



使用者指南

AWS Elemental MediaStore



AWS Elemental MediaStore: 使用者指南

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商標和商業外觀不得用於任何非 Amazon 的產品或服務，也不能以任何可能造成客戶混淆、任何貶低或使 Amazon 名譽受損的方式使用 Amazon 的商標和商業外觀。所有其他非 Amazon 擁有的商標均為其各自擁有者的財產，這些擁有者可能附屬於 Amazon，或與 Amazon 有合作關係，亦或受到 Amazon 贊助。

Table of Contents

.....	vi
什麼是 MediaStore ?	1
概念和術語	1
相關服務	2
存取 MediaStore	3
定價	4
區域與端點	4
設定 AWS Elemental MediaStore	5
註冊 AWS 帳戶	5
建立具有管理存取權的使用者	5
開始使用	7
步驟 1：存取 AWS Elemental MediaStore	7
步驟 2：建立容器	7
步驟 3：上傳物件	8
步驟 4：存取物件	8
容器	9
容器名稱規則	9
建立容器	9
檢視容器詳細資訊	10
檢視容器清單	11
刪除容器	12
政策	14
容器政策	14
檢視容器政策	14
編輯容器政策	16
範例容器政策	17
CORS 政策	24
使用案例情境	25
新增 CORS 政策	25
檢視 CORS 政策	26
編輯 CORS 政策	27
刪除 CORS 政策	28
故障診斷	29
範例 CORS 政策	30

物件生命週期政策	31
物件生命週期政策的元件	32
新增物件生命週期政策	37
檢視物件生命週期政策	39
編輯物件生命週期政策	40
刪除物件生命週期政策	41
範例物件生命週期政策	42
指標政策	46
新增指標政策	47
檢視指標政策	47
編輯指標政策	47
指標政策範例	48
資料夾	51
資料夾名稱規則	51
建立資料夾	52
刪除資料夾	52
物件	53
上傳物件	53
檢視清單	55
檢視物件詳細資訊	57
下載物件	58
刪除物件	59
刪除一個物件	59
清空容器	60
安全	62
資料保護	62
資料加密	63
身分和存取權管理	63
目標對象	64
使用身分驗證	64
使用政策管理存取權	65
AWS Elemental MediaStore 如何與 IAM 搭配使用	66
身分型政策範例	71
疑難排解	74
日誌記錄和監控	75
Amazon CloudWatch 警示	76

AWS CloudTrail 日誌	76
AWS Trusted Advisor	76
法規遵循驗證	76
恢復能力	76
基礎設施安全性	77
預防跨服務混淆代理人	77
監控和標記	79
使用 CloudTrail 記錄 API 呼叫	79
CloudTrail 中的 MediaStore 資訊	80
範例：日誌檔案項目	81
使用 CloudWatch 進行監控	82
CloudWatch Logs	83
CloudWatch Events	91
CloudWatch 指標	95
標記	99
AWS Elemental MediaStore 中支援的資源	100
標籤命名和使用慣例	100
管理標籤	101
使用 CDN	102
允許 CloudFront 存取您的容器	102
使用原始存取控制 (OAC)	103
使用共用秘密	103
MediaStore 與 HTTP 快取的互動	104
條件式請求	105
使用 AWS SDKs	106
程式碼範例	108
基本概念	108
動作	109
配額	130
相關資訊	132
文件歷史紀錄	133
AWS 詞彙表	136

支援終止通知：2025 年 11 月 13 日，AWS 將停止對 AWS Elemental MediaStore 的支援。2025 年 11 月 13 日之後，您將無法再存取 MediaStore 主控台或 MediaStore 資源。如需詳細資訊，請造訪此[部落格文章](#)。

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。

什麼是 AWS Elemental MediaStore ？

AWS Elemental MediaStore 是一種影片起始和儲存服務，可提供即時起始所需的高效能和立即一致性。使用 MediaStore，您可以將視訊資產管理為容器中的物件，以建置可靠的雲端媒體工作流程。

若要使用 服務，您可以將物件從編碼器或資料饋送等來源上傳至您在 MediaStore 中建立的容器。

當您需要高度一致性、低延遲讀取和寫入，以及能夠處理大量並行請求時，MediaStore 是存放分段影片檔案的理想選擇。如果您未提供即時串流影片，請考慮改用 [Amazon Simple Storage Service \(Amazon S3\)](#)。

主題

- [AWS Elemental MediaStore 概念和術語](#)
- [相關服務](#)
- [存取 AWS Elemental MediaStore](#)
- [AWS Elemental MediaStore 定價](#)
- [AWS Elemental MediaStore 的區域和端點](#)

AWS Elemental MediaStore 概念和術語

ARN

[Amazon Resource Name](#)。

Body

要上傳至物件中的資料。

(位元組) 範圍

要加以定址之物件資料的子集。如需詳細資訊，請參閱 HTTP 規格中的[範圍](#)。

容器

保存物件的命名空間。容器會有一個您可用於附加寫入和擷取物件以及附加存取政策的端點。

端點

MediaStore 服務的進入點，以 HTTPS 根 URL 提供。

ETag

[實體標籤](#)，這是物件資料的雜湊值。

資料夾

容器的劃分區。資料夾可以保存物件及其他資料夾。

項目

用來指稱物件及資料夾的詞彙。

物件

類似 [Amazon S3 物件](#) 的資產。物件是存放在 MediaStore 中的基本實體。此服務接受所有檔案類型。

發送服務

MediaStore 被視為起始服務，因為它是媒體內容交付的分佈點。

路徑

物件或資料夾的唯一識別符，這可表示其在容器中的位置。

部分

物件的資料子集 (區塊)。

政策

[IAM 政策](#)。

資源

在 AWS 中您可以使用的實體。會為每個 AWS 資源指派 Amazon Resource Name (ARN)，此名稱可做為唯一識別符。在 MediaStore 中，這是資源及其 ARN 格式：

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

相關服務

- Amazon CloudFront 是一種全域內容交付網路 (CDN) 服務，可安全地將資料和影片交付給您的瀏覽 者。使用 CloudFront 以最佳效能交付內容。如需詳細資訊，請參閱 [Amazon CloudFront 開發人員指南](#)。
- CloudFormation 是一項服務，可協助您建立和設定 AWS 資源的模型。您可以建立範本來描述您想 要的所有 AWS 資源（例如 MediaStore 容器），並 CloudFormation 負責為您佈建和設定這些資

源。您不需要個別建立和設定 AWS 資源，並了解什麼依賴；CloudFormation 處理所有這些。如需詳細資訊，請參閱 [「AWS CloudFormation 使用者指南」](#)。

- AWS CloudTrail 是一項服務，可讓您監控對帳戶的 CloudTrail API 發出的呼叫，包括 AWS 管理主控台發出的呼叫，AWS CLI 以及其他服務。如需詳細資訊，請參閱 [「AWS CloudTrail 使用者指南」](#)。
- Amazon CloudWatch 是一種監控服務，適用於 AWS 雲端資源和您在其中執行的應用程式 AWS。使用 CloudWatch Events 追蹤 MediaStore 中容器和物件狀態的變更。如需詳細資訊，請參閱 [Amazon CloudWatch 文件](#)。
- AWS Identity and Access Management (IAM) 是一種 Web 服務，可協助您安全地控制使用者對 AWS 資源的存取。使用 IAM 控制誰可以使用您的 AWS 資源（身分驗證），以及使用者可以使用哪些資源（授權）。如需詳細資訊，請參閱 [設定 AWS Elemental MediaStore](#)。
- Amazon Simple Storage Service (Amazon S3) 是一種物件儲存體，用於從任何地方存放和擷取任意數量的資料。如需詳細資訊，請參閱 [Amazon S3 說明文件](#)。

存取 AWS Elemental MediaStore

您可以使用下列任一方法存取 MediaStore：

- AWS 管理主控台 - 本指南中的程序說明如何使用 AWS 管理主控台來執行 MediaStore 的任務。若要使用主控台存取 MediaStore：

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

- AWS Command Line Interface – 如需詳細資訊，請參閱 [AWS Command Line Interface 《使用者指南》](#)。若要使用 CLI 端點存取 MediaStore：

```
aws mediastore
```

- MediaStore API – 如果您使用的是無法使用 SDK 的程式設計語言，請參閱 [AWS Elemental MediaStore API 參考](#) 以取得 API 動作以及如何提出 API 請求的相關資訊。若要使用 REST API 端點存取 MediaStore：

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

- AWS SDKs – 如果您使用的是 AWS 提供開發套件的程式設計語言，您可以使用開發套件來存取 MediaStore。SDKs 可簡化身分驗證、輕鬆與您的開發環境整合，以及輕鬆存取 MediaStore 命令。如需詳細資訊，請參閱 [Amazon Web Services 適用工具](#)。

- 適用於 Windows PowerShell 的 AWS 工具 – 如需詳細資訊，請參閱 [AWS Tools for PowerShell 使用者指南](#)。

AWS Elemental MediaStore 定價

與其他 AWS 產品一樣，使用 MediaStore 沒有合約或最低承諾。我們向您收取的費用僅限內容進入服務時的每 GB 提取費，以及您在服務中存放內容的每 GB 月費。如需詳細資訊，請參閱 [AWS Elemental MediaStore 定價](#)。

AWS Elemental MediaStore 的區域和端點

為了減少應用程式中的資料延遲，MediaStore 提供區域端點來提出請求：

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

若要檢視可使用 MediaStore 的 AWS 區域完整清單，請參閱 [AWS 一般參考中的 AWS Elemental MediaStore 端點和配額](#)。

設定 AWS Elemental MediaStore

本節會引導您完成設定使用者存取 AWS Elemental MediaStore 所需的步驟。如需 MediaStore 身分和存取管理的背景和其他資訊，請參閱 [AWS Elemental MediaStore 的 Identity and Access Management](#)。

若要開始使用 AWS Elemental MediaStore，請完成下列步驟。

主題

- [註冊 AWS 帳戶](#)
- [建立具有管理存取權的使用者](#)

註冊 AWS 帳戶

如果您沒有 AWS 帳戶，請完成下列步驟來建立一個。

註冊 AWS 帳戶

1. 開啟 <https://portal.aws.amazon.com/billing/signup>。
2. 請遵循線上指示進行。

註冊程序的一部分包括接聽電話或文字訊息，並在電話鍵盤上輸入驗證碼。

當您註冊時 AWS 帳戶，AWS 帳戶根使用者會建立。根使用者有權存取該帳戶中的所有 AWS 服務和資源。作為安全最佳實務，請將管理存取權指派給使用者，並且僅使用根使用者來執行 [需要根使用者存取權的任務](#)。

AWS 會在註冊程序完成後傳送確認電子郵件給您。您可以隨時登錄 <https://aws.amazon.com/> 並選擇我的帳戶，以檢視您目前的帳戶活動並管理帳戶。

建立具有管理存取權的使用者

註冊後 AWS 帳戶，請保護 AWS 帳戶根使用者、啟用 AWS IAM Identity Center 和建立管理使用者，以免將根使用者用於日常任務。

保護您的 AWS 帳戶根使用者

1. 選擇根使用者並輸入 AWS 帳戶 您的電子郵件地址，以帳戶擁有者[AWS 管理主控台](#)身分登入。在下一頁中，輸入您的密碼。

如需使用根使用者登入的說明，請參閱 AWS 登入 使用者指南中的[以根使用者身分登入](#)。

2. 若要在您的根使用者帳戶上啟用多重要素驗證 (MFA)。

如需說明，請參閱《IAM 使用者指南》中的[為您的 AWS 帳戶 根使用者（主控台）啟用虛擬 MFA 裝置](#)。

建立具有管理存取權的使用者

1. 啟用 IAM Identity Center。

如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的[啟用 AWS IAM Identity Center](#)。

2. 在 IAM Identity Center 中，將管理存取權授予使用者。

如需使用 IAM Identity Center 目錄 做為身分來源的教學課程，請參閱AWS IAM Identity Center 《使用者指南》中的[使用預設值設定使用者存取 IAM Identity Center 目錄](#)。

以具有管理存取權的使用者身分登入

- 若要使用您的 IAM Identity Center 使用者簽署，請使用建立 IAM Identity Center 使用者時傳送至您電子郵件地址的簽署 URL。

如需使用 IAM Identity Center 使用者登入的說明，請參閱AWS 登入 《使用者指南》中的[登入 AWS 存取入口網站](#)。

指派存取權給其他使用者

1. 在 IAM Identity Center 中，建立一個許可集來遵循套用最低權限的最佳實務。

如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的[建立許可集](#)。

2. 將使用者指派至群組，然後對該群組指派單一登入存取權。

如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的[新增群組](#)。

AWS Elemental MediaStore 入門

本入門教學課程說明如何使用 AWS Elemental MediaStore 建立容器並上傳物件。

主題

- [步驟 1：存取 AWS Elemental MediaStore](#)
- [步驟 2：建立容器](#)
- [步驟 3：上傳物件](#)
- [步驟 4：存取物件](#)

步驟 1：存取 AWS Elemental MediaStore

設定 AWS 帳戶並建立使用者和角色之後，您就可以登入 AWS Elemental MediaStore 的主控制台。

存取 AWS Elemental MediaStore

- 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/mediastore/> 開啟 MediaStore 主控台。

Note

您可以使用您已為此帳戶建立的任何 IAM 登入資料來登入。如需有關建立 IAM 登入資料的詳細資訊，請參閱[設定 AWS Elemental MediaStore](#)。

步驟 2：建立容器

您可以使用 AWS Elemental MediaStore 中的容器來存放資料夾和物件。您可以使用容器將相關物件分組，其方式與使用目錄將檔案系統中的檔案分組相同。建立容器時，您不需付費；只有在您上傳物件到容器時才付費。

若要建立容器

1. 在 Containers (容器) 頁面上，選擇 Create container (建立容器)。
2. 在 Container name (容器名稱) 中，輸入容器的名稱。如需詳細資訊，請參閱[容器名稱規則](#)。

3. 選擇建立容器。AWS Elemental MediaStore 會將新容器新增至容器清單。容器一開始的狀態會是 Creating (建立中)，然後變更為 Active (啟用)。

步驟 3：上傳物件

您可以將物件 (各物件最多 25 MB) 上傳至容器或容器中的資料夾。若要將物件上傳至資料夾，請指定資料夾的路徑。如果資料夾已存在，AWS Elemental MediaStore 會將物件存放在資料夾中。如果資料夾不存在，則服務會建立此資料夾，再將物件存放在其中。

Note

物件檔案名稱只能包含字母、數字、句點 (.)、底線 (_)、波狀符號 (~) 和連字號 (-)。

若要上傳物件

1. 在 Containers (容器) 頁面上，選擇您剛才建立的容器的名稱。容器的詳細資訊頁面隨即出現。
2. 選擇 Upload object (上傳物件)。
3. 在 Target path (目標路徑) 中，輸入資料夾的路徑。例如 premium/canada。如果路徑中的任何資料夾尚不存在，AWS Elemental MediaStore 會自動建立它們。
4. 針對 Object (物件) 選擇 Browse (瀏覽)。
5. 導覽至適當的資料夾，然後選擇一個物件來上傳。
6. 選擇 Open (開啟)，然後選擇 Upload (上傳)。

步驟 4：存取物件

您可以將物件下載到指定的端點。

1. 在 Containers (容器) 頁面上，選擇有您所要下載之物件的容器的名稱。
2. 如果您想要下載的物件是在某個子資料夾中，請繼續選擇資料夾名稱，直到您看到該物件為止。
3. 選擇物件的名稱。
4. 在物件的詳細資訊頁面上，選擇 Download (下載)。

AWS Elemental MediaStore 中的容器

您可以在 MediaStore 中使用容器來存放資料夾和物件。相關物件可以在容器中進行分組，其方式與使用目錄將檔案系統中的檔案分組相同。建立容器時，您不需付費；只有在您上傳物件到容器時才付費。如需費用的詳細資訊，請參閱 [AWS Elemental MediaStore 定價](#)。

主題

- [容器名稱規則](#)
- [建立容器](#)
- [檢視容器的詳細資訊](#)
- [檢視容器清單](#)
- [刪除容器](#)

容器名稱規則

為容器選擇名稱時，請記得下列項目：

- 此名稱在目前帳戶的目前 AWS 區域中必須是唯一的。
- 此名稱可以包含大寫字母、小寫字母、數字和底線 (_)。
- 名稱長度必須介於 1 至 255 個字元。
- 名稱區分大小寫。例如，您可以有名為 myContainer 的容器以及名為 mycontainer 的資料夾，因為這些名稱是唯一的。
- 不可在建立容器後加以重新命名。

建立容器

您可以為每個 AWS 帳戶建立最多 100 個容器。您可以視需要建立任何數量的資料夾，只要容器中的巢狀層級不超過 10 層。此外，您可以將任意數量的物件上傳到每個容器。

Tip

您也可以使用 CloudFormation 範本自動建立容器。CloudFormation 範本可管理五個 API 動作的資料：建立容器、設定存取記錄、更新預設容器政策、新增跨來源資源共享 (CORS) 政策，以及新增物件生命週期政策。如需詳細資訊，請參閱 [「AWS CloudFormation 使用者指南」](#)。

若要建立容器 (主控台)

1. 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇 Create container (建立容器)。
3. 在 Container (容器) 名稱中，輸入容器的名稱。如需詳細資訊，請參閱[容器名稱規則](#)。
4. 選擇建立容器。AWS Elemental MediaStore 會將新容器新增至容器清單。容器一開始的狀態會是 Creating (建立中)，然後變更為 Active (啟用)。

若要建立容器 (AWS CLI)

- 在中 AWS CLI，使用 create-container 命令：

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

以下範例顯示傳回值：

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

檢視容器的詳細資訊

容器的詳細資訊包括容器政策、端點、ARN 和建立時間。

若要檢視容器詳細資訊 (主控台)

1. 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇容器的名稱。

容器詳細資訊頁面隨即出現。此頁面分為兩個區段：

- Objects (物件) 區段，其中列出容器中的物件和資料夾。
- Container policy (容器政策) 區段，其中顯示與此容器相關聯之依資源而定的政策。如需資源政策的相關資訊，請參閱[容器政策](#)。

若要檢視容器詳細資訊 (AWS CLI)

- 在中 AWS CLI，使用 `describe-container` 命令：

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

以下範例顯示傳回值：

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

檢視容器清單

您可以檢視所有與您帳戶相關之容器的清單。

若要檢視容器的清單 (主控台)

- 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。

Containers (容器) 頁面會出現，列出所有與您帳戶相關聯的容器。

若要檢視容器的清單 (AWS CLI)

- 在中 AWS CLI，使用 `list-containers` 命令。

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

以下範例顯示傳回值：

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

刪除容器

只有當容器沒有任何物件時，您才可以將該容器刪除。

若要刪除容器 (主控台)

1. 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。

2. 在 Containers (容器) 頁面上，選擇容器名稱左側的選項。
3. 選擇 刪除。

若要刪除容器 (AWS CLI)

- 在中 AWS CLI，使用 `delete-container` 命令：

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

此命令沒有傳回值。

AWS Elemental MediaStore 中的政策

您可以將下列一或多個政策套用至 AWS Elemental MediaStore 容器：

- [容器政策](#) - 設定容器內所有資料夾和物件的存取權限。MediaStore 會設定預設政策，允許使用者在容器上執行所有 MediaStore 操作。此政策指定所有操作必須透過 HTTPS 執行。建立容器之後，您可以編輯容器政策。
- [跨來源資源共用 \(CORS\) 政策](#) - 允許來自一個網域的用戶端 Web 應用程式與不同網域中的資源互動。MediaStore 不會設定預設 CORS 政策。
- [指標政策](#) - 允許 MediaStore 將指標傳送至 Amazon CloudWatch。MediaStore 不會設定預設指標政策。
- [物件生命週期政策](#) - 控制物件在 MediaStore 容器中保留的時間長度。MediaStore 不會設定預設物件生命週期政策。

AWS Elemental MediaStore 中的容器政策

每個容器都有依資源而定的政策，可管理該容器中所有資料夾及物件的存取權利。預設政策會自動連接到所有新容器，允許存取容器上的所有 AWS Elemental MediaStore 操作。政策會指定此存取有其必要條件，也就是必須使用 HTTPS 進行操作。建立容器後，您就可以編輯容器中連接的政策。

您也可以指定[物件生命週期政策](#)，此政策會在容器中管理物件的過期日期。物件達到您指定的年齡上限後，此服務就會將這些物件從容器中刪除。

主題

- [檢視容器政策](#)
- [編輯容器政策](#)
- [範例容器政策](#)

檢視容器政策

您可以使用 主控台 或 AWS CLI 檢視容器的資源型政策。

若要檢視容器政策 (主控台)

1. 在 <https://console.aws.amazon.com/mediastore/> 開啟 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇容器名稱。

容器詳細資訊頁面隨即出現。政策會顯示在 Container policy (容器政策) 區段中。

若要檢視容器政策 (AWS CLI)

- 在中 AWS CLI，使用 `get-container-policy` 命令：

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

以下範例顯示傳回值：

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

編輯容器政策

您可以編輯預設容器政策中的許可，也可以建立取代預設政策的新政策。新的政策需要最多 5 分鐘就會生效。

若要編輯容器政策 (主控台)

1. 在 <https://console.aws.amazon.com/mediastore/> 開啟 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇容器名稱。
3. 選擇 Edit Policy (編輯政策)。如需說明如何設定不同許可的範例，請參閱 [the section called “範例容器政策”](#)。
4. 進行適當變更，然後選擇 Save (儲存)。

若要編輯容器政策 (AWS CLI)

1. 建立檔案，此檔案會定義容器政策：

JSON

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

2. 在中 AWS CLI，使用 `put-container-policy` 命令：

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

此命令沒有傳回值。

範例容器政策

下列範例示範針對不同使用者群組所建構的容器政策。

主題

- [範例容器政策：預設](#)
- [範例容器政策：透過 HTTPS 的公有讀取存取](#)
- [範例容器政策：透過 HTTP 或 HTTPS 的公有讀取存取](#)
- [範例容器政策：跨帳戶讀取存取 - 啟用 HTTP](#)
- [範例容器政策：透過 HTTPS 的跨帳戶讀取存取](#)
- [範例容器政策：角色的跨帳戶讀取存取](#)
- [範例容器政策：角色的跨帳戶完整存取](#)
- [範例容器政策：限制存取特定 IP 地址](#)

範例容器政策：預設

當您建立容器時，AWS Elemental MediaStore 會自動連接下列以資源為基礎的政策：

JSON

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "MediaStoreFullAccess",  
      "Action": [  
        "mediastore:*"  
      ],  
      "Principal": {  
        "AWS": "arn:aws:iam::333333333333:root"  
      }  
    }  
  ]  
}
```

```

    },
    "Effect": "Allow",
    "Resource": "arn:aws:mediastore:us-
east-2:333333333333:container/<container name>/*",
    "Condition": {
      "Bool": {
        "aws:SecureTransport": "true"
      }
    }
  }
]
}

```

政策已內建於服務，因此您不需要建立。不過，如果預設[政策中的許可與您要用於容器的許可不相符](#)，您可以在容器上編輯政策。

指派給所有新容器的預設政策允許存取容器上的所有 MediaStore 操作。政策會指定此存取有其必要條件，也就是必須使用 HTTPS 進行操作。

範例容器政策：透過 HTTPS 的公有讀取存取

此範例政策允許使用者透過 HTTPS 請求擷取物件。此政策允許任何人透過安全的 SSL/TLS 連線進行讀取存取：驗證過身分的使用者以及匿名使用者 (未登入的使用者)。此陳述式的名稱為 PublicReadOverHttps。這允許對任何物件 (如資源路徑結尾的 * 所指定) 進行 GetObject 和 DescribeObject 操作存取。政策會指定此存取有其必要條件，也就是必須使用 HTTPS 進行操作：

JSON

```

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

```

```

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

```

範例容器政策：透過 HTTP 或 HTTPS 的公有讀取存取

此範例政策允許對任何物件 (如資源路徑結尾的 * 所指定) 進行 GetObject 和 DescribeObject 操作存取。此政策允許任何人進行讀取存取，包括所有驗證過身分的使用者以及匿名使用者 (未登入的使用者)：

JSON

```

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

```

範例容器政策：跨帳戶讀取存取 - 啟用 HTTP

此範例政策允許使用者透過 HTTP 請求擷取物件。這允許具備跨帳戶存取權且驗證過身分的使用者進行此存取。物件不一定要在使用 SSL/TLS 憑證的伺服器上託管：

JSON

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

範例容器政策：透過 HTTPS 的跨帳戶讀取存取

此範例政策允許存取指定 <other acct number> 之根使用者擁有的任何物件（如資源路徑結尾的 * 所指定）上的 GetObject 和 DescribeObject 操作。政策會指定此存取有其必要條件，也就是必須使用 HTTPS 進行操作：

JSON

```
{
```

```

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

```

範例容器政策：角色的跨帳戶讀取存取

此範例政策允許對任何由 <擁有者帳號> 所擁有的物件 (如資源路徑結尾的 * 所指定) 進行 GetObject 和 DescribeObject 操作存取。這允許 <其他帳號> 的任何使用者進行此存取 (如果該帳戶有擔任 <角色名稱> 所指定的角色)。

```

{
  "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>/*",
    }
  ]
}

```

}

範例容器政策：角色的跨帳戶完整存取

此範例政策允許跨帳戶存取來更新帳戶中的任何物件，只要使用者是透過 HTTP 登入即可。這也允許對已擔任指定角色的帳戶進行跨帳戶存取，透過 HTTP 或 HTTPS 來刪除、下載和描述物件：

- 第一個陳述式是 `CrossAccountRolePostOverHttps`。這允許對任何物件進行 `PutObject` 操作存取，並允許指定之帳戶的任何使用者進行此存取 (如果該帳戶有擔任 <角色名稱> 所指定的角色)。政策會指定此存取有其必要條件，也就是必須使用 HTTPS 進行操作 (提供 `PutObject` 存取權時，務必要包含此條件)。

換言之，任何擁有跨帳戶存取權的委託人都可以存取 `PutObject`，但只能透過 HTTPS 進行。

- 第二個陳述式是 `CrossAccountFullAccessExceptPost`。這允許對任何物件進行除了 `PutObject` 以外的所有操作存取。這允許指定之帳戶的任何使用者進行此存取 (如果該帳戶有擔任 <角色名稱> 所指定的角色)。此存取沒有限定條件要求使用 HTTPS 進行操作。

換言之，任何具有跨帳戶存取權的帳戶都可以進行 `DeleteObject`、`GetObject` 等存取 (但 `PutObject` 除外)，而且可以透過 HTTP 或 HTTPS 進行此存取。

如果沒有將 `PutObject` 從第二個陳述式排除，則陳述式不會有效 (因為要是包含 `PutObject`，您就必須依條件明確設定 HTTPS)。

JSON

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

```

    }
  },
  {
    "Sid": "CrossAccountFullAccessExceptPost",
    "Effect": "Allow",
    "NotAction": "mediastore:PutObject",
    "Principal": {
      "AWS": "arn:aws:iam::333333333333:role/<role name>"
    },
    "Resource": "arn:aws:mediastore:us-
east-2:333333333333:container/<container name>/*"
  }
]
}

```

範例容器政策：限制存取特定 IP 地址

此範例政策允許存取指定容器中物件的所有 AWS Elemental MediaStore 操作。不過，要求必須源自於條件中所指定的 IP 地址範圍。

此陳述式中的條件會識別允許之 Internet Protocol Version 4 (IPv4) IP 地址的 198.51.100.* 範圍，但有一個例外：198.51.100.188。

Condition 區塊使用 IpAddress 與 NotIpAddress 條件以及 aws:SourceIp 條件鍵，其為整個 AWS 的條件鍵。aws:sourceIp IPv4 值會使用標準 CIDR 表示法。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IP 地址條件運算子](#)。

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessBySpecificIPAddress",
      "Effect": "Allow",
      "Action": [
        "mediastore:GetObject",
        "mediastore:DescribeObject"
      ],
      "Principal": "*"
    }
  ]
}

```

```
"Resource": "arn:aws:mediastore:us-east-2:333333333333:container/<container name>/*",
"Condition": {
  "IpAddress": {
    "aws:SourceIp": [
      "198.51.100.0/24"
    ]
  },
  "NotIpAddress": {
    "aws:SourceIp": "198.51.100.188/32"
  }
}
}
```

AWS Elemental MediaStore 中的跨來源資源共享 (CORS) 政策

跨來源資源分享 (CORS) 會定義一種方式，讓載入單一個網域的用戶端 Web 應用程式，能與不同網域中的資源互動。透過 AWS Elemental MediaStore 中的 CORS 支援，您可以使用 MediaStore 建置豐富的用戶端 Web 應用程式，並選擇性地允許跨來源存取您的 MediaStore 資源。

Note

如果您使用 Amazon CloudFront 從具有 CORS 政策的容器分發內容，請務必[設定 AWS Elemental MediaStore 的分發](#)（包括編輯快取行為以設定 CORS 的步驟）。

本節提供 CORS 的概觀。子主題說明如何使用 AWS Elemental MediaStore 主控台啟用 CORS，或使用 MediaStore REST API 和 AWS SDKs 以程式設計方式啟用 CORS。

主題

- [CORS 使用案例情境](#)
- [將 CORS 政策新增至容器](#)
- [檢視 CORS 政策](#)
- [編輯 CORS 政策](#)
- [刪除 CORS 政策](#)
- [對 CORS 問題進行故障診斷](#)

- [範例 CORS 政策](#)

CORS 使用案例情境

以下為使用 CORS 的情境範例：

- 案例 1：假設您正在名為 LiveVideo 的 AWS Elemental MediaStore 容器中分發即時串流影片。您的使用者從特定來源 (如 <http://livevideo.mediastore.ap-southeast-2.amazonaws.com>) 載入資訊清單端點 www.example.com。您想要使用 JavaScript 影片播放器，透過未經驗證的 GET 及 PUT 要求來存取源自此容器的視訊。瀏覽器通常會封鎖 JavaScript，不讓系統允許這些要求，但您可以在容器上設定 CORS 政策，明確地啟用這些來自 www.example.com 的要求。
- 案例 2：假設您想要託管與 MediaStore 容器案例 1 中相同的即時串流，但想要允許來自任何原始伺服器的請求。您可設定 CORS 政策以允許使用萬用字元 (*) 來源，使任何來源的要求都能存取該視訊。

將 CORS 政策新增至容器

本節說明如何將跨來源資源共享 (CORS) 組態新增至 AWS Elemental MediaStore 容器。CORS 允許載入同一個網域的用戶端 Web 應用程式，可以與不同網域中的資源互動。

若要設定容器以允許跨來源要求，請將 CORS 政策新增至容器。CORS 政策會定義一些規則，這些規則可識別您所允許可存取容器的來源、每個來源支援的操作 (HTTP 方法)，以及其他操作特定資訊。

當您將 CORS 政策新增至容器時，[容器政策](#) (管理容器存取權利的政策) 照常適用。

若要新增 CORS 政策 (主控台)

1. 在 <https://console.aws.amazon.com/mediastore/> 開啟 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇您要為其建立 CORS 政策之容器的名稱。
容器詳細資訊頁面隨即出現。
3. 在 Container CORS policy (容器 CORS 政策) 區段中，選擇 Create CORS policy (建立 CORS 政策)。
4. 依 JSON 格式插入政策，然後選擇 Save (儲存)。

若要新增 CORS 政策 (AWS CLI)

1. 建立檔案，此檔案會定義 CORS 政策：

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

2. 在中 AWS CLI，使用 `put-cors-policy` 命令。

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

此命令沒有傳回值。

檢視 CORS 政策

跨來源資源分享 (CORS) 會定義一種方式，讓載入單一個網域的用戶端 Web 應用程式，能與不同網域中的資源互動。

若要檢視 CORS 政策 (主控台)

1. 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇您要檢視其 CORS 政策之容器的名稱。

容器詳細資訊頁面隨即出現，並將 CORS 政策顯示在 Container CORS policy (容器 CORS 政策) 區段中。

若要檢視 CORS 政策 (AWS CLI)

- 在中 AWS CLI，使用 `get-cors-policy` 命令：

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

以下範例顯示傳回值：

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

編輯 CORS 政策

跨來源資源分享 (CORS) 會定義一種方式，讓載入單一個網域的用戶端 Web 應用程式，能與不同網域中的資源互動。

若要編輯 CORS 政策 (主控台)

1. 在 <https://console.aws.amazon.com/mediastore/> 開啟 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇您要編輯其 CORS 政策之容器的名稱。

容器詳細資訊頁面隨即出現。

3. 在 Container CORS policy (容器 CORS 政策) 區段中，選擇 Edit CORS policy (編輯 CORS 政策)。
4. 對政策進行變更，然後選擇 Save (儲存)。

若要編輯 CORS 政策 (AWS CLI)

1. 建立檔案，此檔案會定義更新的 CORS 政策：

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

2. 在中 AWS CLI，使用 `put-cors-policy` 命令。

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

此命令沒有傳回值。

刪除 CORS 政策

跨來源資源分享 (CORS) 會定義一種方式，讓載入單一個網域的用戶端 Web 應用程式，能與不同網域中的資源互動。從容器刪除 CORS 政策會移除跨來源要求許可。

若要刪除 CORS 政策 (主控台)

1. 在 <https://console.aws.amazon.com/mediastore/> 開啟 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇您要刪除其 CORS 政策之容器的名稱。

容器詳細資訊頁面隨即出現。

3. 在 Container CORS policy (容器 CORS 政策) 區段中，選擇 Delete CORS policy (刪除 CORS 政策)。

4. 選擇 Continue (繼續) 來確認，然後選擇 Save (儲存)。

若要刪除 CORS 政策 (AWS CLI)

- 在中 AWS CLI，使用 `delete-cors-policy` 命令：

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

此命令沒有傳回值。

對 CORS 問題進行故障診斷

如果您在存取具有 CORS 政策的容器時遇到非預期行為，請依照下列步驟，對問題進行疑難排解。

1. 確認已將 CORS 政策附加至容器。

如需說明，請參閱 [the section called “檢視 CORS 政策”](#)。

2. 使用您選擇的工具 (例如，瀏覽器的開發人員主控台)，擷取完整的要求和回應。確認附加至容器的 CORS 政策至少包含一個比對您的要求中資料的 CORS 規則，如下所示：

- a. 確認要求含有 Origin 標頭。

如果缺少此標頭，AWS Elemental MediaStore 不會將要求視為跨來源要求，也不會在回應中傳送 CORS 回應標頭。

- b. 確認要求中的 Origin 標頭至少與特定 AllowedOrigins 的其中一個 CORSRule 元素比對相符。

Origin 要求標頭中的配置、主機及連接埠值，都必須符合 AllowedOrigins 中的 CORSRule。例如，若將 CORSRule 設定為允許來源 `http://www.example.com`，則要求中的 `https://www.example.com` 及 `http://www.example.com:80` 來源，都不符合您組態中允許的來源。

- c. 確認要求中的 Method (若為預檢要求，則是 Access-Control-Request-Method 中指定的方法) 是相同 AllowedMethods 的其中一個 CORSRule 元素。
- d. 若是預檢要求，如果要求包含 Access-Control-Request-Headers 標頭，則請確認 CORSRule 對於 AllowedHeaders 標頭中的每個值，都包含了 Access-Control-Request-Headers 項目。

範例 CORS 政策

下列範例示範跨來源資源分享 (CORS) 政策。

主題

- [範例 CORS 政策：任何網域皆可的讀取存取](#)
- [範例 CORS 政策：特定網域方可的讀取存取](#)

範例 CORS 政策：任何網域皆可的讀取存取

下列政策允許任何網域的網頁從您的 AWS Elemental MediaStore 容器擷取內容。要求包含所有來自原始網域的 HTTP 標頭，而且服務只會回應來自原始網域的 HTTP GET 和 HTTP HEAD 要求。系統先將結果暫存於快取達 3,000 秒之後，再傳送一組新的結果。

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

範例 CORS 政策：特定網域方可的讀取存取

下列政策允許 <https://www.example.com> 的網頁從您的 AWS Elemental MediaStore 容器擷取內容。要求包含所有來自 <https://www.example.com> 的 HTTP 標頭，而且服務只會回應來自 <https://www.example.com> 的 HTTP GET 和 HTTP HEAD 要求。系統先將結果暫存於快取達 3,000 秒之後，再傳送一組新的結果。

```
[
  {
```

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

AWS Elemental MediaStore 中的物件生命週期政策

對於每個容器，您可以建立物件生命週期政策，此政策可管理物件在容器中應存放的時間。當物件達到您指定的最長存留期時，AWS Elemental MediaStore 會刪除物件。您可以刪除不再需要的物件以節省儲存成本。

您也可以指定 MediaStore 應在物件達到特定年齡後，將物件移至不常存取 (IA) 儲存類別。存放在 IA 儲存體方案中的物件，其儲存和擷取的費率會和存放在標準儲存體方案中的物件不同。如需詳細資訊，請參閱 [MediaStore 定價](#)。

物件生命週期政策會包含規則，這些規則會依子資料夾規定物件的生命週期。(您無法將物件生命週期政策指派至個別物件)。您可以只將一個物件生命週期政策連接到容器，但您可以在每個物件生命週期政策中新增最多 10 個規則。如需詳細資訊，請參閱 [物件生命週期政策的元件](#)。

主題

- [物件生命週期政策的元件](#)
- [將物件生命週期政策新增至容器](#)
- [檢視物件生命週期政策](#)
- [編輯物件生命週期政策](#)
- [刪除物件生命週期政策](#)
- [範例物件生命週期政策](#)

物件生命週期政策的元件

物件生命週期政策可管理物件在 AWS Elemental MediaStore 容器中的保留時間。物件生命週期政策由一或多個規則組成，這些規則會規定物件的生命週期。規則可套用到一個資料夾、多個資料夾或者整個容器。

您可以只將一個物件生命週期政策連接到容器，且每個物件生命週期政策可包含最多 10 個規則。您無法將物件生命週期政策指派至個別物件。

物件生命週期政策中的規則

您可以建立三種規則類型：

- [暫時性資料](#)
- [刪除物件](#)
- [生命週期轉換](#)

暫時性資料

暫時性資料規則會設定物件在幾秒內過期。這種規則類型僅適用於在政策生效後新增到容器的物件。MediaStore 最多需要 20 分鐘才能將新政策套用至容器。

暫時性資料的規則範例如下所示：

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

暫時性資料規則有三個部分：

- **path**：一律設為 `wildcard`。您可以使用此部分來定義要刪除的物件。您可以使用一或多個萬用字元，以星號 (*) 表示。每個萬用字元代表零個或多個字元的任何組合。例如，`"path": [{"wildcard": "Football/index*.m3u8"}]`，會套用到 Football 資料夾中符合

`index*.m3u8` 模式 (例如 `index.m3u8`、`index1.m3u8` 和 `index123456.m3u8`) 的所有檔案。單一規則最多可以包含 10 個路徑。

- `seconds_since_create` : 一律設為 `numeric`。您可指定介於 1-300 秒之間的值。您也可以將運算子設定為大於 (`>`) 或大於或等於 (`>=`)。
- `action` : 一律設為 `EXPIRE`。

針對暫時性資料規則 (物件會在幾秒內過期)，在物件過期與刪除物件之間沒有延遲。

Note

`list-items` 回應中不會包含受暫時性資料規則限制的物件。此外，由於暫時性資料規則而過期的物件不會在過期時發出 CloudWatch 事件。

刪除物件

刪除物件規則會將物件設定為在幾天內過期。這種規則類型適用於容器中的所有物件，即使它們在建立原則之前已新增至容器。MediaStore 最多需要 20 分鐘才能套用新政策，但物件最多可能需要 24 小時才能從容器清除。

刪除物件的兩個規則範例如下所示：

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

刪除物件規則有三個部分：

- `path`：設為 `prefix` 或 `wildcard`。您不可在相同規則中混合使用 `prefix` 和 `wildcard`。如果要同時使用兩者，則必須為 `prefix` 建立一個規則，並為 `wildcard` 建立另一個規則，如上述範例所示。
- `prefix` - 如果您想要刪除特定資料夾內的所有物件，可以將路徑設為 `prefix`。如果參數是空的 (`"path": [{ "prefix": "" }],`)，目標就是存放在目前容器內任何位置的所有物件。單一規則最多可以包含 10 個 `prefix` 路徑。
- `wildcard` - 如果您要根據檔案名稱及/或檔案類型刪除特定物件，則可將路徑設為 `wildcard`。您可以使用一或多個萬用字元，以星號 (*) 表示。每個萬用字元代表零個或多個字元的任何組合。例如，`"path": [{"wildcard": "Football/*.ts"}]`，適用於 `Football` 資料夾中符合 `*.ts` 模式的所有檔案 (例如 `filename.ts`、`filename1.ts`，以及 `filename123456.ts`)。單一規則最多可以包含 10 個 `wildcard` 路徑。
- `days_since_create`：一律設為 `numeric`。您可指定介於 1-36,500 天之間的值。您也可以將運算子設定為大於 (>) 或大於或等於 (>=)。
- `action`：一律設為 `EXPIRE`。

若為刪除物件規則 (物件會在幾天內過期)，在物件過期與刪除物件之間可能會有些許延遲。不過，只要物件過期，帳單也會立即變更。例如，如果生命週期規則指定 10 `days_since_create`，在物件達到第 10 天後，此帳戶就不會針對此物件收費，即使尚未刪除此物件。

生命週期轉換

生命週期轉換規則會將物件設為在到達一定存留期後移動到不常存取 (IA) 儲存體方案 (以天數計)。存放在 IA 儲存體方案中的物件，其儲存和擷取的費率會和存放在標準儲存體方案中的物件不同。如需詳細資訊，請參閱 [MediaStore 定價](#)。

一旦物件移動到 IA 儲存體方案，您就無法將其移回標準儲存體方案。

生命週期轉換規則適用於容器中的所有物件，即使它們在建立原則之前已新增至容器。MediaStore 最多需要 20 分鐘才能套用新政策，但物件最多可能需要 24 小時才能從容器清除。

生命週期轉移規則的範例如下所示：

```
{
  "definition": {
    "path": [
      {"prefix": "AwardsShow/"}
    ],

```

```

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

```

生命週期轉換規則有三個部分：

- **path**：設為 **prefix** 或 **wildcard**。您不可在相同規則中混合使用 **prefix** 和 **wildcard**。如果要同時使用兩者，您必須為 **prefix** 建立一個規則，並為 **wildcard** 建立單獨建立另一個規則。
- **prefix** – 如果您希望將特定資料夾內的所有物件移動到 IA 儲存體方案，您可以將路徑設為 **prefix**。如果參數為空白 ("path": [{ "prefix": "" }],)，則目標就會是儲存在目前容器中任何位置的所有物件。單一規則最多可以包含 10 個 **prefix** 路徑。
- **wildcard** – 如果您希望根據檔案名稱和 (或) 檔案類型，將特定物件移動到 IA 儲存體方案，您可以將路徑設為 **wildcard**。您可以使用一或多個萬用字元，以星號 (*) 表示。每個萬用字元代表零個或多個字元的任何組合。例如，"path": [{"wildcard": "Football/*.ts"}]，適用於 Football 資料夾中符合 *.ts 模式的所有檔案 (例如 filename.ts、filename1.ts，以及 filename123456.ts)。單一規則最多可以包含 10 個 **wildcard** 路徑。
- **days_since_create**：一律設為 "numeric": [">=" , 30]。
- **action**：一律設為 ARCHIVE。

範例

例如，名為 LiveEvents 的容器有四個子資料夾：Football、Baseball、Basketball 和 AwardsShow。指派給 LiveEvents 資料夾的物件生命週期政策可能如下所示：

```

{
  "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"
}
]
```

上述政策會指定以下項目：

- 第一個規則會指示 AWS Elemental MediaStore 在LiveEvents/Football資料夾和LiveEvents/Baseball資料夾超過 28 天後刪除存放在資料夾中的物件。
- 第二個規則會指示此服務在 LiveEvents/AwardsShow 資料夾中存放的物件年齡為 15 天以上後，將其刪除。
- 第三個規則會指示此服務在 LiveEvents 容器中任何位置存放的物件年齡超過 40 天後，將其刪除。這個規則適用於直接在 LiveEvents 容器中存放的物件，以及在容器的四個子資料夾中任一項存放的物件。
- 第四個規則會服務在 Football 資料夾中符合模式 *.ts 的物件超過 20 天後，將其刪除。
- 第五個規則會指示服務在物件超過 15 秒index*.m3u8之後，刪除Football資料夾中符合模式的物件。MediaStore 會在這些檔案放入容器後 16 秒刪除這些檔案。
- 第六個規則會指示服務在物件存留期超過 30 天後，將 Program 資料夾中的物件移動到 IA 儲存體方案。

如需物件生命週期政策的更多範例，請參閱 [範例物件生命週期政策](#)。

將物件生命週期政策新增至容器

物件生命週期政策可讓您指定要將物件存放在容器多久。您可以設定過期日期，在過期日期之後，AWS Elemental MediaStore 會刪除物件。此服務最多需要 20 分鐘的時間，才能將新政策套用至容器。

如需有關如何建構生命週期政策的資訊，請參閱 [物件生命週期政策的元件](#)。

Note

若為刪除物件規則 (物件會在幾天內過期)，在物件過期與刪除物件之間可能會有些許延遲。不過，只要物件過期，帳單也會立即變更。例如，如果生命週期規則指定 10 `days_since_create`，在物件達到第 10 天後，此帳戶就不會針對此物件收費，即使尚未刪除此物件。

新增物件生命週期政策 (主控台)

1. 在 <https://console.aws.amazon.com/mediastore/> 開啟 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇您要為其建立物件生命週期政策之容器的名稱。

容器詳細資訊頁面隨即出現。

3. 在 Object lifecycle policy (物件生命週期政策) 區段中，選擇 Create object lifecycle policy (建立物件生命週期政策)。
4. 依 JSON 格式插入政策，然後選擇 Save (儲存)。

新增物件生命週期政策 (AWS CLI)

1. 建立檔案，此檔案會定義物件生命週期政策：

```
{
  "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. 在中 AWS CLI，使用 `put-lifecycle-policy` 命令：

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

此命令沒有傳回值。此服務會將指定政策連接至容器。

檢視物件生命週期政策

物件生命週期政策會指定物件在容器中應保留的時間。

檢視物件生命週期政策 (主控台)

1. 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇您要檢視其物件生命週期政策之容器的名稱。

容器詳細資訊頁面隨即出現，物件生命週期政策顯示在 Object lifecycle policy (物件生命週期政策) 區段中。

查看物件生命週期政策 (AWS CLI)

- 在中 AWS CLI，使用 `get-lifecycle-policy` 命令：

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

以下範例顯示傳回值：

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

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

編輯物件生命週期政策

您無法編輯現有的物件生命週期政策。不過，您可以透過上傳取代政策來變更現有的政策。此服務最多需要 20 分鐘的時間，才能將更新的政策套用至容器。

編輯物件生命週期政策 (主控台)

1. 在 <https://console.aws.amazon.com/mediastore/> 開啟 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇您要編輯其物件生命週期政策之容器的名稱。

容器詳細資訊頁面隨即出現。

3. 在 Object lifecycle policy (物件生命週期政策) 區段中，選擇 Edit object lifecycle policy (編輯物件生命週期政策)。
4. 對政策進行變更，然後選擇 Save (儲存)。

編輯物件生命週期政策 (AWS CLI)

1. 建立檔案，此檔案會定義更新的物件生命週期政策：

```
{
  "rules": [
    {
      "definition": {
        "path": [
```

```

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

```

2. 在中 AWS CLI，使用 `put-lifecycle-policy` 命令：

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

此命令沒有傳回值。此服務會將指定政策連接至容器，取代先前的政策。

刪除物件生命週期政策

刪除物件生命週期政策時，此服務最多需要 20 分鐘的時間，才能將變更套用至容器。

刪除物件生命週期政策 (主控台)

1. 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇您要刪除其物件生命週期政策之容器的名稱。

容器詳細資訊頁面隨即出現。

3. 在 Object lifecycle policy (物件生命週期政策) 區段中，選擇 Delete lifecycle policy (刪除生命週期政策)。
4. 選擇 Continue (繼續) 來確認，然後選擇 Save (儲存)。

刪除物件生命週期政策 (AWS CLI)

- 在中 AWS CLI，使用 `delete-lifecycle-policy` 命令：

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

此命令沒有傳回值。

範例物件生命週期政策

下列範例顯示物件生命週期政策。

主題

- [範例物件生命週期政策：在幾秒鐘內過期](#)
- [範例物件生命週期政策：在幾天內過期](#)
- [範例物件生命週期政策：轉換至不常存取儲存體方案](#)
- [範例物件生命週期政策：多重規則](#)
- [範例物件生命週期政策：空白容器](#)

範例物件生命週期政策：在幾秒鐘內過期

下列政策指定 MediaStore 刪除符合下列所有條件的物件：

- 物件是在政策生效後新增到容器的。
- 物件存放在 Football 資料夾中。
- 物件具有副檔名 m3u8。
- 物件已在容器中超過 20 秒。

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

```

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

```

範例物件生命週期政策：在幾天內過期

下列政策指定 MediaStore 刪除符合下列所有條件的物件：

- 物件存放在 Program 資料夾中
- 物件具有副檔名 ts
- 物件已在容器中超過 5 天

```

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

```

範例物件生命週期政策：轉換至不常存取儲存體方案

下列政策指定 MediaStore 在物件 30 天大時，將物件移至不常存取 (IA) 儲存類別。存放在 IA 儲存體方案中的物件，其儲存和擷取的費率會和存放在標準儲存體方案中的物件不同。

days_since_create 欄位必須設為 "numeric": [">=" ,30]。

```

{
  "rules": [
    {

```

```

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

```

範例物件生命週期政策：多重規則

下列政策指定 MediaStore 執行下列動作：

- 在物件存留期達 30 天後，將存放在 AwardsShow 資料夾中的物件移動到不常存取 (IA) 儲存體方案。
- 刪除副檔名為 m3u8 並且存放在 Football 資料夾中超過 20 秒的物件。
- 刪除存放在 April 資料夾中超過 10 天的物件。
- 刪除副檔名為 ts，且存放在 Program 資料夾中超過 5 天的物件。

```

{
  "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"
}
]
}

```

範例物件生命週期政策：空白容器

下列物件生命週期政策指定 MediaStore 會在容器新增至容器的 1 天後刪除容器中的所有物件，包括資料夾和子資料夾。如果容器在套用這項政策之前保留了任何物件，MediaStore 會在政策生效 1 天後刪除物件。此服務最多需要 20 分鐘的時間，才能將新政策套用至容器。

```

{
  "rules": [
    {
      "definition": {
        "path": [

```

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

AWS Elemental MediaStore 中的指標政策

對於每個容器，您可以新增指標政策，以允許 AWS Elemental MediaStore 將指標傳送至 Amazon CloudWatch。新的政策需要最多 20 分鐘就會生效。如需每個 MediaStore 指標的說明，請參閱 [MediaStore 指標](#)。

指標政策包含下列項目：

- 在容器層級啟用或停用指標的設定。
- 從零到五之間任意數量的規則，用在物件層級啟用指標。如果政策包含規則，則每個規則都必須包含下列兩項：
 - 一個物件群組，定義要包括在群組中的物件。定義可以是路徑或檔案名稱，但不能超過 900 個字元。有效字元包括：a-z、A-Z、0-9、_ (底線)、= (等號)、:(冒號)、.(句號)、-(連字號)、~ (波浪號)、/(正斜線) 和 *(星號)。可接受萬用字元 (*)。
 - 允許您參照物件群組的物件群組名稱。名稱不能超過 30 個字元。有效字元為 a-z、A-Z、0-9 和 _ (底線)。

如果物件符合多個規則，CloudWatch 會顯示每個相符規則的資料點。例如，如果一個物件符合兩個名為 rule1 和 rule2 的規則，則 CloudWatch 會針對這些規則顯示兩個資料點。第一個具有 ObjectGroupName=rule1 的維度，第二個具有 ObjectGroupName=rule2 的維度。

主題

- [新增指標政策](#)
- [檢視指標政策](#)
- [編輯指標政策](#)
- [指標政策範例](#)

新增指標政策

指標政策包含規定 AWS Elemental MediaStore 傳送至 Amazon CloudWatch 之指標的規則。如需指標政策的範例，請參閱[指標政策範例](#)。

新增指標政策 (主控台)

1. 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇您要為其新增指標政策之容器的名稱。

容器詳細資訊頁面隨即出現。

3. 在 Metric policy (指標政策) 區段中，選擇 Create metric policy (建立指標政策)。
4. 依 JSON 格式插入政策，然後選擇 Save (儲存)。

檢視指標政策

您可以使用 主控台或 AWS CLI 檢視容器的指標政策。

檢視指標政策 (主控台)

1. 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇容器名稱。

容器詳細資訊頁面隨即出現。政策會顯示在 Metric policy (指標政策) 區段中。

編輯指標政策

指標政策包含規定 AWS Elemental MediaStore 傳送至 Amazon CloudWatch 之指標的規則。當您編輯現有的指標政策時，新政策需要長達 20 分鐘才會生效。如需指標政策的範例，請參閱[指標政策範例](#)。

編輯指標政策 (主控台)

1. 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇容器名稱。
3. 在 Metric policy (指標政策) 區段中，選擇 Edit metric policy (編輯指標政策)。
4. 進行適當變更，然後選擇 Save (儲存)。

指標政策範例

下列範例示範針對不同使用案例所建構的指標政策。

主題

- [指標政策範例：容器層級指標](#)
- [指標政策範例：路徑層級指標](#)
- [指標政策範例：容器層級和路徑層級指標](#)
- [指標政策範例：使用萬用字元的路徑層級指標](#)
- [指標政策範例：具有重疊規則的路徑層級指標](#)

指標政策範例：容器層級指標

此範例政策指出 AWS Elemental MediaStore 應將指標傳送至容器層級的 Amazon CloudWatch。例如，這包括 RequestCount 指標，此指標會計算對容器提出的 Put 請求數目。或者，您可以將其設定為 DISABLED。

由於此政策中沒有規則，MediaStore 不會在路徑層級傳送指標。例如，您看不到對此容器內的特定資料夾提出的 Put 請求數。

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

指標政策範例：路徑層級指標

此範例政策指出 AWS Elemental MediaStore 不應將指標傳送至容器層級的 Amazon CloudWatch。此外，MediaStore 應該傳送兩個特定資料夾中物件的指標：baseball/saturday和 football/saturday。MediaStore 請求的指標如下所示：

- baseball/saturday 資料夾的請求具有的 CloudWatch 維度 ObjectGroupName=baseballGroup。
- football/saturday 資料夾的請求具有維度 ObjectGroupName=footballGroup。

```
{
```

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

指標政策範例：容器層級和路徑層級指標

此範例政策指出 AWS Elemental MediaStore 應將指標傳送至容器層級的 Amazon CloudWatch。此外，MediaStore 應該傳送兩個特定資料夾中物件的指標：baseball/saturday 和 football/saturday。MediaStore 請求的指標如下所示：

- baseball/saturday 資料夾的請求具有的 CloudWatch 維度 ObjectGroupName=baseballGroup。
- football/saturday 資料夾的請求具有 CloudWatch 維度 ObjectGroupName=footballGroup。

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

指標政策範例：使用萬用字元的路徑層級指標

此範例政策指出 AWS Elemental MediaStore 應將指標傳送至容器層級的 Amazon CloudWatch。此外，MediaStore 還應根據物件的檔案名稱傳送物件的指標。萬用字元表示物件可以儲存在容器中的任何位置，而且它們可以有任何檔案名稱，只要它以 `.m3u8` 副檔名結尾即可。

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

指標政策範例：具有重疊規則的路徑層級指標

此範例政策指出 AWS Elemental MediaStore 應將指標傳送至容器層級的 Amazon CloudWatch。此外，MediaStore 應該傳送兩個資料夾的指標：`sports/football/saturday`和 `sports/football`。

MediaStore 對 `sports/football/saturday` 資料夾的請求指標具有的 CloudWatch 維度 `ObjectGroupName=footballGroup1`。由於存放在 `sports/football` 資料夾中的物件符合兩個規則，CloudWatch 會顯示這些物件的兩個資料點：一個維度為 `ObjectGroupName=footballGroup1`，第二個維度為 `ObjectGroupName=footballGroup2`。

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

AWS Elemental MediaStore 中的資料夾

資料夾是容器內的劃分區。您使用資料夾細分容器，其方式與您建立子資料夾來劃分檔案系統中的資料夾相同。您可以建立最多 10 個層級的資料夾 (不包括容器本身)。

資料夾可選用；您可以選擇直接上傳物件給容器，而不是資料夾。不過，資料夾不失為組織物件的簡便方法。

若要將物件上傳至資料夾，請指定資料夾的路徑。如果資料夾已存在，AWS Elemental MediaStore 會將物件存放在資料夾中。如果資料夾不存在，則服務會建立此資料夾，再將物件存放在其中。

例如，假設您有一個名為 `movies` 的容器，並上傳一個名為 `m1aw.ts` 且路徑為 `premium/canada` 的檔案。AWS Elemental MediaStore 會將物件存放在資料夾高級下的子資料夾 `Canada` 中。如果這兩個資料夾都不存在，服務就會同時建立 `premium` 資料夾和 `canada` 子資料夾，然後將您的物件存放在 `canada` 子資料夾中。如果您只有指定 `movies` 容器 (沒有路徑)，服務則直接將物件存放在容器中。

當您刪除該資料夾中的最後一個物件時，AWS Elemental MediaStore 會自動刪除資料夾。服務也會刪除任何在該資料夾上層的空資料夾。例如，假設您有一個不含任何檔案、名為 `premium` 的資料夾，但其中包含一個名為 `canada` 的子資料夾。`canada` 子資料夾包含一個名為 `m1aw.ts` 的檔案。如果您刪除檔案 `m1aw.ts`，則服務會同時刪除 `premium` 和 `canada` 資料夾。此自動刪除動作僅適用於資料夾。服務並不會刪除空的容器。

主題

- [資料夾名稱規則](#)
- [建立資料夾](#)
- [刪除資料夾](#)

資料夾名稱規則

為資料夾選擇名稱時，請記得下列項目：

- 名稱只能包含下列字元：大寫字母 (A-Z)、小寫字母 (a-z)、數字 (0-9)、句點 (.)、連字號 (-)、波紋 (~)、底線 (_)、等號 (=) 和冒號 (:)。
- 名稱長度必須至少為一個字元。不允許空白資料夾名稱 (例如 `folder1//folder3/`)。

- 名稱區分大小寫。例如，您在同一個容器或資料夾中可以有名為 myFolder 的資料夾以及名為 myfolder 的資料夾，因為這些名稱是唯一的。
- 此名稱在其父系容器或資料夾內必須是唯一的。例如，您可以在兩個不同的容器 (myfolder 和 movies/myfolder) 中各建立一個名為 sports/myfolder 的資料夾。
- 此名稱可以與其父容器相同。
- 不可在建立資料夾後加以重新命名。

建立資料夾

您可以在上傳物件時建立資料夾。若要將物件上傳至資料夾，請指定資料夾的路徑。如果資料夾已存在，AWS Elemental MediaStore 會將物件存放在資料夾中。如果資料夾不存在，則服務會建立此資料夾，再將物件存放在其中。

如需詳細資訊，請參閱[the section called “上傳物件”](#)。

刪除資料夾

只有在資料夾是空的時，您才可以刪除資料夾；您無法刪除含有物件的資料夾。

當您刪除該資料夾中的最後一個物件時，AWS Elemental MediaStore 會自動刪除資料夾。服務也會刪除任何在該資料夾上層的空資料夾。例如，假設您有一個不含任何檔案、名為 premium 的資料夾，但其中包含一個名為 canada 的子資料夾。canada 子資料夾包含一個名為 mlaw.ts 的檔案。如果您刪除檔案 mlaw.ts，則服務會同時刪除 premium 和 canada 資料夾。此自動刪除動作僅適用於資料夾。服務並不會刪除空的容器。

如需詳細資訊，請參閱[刪除物件](#)。

AWS Elemental MediaStore 中的物件

AWS Elemental MediaStore 資產稱為物件。您可以將物件上傳至容器或容器中的資料夾。

在 MediaStore 中，您可以上傳、下載和刪除物件：

- 上傳 – 將物件新增至容器或資料夾。這與建立物件有所不同。您必須先在本機建立物件，才能將其上傳至 MediaStore。
- 下載 – 將物件從 MediaStore 複製到另一個位置。這不會從 MediaStore 移除物件。
- 刪除 – 從 MediaStore 完全移除物件。您可以個別刪除物件，或者可以[新增物件生命週期政策](#)，來自動在指定的持續時間後刪除容器中的物件。

MediaStore 接受所有檔案類型。

主題

- [上傳物件](#)
- [檢視物件的清單](#)
- [檢視物件的詳細資訊](#)
- [下載物件](#)
- [刪除物件](#)

上傳物件

您可以將物件上傳至容器，或容器中的資料夾。若要將物件上傳至資料夾，請指定資料夾的路徑。如果資料夾已存在，AWS Elemental MediaStore 會將物件存放在資料夾中。如果資料夾不存在，則服務會建立此資料夾，再將物件存放在其中。如需資料夾的詳細資訊，請參閱[AWS Elemental MediaStore 中的資料夾](#)。

您可以使用 MediaStore 主控台或 AWS CLI 來上傳物件。

MediaStore 支援物件區塊傳輸，這可透過讓物件在仍在上傳時可供下載來降低延遲。若要使用此功能，請將物件的上傳可用性設為 streaming。您可以在[使用 API 上傳物件](#)時，設定此標頭的值。如果您未在請求中指定此標頭，MediaStore 會 standard 為物件的上傳可用性指派預設值。

標準上傳可用性的物件大小不可超過 25 MB，串流上傳可用性的物件大小限制為 10 MB。

Note

物件檔案名稱只能包含字母、數字、句點 (.)、底線 (_)、波狀符號 (~)、連字號 (-)、等號 (=) 和冒號 (:)。

若要上傳物件 (主控台)

1. 在 <https://console.aws.amazon.com/mediastore/> 開啟 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇容器的名稱。容器的詳細資訊面板隨即出現。
3. 選擇 Upload object (上傳物件)。
4. 在 Target path (目標路徑) 中，輸入資料夾的路徑。例如 premium/canada。如果您所指定的路徑中有任何資料夾尚不存在，服務會自動建立該資料夾。
5. 在 Object (物件) 區段中，選擇 Browse (瀏覽)。
6. 導覽至適當的資料夾，然後選擇一個物件來上傳。
7. 選擇 Open (開啟)，然後選擇 Upload (上傳)。

Note

如果名稱相同的檔案已經存在於選取的資料夾，服務會將原始檔案取代為上傳的檔案。

若要上傳物件 (AWS CLI)

- 在中 AWS CLI，使用 put-object 命令。您也可以包含下列任何參數：content-type、cache-control (允許呼叫者控制物件的快取行為) 以及 path (將物件放在容器的某個資料夾中)。

Note

上傳物件後，就無法編輯 content-type、cache-control 或 path。

```
aws mediastore-data put-object --endpoint https://  
aaabbbcccdddee.data.mediastore.us-west-2.amazonaws.com --body README.md --path /
```

```
folder_name/README.md --cache-control "max-age=6, public" --content-type binary/octet-stream --region us-west-2
```

以下範例顯示傳回值：

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

檢視物件的清單

您可以使用 AWS Elemental MediaStore 主控台來檢視存放在容器最上層或資料夾中的項目（物件和資料夾）。不會顯示存放在目前容器或資料夾之子資料夾中的項目。您可以使用 AWS CLI 檢視容器內的物件和資料夾清單，無論容器內有多少個資料夾或子資料夾。

若要檢視特定容器中的物件清單 (主控台)

1. 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇有您所要檢視之資料夾的容器的名稱。
3. 從清單中選擇資料夾的名稱。

詳細資訊頁面隨即出現，並顯示所有存放在該資料夾中的資料夾及物件。

若要檢視特定資料夾中的物件清單 (主控台)

1. 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇有您所要檢視之資料夾的容器的名稱。

詳細資訊頁面隨即出現，並顯示所有存放在該容器中的資料夾及物件。

若要檢視特定容器中的物件及資料夾清單 (AWS CLI)

- 在中 AWS CLI，使用 `list-items` 命令：

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

以下範例顯示傳回值：

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

Note

受 `seconds_since_create` 規則限制的物件不會包含在 `list-items` 回應中。

若要檢視特定資料夾中的物件及資料夾清單 (AWS CLI)

- 在中 AWS CLI，使用 `list-items` 命令，並在請求結尾指定資料夾名稱：

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

以下範例顯示傳回值：

```
{
```

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

Note

受 `seconds_since_create` 規則限制的物件不會包含在 `list-items` 回應中。

檢視物件的詳細資訊

上傳物件之後，AWS Elemental MediaStore 會存放修改日期、內容長度、ETag（實體標籤）和內容類型等詳細資訊。若要了解如何使用物件的中繼資料，請參閱 [MediaStore 與 HTTP 快取的互動](#)。

若要檢視物件的詳細資訊 (主控台)

1. 開啟位於 <https://console.aws.amazon.com/mediastore/> 的 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇有您所要檢視之物件的容器的名稱。
3. 如果您想要檢視的物件是在某個資料夾中，請繼續選擇資料夾名稱，直到您看到該物件為止。
4. 選擇物件的名稱。

詳細資訊頁面隨即出現，並顯示物件的相關資訊。

若要檢視物件的詳細資訊 (AWS CLI)

- 在中 AWS CLI，使用 `describe-object` 命令：

Note

當您刪除資料夾中的唯一物件時，AWS Elemental MediaStore 會自動刪除資料夾和該資料夾上方的任何空資料夾。例如，假設您有一個不含任何檔案、名為 premium 的資料夾，但其中包含一個名為 canada 的子資料夾。canada 子資料夾包含一個名為 mlaw.ts 的檔案。如果您刪除檔案 mlaw.ts，則服務會同時刪除 premium 和 canada 資料夾。

若要刪除物件 (主控台)

1. 在 <https://console.aws.amazon.com/mediastore/> 開啟 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇具備您要刪除之物件的容器名稱。
3. 如果您想要刪除的物件是在某個資料夾中，請繼續選擇資料夾名稱，直到您看到該物件為止。
4. 選擇物件名稱左側的選項。
5. 選擇 刪除。

若要刪除物件 (AWS CLI)

- 在中 AWS CLI，使用 delete-object 命令。

範例：

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

此命令沒有傳回值。

清空容器

您可以清空容器，以刪除儲存在容器內的所有物件。或者，您可以[新增物件生命週期政策](#)，在物件於容器中存在超過特定存留期後自動刪除物件，或者您可以[個別刪除物件](#)。

清空容器 (主控台)

1. 在 <https://console.aws.amazon.com/mediastore/> 開啟 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，為您要清空的容器選擇選項。

3. 選擇 Empty container (清空容器)。會出現確認訊息。
4. 在文字欄位中輸入容器名稱，確認您想要清空容器，然後選擇空白。

AWS Elemental MediaStore 的安全性

的雲端安全性 AWS 是最高優先順序。身為 AWS 客戶，您可以受益於資料中心和網路架構，這些架構專為符合最安全敏感組織的需求而建置。

安全性是 AWS 與您之間共同責任。[共同責任模型](#)將其描述為雲端的安全性和雲端中的安全性：

- 雲端的安全性 – AWS 負責保護在 中執行 AWS 服務的基礎設施 AWS 雲端。AWS 也為您提供可安全使用的服務。作為[AWS 合規計畫](#)的一部分，第三方稽核人員會定期測試和驗證我們安全的有效性。若要了解適用於 AWS Elemental MediaStore 的合規計畫，請參閱[AWS 合規計畫的服務範圍](#)。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。您也必須對其他因素負責，包括資料的機密性、您公司的要求和適用法律和法規。

本文件可協助您了解如何在使用 MediaStore 時套用共同責任模型。下列主題說明如何設定 MediaStore 以符合您的安全與合規目標。您也會了解如何使用其他 AWS 服務來協助您監控和保護 MediaStore 資源。

主題

- [AWS Elemental MediaStore 中的資料保護](#)
- [AWS Elemental MediaStore 的 Identity and Access Management](#)
- [在 中記錄和監控 AWS Elemental MediaStore](#)
- [AWS Elemental MediaStore 的合規驗證](#)
- [AWS Elemental MediaStore 中的彈性](#)
- [AWS Elemental MediaStore 中的基礎設施安全](#)
- [預防跨服務混淆代理人](#)

AWS Elemental MediaStore 中的資料保護

AWS [共同責任模型](#)適用於 AWS Elemental MediaStore 中的資料保護。如此模型所述，AWS 負責保護執行所有的全域基礎設施 AWS 雲端。您負責維護在此基礎設施上託管內容的控制權。您也同時負責所使用 AWS 服務的安全組態和管理任務。如需資料隱私權的詳細資訊，請參閱[資料隱私權常見問答集](#)。如需有關歐洲資料保護的相關資訊，請參閱AWS 安全性部落格上的[AWS 共同責任模型和 GDPR](#) 部落格文章。

基於資料保護目的，我們建議您保護 AWS 帳戶 登入資料，並使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 設定個別使用者。如此一來，每個使用者都只會獲得授與完成其任務所必須的許可。我們也建議您採用下列方式保護資料：

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源通訊。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 使用 設定 API 和使用者活動記錄 AWS CloudTrail。如需有關使用 CloudTrail 追蹤擷取 AWS 活動的資訊，請參閱AWS CloudTrail 《使用者指南》中的[使用 CloudTrail 追蹤](#)。
- 使用 AWS 加密解決方案，以及其中的所有預設安全控制 AWS 服務。
- 使用進階的受管安全服務 (例如 Amazon Macie)，協助探索和保護儲存在 Amazon S3 的敏感資料。
- 如果您在 AWS 透過命令列界面或 API 存取 時需要 FIPS 140-3 驗證的密碼編譯模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱[聯邦資訊處理標準 \(FIPS\) 140-3](#)。

我們強烈建議您絕對不要將客戶的電子郵件地址等機密或敏感資訊，放在標籤或自由格式的文字欄位中，例如名稱欄位。這包括當您使用 MediaStore 或使用 AWS 服務 主控台、API AWS CLI或其他 AWS SDKs 時。您在標籤或自由格式文字欄位中輸入的任何資料都可能用於計費或診斷日誌。如果您提供外部伺服器的 URL，我們強烈建議請勿在驗證您對該伺服器請求的 URL 中包含憑證資訊。

資料加密

MediaStore 使用業界標準的 AES-256 演算法加密靜態容器和物件。我們建議您使用 MediaStore 以下列方式保護您的資料：

- 建立容器政策，以控制該容器中所有資料夾和物件的存取權。如需詳細資訊，請參閱[the section called “容器政策”](#)。
- 建立跨來源資源共用 (CORS) 政策，以選擇性地允許跨來源存取您的 MediaStore 資源。您可以使用 CORS，允許在一個網域載入的用戶端 Web 應用程式與不同網域中的資源互動。如需詳細資訊，請參閱[the section called “CORS 政策”](#)。

AWS Elemental MediaStore 的 Identity and Access Management

AWS Identity and Access Management (IAM) 是 AWS 服務，可協助管理員安全地控制對 AWS 資源的存取。IAM 管理員可控制誰可以進行身分驗證（登入）和授權（具有許可），以使用 MediaStore 資源。IAM 是您可以免費使用 AWS 服務的。

主題

- [目標對象](#)
- [使用身分驗證](#)
- [使用政策管理存取權](#)
- [AWS Elemental MediaStore 如何與 IAM 搭配使用](#)
- [AWS Elemental MediaStore 的身分型政策範例](#)
- [故障診斷 AWS Elemental MediaStore 身分和存取](#)

目標對象

如何使用 AWS Identity and Access Management (IAM) 會因您的角色而有所不同：

- 服務使用者 - 如果您無法存取功能，請向系統管理員請求許可權 (請參閱 [故障診斷 AWS Elemental MediaStore 身分和存取](#))
- 服務管理員 - 判斷使用者存取權限，並提交許可權請求 (請參閱 [AWS Elemental MediaStore 如何與 IAM 搭配使用](#))
- IAM 管理員 - 撰寫政策以管理存取權 (請參閱 [AWS Elemental MediaStore 的身分型政策範例](#))

使用身分驗證

身分驗證是您 AWS 使用身分憑證登入的方式。您必須驗證為 AWS 帳戶根使用者、IAM 使用者或擔任 IAM 角色。

您可以使用身分來源的登入資料，例如 AWS IAM Identity Center (IAM Identity Center)、單一登入身分驗證或 Google/Facebook 登入資料，以聯合身分的形式登入。如需有關登入的詳細資訊，請參閱《AWS 登入 使用者指南》中的[如何登入您的 AWS 帳戶](#)。

對於程式設計存取，AWS 提供 SDK 和 CLI 以密碼編譯方式簽署請求。如需詳細資訊，請參閱《IAM 使用者指南》中的[API 請求的AWS 第 4 版簽署程序](#)。

AWS 帳戶 根使用者

當您建立時 AWS 帳戶，您會從一個名為 AWS 帳戶 theroot 使用者的登入身分開始，該身分具有對所有 AWS 服務和資源的完整存取權。強烈建議您不要以根使用者處理日常任務。有關需要根使用者憑證的任務，請參閱《IAM 使用者指南》中的[需要根使用者憑證的任務](#)。

聯合身分

最佳實務是要求人類使用者使用聯合身分提供者，以 AWS 服務 使用臨時憑證存取。

聯合身分是您企業目錄、Web 身分提供者的使用者，或是 AWS 服務 使用身分來源的憑證 Directory Service 存取的使用者。聯合身分會擔任提供臨時憑證的角色。

對於集中式存取權管理，我們建議使用 AWS IAM Identity Center。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的[什麼是 IAM Identity Center？](#)。

IAM 使用者和群組

[IAM 使用者](#)是一種身分具備單人或應用程式的特定許可權。我們建議使用臨時憑證，而不是具有長期使用權憑證的 IAM 使用者。如需詳細資訊，請參閱《IAM 使用者指南》中的[要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取](#)。

[IAM 群組](#)會指定 IAM 使用者集合，使管理大量使用者的許可權更加輕鬆。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 使用者的使用案例](#)。

IAM 角色

[IAM 角色](#)的身分具有特定許可權，其可以提供臨時憑證。您可以透過[從使用者切換到 IAM 角色（主控台）](#)或呼叫 AWS CLI 或 AWS API 操作來擔任角色。如需詳細資訊，請參閱《IAM 使用者指南》中的[擔任角色的方法](#)。

IAM 角色適用於聯合身分使用者存取、臨時 IAM 使用者許可、跨帳戶存取權、跨服務存取，以及在 Amazon EC2 上執行應用程式。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 中的快帳戶資源存取](#)。

使用政策管理存取權

您可以透過建立政策並將其連接到身分或資源 AWS 來控制 AWS 中的存取。政策定義與身分或資源相關聯的許可。當委託人提出請求時 AWS，會評估這些政策。大多數政策會以 JSON 文件 AWS 的形式存放在 中。如需有關 JSON 政策文件的詳細資訊，請參閱《IAM 使用者指南》中的[JSON 政策概觀](#)。

使用政策時，管理員會定義哪些主體可以對哪些資源執行動作，以及在哪些條件下執行動作，藉此指定誰可以存取哪些內容。

預設情況下，使用者和角色沒有許可。IAM 管理員會建立 IAM 政策，並將其新增至角色，然後使用者就可以擔任該角色。IAM 政策會定義該動作的許可條件，但與使用何種方法進行操作無關。

身分型政策

身分型政策是可以連接身分 (使用者、群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。如需了解如何建立身分型政策，請參閱 IAM 使用者指南中的[透過客戶管理政策定義自訂 IAM 許可](#)。

身分型政策可以是內嵌政策 (直接嵌入單一身分) 或受管政策 (連接多個身分的獨立政策)。如需了解如何在受管政策及內嵌政策之間做選擇，請參閱《IAM 使用者指南》中的[在受管政策與內嵌政策之間選擇](#)。

資源型政策

資源型政策是附加到資源的 JSON 政策文件。範例包括 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。您必須在資源型政策中[指定主體](#)。

資源型政策是位於該服務中的內嵌政策。您無法在資源型政策中使用來自 IAM 的 AWS 受管政策。

其他政策類型

AWS 支援其他政策類型，可設定更多常見政策類型授予的最大許可：

- 許可界限 – 設定身分型政策可授予 IAM 實體的最大許可權。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 實體許可界限](#)。
- 服務控制政策 (SCP) – 指定 AWS Organizations 中的組織或組織單位的最大許可權。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的[服務控制政策](#)。
- 資源控制政策 (RCP) – 定義組織中資源可用的最大許可權。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的[資源控制政策 \(RCP\)](#)。
- 工作階段政策 – 這是一種在為角色或聯合身分使用者建立臨時工作階段時，以參數形式傳遞的進階政策。如需詳細資訊，請參閱《IAM 使用者指南》中的[工作階段政策](#)。

多種政策類型

當多種類型的政策適用於請求時，產生的許可會更複雜而無法理解。若要了解如何 AWS 在涉及多個政策類型時決定是否允許請求，請參閱《IAM 使用者指南》中的[政策評估邏輯](#)。

AWS Elemental MediaStore 如何與 IAM 搭配使用

使用 IAM 管理 MediaStore 的存取權之前，請先了解哪些 IAM 功能可與 MediaStore 搭配使用。

您可以搭配 AWS Elemental MediaStore 使用的 IAM 功能

IAM 功能	MediaStore 支援
身分型政策	是
資源型政策	是
政策動作	是
政策資源	是
政策條件索引鍵 (服務特定)	是
ACL	否
ABAC(政策中的標籤)	部分
臨時憑證	是
主體許可	是
服務角色	是
服務連結角色	否

若要全面了解 MediaStore 和其他 AWS 服務如何與大多數 IAM 功能搭配使用，請參閱 [《AWS IAM 使用者指南》](#) 中的與 IAM 搭配使用的 [服務](#)。

MediaStore 的身分型政策

支援身分型政策：是

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。如需了解如何建立身分型政策，請參閱 IAM 使用者指南中的 [透過客戶管理政策定義自訂 IAM 許可](#)。

使用 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，以及在何種條件下允許或拒絕動作。如要了解您在 JSON 政策中使用的所有元素，請參閱 [《IAM 使用者指南》](#) 中的 [IAM JSON 政策元素參考](#)。

MediaStore 的身分型政策範例

若要檢視 MediaStore 身分型政策的範例，請參閱 [AWS Elemental MediaStore 的身分型政策範例](#)。

MediaStore 中的資源型政策

支援資源型政策：是

資源型政策是附加到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中[指定主體](#)。委託人可以包含帳戶、使用者、角色、聯合身分使用者或 AWS 服務。

如需啟用跨帳戶存取權，您可以在其他帳戶內指定所有帳戶或 IAM 實體作為資源型政策的主體。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的快帳戶資源存取](#)。

Note

MediaStore 也支援容器政策，可定義哪些委託人實體（帳戶、使用者、角色和聯合身分使用者）可以在容器上執行動作。如需詳細資訊，請參閱[容器政策](#)。

MediaStore 的政策動作

支援政策動作：是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

JSON 政策的 Action 元素描述您可以用來允許或拒絕政策中存取的動作。政策會使用動作來授予執行相關聯動作的許可。

若要查看 MediaStore 動作清單，請參閱《服務授權參考》中的 [AWS Elemental MediaStore 定義的動作](#)。

MediaStore 中的政策動作在動作之前使用以下字首：

```
mediastore
```

若要在單一陳述式中指定多個動作，請用逗號分隔。

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

若要檢視 MediaStore 身分型政策的範例，請參閱 [AWS Elemental MediaStore 的身分型政策範例](#)。

MediaStore 的政策資源

支援政策資源：是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Resource JSON 政策元素可指定要套用動作的物件。最佳實務是使用其 [Amazon Resource Name \(ARN\)](#) 來指定資源。對於不支援資源層級許可的動作，使用萬用字元 (*) 表示陳述式適用於所有資源。

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

若要查看 MediaStore 資源類型及其 ARNs，請參閱《服務授權參考》中的 [AWS Elemental MediaStore 定義的資源](#)。若要了解您可以使用哪些動作指定每個資源的 ARN，請參閱 [AWS Elemental MediaStore 定義的動作](#)。

MediaStore 容器資源具有下列 ARN：

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

如需 ARNs 格式的詳細資訊，請參閱 [Amazon Resource Name \(ARNs\) AWS 和服務命名空間](#)。

例如，若要在您的陳述式中指定 AwardsShow 容器，請使用以下 ARN：

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

MediaStore 的政策條件索引鍵

支援服務特定政策條件金鑰：是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Condition 元素指定何時根據定義的條件執行陳述式。您可以建立使用[條件運算子](#)的條件運算式 (例如等於或小於)，來比對政策中的條件和請求中的值。若要查看所有 AWS 全域條件索引鍵，請參閱《IAM 使用者指南》中的[AWS 全域條件內容索引鍵](#)。

若要查看 MediaStore 條件索引鍵的清單，請參閱《服務授權參考》中的 [AWS Elemental MediaStore 的條件索引鍵](#)。若要了解您可以使用條件金鑰的動作和資源，請參閱 [AWS Elemental MediaStore 定義的動作](#)。

若要檢視 MediaStore 身分型政策的範例，請參閱 [AWS Elemental MediaStore 的身分型政策範例](#)。

MediaStore ACLs

支援 ACL：否

存取控制清單 (ACL) 可控制哪些主體 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

ABAC 搭配 MediaStore

支援 ABAC (政策中的標籤)：部分

屬性型存取控制 (ABAC) 是一種授權策略，根據稱為標籤的屬性定義許可權。您可以將標籤連接至 IAM 實體 AWS 和資源，然後設計 ABAC 政策，以便在委託人的標籤符合資源上的標籤時允許操作。

如需根據標籤控制存取，請使用 `aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 條件索引鍵，在政策的[條件元素](#)中，提供標籤資訊。

如果服務支援每個資源類型的全部三個條件金鑰，則對該服務而言，值為 Yes。如果服務僅支援某些資源類型的全部三個條件金鑰，則值為 Partial。

如需 ABAC 的詳細資訊，請參閱 IAM 使用者指南中的[使用 ABAC 授權定義許可](#)。如要查看含有設定 ABAC 步驟的教學課程，請參閱《IAM 使用者指南》中的[使用屬性型存取控制 \(ABAC\)](#)。

搭配 MediaStore 使用臨時登入資料

支援臨時憑證：是

臨時登入資料提供 AWS 資源的短期存取權，當您使用聯合或切換角色時，會自動建立。AWS 建議您動態產生臨時登入資料，而不是使用長期存取金鑰。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 中的臨時安全憑證](#)，以及[使用 IAM 的 AWS 服務](#)。

MediaStore 的跨服務主體許可

支援轉寄存取工作階段 (FAS)：是

轉送存取工作階段 (FAS) 使用呼叫的委託人許可 AWS 服務，並結合 AWS 服務請求向下游服務提出請求。如需提出 FAS 請求時的政策詳細資訊，請參閱[轉發存取工作階段](#)。

MediaStore 的服務角色

支援服務角色：是

服務角色是服務擔任的 [IAM 角色](#)，可代您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的[建立角色以委派許可權給 AWS 服務](#)。

Warning

變更服務角色的許可可能會中斷 MediaStore 功能。只有在 MediaStore 提供指引時，才能編輯服務角色。

MediaStore 的服務連結角色

支援服務連結角色：否

服務連結角色是連結至的一種服務角色 AWS 服務。服務可以擔任代表您執行動作的角色。服務連結角色會出現在您的 AWS 帳戶中，並由服務擁有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

如需建立或管理服务連結角色的詳細資訊，請參閱[可搭配 IAM 運作的 AWS 服務](#)。在資料表中尋找服務，其中包含服務連結角色欄中的 Yes。選擇是連結，以檢視該服務的服務連結角色文件。

AWS Elemental MediaStore 的身分型政策範例

根據預設，使用者和角色沒有建立或修改 MediaStore 資源的許可。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。

如需了解如何使用這些範例 JSON 政策文件建立 IAM 身分型政策，請參閱《IAM 使用者指南》中的[建立 IAM 政策 \(主控台\)](#)。

如需 MediaStore 定義之動作和資源類型的詳細資訊，包括每種資源類型的 ARNs 格式，請參閱《服務授權參考》中的[AWS Elemental MediaStore 的動作、資源和條件索引鍵](#)。

主題

- [政策最佳實務](#)
- [使用 MediaStore 主控台](#)
- [允許使用者檢視他們自己的許可](#)

政策最佳實務

身分型政策會判斷您帳戶中的某個人員是否可以建立、存取或刪除 MediaStore 資源。這些動作可能會讓您的 AWS 帳戶產生費用。當您建立或編輯身分型政策時，請遵循下列準則及建議事項：

- 開始使用 AWS 受管政策並邁向最低權限許可 – 若要開始將許可授予您的使用者和工作負載，請使用將許可授予許多常見使用案例的 AWS 受管政策。它們可在您的 中使用 AWS 帳戶。我們建議您定義特定於使用案例 AWS 的客戶受管政策，以進一步減少許可。如需更多資訊，請參閱《IAM 使用者指南》中的 [AWS 受管政策](#)或[任務職能的AWS 受管政策](#)。
- 套用最低權限許可 – 設定 IAM 政策的許可時，請僅授予執行任務所需的許可。為實現此目的，您可以定義在特定條件下可以對特定資源採取的動作，這也稱為最低權限許可。如需使用 IAM 套用許可的更多相關資訊，請參閱《IAM 使用者指南》中的 [IAM 中的政策和許可](#)。
- 使用 IAM 政策中的條件進一步限制存取權 – 您可以將條件新增至政策，以限制動作和資源的存取。例如，您可以撰寫政策條件，指定必須使用 SSL 傳送所有請求。如果透過特定 等使用服務動作 AWS 服務，您也可以使用條件來授予其存取權 CloudFormation。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM JSON 政策元素：條件](#)。
- 使用 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作 – IAM Access Analyzer 驗證新政策和現有政策，確保這些政策遵從 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您撰寫安全且實用的政策。如需詳細資訊，請參閱 IAM 使用者指南中的 [使用 IAM Access Analyzer 驗證政策](#)。
- 需要多重要素驗證 (MFA) – 如果您的案例需要 IAM 使用者或 中的根使用者 AWS 帳戶，請開啟 MFA 以提高安全性。如需在呼叫 API 操作時請求 MFA，請將 MFA 條件新增至您的政策。如需詳細資訊，請參閱 IAM 使用者指南中的 [透過 MFA 的安全 API 存取](#)。

如需 IAM 中最佳實務的相關資訊，請參閱《IAM 使用者指南》中的 [IAM 安全最佳實務](#)。

使用 MediaStore 主控台

若要存取 AWS Elemental MediaStore 主控台，您必須擁有一組最低許可。這些許可必須允許您列出和檢視 中 MediaStore 資源的詳細資訊 AWS 帳戶。如果您建立比最基本必要許可更嚴格的身分型政策，則對於具有該政策的實體 (使用者或角色) 而言，主控台就無法如預期運作。

對於僅呼叫 AWS CLI 或 AWS API 的使用者，您不需要允許最低主控台許可。反之，只需允許存取符合他們嘗試執行之 API 操作的動作就可以了。

為了確保使用者和角色仍然可以使用 MediaStore 主控台，請將 MediaStore *ConsoleAccess* 或 *ReadOnly* AWS 受管政策連接到實體。如需詳細資訊，請參閱《IAM 使用者指南》中的[新增許可到使用者](#)。

允許使用者檢視他們自己的許可

此範例會示範如何建立政策，允許 IAM 使用者檢視附加到他們使用者身分的內嵌及受管政策。此政策包含在主控台或使用 或 AWS CLI AWS API 以程式設計方式完成此動作的許可。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupForUser",
        "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": "*"
    }
  ]
}
```

```
}
```

故障診斷 AWS Elemental MediaStore 身分和存取

使用以下資訊來協助您診斷和修正使用 MediaStore 和 IAM 時可能遇到的常見問題。

主題

- [我無權在 MediaStore 中執行動作](#)
- [我未獲得執行 iam:PassRole 的授權](#)
- [我想要允許 以外的人員 AWS 帳戶 存取我的 MediaStore 資源](#)

我無權在 MediaStore 中執行動作

如果您收到錯誤，告知您未獲授權執行動作，您的政策必須更新，允許您執行動作。

下列範例錯誤會在 mateojackson IAM 使用者嘗試使用主控台檢視一個虛構 *my-example-widget* 資源的詳細資訊，但卻無虛構 *mediastore:GetWidget* 許可時發生。

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

在此情況下，必須更新 mateojackson 使用者的政策，允許使用 *mediastore:GetWidget* 動作存取 *my-example-widget* 資源。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我未獲得執行 iam:PassRole 的授權

如果您收到錯誤，告知您無權執行 *iam:PassRole* 動作，您的政策必須更新，以允許您將角色傳遞至 MediaStore。

有些 AWS 服務可讓您將現有角色傳遞給該服務，而不是建立新的服務角色或服務連結角色。如需執行此作業，您必須擁有將角色傳遞至該服務的許可。

當名為 *marymajor* 的 IAM 使用者嘗試使用主控台在 MediaStore 中執行動作時，會發生下列範例錯誤。但是，動作請求服務具備服務角色授予的許可。Mary 沒有將角色傳遞給服務的許可。

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

在這種情況下，Mary 的政策必須更新，允許她執行 `iam:PassRole` 動作。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我想要允許以外的人員 AWS 帳戶 存取我的 MediaStore 資源

您可以建立一個角色，讓其他帳戶中的使用者或您組織外部的人員存取您的資源。您可以指定要允許哪些信任物件取得該角色。針對支援基於資源的政策或存取控制清單 (ACL) 的服務，您可以使用那些政策來授予人員存取您的資源的許可。

如需進一步了解，請參閱以下內容：

- 若要了解 MediaStore 是否支援這些功能，請參閱 [AWS Elemental MediaStore 如何與 IAM 搭配使用](#)。
- 若要了解如何 AWS 帳戶 在您擁有的 資源間提供存取權，請參閱 [《IAM 使用者指南》中的在您擁有 AWS 帳戶 的另一個 IAM 使用者中提供存取權](#)。
- 若要了解如何將資源的存取權提供給第三方 AWS 帳戶，請參閱 [《IAM 使用者指南》中的將存取權提供給第三方 AWS 帳戶 擁有](#)。
- 如需了解如何透過聯合身分提供存取權，請參閱 [《IAM 使用者指南》中的將存取權提供給在外部進行身分驗證的使用者 \(聯合身分\)](#)。
- 如需了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱 [《IAM 使用者指南》中的 IAM 中的跨帳戶資源存取](#)。

在 中記錄和監控 AWS Elemental MediaStore

本章節會概述 AWS Elemental MediaStore 中的記錄和監控選項，可讓您用來保障安全。如需在 MediaStore 中記錄和監控的詳細資訊，請參閱 [在 AWS Elemental MediaStore 中監控和標記](#)。

監控是維護和 AWS 解決方案的可靠性、可用性 AWS Elemental MediaStore 和效能的重要部分。您應該從 AWS 解決方案的所有部分收集監控資料，以便在發生多點失敗時更輕鬆地偵錯。AWS 提供數種工具來監控 MediaStore 資源並回應潛在事件。

Amazon CloudWatch 警示

您可以使用 CloudWatch 警示觀察單一指標一段指定的時間。如果指標超過指定的閾值，會傳送一則通知至 Amazon SNS 主題或 AWS Auto Scaling 政策。CloudWatch 警示不會因為它們處於特定狀態而叫用動作。必須是狀態已變更並維持了所指定的時間長度，才會呼叫動作。如需詳細資訊，請參閱[使用 CloudWatch 進行監控](#)。

AWS CloudTrail 日誌

CloudTrail 提供使用者、角色或 AWS 服務在 中採取之動作的記錄 AWS Elemental MediaStore。您可以使用 CloudTrail 所收集的資訊，判斷向 MediaStore 提出的請求、提出請求的 IP 地址、提出請求的人員、提出請求的時間，以及其他詳細資訊。如需詳細資訊，請參閱[使用 CloudTrail 記錄 API 呼叫](#)。

AWS Trusted Advisor

Trusted Advisor 利用從為數十萬 AWS 客戶提供服務的最佳實務。會 Trusted Advisor 檢查您的 AWS 環境，然後在有機會節省成本、改善系統可用性和效能，或協助填補安全漏洞時提出建議。所有 AWS 客戶都可以存取五個 Trusted Advisor 檢查。擁有商業或企業支援計劃的客戶可以檢視所有 Trusted Advisor 檢查。

如需詳細資訊，請參閱[AWS Trusted Advisor](#)。

AWS Elemental MediaStore 的合規驗證

若要了解 是否 AWS 服務 在特定合規計劃範圍內，請參閱[AWS 服務 合規計劃範圍內](#)然後選擇您感興趣的合規計劃。如需一般資訊，請參閱[AWS 合規計劃](#)。

您可以使用 下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱[下載報告 in AWS Artifact](#)

您使用 時的合規責任 AWS 服務 取決於資料的敏感度、您公司的合規目標，以及適用的法律和法規。如需使用 時合規責任的詳細資訊 AWS 服務，請參閱 [AWS 安全文件](#)。

AWS Elemental MediaStore 中的彈性

AWS 全球基礎設施是以 AWS 區域 和 可用區域為基礎建置。AWS 區域 提供多個實體隔離和隔離的可用區域，這些可用區域與低延遲、高輸送量和高度備援聯網連接。透過可用區域，您可以設計與操作的應用程式和資料庫，在可用區域之間自動容錯移轉而不會發生中斷。可用區域的可用性、容錯能力和擴展能力，均較單一或多個資料中心的傳統基礎設施還高。

如需 AWS 區域 和可用區域的詳細資訊，請參閱 [AWS 全球基礎設施](#)。

除了 AWS 全球基礎設施之外，MediaStore 還提供數種功能，以協助支援您的資料彈性和備份需求。

AWS Elemental MediaStore 中的基礎設施安全

作為受管服務，AWS Elemental MediaStore 受到 AWS 全球網路安全的保護。如需 AWS 安全服務以及 如何 AWS 保護基礎設施的相關資訊，請參閱[AWS 雲端安全](#)。若要使用基礎設施安全的最佳實務來設計您的 AWS 環境，請參閱安全支柱 AWS Well-Architected Framework 中的[基礎設施保護](#)。

您可以使用 AWS 發佈的 API 呼叫，透過網路存取 MediaStore。使用者端必須支援下列專案：

- Transport Layer Security (TLS)。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 具備完美轉送私密(PFS)的密碼套件，例如 DHE (Ephemeral Diffie-Hellman)或 ECDHE (Elliptic Curve Ephemeral Diffie-Hellman)。現代系統(如 Java 7 和更新版本)大多會支援這些模式。

預防跨服務混淆代理人

混淆代理人問題屬於安全性問題，其中沒有執行動作許可的實體可以強制具有更多許可的實體執行該動作。在中 AWS，跨服務模擬可能會導致混淆代理人問題。在某個服務(呼叫服務)呼叫另一個服務(被呼叫服務)時，可能會發生跨服務模擬。可以操縱呼叫服務來使用其許可，以其不應有存取許可的方式對其他客戶的資源採取動作。為了預防這種情況，AWS 提供的工具可協助您保護所有服務的資料，而這些服務主體已獲得您帳戶中資源的存取權。

我們建議在資源政策中使用 [aws:SourceArn](#)和 [aws:SourceAccount](#)全域條件內容金鑰，以限制 AWS Elemental MediaStore 為資源提供其他服務的許可。如果您想要僅允許一個資源與跨服務存取相關聯，則請使用 `aws:SourceArn`。如果您想要允許該帳戶中的任何資源與跨服務使用相關聯，請使用 `aws:SourceAccount`。

防範混淆代理人問題的最有效方法是使用 `aws:SourceArn` 全域條件內容索引鍵，以及資源的完整 ARN。如果不知道資源的完整 ARN，或者如果您指定了多個資源，請使用 `aws:SourceArn` 全域內容條件索引鍵搭配萬用字元 (*) 來表示 ARN 的未知部分。例如 `arn:aws:servicename:*:123456789012:*`。

如果 `aws:SourceArn` 值不包含帳戶 ID (例如 Amazon S3 儲存貯體 ARN)，您必須使用這兩個全域條件內容索引鍵來限制許可。

的值`aws:SourceArn`必須是 MediaStore 在您的區域和帳戶中發佈 CloudWatch 日誌的組態。

下列範例顯示如何使用 MediaStore 中的 `aws:SourceArn` 和 `aws:SourceAccount` 全域條件內容索引鍵，以防止混淆代理人問題。

JSON

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

在 AWS Elemental MediaStore 中監控和標記

監控是維護 AWS Elemental MediaStore 及其他 AWS 解決方案可靠性、可用性和效能的重要部分。AWS 提供下列監控工具來監看 MediaStore、在發生錯誤時回報，並適時採取自動動作：

- AWS CloudTrail 會擷取由您的帳戶或代表 AWS 您的帳戶發出的 API 呼叫和相關事件，並將日誌檔案交付至您指定的 Amazon S3 儲存貯體。您可以識別呼叫的使用者和帳戶 AWS、進行呼叫的來源 IP 地址，以及呼叫的時間。如需詳細資訊，請參閱 [「AWS CloudTrail 使用者指南」](#)。
- Amazon CloudWatch AWS 會即時監控您的 AWS 資源和您在 上執行的應用程式。您可以收集和追蹤指標、建立自訂儀板表，以及設定警示，在特定指標達到您指定的閾值時通知您或採取動作。例如，您可以讓 CloudWatch 追蹤 CPU 使用量或其他 Amazon EC2 執行個體指標，並在需要時自動啟動新的執行個體。如需詳細資訊，請參閱 [Amazon CloudWatch 使用者指南](#)。
- Amazon CloudWatch Events 提供描述 AWS 資源變更的系統事件串流。一般而言，AWS 服務會在幾秒鐘內將事件通知傳遞至 CloudWatch Events，但有時可能需要一分鐘或更長的時間。CloudWatch Events 可啟用自動事件驅動型運算，因為您可以撰寫規則來監控特定事件，並在發生這些事件時觸發 AWS 其他服務中的自動動作。如需詳細資訊，請參閱 [Amazon CloudWatch Events 使用者指南](#)。
- Amazon CloudWatch Logs 可讓您監控、存放和存取來自 Amazon EC2 執行個體、CloudTrail 及其他來源的日誌檔案。CloudWatch Logs 可監控日誌檔案中的資訊，並在達到特定閾值時通知您。您也可以將日誌資料存檔在高耐用性的儲存空間。如需詳細資訊，請參閱 [Amazon CloudWatch Logs 使用者指南](#)。

您也可以以標籤形式將中繼資料指派給 MediaStore 容器。每個標籤都包含您定義的金鑰和值。標籤可讓您更輕鬆地管理、搜尋和篩選資源。您可以使用標籤在 AWS 管理主控台中組織 AWS 資源、建立所有資源的 AWS 用量和帳單報告，以及在基礎設施自動化活動期間篩選資源。

主題

- [使用 記錄 AWS Elemental MediaStore API 呼叫 AWS CloudTrail](#)
- [使用 Amazon CloudWatch 監控 AWS Elemental MediaStore](#)
- [標記 AWS Elemental MediaStore 資源](#)

使用 記錄 AWS Elemental MediaStore API 呼叫 AWS CloudTrail

AWS Elemental MediaStore 已與 服務整合 AWS CloudTrail，此服務提供由使用者、角色或 MediaStore 中的 AWS 服務所採取之動作的記錄。CloudTrail 會將 MediaStore 的 API 呼叫子集擷取為

事件，包括來自 MediaStore 主控台的呼叫，以及來自對 MediaStore API 的程式碼呼叫。如果您建立線索，則可以將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括 MediaStore 的事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台的事件歷史記錄檢視最新事件。使用 CloudTrail 收集的資訊，您可以判斷向 MediaStore 提出的請求、提出請求的 IP 地址、提出請求的人員、提出請求的時間等等。

若要進一步了解 CloudTrail，包括如何設定及啟用，請參閱 [《AWS CloudTrail 使用者指南》](#)。

主題

- [CloudTrail 中的 AWS Elemental MediaStore 資訊](#)
- [範例：AWS Elemental MediaStore 日誌檔案項目](#)

CloudTrail 中的 AWS Elemental MediaStore 資訊

當您建立 AWS 帳戶時，會在您的帳戶上啟用 CloudTrail。當 AWS Elemental MediaStore 中發生支援的事件活動時，該活動會與事件歷史記錄中的其他 AWS 服務事件一起記錄在 CloudTrail 事件中。您可以在 AWS 帳戶中檢視、搜尋和下載最近的事件。如需詳細資訊，請參閱《使用 CloudTrail 事件歷史記錄檢視事件》<https://docs.aws.amazon.com/awscloudtrail/latest/userguide/view-cloudtrail-events.html>。

若要持續記錄您 AWS 帳戶中的事件，包括 MediaStore 的事件，請建立追蹤。線索能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。根據預設，當您在主控台建立線索時，線索會套用到所有 AWS 區域。線索會記錄 AWS 分割區中所有區域的事件，並將日誌檔案傳送到您指定的 Amazon S3 儲存貯體。此外，您可以設定其他 AWS 服務，以進一步分析和處理 CloudTrail 日誌中所收集的事件資料。如需詳細資訊，請參閱下列主題：

- [建立追蹤的概觀](#)
- [CloudTrail 支援的服務和整合](#)
- [設定 CloudTrail 的 Amazon SNS 通知](#)
- [從多個區域接收 CloudTrail 日誌檔案，以及從多個帳戶接收 CloudTrail 日誌檔案](#)

AWS Elemental MediaStore 支援將下列操作記錄為 CloudTrail 日誌檔案中的事件：

- [CreateContainer](#)
- [DeleteContainer](#)
- [DeleteContainerPolicy](#)

- [DeleteCorsPolicy](#)
- [DescribeContainer](#)
- [GetContainerPolicy](#)
- [GetCorsPolicy](#)
- [ListContainers](#)
- [PutContainerPolicy](#)
- [PutCorsPolicy](#)

每一筆事件或日誌專案都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 是否使用根使用者或使用者登入資料提出請求
- 提出該請求時，是否使用了特定角色或聯合身分使用者的臨時安全憑證
- 請求是否由其他 AWS 服務提出

如需詳細資訊，請參閱 [CloudTrail userIdentity 元素](#)。

範例：AWS Elemental MediaStore 日誌檔案項目

追蹤是一種組態，能讓事件以日誌檔案的形式交付到您指定的 Amazon S3 儲存貯體。CloudTrail 日誌檔案包含一或多個日誌專案。一個事件為任何來源提出的單一請求，並包含請求動作、請求的日期和時間、請求參數等資訊。CloudTrail 日誌檔案並非依公有 API 呼叫追蹤記錄的堆疊排序，因此不會以任何特定順序出現。

以下範例顯示的 CloudTrail 日誌項目會示範 CreateContainer 操作：

```
{
  "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":
    "MNCTGH4HRQJ27GRMBVDPIVHEP4L02BN6MUVHBCPSH0AWNS0KSXC024B2UE0BBND5D0NRXTMFK3TOJ4G7AHWMESI",
    "eventID": "7085b140-fb2c-409b-a329-f567912d704c",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}

```

使用 Amazon CloudWatch 監控 AWS Elemental MediaStore

您可以使用 CloudWatch 監控 AWS Elemental MediaStore，這會收集原始資料並將其處理為可讀指標。CloudWatch 會保留統計資料 15 個月，讓您可以存取歷史資訊，並更清楚 Web 應用程式或服務的效能。您也可以設定留意特定閾值的警示，當滿足這些閾值時傳送通知或採取動作。如需詳細資訊，請參閱 [Amazon CloudWatch 使用者指南](#)。

AWS 提供下列監控工具來監看 MediaStore、在發生錯誤時回報，以及適時採取自動動作：

- Amazon CloudWatch Logs 可讓您監控、存放和存取 AWS Elemental MediaStore 等服務的日誌檔案 AWS。您可以使用 CloudWatch Logs 來監控使用日誌資料的應用程式和系統。例

如，CloudWatch Logs 可以追蹤應用程式日誌中發生的錯誤數量，並在錯誤率超過您指定的閾值時傳送通知給您。CloudWatch Logs 使用您的日誌資料進行監控，因此不需要變更程式碼。例如，您可以監控應用程式日誌是否有特定的文字用語（例如「ValidationException」）或計數在特定時間期限所做的 PutObject 請求數量。找到您要搜尋的詞彙時，CloudWatch Logs 會將資料報告至您指定的 CloudWatch 指標。日誌資料會在移轉和靜態時加密。

- Amazon CloudWatch Events 提供描述 AWS 資源變更的系統事件，例如 MediaStore 物件。一般而言，AWS 服務會在幾秒鐘內將事件通知傳遞至 CloudWatch Events，但有時可能需要一分鐘或更長的時間。您可以設定規則來比對事件（例如 DeleteObject 請求），並將其路由至一或多個目標函數或串流。CloudWatch Events 在操作變更時會查覺到。此外，CloudWatch Events 會回應這些操作變更，並視需要採取修正動作，方法是傳送訊息來回應環境、啟用函數、進行變更，以及擷取狀態資訊。

CloudWatch Logs

存取記錄會為對容器中的物件所做的請求提供詳細的記錄。存取記錄對於許多應用程式都相當實用，例如安全和存取稽核。他們也可以協助您了解您的客戶群，並了解您的 MediaStore 帳單。CloudWatch Logs 分類如下：

- 日誌串流是共享相同來源的一系列日誌事件。
- 日誌群組是共享相同保留、監控和存取控制設定的日誌串流群組。當您在容器上啟用存取記錄時，MediaStore 會建立名稱為 `aws/mediastore/MyContainerName` 的日誌群組。您可以定義日誌群組，並指定放入每個群組的串流。可以屬於一個日誌群組的日誌串流數量並沒有配額。

根據預設，日誌將無限期保留且永遠不會過期。您可以調整每個日誌群組的保留政策，維持無限期保留，或選擇 1 天至 10 年之間的保留期間。

設定 Amazon CloudWatch 的許可

使用 AWS Identity and Access Management (IAM) 建立角色，讓 AWS Elemental MediaStore 存取 Amazon CloudWatch。您必須執行這些步驟，才能為您的帳戶發佈 CloudWatch Logs。CloudWatch 會自動發佈您帳戶的指標。

允許 MediaStore 存取 CloudWatch

1. 前往 <https://console.aws.amazon.com/iam/> 開啟 IAM 主控台。
2. 在 IAM 主控台的導覽窗格中，選擇 Policies (政策)，接著選擇 Create Policy (建立政策)。
3. 選擇 JSON 標籤，並將下列政策貼上：

JSON

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

此政策允許 MediaStore 為您 AWS 帳戶內任何區域中的任何容器建立日誌群組和日誌串流。

4. 選擇檢閱政策。
5. 在 Review policy (檢閱政策) 頁面上，針對 Name (名稱)，輸入 **MediaStoreAccessLogsPolicy**，然後選擇 Create policy (建立政策)。
6. 在 IAM 主控台的導覽窗格中，選擇角色，然後選擇建立角色。
7. 選擇 Another AWS account (另一個 AWS 帳戶) 角色類型。
8. 針對帳戶 ID，輸入 AWS 您的帳戶 ID。
9. 選擇下一步：許可。
10. 在搜尋方塊中，輸入 **MediaStoreAccessLogsPolicy**。
11. 選取新政策旁的核取方塊，然後選擇 Next: Tags (下一步：標籤)。
12. 選擇 Next: Review (下一步：檢閱) 來預覽新使用者。

13. 針對角色名稱，輸入 **MediaStoreAccessLogs**，然後選擇建立角色。
14. 在確認訊息中，選擇您剛建立的角色名稱 (**MediaStoreAccessLogs**)。
15. 在角色的 Summary (摘要) 頁面上，選擇 Trust relationships (信任關係) 標籤。
16. 選擇編輯信任關係。
17. 在政策文件中，將主體變更為 MediaStore 服務。它應該如下所示：

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

整個政策看起來應該如下所示：

JSON

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

18. 選擇 Update Trust Policy (更新信任政策)。

啟用容器的存取記錄

根據預設，AWS Elemental MediaStore 不會收集存取日誌。當您在容器上啟用存取記錄時，MediaStore 會將該容器中存放物件的存取日誌提供給 Amazon CloudWatch。這些存取日誌會為對容器中存放的任何物件所做的請求提供詳細的記錄。這資訊可能包含要求類型、要求中指定的資源，以及要求的處理時間和日期。

Important

在 MediaStore 容器上啟用存取記錄無需額外費用。不過，此服務提供給您的任何日誌檔都會產生一般儲存費用。(您可以隨時刪除這些日誌檔。) AWS 不會估算日誌檔傳遞的資料傳輸費，但會收取用於存取日誌檔之一般資料傳輸率的費用。

若要啟用存取記錄 (AWS CLI)

- 在中 AWS CLI，使用 `start-access-logging` 命令：

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

此命令沒有傳回值。

停用容器的存取記錄

當您停用容器上的存取記錄時，AWS Elemental MediaStore 會停止將存取日誌傳送至 Amazon CloudWatch。系統不會儲存這些存取日誌而且這些日誌不供擷取。

停用存取記錄 (AWS CLI)

- 在中 AWS CLI，使用 `stop-access-logging` 命令：

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

此命令沒有傳回值。

故障診斷 AWS Elemental MediaStore 中的存取記錄

當 AWS Elemental MediaStore 存取日誌未出現在 Amazon CloudWatch 中時，請參閱下表了解潛在原因和解決方案。

Note

請務必啟用 AWS CloudTrail Logs 以協助進行疑難排解程序。

徵狀	問題可能是 ...	嘗試這麼做 ...
<p>即使已啟用 CloudTrail 日誌，您也不會看到任何 CloudTrail 事件。</p>	<p>IAM 角色不存在或其名稱、許可或信任政策不正確。</p>	<p>建立具有正確名稱、許可和信任政策的角色。請參閱 the section called “設定 CloudWatch 的許可”。</p>
<p>您已提交 DescribeContainer API 請求，但回應顯示 AccessLoggingEnabled 參數具有值 False。此外，您看不到任何讓 MediaStoreAccessLogs 角色成功進行 DescribeLogGroup、DescribeLogStream、CreateLogGroup 或 CreateLogStream 呼叫的 CloudTrail 事件。</p>	<p>IAM 角色不存在或其名稱、許可或信任政策不正確。</p> <p>容器上沒有啟用存取記錄。</p>	<p>建立具有正確名稱、許可和信任政策的角色。請參閱 the section called “設定 CloudWatch 的許可”。</p> <p>啟用容器的存取日誌。請參閱 the section called “啟用存取記錄”。</p>
<p>在 CloudTrail 主控台上，您會看到事件，其中包含與 MediaStoreAccessLogs 角色相關的存取遭拒錯誤。CloudTrail 事件可能包含如下所示的行：</p> <pre> "eventSource": "logs.amazonaws.com", "errorCode": "AccessDenied", "errorMessage": "User: arn:aws:sts::11112223333:assumed-role/MediaStoreAccessLogs/MediaStoreAccessLogsSession is not authorized to perform: logs:DescribeLogGroups on resource: arn:aws:logs:us-west-2:1111 </pre>	<p>IAM 角色沒有 AWS Elemental MediaStore 的正確許可。</p>	<p>將 IAM 角色更新為具備正確的許可和信任政策。請參閱 the section called “設定 CloudWatch 的許可”。</p>

徵狀	問題可能是 ...	嘗試這麼做 ...
<p>22223333:log-group::log-stream:",</p> <p>您沒有看到適用於一或多個整個容器的任何日誌。</p>	<p>您的帳戶可能超過每個區域每個帳戶日誌群組的 CloudWatch 配額。</p> <p>請參閱《Amazon CloudWatch Logs 使用者指南》中的日誌群組配額。</p>	<p>在 CloudWatch 主控台上，判斷您的帳戶是否符合日誌群組的 CloudWatch 配額。如果必要，請求增加配額。</p>
<p>您會在 CloudWatch 中看到一些日誌，但並非所有您預期看到的日誌。</p>	<p>您的帳戶可能超過每個區域每個帳戶每秒交易的 CloudWatch 配額。</p> <p>請參閱《Amazon CloudWatch Logs 使用者指南 PutLogEvents 》中的配額。</p>	<p>請求提高每個區域每個帳戶每秒 CloudWatch 交易的配額。</p> <p>CloudWatch</p>

存取日誌格式

存取日誌檔是由一系列的 JSON 格式の日誌記錄所組成，其中每個日誌記錄代表一個請求。日誌中欄位的順序可能有所不同。以下是包含兩個日誌記錄的範例日誌：

```
{
  "Path": "/FootballMatch/West",
  "Requester": "arn:aws:iam::111122223333:user/maria-garcia",
  "AWSAccountId": "111122223333",
  "RequestID":
  "aaaAAA111bbbBBB222cccCCC333dddDDD444eeeEEE555ffffFFF666gggGGG777hhhHHH888iiiIII999jjjJJJ",
  "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":
"dddDDD444eeeEEE555ffffFFF666gggGGG777hhhHHH888iiiIII999jjjJJJ000cccCCC333bbbBBB222aaaAAA",
"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
}

```

下列清單說明日誌記錄欄位：

AWSAccountId

用於提出請求的帳戶的帳戶 AWS ID。

BytesReceived

MediaStore 伺服器在請求內文中接收的位元組數。

BytesSent

MediaStore 伺服器在回應內文中傳送的位元組數。此值通常與在伺服器回應中包含的 Content-Length 標頭值相同。

ContainerName

收到請求的容器名稱。

ErrorCode

MediaStore 錯誤代碼 (例如 `InternalServerError`)。如果沒有錯誤發生，則會顯示 - 字元。即使狀態碼為 200 (表示關閉連線或在伺服器開始串流回應後發生錯誤)，仍可能會顯示錯誤碼。

ExpiresAt

物件的過期日期和時間。此值是根據套用到容器的生命週期政策 [transient data rule](#) 中由設定的過期時間。此值是 ISO-8601 日期時間，且會根據服務請求之主機的系統時鐘。如果生命週期政策沒有套用到物件的暫時性資料規則，或者沒有套用到容器的生命週期政策，則此欄位的值為 `null`。此欄位僅適用於下列操作：`PutObject`、`DescribeObject`、`GetObject` 和 `DeleteObject`。

HTTPStatus

回應的數字 HTTP 狀態碼。

作業

已執行的操作，例如 `PutObject` 或 `ListItems`。

路徑

容器中的路徑，此容器為物件的存放位置。如果操作不會採用路徑參數，則會出現 - 字元。

ReceivedTime

收到請求的時間。此值是 ISO-8601 日期時間，且會根據服務請求之主機的系統時鐘。

要求者

用來提出請求的帳戶使用者 Amazon Resource Name (ARN)。若是未經授權的請求，這個值會是 `anonymous`。如果在身分驗證完成前要求失敗，則此欄位可能會從日誌中遺失。對於這類要求，`ErrorCode` 可能會識別授權問題。

RequestID

AWS Elemental MediaStore 產生的字串，用於唯一識別每個請求。

來源

請求者的明顯網際網路地址或進行呼叫的 AWS 服務服務委託人。如果中間代理伺服器與防火牆會將提出請求的機器地址模糊處理，此值會設為 `Null`。

TotalTime

從伺服器角度計算的請求所經過的毫秒數 (ms)。此值是從服務收到您請求的時間開始，計算到回應傳送最後一組位元組的時間。從用戶端角度進行的測量會受網路延遲影響，因此會從伺服器角度來進行此值的測量。

TurnAroundTime

MediaStore 處理您的請求所花費的毫秒數。此值是從收到您要求的最後位元組的時間開始，計算到回應傳送出第一組位元組的時間。

日誌中欄位的順序可能有所不同。

記錄狀態變更會在一段時間後生效

記錄容器狀態的變更，要一段時間後才會實際影響到日誌檔交付。例如，若已啟用容器記錄，則在接下來的一小時內提出之請求，可能有些會記錄下來，有些則不會。如果您停用容器 B 的記錄，系統可能會繼續交付下一個小時的日誌，有些則不會。在所有情況下，新的設定最終都會生效，您無需採取任何進一步動作。

伺服器日誌交付最佳作法

存取日誌記錄會依最佳作法交付。已適合設定為在交付日誌記錄中記錄結果的容器的多數請求。大多數的日誌記錄會於記錄後的數小時內交付，但也可以常交付。

並不保證存取記錄的完成程度與時間先後順序。特定要求的日誌記錄，可能會在實際處理要求之後很久才交付，或者有可能完全不會交付。存取日誌的目的在於讓您能了解容器流量的真實狀態。雖然日誌記錄極少會缺失，但存取記錄並不代表所有請求的完整記錄。

其遵循存取記錄功能的最佳做法，在 AWS 入口網站提供的用量報告 ([AWS 管理主控台](#)上的帳單和成本管理報告) 中，可能包含一或多個未出現在已交付存取日誌中的存取請求。

存取記錄格式的程式設計考量

有時候，我們可能要在新增欄位，來擴展存取日誌的格式。必須寫入剖析存取日誌的程式碼，才能處理它不了解的其他欄位。

CloudWatch Events

Amazon CloudWatch Events 可讓您自動化您的 AWS 服務，並自動回應系統事件，例如應用程式可用性問題或資源變更。您可編寫簡單的規則，來指示您在意的事件，以及當事件符合規則時所要自動執行的動作。

⚠ Important

一般而言，AWS 服務會在幾秒鐘內將事件通知傳遞至 CloudWatch Events，但有時可能需要一分鐘或更長的時間。

當檔案上傳到容器或從容器移除時，CloudWatch 服務中會連續觸發兩個事件：

1. [the section called “物件狀態變更事件”](#)
2. [the section called “容器狀態變更事件”](#)

如需訂閱這些事件的資訊，請參閱 [Amazon CloudWatch](#)。

可以自動觸發的動作如下：

- 叫用 AWS Lambda 函數
- 調用 Amazon EC2 執行命令
- 將事件轉傳至 Amazon Kinesis Data Streams
- 啟用 AWS Step Functions 狀態機器
- 通知 Amazon SNS 主題或 AWS SMS 佇列

搭配 AWS Elemental MediaStore 使用 CloudWatch Events 的一些範例包括：

- 每當建立容器時啟用 Lambda 函數
- 刪除物件時通知 Amazon SNS 主題

如需詳細資訊，請參閱 [Amazon CloudWatch Events 使用者指南](#)。

主題

- [AWS Elemental MediaStore 物件狀態變更事件](#)
- [AWS Elemental MediaStore 容器狀態變更事件](#)

AWS Elemental MediaStore 物件狀態變更事件

物件的狀態已變更時 (當物件已上傳或刪除時)，將會發佈此事件。

Note

因暫時性資料規則而過期的物件在過期時不會發出 CloudWatch 事件。

如需訂閱此事件的資訊，請參閱 [Amazon CloudWatch](#)。

物件已更新

```
{
  "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"
  }
}
```

物件已移除

```
{
  "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 容器狀態變更事件

容器的狀態已變更時 (當容器已新增或刪除時)，將會發佈此事件。如需訂閱此事件的資訊，請參閱 [Amazon CloudWatch](#)。

容器已建立

```
{
  "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"
  }
}
```

容器已移除

```
{
  "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"
}
}
```

使用 Amazon CloudWatch 指標監控 AWS Elemental MediaStore

您可以使用 CloudWatch 監控 AWS Elemental MediaStore，這會收集原始資料並將其處理為可讀指標。CloudWatch 會保留統計資料 15 個月，讓您可以存取歷史資訊，並更清楚 Web 應用程式或服務的效能。您也可以設定留意特定閾值的警示，當滿足這些閾值時傳送通知或採取動作。如需詳細資訊，請參閱 [Amazon CloudWatch 使用者指南](#)。

對於 AWS Elemental MediaStore，您可能想要在該指標達到特定閾值時，監看 BytesDownloaded 並傳送電子郵件給自己。

使用 CloudWatch 主控台檢視指標

指標會先依服務命名空間分組，再依各命名空間內不同的維度組合分類。

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/cloudwatch/> 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 指標。
3. 在 All metrics (所有指標) 下，選擇 AWS/MediaPackage 命名空間。
4. 選擇要檢視指標的指標維度。例如，選擇 Request metrics by container 以檢視已傳送至容器之不同請求類型的指標。

使用 檢視指標 AWS CLI

- 在命令提示中，使用下列命令：

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

AWS Elemental MediaStore 指標

下表列出 AWS Elemental MediaStore 傳送至 CloudWatch 的指標。

Note

若要檢視指標，您必須將[指標政策](#)新增至容器，以允許 MediaStore 將指標傳送至 Amazon CloudWatch。

指標	說明
RequestCount	<p>對 MediaStore 容器提出的 HTTP 請求總數，依操作類型 (Put、Get、Delete、Describe、) 分隔 List。</p> <p>單位：Count</p> <p>有效維度：</p> <ul style="list-style-type: none"> • 容器名稱 • 物件群組名稱 • 要求類型 <p>有效的統計資訊：總和</p>
4xxErrorCount	<p>對 MediaStore 發出導致 4xx 錯誤的 HTTP 請求數目。</p> <p>單位：Count</p> <p>有效維度：</p> <ul style="list-style-type: none"> • 容器名稱 • 物件群組名稱 • 要求類型 <p>有效的統計資訊：總和</p>

指標	說明
5xxErrorCount	<p>對 MediaStore 發出導致 5xx 錯誤的 HTTP 請求數目。</p> <p>單位：Count</p> <p>有效維度：</p> <ul style="list-style-type: none">• 容器名稱• 物件群組名稱• 要求類型 <p>有效的統計資訊：總和</p>
BytesUploaded	<p>針對對 MediaStore 容器提出的請求上傳的位元組數，其中請求包含內文。</p> <p>單位：位元組</p> <p>有效維度：</p> <ul style="list-style-type: none">• 容器名稱• 物件群組名稱 <p>有效的統計資訊：平均 (每個請求的位元組數)、總和 (每個期間的位元組數)、範例計數、最小值 (相容於 P0.0)、最大值 (相同於 p100)，任何介於 p0.0 與 p99.9 的百分位數</p>

指標	說明
BytesDownloaded	<p>針對對 MediaStore 容器提出的請求下載的位元組數，其中回應包含內文。</p> <p>單位：位元組</p> <p>有效維度：</p> <ul style="list-style-type: none">• 容器名稱• 物件群組名稱 <p>有效的統計資訊：平均 (每個請求的位元組數)、總和 (每個期間的位元組數)、範例計數、最小值 (相容於 P0.0)、最大值 (相同於 p100)，任何介於 p0.0 與 p99.9 的百分位數</p>
TotalTime	<p>從伺服器角度計算的請求所經過的毫秒數。此值是從 MediaStore 收到您的請求的時間開始測量，到它傳送回應的最後一個位元組的時間。從用戶端角度進行的測量會受網路延遲影響，因此會從伺服器角度來進行此值的測量。</p> <p>單位：毫秒</p> <p>有效維度：</p> <ul style="list-style-type: none">• 容器名稱• 物件群組名稱• 要求類型 <p>有效的統計資訊：平均、最小值 (相容於 P0.0)、最大值 (相同於 p100)、任何介於 p0.0 與 p100 的百分位數</p>

指標	說明
TurnaroundTime	<p>MediaStore 處理您的請求所花費的毫秒數。此值是從 MediaStore 收到請求的最後一個位元組到傳送回應的第一個位元組的時間進行測量。</p> <p>單位：毫秒</p> <p>有效維度：</p> <ul style="list-style-type: none"> • 容器名稱 • 物件群組名稱 • 要求類型 <p>有效的統計資訊：平均、最小值 (相容於 P0.0)、最大值 (相同於 p100)、任何介於 p0.0 與 p100 的百分位數</p>
ThrottleCount	<p>對 MediaStore 提出的 HTTP 請求數量已調節。</p> <p>單位：Count</p> <p>有效維度：</p> <ul style="list-style-type: none"> • 容器名稱 • 物件群組名稱 • 要求類型 <p>有效的統計資訊：總和</p>

標記 AWS Elemental MediaStore 資源

標籤是您指派或 AWS 指派給 AWS 資源的自訂屬性標籤。每個標籤都有兩個部分：

- 標籤鍵 (例如, CostCenter、Environment 或 Project)。標籤鍵會區分大小寫。
- 選用欄位, 稱為標籤值 (例如 111122223333 或 Production)。忽略標籤值基本上等同於使用空字串。與標籤金鑰相同, 標籤值會區分大小寫。

標籤可協助您執行以下操作：

- 識別和組織您的 AWS 資源。許多 AWS 服務支援標籤，因此您可以對來自不同服務的資源指派相同的標籤，以指出資源是相關的。例如，您可以將相同的標籤指派給輸入的 AWS Elemental MediaStore AWS Elemental MediaLive `##`。
- 追蹤您的 AWS 成本。您可以在 AWS 帳單與成本管理 儀表板上啟用這些標籤。AWS 會使用標籤來分類您的成本，並提供每月成本分配報告給您。如需詳細資訊，請參閱《[AWS Billing 使用者指南](#)》中的[使用成本分配標籤](#)。

下列各節提供有關 AWS Elemental MediaStore 標籤的詳細資訊。

AWS Elemental MediaStore 中支援的資源

AWS Elemental MediaStore 中的下列資源支援標記：

- `##`

如需新增和管理標籤的詳細資訊，請參閱[管理標籤](#)。

AWS Elemental MediaStore 不支援 AWS Identity and Access Management (IAM) 的標籤型存取控制功能。

標籤命名和使用慣例

下列基本命名和使用慣例適用於搭配 AWS Elemental MediaStore 資源使用標籤的：

- 每個資源的上限為 50 個標籤。
- 對於每一個資源，每個標籤金鑰必須是唯一的，且每個標籤金鑰只能有一個值。
- 最大標籤索引鍵長度為 128 個 UTF-8 形式的 Unicode 字元。
- 最大標籤值長度為 256 個 UTF-8 形式的 Unicode 字元。
- 允許的字元包括可用 UTF-8 表示的英文字母、數字、空格，還有以下特殊字元：`.:+=@_/-` (連字號)。Amazon EC2 資源允許任何字元。
- 標籤鍵與值皆區分大小寫。做為最佳實務，請決定大寫標籤的策略，並一致地在所有資源類型中實作該策略。例如，決定要使用 `Costcenter`、`costcenter` 還是 `CostCenter`，並針對所有標籤使用相同的慣例。避免針對相似的標籤使用不一致的大小寫處理。
- 標籤禁止使用 `aws:` 字首，保留供 AWS 使用。您不可編輯或刪除具此字首的標籤金鑰或值。具此字首的標籤，不會算在每個資源配額的標籤計數內。

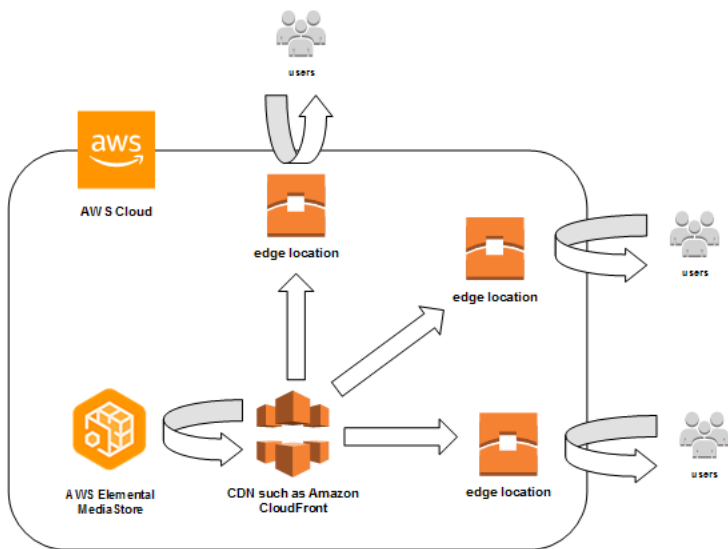
管理標籤

標籤是由資源上的 Key 和 Value 屬性組成。您可以使用 AWS CLI 或 MediaStore API 來新增、編輯或刪除這些屬性的值。如需有關使用標籤的資訊，請參閱 AWS Elemental MediaStore API 參考中的下列章節：

- [CreateContainer](#)
- [ListTagsForResource](#)
- [資源](#)
- [TagResource](#)
- [UntagResource](#)

使用內容交付網路 (CDN)

您可以使用內容交付網路 (CDN)，例如 [Amazon CloudFront](#)，為您存放在 AWS Elemental MediaStore 的內容提供服務。CDN 是快取影片等內容的全球分佈伺服器。當使用者要求提供您的內容時，CDN 將此要求路由至產生最低延遲的節點。如果您的內容已於該節點中快取，CDN 就會立即提供該內容。如果您的內容目前不在該節點，CDN 會從原始伺服器（例如您的 MediaStore 容器）擷取內容，並將其分發給使用者。



主題

- [允許 Amazon CloudFront 存取您的 AWS Elemental MediaStore 容器](#)
- [AWS Elemental MediaStore 與 HTTP 快取的互動](#)

允許 Amazon CloudFront 存取您的 AWS Elemental MediaStore 容器

您可以使用 Amazon CloudFront 來提供存放在 AWS Elemental MediaStore 容器中的內容。您可以透過下列其中一種方式執行此操作：

- [使用原始存取控制 \(OAC\)](#) - (建議) 如果您的 AWS 區域支援 CloudFront 的 OAC 功能，請使用此選項。
- [使用共用秘密](#) - 如果您的 AWS 區域不支援 CloudFront 的 OAC 功能，請使用此選項。

使用原始存取控制 (OAC)

您可以使用 Amazon CloudFront 的原始存取控制 (OAC) 功能，以改善安全性來保護 AWS Elemental MediaStore 原始伺服器。您可以在 MediaStore 原始伺服器的 CloudFront 請求上啟用 [AWS Signature 第 4 版 \(SigV4\)](#)，並設定 CloudFront 是否應簽署請求的時間和時間。您可以透過 主控台、APIs、SDK 或 CLI 存取 CloudFront 的 OAC 功能，而且使用不需額外付費。

如需搭配 MediaStore 使用 OAC 功能的詳細資訊，請參閱 [《Amazon CloudFront 開發人員指南》](#) 中的 [限制對 MediaStore 原始伺服器的存取](#)。

使用共用秘密

如果您的 AWS 區域 不支援 Amazon CloudFront 的 OAC 功能，您可以將政策連接至 AWS Elemental MediaStore 容器，以授予 CloudFront 讀取存取權或更高版本。

Note

如果您的 AWS 區域 支援，建議使用 OAC 功能。下列程序要求您使用共用秘密設定 MediaStore 和 CloudFront，以限制對 MediaStore 容器的存取。若要遵循最佳安全實務，此手動組態需要定期輪換秘密。透過 MediaStore 原始伺服器上的 OAC，您可以指示 CloudFront 使用 SigV4 簽署請求，並將其轉送至 MediaStore 以進行簽章比對，無需使用和輪換秘密。這可確保在提供媒體內容之前自動驗證請求，讓透過 MediaStore 和 CloudFront 交付媒體內容變得更簡單且更安全。

允許 CloudFront 存取您的容器（主控台）

1. 在 <https://console.aws.amazon.com/mediastore/> 開啟 MediaStore 主控台。
2. 在 Containers (容器) 頁面上，選擇容器名稱。


容器詳細資訊頁面隨即出現。

3. 在容器政策區段中，連接授予 Amazon CloudFront 讀取存取權或更高權限的政策。

Example

下列範例政策類似於 [透過 HTTPS 公開讀取存取](#) 的範例政策，符合這些要求，因為它允許透過 HTTPS 向網域提交請求的任何人提供 GetObject 和 DescribeObject 命令。此外，下列範例政策可更妥善保護您的工作流程，因為它僅允許 CloudFront 存取 MediaStore 物件，前提是請求是透過 HTTPS 連線發生，並包含正確的參考標頭。

- 在 Container CORS policy (容器 CORS 政策) 區段中，指派允許適當存取等級的政策。

 Note

只有在您想要讓使用者存取瀏覽器型播放程式時，[CORS 政策](#)才為必要。

- 請記下列詳細資訊：
 - 指定給容器的資料端點。您可以在 Containers (容器) 頁面的 Info (資訊) 區段中找到這項資訊。在 CloudFront 中，資料端點稱為原始網域名稱。
 - 存放物件之容器中的資料夾結構。在 CloudFront 中，這稱為原始伺服器路徑。請注意，這是選擇性設定。如需原始伺服器路徑的詳細資訊，請參閱 [Amazon CloudFront 開發人員指南](#)。
- 在 CloudFront 中，建立[設定為從 AWS Elemental MediaStore 提供內容](#)的分佈。您將需要您在前一步驟中收集的資訊。

將政策連接至 MediaStore 容器之後，您必須將 CloudFront 設定為僅使用原始伺服器請求的 HTTPS 連線，並新增具有正確秘密值的自訂標頭。

設定 CloudFront 透過 HTTPS 連線存取您的容器，其中包含參考者標頭 (主控台) 的秘密值

- 開啟 CloudFront 主控台。
- 在原始伺服器頁面上，選擇您的 MediaStore 原始伺服器。
- 選擇編輯。
- 僅針對通訊協定選擇 HTTPS。
- 在新增自訂標頭區段中，選擇新增標頭。
- 針對名稱，選擇推薦者。對於值，請使用您在容器政策中使用的相同 `<secretValue>` 字串。
- 選擇儲存並讓變更部署。

AWS Elemental MediaStore 與 HTTP 快取的互動

AWS Elemental MediaStore 會存放物件，讓 Amazon CloudFront 等內容交付網路 (CDNs) 可以正確且有效率地快取物件。當最終使用者或 CDN 從 MediaStore 擷取物件時，服務會傳回影響物件快取行為的 HTTP 標頭。HTTP 1.1 快取行為的標準可在 [RFC2616 第 13 節](#)中取得)。這些標頭包括：

- **ETag** (不可自訂) – 實體標籤標頭是 MediaStore 傳送之回應的唯一識別符。標準合規 CDNs 和 Web 瀏覽器使用此標籤做為快取物件的金鑰。MediaStore ETag會在上傳每個物件時自動產生。您可以[檢視物件的詳細資料](#)，以判斷其 ETag 值。
- **Last-Modified** (不可自訂) – 此標頭的值表示物件修改的日期和時間。MediaStore 會在物件上傳時自動產生此值。
- **Cache-Control** (可自訂) — 此標頭的值會控制物件在 CDN 查看它是否已遭修改前，應該快取的時間長度。當您使用 [CLI](#) 或 [API](#) 將物件上傳至 MediaStore 容器時，您可以將此標頭設定為任何值。完整一組的有效值會在 [HTTP/1.1 文件](#) 中加以說明。如果您在上傳物件時未設定此值，MediaStore 不會在擷取物件時傳回此標頭。

快取控制標頭常見的使用案例是指定快取物件的持續時間。例如，假設您的視訊資訊清單檔案經常遭編碼器覆寫。您可以將 `max-age` 設為 10，表示該物件應該快取只有 10 秒。或者，假設您有永遠不會遭覆寫的儲存視訊區段。您可以將這個物件的 `max-age` 設定為 31536000 來快取約 1 年的長度。

條件式請求

MediaStore 的條件式請求

MediaStore 對條件式請求 (使用請求標頭，例如 `If-Modified-Since` 和 `If-None-Match`，如 [RFC7232](#) 中所述) 和無條件請求的回應相同。這表示當 MediaStore 收到有效的 `GetObject` 請求時，即使用戶端已有物件，服務仍會傳回物件。

對 CDN 的條件式請求

代表 MediaStore 提供內容的 CDNs 可以透過傳回 `304 Not Modified` 來處理條件式請求，如 [RFC7232 第 4.1 節](#) 所述。這表示系統已不需要傳輸完整的物件內容，因為請求者已經有符合條件請求的物件。

CDN (以及與 HTTP/1.1 相容的其他快取) 會將這些決策基於原始服務器轉發的 `ETag` 和 `Cache-Control` 標頭。若要控制 CDNs 查詢 MediaStore 原始伺服器以更新重複擷取物件的頻率，請在將這些物件上傳至 MediaStore 時設定這些物件的 `Cache-Control` 標頭。

搭配 AWS SDK 使用此服務

AWS 軟體開發套件 (SDKs) 適用於許多熱門的程式設計語言。每個 SDK 都提供 API、程式碼範例和說明文件，讓開發人員能夠更輕鬆地以偏好的語言建置應用程式。

SDK 文件	代碼範例
適用於 C++ 的 AWS SDK	適用於 C++ 的 AWS SDK 程式碼範例
AWS CLI	AWS CLI 程式碼範例
適用於 Go 的 AWS SDK	適用於 Go 的 AWS SDK 程式碼範例
適用於 Java 的 AWS SDK	適用於 Java 的 AWS SDK 程式碼範例
適用於 JavaScript 的 AWS SDK	適用於 JavaScript 的 AWS SDK 程式碼範例
適用於 Kotlin 的 AWS SDK	適用於 Kotlin 的 AWS SDK 程式碼範例
適用於 .NET 的 AWS SDK	適用於 .NET 的 AWS SDK 程式碼範例
適用於 PHP 的 AWS SDK	適用於 PHP 的 AWS SDK 程式碼範例
AWS Tools for PowerShell	AWS Tools for PowerShell 程式碼範例
適用於 Python (Boto3) 的 AWS SDK	適用於 Python (Boto3) 的 AWS SDK 程式碼範例
適用於 Ruby 的 AWS SDK	適用於 Ruby 的 AWS SDK 程式碼範例
適用於 Rust 的 AWS SDK	適用於 Rust 的 AWS SDK 程式碼範例
適用於 SAP ABAP 的 AWS SDK	適用於 SAP ABAP 的 AWS SDK 程式碼範例
適用於 Swift 的 AWS SDK	適用於 Swift 的 AWS SDK 程式碼範例

如需此服務的特定範例，請參閱 [MediaStore AWS SDKs 的程式碼範例](#)。

i 可用性範例

找不到所需的內容嗎？請使用本頁面底部的提供意見回饋連結申請程式碼範例。

MediaStore AWS SDKs的程式碼範例

下列程式碼範例示範如何使用 MediaStore 搭配 AWS 軟體開發套件 (SDK)。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境中查看內容中的動作。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

程式碼範例

- [MediaStore AWS SDKs的基本範例](#)
 - [MediaStore AWS SDKs的動作](#)
 - [CreateContainer 搭配 AWS SDK 或 CLI 使用](#)
 - [DeleteContainer 搭配 AWS SDK 或 CLI 使用](#)
 - [DeleteObject 搭配 AWS SDK 使用](#)
 - [DescribeContainer 搭配 AWS SDK 或 CLI 使用](#)
 - [GetObject 搭配 AWS SDK 或 CLI 使用](#)
 - [ListContainers 搭配 AWS SDK 或 CLI 使用](#)
 - [PutObject 搭配 AWS SDK 或 CLI 使用](#)

MediaStore AWS SDKs的基本範例

下列程式碼範例示範如何 AWS Elemental MediaStore 搭配 AWS SDKs 使用的基本概念。

範例

- [MediaStore AWS SDKs的動作](#)
 - [CreateContainer 搭配 AWS SDK 或 CLI 使用](#)
 - [DeleteContainer 搭配 AWS SDK 或 CLI 使用](#)
 - [DeleteObject 搭配 AWS SDK 使用](#)
 - [DescribeContainer 搭配 AWS SDK 或 CLI 使用](#)
 - [GetObject 搭配 AWS SDK 或 CLI 使用](#)
 - [ListContainers 搭配 AWS SDK 或 CLI 使用](#)
 - [PutObject 搭配 AWS SDK 或 CLI 使用](#)

MediaStore AWS SDKs的動作

下列程式碼範例示範如何使用 AWS SDKs 執行個別 MediaStore 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

下列範例僅包含最常使用的動作。如需完整清單，請參閱 [《AWS Elemental MediaStore API 參考》](#)。

範例

- [CreateContainer 搭配 AWS SDK 或 CLI 使用](#)
- [DeleteContainer 搭配 AWS SDK 或 CLI 使用](#)
- [DeleteObject 搭配 AWS SDK 使用](#)
- [DescribeContainer 搭配 AWS SDK 或 CLI 使用](#)
- [GetObject 搭配 AWS SDK 或 CLI 使用](#)
- [ListContainers 搭配 AWS SDK 或 CLI 使用](#)
- [PutObject 搭配 AWS SDK 或 CLI 使用](#)

CreateContainer 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreateContainer。

CLI

AWS CLI

建立容器

下列create-container範例會建立新的空容器。

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

輸出：

```
{
  "Container": {
    "AccessLoggingEnabled": false,
    "CreationTime": 1563557265,
    "Name": "ExampleContainer",
    "Status": "CREATING",
```

```
        "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/  
ExampleContainer"  
    }  
}
```

如需詳細資訊，請參閱 AWS Elemental MediaStore 使用者指南中的[建立容器](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的[CreateContainer](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[AWS 程式碼範例儲存庫](#)中設定和執行。

```
import software.amazon.awssdk.services.mediastore.MediaStoreClient;  
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.mediastore.model.CreateContainerRequest;  
import software.amazon.awssdk.services.mediastore.model.CreateContainerResponse;  
import software.amazon.awssdk.services.mediastore.model.MediaStoreException;  
  
/**  
 * Before running this Java V2 code example, set up your development  
 * environment, including your credentials.  
 *  
 * For more information, see the following documentation topic:  
 *  
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-  
started.html  
 */  
public class CreateContainer {  
    public static long sleepTime = 10;  
  
    public static void main(String[] args) {  
        final String usage = ""  
  
            Usage:    <containerName>
```

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

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

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

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

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

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

        System.out.println("The container ARN value is " +
containerResponse.container().arn());
        System.out.println("Finished ");
    } catch (MediaStoreException | InterruptedException e) {
        System.err.println(e.getMessage());
    }
}
```

```
        System.exit(1);
    }
}
}
```

- 如需 API 詳細資訊，請參閱AWS SDK for Java 2.x 《API 參考》中的 [CreateContainer](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteContainer 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteContainer。

CLI

AWS CLI

刪除容器

下列delete-container範例會刪除指定的容器。只有當容器沒有任何物件時，您才可以將該容器刪除。

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

此命令不會產生輸出。

如需詳細資訊，請參閱 AWS Elemental MediaStore 使用者指南中的[刪除容器](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteContainer](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

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

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class CreateContainer {
    public static long sleepTime = 10;

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

            Usage:    <containerName>

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

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

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

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

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

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

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

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

- 如需 API 詳細資訊，請參閱AWS SDK for Java 2.x 《API 參考》中的 [DeleteContainer](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteObject 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 DeleteObject。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

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

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class DeleteObject {
    public static void main(String[] args) throws URISyntaxException {
        final String usage = ""

            Usage:    <completePath> <containerName>

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

            """;
    }
}
```

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

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

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

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

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

        mediaStoreData.deleteObject(deleteObjectRequest);

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

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

    DescribeContainerRequest containerRequest =
DescribeContainerRequest.builder()
```

```
        .containerName(containerName)
        .build();

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

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DescribeContainer 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DescribeContainer。

CLI

AWS CLI

檢視容器的詳細資訊

下列 describe-container 範例顯示指定容器的詳細資訊。

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

輸出：

```
{  
  "Container": {  
    "CreationTime": 1563558086,  
    "AccessLoggingEnabled": false,  
    "ARN": "arn:aws:mediastore:us-west-2:111122223333:container/  
ExampleContainer",  
    "Status": "ACTIVE",  
    "Name": "ExampleContainer",
```

```
        "Endpoint": "https://aaabbbcccddee.data.mediastore.us-  
west-2.amazonaws.com"  
    }  
}
```

如需詳細資訊，請參閱 AWS Elemental MediaStore 使用者指南中的[檢視容器的詳細資訊](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DescribeContainer](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.mediastore.MediaStoreClient;  
import software.amazon.awssdk.services.mediastore.model.DescribeContainerRequest;  
import  
    software.amazon.awssdk.services.mediastore.model.DescribeContainerResponse;  
import software.amazon.awssdk.services.mediastore.model.MediaStoreException;  
  
/**  
 * Before running this Java V2 code example, set up your development  
 * environment, including your credentials.  
 *  
 * For more information, see the following documentation topic:  
 *  
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-  
started.html  
 */  
public class DescribeContainer {  
  
    public static void main(String[] args) {  
        final String usage = ""  
  
            Usage:    <containerName>
```

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

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

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

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

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

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

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


輸出：

```
{
  "StatusCode": 206,
  "ContentRange": "bytes 0-100/2307346",
  "ContentLength": "101",
  "LastModified": "Fri, 19 Jul 2019 21:32:20 GMT",
  "ContentType": "image/jpeg",
  "ETag": "2aa333bbcc8d8d22d777e999c88d4aa9eeeeee4dd89ff7f5555555555555555da6d3"
}
```

如需詳細資訊，請參閱 AWS Elemental MediaStore 使用者指南中的 [下載物件](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetObject](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

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

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class GetObject {
    public static void main(String[] args) throws URISyntaxException {
        final String usage = ""

            Usage:    <completePath> <containerName> <savePath>

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

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

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

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

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

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

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

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

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

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

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

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

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

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [GetObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListContainers 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListContainers。

CLI

AWS CLI

檢視容器清單

下列 list-containers 範例顯示與您的帳戶相關聯的所有容器清單。

```
aws mediastore list-containers
```

輸出：

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

如需詳細資訊，請參閱 AWS Elemental MediaStore 使用者指南中的[檢視容器清單](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的[ListContainers](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[AWS 程式碼範例儲存庫](#)中設定和執行。

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

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

    public static void main(String[] args) {

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

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

```
    }

    public static void listAllContainers(MediaStoreClient mediaStoreClient) {
        try {
            ListContainersResponse containersResponse =
mediaStoreClient.listContainers();
            List<Container> containers = containersResponse.containers();
            for (Container container : containers) {
                System.out.println("Container name is " + container.name());
            }

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

- 如需 API 詳細資訊，請參閱AWS SDK for Java 2.x 《API 參考》中的 [ListContainers](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutObject 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutObject。

CLI

AWS CLI

上傳物件

下列put-object範例會將物件上傳至指定的容器。您可以指定將物件儲存在容器中的資料夾路徑。如果資料夾已存在，AWS Elemental MediaStore 會將物件存放在資料夾中。如果資料夾不存在，服務會建立資料夾，然後將物件存放在資料夾中。

```
aws mediastore-data put-object \  
  --endpoint https://aaabbbcccddee.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
```

輸出：

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

如需詳細資訊，請參閱 AWS Elemental MediaStore 使用者指南中的[上傳物件](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutObject](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

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

/**
 * Before running this Java V2 code example, set up your development
```

```
* environment, including your credentials.
*
* For more information, see the following documentation topic:
*
* https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
*/
public class PutObject {
    public static void main(String[] args) throws URISyntaxException {
        final String USAGE = ""

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

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

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

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

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

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

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

            PutObjectRequest objectRequest = PutObjectRequest.builder()
```

```
        .path(completePath)
        .contentType("video/mp4")
        .build();

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

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

public static String getEndpoint(String containerName) {

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

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

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

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [PutObject](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

AWS Elemental MediaStore 中的配額

Service Quotas 主控台提供 AWS Elemental MediaStore 配額的相關資訊。除了檢視預設配額之外，您還可以使用 Service Quotas 主控台來[請求增加配額](#)以取得可調整配額。

下表說明 AWS Elemental MediaStore 中先前稱為限制的配額。配額是 AWS 帳戶的服務資源或操作數量上限。

Note

若要將配額指派給帳戶中的個別容器，請聯絡 AWS Support 或您的帳戶管理員。此選項可協助您在容器之間分割帳戶層級限制，以防止一個容器用完整個配額。

資源或操作	預設配額	說明
容器	100	此帳戶中可建立的容器數量上限。
資料夾層級	10	容器中可建立的資料夾層級數量上限。您可以視需要建立任何數量的資料夾，只要容器中的巢狀層級不超過 10 層。
資料夾	無限制	您可以視需要建立任何數量的資料夾，只要容器中的巢狀層級不超過 10 層。
物件大小	25 MB	單一物件的檔案大小上限。
物件	無限制	您可以將任意數量的物件上傳到帳戶中的資料夾或容器。
DeleteObject API 請求的速率	100	您每秒可以提出的操作請求上限數量。其他請求會受到限流。 您可以 要求增加配額 。
DescribeObject API 請求的速率	1,000	您每秒可以提出的操作請求上限數量。其他請求會受到限流。 您可以 要求增加配額 。

資源或操作	預設配額	說明
標準上傳可用性的 GetObject API 請求率	1,000	您每秒可以提出的操作請求上限數量。其他請求會受到限流。 您可以 要求增加配額 。
串流上傳可用性的 GetObject API 請求速率	25	您每秒可以提出的操作請求上限數量。其他請求會受到限流。 您可以 要求增加配額 。
ListItems API 請求的速率	5	您每秒可以提出的操作請求上限數量。其他請求會受到限流。 您可以 要求增加配額 。
PutObject API 請求的區塊傳輸編碼速率 (也稱為串流上傳可用性)	10	您每秒可以提出的操作請求上限數量。其他請求會受到限流。 您可以 要求增加配額 。在請求中，指定請求的 TPS 和平均物件大小。
PutObject API 請求的標準上傳可用性速率	100	您每秒可以提出的操作請求上限數量。其他請求會受到限流。 您可以 要求增加配額 。在請求中，指定請求的 TPS 和平均物件大小。
指標政策中的規則	10	您可以包含在指標政策中的規則數量上限。
物件生命週期政策中的規則	10	您可以包含在物件生命週期政策中包含的規則數量上限。

AWS Elemental MediaStore 相關資訊

下表列出使用 AWS Elemental MediaStore 時，您會發現有用的相關資源。

- [課程和研討會](#) – 連結至以角色為基礎的專門課程，以及自行安排進度的實驗室，以協助強化您的 AWS 技能並取得實際體驗。
- [AWS 開發人員中心](#) – 探索教學課程、下載工具，並了解 AWS 開發人員事件。
- [AWS 開發人員工具](#) – 開發人員工具、SDKs、IDE 工具組和命令列工具的連結，用於開發和管理 AWS 應用程式。
- [入門資源中心](#) – 了解如何設定您的 AWS 帳戶、加入 AWS 社群，以及啟動您的第一個應用程式。
- [實用的教學課程](#) - 按照逐步教學課程在 AWS 上啟動第一個應用程式。
- [AWS 白皮書](#) – 技術 AWS 白皮書的完整清單連結，涵蓋架構、安全和經濟等主題，並由 AWS 解決方案架構師或其他技術專家撰寫。
- [AWS 支援中心](#) – 建立和管理 AWS 支援案例的中樞。也包含其他實用資源的連結，例如論壇、技術 FAQs、服務運作狀態和 AWS Trusted Advisor。
- [支援](#) – 有關 one-on-one 快速回應支援管道的資訊的主要網頁 支援，可協助您在雲端中建置和執行應用程式。
- [聯絡我們](#) – 查詢有關 AWS 帳單、帳戶、事件、濫用與其他問題的聯絡中心。
- [AWS 網站條款](#) – 關於我們的著作權和商標、您的帳戶、授權和網站存取，以及其他主題的詳細資訊。

使用者指南的文件歷史記錄

下表說明此 AWS Elemental MediaStore 版本的文件。如需有關此文件更新的通知，您可以訂閱 RSS 訂閱源。

變更	描述	日期
終止支援通知	支援終止通知：2025 年 11 月 13 日，AWS 將停止支援 AWS Elemental MediaStore。2025 年 11 月 13 日之後，您將無法再存取 MediaStore 主控台或 MediaStore 資源。如需詳細資訊，請造訪此 部落格文章 。	2024 年 11 月 12 日
原始存取控制 (OAC) 改進	新增如何搭配 AWS Elemental MediaStore 使用 OAC 的相關資訊。	2023 年 4 月 17 日
配額更新	更正的配額值和描述 Rules in a Metric Policy。	2022 年 10 月 25 日
ExpiresAt 欄位	存取日誌現在包含 ExpiresAt 欄位，根據容器生命週期政策中的暫時性資料規則，指出物件的過期日期和時間。	2020 年 7 月 16 日
生命週期轉換規則	您現在可以將生命週期轉換規則新增到物件的生命週期政策中，將物件設為在到達一定存留期後移動到不常存取 (IA) 儲存體方案。	2020 年 4 月 20 日
空容器	您現在可以一次刪除容器內的所有物件。	2020 年 4 月 7 日

支援 Amazon CloudWatch 指標	您可以設定指標政策，以決定 MediaStore 傳送至 CloudWatch 的指標。	2020 年 3 月 30 日
刪除物件規則中的萬用字元	在物件生命週期政策中，您現在可以在刪除物件規則中使用萬用字元。這可讓您根據要在特定天數後刪除服務的檔案名稱或副檔名來指定檔案。	2019 年 12 月 20 日
物件生命週期政策	您現在可以將規則新增至物件生命週期政策，以秒為單位指定過期時間。	2019 年 9 月 13 日
CloudFormation 支援	您現在可以使用 CloudFormation 範本自動建立容器。CloudFormation 範本可管理五個 API 動作的資料：建立容器、設定存取記錄、更新預設容器政策、新增跨來源資源共享 (CORS) 政策，以及新增物件生命週期政策。	2019 年 5 月 17 日
串流上傳可用性配額	對於具有串流上傳可用性的物件 (物件的區塊傳輸)，PutObject 作業不能超過 10 TPS，且 GetObject 作業不能超過 25 TPS。	2019 年 4 月 8 日
物件的區塊傳輸	新增對物件區塊傳輸的支援。此功能可讓您指定物件可在完全上傳前供下載使用。	2019 年 4 月 5 日
存取記錄	AWS Elemental MediaStore 現在支援存取記錄，提供對容器中物件提出之請求的詳細記錄。	2019 年 2 月 25 日

物件生命週期政策	新增對物件生命週期政策的支援，此政策可管理目前容器中物件的過期日期。	2018 年 12 月 12 日
增加物件大小配額	物件大小的配額現在是 25 MB。	2018 年 10 月 10 日
增加物件大小配額	物件大小的配額現在是 20 MB。	2018 年 9 月 6 日
AWS CloudTrail 整合	CloudTrail 整合內容已更新，以符合 CloudTrail 服務的最新變更。	2018 年 7 月 12 日
CDN 協同合作	新增如何將 AWS Elemental MediaStore 與內容交付網路 (CDN) 搭配使用的相關資訊，例如 Amazon CloudFront。	2018 年 4 月 14 日
CORS 組態	AWS Elemental MediaStore 現在支援跨來源資源共用 (CORS)，可讓載入某個網域的用戶端 Web 應用程式與不同網域的資源互動。	2018 年 2 月 7 日
新的服務與指南	這是影片起始和儲存服務的初始版本、AWS Elemental MediaStore 和 AWS Elemental MediaStore 使用者指南。	2017 年 11 月 27 日

Note

- AWS 媒體服務並非設計用於應用程式或需要故障安全效能的情況下，例如生命安全操作、導航或通訊系統、空氣流量控制或生命支援機器，其中服務的無法使用、中斷或故障可能會導致死亡、人員受傷、資產損壞或環境損壞。

AWS 詞彙表

如需最新的 AWS 術語，請參閱 AWS 詞彙表 參考中的[AWS 詞彙表](#)。