째 규칙은 `LiveEvents/AwardsShow` 폴더에 저장된 객체가 15일을 경과하면 서비스가 해당 객체를 삭제하도록 지시합니다.
- 세 번째 규칙은 `LiveEvents` 컨테이너에 저장된 객체가 40일을 경과하면 서비스가 해당 객체를 삭제하도록 지시합니다. 이 규칙은 `LiveEvents` 컨테이너에 직접 저장된 객체뿐 아니라 컨테이너의 4개 하위 폴더에 저장된 객체에도 적용됩니다.
- 네 번째 규칙은 `*.ts` 패턴과 일치하는 `Football` 폴더의 객체가 20일을 경과하면 서비스가 해당 객체를 삭제하도록 지시합니다.
- 다섯 번째 규칙은 `Football` 패턴과 일치하는 `index*.m3u8` 폴더의 객체가 15초를 경과하면 서비스가 해당 객체를 삭제하도록 지시합니다. MediaStore는 이러한 파일을 컨테이너에 배치한 후 16초 후에 삭제합니다.
- 여섯 번째 규칙은 30일이 지난 `Program` 폴더의 객체를 IA 스토리지 클래스로 이동하도록 서비스에 지시합니다.

객체 수명 주기 정책의 자세한 예는 객체 수명 주기 정책의 예 단원을 참조하십시오.

컨테이너에 객체 수명 주기 정책 추가

객체 수명 주기 정책을 사용하면 객체가 컨테이너에 저장되는 기간을 저장할 수 있습니다. 만료 날짜를 설정하면 만료 날짜 후 AWS Elemental MediaStore가 해당 객체를 삭제합니다. 서비스가 새 정책을 컨테이너에 적용하는 데 최대 20분 걸립니다.

수명 주기 정책을 구성하는 방법에 대한 자세한 내용은 객체 수명 주기 정책의 구성 요소 단원을 참조하십시오.

> **Note**
>
> 객체 삭제 규칙(객체가 며칠 내에 만료됨)의 경우 객체 만료와 객체 삭제 사이에 약간의 지연이 있을 수 있습니다. 그러나 객체가 만료되자마자 결제에 변경 사항이 발생합니다. 예를 들어 수

명 주기 규칙에서 days_since_create를 10으로 지정하는 경우 계정은 객체가 아직 삭제되지 않았어도 객체 생성 후 10일이 지나면 객체에 대한 요금이 부과되지 않습니다.

객체 수명 주기 정책을 추가하려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서, 객체 수명 주기 정책을 만들려는 컨테이너의 이름을 선택합니다.

   컨테이너 세부 정보 페이지가 나타납니다.
3. 객체 수명 주기 정책 섹션에서 객체 수명 주기 정책 생성을 선택합니다.
4. JSON 형식으로 정책을 삽입한 후 저장을 선택합니다.

객체 수명 주기 정책 추가(AWS CLI)

1. 객체 수명 주기 정책을 정의하는 파일을 만듭니다.

```
{
    "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. AWS CLI에서 `put-lifecycle-policy` 명령을 사용합니다.

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

이 명령은 반환 값이 없습니다. 서비스가 지정된 정책을 컨테이너에 연결합니다.

객체 수명 주기 정책 보기

객체 수명 주기 정책은 객체를 컨테이너에 유지할 기간을 지정합니다.

객체 수명 주기 정책 조회(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서, 객체 수명 주기 정책을 보려는 컨테이너의 이름을 선택합니다.

   객체 수명 주기 정책 섹션에 객체 수명 주기 정책과 함께 컨테이너 세부 정보 페이지가 나타납니다.

객체 수명 주기 정책을 보려면(AWS CLI)

- AWS CLI에서 `get-lifecycle-policy` 명령을 사용합니다.

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

다음 예제는 반환 값을 보여줍니다.

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

```
                    "days_since_create": [
                        {"numeric": [">" , 28]}
                    ]
                },
             "action": "EXPIRE"
            }
        ]
    }"
}
```

객체 수명 주기 정책 편집

기존 객체 수명 주기 정책을 편집할 수는 없습니다. 하지만 대체 정책을 업로드하면 기존 정책을 변경할 수 있습니다. 서비스가 업데이트된 정책을 컨테이너에 적용하는 데 최대 20분 걸립니다.

객체 수명 주기 정책 편집(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서, 객체 수명 주기 정책을 편집하려는 컨테이너의 이름을 선택합니다.

   컨테이너 세부 정보 페이지가 나타납니다.
3. 객체 수명 주기 정책 섹션에서 객체 수명 주기 정책 편집을 선택합니다.
4. 정책을 변경하고 저장을 선택합니다.

객체 수명 주기 정책을 편집하려면(AWS CLI)

1. 업데이트된 객체 수명 주기 정책을 정의하는 파일을 만듭니다.

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

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

2. AWS CLI에서 `put-lifecycle-policy` 명령을 사용합니다.

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

이 명령은 반환 값이 없습니다. 서비스가 지정된 정책을 이전 정책 대신 컨테이너에 연결합니다.

객체 수명 주기 정책 삭제

객체 수명 주기 정책을 삭제할 경우 서비스가 변경 사항을 컨테이너에 적용하려면 최대 20분이 걸립니다.

객체 수명 주기 정책을 삭제하려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서, 객체 수명 주기 정책을 삭제하려는 컨테이너의 이름을 선택합니다.

   컨테이너 세부 정보 페이지가 나타납니다.
3. 객체 수명 주기 정책 섹션에서 객체 수명 주기 정책 삭제를 선택합니다.
4. 계속을 선택하여 확인한 다음 저장을 선택하세요.

객체 수명 주기 정책을 삭제하려면(AWS CLI)

- AWS CLI에서 `delete-lifecycle-policy` 명령을 사용합니다.

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

이 명령은 반환 값이 없습니다.

객체 수명 주기 정책의 예

다음은 객체 수명 주기 정책의 예입니다.

주제

객체 수명 주기 정책의 예: 몇 초 안에 만료

다음 정책은 다음 조건을 모두 충족하는 객체를 삭제하도록 MediaStore에 지시합니다.

- 정책이 적용되면 객체가 컨테이너에 추가됩니다.
- 객체가 `Football` 폴더에 저장됩니다.
- 객체의 파일 확장자는 `m3u8`입니다.
- 객체가 컨테이너에 20초 이상 있었습니다.

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

객체 수명 주기 정책의 예: 며칠 안에 만료

다음 정책은 다음 조건을 모두 충족하는 객체를 삭제하도록 MediaStore에 지시합니다.

- 객체가 `Program` 폴더에 저장됩니다.
- 이 객체의 파일 확장자는 `ts`입니다.
- 객체가 컨테이너에 5일 이상 있었습니다.

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

객체 수명 주기 정책의 예: 액세스 빈도가 낮은 스토리지 클래스로 전환

다음 정책은 30일이 지난 객체를 IA(액세스 빈도 낮음) 스토리지 클래스로 이동하도록 MediaStore에 지시합니다. IA 스토리지 클래스에 저장된 객체는 저장 및 검색 속도가 표준 스토리지 클래스에 저장된 객체와 다릅니다.

`days_since_create` 필드를 `"numeric": [">=" ,30]`으로 설정해야 합니다.

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

```
                "days_since_create": [
                    {"numeric": [">=" , 30]}
                ]
            },
            "action": "ARCHIVE"
        }
    ]
}
```

객체 수명 주기 정책의 예: 복수의 규칙

다음 정책은 다음을 수행하도록 MediaStore에 지시합니다.

- AwardsShow 폴더에 저장된 객체를 30일 후 IA(액세스 빈도 낮음) 스토리지 클래스로 이동
- 파일 확장자가 m3u8이고 Football 폴더에 저장된 객체를 20초 후 삭제
- April 폴더에 저장된 객체를 10일 후 삭제
- 파일 확장자가 ts이고 Program 폴더에 저장된 객체를 5일 후 삭제

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

객체 수명 주기 정책의 예: 빈 컨테이너

다음 객체 수명 주기 정책은 컨테이너에 객체가 추가되고 1일 후 폴더 및 하위 폴더를 포함하여 컨테이너에 있는 모든 객체를 삭제하도록 MediaStore에 지시합니다. 이 정책이 적용되기 전에 컨테이너에 객체가 있었다면 MediaStore는 정책이 적용되고 1일 후에 해당 개체를 삭제합니다. 서비스가 새 정책을 컨테이너에 적용하는 데 최대 20분 걸립니다.

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

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

AWS Elemental MediaStore의 지표 정책

각 컨테이너의 경우 지표 정책을 추가하여 AWS Elemental MediaStore가 Amazon CloudWatch에 지표를 전송할 수 있습니다. 새 정책이 적용되려면 최대 20분이 걸립니다. 각 MediaStore 지표에 대한 설명은 MediaStore 지표 섹션을 참조하세요.

지표 정책에는 다음이 포함됩니다.

- 컨테이너 수준에서 지표를 활성화 또는 비활성화하는 설정입니다.
- 객체 수준에서 지표를 활성화하는 0에서 5개의 규칙 정책에 규칙이 포함되어 있는 경우 각 규칙에 다음 사항이 모두 포함되어야 합니다.
  - 그룹에 포함할 객체를 정의하는 객체 그룹입니다. 정의는 경로 또는 파일 이름일 수 있지만 900자를 초과할 수 없습니다. 유효한 문자는 a-z, A-Z, 0-9, _(밑줄), =(같음), :(콜론), .(마침표), -(하이픈), ~(물결표), /(슬래시) 및 *(별표)입니다. 와일드카드(*)를 사용할 수 있습니다.
  - 객체 그룹을 참조할 수 있는 객체 그룹 이름입니다. 이름은 30자를 초과할 수 없습니다. 유효한 문자는 a-z, A-Z, 0-9 및 –(밑줄)입니다.

객체가 여러 규칙과 일치하는 경우 CloudWatch는 일치하는 각 규칙에 대한 데이터 포인트를 표시합니다. 예를 들어 객체가 `rule1` 및 `rule2`라는 두 개의 규칙과 일치하는 경우 CloudWatch는 이러한 규칙에 대한 두 개의 데이터 포인트를 표시합니다. 첫 번째는 차원이 `ObjectGroupName=rule1`이고 두 번째는 차원이 `ObjectGroupName=rule2`입니다.

주제

- 지표 정책 추가
- 지표 정책 보기
- 지표 정책 편집
- 지표 정책 예제

지표 정책 추가

지표 정책에는 AWS Elemental MediaStore가 Amazon CloudWatch로 전송하는 지표를 지정하는 규칙이 포함되어 있습니다. 지표 정책의 예제는 지표 정책 예제 섹션을 참조하세요.

지표 정책을 추가하려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서 지표 정책을 추가할 컨테이너의 이름을 선택합니다.

   컨테이너 세부 정보 페이지가 나타납니다.
3. 지표 정책 섹션에서 지표 정책 생성을 선택합니다.
4. JSON 형식으로 정책을 삽입한 후 저장을 선택합니다.

지표 정책 보기

콘솔이나 AWS CLI를 사용하여 컨테이너의 지표 정책을 볼 수 있습니다.

지표 정책을 보려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서 컨테이너 이름을 선택합니다.

   컨테이너 세부 정보 페이지가 나타납니다. 정책이 지표 정책 섹션에 표시됩니다.

지표 정책 편집

지표 정책에는 AWS Elemental MediaStore가 Amazon CloudWatch로 전송하는 지표를 지정하는 규칙이 포함되어 있습니다. 기존 지표 정책을 편집할 때 새 정책이 적용되기까지 최대 20분이 걸립니다. 지표 정책의 예제는 지표 정책 예제 섹션을 참조하세요.

지표 정책을 편집하려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서 컨테이너 이름을 선택합니다.
3. 지표 정책 섹션에서 지표 정책 편집을 선택합니다.
4. 정책을 적절히 변경하고 저장을 선택합니다.

지표 정책 예제

다음 예제는 여러 사용 사례에 대해 구성된 지표 정책을 보여줍니다.

주제

지표 정책 예제: 컨테이너 수준 지표

이 정책 예제는 AWS Elemental MediaStore가 컨테이너 수준에서 지표를 Amazon CloudWatch에 전송해야 함을 나타냅니다. 예를 들어 컨테이너에 대한 `Put` 요청 수를 계산하는 `RequestCount` 지표가 포함됩니다. 또는 이를 `DISABLED`로 설정할 수 있습니다.

이 정책에는 규칙이 없으므로 MediaStore에서는 경로 수준에서 지표를 전송하지 않습니다. 예를 들어 이 컨테이너 내의 특정 폴더에 대한 `Put` 요청 수를 볼 수 없습니다.

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

지표 정책 예제: 경로 수준 지표

이 정책 예제는 AWS Elemental MediaStore가 컨테이너 수준에서 지표를 Amazon CloudWatch에 전송하지 않아야 함을 나타냅니다. 또한 MediaStore에서는 `baseball/saturday` 및 `football/saturday`의 두 폴더의 객체에 대한 지표를 전송해야 합니다. MediaStore 요청에 대한 지표는 다음과 같습니다.

- `baseball/saturday` 폴더에 대한 요청은 CloudWatch 차원이 `ObjectGroupName=baseballGroup`입니다.
- `football/saturday` 폴더에 대한 요청은 차원이 `ObjectGroupName=footballGroup`입니다.

```
{
```

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

지표 정책 예제: 컨테이너 수준 및 경로 수준 지표

이 정책 예제는 AWS Elemental MediaStore가 컨테이너 수준에서 지표를 Amazon CloudWatch에 전송해야 함을 나타냅니다. 또한 MediaStore에서는 `baseball/saturday` 및 `football/saturday`의 두 폴더의 객체에 대한 지표를 전송해야 합니다. MediaStore 요청에 대한 지표는 다음과 같습니다.

- `baseball/saturday` 폴더에 대한 요청은 CloudWatch 차원이 `ObjectGroupName=baseballGroup`입니다.
- `football/saturday` 폴더에 대한 요청은 CloudWatch 차원이 `ObjectGroupName=footballGroup`입니다.

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

지표 정책 예제: 와일드카드를 사용하는 경로 수준 지표

이 정책 예제는 AWS Elemental MediaStore가 컨테이너 수준에서 지표를 Amazon CloudWatch에 전송해야 함을 나타냅니다. 또한 MediaStore는 해당 파일 이름을 기준으로 객체에 대한 지표를 전송해야 합니다. 와일드카드는 객체가 컨테이너의 아무 곳에나 저장될 수 있으며 확장명이 `.m3u8`로 끝나는 한 모든 파일 이름을 가질 수 있음을 나타냅니다.

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

지표 정책 예제: 중복 규칙이 있는 경로 레벨 지표

이 정책 예제는 AWS Elemental MediaStore가 컨테이너 수준에서 지표를 Amazon CloudWatch에 전송해야 함을 나타냅니다. 또한 MediaStore에서는 `sports/football/saturday` 및 `sports/football`의 두 폴더에 대한 지표를 전송해야 합니다.

`sports/football/saturday` 폴더에 대한 MediaStore 요청의 지표는 CloudWatch 차원이 `ObjectGroupName=footballGroup1`입니다. `sports/football` 폴더에 저장된 객체는 두 규칙과 일치하므로, CloudWatch는 이러한 객체에 대한 두 개의 데이터 포인트를 표시합니다. 하나는 차원이 `ObjectGroupName=footballGroup1`이고 다른 하나는 차원이 `ObjectGroupName=footballGroup2`입니다.

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

```
}
```

AWS Elemental MediaStore의 폴더

폴더는 컨테이너를 분할합니다. 파일 시스템에서 폴더에 하위 폴더를 만들어 폴더를 나누듯이 폴더를 사용하여 컨테이너를 나눕니다. 최대 10개 수준의 폴더(해당 컨테이너 자체는 포함되지 않음)를 생성할 수 있습니다.

폴더는 선택 사항입니다. 객체를 폴더 대신 컨테이너에 직접 업로드하도록 선택할 수 있습니다. 하지만 폴더는 객체를 정리하기 위한 간편한 방법입니다.

폴더에 객체를 업로드하려면 폴더 경로를 지정합니다. 폴더가 이미 있을 경우 AWS Elemental MediaStore는 폴더에 객체를 저장합니다. 폴더가 없으면 폴더를 만든 후 그 폴더에 객체를 저장합니다.

예를 들어 `movies`라는 이름이 지정된 컨테이너가 있고 경로 `mlaw.ts`가 포함된 `premium/canada`라는 이름의 파일을 업로드한다고 가정해 보겠습니다. AWS Elemental MediaStore는 프리미엄 폴더의 캐나다 하위 폴더에 객체를 저장합니다. 폴더가 없으면 서비스는 `premium` 폴더와 `canada` 하위 폴더를 생성한 후 `canada` 하위 폴더에 객체를 저장합니다. 경로 없이 `movies` 컨테이너만 지정한 경우, 서비스는 객체는 컨테이너에 직접 저장합니다.

해당 폴더에서 마지막 객체를 삭제하면 AWS Elemental MediaStore는 폴더를 자동으로 삭제합니다. 또한 서비스는 해당 폴더 위의 빈 폴더를 삭제합니다. 예를 들어 premium이라는 폴더가 있고 이 폴더에는 파일이 없으며 `canada`라는 하위 폴더만 한 개 있다고 가정해 보겠습니다. `canada` 하위 폴더에는 `mlaw.ts`라는 파일이 한 개 있습니다. `mlaw.ts` 파일을 삭제하면 서비스는 `premium` 폴더와 `canada` 폴더를 모두 삭제합니다. 이 자동 삭제는 폴더에만 적용됩니다. 서비스는 빈 컨테이너를 삭제하지 않습니다.

주제

폴더 이름에 대한 규칙

폴더의 이름을 선택할 때 다음에 유의하세요.

- 이름에는 대문자(A~Z), 소문자(a~z), 숫자(0~9), 마침표(.), 하이픈(-), 물결표(~), 밑줄(_), 콜론(:)과 같은 문자만 포함할 수 있습니다.

- 이름은 1자 이상 있어야 합니다. 빈 폴더 이름(예:`folder1//folder3/`)은 허용되지 않습니다.
- 이름은 대/소문자를 구분합니다. 예를 들어 `myFolder`라는 폴더와, `myfolder`라는 폴더를 동일한 컨테이너나 폴더에 둘 수 있습니다. 해당 이름이 고유하기 때문입니다.
- 이름은 상위 컨테이너 또는 폴더에서만 고유하면 됩니다. 예를 들어 `myfolder`라는 폴더를 2개의 서로 다른 컨테이너인 `movies/myfolder`와 `sports/myfolder`에 만들 수 있습니다.
- 이름은 상위 컨테이너의 이름과 같을 수 있습니다.
- 폴더를 만든 후 이름을 변경할 수 없습니다.

폴더 생성

객체를 업로드할 때 폴더를 만들 수 있습니다. 폴더에 객체를 업로드하려면 폴더 경로를 지정합니다. 폴더가 이미 있을 경우 AWS Elemental MediaStore는 폴더에 객체를 저장합니다. 폴더가 없으면 폴더를 만든 후 그 폴더에 객체를 저장합니다.

자세한 내용은 the section called “객체 업로드” 섹션을 참조하세요.

폴더 삭제

폴더가 빈 경우에만 폴더를 삭제할 수 있으며 객체가 들어 있는 폴더는 삭제할 수 없습니다.

해당 폴더에서 마지막 객체를 삭제하면 AWS Elemental MediaStore는 폴더를 자동으로 삭제합니다. 또한 서비스는 해당 폴더 위의 빈 폴더를 삭제합니다. 예를 들어 `premium`이라는 폴더가 있고 이 폴더에는 파일이 없으며 `canada`라는 하위 폴더만 한 개 있다고 가정해 보겠습니다. `canada` 하위 폴더에는 `mlaw.ts`라는 파일이 한 개 있습니다. `mlaw.ts` 파일을 삭제하면 서비스는 `premium` 폴더와 `canada` 폴더를 모두 삭제합니다. 이 자동 삭제는 폴더에만 적용됩니다. 서비스는 빈 컨테이너를 삭제하지 않습니다.

자세한 내용은 객체 삭제 섹션을 참조하세요.

AWS Elemental MediaStore의 객체

AWS Elemental MediaStore 자산을 객체라고 합니다. 컨테이너 또는 컨테이너의 폴더에 객체를 업로드할 수 있습니다.

MediaStore에서 객체를 업로드 및 다운로드하고 삭제할 수 있습니다.

- 업로드 – 컨테이너나 폴더에 객체를 추가합니다. 객체를 만드는 것과는 다릅니다. 객체를 MediaStore에 업로드하려면 먼저 로컬에 객체를 만들어야 합니다.
- 다운로드 – MediaStore의 객체를 다른 위치로 복사합니다. MediaStore에서 객체가 제거되지는 않습니다.
- 삭제 – MediaStore에서 객체를 완전히 제거합니다. 객체를 개별적으로 삭제할 수도 있고 객체 수명 주기 정책을 추가하여 지정된 지속 시간 이후 자동으로 컨테이너의 객체를 삭제할 수도 있습니다.

MediaStore는 모든 파일 유형을 허용합니다.

주제

객체 업로드

객체를 컨테이너 또는 컨테이너의 폴더로 업로드할 수 있습니다. 폴더에 객체를 업로드하려면 폴더 경로를 지정합니다. 폴더가 이미 있을 경우 AWS Elemental MediaStore는 폴더에 객체를 저장합니다. 폴더가 없으면 폴더를 만든 후 그 폴더에 객체를 저장합니다. 폴더에 대한 자세한 내용은 AWS Elemental MediaStore의 폴더 섹션을 참조하세요.

MediaStore 콘솔이나 AWS CLI를 사용하여 객체를 업로드할 수 있습니다.

MediaStore는 객체의 청크 분할 전송을 지원합니다. 이 기능은 객체를 업로드하는 동안에도 다운로드가 가능하게 만들어 지연 시간을 단축합니다. 이 기능을 사용하려면 객체의 업로드 가용성을 `streaming`으로 설정합니다. API를 사용하여 객체를 업로드할 때 이 헤더의 값을 설정할 수 있습니다.

요청에서 이 헤더를 지정하지 않으면 MediaStore가 객체 업로드 가용성으로 기본값 `standard`을 지정합니다.

표준 업로드 가용성에서는 객체 크기가 25MB를 초과할 수 없고 스트리밍 업로드 가용성은 10MB입니다.

> **Note**
>
> 객체 파일 이름에는 문자, 숫자, 마침표(.), 밑줄(_), 물결 기호(~), 하이픈(-), 등호(=), 콜론(:)만 사용할 수 있습니다.

객체를 업로드하려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서 컨테이너 이름을 선택합니다. 컨테이너의 세부 정보 창이 나타납니다.
3. 객체 업로드를 선택합니다.
4. 대상 경로에 폴더 경로를 입력합니다. 예: `premium/canada` 지정한 경로에 폴더가 존재하지 않으면 해당 폴더가 자동으로 생성됩니다.
5. 객체 섹션에서 찾아보기를 선택합니다.
6. 해당 폴더로 이동한 후 업로드할 객체를 하나 선택합니다.
7. 열기를 선택한 후 업로드를 선택합니다.

> **Note**
>
> 이름이 같은 파일이 해당 폴더에 이미 있으면, 업로드된 파일이 원래 파일을 겹쳐 씁니다.

객체를 업로드하려면(AWS CLI)

- AWS CLI에서 `put-object` 명령을 사용합니다. 다음과 같은 파라미터를 포함시킬 수 있습니다. `content-type`, `cache-control`(호출자가 객체의 캐시 동작을 제어하도록 허용), `path`(컨테이너 내에 있는 폴더의 객체 저장).

> **Note**
>
> 객체를 업로드한 후에는 content-type, cache-control 또는 path를 변경할 수 없습니다.

```
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
```

다음 예제는 반환 값을 보여줍니다.

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

객체 목록 보기

AWS Elemental MediaStore 콘솔을 사용하여 컨테이너나 폴더의 최상위에 저장된 항목(객체 및 폴더)을 볼 수 있습니다. 현재 컨테이너나 폴더의 하위 폴더에 저장된 항목은 표시되지 않습니다. AWS CLI를 사용하여 컨테이너의 객체 및 폴더 목록을 볼 수 있습니다. 컨테이너에 있는 폴더나 하위 폴더의 개수에는 제한이 없습니다.

특정 컨테이너에 있는 객체의 목록을 보려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서, 보려는 폴더가 들어 있는 컨테이너의 이름을 선택합니다.
3. 목록에서 폴더의 이름을 선택합니다.

   세부 정보 페이지가 나타나고, 해당 폴더에 저장된 모든 폴더와 객체가 표시됩니다.

특정 폴더에 있는 객체의 목록을 보려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서, 보려는 폴더가 들어 있는 컨테이너의 이름을 선택합니다.

   세부 정보 페이지가 나타나고, 해당 컨테이너에 저장된 모든 폴더와 객체가 표시됩니다.

특정 컨테이너에 있는 객체 및 폴더의 목록을 보려면(AWS CLI)

- AWS CLI에서 `list-items` 명령을 사용합니다.

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

다음 예제는 반환 값을 보여줍니다.

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

> **Note**
>
> `seconds_since_create` 규칙이 적용되는 객체는 `list-items` 응답에 포함되지 않습니다.

특정 폴더에 있는 객체 및 폴더의 목록을 보려면(AWS CLI)

- AWS CLI에서 `list-items` 명령을 사용합니다. 요청 끝에 폴더 이름을 지정합니다.

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

다음 예제는 반환 값을 보여줍니다.

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

> **Note**
>
> `seconds_since_create` 규칙이 적용되는 객체는 `list-items` 응답에 포함되지 않습니다.

객체 세부 정보 보기

객체를 업로드하면 AWS Elemental MediaStore가 세부 정보(수정일, 콘텐츠 길이, ETag(엔터티 태그), 콘텐츠 유형 등)를 저장합니다. 객체의 메타데이터가 사용되는 방법에 대한 자세한 내용은 MediaStore의 HTTP 캐시와의 상호 작용 단원을 참조하십시오.

객체의 세부 정보를 보려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서, 보려는 객체가 들어 있는 컨테이너의 이름을 선택합니다.
3. 보려는 객체가 폴더 내에 있으면 해당 객체가 보일 때까지 폴더 이름을 계속 선택합니다.
4. 객체 이름을 선택합니다.

   세부 정보 창이 나타나고 객체에 대한 정보가 표시됩니다.

객체의 세부 정보를 보려면(AWS CLI)

- AWS CLI에서 `describe-object` 명령을 사용합니다.

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

다음 예제는 반환 값을 보여줍니다.

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

객체 다운로드

콘솔을 사용하여 객체를 다운로드할 수 있습니다. AWS CLI를 사용하여 객체 또는 객체의 일부를 다운로드할 수 있습니다.

객체를 다운로드하려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서, 다운로드하려는 객체가 들어 있는 컨테이너 이름을 선택합니다.
3. 다운로드하려는 객체가 폴더 내에 있으면 해당 객체가 보일 때까지 폴더 이름을 계속 선택합니다.
4. 객체 이름을 선택합니다.

5. 객체 세부 정보 페이지에서 다운로드를 선택합니다.

객체를 다운로드하려면(AWS CLI)

- AWS CLI에서 `get-object` 명령을 사용합니다.

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

다음 예제는 반환 값을 보여줍니다.

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

객체의 일부를 다운로드하려면(AWS CLI)

- AWS CLI에서 `get-object` 명령을 사용하고 범위를 지정합니다.

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

다음 예제는 반환 값을 보여줍니다.

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

객체 삭제

AWS Elemental MediaStore는 컨테이너에서 객체를 삭제하는 다양한 옵션을 제공합니다.

- 개별 객체를 삭제합니다. 요금이 부과되지 않습니다.
- 컨테이너 내의 모든 객체를 한 번에 삭제하도록 컨테이너를 비웁니다. 이 프로세스는 API 호출을 사용하므로 일반 API 요금이 적용됩니다.
- 특정 수명에 도달할 때 객체를 삭제하도록 객체 수명 주기 정책을 추가합니다. 요금이 부과되지 않습니다.

객체 삭제

콘솔 또는 AWS CLI를 사용하여 객체를 개별적으로 삭제할 수 있습니다. 또는 컨테이너에서 특정 수명에 도달하면 객체를 자동으로 삭제하도록 객체 수명 주기 정책을 추가하거나 해당 컨테이너 내의 모든 객체를 삭제하도록 컨테이너를 비우기할 수 있습니다.

> **Note**
>
> 폴더에 있는 유일한 객체를 삭제하면 AWS Elemental MediaStore에서 해당 폴더와 그 상위의 빈 폴더를 자동으로 삭제합니다. 예를 들어 `premium`이라는 폴더가 있고 이 폴더에는 파일이 없으며 `canada`라는 하위 폴더만 한 개 있다고 가정해 보겠습니다. `canada` 하위 폴더에는 `mlaw.ts`라는 파일이 한 개 있습니다. `mlaw.ts` 파일을 삭제하면 서비스는 `premium` 폴더와 `canada` 폴더를 모두 삭제합니다.

객체를 삭제하려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서 삭제하려는 객체가 들어 있는 컨테이너 이름을 선택합니다.
3. 삭제하려는 객체가 폴더 내에 있으면 해당 객체가 보일 때까지 폴더 이름을 계속 선택합니다.
4. 객체 이름 왼쪽에 있는 옵션을 선택합니다.
5. 삭제를 선택합니다.

객체를 삭제하려면(AWS CLI)

- AWS CLI에서 `delete-object` 명령을 사용합니다.

예제:

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

이 명령은 반환 값이 없습니다.

컨테이너 비우기

컨테이너를 비워 컨테이너 내에 저장된 모든 객체를 삭제할 수 있습니다. 또는 특정 기간이 지난 객체를 컨테이너에서 자동으로 삭제하는 객체 수명 주기 정책을 추가하거나, 객체를 개별적으로 삭제할 수 있습니다.

컨테이너를 비우려면(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서 비우려는 컨테이너에 대한 옵션을 선택합니다.
3. 컨테이너 비우기를 선택합니다. 확인 메시지가 표시됩니다.
4. 텍스트 필드에 컨테이너 이름을 입력하여 컨테이너를 비우는지 확인한 다음 비움을 선택합니다.

AWS Elemental의 보안 MediaStore

클라우드 AWS 보안이 최우선 과제입니다. AWS 고객은 가장 보안에 민감한 조직의 요구 사항을 충족하도록 구축된 데이터 센터 및 네트워크 아키텍처의 혜택을 누릴 수 있습니다.

보안은 기업과 기업 간의 공동 책임입니다. AWS 공동 책임 모델은 이 사항을 클라우드의 보안 및 클라우드 내 보안으로 설명합니다.

- 클라우드 보안 - AWS 클라우드에서 AWS 서비스를 실행하는 인프라를 보호하는 역할을 합니다 AWS 클라우드. AWS 또한 안전하게 사용할 수 있는 서비스를 제공합니다. AWS MediaStore Elemental에 적용되는 규정 준수 프로그램에 대해 자세히 알아보려면 규정 준수 프로그램별 범위 내 AWS 서비스 규정 준수 참조하십시오.
- 클라우드에서의 보안 — 사용하는 AWS 서비스에 따라 책임이 결정됩니다. 또한 귀하는 귀사의 데이터의 민감도, 귀사의 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는 공동 책임 모델을 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 MediaStore 됩니다. 다음 항목에서는 보안 및 규정 준수 목표를 MediaStore 충족하도록 구성하는 방법을 보여줍니다. 또한 MediaStore 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법도 알아봅니다.

주제

- AWS Elemental에서의 데이터 보호 MediaStore
- AWS Elemental용 자격 증명 및 액세스 관리 MediaStore
- 로그인 및 모니터링 AWS Elemental MediaStore
- AWS Elemental에 대한 규정 준수 검증 MediaStore
- AWS Elemental의 레질리언스 MediaStore
- AWS Elemental의 인프라 보안 MediaStore
- 교차 서비스 혼동된 대리인 방지

AWS Elemental에서의 데이터 보호 MediaStore

AWS 공동 책임 모델 AWS MediaStore Elemental의 데이터 보호에 적용됩니다. 이 모델에 설명된 대로, AWS 는 모든 모델을 실행하는 글로벌 인프라를 보호할 책임이 있습니다. AWS 클라우드사용자는 인프라에서 호스팅되는 콘텐츠를 관리해야 합니다. 사용하는 AWS 서비스 의 보안 구성과 관리 작

업에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 대한 자세한 내용은 데이터 프라이버시 FAQ를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은AWS 보안 블로그에서 AWS 공동 책임 모델 및 GDPR 블로그 게시물을 참조하세요.

데이터 보호를 위해 AWS 계정 자격 증명을 보호하고 AWS IAM Identity Center OR AWS Identity and Access Management (IAM) 을 사용하여 개별 사용자를 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 멀티 팩터 인증 설정(MFA)을 사용하세요.
- SSL/TLS를 사용하여 리소스와 통신하세요. AWS TLS 1.2는 필수이며 TLS 1.3를 권장합니다.
- 를 사용하여 API 및 사용자 활동 로깅을 설정합니다. AWS CloudTrail
- 포함된 모든 기본 보안 제어와 함께 AWS 암호화 솔루션을 사용하십시오 AWS 서비스.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용하세요.
- 명령줄 인터페이스 또는 API를 AWS 통해 액세스할 때 FIPS 140-2로 검증된 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용하십시오. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 FIPS(Federal Information Processing Standard) 140-2를 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 양식 필드에 입력하지 않는 것이 좋습니다. 여기에는 콘솔, API MediaStore 또는 AWS 서비스 SDK를 사용하거나 다른 방법으로 작업하는 경우가 포함됩니다. AWS CLI AWS 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 자격 보안 인증를 URL에 포함시켜서는 안 됩니다.

데이터 암호화

MediaStore 업계 표준 AES-256 알고리즘을 사용하여 저장되어 있는 컨테이너와 객체를 암호화합니다. 다음과 같은 방법으로 데이터를 보호하는 MediaStore 데 사용하는 것이 좋습니다.

- 컨테이너 정책을 생성하여 해당 컨테이너의 모든 폴더와 개체에 대한 액세스 권한을 제어합니다. 자세한 정보는 the section called “컨테이너 정책”을 참조하세요.
- 크로스 오리진 리소스 공유 (CORS) 정책을 만들어 리소스에 대한 크로스 오리진 액세스를 선택적으로 허용하세요. MediaStore CORS를 사용하면 한 도메인에서 로드된 클라이언트 웹 애플리케이션이 다른 도메인에 있는 리소스와 상호 작용하도록 허용할 수 있습니다. 자세한 정보는 the section called “CORS 정책”을 참조하세요.

AWS Elemental용 자격 증명 및 액세스 관리 MediaStore

AWS Identity and Access Management (IAM) 은 관리자가 리소스에 대한 액세스를 안전하게 제어할 수 AWS 서비스 있도록 AWS 도와줍니다. IAM 관리자는 리소스를 사용할 수 있는 인증 (로그인) 및 권한 부여 (권한 보유) 를 받을 수 있는 사용자를 제어합니다. MediaStore IAM은 추가 AWS 서비스 비용 없이 사용할 수 있습니다.

주제

고객

사용하는 방식 AWS Identity and Access Management (IAM) 은 수행하는 작업에 따라 다릅니다. MediaStore

서비스 사용자 - MediaStore 서비스를 사용하여 작업을 수행하는 경우 관리자가 필요한 자격 증명과 권한을 제공합니다. 더 많은 MediaStore 기능을 사용하여 작업을 수행함에 따라 추가 권한이 필요할 수 있습니다. 액세스 권한 관리 방식을 이해하면 적절한 권한을 관리자에게 요청할 수 있습니다. 에서 MediaStore 기능에 액세스할 수 없는 경우 을 참조하십시오AWS Elemental MediaStore 자격 증명 및 액세스 문제 해결.

서비스 관리자 — 회사에서 MediaStore 리소스를 담당하고 있다면 전체 액세스 권한이 있을 것입니다 MediaStore. 서비스 사용자가 액세스해야 하는 MediaStore 기능과 리소스를 결정하는 것은 여러분의 몫입니다. 그런 다음, IAM 관리자에게 요청을 제출하여 서비스 사용자의 권한을 변경해야 합니다. 이 페이지의 정보를 검토하여 IAM의 기본 개념을 이해합니다. 회사에서 IAM을 어떻게 사용할 수 있는지 자세히 MediaStore 알아보려면 을 참조하십시오AWS Elemental이 IAM과 MediaStore 함께 작동하는 방식.

IAM 관리자 — IAM 관리자라면 액세스 관리를 위한 정책을 작성하는 방법에 대해 자세히 알고 싶을 것입니다. MediaStore IAM에서 사용할 수 있는 MediaStore ID 기반 정책의 예를 보려면 을 참조하십시오. AWS Elemental의 자격 증명 기반 정책 예제 MediaStore

자격 증명을 통한 인증

인증은 자격 증명 자격 증명을 AWS 사용하여 로그인하는 방법입니다. IAM 사용자로 인증 (로그인 AWS) 하거나 IAM 역할을 맡아 인증 (로그인) 해야 합니다. AWS 계정 루트 사용자

ID 소스를 통해 제공된 자격 증명을 사용하여 페더레이션 ID로 로그인할 수 있습니다. AWS AWS IAM Identity Center (IAM ID 센터) 사용자, 회사의 싱글 사인온 인증, Google 또는 Facebook 자격 증명이 페더레이션 ID의 예입니다. 페더레이션형 ID로 로그인할 때 관리자가 이전에 IAM 역할을 사용하여 ID 페더레이션을 설정했습니다. 페더레이션을 사용하여 액세스하는 경우 AWS 간접적으로 역할을 맡게 됩니다.

사용자 유형에 따라 AWS Management Console 또는 AWS 액세스 포털에 로그인할 수 있습니다. 로그인에 대한 자세한 내용은 AWS 로그인 사용 설명서의 내 로그인 방법을 참조하십시오. AWS AWS 계정

AWS 프로그래밍 방식으로 액세스하는 경우 자격 증명을 사용하여 요청에 암호화 방식으로 서명할 수 있는 소프트웨어 개발 키트 (SDK) 와 명령줄 인터페이스 (CLI) 를 AWS 제공합니다. AWS 도구를 사용하지 않는 경우 요청에 직접 서명해야 합니다. 권장 방법을 사용하여 직접 요청에 서명하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 AWS API 요청 서명을 참조하십시오.

사용하는 인증 방법에 상관없이 추가 보안 정보를 제공해야 할 수도 있습니다. 예를 들어, AWS 계정의 보안을 강화하기 위해 다단계 인증 (MFA) 을 사용할 것을 권장합니다. 자세한 내용은 AWS IAM Identity Center 사용 설명서의 다중 인증 및 IAM 사용자 설명서의 AWS에서 다중 인증(MFA) 사용을 참조합니다.

AWS 계정 루트 사용자

계정을 AWS 계정만들 때는 먼저 계정의 모든 AWS 서비스 리소스에 대한 완전한 액세스 권한을 가진 하나의 로그인 ID로 시작합니다. 이 ID를 AWS 계정 루트 사용자라고 하며, 계정을 만들 때 사용한 이메일 주소와 비밀번호로 로그인하여 액세스할 수 있습니다. 일상적인 작업에 루트 사용자를 사용하지 않을 것을 강력히 권장합니다. 루트 사용자 보안 인증 정보를 보호하고 루트 사용자만 수행할 수 있는 태스크를 수행하는 데 사용하세요. 루트 사용자로 로그인해야 하는 태스크의 전체 목록은 IAM 사용자 안내서의 루트 사용자 보안 인증이 필요한 태스크를 참조하세요.

페더레이션 자격 증명

가장 좋은 방법은 관리자 액세스가 필요한 사용자를 비롯한 수동 AWS 서비스 사용자가 ID 공급자와의 페더레이션을 사용하여 임시 자격 증명을 사용하여 액세스하도록 하는 것입니다.

페더레이션 ID는 기업 사용자 디렉토리, 웹 ID 공급자, Identity Center 디렉터리의 사용자 또는 ID 소스를 통해 제공된 자격 증명을 사용하여 액세스하는 AWS 서비스 모든 사용자를 말합니다. AWS

Directory Service페더레이션 ID에 AWS 계정액세스하면 이들이 역할을 맡고 역할은 임시 자격 증명을 제공합니다.

중앙 집중식 액세스 관리를 위해 AWS IAM Identity Center을 사용하는 것이 좋습니다. IAM Identity Center에서 사용자 및 그룹을 생성하거나 자체 ID 소스의 사용자 및 그룹 집합에 연결하고 동기화하여 모든 사용자 및 애플리케이션에서 사용할 수 있습니다. AWS 계정 IAM Identity Center에 대한 자세한 내용은AWS IAM Identity Center 사용 설명서에서 IAM Identity Center란 무엇입니까?를 참조하세요.

IAM 사용자 및 그룹

IAM 사용자는 단일 사용자 또는 애플리케이션에 대한 특정 권한을 AWS 계정 가진 사용자 내 자격 증명입니다. 가능하면 암호 및 액세스 키와 같은 장기 자격 증명이 있는 IAM 사용자를 생성하는 대신 임시 자격 증명을 사용하는 것이 좋습니다. 하지만 IAM 사용자의 장기 자격 증명이 필요한 특정 사용 사례가 있는 경우 액세스 키를 교체하는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 장기 보안 인증이 필요한 사용 사례의 경우 정기적으로 액세스 키 교체를 참조하세요.

IAM 그룹은 IAM 사용자 컬렉션을 지정하는 자격 증명입니다. 사용자는 그룹으로 로그인할 수 없습니다. 그룹을 사용하여 여러 사용자의 권한을 한 번에 지정할 수 있습니다. 그룹을 사용하면 대규모 사용자 집합의 권한을 더 쉽게 관리할 수 있습니다. 예를 들어, IAMAdmins라는 그룹이 있고 이 그룹에 IAM 리소스를 관리할 권한을 부여할 수 있습니다.

사용자는 역할과 다릅니다. 사용자는 한 사람 또는 애플리케이션과 고유하게 연결되지만, 역할은 해당 역할이 필요한 사람이라면 누구나 수임할 수 있습니다. 사용자는 영구적인 장기 자격 증명을 가지고 있지만, 역할은 임시 보안 인증만 제공합니다. 자세한 정보는 IAM 사용 설명서의 IAM 사용자를 만들어야 하는 경우(역할이 아님)를 참조하세요.

IAM 역할

IAM 역할은 특정 권한을 가진 사용자 AWS 계정 내의 자격 증명입니다. IAM 사용자와 유사하지만, 특정 개인과 연결되지 않습니다. 역할을 AWS Management Console 전환하여 에서 일시적으로 IAM 역할을 맡을 수 있습니다. AWS CLI 또는 AWS API 작업을 호출하거나 사용자 지정 URL을 사용하여 역할을 수임할 수 있습니다. 역할 사용 방법에 대한 자세한 정보는 IAM 사용 설명서의 IAM 역할 사용을 참조하세요.

임시 보안 인증 정보가 있는 IAM 역할은 다음과 같은 상황에서 유용합니다.

- 페더레이션 사용자 액세스 - 페더레이션 아이덴티티에 권한을 부여하려면 역할을 생성하고 해당 역할의 권한을 정의합니다. 연동 자격 증명이 인증되면 역할이 연결되고 역할에 정의된 권한이 부여됩니다. 페더레이션 역할에 대한 자세한 내용은 IAM 사용 설명서의 타사 자격 증명 공급자의 역할 만들기를 참조하세요. IAM Identity Center를 사용하는 경우 권한 세트를 구성합니다. 인증 후 자격 증

명이 액세스할 수 있는 항목을 제어하기 위해 IAM Identity Center는 권한 세트를 IAM의 역할과 연관짓습니다. 권한 세트에 대한 자세한 내용은AWS IAM Identity Center 사용 설명서의 권한 세트를 참조하세요.

- 임시 IAM 사용자 권한 - IAM 사용자 또는 역할은 IAM 역할을 수임하여 특정 태스크에 대한 다양한 권한을 임시로 받을 수 있습니다.
- 크로스 계정 액세스 - IAM 역할을 사용하여 다른 계정의 사용자(신뢰할 수 있는 보안 주체)가 내 계정의 리소스에 액세스하도록 허용할 수 있습니다. 역할은 계정 간 액세스를 부여하는 기본적인 방법입니다. 그러나 일부 AWS 서비스경우에는 역할을 프록시로 사용하는 대신 정책을 리소스에 직접 연결할 수 있습니다. 크로스 계정 액세스를 위한 역할과 리소스 기반 정책의 차이점을 알아보려면 IAM 사용 설명서의 IAM 역할과 리소스 기반 정책의 차이를 참조하세요.
- 서비스 간 액세스 — 일부는 다른 AWS 서비스서비스의 기능을 AWS 서비스 사용합니다. 예컨대, 어떤 서비스에서 호출을 수행하면 일반적으로 해당 서비스는 Amazon EC2에서 애플리케이션을 실행하거나 Amazon S3에 객체를 저장합니다. 서비스는 호출하는 보안 주체의 권한을 사용하거나, 서비스 역할을 사용하거나, 또는 서비스 연결 역할을 사용하여 이 작업을 수행할 수 있습니다.
  - 순방향 액세스 세션 (FAS) — IAM 사용자 또는 역할을 사용하여 작업을 수행하는 경우 보안 AWS 주체로 간주됩니다. 일부 서비스를 사용하는 경우 다른 서비스에서 다른 작업을 시작하는 작업을 수행할 수 있습니다. FAS는 전화를 거는 주체의 권한을 다운스트림 AWS 서비스서비스에 AWS 서비스 요청하기 위한 요청과 결합하여 사용합니다. FAS 요청은 다른 서비스 AWS 서비스 또는 리소스와의 상호 작용이 필요한 요청을 서비스가 수신한 경우에만 이루어집니다. 이 경우 두 작업을 모두 수행할 수 있는 권한이 있어야 합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.
  - 서비스 역할 - 서비스 역할은 서비스가 사용자를 대신하여 태스크를 수행하기 위해 맡는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 정보는 IAM 사용자 설명서의 AWS 서비스에 대한 권한을 위임할 역할 생성을 참조합니다.
  - 서비스 연결 역할 — 서비스 연결 역할은 에 연결된 서비스 역할의 한 유형입니다. AWS 서비스서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은 사용자에게 AWS 계정 표시되며 해당 서비스가 소유합니다. IAM 관리자는 서비스 링크 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.
- Amazon EC2에서 실행되는 애플리케이션 — IAM 역할을 사용하여 EC2 인스턴스에서 실행되고 API 요청을 AWS CLI 하는 애플리케이션의 임시 자격 증명을 관리할 수 있습니다. AWS 이는 EC2 인스턴스 내에 액세스 키를 저장할 때 권장되는 방법입니다. EC2 인스턴스에 AWS 역할을 할당하고 모든 애플리케이션에서 사용할 수 있게 하려면 인스턴스에 연결된 인스턴스 프로필을 생성합니다. 인스턴스 프로파일에는 역할이 포함되어 있으며 EC2 인스턴스에서 실행되는 프로그램이 임시 보안 인증을 얻을 수 있습니다. 자세한 정보는 IAM 사용 설명서의 IAM 역할을 사용하여 Amazon EC2 인스턴스에서 실행되는 애플리케이션에 권한 부여를 참조하세요.

IAM 역할을 사용할지 또는 IAM 사용자를 사용할지를 알아보려면 IAM 사용자 설명서의 IAM 역할(사용자 대신)을 생성하는 경우를 참조합니다.

정책을 사용한 액세스 관리

정책을 생성하고 이를 AWS ID 또는 리소스에 AWS 연결하여 액세스를 제어할 수 있습니다. 정책은 ID 또는 리소스와 연결될 때 AWS 해당 권한을 정의하는 객체입니다. AWS 주도자 (사용자, 루트 사용자 또는 역할 세션) 가 요청할 때 이러한 정책을 평가합니다. 정책에서 권한은 요청이 허용되거나 거부되는 지를 결정합니다. 대부분의 정책은 JSON 문서로 AWS 저장됩니다. JSON 정책 문서의 구조와 콘텐츠에 대한 자세한 정보는 IAM 사용 설명서의 JSON 정책 개요를 참조하세요.

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

기본적으로, 사용자와 역할에는 어떠한 권한도 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다. 그런 다음 관리자가 IAM 정책을 역할에 추가하고, 사용자가 역할을 수임할 수 있습니다.

IAM 정책은 작업을 수행하기 위해 사용하는 방법과 상관없이 작업에 대한 권한을 정의합니다. 예를 들어, `iam:GetRole`태스크를 허용하는 정책이 있다고 가정합니다. 해당 정책을 사용하는 사용자는 AWS Management Console, AWS CLI, 또는 AWS API에서 역할 정보를 가져올 수 있습니다.

ID 기반 정책

ID 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 자격 증명에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자와 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성을 참조하세요.

자격 증명 기반 정책은 인라인 정책 또는 관리형 정책으로 한층 더 분류할 수 있습니다. 인라인 정책은 단일 사용자, 그룹 또는 역할에 직접 포함됩니다. 관리형 정책은 내 여러 사용자, 그룹 및 역할에 연결할 수 있는 독립형 정책입니다. AWS 계정관리형 정책에는 AWS 관리형 정책과 고객 관리형 정책이 포함됩니다. 관리형 정책 또는 인라인 정책을 선택하는 방법을 알아보려면 IAM 사용 설명서의 관리형 정책과 인라인 정책의 선택을 참조하세요.

리소스 기반 정책

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 리소스 기반 정책의 예는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는

이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 보안 주체에는 계정, 사용자, 역할, 연동 사용자 등이 포함될 수 있습니다. AWS 서비스

리소스 기반 정책은 해당 서비스에 있는 인라인 정책입니다. IAM의 AWS 관리형 정책은 리소스 기반 정책에 사용할 수 없습니다.

액세스 제어 목록(ACLs)

액세스 제어 목록(ACLs)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는 지를 제어합니다. ACLs는 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

ACL을 지원하는 서비스의 예로는 아마존 S3와 아마존 VPC가 있습니다. AWS WAF ACL에 대해 자세히 알아보려면 Amazon Simple Storage Service 개발자 안내서의 액세스 제어 목록(ACL) 개요를 참조하세요.

기타 정책 타입

AWS 일반적이지 않은 추가 정책 유형을 지원합니다. 이러한 정책 타입은 더 일반적인 정책 타입에 따라 사용자에게 부여되는 최대 권한을 설정할 수 있습니다.

- 권한 경계 – 권한 경계는 보안 인증 기반 정책에 따라 IAM 엔터티(IAM 사용자 또는 역할)에 부여할 수 있는 최대 권한을 설정하는 고급 기능입니다. 개체에 대한 권한 경계를 설정할 수 있습니다. 그 결과로 얻는 권한은 엔터티의 자격 증명 기반 정책과 그 권한 경계의 교집합입니다. `Principal` 필드에서 사용자나 역할을 보안 주체로 지정하는 리소스 기반 정책은 권한 경계를 통해 제한되지 않습니다. 이러한 정책 중 하나에 포함된 명시적 거부는 허용을 재정의합니다. 권한 경계에 대한 자세한 정보는 IAM 사용자 설명서의 IAM 엔터티에 대한 권한 경계를 참조합니다.
- 서비스 제어 정책 (SCP) - SCP는 조직 또는 조직 단위 (OU) 에 대한 최대 권한을 지정하는 JSON 정책입니다. AWS Organizations AWS Organizations 사업체가 소유한 여러 AWS 계정 개를 그룹화하고 중앙에서 관리하는 서비스입니다. 조직에서 모든 기능을 활성화할 경우 서비스 제어 정책 (SCP)을 임의의 또는 모든 계정에 적용할 수 있습니다. SCP는 구성원 계정의 엔티티 (각 엔티티 포함) 에 대한 권한을 제한합니다. AWS 계정 루트 사용자조직 및 SCP에 대한 자세한 정보는AWS Organizations 사용 설명서의 SCP 작동 방식을 참조하세요.
- 세션 정책 – 세션 정책은 역할 또는 연합된 사용자에 대해 임시 세션을 프로그래밍 방식으로 생성할 때 파라미터로 전달하는 고급 정책입니다. 결과적으로 얻는 세션의 권한은 사용자 또는 역할 자격 증명 기반 정책의 교차 및 세션 정책입니다. 또한 권한을 리소스 기반 정책에서 가져올 수도 있습니다.

이러한 정책 중 하나에 포함된 명시적 거부는 허용을 재정의합니다. 자세한 정보는 IAM 사용자 설명서의 세션 정책을 참조합니다.

여러 정책 타입

여러 정책 타입이 요청에 적용되는 경우 결과 권한은 이해하기가 더 복잡합니다. 여러 정책 유형이 관련되어 있을 때 요청을 허용할지 여부를 AWS 결정하는 방법을 알아보려면 IAM 사용 설명서의 정책 평가 로직을 참조하십시오.

AWS Elemental이 IAM과 MediaStore 함께 작동하는 방식

IAM을 사용하여 액세스를 MediaStore 관리하기 전에 어떤 IAM 기능과 함께 사용할 수 있는지 알아보십시오. MediaStore

AWS Elemental과 함께 사용할 수 있는 IAM 기능 MediaStore

IAM 특성	MediaStore 지원
ID 기반 정책	예
리소스 기반 정책	예
정책 작업	예
정책 리소스	예
정책 조건 키(서비스별)	예
ACLs	아니요
ABAC(정책 내 태그)	부분
임시 보안 인증	예
보안 주체 권한	예
서비스 역할	예
서비스 연결 역할	아니요

MediaStore 및 기타 AWS 서비스가 대부분의 IAM 기능과 어떻게 작동하는지 자세히 알아보려면 IAM 사용 설명서의 IAM과 함께 작동하는AWS 서비스를 참조하십시오.

ID 기반 정책은 다음과 같습니다. MediaStore

ID 기반 정책 지원	예

자격 증명 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 자격 증명에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자와 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성을 참조하세요.

IAM 자격 증명 기반 정책을 사용하면 허용되거나 거부되는 작업와 리소스뿐 아니라 작업이 허용되거나 거부되는 조건을 지정할 수 있습니다. 자격 증명 기반 정책에서는 보안 주체가 연결된 사용자 또는 역할에 적용되므로 보안 주체를 지정할 수 없습니다. JSON 정책에서 사용하는 모든 요소에 대해 알아보려면 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하세요.

다음에 대한 ID 기반 정책 예제 MediaStore

MediaStore ID 기반 정책의 예를 보려면 을 참조하십시오. AWS Elemental의 자격 증명 기반 정책 예제 MediaStore

내 리소스 기반 정책 MediaStore

리소스 기반 정책 지원	예

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 리소스 기반 정책의 예는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 보안 주체에는 계정, 사용자, 역할, 연동 사용자 등이 포함될 수 있습니다. AWS 서비스

계정 간 액세스를 활성화하려는 경우 전체 계정이나 다른 계정의 IAM 엔터티를 리소스 기반 정책의 보안 주체로 지정할 수 있습니다. 리소스 기반 정책에 크로스 계정 보안 주체를 추가하는 것은 트러스트

관계 설정의 절반밖에 되지 않는다는 것을 유념하세요. 보안 주체와 리소스가 다른 AWS 계정경우 신뢰할 수 있는 계정의 IAM 관리자는 보안 주체 개체 (사용자 또는 역할) 에게 리소스에 액세스할 수 있는 권한도 부여해야 합니다. 개체에 자격 증명 기반 정책을 연결하여 권한을 부여합니다. 하지만 리소스 기반 정책이 동일 계정의 보안 주체에 액세스를 부여하는 경우 추가 자격 증명 기반 정책이 필요하지 않습니다. 자세한 정보는 IAM 사용 설명서의 IAM 역할과 리소스 기반 정책의 차이를 참조하세요.

> **Note**
>
> MediaStore 컨테이너에서 작업을 수행할 수 있는 보안 주체 개체 (계정, 사용자, 역할, 연동 사용자) 를 정의하는 컨테이너 정책도 지원합니다. 자세한 정보는 컨테이너 정책을 참조하세요.

에 대한 정책 조치 MediaStore

정책 작업 지원	예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 `Action`요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 태스크를 설명합니다. 정책 작업은 일반적으로 관련 AWS API 작업과 이름이 같습니다. 일치하는 API 작업이 없는 권한 전용 작업 같은 몇 가지 예외도 있습니다. 정책에서 여러 작업이 필요한 몇 가지 작업도 있습니다. 이러한 추가 작업을 일컬어 종속 작업이라고 합니다.

연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하십시오.

MediaStore 작업 목록을 보려면 서비스 권한 부여 참조의 AWS MediaStore Elemental에서 정의한 작업을 참조하십시오.

정책 조치는 조치 앞에 다음 접두사를 MediaStore 사용합니다.

```
mediastore
```

단일 문에서 여러 작업을 지정하려면 다음과 같이 쉼표로 구분합니다.

```
"Action": [
```

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

MediaStore ID 기반 정책의 예를 보려면 을 참조하십시오. AWS Elemental의 자격 증명 기반 정책 예제 MediaStore

에 대한 정책 리소스 MediaStore

정책 리소스 지원	예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지 지정할 수 있습니다.

`Resource` JSON 정책 요소는 작업이 적용되는 하나 이상의 개체를 지정합니다. 문장에는 `Resource`또는 `NotResource`요소가 반드시 추가되어야 합니다. 모범 사례에 따라 Amazon 리소스 이름(ARN)을 사용하여 리소스를 지정합니다. 리소스 수준 권한이라고 하는 특정 리소스 타입을 지원하는 작업에 대해 이 작업을 수행할 수 있습니다.

작업 나열과 같이 리소스 수준 권한을 지원하지 않는 작업의 경우, 와일드카드(*)를 사용하여 해당 문이 모든 리소스에 적용됨을 나타냅니다.

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

MediaStore 리소스 유형 및 ARN 목록을 보려면 서비스 권한 부여 참조의 AWS MediaStore Elemental에서 정의한 리소스를 참조하십시오. 각 리소스의 ARN을 지정할 수 있는 작업에 대해 알아보려면 AWS Elemental에서 정의한 작업을 참조하십시오. MediaStore

MediaStore 컨테이너 리소스의 ARN은 다음과 같습니다.

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

ARN 형식에 대한 자세한 내용은 Amazon 리소스 이름 (ARN) 및 AWS 서비스 네임스페이스를 참조하십시오.

예를 들어, 문에서 AwardsShow 컨테이너를 지정하려면 다음 ARN을 사용합니다.

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

에 대한 정책 조건 키 MediaStore

서비스별 정책 조건 키 지원	예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지 지정할 수 있습니다.

Condition 요소(또는 Condition 블록)를 사용하면 정책이 발효되는 조건을 지정할 수 있습니다. Condition 요소는 옵션입니다. 같거나 작음과 같은 조건 연산자를 사용하여 정책의 조건을 요청의 값과 일치시키는 조건식을 생성할 수 있습니다.

한 문에서 여러 Condition요소를 지정하거나 단일 Condition요소에서 여러 키를 지정하는 경우 AWS 는 논리적 AND태스크를 사용하여 평가합니다. 단일 조건 키에 여러 값을 지정하는 경우 는 논리적 OR 연산을 사용하여 조건을 AWS 평가합니다. 명문의 권한을 부여하기 전에 모든 조건을 충족해야 합니다.

조건을 지정할 때 자리 표시자 변수를 사용할 수도 있습니다. 예를 들어, IAM 사용자에게 IAM 사용자 이름으로 태그가 지정된 경우에만 리소스에 액세스할 수 있는 권한을 부여할 수 있습니다. 자세한 정보는 IAM 사용 설명서의 IAM 정책 요소: 변수 및 태그를 참조하세요.

AWS 글로벌 조건 키 및 서비스별 조건 키를 지원합니다. 모든 AWS 글로벌 조건 키를 보려면 IAM 사용 AWS 설명서의 글로벌 조건 컨텍스트 키를 참조하십시오.

MediaStore 조건 키 목록을 보려면 서비스 권한 부여 참조의 AWS MediaStore Elemental의 조건 키를 참조하십시오. 조건 키를 사용할 수 있는 작업 및 리소스를 알아보려면 AWS MediaStore Elemental에서 정의한 작업을 참조하십시오.

MediaStore 자격 증명 기반 정책의 예를 보려면 을 참조하십시오. AWS Elemental의 자격 증명 기반 정책 예제 MediaStore

내 ACL MediaStore

ACL 지원	아니요

액세스 제어 목록(ACLs)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는 지를 제어합니다. ACLs는 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

ABAC 포함 MediaStore

ABAC(정책 내 태그) 지원	부분

속성 기반 액세스 제어(ABAC)는 속성을 기반으로 권한을 정의하는 권한 부여 전략입니다. AWS에서는 이러한 속성을 태그라고 합니다. IAM 개체 (사용자 또는 역할) 및 여러 AWS 리소스에 태그를 첨부할 수 있습니다. ABAC의 첫 번째 단계로 개체 및 리소스에 태그를 지정합니다. 그런 다음 보안 주체의 태그가 액세스하려는 리소스의 태그와 일치할 때 작업을 허용하도록 ABAC 정책을 설계합니다.

ABAC는 빠르게 성장하는 환경에서 유용하며 정책 관리가 번거로운 상황에 도움이 됩니다.

태그를 기반으로 액세스를 제어하려면 `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` 또는 `aws:TagKeys` 조건 키를 사용하여 정책의 조건 요소에 태그 정보를 제공합니다.

서비스가 모든 리소스 타입에 대해 세 가지 조건 키를 모두 지원하는 경우, 값은 서비스에 대해 예입니다. 서비스가 일부 리소스 타입에 대해서만 세 가지 조건 키를 모두 지원하는 경우, 값은 부분적입니다.

ABAC에 대한 자세한 정보는 IAM 사용 설명서의 ABAC란 무엇인가요?를 참조하세요. ABAC 설정 단계가 포함된 자습서를 보려면 IAM 사용 설명서의 속성 기반 액세스 제어(ABAC) 사용을 참조하세요.

임시 자격 증명 사용: MediaStore

임시 보안 인증 지원	예

임시 자격 증명을 사용하여 로그인하면 작동하지 AWS 서비스 않는 것도 있습니다. 임시 자격 증명을 사용하는 방법을 AWS 서비스 비롯한 추가 정보는 IAM 사용 설명서의 IAM과AWS 서비스 연동되는 내용을 참조하십시오.

사용자 이름과 암호를 제외한 다른 방법을 AWS Management Console 사용하여 로그인하면 임시 자격 증명을 사용하는 것입니다. 예를 들어 회사의 SSO (Single Sign-On) 링크를 AWS 사용하여 액세스하는 경우 이 프로세스에서 자동으로 임시 자격 증명을 생성합니다. 또한 콘솔에 사용자로 로그인한 다음 역할을 전환할 때 임시 보안 인증을 자동으로 생성합니다. 역할 전환에 대한 자세한 정보는 IAM 사용 설명서의 역할로 전환(콘솔)을 참조하세요.

또는 API를 사용하여 임시 자격 증명을 수동으로 생성할 수 있습니다 AWS CLI . AWS 그런 다음 해당 임시 자격 증명을 사용하여 액세스할 수 AWS있습니다. AWS 장기 액세스 키를 사용하는 대신 임시 자격 증명을 동적으로 생성할 것을 권장합니다. 자세한 정보는 IAM의 임시 보안 인증 섹션을 참조하세요.

서비스 간 보안 주체 권한에 대한 MediaStore

전달 액세스 세션(FAS) 지원	예

IAM 사용자 또는 역할을 사용하여 작업을 수행하는 AWS경우 보안 주체로 간주됩니다. 일부 서비스를 사용하는 경우 다른 서비스에서 다른 작업을 시작하는 작업을 수행할 수 있습니다. FAS는 전화를 거는 주체의 권한을 다운스트림 서비스에 AWS 서비스 요청하라는 요청과 결합하여 사용합니다. AWS 서비스 FAS 요청은 다른 서비스 AWS 서비스 또는 리소스와의 상호 작용이 필요한 요청을 서비스가 수신한 경우에만 이루어집니다. 이 경우 두 작업을 모두 수행할 수 있는 권한이 있어야 합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.

MediaStore의 서비스 역할

서비스 역할 지원	예

서비스 역할은 서비스가 사용자를 대신하여 작업을 수행하는 것으로 가정하는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 정보는 IAM 사용자 설명서의 AWS 서비스에 대한 권한을 위임할 역할 생성을 참조합니다.

> **Warning**
>
> 서비스 역할의 권한을 변경하면 MediaStore 기능이 중단될 수 있습니다. 서비스 역할을 편집하기 위한 지침이 MediaStore 제공되는 경우에만 서비스 역할을 편집하십시오.

서비스 연결 역할은 다음과 같습니다. MediaStore

서비스 연결 역할 지원	아니요

서비스 연결 역할은 에 연결된 서비스 역할 유형입니다. AWS 서비스서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은 사용자에게 AWS 계정 표시되며 해

당 서비스가 소유합니다. IAM 관리자는 서비스 링크 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.

서비스 연결 역할 생성 또는 관리에 대한 자세한 내용은 IAM으로 작업하는AWS 서비스 단원을 참조하세요. 서비스 연결 역할 열에서 Yes가 포함된 서비스를 테이블에서 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 Yes(네) 링크를 선택합니다.

AWS Elemental의 자격 증명 기반 정책 예제 MediaStore

기본적으로 사용자와 역할에는 리소스를 생성하거나 수정할 권한이 없습니다. MediaStore 또한 AWS Management Console, AWS Command Line Interface (AWS CLI) 또는 AWS API를 사용하여 작업을 수행할 수 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다. 그런 다음 관리자가 IAM 정책을 역할에 추가하고, 사용자가 역할을 맡을 수 있습니다.

이러한 예제 JSON 정책 문서를 사용하여 IAM ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성을 참조하세요.

각 리소스 유형의 ARN 형식을 비롯하여 에서 정의한 MediaStore 작업 및 리소스 유형에 대한 자세한 내용은 서비스 권한 부여 참조의 AWS MediaStore Elemental의 작업, 리소스 및 조건 키를 참조하십시오.

주제

- 정책 모범 사례
- MediaStore 콘솔 사용
- 사용자가 자신의 고유한 권한을 볼 수 있도록 허용

정책 모범 사례

ID 기반 정책은 누군가가 계정에서 MediaStore 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부를 결정합니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. ID 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따르세요.

- AWS 관리형 정책으로 시작하여 최소 권한 권한으로 이동 — 사용자와 워크로드에 권한을 부여하려면 여러 일반적인 사용 사례에 권한을 부여하는 AWS 관리형 정책을 사용하세요. 해당 내용은 에서 사용할 수 있습니다. AWS 계정사용 사례에 맞는 AWS 고객 관리형 정책을 정의하여 권한을 더 줄이는 것이 좋습니다. 자세한 정보는 IAM 사용 설명서의 AWS managed policies(관리형 정책) 또는 AWS managed policies for job functions(직무에 대한 관리형 정책)를 참조하세요.

- 최소 권한 적용 – IAM 정책을 사용하여 권한을 설정하는 경우 태스크를 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 최소 권한으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용 설명서에 있는 Policies and permissions in IAM(IAM의 정책 및 권한)을 참조하세요.
- IAM 정책의 조건을 사용하여 액세스 추가 제한 – 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어 SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정책 조건을 작성할 수 있습니다. 예를 AWS 서비스들어 특정 작업을 통해 서비스 작업을 사용하는 경우 조건을 사용하여 서비스 작업에 대한 액세스 권한을 부여할 수도 AWS CloudFormation있습니다. 자세한 정보는 IAM 사용 설명서의 IAM JSON 정책 요소: 조건을 참조하세요.
- IAM Access Analyzer를 통해 IAM 정책을 검증하여 안전하고 기능적인 권한 보장 – IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 모범 사례가 정책에서 준수되도록 신규 및 기존 정책을 검증합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 추천을 제공하여 안전하고 기능적인 정책을 작성하도록 돕습니다. 자세한 정보는 IAM 사용 설명서의 IAM Access Analyzer 정책 검증을 참조하tpdy.
- 멀티 팩터 인증 (MFA) 필요 - IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우 추가 보안을 위해 AWS 계정 MFA를 활성화하십시오. API 작업을 직접 호출할 때 MFA가 필요하면 정책에 MFA 조건을 추가합니다. 자세한 정보는 IAM 사용 설명서의 Configuring MFA-protected API access(MFA 보호 API 액세스 구성)를 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 IAM의 보안 모범 사례를 참조하세요.

MediaStore 콘솔 사용

AWS Elemental MediaStore 콘솔에 액세스하려면 최소 권한 세트가 있어야 합니다. 이러한 권한을 통해 내 MediaStore 리소스의 세부 정보를 나열하고 볼 수 AWS 계정있어야 합니다. 최소 필수 권한보다 더 제한적인 자격 증명 기반 정책을 만들면 콘솔이 해당 정책에 연결된 엔터티(사용자 또는 역할)에 대해 의도대로 작동하지 않습니다.

AWS CLI 또는 AWS API만 호출하는 사용자에게 최소 콘솔 권한을 허용할 필요는 없습니다. 그 대신, 수행하려는 API 작업과 일치하는 작업에만 액세스할 수 있도록 합니다.

사용자와 역할이 MediaStore 콘솔을 계속 사용할 수 있도록 하려면 엔티티에 MediaStore `ConsoleAccess` 또는 `ReadOnly` AWS 관리형 정책도 연결하세요. 자세한 내용은 IAM 사용 설명서의 사용자에게 권한 추가를 참조하십시오.

사용자가 자신의 고유한 권한을 볼 수 있도록 허용

이 예시는 IAM 사용자가 자신의 사용자 자격 증명에 연결된 인라인 및 관리형 정책을 볼 수 있도록 허용하는 정책을 생성하는 방법을 보여줍니다. 이 정책에는 콘솔에서 AWS CLI 또는 AWS API를 사용하여 프로그래밍 방식으로 이 작업을 완료할 수 있는 권한이 포함됩니다.

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

AWS Elemental MediaStore 자격 증명 및 액세스 문제 해결

다음 정보를 사용하면 및 IAM을 사용할 때 발생할 수 있는 일반적인 문제를 MediaStore 진단하고 해결하는 데 도움이 됩니다.

주제

저는 다음과 같은 작업을 수행할 권한이 없습니다. MediaStore

작업을 수행할 권한이 없다는 오류가 수신되면, 작업을 수행할 수 있도록 정책을 업데이트해야 합니다.

다음 예제 오류는 `mateojackson`IAM 사용자가 콘솔을 사용하여 가상 *`my-example-widget`*리소스에 대한 세부 정보를 보려고 하지만 가상 `mediastore:`*`GetWidget`*권한이 없을 때 발생합니다.

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

이 경우 `mediastore:`*`GetWidget`*작업을 사용하여 *`my-example-widget`*리소스에 액세스할 수 있도록 `mateojackson`사용자 정책을 업데이트해야 합니다.

도움이 필요하면 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

저는 IAM을 수행할 권한이 없습니다. PassRole

작업을 수행할 권한이 없다는 오류가 발생하는 경우 역할을 넘길 수 있도록 정책을 업데이트해야 합니다. `iam:PassRole` MediaStore

일부 AWS 서비스 서비스에서는 새 서비스 역할 또는 서비스 연결 역할을 만드는 대신 기존 역할을 해당 서비스에 전달할 수 있습니다. 이렇게 하려면 사용자가 서비스에 역할을 전달할 수 있는 권한을 가지고 있어야 합니다.

다음 예제 오류는 이라는 IAM 사용자가 콘솔을 사용하여 작업을 `marymajor` 수행하려고 할 때 발생합니다. MediaStore 하지만 작업을 수행하려면 서비스 역할이 부여한 권한이 서비스에 있어야 합니다. Mary는 서비스에 역할을 전달할 수 있는 권한을 가지고 있지 않습니다.

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

이 경우 Mary가 `iam:PassRole`작업을 수행할 수 있도록 Mary의 정책을 업데이트해야 합니다.

도움이 필요하면 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

외부 사용자가 내 MediaStore 리소스에 액세스할 수 있도록 AWS 계정 허용하고 싶습니다.

다른 계정의 사용자 또는 조직 외부의 사람이 리소스에 액세스할 때 사용할 수 있는 역할을 생성할 수 있습니다. 역할을 수임할 신뢰할 수 있는 사람을 지정할 수 있습니다. 리소스 기반 정책 또는 액세스 제어 목록(ACL)을 지원하는 서비스의 경우 이러한 정책을 사용하여 다른 사람에게 리소스에 대한 액세스 권한을 부여할 수 있습니다.

자세히 알아보려면 다음을 참조하세요.

- 이러한 기능의 MediaStore 지원 여부를 알아보려면 을 참조하십시오AWS Elemental이 IAM과 MediaStore 함께 작동하는 방식.
- 소유한 리소스에 대한 액세스 권한을 AWS 계정 부여하는 방법을 알아보려면 IAM 사용 설명서에서 자신이 소유한 다른 AWS 계정 IAM 사용자에게 액세스 권한 제공을 참조하십시오.
- 제3자에게 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 타사 AWS 계정AWS 계정 소유에 대한 액세스 제공을 참조하십시오.
- ID 페더레이션을 통해 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 외부에서 인증된 사용자에게 액세스 권한 제공(자격 증명 연동)을 참조하세요.
- 크로스 계정 액세스를 위한 역할과 리소스 기반 정책 사용의 차이점을 알아보려면 IAM 사용 설명서의 IAM 역할과 리소스 기반 정책의 차이를 참조하세요.

로그인 및 모니터링 AWS Elemental MediaStore

이 단원에는 보안을 위한 AWS Elemental MediaStore 의 로깅 및 모니티링 옵션에 대한 개요가 나와 있습니다. 로그인 및 모니터링에 대한 자세한 내용은 MediaStore 을 참조하십시오AWS Elemental MediaStore의 모니터링 및 태깅.

모니터링은 AWS 솔루션의 신뢰성, 가용성 및 성능을 유지하는 데 AWS Elemental MediaStore 있어 중요한 부분입니다. 다중 지점 장애가 발생할 경우 이를 보다 쉽게 디버깅할 수 있도록 AWS 솔루션의 모든 부분에서 모니터링 데이터를 수집해야 합니다. AWS MediaStore 리소스를 모니터링하고 잠재적 사고에 대응하기 위한 여러 도구를 제공합니다.

아마존 CloudWatch 알람

CloudWatch 경보를 사용하면 지정한 기간 동안 단일 지표를 관찰할 수 있습니다. 지표가 지정된 임계값을 초과하는 경우 Amazon SNS 주제 또는 AWS Auto Scaling 정책에 알림이 전송됩니다. CloudWatch 경보는 특정 상태에 있기 때문에 작업을 호출하지 않습니다. 대신, 상태가 변경되어 지정된 기간 동안 유지되어야 합니다. 자세한 정보는 CloudWatch를 사용한 모니터링을 참조하세요.

AWS CloudTrail 로그

CloudTrail 에서 사용자, 역할 또는 AWS 서비스가 수행한 작업의 기록을 제공합니다 AWS Elemental MediaStore. 에서 수집한 CloudTrail 정보를 사용하여 요청을 받은 사람 MediaStore, 요청한 IP 주소, 요청한 사람, 요청 시기 및 추가 세부 정보를 확인할 수 있습니다. 자세한 정보는 CloudTrail을 사용하여 API 호출 로깅을 참조하세요.

AWS Trusted Advisor

Trusted Advisor 수십만 명의 AWS 고객에게 서비스를 제공하면서 배운 모범 사례를 기반으로 합니다. Trusted Advisor AWS 환경을 검사한 다음 비용을 절감하거나, 시스템 가용성 및 성능을 개선하거나, 보안 격차를 줄이는 데 도움이 될 기회가 있을 때 권장 사항을 제시합니다. 모든 AWS 고객은 Trusted Advisor 검사 5개에 액세스할 수 있습니다. 비즈니스 또는 엔터프라이즈 지원 플랜을 보유한 고객은 모든 Trusted Advisor 확인 사항을 볼 수 있습니다.

자세한 정보는 AWS Trusted Advisor을 참조하세요.

AWS Elemental에 대한 규정 준수 검증 MediaStore

특정 규정 준수 프로그램의 범위 내에 AWS 서비스 있는지 알아보려면AWS 서비스 규정 준수 프로그램의AWS 서비스 범위별, 규정 참조하여 관심 있는 규정 준수 프로그램을 선택하십시오. 일반 정보는 AWS 규정 준수 프로그램AWS 보증 프로그램 규정AWS 참조하십시오.

를 사용하여 AWS Artifact타사 감사 보고서를 다운로드할 수 있습니다. 자세한 내용은 의 보고서 https://docs.aws.amazon.com/artifact/latest/ug/downloading-documents.html 참조하십시오 AWS Artifact.

사용 시 규정 준수 AWS 서비스 책임은 데이터의 민감도, 회사의 규정 준수 목표, 관련 법률 및 규정에 따라 결정됩니다. AWS 규정 준수에 도움이 되는 다음 리소스를 제공합니다.

- 보안 및 규정 준수 퀵 스타트 가이드 - 이 배포 가이드에서는 아키텍처 고려 사항을 설명하고 보안 및 규정 준수에 AWS 중점을 둔 기본 환경을 배포하기 위한 단계를 제공합니다.

- Amazon Web Services의 HIPAA 보안 및 규정 준수를 위한 설계 — 이 백서에서는 기업이 HIPAA 적격 애플리케이션을 만드는 AWS 데 사용할 수 있는 방법을 설명합니다.

  > **Note**
  >
  > 모든 AWS 서비스 사람이 HIPAA 자격을 갖춘 것은 아닙니다. 자세한 내용은 HIPAA 적격 서비스 참조를 참조하십시오.

- AWS 규정 준수 리소스AWS — 이 워크북 및 가이드 모음은 해당 산업 및 지역에 적용될 수 있습니다.
- AWS 고객 규정 준수 가이드 — 규정 준수의 관점에서 공동 책임 모델을 이해하십시오. 이 가이드에서는 보안을 유지하기 위한 모범 사례를 AWS 서비스 요약하고 여러 프레임워크 (미국 표준 기술 연구소 (NIST), 결제 카드 산업 보안 표준 위원회 (PCI), 국제 표준화기구 (ISO) 등) 에서 보안 제어에 대한 지침을 매핑합니다.
- AWS Config 개발자 안내서의 규칙을 사용하여 리소스 평가 — 이 AWS Config 서비스는 리소스 구성이 내부 관행, 업계 지침 및 규정을 얼마나 잘 준수하는지 평가합니다.
- AWS Security Hub— 이를 AWS 서비스 통해 내부 AWS보안 상태를 포괄적으로 파악할 수 있습니다. Security Hub는 보안 제어를 사용하여 AWS 리소스를 평가하고 보안 업계 표준 및 모범 사례에 대한 규정 준수를 확인합니다. 지원되는 서비스 및 제어 목록은 Security Hub 제어 참조를 참조하십시오.
- Amazon GuardDuty — 환경에 의심스럽고 악의적인 활동이 있는지 AWS 계정모니터링하여 워크로드, 컨테이너 및 데이터에 대한 잠재적 위협을 AWS 서비스 탐지합니다. GuardDuty 특정 규정 준수 프레임워크에서 요구하는 침입 탐지 요구 사항을 충족하여 PCI DSS와 같은 다양한 규정 준수 요구 사항을 해결하는 데 도움이 될 수 있습니다.
- AWS Audit Manager— 이를 AWS 서비스 통해 AWS 사용량을 지속적으로 감사하여 위험을 관리하고 규정 및 업계 표준을 준수하는 방법을 단순화할 수 있습니다.

AWS Elemental의 레질리언스 MediaStore

AWS 글로벌 인프라는 가용 영역을 중심으로 AWS 리전 구축됩니다. AWS 리전 물리적으로 분리되고 격리된 여러 가용 영역을 제공합니다. 이 가용 영역은 지연 시간이 짧고 처리량이 높으며 중복성이 높은 네트워킹으로 연결됩니다. 가용 영역을 사용하면 중단 없이 영역 간에 자동으로 장애 극복 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

가용 영역에 대한 AWS 리전 자세한 내용은 글로벌 인프라를 참조하십시오AWS .

AWS 글로벌 인프라 외에도 데이터 복원력 및 백업 요구 사항을 지원하는 데 도움이 되는 여러 기능을 MediaStore 제공합니다.

AWS Elemental의 인프라 보안 MediaStore

관리형 서비스인 AWS MediaStore Elemental은 AWS 글로벌 네트워크 보안의 보호를 받습니다. AWS 보안 서비스 및 인프라 AWS 보호 방법에 대한 자세한 내용은 AWS 클라우드 보안을 참조하십시오. 인프라 보안 모범 사례를 사용하여 AWS 환경을 설계하려면 Security Pillar AWS Well-Architected Framework의 인프라 보호를 참조하십시오.

AWS 게시된 API 호출을 사용하여 네트워크를 통해 액세스할 MediaStore 수 있습니다. 고객은 다음을 지원해야 합니다.

- 전송 계층 보안(TLS) TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군 Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

또한 요청은 액세스 키 ID 및 IAM 주체와 관련된 비밀 액세스 키를 사용하여 서명해야 합니다. 또는 AWS Security Token Service(AWS STS)를 사용하여 임시 보안 인증을 생성하여 요청에 서명할 수 있습니다.

교차 서비스 혼동된 대리인 방지

혼동된 대리자 문제는 작업을 수행할 권한이 없는 엔터티가 권한이 더 많은 엔터티에게 작업을 수행하도록 강요할 수 있는 보안 문제입니다. 에서 AWS크로스 서비스 사칭으로 인해 대리인 문제가 발생할 수 있습니다. 교차 서비스 가장은 한 서비스(직접 호출하는 서비스)가 다른 서비스(직접 호출되는 서비스)를 직접 호출할 때 발생할 수 있습니다. 직접 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해 AWS 에서는 계정의 리소스에 대한 액세스 권한이 부여된 서비스 보안 주체를 사용하여 모든 서비스에 대한 데이터를 보호하는 데 도움이 되는 도구를 제공합니다.

AWS Elemental이 리소스에 다른 서비스에 MediaStore 부여하는 권한을 제한하려면 리소스 정책에서 `aws:SourceArn`및 `aws:SourceAccount`글로벌 조건 컨텍스트 키를 사용하는 것이 좋습니다. 하나의 리소스만 교차 서비스 액세스와 연결되도록 허용하려는 경우 `aws:SourceArn`을 사용하세요. 해당 계정의 모든 리소스가 교차 서비스 사용과 연결되도록 허용하려는 경우 `aws:SourceAccount`을 사용하세요.

혼동된 대리자 문제로부터 보호하는 가장 효과적인 방법은 리소스의 전체 ARN이 포함된 aws:SourceArn 전역 조건 컨텍스트 키를 사용하는 것입니다. 리소스의 전체 ARN을 모르거나 여러 리소스를 지정하는 경우, ARN의 알 수 없는 부분에 대해 와일드카드 문자(*)를 포함한 aws:SourceArn 글로벌 조건 컨텍스트 키를 사용합니다. 예를 들어 arn:aws:*servicename*:*:*123456789012*:*입니다.

만약 aws:SourceArn 값에 Amazon S3 버킷 ARN과 같은 계정 ID가 포함되어 있지 않은 경우, 권한을 제한하려면 두 글로벌 조건 컨텍스트 키를 모두 사용해야 합니다.

의 값은 지역 및 계정에 CloudWatch 로그를 MediaStore 게시하는 aws:SourceArn 구성이어야 합니다.

다음 예제는 aws:SourceArn 및 aws:SourceAccount 글로벌 조건 컨텍스트 키를 사용하여 혼동되는 부정 문제를 방지하는 MediaStore 방법을 보여줍니다.

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

AWS Elemental MediaStore의 모니터링 및 태깅

AWS Elemental MediaStore 및 다른 AWS 솔루션의 안정성, 가용성 및 성능을 유지하려면 모니터링이 중요합니다. AWS에서 MediaStore를 모니터링하고, 이상이 있을 때 이를 보고하고, 필요한 경우 자동 조치를 취할 수 있도록 다음과 같은 모니터링 도구를 제공합니다.

- AWS CloudTrail은 직접 수행하거나 AWS 계정을 대신하여 수행한 API 호출 및 관련 이벤트를 캡처하고 지정한 Amazon S3 버킷에 로그 파일을 전송합니다. 어떤 사용자 및 계정이 AWS를 호출했는지, 어떤 소스 IP 주소에 호출이 이루어졌는지, 언제 호출이 발생했는지 확인할 수 있습니다. 자세한 내용은 AWS CloudTrail 사용 설명서를 참조하세요.
- Amazon CloudWatch는 AWS에서 실행하는 AWS 리소스와 애플리케이션을 실시간으로 모니터링합니다. 지표를 수집 및 추적하고, 사용자 지정 대시보드를 생성할 수 있으며, 지정된 지표가 지정한 임계값에 도달하면 사용자에게 알리거나 조치를 취하도록 경보를 설정할 수 있습니다. 예를 들어 CloudWatch에서 Amazon EC2 인스턴스의 CPU 사용량 또는 기타 지표를 추적하고 필요할 때 자동으로 새 인스턴스를 시작할 수 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하세요.
- Amazon CloudWatch Events는 AWS 리소스의 변경 사항을 설명하는 시스템 이벤트의 스트림을 제공합니다. AWS 서비스는 이벤트 알림을 일반적으로 몇 초 안에 CloudWatch Events로 전송하지만 1분 이상 소요되는 경우도 있습니다. CloudWatch Events는 특정 이벤트를 감시하는 규칙을 작성하고 이러한 이벤트가 발생할 때 다른 AWS 서비스에서 자동화된 작업을 트리거할 수 있으므로 자동화된 이벤트 기반 컴퓨팅이 가능합니다. 자세한 내용은 Amazon CloudWatch Events 사용 설명서를 참조하세요.
- Amazon CloudWatch Logs로 Amazon EC2 인스턴스, CloudTrail, 기타 소스의 로그 파일을 모니터링, 저장 및 액세스할 수 있습니다. CloudWatch Logs는 로그 파일의 정보를 모니터링하고 특정 임계값에 도달하면 사용자에게 알릴 수 있습니다. 또한 매우 내구력 있는 스토리지에 로그 데이터를 저장할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서를 참조하세요.

MediaStore 컨테이너에 태그 형태로 메타데이터를 지정할 수도 있습니다. 각 태그는 사용자가 정의하는 키와 값으로 구성되는 레이블입니다. 태그를 사용하면 리소스를 보다 쉽게 관리, 검색 및 필터링할 수 있습니다. AWS Management Console에서 AWS 리소스를 구성하는 태그를 사용할 수 있으며, 인프라 자동화 작업 중에 리소를 필터링하고 전체 AWS 리소스에 대해 결제 및 사용 보고서를 생성할 수 있습니다.

주제

- Amazon CloudWatch를 사용한 AWS Elemental MediaStore 모니터링
- AWS Elemental MediaStore 리소스에 태그 지정

AWS CloudTrail을 사용한 AWS Elemental MediaStore API 호출 로깅

AWS Elemental MediaStore는 MediaStore에서 사용자, 역할 또는 AWS 서비스가 수행한 작업의 레코드를 제공하는 서비스인 AWS CloudTrail과 통합됩니다. CloudTrail은 MediaStore 콘솔의 호출 및 MediaStore API에 대한 코드 호출을 포함하여 MediaStore에 대한 API 호출의 하위 세트를 이벤트로 캡처합니다. 추적을 생성하면 MediaStore에 대한 이벤트를 포함한 CloudTrail 이벤트를 지속적으로 Amazon S3 버킷에 배포할 수 있습니다. 추적을 구성하지 않은 경우에도 CloudTrail 콘솔의 이벤트 기록에서 최신 이벤트를 볼 수 있습니다. CloudTrail에서 수집한 정보를 사용하여 MediaStore에 수행된 요청, 요청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 등을 확인할 수 있습니다.

구성 및 사용 방법을 포함하여 CloudTrail에 대한 자세한 내용은 AWS CloudTrail 사용 설명서를 참조하세요.

주제

- CloudTrail의 AWS Elemental MediaStore 정보
- 예: AWS Elemental MediaStore 로그 파일 항목

CloudTrail의 AWS Elemental MediaStore 정보

CloudTrail은 계정 생성 시 AWS 계정에서 사용되도록 설정됩니다. 지원되는 이벤트 활동이 AWS Elemental MediaStore에서 발생하면 해당 활동이 이벤트 기록의 다른 AWS 서비스 이벤트와 함께 CloudTrail 이벤트에 기록됩니다. AWS 계정에서 최신 이벤트를 확인, 검색 및 다운로드할 수 있습니다. 자세한 내용은 CloudTrail 이벤트 기록을 사용하여 이벤트 보기를 참조하세요.

MediaStore에 대한 이벤트를 포함하여 AWS 계정에 이벤트를 지속적으로 기록하려면 추적을 생성합니다. CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 콘솔에서 추적을 생성하면 기본적으로 모든 AWS 리전에 추적이 적용됩니다. 추적은 AWS 파티션에 있는 모든 리전의 이벤트를 로깅하고 지정된 Amazon S3 버킷으로 로그 파일을 전송합니다. 또는 CloudTrail 로그에서 수집된 이벤트 데이터를 추가 분석 및 처리하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 정보는 다음 주제를 참조하세요.

- 추적 생성 개요

- CloudTrail 지원 서비스 및 통합
- CloudTrail에 대한 Amazon SNS 알림 구성
- 여러 리전에서 CloudTrail 로그 파일 받기 및 여러 계정에서 CloudTrail 로그 파일 받기

AWS Elemental MediaStore는 CloudTrail 로그 파일에 있는 이벤트로 다음 작업의 로깅을 지원합니다.

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

모든 이벤트 및 로그 항목에는 요청을 생성한 사용자에 대한 정보가 들어 있습니다. 자격 증명 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트 사용자로 했는지 사용자 보안 인증으로 했는지 여부
- 역할 또는 연합된 사용자에 대한 임시 보안 자격 증명을 사용하여 요청이 생성되었는지 여부
- 다른 AWS 서비스에서 요청했는지 여부

자세한 내용은 CloudTrail userIdentity 요소를 참조하세요.

예: AWS Elemental MediaStore 로그 파일 항목

추적이란 지정한 Amazon S3 버킷에 이벤트를 로그 파일로 입력할 수 있게 하는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함될 수 있습니다. 이벤트는 모든 소스의 단일 요청을 나타내며 요청된 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보를 포함하고 있습니다. CloudTrail 로그 파일은 퍼블릭 API 호출에 대한 순서 지정된 스택 추적이 아니기 때문에 특정 순서로 표시되지 않습니다.

다음 예제는 CreateContainer 작업을 보여주는 CloudTrail 로그 항목입니다.

```
{
    "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"
}
```

Amazon CloudWatch를 사용한 AWS Elemental MediaStore 모니터링

원시 데이터를 수집하여 읽을 수 있는 지표로 처리하는 CloudWatch를 사용하면 AWS Elemental MediaStore를 모니터링할 수 있습니다. CloudWatch는 통계를 15개월간 보관하므로 기록 정보에 액세스하고 웹 애플리케이션 또는 서비스가 어떻게 실행되고 있는지 전체적으로 더 잘 파악할 수 있습니다. 특정 임계값을 주시하다가 해당 임계값이 충족될 때 알림을 전송하거나 조치를 취하도록 경보를 설정할 수도 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하세요.

AWS에서 MediaStore를 관찰하고, 문제 발생 시 보고를 하며 적절한 경우 자동 조치를 취하는 다음과 같은 모니터링 도구를 제공합니다.

- Amazon CloudWatch Logs는 AWS Elemental MediaStore와 같은 AWS 서비스에서 로그 파일을 모니터링, 저장 및 액세스할 수 있게 해줍니다. CloudWatch Logs를 사용하여 로그 데이터로 애플리케이션 및 시스템을 모니터링할 수 있습니다. 예를 들어 CloudWatch Logs에서는 애플리케이션 로그에서 발생하는 오류의 수를 추적하고 오류 비율이 지정한 임계값을 초과할 때마다 알림을 전송할 수 있습니다. CloudWatch Logs는 모니터링하는 데 로그 데이터를 사용하므로 코드를 변경할 필요가 없습니다. 예를 들어 애플리케이션 로그에서 특정 리터럴(예: "ValidationException")을 모니터링하거나 일정 기간 동안 생성된 `PutObject` 요청을 계수할 수 있습니다. 검색할 단어가 발견되면 CloudWatch Logs는 지정한 CloudWatch 지표로 데이터를 보고합니다. 로그 데이터는 전송 시는 물론 저장 시에도 암호화됩니다.
- Amazon CloudWatch Events는 MediaStore 객체와 같이 AWS 리소스의 변경 내용을 설명하는 시스템 이벤트를 제공합니다. AWS 서비스는 이벤트 알림을 일반적으로 몇 초 안에 CloudWatch Events로 전송하지만 1분 이상 소요되는 경우도 있습니다. 규칙을 설정하면 일치하는 이벤트(예: `DeleteObject` 요청)를 검색하고 하나 이상의 대상 함수 또는 스트림으로 이벤트를 라우팅할 수 있습니다. CloudWatch Events는 운영 변경 사항이 발생할 때 이를 인식하게 됩니다. 또한 CloudWatch Events는 환경에 응답하기 위한 메시지를 전송하고 함수를 활성화하고 변경을 수행하고 상태 정보를 기록하는 등 이러한 운영 변경 사항에 응답하고 필요에 따라 교정 조치를 취합니다.

CloudWatch Logs

액세스 로깅은 컨테이너의 객체에 대해 이루어진 요청의 상세 레코드를 제공합니다. 액세스 로그는 보안 및 액세스 감사와 같은 여러 애플리케이션에 유용합니다. 또한 고객층을 파악하고 MediaStor 결제 요금을 확인할 수 있습니다. CloudWatch 로그는 다음과 같이 분류됩니다.

- 로그 스트림은 동일한 소스를 공유하는 로그 이벤트 시퀀스입니다.

- 로그 그룹은 동일한 보존 기간, 모니터링 및 액세스 제어 설정을 공유하는 로그 스트림의 그룹입니다. 컨테이너에서 액세스 로깅에 액세스하면 MediaStore가 `/aws/mediastore/MyContainerName`과 같은 이름으로 로그 그룹을 생성합니다. 로그 그룹을 정의하고 각 그룹에 배치할 스트림을 지정할 수 있습니다. 하나의 로그 그룹에서 포함할 수 있는 로그 스트림의 수에는 할당량이 없습니다.

기본적으로 로그는 무기한으로 저장되고 만료 기간이 없습니다. 로그 그룹별로 보존 정책을 조정하여 무기한으로 보존하거나 1일부터 10년까지 보존 기간을 선택할 수 있습니다.

Amazon CloudWatch에 대한 권한 설정

AWS Identity and Access Management(IAM)을 사용하여 AWS Elemental MediaStore에 Amazon CloudWatch에 대한 액세스 권한을 부여하는 역할을 생성합니다. 계정에 대해 CloudWatch Logs를 게시하려면 다음 단계를 수행해야 합니다. CloudWatch는 계정에 대한 지표를 자동으로 게시합니다.

CloudWatch에 대한 MediaStore 액세스를 허용하기

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. IAM 콘솔의 탐색 창에서 정책(Policies)을 선택한 후 정책 생성(Create policy)을 선택합니다.
3. JSON 탭을 선택하고 다음 정책을 붙여넣습니다.

```
{
    "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/*"
```

```
        }
    ]
}
```

이 정책은 MediaStore가 모든 리전에서 AWS 계정의 모든 컨테이너에 대해 로그 그룹 및 로그 스트림을 생성하도록 허용합니다.

4. Review policy(정책 검토)를 선택합니다.
5. 정책 검토 페이지에서 이름에 **MediaStoreAccessLogsPolicy**를 입력한 다음 정책 생성을 선택합니다.
6. IAM 콘솔의 탐색 창에서 역할을 선택하고 역할 만들기를 선택합니다.
7. Another AWS account(다른 AWS 계정) 역할 유형을 선택합니다.
8. 계정 ID에 AWS 계정 ID를 입력합니다.
9. 다음: 권한을 선택합니다.
10. 검색 상자에 **MediaStoreAccessLogsPolicy**를 입력합니다.
11. 새 정책 옆에 있는 확인란을 선택한 다음, 다음: 태그를 선택합니다.
12. 다음: 검토를 선택하여 새 사용자를 미리 봅니다.
13. 역할 이름에 **MediaStoreAccessLogs**를 입력한 다음 역할 생성을 선택합니다.
14. 확인 메시지에서 방금 생성한 역할의 이름(**MediaStoreAccessLogs**)을 선택합니다.
15. 역할의 요약 페이지에서 신뢰 관계 탭을 선택합니다.
16. 신뢰 관계 편집(Edit trust relationship)을 선택합니다.
17. 정책 문서에서 보안 주체부터 MediaStore 서비스까지 변경합니다. 형식은 다음과 같아야 합니다.

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

전체 결과는 다음과 같습니다.

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

```
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

18. 신뢰 정책 업데이트(Update Trust Policy)를 선택합니다.

컨테이너에 대한 액세스 로깅 활성화

기본적으로 AWS Elemental MediaStore는 액세스 로그를 수집하지 않습니다. 컨테이너에서 액세스 로깅을 활성화하면 MediaStore가 해당 컨테이너에 저장된 객체의 액세스 로그를 Amazon CloudWatch로 전송합니다. 액세스 로그는 컨테이너에 저장된 객체에 대해 이루어진 요청의 상세 레코드를 제공합니다. 이 정보에는 요청 유형, 요청에 지정된 리소스, 요청을 처리한 날짜 및 시간 등이 포함됩니다.

> **Important**
>
> MediaStore 컨테이너에서 액세스 로깅을 활성화하더라도 별도 요금이 부과되지 않습니다. 단, 이 서비스가 사용자에게 전달하는 로그 파일에 대해서는 일반적인 스토리지 요금이 발생합니다. (로그 파일은 언제든지 삭제할 수 있습니다.) AWS는 로그 파일 전달에 따른 데이터 전송 요금이 발생하지는 않지만 로그 파일 액세스에 따른 일반 데이터 전송 요금은 부과됩니다.

액세스 로깅을 활성화하려면(AWS CLI)

- AWS CLI에서 `start-access-logging` 명령을 사용합니다.

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

  이 명령은 반환 값이 없습니다.

컨테이너에 대한 액세스 로깅 비활성화

컨테이너에서 액세스 로깅을 비활성화하면 AWS Elemental MediaStore가 액세스 로그를 Amazon CloudWatch로 전송하지 않습니다. 이러한 액세스 로그는 저장되지 않으며 따라서 검색할 수 없습니다.

액세스 로그를 비활성화하려면(AWS CLI)

- AWS CLI에서 `stop-access-logging` 명령을 사용합니다.

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

이 명령은 반환 값이 없습니다.

AWS Elemental MediaStore의 액세스 로깅 문제 해결

AWS Elemental MediaStore 액세스 로그가 Amazon CloudWatch에 표시되지 않을 경우 다음 표에서 예상 원인 및 해결책을 참조하세요.

> **Note**
>
> 문제 해결 프로세스에 도움이 되도록 AWS CloudTrail Logs를 활성화해야 합니다.

증상	예상 원인	해결책
CloudTrail 로그가 활성화되었지만 CloudTrail 이벤트가 전혀 보이지 않습니다.	IAM 역할이 존재하지 않거나 그 이름, 권한 또는 신뢰 정책이 잘못되었습니다.	올바른 이름, 권한 및 신뢰 정책으로 역할을 생성합니다. the section called “CloudWatch에 대한 권한 설정” 섹션을 참조하세요.
`DescribeContainer` API 요청을 제출했지만 응답에 `AccessLoggingEnabled` 파라미터가 `False`의 값을 가지는 것으로 나타납니다. 또한 성공적인 `DescribeLogGroup`, `CreateLogGroup`, `DescribeLogStream` 또는 `CreateLogStream` 호출을 생성한 `MediaStoreAccessLogs` 역할에 대한 CloudTrail 이벤트가 전혀 보이지 않습니다.	IAM 역할이 존재하지 않거나 그 이름, 권한 또는 신뢰 정책이 잘못되었습니다.	올바른 이름, 권한 및 신뢰 정책으로 역할을 생성합니다. the section called “CloudWatch에 대한 권한 설정” 섹션을 참조하세요.
	컨테이너에서 액세스 로깅이 활성화되지 않았습니다.	컨테이너에서 액세스 로그를 활성화합니다. the section called “액세스 로깅 활성화” 섹션을 참조하세요.

증상	예상 원인	해결책

CloudTrai 콘솔에서 MediaStoreAccessLogs 역할과 관련된 액세스 거부 오류 이벤트가 표시됩니다. 이 CloudTrail 이벤트는 다음과 같은 행을 포함할 수 있습니다.
 	
`"eventSource": "logs.amazonaws.com",`
 	
`"errorCode": "AccessDenied",`
 	
`"errorMessage": "User: arn:aws:sts::111122223333:assumed-role/MediaStoreAccessLogs/MediaStoreAccessLogsSession is not authorized to perform: logs:DescribeLogGroups on resource: arn:aws:logs:us-west-2:111122223333:log-group::log-stream:",`
	IAM 역할에는 AWS Elemental MediaStore에 대해 올바른 권한이 없습니다.	IAM 역할을 업데이트하여 올바른 권한 및 신뢰 정책을 부여합니다. the section called “CloudWatch에 대한 권한 설정” 섹션을 참조하세요.

전체 컨테이너에 대해 로그가 전혀 보이지 않습니다.	계정이 리전별 계정당 로그 그룹에 대한 CloudWatch 할당량을 초과했을 수 있습니다. Amazon CloudWatch Logs 사용 설명서의 로그 그룹 할당량을 참조하세요.	CloudWatch 콘솔에서 계정이 로그 그룹에 대한 CloudWatch 할당량에 도달했는지 확인합니다. 필요한 경우 할당량 증가를 요청합니다.

증상	예상 원인	해결책
CloudWatch에서 일부 로그가 보이지만 예상대로 모든 로그가 보이지는 않습니다.	계정이 리전별 계정당 초당 트랜잭션에 대한 CloudWatch 할당량을 초과했을 수 있습니다. Amazon CloudWatch Logs 사용 설명서의 PutLogEvents 할당량을 참조하세요.	리전별 계정당 초당 CloudWatch 트랜잭션의 할당량 증가를 요청합니다.

액세스 로그 형식

액세스 로그 파일은 일련의 JSON 형식 로그 레코드로 구성되며, 각 로그 레코드마다 한 요청이 표시됩니다. 로그 안의 필드 순서는 다를 수 있습니다. 다음은 2개의 로그 레코드로 구성된 로그의 예입니다.

```
{
  "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
}
```


다음 목록에서는 로그 레코드 필드에 대해 설명합니다.

AWSAccountId

요청을 생성하는 데 사용된 계정의 AWS 계정 ID.

BytesReceived

MediaStore 서버가 수신하는 요청 본문의 바이트 수입니다.

BytesSent

MediaStore 서버가 송신하는 요청 본문의 바이트 수입니다. 이 값은 서버 응답과 함께 포함되는 `Content-Length` 헤더의 값과 동일한 경우가 자주 있습니다.

ContainerName

요청을 수신한 컨테이너의 이름입니다.

ErrorCode

MediaStore 오류 코드(예: `InternalServerError`) 발생한 오류가 없는 경우 `-` 문자가 표시됩니다. 상태 코드가 200(닫힌 연결 또는 서버가 응답을 스트리밍하기 시작한 후 오류를 나타냄)이라도 오류 코드가 표시될 수 있습니다.

ExpiresAt

객체의 만료 날짜 및 시간입니다. 이 값은 컨테이너에 적용되는 수명 주기 정책의 transient data rule에서 설정한 만료 기간을 기반으로 합니다. 이 값은 `ISO-8601` 날짜 시간이며 요청을 처리하는 호스트의 시스템 클록을 기준으로 합니다. 수명 주기 정책에 객체에 적용되는 임시 데이터 규칙

이 없거나 컨테이너에 적용된 수명 주기 정책이 없는 경우 이 필드의 값은 null입니다. 이 필드는 PutObject, GetObject, DescribeObject, DeleteObject 작업에만 적용됩니다.

HTTPStatus

응답의 숫자 HTTP 상태 코드.

작업

수행된 작업입니다(예: PutObject 또는 ListItems).

경로

컨테이너에서 객체가 저장된 경로. 작업이 경로 파라미터를 사용하지 않을 경우 - 문자가 표시됩니다.

ReceivedTime

요청이 수신된 시간입니다. 이 값은 ISO-8601 날짜 시간이며 요청을 처리하는 호스트의 시스템 클록을 기준으로 합니다.

요청자

요청을 생성하는 데 사용된 계정의 사용자 Amazon Resource Name(ARN). 인증되지 않은 요청은 이 값이 anonymous입니다. 인증이 완료되기 전에 요청이 실패하는 경우 이 필드가 로그에서 누락되었을 수 있습니다. 이러한 요청의 경우 ErrorCode에서 승인 문제를 식별할 수 있습니다.

RequestID

각 요청을 고유하게 식별하기 위해 AWS Elemental MediaStore에서 생성한 문자열입니다.

소스

호출을 생성한 AWS 서비스의 요청자 또는 서비스 보안 주체의 명백한 인터넷 주소. 중간 프록시 또는 방화벽이 요청을 생성한 시스템의 주소를 가릴 경우 이 값이 null로 설정됩니다.

TotalTime

서버 관점에서 요청이 플라이트 상태를 유지한 밀리초(ms) 단위 시간. 이 값은 서비스가 사용자로부터 요청을 수신한 시간에서 응답의 마지막 바이트를 전송한 시간까지 측정됩니다. 이 값은 클라이언트 관점에서 측정될 경우 네트워크 지연 시간에 의해 영향을 받으므로 서버 관점에서 측정됩니다.

TurnAroundTime

MediaStore가 요청을 처리하는 데 소비한 시간(밀리초)입니다. 이 값은 요청의 마지막 바이트가 수신된 시간부터 응답의 첫 바이트가 전송된 시간까지 측정됩니다.

로그 안의 필드 순서는 다를 수 있습니다.

로깅 상태 변경 시 일정 기간에 걸쳐 단계적으로 반영됨

로깅 상태를 변경한 후 실제 로그 파일의 전송에 반영되려면 어느 정도 시간이 지나야 합니다. 예를 들어, 컨테이너 A에 대해 로깅을 활성화할 경우 이후 1시간 동안 이루어진 요청 중 일부는 기록되지만 일부는 기록되지 않을 수도 있습니다. 컨테이너 B에 대해 로깅을 비활성화할 경우 이후 1시간 동안 일부 로그가 계속 전송될 수 있지만 다른 로그는 그렇지 않을 수 있습니다. 어떤 경우에도 사용자에 의한 추가 조치 없이 새로운 설정이 적용됩니다.

서버 로그 전송이 항상 보장되지는 않음

액세스 로그 레코드는 최대한 전송하겠지만 항상 모든 레코드가 전송된다고 보장할 수는 없습니다. 컨테이너에 대해 적절히 로깅이 구성된 대부분의 요청은 로그 레코드가 전송됩니다. 대부분 기록된 지 몇 시간 내로 로그 레코드가 전송되지만 더 자주 전송될 수 있습니다.

모든 액세스 로깅이 제때 전송될 것이라고 보장할 수는 없습니다. 특정 요청에 대한 로그 레코드는 요청이 실제로 처리된 후에 오랫동안 전송되거나 전혀 전송되지 않을 수도 있습니다. 액세스 로그는 컨테이너에 대한 트래픽의 특성을 파악할 용도로 제공되며, 실제로 로그 레코드가 누락되는 경우는 매우 드물지만 액세스 로깅 자체가 모든 요청을 완벽하게 기록할 목적으로 제공되는 것이 아닙니다.

완벽한 전송을 보장할 수 없는 액세스 로깅의 특성에 따라 AWS 포털에 제공되는 사용 보고서(AWS Management Console의 청구 및 비용 관리 보고서)에는 전송된 액세스 로그에 포함되지 않은 액세스 요청이 하나 이상 포함될 수 있습니다.

액세스 로그 형식 관련 프로그래밍 고려 사항

수시로 새로운 필드를 추가하여 액세스 로그 형식을 확장할 수 있습니다. 액세스 로그를 파싱하는 코드가 인식하지 못하는 추가 필드를 처리하도록 코드를 작성해야 합니다.

CloudWatch Events

Amazon CloudWatch Events를 사용하여 AWS 서비스를 자동화하고 애플리케이션 가용성 문제나 리소스 변경 같은 시스템 이벤트에 자동으로 응답합니다. 원하는 이벤트만 표시하도록 간단한 규칙을 작성한 후 규칙과 일치하는 이벤트 발생 시 실행할 자동화 작업을 지정할 수 있습니다.

> **Important**
>
> AWS 서비스는 이벤트 알림을 일반적으로 몇 초 안에 CloudWatch Events로 전송하지만 1분 이상 소요되는 경우도 있습니다.

파일을 컨테이너에 업로드하거나 컨테이너에서 제거할 경우 CloudWatch 서비스에서 두 가지 이벤트가 연속하여 수행됩니다.

1. the section called “객체 상태 변경 이벤트”
2. the section called “컨테이너 상태 변경 이벤트”

이러한 이벤트를 구독하는 자세한 방법은 Amazon CloudWatch를 참조하세요.

자동으로 트리거할 수 있는 작업은 다음과 같습니다.

- AWS Lambda 함수 호출
- Amazon EC2 Run Command 호출
- Amazon Kinesis Data Streams로 이벤트 릴레이
- AWS Step Functions 상태 머신 활성화
- Amazon SNS 주제 또는 AWS SMS 대기열 알림

CloudWatch Events를 AWS Elemental MediaStore에 사용하는 몇 가지 예는 다음과 같습니다.

- 컨테이너를 생성할 때마다 Lambda 함수를 실행
- 객체가 삭제될 때 Amazon SNS 주제를 알림

자세한 내용은 Amazon CloudWatch Events 사용 설명서를 참조하세요.

주제

- AWS Elemental MediaStore 객체 상태 변경 이벤트
- AWS Elemental MediaStore 컨테이너 상태 변경 이벤트

AWS Elemental MediaStore 객체 상태 변경 이벤트

이 이벤트는 객체의 상태가 변경되면(객체가 업로드되거나 삭제되면) 게시됩니다.

> **Note**
>
> 임시 데이터 규칙으로 인해 만료되는 객체는 만료 시 CloudWatch 이벤트를 생성하지 않습니다.

이 이벤트를 구독하는 자세한 방법은 Amazon CloudWatch를 참조하세요.

객체 업데이트됨

```
{
  "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"
  }
}
```

객체 제거됨

```
{
  "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"
  }
}
```

AWS Elemental MediaStore 컨테이너 상태 변경 이벤트

이 이벤트는 컨테이너의 상태가 변경되면(컨테이너가 추가되거나 삭제되면) 게시됩니다. 이 이벤트를 구독하는 자세한 방법은 Amazon CloudWatch를 참조하세요.

컨테이너 생성됨

```
{
  "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"
  }
}
```

컨테이너 제거됨

```
{
  "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"
  }
}
```

Amazon CloudWatch 지표를 사용한 AWS Elemental MediaStore 모니터링

원시 데이터를 수집하여 읽을 수 있는 지표로 처리하는 CloudWatch를 사용하면 AWS Elemental MediaStore를 모니터링할 수 있습니다. CloudWatch는 통계를 15개월간 보관하므로 기록 정보에 액세스하고 웹 애플리케이션 또는 서비스가 어떻게 실행되고 있는지 전체적으로 더 잘 파악할 수 있습니다. 특정 임계값을 주시하다가 해당 임계값이 충족될 때 알림을 전송하거나 조치를 취하도록 경보를 설정할 수도 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하세요.

AWS Elemental MediaStore의 경우 `BytesDownloaded`를 감시하다가 지표가 특정 임계값에 도달하면 본인에게 이메일을 보낼 수 있습니다.

CloudWatch 콘솔을 사용하여 지표를 보려면

지표는 먼저 서비스 네임스페이스별로 그룹화된 다음, 각 네임스페이스 내에서 다양한 차원 조합별로 그룹화됩니다.

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 지표를 선택합니다.
3. 모든 지표 아래에서 AWS/MediaPackage 네임스페이스를 선택합니다.
4. 지표를 보려면 지표 차원을 선택합니다. 예를 들어 컨테이너로 전송된 다양한 유형의 요청에 대한 지표를 보려면 `Request metrics by container`를 선택합니다.

AWS CLI를 사용하여 지표를 확인하려면

- 명령 프롬프트에서 다음 명령을 사용합니다.

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

AWS Elemental MediaStore 지표

다음 표에는 AWS Elemental MediaStore가 CloudWatch로 전송하는 지표가 나와 있습니다.

> **Note**
>
> 지표를 보려면 컨테이너에 지표 정책을 추가하여 MediaStore가 Amazon CloudWatch에 지표를 전송하도록 허용해야 합니다.

지표	설명
`RequestCount`	작업 유형(`Put`, `Get`, `Delete`, `Describe`, `List`)으로 구분된 MediaStore 컨테이너에 대한 총 HTTP 요청 수입니다. 단위: 개수 유효한 차원: • 컨테이너 이름 • 객체 그룹 이름 • 요청 유형 유효 통계: Sum
`4xxErrorCount`	4xx 오류를 초래한 MediaStore에 대한 HTTP 요청 수입니다. 단위: 개수 유효한 차원: • 컨테이너 이름 • 객체 그룹 이름 • 요청 유형 유효 통계: Sum

지표	설명
5xxErrorCount	5xx 오류를 초래한 MediaStore에 대한 HTTP 요청 수입니다. 단위: 개수 유효한 차원: • 컨테이너 이름 • 객체 그룹 이름 • 요청 유형 유효 통계: Sum
BytesUploaded	MediaStore 컨테이너에 대한 요청에 대해 업로드된 바이트 수로, 여기서 요청에는 본문이 포함됩니다. 단위: 바이트 유효한 차원: • 컨테이너 이름 • 객체 그룹 이름 유효한 통계: Average(요청당 바이트), Sum(기간당 바이트), Sample Count, Min(P0.0과 동일), Max(p100과 동일), p0.0과 p99.9 사이의 모든 백분위수

지표	설명
BytesDownloaded	응답에 본문이 포함된 MediaStore 컨테이너에 대한 요청에 대해 다운로드된 바이트 수입니다. 단위: 바이트 유효한 차원: • 컨테이너 이름 • 객체 그룹 이름 유효한 통계: Average(요청당 바이트), Sum(기간당 바이트), Sample Count, Min(P0.0과 동일), Max(p100과 동일), p0.0과 p99.9 사이의 모든 백분위수
TotalTime	서버 관점에서 요청이 플라이트 상태를 유지한 밀리초 단위 시간. 이 값은 MediaStore가 요청을 수신한 시간부터 응답의 마지막 바이트를 전송한 시간까지 측정됩니다. 이 값은 클라이언트 관점에서 측정될 경우 네트워크 지연 시간에 의해 영향을 받으므로 서버 관점에서 측정됩니다. 단위: 밀리초 유효한 차원: • 컨테이너 이름 • 객체 그룹 이름 • 요청 유형 유효한 통계: Average, Min(P0.0과 동일), Max(p100과 동일), p0.0과 p100 사이의 모든 백분위수

지표	설명
TurnaroundTime	MediaStore가 요청을 처리하는 데 소비한 시간(밀리초)입니다. 이 값은 MediaStore가 요청의 마지막 바이트를 수신한 시간부터 응답의 첫 번째 바이트를 전송한 시간까지 측정됩니다. 단위: 밀리초 유효한 차원: • 컨테이너 이름 • 객체 그룹 이름 • 요청 유형 유효한 통계: Average, Min(P0.0과 동일), Max(p100과 동일), p0.0과 p100 사이의 모든 백분위수
ThrottleCount	제한된 MediaStore에 대한 HTTP 요청 수입니다. 단위: 개수 유효한 차원: • 컨테이너 이름 • 객체 그룹 이름 • 요청 유형 유효 통계: Sum

AWS Elemental MediaStore 리소스에 태그 지정

태그는 사용자 또는 AWS가 AWS 리소스에 할당하는 사용자 지정 속성 레이블입니다. 각 태그에는 다음 두 가지 부분이 있습니다.

- 태그 키 (예: `CostCenter`, `Environment` 또는 `Project`) 태그 키는 대/소문자를 구별합니다.
- 태그 값(예: `111122223333` 또는 `Production`)으로 알려진 선택적 필드 태그 값을 생략하는 것은 빈 문자열을 사용하는 것과 같습니다. 태그 키처럼 태그 값은 대/소문자를 구별합니다.

태그는 다음을 지원합니다.

- AWS 리소스를 식별하고 정리합니다. 많은 AWS 서비스가 태그 지정을 지원합니다. 따라서 서로 다른 서비스의 리소스에 동일한 태그를 할당하여 리소스가 서로 관련이 있음을 나타낼 수 있습니다. 예를 들어 AWS Elemental MediaLive 입력에 할당한 것과 동일한 태그를 AWS Elemental MediaStore ####에 할당할 수 있습니다.
- AWS 비용을 추적합니다. AWS Billing and Cost Management 대시보드에서 이러한 태그를 활성화합니다. AWS는 태그를 사용하여 비용을 분류하고 월별 비용 할당 보고서를 제공합니다. 자세한 내용은 AWS Billing 사용 설명서의 비용 할당 태그 사용을 참조하세요.

다음 섹션에서는 AWS Elemental MediaStore의 태그에 대한 추가 정보를 제공합니다.

AWS Elemental MediaStore의 지원 리소스

AWS Elemental MediaStore의 다음 리소스는 태그 지정을 지원합니다.

- *"####"*

태그 추가 및 관리에 대한 자세한 내용은 태그 관리 단원을 참조하세요.

AWS Elemental MediaStore에서는 AWS Identity and Access Management(IAM)의 태그 기반 액세스 제어 기능을 지원하지 않습니다.

태그 이름 지정 및 사용 규칙

다음 기본 이름 지정 및 사용 규칙은 AWS Elemental MediaStore 리소스와 함께 태그를 사용할 때 적용합니다.

- 각 리소스는 최대 50개의 태그를 보유할 수 있습니다.
- 각 리소스에 대해 각 태그 키는 고유하며 하나의 값만 가질 수 있습니다.
- 태그 키의 최대 길이는 UTF-8 형식의 유니코드 문자 128자입니다.
- 태그 값의 최대 길이는 UTF-8 형식의 유니코드 문자 256자입니다.
- 허용되는 문자는 UTF-8로 표현할 수 있는 문자, 숫자, 공백 및 . : + = @ _ / -(하이픈) 문자도 있습니다. Amazon EC2 리소스는 모든 문자를 허용합니다.
- 태그 키와 값은 대/소문자를 구분합니다. 태그를 대문자로 사용하는 전략을 세우고 이러한 전략을 모든 리소스 유형에 대해 일관되게 구현하는 것이 가장 좋습니다. 예를 들어, `Costcenter`,

costcenter 또는 CostCenter를 사용할지 결정하고 모든 태그에 대해 동일한 규칙을 사용합니다. 대/소문자가 일치하지 않는 유사한 태그를 사용하지 마십시오.

- aws: 접두사는 AWS가 사용하도록 예약되어 있으므로 태그에 사용할 수 없습니다. 이 접두사가 지정된 태그 키나 값은 편집하거나 삭제할 수 없습니다. 이 접두사가 지정된 태그는 리소스당 태그 수 할당량에 포함되지 않습니다.

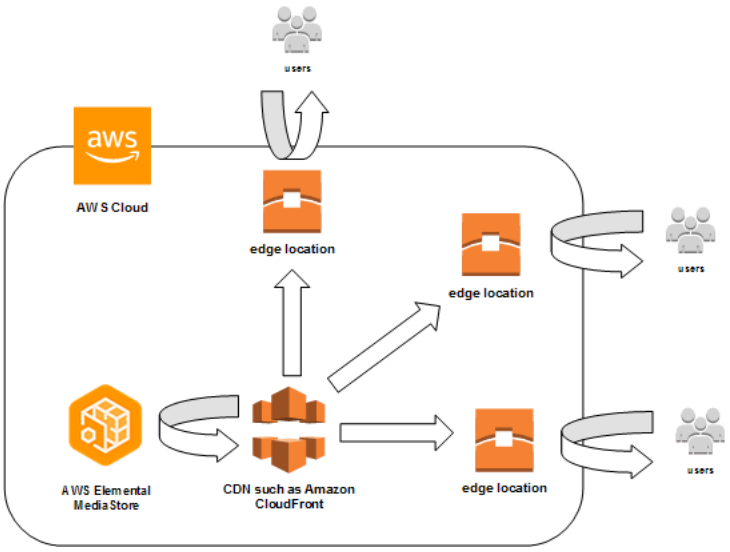
태그 관리

태그는 리소스의 Key 및 Value 속성으로 구성됩니다. AWS CLI 또는 MediaStore API를 사용하여 이 속성의 값을 추가, 편집 또는 삭제할 수 있습니다. 태그 작업에 대한 자세한 내용은 AWS Elemental MediaStore API 참조에서 다음 섹션을 참조하세요.

- CreateContainer
- ListTagsForResource
- 리소스
- TagResource
- UntagResource

CDN(콘텐츠 전송 네트워크) 작업

Amazon CloudFront 같은 콘텐츠 전송 네트워크(CDN)를 사용하여 AWS Elemental MediaStore에 저장된 콘텐츠를 제공할 수 있습니다. CDN은 전역적으로 배포된 서버 세트로 비디오 등의 콘텐츠를 캐싱합니다. 사용자가 콘텐츠를 요청하면 CDN은 이 요청을 지연 시간이 가장 짧은 엣지 로케이션으로 라우팅합니다. 콘텐츠가 해당 엣지 로케이션의 캐시에 이미 저장되어 있는 경우, CDN은 즉시 콘텐츠를 제공합니다. 엣지 로케이션에 콘텐츠가 없으면 CDN은 오리진(예: MediaStore 컨테이너)에서 콘텐츠를 가져와 사용자에게 배포합니다.



주제

- Amazon CloudFront가 AWS Elemental MediaStore 컨테이너에 액세스하도록 허용
- AWS Elemental MediaStore와 HTTP 캐시의 상호 작용

Amazon CloudFront가 AWS Elemental MediaStore 컨테이너에 액세스하도록 허용

Amazon CloudFront를 사용하면 AWS Elemental MediaStore의 컨테이너에 저장한 콘텐츠를 제공할 수 있습니다. 이 작업을 다음 중 한 가지 방법으로 수행할 수 있습니다.

- 원본 액세스 제어(OAC) 사용 - (권장) CloudFront의 OAC 기능을 AWS 리전에서 지원하는 경우 이 옵션을 사용합니다.
- 공유 암호 사용 - CloudFront의 OAC 기능을 AWS 리전에서 지원하지 않는 경우 이 옵션을 사용합니다.

원본 액세스 제어(OAC) 사용

Amazon CloudFront의 원본 액세스 제어(OAC) 기능을 사용하여 AWS Elemental MediaStore 오리진을 향상된 보안으로 보호할 수 있습니다. MediaStore 오리진에 대한 CloudFront 요청에서 AWS Signature Version 4(SigV4)를 활성화하고 CloudFront가 요청에 서명해야 하는 시기와 서명 여부를 설정할 수 있습니다. 콘솔, API, SDK 또는 CLI를 통해 CloudFront의 OAC 기능에 액세스할 수 있으며 사용에 따른 추가 요금은 없습니다.

MediaStore를 사용하여 OAC 기능을 사용하는 방법에 대한 자세한 내용은 Amazon CloudFront 개발자 안내서의 MediaStore 오리진에 대한 액세스 제한을 참조하세요.

공유 암호 사용

Amazon CloudFront의 OAC 기능을 AWS 리전에서 지원하지 않는 경우, CloudFront에 읽기 권한 이상을 부여하는 정책을 AWS Elemental MediaStore 컨테이너에 연결할 수 있습니다.

> **Note**
>
> AWS 리전에서 지원하는 경우 OAC 기능을 사용하는 것이 좋습니다. 다음 절차에서는 MediaStore 컨테이너에 대한 액세스를 제한하기 위해 공유 암호로 MediaStore 및 CloudFront를 구성해야 합니다. 모범 보안 관행을 따르려면 이 수동 구성에 암호를 주기적으로 교체해야 합니다. MediaStore 오리진에서 OAC를 사용하면 CloudFront가 SigV4를 사용하여 요청에 서명하고 서명 매칭을 위해 MediaStore에 전달하도록 지시할 수 있으므로 암호를 사용하고 교체할 필요가 없습니다. 이렇게 하면 미디어 콘텐츠가 제공되기 전에 요청이 자동으로 확인되므로 MediaStore와 CloudFront를 통해 미디어 콘텐츠를 더 간단하고 안전하게 전송할 수 있습니다.

CloudFront가 컨테이너에 액세스하도록 허용하기(콘솔)

1. https://console.aws.amazon.com/mediastore/에서 MediaStore 콘솔을 엽니다.
2. 컨테이너 페이지에서 컨테이너 이름을 선택합니다.

   컨테이너 세부 정보 페이지가 나타납니다.
3. 컨테이너 정책 섹션에서 읽기 이상의 액세스 권한을 Amazon CloudFront에 부여하는 정책을 연결합니다.

Example

HTTPS를 통해 퍼블릭 읽기 액세스에 대한 예제 정책과 유사한 다음 예제 정책은 HTTPS를 통해 요청을 제출하는 누구든 GetObject 및 DescribeObject 명령을 허용하므로 이러한 요구 사항과 일치합니다. 또한 다음 예제 정책은 HTTPS 연결을 통해 요청이 발생하고 올바른 레퍼러 헤더를 포함하는 경우에만 CloudFront가 MediaStore 객체에 액세스할 수 있도록 허용하므로 워크플로의 보안을 더욱 강화합니다.

```
{
  "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. 컨테이너 CORS 정책 섹션에서 적절한 액세스 수준을 허용하는 정책을 할당합니다.

> **Note**
>
> CORS 정책은 브라우저 기반 플레이어에 액세스를 제공하려는 경우에만 필요합니다.

5. 다음과 같은 세부 정보를 기록해 두십시오.

- 컨테이너에 할당된 데이터 엔드포인트. 이 정보는 컨테이너 페이지의 정보 섹션에서 확인할 수 있습니다. CloudFront에서 데이터 엔드포인트를 오리진 도메인 이름이라고 합니다.
- 객체가 저장되는 컨테이너의 폴더 구조. CloudFront에서는 이를 오리진 경로라고 합니다. 이 설정은 선택 사항입니다. 오리진 경로에 대한 자세한 내용을 알아보려면 Amazon CloudFront 개발자 안내서를 참조하세요.

6. In CloudFront에서 AWS Elemental MediaStore의 서비스 콘텐츠를 제공하도록 구성된 배포를 만듭니다. 앞 단계에서 수집한 정보가 필요합니다.

정책을 MediaStore 컨테이너에 연결한 후에는 원본 요청에 HTTPS 연결만 사용하도록 CloudFront를 구성하고 올바른 보안 값이 포함된 사용자 지정 헤더도 추가해야 합니다.

리퍼러 헤더(콘솔)에 대한 비밀 값을 사용하여 HTTPS 연결을 통해 컨테이너에 액세스하도록 CloudFront를 구성하기

1. CloudFront 콘솔을 엽니다.
2. 오리진 페이지에서 MediaStore 오리진을 선택합니다.
3. 편집을 선택합니다.
4. 프로토콜에 대해 HTTP만 선택합니다.
5. 사용자 지정 헤더 추가 섹션에서 헤더 추가를 선택합니다.
6. 이름의 경우 리퍼러를 선택합니다. 값의 경우 컨테이너 정책에서 사용한 것과 동일한 *<secretValue>* 문자열을 사용하세요.
7. 저장을 선택하고 변경 내용을 배포합니다.

AWS Elemental MediaStore와 HTTP 캐시의 상호 작용

AWS Elemental MediaStore에서는 Amazon CloudFront와 같은 콘텐츠 전송 네트워크(CDN)를 통해 객체를 정확하고 효율적으로 캐싱할 수 있도록 객체를 저장합니다. 최종 사용자 또는 CDN이 MediaStore에서 개체를 검색할 때 서비스는 개체의 캐싱 동작에 영향을 주는 HTTP 헤더를 반환합니다. HTTP 1.1 캐싱 동작에 대한 표준은 RFC2616 섹션 13에서 찾을 수 있습니다. 이러한 헤더는 다음과 같습니다.

- **ETag**(사용자 지정할 수 없음) - 엔터티 태그 헤더는 MediaStore에서 보내는 응답의 고유 식별자입니다. 표준을 준수하는 CDN 및 웹 브라우저는 이 태그를 객체를 캐시하는 키로 사용합니다.

MediaStore는 각 ETag 객체를 업로드할 때 각 객체에 대해 자동으로 생성합니다. 객체의 세부 정보를 보고 ETag 값을 결정할 수 있습니다.

- **Last-Modified**(사용자 지정할 수 없음) - 이 헤더의 값은 객체가 수정된 날짜 및 시간을 나타냅니다. MediaStore는 객체가 업로드될 때 이 값을 자동으로 생성합니다.
- **Cache-Control**(사용자 지정 가능) - 이 헤더의 값은 CDN이 객체가 수정되었는지 확인하기 전에 개체를 캐싱해야 하는 기간을 제어합니다. CLI 또는 API를 사용하여 MediaStore 컨테이너에 객체를 업로드할 때 이 헤더를 임의의 값으로 설정할 수 있습니다. 전체 유효한 값은 HTTP/1.1 설명서에 기술되어 있습니다. 객체를 업로드할 때 이 값을 설정하지 않으면 MediaStore에서는 객체를 검색할 때 이 헤더를 반환하지 않습니다.

  Cache-Control 헤더의 일반적인 사용 사례는 개체를 캐싱하는 기간을 지정하는 것입니다. 예를 들어, 인코더에서 자주 덮어쓰는 비디오 매니페스트 파일이 있다고 가정합니다. max-age를 10으로 설정하여 객체가 10초 동안만 캐싱되어야 함을 나타낼 수 있습니다. 또는 덮어쓰지 않는 저장된 비디오 세그먼트가 있다고 가정합니다. 이 객체에 대한 max-age를 31536000으로 설정하여 약 1년 동안 캐싱할 수 있습니다.

조건부 요청

MediaStore에 대한 조건부 요청

MediaStore는 조건부 요청(RFC7232의 설명에 따라 If-None-Match 및 If-Modified-Since 등의 요청 헤더 사용)과 무조건적인 요청에 동일하게 응답합니다. 즉, MediaStore에 유효한 GetObject 요청이 접수되면 클라이언트가 이미 개체를 가지고 있더라도 서비스는 항상 객체를 반환합니다.

CDN에 대한 조건부 요청

MediaStore를 대신하여 콘텐츠를 제공하는 CDN은 RFC7232 섹션 4.1에 설명된 대로 304 Not Modified을 반환하여 조건부 요청을 처리할 수 있습니다. 이는 요청자가 조건부 요청과 일치하는 객체를 이미 가지고 있기 때문에 전체 객체 컨텐츠를 전송할 필요가 없음을 나타냅니다.

CDN(및 HTTP/1.1을 준수하는 기타 캐시)은 오리진 서버에서 전달하는 ETag 및 Cache-Control 헤더를 기반으로 이러한 결정을 내립니다. 반복적으로 검색된 객체에 대한 업데이트를 위해 CDN이 MediaStore 오리진 서버를 쿼리하는 빈도를 제어하려면 객체를 MediaStore에 업로드할 때 해당 객체에 대한 Cache-Control 헤더를 설정합니다.

AWS Elemental MediaStore 내 할당량

Service Quotas 콘솔은 AWS Elemental MediaStore 할당량에 대한 정보를 제공합니다. 기본 할당량을 볼 수 있을 뿐만 아니라 Service Quotas 콘솔을 사용하여 조정 가능한 할당량에 대한 할당량 증가를 요청할 수 있습니다.

다음 표에는 AWS Elemental MediaStore의 할당량(이전에는 한도라고 함)이 설명되어 있습니다. 할당량은 AWS 계정의 최대 서비스 리소스 또는 작업 수입니다.

> **Note**
>
> 계정 내 개별 컨테이너에 할당량을 할당하려면 AWS Support 또는 계정 관리자에게 문의하세요. 이 옵션을 사용하면 컨테이너 간에 계정 수준 한도를 나누어 하나의 컨테이너가 전체 할당량을 사용하지 않도록 할 수 있습니다.

리소스 또는 작업	기본 할당량	설명
컨테이너	100	계정당 생성할 수 있는 컨테이너의 최대 수입니다.
폴더 수준	10	컨테이너에서 생성할 수 있는 폴더 수준의 최대 수입니다. 폴더가 컨테이너 내에서 10개를 초과하는 수준으로 중첩되지 않는 한 원하는 개수만큼 폴더를 생성할 수 있습니다.
폴더	무제한	폴더가 컨테이너 내에서 10개를 초과하는 수준으로 중첩되지 않는 한 원하는 개수만큼 폴더를 생성할 수 있습니다.
객체 크기	25MB	단일 객체의 최대 파일 크기
객체	무제한	계정의 폴더 또는 컨테이너에 원하는 개수만큼 객체를 업로드할 수 있습니다.
DeleteObject 요청 비율	100	초당 생성할 수 있는 작업 요청의 최대 수입니다. 추가 요청은 제한됩니다. 할당량 증가를 요청할 수 있습니다.

리소스 또는 작업	기본 할당량	설명
DeleteObject API 요청 비율	1,000	초당 생성할 수 있는 작업 요청의 최대 수입니다. 추가 요청은 제한됩니다. 할당량 증가를 요청할 수 있습니다.
표준 업로드 가용성에 대한 GetObject API 요청 비율	1,000	초당 생성할 수 있는 작업 요청의 최대 수입니다. 추가 요청은 제한됩니다. 할당량 증가를 요청할 수 있습니다.
스트리밍 업로드 가용성에 대한 GetObject API 요청 비율	25	초당 생성할 수 있는 작업 요청의 최대 수입니다. 추가 요청은 제한됩니다. 할당량 증가를 요청할 수 있습니다.
ListItems API 요청 비율	5	초당 생성할 수 있는 작업 요청의 최대 수입니다. 추가 요청은 제한됩니다. 할당량 증가를 요청할 수 있습니다.
청크 전송 인코딩에 대한 PutObject API 요청 비율(스트리밍 업로드 가용성이라고도 함)	10	초당 생성할 수 있는 작업 요청의 최대 수입니다. 추가 요청은 제한됩니다. 할당량 증가를 요청할 수 있습니다. 요청에서 요청한 TPS 및 평균 객체 크기를 지정합니다.
표준 업로드 가용성에 대한 PutObject API 요청 비율	100	초당 생성할 수 있는 작업 요청의 최대 수입니다. 추가 요청은 제한됩니다. 할당량 증가를 요청할 수 있습니다. 요청에서 요청한 TPS 및 평균 객체 크기를 지정합니다.
지표 정책의 규칙	10	지표 정책에 포함할 수 있는 최대 규칙 수입니다.
객체 수명 주기 정책의 규칙	10	단일 객체 수명 주기 정책에 포함할 수 있는 규칙의 최대 수입니다.

AWS Elemental MediaStore 관련 정보

다음 표에는 AWS Elemental MediaStore를 사용할 때 참조할 수 있는 관련 리소스가 나와 있습니다.

- 교육 및 워크숍 – 역할 기반의 과정 및 전문 과정은 물론 자습형 실습에 대한 링크를 통해 AWS 기술을 연마하고 실무에 도움이 되는 경험을 쌓을 수 있습니다.
- AWS 개발자 센터 – 자습서를 살펴보고, 도구를 다운로드하고, AWS 개발자 이벤트에 대해 알아보세요.
- AWS 개발자 도구 – AWS 애플리케이션을 개발 및 관리하기 위한 개발자 도구, SDK, IDE 도구 키트 및 명령줄 도구 링크입니다.
- 시작하기 리소스 센터 - AWS 계정을(를) 설정하고 AWS 커뮤니티에 가입하고 첫 번째 애플리케이션을 시작하는 방법을 알아보세요.
- 실습 자습서 – 단계별 자습서에 따라 AWS에서 첫 번째 애플리케이션을 시작하세요.
- AWS 백서 – AWS 솔루션 아키텍트 또는 기타 기술 전문가가 아키텍처, 보안 및 경제 등의 주제에 대해 작성한 포괄적 AWS 기술 백서 목록의 링크입니다.
- AWS Support 센터 – AWS Support 사례를 생성하고 관리할 수 있는 허브입니다. 또한 포럼, 기술 FAQ, 서비스 상태 및 AWS Trusted Advisor 등의 기타 유용한 자료에 대한 링크가 있습니다.
- AWS Support – 클라우드에서 1대 1로 애플리케이션을 구축 및 실행하도록 지원하는 빠른 응답 지원 채널인 AWS Support에 대한 정보가 포함된 기본 웹 페이지입니다.
- 문의처 - AWS 결제, 계정, 이벤트, 침해 및 기타 문제에 대해 문의할 수 있는 중앙 연락 창구입니다.
- AWS 사이트 약관 – 저작권 및 상표, 사용자 계정, 라이선스 및 사이트 액세스와 기타 주제에 대한 세부 정보입니다.

사용 설명서에 대한 문서 이력

다음 표는 본 AWS Elemental MediaStore 릴리스 관련 설명서를 소개합니다. 이 설명서에 대한 업데이트 알림을 받으려면 RSS 피드를 구독하면 됩니다.

변경 사항	설명	날짜
원본 액세스 제어(OAC) 개선	AWS Elemental MediaStore와 함께 OAC를 사용하는 방법에 대한 정보를 추가했습니다.	2023년 4월 17일
할당량 업데이트	`Rules in a Metric Policy`에 대한 할당량 값 및 설명을 수정했습니다.	2022년 10월 25일
ExpiresAt 필드	이제 액세스 로그에는 컨테이너 수명 주기 정책의 임시 데이터 규칙을 기반으로 객체의 만료 날짜 및 시간을 나타내는 `ExpiresAt` 필드가 포함됩니다.	2020년 7월 16일
수명 주기 전환 규칙	이제 객체 수명 주기 정책에 수명 주기 전환 규칙을 추가하여 특정 기간이 지난 객체를 IA(액세스 빈도 낮음) 스토리지 클래스로 이동하도록 설정할 수 있습니다.	2020년 4월 20일
컨테이너 비우기	이제 컨테이너 내의 모든 객체를 한 번에 삭제할 수 있습니다.	2020년 4월 7일
Amazon CloudWatch 지표 지원	지표 정책을 설정하여 MediaStore가 CloudWatch로 전송하는 지표를 지정할 수 있습니다.	2020년 3월 30일

객체 삭제 규칙의 와일드카드	이제 객체 수명 주기 정책에서 객체 삭제 규칙에 와일드카드를 사용할 수 있습니다. 이를 통해 특정 기간(일)이 지난 후 서비스에서 삭제할 파일을 파일 이름 또는 확장명을 기준으로 지정할 수 있습니다.	2019년 12월 20일
객체 수명 주기 정책	이제 수명을 초 단위로 나타내는 만료를 나타내는 규칙을 객체 수명 주기 정책에 추가할 수 있습니다.	2019년 9월 13일
AWS CloudFormation 지원	이제 AWS CloudFormation 템플릿을 사용하여 자동으로 컨테이너를 만들 수 있습니다. AWS CloudFormation 템플릿은 컨테이너 생성, 액세스 로깅 설정, 기본 컨테이너 정책 업데이트, CORS(교차 원본 리소스 공유) 정책 추가, 객체 수명 주기 정책 추가 등의 5가지 API 작업에 대한 데이터를 관리합니다.	2019년 5월 17일
스트리밍 업로드 가용성 할당량	스트리밍 업로드 가용성을 사용하는 객체의 경우(객체의 청크 분할 전송) `PutObject` 작업이 10TPS를 초과할 수 없고 `GetObject` 작업이 25TPS를 초과할 수 없습니다.	2019년 4월 8일
객체의 청크 분할 전송	객체의 청크 분할 전송에 대한 지원이 추가되었습니다. 이 기능은 객체가 완전히 업로드되기 전에 객체를 다운로드하도록 지정할 수 있게 해줍니다.	2019년 4월 5일

액세스 로깅	AWS Elemental MediaStore는 이제 컨테이너의 객체에 대해 이루어진 요청의 상세 레코드를 제공하는 액세스 로깅을 지원합니다.	2019년 2월 25일
객체 수명 주기 정책	객체 수명 주기 정책에 대한 지원이 추가되었습니다. 이 정책은 현재 컨테이너 안의 객체의 만료 날짜를 관리합니다.	2018년 12월 12일
객체 크기 할당량 증가	객체 크기의 할당량은 현재 25MB입니다.	2018년 10월 10일
객체 크기 할당량 증가	객체 크기의 할당량은 현재 20MB입니다.	2018년 9월 6일
AWS CloudTrail 통합	CloudTrail 통합 콘텐츠가 업데이트되어 CloudTrail 서비스에 대한 최신 변경 사항과 부합됩니다.	2018년 7월 12일
CDN 협업	Amazon CloudFront와 같은 콘텐츠 전송 네트워크(CDN)를 AWS Elemental MediaStore와 함께 사용하는 방법에 대한 정보를 추가했습니다.	2018년 4월 14일
CORS 구성	AWS Elemental MediaStore는 이제 교차 오리진 리소스 공유(CORS)를 지원합니다. 따라서 한 도메인에 로드된 클라이언트 웹 애플리케이션이 다른 도메인의 리소스와 상호 작용할 수 있습니다.	2018년 2월 7일

새로운 서비스 및 안내서	비디오 제작 및 스토리지 서비스인 AWS Elemental MediaStore와 AWS Elemental MediaStore 사용 설명서의 첫 릴리스입니다.	2017년 11월 27일

Note

- AWS Media Services는 안전 수명 작업, 탐색 또는 통신 시스템, 항공 교통 관제, 생명 유지 시스템 같이 서비스의 사용 불가, 중단 또는 장애가 사망, 개인 상해, 재산 손해, 환경 손해로 이어질 수 있다는 점에서 안전- 장치 성능이 필요한 애플리케이션이나 환경을 위한 용도로 설계되지 않았습니다.

AWS 용어집

최신 AWS 용어는 AWS 용어집 참조서의 AWS 용어집을 참조하세요.

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.