Amazon EMR 無服務器用戶指南

Amazon EMR

Amazon EMR: Amazon EMR 無服務器用戶指南

Table of Contents

什麼是 Amazon EMR 無伺服器？

Amazon EMR 無伺服器是 Amazon EMR 提供無伺服器執行階段環境的部署選項。如此可簡化使用最新開放原始碼架構 (例如 Apache Spark 和 Apache Hive) 之分析應用程式的作業。有了EMR無伺服器，您不需要設定、最佳化、保護或操作叢集，就能使用這些架構執行應用程式。

EMR無伺服器可協助您避免資料處理工作的佈建過度或佈建不足的資源。EMR無伺服器會自動判斷應用程式所需的資源、取得這些資源來處理您的工作，並在工作完成時釋放資源。對於應用程式需要在幾秒鐘內回應的使用案例 (例如互動式資料分析)，您可以在建立應用程式時預先初始化應用程式所需的資源。

有了EMR無伺服器，您將繼續獲得 Amazon 的優勢EMR，例如適用於熱門架構的開放原始碼相容性、並行性和最佳化執行時期效能。

EMR無伺服器適合想要使用開放原始碼架構輕鬆操作應用程式的客戶。它提供快速的作業啟動、自動容量管理，以及直接的成本控制。

概念

在本節中，我們涵蓋EMR無伺服器使用者指南中出現的EMR無伺服器術語和概念。

發行版本

Amazon EMR 版本是來自大數據生態系統的一組開放原始碼應用程式。每個版本都包含不同的大數據應用程式、元件和功能，這些應用程式和功能可供EMR無伺服器部署和設定，以便它們能夠執行您的應用程式。建立應用程式時，您必須指定其發行版本。選擇您要在應用程式中使用的 Amazon EMR 發行版本和開放原始碼架構版本。若要深入瞭解發行前版本，請參閱Amazon EMR 無伺服器發行版本。

應用程式

使用EMR無伺服器，您可以建立一或多個使用開放原始碼分析架構的EMR無伺服器應用程式。若要建立應用程式，您必須指定下列屬性：

- 您要使用的開EMR放原始碼架構版本的 Amazon 發行版本。若要判斷您的發行版本，請參閱Amazon EMR 無伺服器發行版本。
- 您希望應用程序使用的特定運行時，例如 Apache 星火或 Apache 蜂房。

建立應用程式之後，您可以將資料處理工作或互動式要求提交至應用程式。

每個EMR無伺服器應用程式都在安全的 Amazon Virtual Private Cloud (VPC) 上執行，與其他應用程式完全不同。此外，您可以使用 AWS Identity and Access Management (IAM) 定義哪些使用者和角色可以存取應用程式的原則。您也可以指定限制，以控制和追蹤應用程式產生的使用成本。

當您需要執行下列動作時，請考慮建立多個應用程式：

- 使用不同的開源框架
- 針對不同使用案例使用不同版本的開放原始碼架構
- 從一個版本升級到另一個版本時執行 A/B 測試
- 為測試和生產方案維護單獨的邏輯環境
- 為不同團隊提供獨立的邏輯環境，提供獨立的成本控制和使用情況
- 分隔不同的 line-of-business 應用

EMR無伺服器是一種區域性服務，可簡化工作負載在一個區域中跨多個可用區域執行的方式。若要進一步瞭解如何搭配EMR無伺服器使用應用程式，請參閱與應用程式互動。

作業執行

工作執行是提交至EMR無伺服器應用程式的要求，應用程式會以不同步方式執行並追蹤完成。工作範例包括您提交至 Apache Hive 應用程式的 HiveQL 查詢，或是您送出至 Apache Spark 應用程式的 PySpark 資料處理指令碼。當您提交工作時，您必須指定作業用來存取的執行時期角色 (編寫於IAM) AWS 資源，例如 Amazon S3 對象。您可以將多個工作執行請求提交至應用程式，而且每個工作執行都可以使用不同的執行時期角色來存取 AWS 的費用。EMR無伺服器應用程式在收到工作後立即開始執行工作，並同時執行多個工作要求。若要深入瞭解EMR無伺服器如何執行工作，請參閱執行任務。

工作程序

EMR無伺服器應用程式內部使用 Worker 來執行您的工作負載。這些 Worker 的預設大小取決於您的應用程式類型和 Amazon EMR 發行版本。當您排定工作執行時，您可以覆寫這些大小。

當您提交工作時，EMR無伺服器會計算應用程式對工作所需的資源，並排程工作者。EMR無伺服器可將您的工作負載分解為任務、下載影像、佈建和設定工作者，並在工作完成時將其解除委任。EMR無伺服器會根據工作每個階段所需的工作負載和平行處理原則，自動擴展或縮減工作者。這種自動擴展功能讓您無需預估應用程式執行工作負載所需的 Worker 數量。

預先初始化容量

EMR無伺服器提供預先初始化的容量功能，可讓 Worker 保持初始化，並在幾秒鐘內做好回應。這種能力有效地為應用程序創建了一個溫暖的工作人員池。若要為每個應用程式設定此功能，請設定應用程式的`initial-capacity`參數。當您設定預先初始化的容量時，工作可以立即啟動，以便您可以實作反覆應用程式和時間敏感的工作。若要進一步瞭解預先初始化的 Worker，請參閱設定應用程式。

EMR工作室

EMRStudio 是可用來管理EMR無伺服器應用程式的使用者主控台。當您建立第一個EMR無伺服器應用程式時，如果您的帳戶中沒有 EMR Studio，我們會自動為您建立一個工作室。您可以從 Amazon EMR 主控台存取 EMR Studio，也可以透過身分識別供應商 (IdP) IAM 或IAM身分中心開啟聯合存取。執行此操作時，使用者無需直接存取 Amazon EMR 主控台即可存取 Studio 和管理EMR無伺服器應用程式。若要深入了解EMR無伺服器應用程式如何與 EMR Studio 搭配運作，請參閱從 EMR Studio 主控台與您的應用程式互動和從 EMR Studio 主控台執行工作。

開始使用EMR無伺服器的先決條件

主題

註冊一個 AWS 帳戶

如果您沒有 AWS 帳戶，請完成下列步驟來建立一個步驟。

若要註冊成為 AWS 帳戶

1. 打開https://portal.aws.amazon.com/billing/註冊。
2. 請遵循線上指示進行。

   部分註冊程序需接收來電，並在電話鍵盤輸入驗證碼。

   當您註冊一個 AWS 帳戶，一個 AWS 帳戶根使用者已建立。根使用者可以存取所有 AWS 服務 和帳戶中的資源。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 註冊過程完成後，會向您發送確認電子郵件。您可以隨時前往 https://aws.amazon.com/並選擇「我的帳戶」，檢視目前的帳戶活動並管理您的帳戶。

建立具有管理存取權的使用者

在您註冊一個 AWS 帳戶，保護您的 AWS 帳戶根使用者，啟用 AWS IAM Identity Center，並建立系統管理使用者，這樣您就不會將 root 使用者用於日常工作。

保護您的 AWS 帳戶根使用者

1. 登入 AWS Management Console通過選擇 Root 用戶並輸入您的帳戶所有者 AWS 帳戶 電子郵件地址。在下一頁中，輸入您的密碼。

如需使用 root 使用者登入的說明，請參閱以 root 使用者身分登入 AWS 登入 使用者指南。

2. 為您的 root 使用者開啟多因素驗證 (MFA)。

   如需指示，請參閱為您的MFA裝置啟用虛擬裝置 AWS 帳戶 使用者指南中的 root IAM 使用者 (主控台)。

建立具有管理存取權的使用者

1. 啟用IAM身分識別中心。

   如需指示，請參閱啟用 AWS IAM Identity Center 中的 AWS IAM Identity Center 使用者指南。

2. 在IAM身分識別中心中，將管理存取權授與使用者。

   若要取得有關使用 IAM Identity Center 目錄 做為您的身分識別來源，請參閱以預設值設定使用者存取 IAM Identity Center 目錄 中的 AWS IAM Identity Center 使用者指南。

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 使用者登入URL，請使用建立IAM身分識別中心使用者時傳送至您電子郵件地址的登入資訊。

  如需使用IAM身分識別中心使用者登入的說明，請參閱登入 AWS 存取入口網站 AWS 登入 使用者指南。

指派存取權給其他使用者

1. 在 IAM Identity Center 中，建立遵循套用最低權限權限的最佳作法的權限集。

   如需指示，請參閱 AWS IAM Identity Center 使用者指南。

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，請參閱新增群組 AWS IAM Identity Center 使用者指南。

授予許可

在生產環境中，建議您使用更精細的原則。如需此類政策的範例，請參閱EMR無伺服器的使用者存取原則範例。若要進一步了解存取管理，請參閱的存取管理 AWS 《IAM使用者指南》中的資源。

對於需要在沙箱環境中開始使用EMR無伺服器的使用者，請使用類似下列內容的原則：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EMRStudioCreate",
            "Effect": "Allow",
            "Action": [
                "elasticmapreduce:CreateStudioPresignedUrl",
                "elasticmapreduce:DescribeStudio",
                "elasticmapreduce:CreateStudio",
                "elasticmapreduce:ListStudios"
            ],
            "Resource": "*"
        },
        {
            "Sid": "EMRServerlessFullAccess",
            "Effect": "Allow",
            "Action": [
                "emr-serverless:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowEC2ENICreationWithEMRTags",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaLast": "ops.emr-serverless.amazonaws.com"
                }
            }
        },
        {
            "Sid": "AllowEMRServerlessServiceLinkedRoleCreation",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/*"
```

```
        }
    ]
}
```

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center:

  建立權限合集。遵循中建立權限集中的指示 AWS IAM Identity Center 使用者指南。

- IAM透過身分識別提供者管理的使用者：

  建立聯合身分的角色。請遵循《使用指南》中的〈為第三方身分識別提供IAM者 (同盟) 建立角色〉中的指示進行。

- IAM使用者：
  - 建立您的使用者可擔任的角色。請按照《用戶指南》中的「為IAM用戶創建角色」中的IAM說明進行操作。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循《使用指南》中的「向使用者 (主控台) 新增權限」IAM 中的指示進行。

授與程式設計存取權

如果用戶想要與之互動，則需要以程式設計方式存取 AWS 的之外 AWS Management Console。 授與程式設計存取權的方式取決於存取的使用者類型 AWS.

若要授與使用者程式設計存取權，請選擇下列其中一個選項。

哪個使用者需要程式設計存取權？	到	By
人力身分 (在IAM身分識別中心管理的使用者)	使用臨時登入資料來簽署程式設計要求 AWS CLI, AWS SDKs，或 AWS APIs.	請依照您要使用的介面所提供的指示操作。 • 對於 AWS CLI，請參閱配置 AWS CLI 若要使用 AWS IAM Identity Center 中的 AWS Command Line Interface 使用者指南。

哪個使用者需要程式設計存取權？	到	By
		• 用於 AWS SDKs、工具和 AWS APIs，請參閱IAM身分識別中心驗證 AWS SDKs和工具參考指南。
IAM	使用臨時登入資料來簽署程式設計要求 AWS CLI, AWS SDKs，或 AWS APIs.	遵循使用臨時認證中的說明進行操作 AWS《IAM使用者指南》中的資源。
IAM	(不建議使用) 使用長期認證來簽署程式設計要求 AWS CLI, AWS SDKs，或 AWS APIs.	請依照您要使用的介面所提供的指示操作。 • 對於 AWS CLI，請參閱「使用使用IAM者認證進行驗證」AWS Command Line Interface 使用者指南。 • 用於 AWS SDKs和工具，請參閱使用長期認證進行身份驗證 AWS SDKs和工具參考指南。 • 用於 AWS APIs，請參閱《使用指南》中的〈管理使IAM用IAM者的存取金鑰〉。

安裝和配置 AWS CLI

如果您想要使用EMR無伺服器APIs，您必須安裝最新版本的 AWS Command Line Interface (AWS CLI). 你不需要 AWS CLI 從 EMR Studio 主控台使用EMR無伺服器，您可以按照中的CLI步驟在從主控台開始使用EMR無伺服器不使用.

若要設定 AWS CLI

1. 若要安裝最新版本的 AWS CLI 如需 macOS、Linux 或視窗，請參閱安裝或更新最新版本的 AWS CLI.

2. 若要設定 AWS CLI 並安全設置您的訪問 AWS 服務，包括EMR無伺服器，請參閱使aws configure用快速設定。
3. 若要驗證設定，請在 DataBrew 命令提示字元中輸入下列命令。

```
aws emr-serverless help
```

AWS CLI 指令使用預設 AWS 區域 從您的配置中，除非您使用參數或配置文件進行設置。若要設定 AWS 區域 使用參數，您可以將--region參數添加到每個命令中。

若要設定 AWS 區域 使用配置文件，首先在~/.aws/config文件或文件中添加一個命名的配置%UserProfile%/.aws/config文件（適用於 Microsoft Windows）。依照 [已命名] 設定檔中的步驟執行 AWS CLI。 接下來，設置 AWS 區域 和其他設定的指令類似於下列範例中的指令。

```
[profile emr-serverless]
aws_access_key_id = ACCESS-KEY-ID-OF-IAM-USER
aws_secret_access_key = SECRET-ACCESS-KEY-ID-OF-IAM-USER
region = us-east-1
output = text
```

開啟 主控台

本節中大多數以主控台為導向的主題都是從 Amazon 主控台開始。EMR如果您尚未登入 AWS 帳戶，登錄，然後打開 Amazon EMR 控制台並繼續下一節以繼續開始使用 Amazon EMR。

開始使用 Amazon EMR 無伺服器

本教學課程可協助您在部署 Spark 或 Hive 工作負載範例時開始使用EMR無伺服器。您將創建，運行和調試自己的應用程序。我們在本教程的大多數部分顯示默認選項。

啟動EMR無伺服器應用程式之前，請先完成下列工作。

主題

- 授與使用EMR無伺服器的權限
- 準備EMR無伺服器的儲存
- 建立 EMR Studio 以執行互動式工作負載
- 建立工作執行時期角色
- 從主控台開始使用EMR無伺服器
- 從開始 AWS CLI

授與使用EMR無伺服器的權限

若要使用EMR無伺服器，您需要具有授與無EMR伺服器權限的附加原則的使用者或IAM角色。若要建立使用者並將適當的策略附加到該使用者，請遵循中的指示授予許可。

準備EMR無伺服器的儲存

在本教學中，您將使用 S3 儲存貯體來存放您將使用EMR無伺服器應用程式執行的範例 Spark 或 Hive 工作負載的輸出檔案和日誌。若要建立值區，請遵循 Amazon 簡單儲存服務主控台使用者指南中建立儲存貯體中的指示進行。將任何進一步*DOC-EXAMPLE-BUCKET*的參照取代為新建立值區的名稱。

建立 EMR Studio 以執行互動式工作負載

如果您想要使用EMR無伺服器透過 EMR Studio 中託管的筆記本執行互動式查詢，則需要為EMR無伺服器指定 S3 儲存貯體和最低服務角色，才能建立工作區。如需設定步驟，請參閱 Amazon EMR 管理指南中的設定EMR工作室。如需互動式工作負載的詳細資訊，請參閱透EMR過 Studio 以EMR無伺服器執行互動式工作負載。

建立工作執行時期角色

在EMR無伺服器中執行 Job 使用提供特定精細權限的執行時期角色 AWS 服務 和運行時的資源。在本教學中，公用 S3 儲存貯體會託管資料和指令碼。*DOC-EXAMPLE-BUCKET*存儲桶存儲輸出。

若要設定工作執行階段角色，請先建立具有信任原則的執行時期角色，以便 EMR Serverless 可以使用新角色。接下來，將必要的 S3 存取政策附加到該角色。以下步驟將引導您完成整個過程。

Console

1. 瀏覽至IAM主控台，位於https://console.aws.amazon.com/iam/。
2. 在左側導覽窗格中，選擇 Roles (角色)。
3. 選擇建立角色。
4. 對於角色類型，請選擇 [自訂信任原則]，然後貼上下列信任原則。這可讓提交至 Amazon EMR 無伺服器應用程式的任務存取其他工作 AWS 服務 代表您。

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

5. 選擇「下一步」導覽至「新增權限」頁面，然後選擇「建立原則」。
6. [建立原則] 頁面會在新索引標籤上開啟。貼上JSON下方的政策。

> **Important**
>
> 將下列政策*DOC-EXAMPLE-BUCKET*中建立的實際值區名稱取代為中建立的值區名稱準備EMR無伺服器的儲存。這是 S3 存取的基本政策。如需更多工作執行階段角色範例，請參閱Amazon EMR 無伺服器的 Job 務執行階段角色。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadAccessForEMRSamples",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::*.elasticmapreduce",
                "arn:aws:s3:::*.elasticmapreduce/*"
            ]
        },
        {
            "Sid": "FullAccessToOutputBucket",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        },
        {
            "Sid": "GlueCreateAndReadDataCatalog",
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase",
                "glue:CreateDatabase",
                "glue:GetDataBases",
                "glue:CreateTable",
                "glue:GetTable",
                "glue:UpdateTable",
                "glue:DeleteTable",
                "glue:GetTables",
                "glue:GetPartition",
```

```
                "glue:GetPartitions",
                "glue:CreatePartition",
                "glue:BatchCreatePartition",
                "glue:GetUserDefinedFunctions"
            ],
            "Resource": ["*"]
        }
    ]
}
```

7. 在 [檢閱原則] 頁面上，輸入原則的名稱，例如EMRServerlessS3AndGlueAccessPolicy。
8. 重新整理 [連結權限原則] 頁面，然後選擇EMRServerlessS3AndGlueAccessPolicy。
9. 在 [名稱、檢閱和建立] 頁面的 [角色名稱] 中，輸入角色的名稱，例如EMRServerlessS3RuntimeRole。若要建立此IAM角色，請選擇 [建立角色]。

CLI

1. 建立名為的檔案emr-serverless-trust-policy.json，其中包含要用於IAM角色的信任原則。檔案應包含下列原則。

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

2. 建立名為的IAM角色EMRServerlessS3RuntimeRole。使用您在上一個步驟中建立的信任原則。

```
aws iam create-role \
    --role-name EMRServerlessS3RuntimeRole \
    --assume-role-policy-document file://emr-serverless-trust-policy.json
```

請記下輸出中的 ARN。您可以在ARN工作提交期間使用新角色，在此之後稱為*job-role-arn*.

3. 建立名為的檔案emr-sample-access-policy.json，該檔案可定義工作負載的IAM原則。這可提供對存放在公有 S3 儲存貯體中的指令碼和資料的讀取存取權，以*DOC-EXAMPLE-BUCKET*及.

> **Important**
>
> 以下列原則*DOC-EXAMPLE-BUCKET*中建立的實際值區名稱取代準備EMR無伺服器的儲存。這是一項基本政策 AWS Glue 和 S3 存取。如需更多工作執行階段角色範例，請參閱Amazon EMR 無伺服器的 Job 務執行階段角色。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadAccessForEMRSamples",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::*.elasticmapreduce",
                "arn:aws:s3:::*.elasticmapreduce/*"
            ]
        },
        {
            "Sid": "FullAccessToOutputBucket",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
```

```
            ]
        },
        {
            "Sid": "GlueCreateAndReadDataCatalog",
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase",
                "glue:CreateDatabase",
                "glue:GetDataBases",
                "glue:CreateTable",
                "glue:GetTable",Understanding default application behavior,
 including auto-start and auto-stop, as well as maximum capacity and worker
 configurations for configuring an application with &EMRServerless;.
                "glue:UpdateTable",
                "glue:DeleteTable",
                "glue:GetTables",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:CreatePartition",
                "glue:BatchCreatePartition",
                "glue:GetUserDefinedFunctions"
            ],
            "Resource": ["*"]
        }
    ]
}
```

4. 建立IAM以您在步驟 3 中建立的原則檔命名EMRServerlessS3AndGlueAccessPolicy的策略。請注意輸出ARN中的，因為您將在下一個步驟中使用新策略。ARN

```
aws iam create-policy \
    --policy-name EMRServerlessS3AndGlueAccessPolicy \
    --policy-document file://emr-sample-access-policy.json
```

請注意輸出ARN中的新政策。您將在下一步*policy-arn*中替代它。

5. 將IAM原則附加EMRServerlessS3AndGlueAccessPolicy至工作執行階段角色EMRServerlessS3RuntimeRole。

```
aws iam attach-role-policy \
    --role-name EMRServerlessS3RuntimeRole \
    --policy-arn policy-arn
```

從主控台開始使用EMR無伺服器

要完成的步驟

步驟 1：建立EMR無伺服器應用程式

使用EMR無伺服器建立新的應用程式，如下所示。

1. 登入 AWS Management Console 並在 https://console.aws.amazon.com/em r 打開 Amazon EMR 控制台。
2. 在左側導覽窗格中，選擇「EMR無伺服器」以導覽至EMR無伺服器登陸頁面。
3. 若要建立或管理EMR無伺服器應用程式，您需要 EMR Studio UI。
   - 如果你已經有一個EMR工作室 AWS 區域 您要建立應用程式的位置，然後選取 [管理應用程式] 以瀏覽至您的 EMR Studio，或選取您要使用的工作室。
   - 如果您沒有EMR工作室 AWS 區域 您想要建立應用程式的位置，選擇 [開始使用]，然後選擇 [建立並啟動 Studio]。EMR無伺服器會為您建立 EMR Studio，讓您可以建立和管理應用程式。
4. 在新索引標籤中開啟的 [建立 Studio UI] 中，輸入應用程式的名稱、類型和發行版本。如果您只想執行批次工作，請選取「僅針對批次工作使用預設設定」。對於互動式工作負載，請選取使用互動式工作負載的預設 您也可以使用此選項，在啟用互動功能的應用程式上執行批次工作。如有需要，您可以稍後變更這些設定。

   如需詳細資訊，請參閱建立工作室。
5. 選取建立應用程式以建立第一個應用程式。

繼續下一節，步驟 2：提交作業執行或互動式工作負載以提交工作執行或互動式工作負載。

步驟 2：提交作業執行或互動式工作負載

Spark job run

在本教程中，我們使用 PySpark 腳本來計算多個文本文件中唯一單詞的出現次數。公用唯讀 S3 儲存貯體同時存放指令碼和資料集。

執行星火工作

1. 使用下列指令`wordcount.py`將範例指令碼上傳至您的新值區。

```
aws s3 cp s3://us-east-1.elasticmapreduce/emr-containers/samples/wordcount/
scripts/wordcount.py s3://DOC-EXAMPLE-BUCKET/scripts/
```

2. 完成會將您步驟 1：建立EMR無伺服器應用程式帶到 EMR Studio 中的「應用程式詳細資訊」頁面。在那裡，選擇「提交工作」選項。
3. 在 [送出工作] 頁面上，完成下列動作。
   - 在「名稱」欄位中，輸入您要呼叫工作執行的名稱。
   - 在「執行時期角色」欄位中，輸入您在中建立之角色的名稱建立工作執行時期角色。
   - 在指令碼位置欄位中`s3://DOC-EXAMPLE-BUCKET/scripts/wordcount.py`，輸入 S3 URI。
   - 在「指令集引數」欄位中，輸入`["s3://DOC-EXAMPLE-BUCKET/emr-serverless-spark/output"]`。
   - 在「S park 屬性」區段中，選擇 「以文字形式編輯」，然後輸入下列組態。

```
--conf spark.executor.cores=1 --conf spark.executor.memory=4g --
conf spark.driver.cores=1 --conf spark.driver.memory=4g --conf
 spark.executor.instances=1
```

4. 若要開始工作執行，請選擇送出工作。
5. 在 [Job 執行] 索引標籤中，您應該會看到新工作以 [執行中] 狀態執行。

Hive job run

在本教程的這一部分中，我們創建了一個表，插入一些記錄，並運行計數聚合查詢。若要執行 Hive 工作，請先建立一個檔案，其中包含要在單一工作中執行的所有 Hive 查詢，將檔案上傳到 S3，並在啟動 Hive 工作時指定此 S3 路徑。

若要執行蜂巢工作

1. 創建一個名為hive-query.ql的文件，其中包含要在 Hive 作業中運行的所有查詢。

```
create database if not exists emrserverless;
use emrserverless;
create table if not exists test_table(id int);
drop table if exists Values__Tmp__Table__1;
insert into test_table values (1),(2),(2),(3),(3),(3);
select id, count(id) from test_table group by id order by id desc;
```

2. 使用以下命令上傳hive-query.ql到您的 S3 儲存貯體。

```
aws s3 cp hive-query.ql s3://DOC-EXAMPLE-BUCKET/emr-serverless-hive/query/hive-
query.ql
```

3. 完成會將您步驟 1：建立EMR無伺服器應用程式帶到 EMR Studio 中的「應用程式詳細資訊」頁面。在那裡，選擇「提交工作」選項。
4. 在 [送出工作] 頁面上，完成下列動作。
   - 在「名稱」欄位中，輸入您要呼叫工作執行的名稱。
   - 在「執行時期角色」欄位中，輸入您在中建立之角色的名稱建立工作執行時期角色。
   - 在指令碼位置欄位中s3://DOC-EXAMPLE-BUCKET/emr-serverless-hive/query/hive-query.ql，輸入 S3 URI。
   - 在 [Hive 屬性] 區段中，選擇 [以文字形式編輯]，然後輸入下列設定。

     ```
     --hiveconf hive.log.explain.output=false
     ```

   - 在「Job 組態」段落中，選擇 「編輯為」JSON，然後輸入下列資訊JSON。

     ```
     {
        "applicationConfiguration":
        [{
            "classification": "hive-site",
               "properties": {
                   "hive.exec.scratchdir": "s3://DOC-EXAMPLE-BUCKET/emr-serverless-
     hive/hive/scratch",
                   "hive.metastore.warehouse.dir": "s3://DOC-EXAMPLE-BUCKET/emr-
     serverless-hive/hive/warehouse",
                   "hive.driver.cores": "2",
                   "hive.driver.memory": "4g",
     ```

```
                "hive.tez.container.size": "4096",
                "hive.tez.cpu.vcores": "1"
            }
    }]
}
```

5. 若要開始工作執行，請選擇送出工作。
6. 在 [Job 執行] 索引標籤中，您應該會看到新工作以 [執行中] 狀態執行。

Interactive workload

使用 Amazon EMR 6.14.0 及更高版本，您可以使用 EMR Studio 中託管的筆記本為無伺服器中的 Spark 執行互動式工作負載。EMR如需包括權限和必要條件的詳細資訊，請參閱透EMR過 Studio 以EMR無伺服器執行互動式工作負載。

建立應用程式並設定必要的權限之後，請使用下列步驟，透過 EMR Studio 執行互動式筆記本：

1. 導覽至 EMR Studio 中的「工作區」索引標籤。如果您仍需要設定 Amazon S3 儲存位置和 EMRStudio 服務角色，請選取畫面頂端橫幅中的 [設定工作室] 按鈕。
2. 若要存取記事本，請選取工作區或建立新的工作區。使用 [快速啟動] 在新索引標籤中開啟您的工作區。
3. 轉到新打開的選項卡。從左側導覽中選取「計算」圖示。選取「EMR無伺服器」做為「運算」類型。
4. 選取您在上一節中建立的互動式啟用應用程式。
5. 在「執行時間角色」欄位中，輸入EMR無伺服器應用程式在工作執行時可承擔的IAM角色名稱。若要進一步了解執行時期角色，請參閱 Amazon EMR 無伺服器使用者指南中的 Job 務執行階段角色。
6. 選取「貼附」。這可能需要長達一分鐘的時間。附加時頁面將重新整理。
7. 選擇一個內核並啟動筆記本。您也可以瀏覽EMR無伺服器上的範例筆記本，並將其複製到您的工作區。若要存取範例記事本，請瀏覽至左側導覽中的**{...}**功能表，並瀏覽記事本檔案名稱`serverless`中的記事本。
8. 在筆記本中，您可以存取驅動程式記錄檔連結和 Apache Spark UI 的連結，這是一個即時介面，可提供監視工作的指標。如需詳細資訊，請參閱 Amazon EMR 無伺服器使用者指南中的監控EMR無伺服器應用程式和任務。

當您將應用程式附加到 Studio 工作區時，如果應用程式尚未執行，應用程式會自動啟動觸發。您也可以在將應用程式附加至工作區之前，預先啟動應用程式並保持準備就緒。

步驟 3：檢視應用程式 UI 和記錄

若要檢視應用程式 UI，請先識別工作執行。Spark UI 或 Hive Tez UI 的選項會根據工作類型，在該工作執行的第一列選項中提供。選取適當的選項。

如果您選擇 Spark UI，請選擇 [執行者] 索引標籤以檢視驅動程式和執行程式記錄檔。如果您選擇了 Hive Tez UI，請選擇所有任務選項卡以查看日誌。

任務執行狀態顯示為「成功」後，您可以在 S3 儲存貯體中檢視任務的輸出。

步驟 4：清理

雖然您建立的應用程式會在閒置 15 分鐘後自動停止，但我們仍建議您釋放不想再次使用的資源。

若要刪除應用程式，請瀏覽至 [列出應用程式] 頁面。選擇您創建的應用程序，然後選擇操作 → 停止以停止應用程序。應用程序處於STOPPED狀態後，選擇相同的應用程序，然後選擇操作 → 刪除。

如需執行 Spark 和 Hive 作業的更多範例，請參閱火花工作和蜂巢工作。

從開始 AWS CLI

步驟 1：建立EMR無伺服器應用程式

使用指emr-serverless create-application令建立您的第一個EMR無伺服器應用程式。您需要指定應用程式類型和與要使用的應用程式EMR版本相關聯的 Amazon 版本標籤。應用程式的名稱是選擇性的。

Spark

若要建立 Spark 應用程式，請執行下列命令。

```
aws emr-serverless create-application \
    --release-label emr-6.6.0 \
    --type "SPARK" \
    --name my-application
```

Hive

要創建一個 Hive 應用程序，運行以下命令。

```
aws emr-serverless create-application \
    --release-label emr-6.6.0 \
```

```
    --type "HIVE" \
    --name my-application
```

請注意輸出中傳回的應用程式 ID。您將使用 ID 來啟動應用程式，並在工作提交期間，在此之後稱為*application-id*.

在繼續之前步驟 2：將作業執行提交至EMR無伺服器應用程式，請確保您的應用程式已達到 get-applicationAPI. CREATED

```
aws emr-serverless get-application \
    --application-id application-id
```

EMR無伺服器可建立員工以配合您要求的工作。依預設，這些是視需求建立的，但您也可以在建立應用程式時設定initialCapacity參數，以指定預先初始化的容量。您也可以限制應用程式可與maximumCapacity參數搭配使用的總容量上限。若要進一步了解這些選項，請參閱設定應用程式。

步驟 2：將作業執行提交至EMR無伺服器應用程式

現在您的EMR無伺服器應用程式已準備好執行作業。

Spark

在此步驟中，我們使用 PySpark 腳本來計算多個文本文件中唯一單詞的出現次數。公用唯讀 S3 儲存貯體同時存放指令碼和資料集。應用程式會將輸出檔案和 Spark 執行階段的日誌資料傳送到您建立的 S3 儲存貯體中的/logs目錄/output和目錄。

執行星火工作

1. 使用以下命令複製我們將運行到新存儲桶的示例腳本。

   ```
   aws s3 cp s3://us-east-1.elasticmapreduce/emr-containers/samples/wordcount/
   scripts/wordcount.py s3://DOC-EXAMPLE-BUCKET/scripts/
   ```

2. 在下面的命令中，用您的應用程序 ID 替換*application-id*。ARN以您在中建立的執行階段角色取代*job-role-arn*建立工作執行時期角色。替代 *job-run-name* 使用您想要調用作業運行的名稱。將所有*DOC-EXAMPLE-BUCKET*字串取代為您建立的 Amazon S3 儲存貯體，然後新增/output至路徑。這會在儲存貯體中建立一個新資料夾，EMRServerless 可以在其中複製應用程式的輸出檔案。

   ```
   aws emr-serverless start-job-run \
   ```

```
    --application-id application-id \
    --execution-role-arn job-role-arn \
    --name job-run-name \
    --job-driver '{
        "sparkSubmit": {
          "entryPoint": "s3://DOC-EXAMPLE-BUCKET/scripts/wordcount.py",
          "entryPointArguments": ["s3://DOC-EXAMPLE-BUCKET/emr-serverless-spark/
output"],
          "sparkSubmitParameters": "--conf spark.executor.cores=1
 --conf spark.executor.memory=4g --conf spark.driver.cores=1 --conf
 spark.driver.memory=4g --conf spark.executor.instances=1"
        }
    }'
```

3. 請注意輸出中傳回的工作執行 ID。請在下列步驟中以此 ID 取*job-run-id*代。

Hive

在本教程中，我們創建了一個表，插入一些記錄，並運行計數聚合查詢。若要執行 Hive 工作，請先建立一個檔案，其中包含要在單一工作中執行的所有 Hive 查詢，將檔案上傳到 S3，並在啟動 Hive 工作時指定此 S3 路徑。

若要執行蜂巢工作

1. 創建一個名為`hive-query.ql`的文件，其中包含要在 Hive 作業中運行的所有查詢。

```
create database if not exists emrserverless;
use emrserverless;
create table if not exists test_table(id int);
drop table if exists Values__Tmp__Table__1;
insert into test_table values (1),(2),(2),(3),(3),(3);
select id, count(id) from test_table group by id order by id desc;
```

2. 使用以下命令上傳`hive-query.ql`到您的 S3 儲存貯體。

```
aws s3 cp hive-query.ql s3://DOC-EXAMPLE-BUCKET/emr-serverless-hive/query/hive-
query.ql
```

3. 在下面的命令中，用您自己*application-id*的應用程序 ID 替換。ARN以您在中建立的執行階段角色取代*job-role-arn*建立工作執行時期角色。將所有*DOC-EXAMPLE-BUCKET*字串取代為您建立的 Amazon S3 儲存貯體，`/output`並將其新增`/logs`至路徑。這會在儲存貯體中建立新資料夾，EMRServerless 可以在其中複製應用程式的輸出和記錄檔。

```
aws emr-serverless start-job-run \
    --application-id application-id \
    --execution-role-arn job-role-arn \
    --job-driver '{
        "hive": {
          "query": "s3://DOC-EXAMPLE-BUCKET/emr-serverless-hive/query/hive-
query.ql",
          "parameters": "--hiveconf hive.log.explain.output=false"
        }
    }' \
    --configuration-overrides '{
      "applicationConfiguration": [{
        "classification": "hive-site",
          "properties": {
            "hive.exec.scratchdir": "s3://DOC-EXAMPLE-BUCKET/emr-serverless-
hive/hive/scratch",
            "hive.metastore.warehouse.dir": "s3://DOC-EXAMPLE-BUCKET/emr-
serverless-hive/hive/warehouse",
            "hive.driver.cores": "2",
            "hive.driver.memory": "4g",
            "hive.tez.container.size": "4096",
            "hive.tez.cpu.vcores": "1"
            }
        }],
        "monitoringConfiguration": {
          "s3MonitoringConfiguration": {
            "logUri": "s3://DOC-EXAMPLE-BUCKET/emr-serverless-hive/logs"
           }
        }
    }'
```

4. 請注意輸出中傳回的工作執行 ID。請在下列步驟中以此 ID 取*job-run-id*代。

步驟 3：檢視作業執行的輸出

工作執行通常需要 3-5 分鐘才能完成。

Spark

您可以使用以下命令檢查 Spark 作業的狀態。

```
aws emr-serverless get-job-run \
```

```
    --application-id application-id \
    --job-run-id job-run-id
```

將記錄目標設定為時s3://*DOC-EXAMPLE-BUCKET*/emr-serverless-spark/logs，您可以在下找到執行此特定工作的記錄s3://*DOC-EXAMPLE-BUCKET*/emr-serverless-spark/logs/applications/*application-id*/jobs/*job-run-id*。

對於 Spark 應用程式，EMR無伺服器會每 30 秒將事件日誌推送到 S3 日誌目標中的sparklogs資料夾。當您的工作完成時，驅動程式和執行者的 Spark 執行階段記錄會上傳至 Worker 類型適當命名的資料夾，例如driver或executor。 PySpark 工作的輸出會上傳至s3://*DOC-EXAMPLE-BUCKET*/output/。

Hive

您可以使用以下命令檢查您的 Hive 作業的狀態。

```
aws emr-serverless get-job-run \
    --application-id application-id \
    --job-run-id job-run-id
```

將記錄目標設定為時s3://*DOC-EXAMPLE-BUCKET*/emr-serverless-hive/logs，您可以在下找到執行此特定工作的記錄s3://*DOC-EXAMPLE-BUCKET*/emr-serverless-hive/logs/applications/*application-id*/jobs/*job-run-id*。

對於 Hive 應用程式，EMR無伺服器會持續將 Hive 驅動程式上傳到HIVE_DRIVER資料夾，Tez 工作記錄到 S3 日誌目的地的TEZ_TASK資料夾。任務執行到達SUCCEEDED狀態後，Hive 查詢的輸出就會在您在的monitoringConfiguration欄位中指定的 Amazon S3 位置中使用configurationOverrides。

步驟 4：清理

完成此教學課程的工作後，請考慮刪除您建立的資源。我們建議您釋放不打算再次使用的資源。

刪除您的應用

要刪除應用程序，請使用以下命令。

```
aws emr-serverless delete-application \
    --application-id application-id
```

刪除 S3 日誌儲存貯體

若要刪除 S3 記錄和輸出儲存貯體，請使用下列命令。*DOC-EXAMPLE-BUCKET*以在中建立的 S3 儲存貯體的實際名稱取代準備EMR無伺服器的儲存。

```
aws s3 rm s3://DOC-EXAMPLE-BUCKET --recursive
aws s3api delete-bucket --bucket DOC-EXAMPLE-BUCKET
```

刪除工作執行時期角色

若要刪除執行階段角色，請從角色中斷連結原則。然後，您可以同時刪除角色和策略。

```
aws iam detach-role-policy \
    --role-name EMRServerlessS3RuntimeRole \
    --policy-arn policy-arn
```

若要刪除角色，請使用下列命令。

```
aws iam delete-role \
    --role-name EMRServerlessS3RuntimeRole
```

若要刪除附加至角色的原則，請使用下列命令。

```
aws iam delete-policy \
    --policy-arn policy-arn
```

如需執行 Spark 和 Hive 作業的更多範例，請參閱火花工作和蜂巢工作。

與應用程式互動

本節介紹如何與 Amazon EMR 無伺服器應用程式互動 AWS CLI 和星火和蜂巢引擎的默認值。

主題

應用狀態

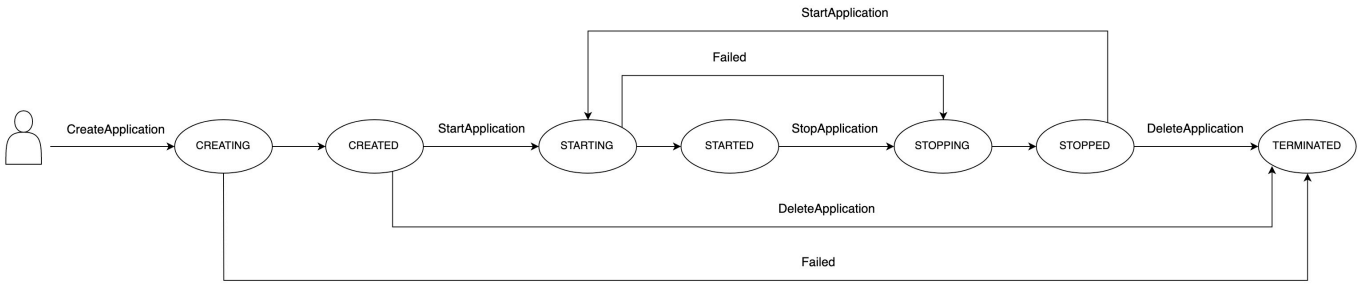
當您使用EMR無伺服器建立應用程式時，應用程式執行會進入CREATING狀態。工作將經過以下狀態，直到其失敗 (以 0 代碼結束) 或失敗 (以與非零代碼結束) 為止。

應用程式可以具有下列狀態：

州	描述
正在建立	該應用程序正在準備中，尚未準備好使用。
已建立	應用程式已建立，但尚未佈建容量。您可以修改應用程式以變更其初始容量組態。
啟動	應用程式正在啟動並正在佈建容量。
已開始	該應用程序已準備好接受新工作。應用程式只會在處於此狀態時接受工作。
Stopping (正在停止)	所有工作都已完成，應用程式正在釋放其容量。

州	描述
已停止	應用程式已停止，應用程式上沒有資源正在執行。您可以修改應用程式以變更其初始容量組態。
已終止	應用程式已終止，不會出現在您的應用程式清單中。

下圖顯示EMR無伺服器應用程式狀態的軌跡。



從 EMR Studio 主控台與您的應用程式互動

您可以從 EMR Studio 主控台建立、檢視和管理EMR無伺服器應用程式。若要瀏覽至 EMR Studio 主控台，請遵循從主控台開始使用中的指示。

建立應用程式

使用「建立應用程式」頁面，您可以依照下列步驟建立EMR無伺服器應用程式。

1. 在「名稱」欄位中，輸入您要呼叫應用程式的名稱。
2. 在「類型」欄位中，選擇 Spark 或 Hive 作為應用程式的類型。
3. 在「核發版本」欄位中，選擇EMR核發編號。
4. 在 [架構] 選項中，選擇要使用的指令集架構。如需詳細資訊，請參閱Amazon EMR 無伺服器架構選項。

   - arm64 — 64 位元ARM架構；使用重力子處理器
   - 64 位元 x86 架構；使用 x86 架構；使用以 x86 為基礎的處理器

5. 其餘欄位有兩個應用程式設定選項：預設設定和自訂設定。這些欄位是選擇性的。

   預設設定 — 預設設定可讓您以預先初始化的容量快速建立應用程式。這包括一個驅動程序和一個執行人的星火, 和一個驅動程序和一個 Tez 任務蜂巢. 預設設定不會啟用您的VPCs. 該應用程序配置為在閒置 15 分鐘時停止，並在工作提交時自動啟動。

   自訂設定 — 自訂設定可讓您修改下列內容。

   - 預先初始化的容量 — 驅動程式和執行者或 Hive Tez Task 工作者的數量，以及每個工作者的大小。
   - 應用程式限制 — 應用程式的最大容量。
   - 應用程式行為 — 應用程式的自動啟動和自動停止行為。
   - 網路連線 — VPC 資源的網路連線。
   - 標籤 — 您可以指派給應用程式的自訂標籤。

   如需有關預先初始化容量、應用程式限制和應用程式行為的詳細資訊，請參閱設定應用程式。如需有關網路連線的更多資訊，請參閱設定VPC存取權。

6. 若要建立應用程式，請選擇建立應用程式。

列出應用

您可以從清單應用程式頁面檢視所有現有的EMR無伺服器應用程式。您可以選擇應用程式的名稱，導覽至該應用程式的「詳細資訊」頁面。

管理應用程式

您可以從 [列出應用程式] 頁面或特定應用程式的 [詳細資訊] 頁面，對應用程式執行下列動作。

開始申請

選擇此選項以手動啟動應用程式。

停止申請

選擇此選項可手動停止應用程式。應用程式應該沒有執行中的作業，才能停止。若要深入瞭解應用程式狀態轉換，請參閱應用狀態。

配置應用

從設定應用程式頁面編輯應用程式的選擇性設定。您可以變更大部分應用程式設定。例如，您可以變更應用程式的版本標籤，將其升級到不同版本的 AmazonEMR，或者您可以將架構從 x86_64 切換到 arm64。其他選擇性設定與 [建立應用程式] 頁面上 [自訂設定] 區段中的設定相同。如需應用程式設定的詳細資訊，請參閱建立應用程式。

刪除申請

選擇此選項可手動刪除應用程式。您必須停止應用程式才能刪除該應用程式。若要深入瞭解應用程式狀態轉換，請參閱應用狀態。

與您的應用程式互動 AWS CLI

從 AWS CLI，您可以建立、描述和刪除個別應用程式。您還可以列出所有應用程序，以便查看它們一目了然。本節說明如何執行這些動作。如需更多應用程式作業 (例如啟動、停止和更新應用程式)，請參閱EMR無伺服器API參考。如需如何使用EMR無伺服器的API範例 AWS SDK for Java，請參閱我們 GitHub 存放庫中的 Java 範例。如需如何使用EMR無伺服器的API範例 AWS SDK for Python (Boto)，請參閱我們 GitHub 儲存庫中的 Python 範例。

若要建立應用程式，請使用`create-application`。您必須指定`SPARK`或`HIVE`作為應用程式`type`。此命令返回應用程序的ARN，名稱和 ID。

```
aws emr-serverless create-application \
--name my-application-name \
--type 'application-type' \
--release-label release-version
```

若要描述應用程式，請使用`get-application`並提供應用程式`application-id`。此命令會傳回應用程式的狀態和容量相關組態。

```
aws emr-serverless get-application \
--application-id application-id
```

若要列出所有應用程式，請撥打`list-applications`。此命令會傳回與所有應用程式相同的屬性，`get-application`但包含所有應用程式。

```
aws emr-serverless list-applications
```

要刪除您的應用程序，請致電delete-application並提供您的application-id.

```
aws emr-serverless delete-application \
--application-id application-id
```

設定應用程式

使用EMR無伺服器，您可以設定您使用的應用程式。例如，您可以設定應用程式可擴充至的最大容量、設定預先初始化的容量，讓驅動程式和 Worker 隨時回應，以及在應用程式層級指定一組通用的執行階段和監視組態。下列頁面說明如何在使用EMR無伺服器時設定應用程式。

主題

- 瞭解應用程式行
- 預先初始化容量
- EMR無伺服器的預設應用程式組態

瞭解應用程式行

預設應用行為

自動啟動 — 預設情況下，應用程式設定為在工作提交時自動啟動。您可以關閉此功能。

自動停止 — 預設情況下，應用程式設定為閒置 15 分鐘時自動停止。當應用程式變更為STOPPED狀態時，會釋放任何已設定的預先初始化容量。您可以在應用程式自動停止之前修改閒置時間長度，也可以關閉此功能。

容量上限

您可以設定應用程式可擴充至的最大容量。您可以根據記憶體 (GB) 和磁碟 (GB) 來指定最大容量。CPU

> **Note**
>
> 我們建議您將最大容量設定為與支援的工作者大小成比例，方法是將 Worker 數目乘以其大小。例如，如果您要將應用程式限制為 50 個背景工作者 (含 2 個vCPUs、16 GB 的記憶體和 20 GB 的磁碟)，請將您的最大容量設定為 100 vCPUs、800 GB (記憶體)，磁碟容量為 1000 GB。

支援的 Worker 組態

下表顯示您可為EMR無伺服器指定的支援 Worker 組態和大小。您可以根據工作負載的需求為驅動程式和執行程式設定不同的大小。

CPU	記憶體	預設暫時儲存
1 伏 CPU	最少 2 GB，最多 8 GB，以 1 GB 為單位增量	二十 GB
2 伏 CPU	最低 4 GB，最多 16 GB，以 1 GB 為單位增量	二十 GB
四伏 CPU	至少 8 GB，最多 30 GB，以 1 GB 為單位增量	二十 GB
八伏 CPU	最低 16 GB，最大 60 GB，以 4 GB 為單位增量	二十 GB
英格蘭 CPU	最低 32 GB，最多 120 GB，以 8 GB 為單位計算	二十 GB

CPU-每個工人可以有 1，2，4，8 或 16 vCPUs。

記憶體 — 每個 Worker 的記憶體 (以 GB 為單位指定) 都在先前表格中列出的限制範圍內。Spark 作業會產生記憶體額外負荷，這表示它們使用的記憶體超過指定的容器大小。此開銷是使用屬性`spark.driver.memoryOverhead`和指定的`spark.executor.memoryOverhead`。額外負荷的預設值為 10% 的容器記憶體，最少為 384 MB。當您選擇背景工作大小時，您應該考慮這個負荷。

例如，如果您vCPUs 為背景工作執行個體選擇 4，而預先初始化的儲存容量為 30 GB，則應將大約 27 GB 的值設定為 Spark 工作的執行程式記憶體。這樣可以最大限度地提高預先初始化容量的使用率。可用記憶體大小為 27 GB，再加上 27 GB (2.7 GB) 的 10%，總共 29.7 GB。

磁碟 — 您可以為每個工作者設定最小大小為 20 GB 且最大 200 GB 的暫存儲磁碟。您只需為每位工作者設定的 20 GB 以上的額外儲存空間付費。

預先初始化容量

EMR無伺服器提供選用功能，可讓驅動程式和 Worker 預先初始化，並在幾秒鐘內做好回應的準備。這有效地為應用程序創建了一個溫暖的工作人員池。此功能稱為預先初始化容量。若要設定此功能，您可以將應用程式的`initialCapacity`參數設定為您要預先初始化的 Worker 數目。使用預先初始化的 Worker 容量，工作會立即開始。當您想要實現迭代應用程序和時間敏感性工作時，這是理想的選擇。

當您提交工作時，如果工作者可`initialCapacity`用，則工作會使用這些資源來開始執行。如果這些 Worker 已被其他工作所使用，或者工作所需的資源多於可用資源`initialCapacity`，則應用程式會要求並取得額外的工作程式，直到為應用程式設定的資源上限為止。當工作完成執行時，會釋放其使用的 Worker，以及應用程式可用的資源數目`initialCapacity`。即使在作業完成執行之後，應用程式仍會維護資源。`initialCapacity`當工作不再需要執行`initialCapacity`時，應用程式會釋放多餘的資源。

預先初始化的容量可供使用，並可在應用程式啟動時使用。當應用程式停止時，預先初始化的容量會變為非作用中。只有在要求的預先初始化容量已建立且可供使用時，應用程式才會移至狀`STARTED`態。在整個應用程式處於`STARTED`狀態時，EMRServerless 會保留預先初始化的容量，以供工作或互動式工作負載使用或使用。此功能還原已發行或故障容器的容量。這會維護`InitialCapacity`參數指定的 Worker 數目。沒有預先初始化容量的應用程式狀態可以立即從變更`CREATED`為`STARTED`。

您可以將應用程式設定為在特定時間內未使用預先初始化的容量 (預設值為 15 分鐘) 釋放該應用程式。當您提交新工作時，已停止的應用程式會自動啟動。您可以在建立應用程式時設定這些自動啟動和停止組態，也可以在應用程式處於`CREATED`或`STOPPED`狀態時變更它們。

您可以變更`InitialCapacity`計數，並為CPU每個 Worker 指定計算組態，例如記憶體和磁碟。因為您無法進行部分修改，因此您應該在變更值時指定所有計算組態。您只能在應用程式處於`CREATED`或`STOPPED`狀態時變更組態。

> **Note**
>
> 若要最佳化應用程式對資源的使用，我們建議您將容器大小與預先初始化的容量背景工作者大小保持一致。例如，如果您將 Spark 執行程式大小設定為 2，將記憶體設定為 8 GB，但預先初始化的容量背景工作者大小為 4，CPUs且CPUs具有 16 GB 記憶體，則 Spark 執行程式在指派給此工作時，只會使用一半的工作者資源。

自訂 Spark 和 Hive 的預先初始化容量

您可以針對在特定巨量資料架構上執行的工作負載進一步自訂預先初始化容量。例如，當工作負載在 Apache Spark 上執行時，您可以指定有多少工作程式以驅動程式的身分開始，以及以執行者身分開始的工作人數。同樣地，當您使用 Apache Hive 時，您可以指定有多少工作者開始做為 Hive 驅動程式，以及應執行 Tez 工作的數量。

使用預先初始化的容量設定執行 Apache Hive 的應用

以下API請求創建一個基於 Amazon EMR 版本 emr-6.6.0 運行 Apache 配置單元的應用程序。該應用程序從 5 個預先初始化的 Hive 驅動程序開始，每個驅動程序具有 2 v CPU 和 4 GB 的內存，以及 50 個預先初始化的 Tez 任務工作線程，每個驅動程序具有 4 v CPU 和 8 GB 的內存。當 Hive 查詢在此應用程序上運行時，他們首先使用預先初始化的 Worker 並立即開始執行。如果所有預先初始化的工作程式都忙碌中，而且提交了更多 Hive 工作，則應用程式可擴充到總共 400 v CPU 和 1024 GB 的記憶體。您可以選擇省略DRIVER或 TEZ_TASK Worker 的容量。

```
aws emr-serverless create-application \
  --type "HIVE" \
  --name my-application-name \
  --release-label emr-6.6.0 \
  --initial-capacity '{
    "DRIVER": {
        "workerCount": 5,
        "workerConfiguration": {
            "cpu": "2vCPU",
            "memory": "4GB"
        }
    },
    "TEZ_TASK": {
        "workerCount": 50,
        "workerConfiguration": {
            "cpu": "4vCPU",
            "memory": "8GB"
        }
    }
  }' \
  --maximum-capacity '{
    "cpu": "400vCPU",
    "memory": "1024GB"
  }'
```

使用預先初始化的容量設定執行 Apache Spark 的應用

以下API請求創建一個基於 Amazon EMR 版本 6.6.0 運行阿帕奇星火 3.2.0 的應用程序。應用程式從 5 個預先初始化的 Spark 驅動程式開始，每個驅動程式都有 2 v CPU 和 4 GB 的記憶體，以及 50 個預先初始化的執行程式，每個執行程式都有 4 v CPU 和 8 GB 的記憶體。當 Spark 作業在此應用程式上執行時，它們會先使用預先初始化的 Worker 並立即開始執行。如果所有預先初始化的工作程式都忙碌中，而且提交了更多 Spark 工作，則應用程式可以擴充到總共 400 v CPU 和 1024 GB 的記憶體。您可以選擇省略DRIVER或的容量EXECUTOR。

> **Note**
>
> Spark 為驅動程序和執行程序請求的內存添加了可配置的內存開銷，默認值為 10%。若要使用預先初始化 Worker 的工作，初始容量記憶體組態應大於工作和額外負荷要求的記憶體。

```
aws emr-serverless create-application \
  --type "SPARK" \
  --name my-application-name \
  --release-label emr-6.6.0 \
  --initial-capacity '{
    "DRIVER": {
        "workerCount": 5,
        "workerConfiguration": {
            "cpu": "2vCPU",
            "memory": "4GB"
        }
    },
    "EXECUTOR": {
        "workerCount": 50,
        "workerConfiguration": {
            "cpu": "4vCPU",
            "memory": "8GB"
        }
    }
  }' \
  --maximum-capacity '{
    "cpu": "400vCPU",
    "memory": "1024GB"
  }'
```

EMR無伺服器的預設應用程式組態

您可以針對在相同應用程式下送出的所有工作，在應用程式層次指定一組通用的執行時間和監視組態。如此可減少需要為每個工作送出相同組態所產生的額外負荷。

您可以在下列時間點修改模型組態：

- 在工作提交時宣告應用程式層級組態。
- 在工作執行期間覆寫預設組態。

下列各節提供更多詳細資訊，以及進一步前後關聯的範例。

在應用程式層級宣告組態

您可以針對在應用程式下送出的工作，指定應用程式層次記錄日誌和程式實際執行組態特性。

`monitoringConfiguration`

若要為您透過應用程式送出的工作指定記錄組態，請使用`monitoringConfiguration`欄位。如需EMR無伺服器記錄的詳細資訊，請參閱儲存記錄。

`runtimeConfiguration`

若要指定執行時期組態特性`spark-defaults`，例如，請在`runtimeConfiguration`欄位中提供組態物件。這會影響您隨應用程式送出之所有工作的預設組態。如需詳細資訊，請參閱 蜂巢配置覆蓋參數 和 星火配置覆蓋參數。

可用的組態分類會因特定的EMR無伺服器版本而異。例如，自訂 Log4j 的分類，`spark-driver-log4j2`且僅`spark-executor-log4j2`適用於 6.8.0 及更高版本。如需應用程式特定性質的清單，請參閱火花工作屬性和蜂巢工作屬性。

您還可以配置阿帕奇 Log4j2 屬性，AWS Secrets Manager 用於數據保護，並在應用程序級別使用 Java 17 運行時。

若要在應用程式層級傳遞 Secrets Manager 密碼，請將下列原則附加至需要建立或更新具有密碼的EMR無伺服器應用程式的使用者和角色。

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

```
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:secretsmanager:your-secret-arn"
        }
    ]
}
```

如需建立密碼自訂原則的詳細資訊，請參閱下列項目的權限原則範例 AWS Secrets Manager 中的 AWS Secrets Manager 用戶指南。

> **Note**
>
> 您在runtimeConfiguration應用程式層級指定的對映至applicationConfiguration中StartJobRunAPI。

宣告範例

下列範例會示範如何使用宣告預設組態create-application。

```
aws emr-serverless create-application \
    --release-label release-version  \
    --type SPARK \
    --name my-application-name \
    --runtime-configuration '[
        {
            "classification": "spark-defaults",
            "properties": {
                "spark.driver.cores": "4",
                "spark.executor.cores": "2",
                "spark.driver.memory": "8G",
                "spark.executor.memory": "8G",
                "spark.executor.instances": "2",

 "spark.hadoop.javax.jdo.option.ConnectionDriverName":"org.mariadb.jdbc.Driver",
                "spark.hadoop.javax.jdo.option.ConnectionURL":"jdbc:mysql://db-host:db-
port/db-name",
```

```
                "spark.hadoop.javax.jdo.option.ConnectionUserName":"connection-user-
name",
                "spark.hadoop.javax.jdo.option.ConnectionPassword":
 "EMR.secret@SecretID"
            }
        },
        {
            "classification": "spark-driver-log4j2",
            "properties": {
                "rootLogger.level":"error",
                "logger.IdentifierForClass.name": "classpathForSettingLogger",
                "logger.IdentifierForClass.level": "info"
            }
        }
    ]' \
    --monitoring-configuration '{
        "s3MonitoringConfiguration": {
            "logUri": "s3://DOC-EXAMPLE-BUCKET-LOGGING/logs/app-level"
        },
        "managedPersistenceMonitoringConfiguration": {
            "enabled": false
        }
    }'
```

在工作執行期間覆寫組態

您可以使用指定應用程式組態和監視組態的組態覆寫StartJobRunAPI。EMR然後，無伺服器會合併您在應用程式層級和工作層級指定的組態，以決定工作執行的組態。

合併發生時的粒度級別如下：

- **ApplicationConfiguration**-例如，分類類型spark-defaults。
- **MonitoringConfiguration**-配置類型，例如s3MonitoringConfiguration。

> **Note**
>
> 您提供的組態優先順序會StartJobRun取代您在應用程式層級提供的組態。

如需優先順序排名的詳細資訊，請參閱蜂巢配置覆蓋參數和星火配置覆蓋參數。

當您啟動工作時，如果您沒有指定特定的組態，它將會從應用程式繼承。如果您在工作層次宣告組態，您可以執行下列作業：

- 覆寫現有的組態-使用覆寫值提供StartJobRun請求中相同的組態參數。
- 新增其他組態-使用您要指定的值，在要StartJobRun求中新增組態參數。
- 移除現有的組態-若要移除應用程式執行階段組態，請提供您要移除之組態的索引鍵，然後傳遞空白宣告以{}供配置使用。我們不建議移除任何包含工作執行所需參數的分類。例如，如果您嘗試移除 Hive 工作所需的屬性，工作將會失敗。

  若要移除應用程式監督組態，請針對相關組態類型使用適當的方法：

  - **cloudWatchLoggingConfiguration**-要刪除cloudWatchLogging，請將啟用的標誌傳遞為false。
  - **managedPersistenceMonitoringConfiguration**-若要移除受管理的持續性設定並回復至預設的啟用狀態，{}請傳遞組態的空白宣告。
  - **s3MonitoringConfiguration**-若要移除s3MonitoringConfiguration，{}請為組態傳遞空白宣告。

覆寫範例

下列範例顯示您在工作提交期間可以執行的不同作業start-job-run。

```
aws emr-serverless start-job-run \
    --application-id your-application-id \
    --execution-role-arn your-job-role-arn \
    --job-driver '{
        "sparkSubmit": {
            "entryPoint": "s3://us-east-1.elasticmapreduce/emr-containers/samples/
wordcount/scripts/wordcount.py",
            "entryPointArguments": ["s3://DOC-EXAMPLE-BUCKET-OUTPUT/wordcount_output"]
        }
    }' \
    --configuration-overrides '{
        "applicationConfiguration": [
            {
                // Override existing configuration for spark-defaults in the
 application
                "classification": "spark-defaults",
                "properties": {
                    "spark.driver.cores": "2",
```

```
                    "spark.executor.cores": "1",
                    "spark.driver.memory": "4G",
                    "spark.executor.memory": "4G"
                }
            },
            {
                // Add configuration for spark-executor-log4j2
                "classification": "spark-executor-log4j2",
                "properties": {
                    "rootLogger.level": "error",
                    "logger.IdentifierForClass.name": "classpathForSettingLogger",
                    "logger.IdentifierForClass.level": "info"
                }
            },
            {
                // Remove existing configuration for spark-driver-log4j2 from the
application
                "classification": "spark-driver-log4j2",
                "properties": {}
            }
        ],
        "monitoringConfiguration": {
            "managedPersistenceMonitoringConfiguration": {
                // Override existing configuration for managed persistence
                "enabled": true
            },
            "s3MonitoringConfiguration": {
                // Remove configuration of S3 monitoring
            },
            "cloudWatchLoggingConfiguration": {
                // Add configuration for CloudWatch logging
                "enabled": true
            }
        }
    }'
```

在工作執行時，會根據和中所述的優先順序覆寫等級套用下列分類蜂巢配置覆蓋參數和組態星火配置覆蓋參數。

- 分類spark-defaults將使用在工作層級指定的屬性進行更新。此分類只StartJobRun會考慮中包含的性質。
- 分類spark-executor-log4j2將新增至現有的分類清單中。

- 該分類spark-driver-log4j2將被刪除。
- 的組態managedPersistenceMonitoringConfiguration將以工作層級的組態更新。
- 的組態s3MonitoringConfiguration將被移除。
- 的組態cloudWatchLoggingConfiguration將新增至現有的監視組態。

自訂EMR無伺服器映像

從 Amazon EMR 6.9.0 開始，您可以使用自訂映像，透過 Amazon EMR 無伺服器將應用程式相依性和執行階段環境封裝到單一容器中。這樣可簡化您管理工作負載相依性的方式，並使套件更具可攜性。當您自訂EMR無伺服器映像時，它具有下列優點：

- 安裝和設定針對您的工作負載最佳化的套件。這些套件可能無法在 Amazon EMR 執行階段環境的公開發佈中廣泛使用。
- 將EMR無伺服器與組織內目前建立的建置、測試和部署程序整合，包括本機開發與測試。
- 套用已建立的安全性程序，例如影像掃描，以符合組織內的合規性和治理需求。
- 可讓您在應用程式中使用自己的JDK和 Python 版本。

EMR無伺服器提供的映像檔可讓您在建立自己的映像檔時做為基礎使用。基本映像檔提供必要的 jar、組態和程式庫，讓影像與EMR無伺服器互動。您可以在 Amazon ECR 公共畫廊中找到基本圖像。使用與您的應用程序類型（Spark 或 Hive）和發布版本匹配的映像。例如，如果您在 Amazon 6.9.0 EMR 版本上建立應用程式，請使用下列影像。

Type	映像
Spark	public.ecr.aws/emr-serverless/spark/emr-6.9.0:latest
Hive	public.ecr.aws/emr-serverless/hive/emr-6.9.0:latest

必要條件

建立EMR無伺服器自訂映像之前，請先完成這些先決條件。

1. 在同一個創建一個 Amazon ECR 存儲庫 AWS 區域 您用來啟動EMR無伺服器應用程式。若要建立 Amazon ECR 私有存放庫，請參閱建立私有存放庫。
2. 若要授與使用者對 Amazon ECR 儲存庫的存取權限，請將下列政策新增至使用者和角色，這些使用者和角色使用此儲存庫中的映像建立或更新EMR無伺服器

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ECRRepositoryListGetPolicy",
            "Effect": "Allow",
            "Action": [
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "ecr:DescribeImages"
            ],
            "Resource": "ecr-repository-arn"
        }
    ]
}
```

如需 Amazon ECR 身分型政策的更多範例，請參閱 Amazon 彈性容器登錄以身分識別為基礎的政策範例。

步驟 1：從EMR無伺服器基礎映像檔建立自訂映像

首先，創建一個 Docker 文件，該文件以使用您首選的基本映像的FROM指令開頭。FROM說明完成後，您可以包含要對影像進行的任何修改。基本影像會自動USER將設定為hadoop。此設定可能沒有您包含的所有修改的權限。因應措施是將設USER定為root，修改映像，然後將設定USER回hadoop:hadoop。若要查看常見使用案例的範例，請參閱搭配EMR無伺服器使用自訂映像。

```
# Dockerfile
FROM public.ecr.aws/emr-serverless/spark/emr-6.9.0:latest

USER root
# MODIFICATIONS GO HERE

# EMRS will run the image as hadoop
```

```
USER hadoop:hadoop
```

你有碼頭文件後，使用以下命令構建圖像。

```
# build the docker image
docker build . -t aws-account-id.dkr.ecr.region.amazonaws.com/my-
repository[:tag]or[@digest]
```

步驟 2：在本機驗證映像

EMR無伺服器提供離線工具，可以靜態檢查您的自訂映像檔，以驗證基本檔案、環境變數和正確的映像設定。如需有關如何安裝和執行工具的資訊，請參閱 Amazon EMR 無伺服器映像CLI GitHub。

安裝工具之後，請執行下列命令來驗證映像檔：

```
amazon-emr-serverless-image \
validate-image -r emr-6.9.0 -t spark \
-i aws-account-id.dkr.ecr.region.amazonaws.com/my-repository:tag/@digest
```

您應該會看到類似下列內容的輸出。

```
Amazon EMR Serverless - Image CLI
Version: 0.0.1
... Checking if docker cli is installed
... Checking Image Manifest
[INFO] Image ID: 9e2f4359cf5beb466a8a2ed047ab61c9d37786c555655fc122272758f761b41a
[INFO] Created On: 2022-12-02T07:46:42.586249984Z
[INFO] Default User Set to hadoop:hadoop : PASS
[INFO] Working Directory Set to  : PASS
[INFO] Entrypoint Set to /usr/bin/entrypoint.sh : PASS
[INFO] HADOOP_HOME is set with value: /usr/lib/hadoop : PASS
[INFO] HADOOP_LIBEXEC_DIR is set with value: /usr/lib/hadoop/libexec : PASS
[INFO] HADOOP_USER_HOME is set with value: /home/hadoop : PASS
[INFO] HADOOP_YARN_HOME is set with value: /usr/lib/hadoop-yarn : PASS
[INFO] HIVE_HOME is set with value: /usr/lib/hive : PASS
[INFO] JAVA_HOME is set with value: /etc/alternatives/jre : PASS
[INFO] TEZ_HOME is set with value: /usr/lib/tez : PASS
[INFO] YARN_HOME is set with value: /usr/lib/hadoop-yarn : PASS
[INFO] File Structure Test for hadoop-files in /usr/lib/hadoop: PASS
[INFO] File Structure Test for hadoop-jars in /usr/lib/hadoop/lib: PASS
[INFO] File Structure Test for hadoop-yarn-jars in /usr/lib/hadoop-yarn: PASS
[INFO] File Structure Test for hive-bin-files in /usr/bin: PASS
```

```
[INFO] File Structure Test for hive-jars in /usr/lib/hive/lib: PASS
[INFO] File Structure Test for java-bin in /etc/alternatives/jre/bin: PASS
[INFO] File Structure Test for tez-jars in /usr/lib/tez: PASS
---------------------------------------------------------------
Overall Custom Image Validation Succeeded.
---------------------------------------------------------------
```

第 3 步：將圖像上傳到您的 Amazon ECR 存儲庫

使用以下命令將您的 Amazon ECR 映像推送到您的 Amazon ECR 存儲庫。確保您具有將映像推送到存放庫的正確IAM權限。如需詳細資訊，請參閱 Amazon ECR 使用者指南中的推送映像。

```
# login to ECR repo
aws ecr get-login-password --region region | docker login --username AWS --password-
stdin aws-account-id.dkr.ecr.region.amazonaws.com

# push the docker image
docker push aws-account-id.dkr.ecr.region.amazonaws.com/my-repository:tag/@digest
```

步驟 4：使用自訂影像建立或更新應用程式

選擇 AWS Management Console 標籤或 AWS CLI 根據您要如何啟動應用程序選項卡，然後完成以下步驟。

Console

1. 在 https://console.aws.amazon.com/em r 登入EMR工作室主控台。瀏覽至您的應用程式，或使用建立應用程式中的指示建立新的應用程式。
2. 若要在建立或更新EMR無伺服器應用程式時指定自訂影像，請在應用程式設定選項中選取「自訂設定」。
3. 在 [自訂映像設定] 區段中，選取 [搭配此應用程式使用自訂映像] 核取方塊。
4. 將 Amazon 圖ECR像粘貼URI到「圖像 URI」字段中。EMR無伺服器會將此映像用於應用程式的所有 Worker 類型。或者，您可以選擇不同的自訂影像，並URIs為每個工作者類型貼上不同的 Amazon ECR 映像。

CLI

- 使用`image-configuration`參數建立應用程式。EMR無伺服器會將此設定套用至所有 Worker 類型。

```
aws emr-serverless create-application \
--release-label emr-6.9.0 \
--type SPARK \
--image-configuration '{
    "imageUri": "aws-account-id.dkr.ecr.region.amazonaws.com/my-repository:tag/
@digest"
}'
```

若要針對每個 Worker 類型建立具有不同影像設定的應用程式，請使用worker-type-specifications參數。

```
aws emr-serverless create-application \
--release-label emr-6.9.0 \
--type SPARK \
--worker-type-specifications '{
    "Driver": {
        "imageConfiguration": {
            "imageUri": "aws-account-id.dkr.ecr.region.amazonaws.com/my-
repository:tag/@digest"
        }
    },
    "Executor" : {
        "imageConfiguration": {
            "imageUri": "aws-account-id.dkr.ecr.region.amazonaws.com/my-
repository:tag/@digest"
        }
    }
}'
```

若要更新應用程式，請使用image-configuration參數。EMR無伺服器會將此設定套用至所有 Worker 類型。

```
aws emr-serverless update-application \
--application-id application-id \
--image-configuration '{
    "imageUri": "aws-account-id.dkr.ecr.region.amazonaws.com/my-repository:tag/
@digest"
}'
```

步驟 5：允許EMR無伺服器存取自訂影像儲存庫

將下列資源政策新增至 Amazon ECR 儲存庫，以允許EMR無伺服器服務主體使用來自此儲存庫的getdescribe、和download請求。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Emr Serverless Custom Image Support",
      "Effect": "Allow",
      "Principal": {
        "Service": "emr-serverless.amazonaws.com"
      },
      "Action": [
        "ecr:BatchGetImage",
        "ecr:DescribeImages",
        "ecr:GetDownloadUrlForLayer"
      ],
      "Condition":{
        "StringEquals":{
          "aws:SourceArn": "arn:aws:emr-serverless:region:aws-account-id:/
applications/application-id"
        }
      }
    }
  ]
}
```

安全性最佳作法是將aws:SourceArn條件金鑰新增至儲存庫原則。IAM全域條件金鑰aws:SourceArn可確保EMR無伺服器僅針對應用程式ARN使用儲存庫。如需 Amazon ECR 儲存庫政策的詳細資訊，請參閱建立私有存放庫。

考量與限制

使用自訂映像時，請考慮下列事項：

- 為您的應用程式使用符合類型 (Spark 或 Hive) 和版本標籤 (例如emr-6.9.0) 的正確基礎影像。
- EMR無伺服器會忽略 Docker 檔案中的[ENTRYPOINT]指示[CMD]或指示。使用 Docker 檔案中的一般指示，例如[COPY][RUN]、和[WORKDIR]。
- 建立自訂映像檔TEZ_HOME時 JAVA_HOMESPARK_HOME，不應修改HIVE_HOME、、的環境變數。

- 自訂映像檔的大小不得超過 5 GB。
- 如果修改 Amazon EMR 基礎映像中的二進位檔或 jar，可能會導致應用程式或任務啟動失敗。
- Amazon ECR 存儲庫應該在相同 AWS 區域 您用來啟動EMR無伺服器應用程式。

設定VPC存取權

您可以將EMR無伺服器應用程式設定為連接到您內部的資料存放區VPC，例如 Amazon Redshift 叢集、Amazon 資料RDS庫或具VPC有端點的 Amazon S3 儲存貯體。您的EMR無伺服器應用程式具有對外連線至. VPC 根據預設，EMR無伺服器會封鎖對應用程式的入站存取，以提升安全性。

> **Note**
>
> 如果您想要為應用程式使用外部 Hive 中繼VPC存放區資料庫，則必須設定存取權。如需如何設定外部 Hive 中繼存放區的相關資訊，請參閱中繼存放區設定。

建立應用程式

在 [建立應用程式] 頁面上，您可以選擇自訂設定VPC，並指定EMR無伺服器應用程式可以使用的子網路和安全群組。

VPCs

選擇包含資料存放區的虛擬私有雲 (VPC) 名稱。[建立應用程式] 頁面會列VPCs出所有您選擇的 AWS 區域.

子網

選擇包含您資料倉庫的子網路。VPC「建立應用程式」頁面會列出您VPC中資料倉庫的所有子網路。

選取的子網路必須是私有子網路。這表示子網路的相關路由表不應該有網際網路閘道。

對於網際網路的輸出連線，子網路必須具有使用NAT閘道的輸出路由。若要設定NAT閘道，請參閱使用NAT閘道。

對於 Amazon S3 連線，子網路必須設定NAT閘道或VPC端點。若要設定 S3 VPC 端點，請參閱建立閘道端點。

用於連接到其他 AWS 服務 在 VPC (例如 Amazon DynamoDB) 之外，您必須設定VPC端點或閘道。NAT若要設定的VPC端點 AWS 服務，請參閱使用VPC端點。

員工可以VPC透過輸出流量連線至您內部的資料存放區。根據預設，EMR無伺服器會封鎖對 Worker 的入站存取，以提高安全性。

當您使用 AWS Config，EMR無伺服器會為每個工作者建立 elastic network interface 項目記錄。若要避免與此資源相關的成本，請考慮關閉 `AWS::EC2::NetworkInterface` AWS Config.

> **Note**
>
> 建議您跨多個可用區域選取多個子網路。這是因為您選擇的子網路會決定可供EMR無伺服器應用程式啟動的可用區域。每個 Worker 都會在啟動它的子網路上消耗一個 IP 位址。請確定指定的子網路具有足夠的 IP 位址，以供您計劃啟動的 Worker 數目使用。如需子網路規劃的詳細資訊，請參閱the section called “子網路規劃的最佳作法”。

安全群組

選擇一個或多個可與您的資料存放區通訊的安全群組。[建立應用程式] 頁面會列出您的VPC. EMR無伺服器會將這些安全性群組與附加至VPC子網路的彈性網路介面產生關聯。

> **Note**
>
> 建議您為EMR無伺服器應用程式建立個別的安全性群組。這使得隔離和管理網路規則更有效率。例如，若要與 Amazon Redshift 叢集通訊，您可以定義 Redshift 和EMR無伺服器安全群組之間的流量規則，如以下範例所示。

Example 範例 — 與 Amazon Redshift 叢集的通訊

1. 將輸入流量的規則從其中一個EMR無伺服器安全群組新增至 Amazon Redshift 安全群組。

Type	通訊協定	連接埠範圍	來源
全部 TCP	TCP	5439	`emr-serverless-security-group`

2. 為來自其中一個EMR無伺服器安全性群組的輸出流量新增規則。您可以使用兩種方式的其中一種來執行此動作。首先，您可以開啟所有連接埠的輸出流量。

Type	通訊協定	連接埠範圍	目的地
所有流量	TCP	ALL	0.0.0.0/0

或者，您也可以將輸出流量限制為 Amazon Redshift 叢集。只有當應用程式必須與 Amazon Redshift 叢集進行通訊時，此功能才有用。

Type	通訊協定	連接埠範圍	來源
全部 TCP	TCP	5439	`redshift-security-group`

配置應用

您可以從「設定」應用程式頁面變更現有EMR無伺服器應用程式的網路組態。

檢視工作執行詳情

在 [Job 執行詳細資料] 頁面上，您可以檢視工作針對特定執行所使用的子網路。請注意，工作只會在從指定子網路選取的一個子網路中執行。

子網路規劃的最佳作法

AWS 資源是在子網路中建立的，子網路是 Amazon 中可用 IP 位址的子集VPC。例如，具有 /16 網路遮罩VPC的可用 IP 位址最多有 65,536 個可用 IP 位址，這些 IP 位址可以使用子網路遮罩分成多個較小的網路。例如，您可以將此範圍分成兩個子網路，每個子網路都使用 /17 遮罩和 32,768 個可用 IP 位址。子網路位於可用區域內，且無法跨區域。

子網路的設計應牢記您的EMR無伺服器應用程式擴展限制。例如，如果您有一個應用程式要求 4 個 vCpu 工作者，而且最多可擴充至 4,000 個vCpu，則您的應用程式最多需要 1,000 個工作者，才能使用 1,000 個網路介面。建議您跨多個可用區域建立子網路。如此一來，當可用區域故障時，EMR無伺服器就能在不太可能的情況下，在不同的可用區域中重試您的工作或佈建預先初始化的容量。因此，至少兩個可用區域中的每個子網路應具有 1,000 個以上的可用 IP 位址。

您需要遮罩大小小小於或等於 22 的子網路，才能佈建 1,000 個網路介面。任何大於 22 的口罩將不符合要求。例如，/23 的子網路遮罩提供 512 個 IP 位址，而 /22 的遮罩則提供 1024 個，而 /21 的遮罩則提供 2048 個 IP 位址。以下是可配置給不同可用區域的 /16 網路遮罩中具有 /22 遮罩VPC的 4 個子網路範例。可用和可用的 IP 位址之間有五個差異，因為每個子網路中的前四個 IP 位址和最後一個 IP 位址是由 AWS.

子網路 ID	子網路位址	子網路遮罩	IP 地址範圍	可用的 IP 位址	可用的 IP 位址
1	10.0.0.0	255.255.252.0/22	10.0.0.0-10.0.3.255	1,024	1,019
2	10.0.4.0	255.255.252.0/22	10.0.4.0-10.0.7.255	1,024	1,019
3	10.0.8.0	255.255.252.0/22	10.0.4.0-10.0.7.255	1,024	1,019
4	10.0.12.0	255.255.252.0/22	10.0.12.0-10.0.15.255	1,024	1,019

您應該評估您的工作負載是否最適合較大的工作者規模。使用較大的 Worker 大小需要較少的網路介面。例如，使用 16 個vCpu 工作程式的應用程式擴展限制為 4,000 個，最多 vCpu 需要 250 個工作程式，總共 250 個可用 IP 位址才能佈建網路介面。您需要遮罩大小小小於或等於 24 的多個可用區域中的子網路，才能佈建 250 個網路介面。任何大於 24 的遮罩大小都會提供少於 250 個 IP 位址。

如果您在多個應用程式之間共用子網路，則每個子網路都應該牢記所有應用程式的集體擴展限制。例如，如果您有 3 個應用程式要求 4 個 vCpu 工作程式，而且每個應用程式最多可以擴展到 4000 個，且擁 vCpu 有 12,000 個vCpu帳戶層級服務配額，則每個子網路將需要 3000 個可用 IP 位址。如果您VPC要使用的 IP 位址數量不足，請嘗試增加可用 IP 位址的數量。您可以通過將其他無類域間路由（CIDR）塊與. VPC 如需詳細資訊，請參閱 Amazon VPC 使用者指南VPC中的將其他IPv4CIDR區塊與您的區塊建立關聯。

您可以使用線上眾多工具之一來快速產生子網路定義，並檢閱其可用的 IP 位址範圍。

Amazon EMR 無伺服器架構選項

Amazon EMR 無伺服器應用程式的指令集架構會決定應用程式用來執行任務的處理器類型。Amazon為您的應用程序EMR提供了兩種架構選項：x86_64 和 arm64。EMR無伺服器會在最新一代的執行個體可用時自動更新，因此您的應用程式可以使用較新的執行個體，而無需您付出額外的努力。

主題

使用 64 架構

x86_64 架構也稱為 64 位元或 64 位元。 x86_64 是無伺服EMR器應用程式的預設選項。此架構使用 x86 處理器，並且與大多數協力廠商工具和程式庫相容。

大多數應用程式都與 x86 硬體平台相容，並且可以在預設的 x86_ 64 架構上成功執行。不過，如果您的應用程式與 64 位元相容ARM，您可以切換至 arm64 來使用 Graviton 處理器來提升效能、運算能力和記憶體。與在 x86 架構上執行相同大小的執行個體相比，在 arm64 架構上執行執行個體的成本更低。

使用 arm64 架構（重力子）

AWS 重力子處理器是客製化設計 AWS 使用 64 位元ARM尼奧維斯核心，並利用 arm64 架構（也稱為 Arch64 或 64 位元）。ARM所以此 AWS EMR無伺服器提供的重力電子處理器系列產品包括重力 on3 與重力 2 處理器。與在 x86_64 架構上執行的同等工作負載相比，這些處理器可為 Spark 和 Hive 工作負載提供優異的價格效能。EMR無伺服器會在可用的情況下自動使用最新一代的處理器，您無需付出任何努力即可升級到最新一代的處理器。

推出具有重力彈支援的新應用程式

使用下列其中一種方法來啟動使用 arm64 架構的應用程式。

AWS CLI

若要啟動使用引力子處理器的應用程式 AWS CLI」中，指定ARM64為中的architecture參數create-applicationAPI。在其他參數中為您的應用程式提供適當的值。

```
aws emr-serverless create-application \
 --name my-graviton-app \
 --release-label emr-6.8.0 \
 --type "SPARK" \
 --architecture "ARM64" \
 --region us-west-2
```

EMR Studio

若要使用 EMR Studio 的 Graviton 處理器啟動應用程式，請在建立或更新應用程式時選擇 arm64 作為架構選項。

設定現有的應用程式以使用重力子

您可以將現有的 Amazon EMR 無伺服器應用程式設定為使用重力 (arm64) 架構搭配 SDK AWS CLI，或EMR工作室。

若要將現有的應用程式從 x86 轉換為 arm64

1. 確認您使用的是最新的主要版本 AWS CLI/SDK支持architecture參數。
2. 確認沒有工作正在執行，然後停止應用程式。

```
aws emr-serverless stop-application \
 --application-id application-id \
 --region us-west-2
```

3. 若要更新應用程式以使用 Graviton，請ARM64為中的architecture參數指定。update-application API

```
aws emr-serverless update-application \
 --application-id application-id \
 --architecture 'ARM64' \
 --region us-west-2
```

4. 若要確認應用程式的CPU架構是否為現在ARM64，請使用 get-applicationAPI.

```
aws emr-serverless get-application \
 --application-id application-id \
 --region us-west-2
```

5. 準備就緒後，請重新啟動應用程式。

```
aws emr-serverless start-application \
 --application-id application-id \
 --region us-west-2
```

使用引力子時的注意事項

在您使用 arm64 支援 Graviton 啟動EMR無伺服器應用程式之前，請先確認下列事項。

庫兼容性

當您選取 Graviton (arm64) 作為架構選項時，請確定協力廠商套件和程式庫與 64 位元架構相容。ARM 如需有關如何將 Python 程式庫封裝到與所選架構相容的 Python 虛擬環境中的資訊，請參閱搭配EMR無伺服器使用 Python 程式庫。

若要進一步了解如何將 Spark 或 Hive 工作負載設定為使用 64 位元ARM，請參閱 AWS 重力子入門存放庫上。 GitHub此儲存庫包含可協助您開始使用ARM基於 Graviton 的必要資源。

使用EMR無伺服器將資料匯入 S3 快速單一區域

在 Amazon 7.2.0 及更高EMR版本中，您可以將EMR無伺服器與 Amazon S3 Express 單區儲存類別搭配使用，以提高執行任務和工作負載時的效能。S3 Express One Zone 是一種高效能的單區域 Amazon S3 儲存類別，可為大多數對延遲敏感的應用程式提供一致、10 毫秒的資料存取。在發布時，S3 Express One Zone 提供 Amazon S3 中最低延遲和最高效能的雲端物件儲存。

必要條件

- S3 快速單一區域許可 — 當 S3 Express 單一區域初始執行動作 (例如 GETLIST，或PUT在 S3 物件上) 時，儲存類別會代表您呼叫CreateSession。您的IAM政策必須允許s3express:CreateSession權限，以便 S3A 連接器可以呼叫 CreateSessionAPI. 如需具有此許可的範例政策，請參閱 開始使用 S3 Express One Zone。
- S3A 連接器 — 若要設定 Spark 從使用 S3 快速單區儲存類別的 Amazon S3 儲存貯體存取資料，您必須使用 Apache Hadoop 連接器 S3A。 若要使用連接器，請確保所有 S3 都URIs使用s3a配置。如果沒有，您可以變更用於 s3 和 s3n 結構描述的檔案系統實作。

若要變更 s3 結構描述，請指定下列叢集組態：

```
[
  {
    "Classification": "core-site",
    "Properties": {
      "fs.s3.impl": "org.apache.hadoop.fs.s3a.S3AFileSystem",
      "fs.AbstractFileSystem.s3.impl": "org.apache.hadoop.fs.s3a.S3A"
    }
  }
]
```

若要變更 s3n 結構描述，請指定下列叢集組態：

```
[
  {
    "Classification": "core-site",
    "Properties": {
      "fs.s3n.impl": "org.apache.hadoop.fs.s3a.S3AFileSystem",
      "fs.AbstractFileSystem.s3n.impl": "org.apache.hadoop.fs.s3a.S3A"
    }
```

```
  }
]
```

開始使用 S3 Express One Zone

請按照下列步驟開始使用 S3 快速單區。

1. 建立VPC端點。將端點新增 `com.amazonaws.us-west-2.s3express`到VPC端點。
2. 請遵循 Amazon EMR 無伺服器入門操作，建立具有 Amazon EMR 版本標籤 7.2.0 或更高版本的應用程式。
3. 將您的應用程式設定為使用新建立的VPC端點、私有子網路群組和安全群組。
4. 將`CreateSession`權限新增至您的工作執行角色。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Resource": "*",
            "Action": [
                "s3express:CreateSession"
            ]
        }
    ]
}
```

5. 執行您的工作。請注意，您必須使用S3A配置來存取 S3 Express 單一區域儲存貯體。

```
aws emr-serverless start-job-run \
--application-id <application-id> \
--execution-role-arn <job-role-arn> \
--name <job-run-name> \
--job-driver '{
 "sparkSubmit": {


 "entryPoint": "s3a://<DOC-EXAMPLE-BUCKET>/scripts/wordcount.py",
 "entryPointArguments":["s3a://<DOC-EXAMPLE-BUCKET>/emr-serverless-spark/output"],
 "sparkSubmitParameters": "--conf spark.executor.cores=4
 --conf spark.executor.memory=8g --conf spark.driver.cores=4
 --conf spark.driver.memory=8g --conf spark.executor.instances=2
```

```
--conf spark.hadoop.fs.s3a.change.detection.mode=none
--conf spark.hadoop.fs.s3a.endpoint.region={<AWS_REGION>}
--conf spark.hadoop.fs.s3a.select.enabled=false
--conf spark.sql.sources.fastS3PartitionDiscovery.enabled=false
}'
```

執行任務

佈建應用程式之後，您可以將工作提交至應用程式。本節介紹如何使用 AWS CLI 以執行這些工作。本節也會識別EMR無伺服器上可用之每種應用程式類型的預設值。

主題

作業執行狀態

當您將任務執行提交至 Amazon EMR 無伺服器任務佇列時，任務執行會進入SUBMITTED狀態。工作狀態的SUBMITTED經過，RUNNING直到達FAILEDSUCCESS、或CANCELLING。

作業執行可能有以下狀態：

州	描述
已提交	將工作執行提交至EMR無伺服器時的初始工作狀態。工作會等待排定應用程式。EMR無伺服器會開始排定工作執行的優先順序和排程。
待定	排程器正在評估工作執行，以排定應用程式的執行優先順序並排定執行。

州	描述
已排程	EMR無伺服器已排定應用程式的工作執行，並且正在分配資源以執行工作。
執行中	EMR無伺服器已配置工作最初需要的資源，且工作正在應用程式中執行。在 Spark 應用程式中，這意味著 Spark 驅動程式進程處於 `running` 狀態。
失敗	EMR無伺服器無法將工作執行提交至應用程式，或未成功完成。如需有關此工作失敗`StateDetails` 的其他資訊，請參閱。
Success	工作執行已成功完成。
取消	`CancelJobRun` API已要求取消工作執行，或工作執行逾時。EMR無伺服器嘗試取消應用程式中的工作並釋放資源。
已取消	工作執行已成功取消，且已釋放其使用的資源。

從 EMR Studio 主控台執行工作

您可以將工作執行提交至EMR無伺服器應用程式，並從 EMR Studio 主控台檢視工作。若要在 EMR Studio 主控台上建立或瀏覽至EMR無伺服器應用程式，請遵循從主控台開始使用中的指示。

提交工作

在 [送出工作] 頁面上，您可以依照下列步驟將工作送出至EMR無伺服器應用程式。

Spark

1. 在「名稱」欄位中，輸入工作執行的名稱。
2. 在「執行時間角色」欄位中，輸入EMR無伺服器應用程式在工作執行時可承擔的IAM角色名稱。若要進一步瞭解執行階段角色，請參閱Amazon EMR 無伺服器的 Job 務執行階段角色。
3. 在指令碼位置欄位中，輸入指令碼或您要執行JAR的 Amazon S3 位置。對於星火作業，腳本可以是一個 Python（`.py`）文件或JAR（`.jar`）文件。

4. 如果您的指令集位置是JAR檔案，請在「主要類別」欄位中輸入作為工作進入點的類別名稱。
5. (選擇性) 輸入其餘欄位的值。

   - 指令碼引數 — 輸入要傳遞至主要JAR或 Python 指令集的任何引數。您的程式碼會讀取這些參數。以逗號分隔陣列中的每個引數。
   - 星火屬性 — 展開 Spark 屬性區段，並在此欄位中輸入任何 Spark 組態參數。

     > **Note**
     >
     > 如果您指定 Spark 驅動程序和執行程序大小，則必須考慮內存開銷。在屬性`spark.driver.memoryOverhead`和中指定記憶體額外負荷值`spark.executor.memoryOverhead`。記憶體額外負荷的預設值為 10% 的容器記憶體，最少為 384 MB。執行程序內存和內存開銷一起不能超過工作內存。例如，30 GB 工作`spark.executor.memory`站的最大值必須是 27 GB。

   - Job 組態 — 在此欄位中指定任何工作組態。您可以使用這些工作組態來覆寫應用程式的預設組態。
   - 其他設置-激活或停用 AWS Glue 資料型錄做為中繼儲存庫，並修改應用程式記錄設定。若要進一步瞭解中繼儲存區設定，請參閱中繼存儲配置。若要進一步瞭解應用程式記錄選項，請參閱儲存記錄。
   - 標籤 — 將自訂標籤指派給應用程式。
6. 選擇 Submit job (提交任務)。

Hive

1. 在「名稱」欄位中，輸入工作執行的名稱。
2. 在「執行時間角色」欄位中，輸入EMR無伺服器應用程式在工作執行時可承擔的IAM角色名稱。
3. 在指令碼位置欄位中，輸入指令碼或您要執行JAR的 Amazon S3 位置。對於 Hive 作業，指令碼必須是 Hive (`.sql`) 檔案。
4. (選擇性) 輸入其餘欄位的值。

   - 初始化指令碼位置 — 輸入 Hive 命令檔執行之前，初始化表格的指令碼位置。
   - 配置單元屬性-展開蜂房屬性部分，並在此字段中輸入任何 Hive 配置參數。

- Job 組態 — 指定任何工作組態。您可以使用這些工作組態來覆寫應用程式的預設組態。對於 Hive 作業，hive.exec.scratchdir並且hive.metastore.warehouse.dir是hive-site組態中的必要屬性。

```
{
    "applicationConfiguration": [
        {
            "classification": "hive-site",
            "configurations": [],
            "properties": {
                "hive.exec.scratchdir": "s3://DOC-EXAMPLE_BUCKET/hive/
scratch",
                "hive.metastore.warehouse.dir": "s3://DOC-EXAMPLE_BUCKET/hive/
warehouse"
            }
        }
    ],
    "monitoringConfiguration": {}
}
```

- 其他設定 — 啟用或停用 AWS Glue 資料型錄做為中繼儲存庫，並修改應用程式記錄設定。若要進一步瞭解中繼儲存區設定，請參閱中繼存儲配置。若要進一步瞭解應用程式記錄選項，請參閱儲存記錄。
- 標籤 — 將任何自訂標籤指派給應用程式。

5. 選擇 Submit job (提交任務)。

檢視任務執行

您可以從應用程式「詳細資訊」頁面上的「J ob 執行」標籤檢視工作執行，並針對工作執行執行執行執行執行以下動作。

取消工作 — 若要取消處於此RUNNING狀態的工作執行，請選擇此選項。若要進一步瞭解工作執行轉換，請參閱作業執行狀態。

複製工作 — 若要複製先前執行的工作並重新提交，請選擇此選項。

執行工作 AWS CLI

您可以建立、描述和刪除 AWS CLI。 您還可以列出所有工作以查看它們一目了然。

若要提交新工作，請使用start-job-run。提供您要執行之應用程式的識別碼，以及工作特定屬性。如需星火範例，請參閱火花工作。如需 Hive 範例，請參閱蜂巢工作。此指令會傳回您的application-idARN、和 new job-id。

每個作業執行都有設定的逾時持續時間。如果工作執行超過此持續時間，EMR無伺服器會自動將其取消。預設逾時為 12 小時。當您開始工作執行時，您可以將此逾時設定設定設定為符合工作需求的值。使用屬性配置executionTimeoutMinutes值。

```
aws emr-serverless start-job-run \
  --application-id application-id \
  --execution-role-arn job-role-arn \
  --execution-timeout-minutes 15 \
  --job-driver '{
    "hive": {
        "query": "s3://DOC-EXAMPLE-BUCKET/scripts/create_table.sql",
        "parameters": "--hiveconf hive.exec.scratchdir=s3://DOC-EXAMPLE-BUCKET/hive/
scratch --hiveconf hive.metastore.warehouse.dir=s3://DOC-EXAMPLE-BUCKET/hive/warehouse"
    }
   }' \
  --configuration-overrides '{
    "applicationConfiguration": [{
        "classification": "hive-site",
        "properties": {
            "hive.client.cores": "2",
            "hive.client.memory": "4GIB"
        }
    }]
}'
```

若要描述工作，請使用get-job-run。此指令會傳回工作特定組態和新工作的設定容量。

```
aws emr-serverless get-job-run \
--job-run-id job-id \
--application-id application-id
```

若要列出您的工作，請使用list-job-runs。此命令會傳回一組縮寫的屬性，其中包括工作類型、狀態和其他高階屬性。如果您不想查看所有工作，您可以指定要查看的最大工作數量，最多 50 個。下列範例會指定您想要查看上次執行的兩個工作。

```
aws emr-serverless list-job-runs \
--max-results 2 \
```

```
--application-id application-id
```

若要取消工作，請使用cancel-job-run。提application-id供您要取消job-id之工作的和。

```
aws emr-serverless cancel-job-run \
--job-run-id job-id \
--application-id application-id
```

如需有關如何執行工作的詳細資訊 AWS CLI，請參閱EMR無伺服器API參考。

使用隨機最佳化磁碟

在 Amazon 7.1.0 及更高EMR版本中，您可以在執行 Apache Spark 或 Hive 任務時使用隨機最佳化的磁碟，以改善 I/O 密集型工作負載的效能。與標準磁碟相比，隨機播放最佳化磁碟可提供更高 IOPS (每秒 I/O 作業數)，以加快資料移動速度並減少隨機播放作業期間的延遲。隨機最佳化磁碟可讓您連接每個 Worker 最多 2 TB 的磁碟大小，因此您可以根據工作負載需求設定適當的容量。

主要優點

隨機最佳化磁碟具有下列優點。

- 高IOPS效能 — 隨機最佳化磁碟可提供IOPS比標準磁碟更高的磁碟，在 Spark 和 Hive 任務以及其他隨機執行密集型工作負載期間，進行資料清洗效率更快速。
- 更大的磁碟大小 — 隨機最佳化磁碟支援每位工作者 20GB 到 2TB 的磁碟大小，因此您可以根據工作負載選擇適當的容量。

開始使用

請參閱下列步驟，以便在工作流程中使用隨機播放最佳化磁碟。

Spark

1. 使用下列EMR命令建立無伺服器 7.1.0 版應用程式。

```
aws emr-serverless create-application \
  --type "SPARK" \
  --name my-application-name \
  --release-label emr-7.1.0 \
```

```
  --region <AWS_REGION>
```

2. 將 Spark 工作設定為包含參數spark.emr-serverless.driver.disk.type和/或spark.emr-serverless.executor.disk.type使用隨機最佳化磁碟執行。您可以使用一個或兩個參數，具體取決於您的使用案例。

```
aws emr-serverless start-job-run \
    --application-id application-id \
    --execution-role-arn job-role-arn \
    --job-driver '{
        "sparkSubmit": {
            "entryPoint": "/usr/lib/spark/examples/jars/spark-examples.jar",
            "entryPointArguments": ["1"],
            "sparkSubmitParameters": "--class org.apache.spark.examples.SparkPi
            --conf spark.executor.cores=4
            --conf spark.executor.memory=20g
            --conf spark.driver.cores=4
            --conf spark.driver.memory=8g
            --conf spark.executor.instances=1
            --conf spark.emr-serverless.executor.disk.type=shuffle_optimized"
        }
    }'
```

如需詳細資訊，請參閱 Spark 工作屬性。

Hive

1. 使用下列EMR命令建立無伺服器 7.1.0 版應用程式。

```
aws emr-serverless create-application \
  --type "HIVE" \
  --name my-application-name \
  --release-label emr-7.1.0 \
  --region <AWS_REGION>
```

2. 將 Hive 工作設定為包含參數hive.driver.disk.type和/或hive.tez.disk.type使用隨機最佳化磁碟執行。您可以使用一個或兩個參數，具體取決於您的使用案例。

```
aws emr-serverless start-job-run \
    --application-id application-id \
    --execution-role-arn job-role-arn \
```

```
    --job-driver '{
        "hive": {
            "query": "s3://<DOC-EXAMPLE-BUCKET>/emr-serverless-hive/query/hive-
query.ql",
            "parameters": "--hiveconf hive.log.explain.output=false"
        }
    }' \
    --configuration-overrides '{
        "applicationConfiguration": [{
            "classification": "hive-site",
            "properties": {
                "hive.exec.scratchdir": "s3://<DOC-EXAMPLE-BUCKET>/emr-
serverless-hive/hive/scratch",
                "hive.metastore.warehouse.dir": "s3://<DOC-EXAMPLE-BUCKET>/emr-
serverless-hive/hive/warehouse",
                "hive.driver.cores": "2",
                "hive.driver.memory": "4g",
                "hive.tez.container.size": "4096",
                "hive.tez.cpu.vcores": "1",
                "hive.driver.disk.type": "shuffle_optimized",
                "hive.tez.disk.type": "shuffle_optimized"
            }
        }]
    }'
```

如需詳細資訊，Hive 工作屬性。

使用預先初始化容量設定應用程式

請參閱下列範例，以建立以 Amazon 7.1.0 EMR 版為基礎的應用程式。這些應用程式具有下列屬性：

- 5 個預先初始化的 Spark 驅動程式，每個驅動程式都有 2 v CPU、4 GB 的記憶體，以及 50 GB 的隨機播放最佳化磁碟。
- 50 個預先初始化的執行程序，每個執行程序都具有 4 vCPU，8 GB 的內存和 500 GB 的隨機播放優化磁盤。

當此應用程式執行 Spark 作業時，它會先使用預先初始化的 Worker，然後將隨選背景工作程式擴充到最大容量 400 v CPU 和 1024 GB 的記憶體。或者，您可以省略DRIVER或的容量EXECUTOR。

Spark

```
aws emr-serverless create-application \
  --type "SPARK" \
  --name <my-application-name> \
  --release-label emr-7.1.0 \
  --initial-capacity '{
    "DRIVER": {
        "workerCount": 5,
        "workerConfiguration": {
            "cpu": "2vCPU",
            "memory": "4GB",
            "disk": "50GB",
            "diskType": "SHUFFLE_OPTIMIZED"
        }
    },
    "EXECUTOR": {
        "workerCount": 50,
        "workerConfiguration": {
            "cpu": "4vCPU",
            "memory": "8GB",
            "disk": "500GB",
            "diskType": "SHUFFLE_OPTIMIZED"
        }
    }
  }' \
  --maximum-capacity '{
    "cpu": "400vCPU",
    "memory": "1024GB"
  }'
```

Hive

```
aws emr-serverless create-application \
  --type "HIVE" \
  --name <my-application-name> \
  --release-label emr-7.1.0 \
  --initial-capacity '{
    "DRIVER": {
        "workerCount": 5,
        "workerConfiguration": {
            "cpu": "2vCPU",
            "memory": "4GB",
```

```
            "disk": "50GB",
            "diskType": "SHUFFLE_OPTIMIZED"
        }
    },
    "EXECUTOR": {
        "workerCount": 50,
        "workerConfiguration": {
            "cpu": "4vCPU",
            "memory": "8GB",
            "disk": "500GB",
            "diskType": "SHUFFLE_OPTIMIZED"
        }
    }
}' \
--maximum-capacity '{
    "cpu": "400vCPU",
    "memory": "1024GB"
}'
```

串流工作

EMR無伺服器中的串流工作是一種工作模式，可讓您以近乎即時的速度分析和處理串流資料。這些長時間執行的工作輪詢串流資料，並在資料到達時持續處理結果。串流作業最適合需要即時資料處理的工作，例如近乎即時的分析、詐騙偵測和建議引擎。EMR無伺服器串流作業可提供最佳化功能，例如內建工作備援、即時監控、增強的記錄管理，以及與串流連接器整合。

以下是一些串流工作的使用案例：

- 近乎即時的分析 — Amazon EMR Serverless 中的串流任務可讓您以近乎即時的方式處理串流資料，因此您可以對連續資料串流 (例如日誌資料、感應器資料或點擊流資料) 執行即時分析，以獲得見解並根據最新資訊及時做出決策。
- 詐騙偵測 — 當您分析資料串流並識別可疑模式或異常發生時，您可以使用串流作業在金融交易、信用卡作業或線上活動中執行近乎即時的詐騙偵測。
- 建議引擎 — 串流作業可以處理使用者活動資料並更新建議模型。這樣做會根據行為和偏好開啟個人化和即時建議的可能性。
- 社交媒體分析 — 串流工作可以處理社交媒體資料，例如推文、留言和貼文，因此組織可以近乎即時地監控趨勢、情緒分析和管理品牌聲譽。
- 物聯網 (IoT) 分析 — 串流任務可以處理和分析來自 IoT 裝置、感應器和連線機械的高速資料串流，因此您可以執行異常偵測、預測性維護和其他 IoT 分析使用案例。

- 點擊流分析 — 流任務可以處理和分析來自網站或移動應用程序的點擊流數據。使用此類資料的企業可以執行分析，進一步瞭解使用者行為、個人化使用者體驗，以及最佳化行銷宣傳活動。
- 記錄監控和分析 — 串流作業也可以處理來自伺服器、應用程式和網路裝置的記錄資料。這可為您提供異常偵測、疑難排解，以及系統健康狀態和效能。

主要優點

EMR無伺服器中的串流作業會自動提供工作備援，這是下列因素的組合：

- 自動重試 — EMR 無伺服器會自動重試任何失敗的工作，而無需您手動輸入。
- 可用區域 (AZ) 備援 — 如果原始 AZ 發生問題，EMR無伺服器會自動將串流作業切換至運作良好的可用區域。
- 日誌管理：
  - 記錄輪替 — 為了更有效率的磁碟儲存管理，EMRServerless 會定期輪換長串流工作的記錄檔。這樣做可防止記錄累積可能會消耗所有磁碟空間。
  - 記錄壓縮 — 協助您有效率地管理及最佳化受管理持續性中的記錄檔。當您使用受管理的 spark 歷程記錄伺服器時，壓縮也會改善偵錯體驗。

支援的資料來源和資料接收器

EMR無伺服器可與多個輸入資料來源和輸出資料接收器搭配使用：

- 支援的輸入資料來源 — Amazon Kinesis Data Streams、適用於 Apache 卡夫卡的 Amazon 受管串流，以及自我管理的 Apache 卡夫卡叢集。根據預設，Amazon 7.1.0 及更高EMR版本包含 Amazon Kinesis Data Streams 連接器，因此您不需要建立或下載任何額外的套件。
- 支援的輸出資料接收器 — AWS Glue 數據目錄表，Amazon S3，Amazon Redshift，我的，Postgre SQL 甲骨文SQL，甲骨文，Microsoft，阿帕奇冰山SQL，三角洲湖和阿帕奇胡迪。

考量與限制

使用串流工作時，請記住下列考量事項和限制。

- Amazon 7.1.0 及更高EMR版本支援串流任務。
- EMRServerless 預期串流作業會長時間執行，因此您無法設定執行逾時來限制工作的執行時間。
- 串流作業僅與 Spark 引擎相容，該引擎建立在結構化串流架構之上。

- EMR無伺服器會無限期地重試串流作業，而且您無法自訂最大嘗試次數。如果失敗的嘗試次數超過了每小時窗口設置的閾值，則會自動包含鞭打防止以停止工作重試。預設臨界值為一小時內的五次失敗嘗試。您可以將此臨界值設定為 1 到 10 次嘗試之間。如需詳細資訊，請參閱 Job 復原。
- 串流工作具有儲存執行階段狀態和進度的檢查點，因此EMR無伺服器可以從最新的檢查點繼續串流工作。如需詳細資訊，請參閱 Apache Spark 文件中的使用檢查點從失敗中復原。

開始串流工作

請參閱下列指示，瞭解如何開始使用串流工作。

1. 請遵循開始使用 Amazon EMR 無伺服器建立應用程式。請注意，您的應用程式必須執行 Amazon 7.1.0 或更高EMR版本。
2. 應用程式準備就緒後，請將mode參數設定STREAMING為提交串流工作，類似下列內容 AWS CLI 例子。

```
aws emr-serverless start-job-run \
--application-id <APPPLICATION_ID> \
--execution-role-arn <JOB_EXECUTION_ROLE> \
--mode 'STREAMING' \
--job-driver '{
    "sparkSubmit": {
        "entryPoint": "s3://<streaming script>",
        "entryPointArguments": ["s3://<DOC-EXAMPLE-BUCKET-OUTPUT>/output"],
        "sparkSubmitParameters": "--conf spark.executor.cores=4
            --conf spark.executor.memory=16g
            --conf spark.driver.cores=4
            --conf spark.driver.memory=16g
            --conf spark.executor.instances=3"
    }
}'
```

支援串流連接器

串流連接器有助於從串流來源讀取資料，也可以將資料寫入串流接收器。

以下是支援的串流連接器：

Amazon Kinesis Data Streams 連接器

適用於 Apache Spark 的 Amazon Kinesis Data Streams 連接器可讓您建立串流應用程式和管道，以取用來自 Amazon Kinesis 資料串流的資料並將資料寫入。此連接器支援增強的扇出消耗，每個碎片的專用讀取輸送率最高可達 2MB/ 秒。根據預設，Amazon EMR 無伺服器 7.1.0 及更高版本包含連接器，因此您不需要建立或下載任何額外的套件。如需有關連接器的詳細資訊，請參閱上的 spark-sql-kinesis-connector 頁面 GitHub。

以下是如何使用 Kinesis Data Streams 連接器相依性開始工作執行的範例。

```
aws emr-serverless start-job-run \
--application-id <APPLICATION_ID> \
--execution-role-arn <JOB_EXECUTION_ROLE> \
--mode 'STREAMING' \
--job-driver '{
    "sparkSubmit": {
        "entryPoint": "s3://<Kinesis-streaming-script>",
        "entryPointArguments": ["s3://<DOC-EXAMPLE-BUCKET-OUTPUT>/output"],
        "sparkSubmitParameters": "--conf spark.executor.cores=4
                --conf spark.executor.memory=16g
                --conf spark.driver.cores=4
                --conf spark.driver.memory=16g
                --conf spark.executor.instances=3
                --jars /usr/share/aws/kinesis/spark-sql-kinesis/lib/spark-streaming-
sql-kinesis-connector.jar"
    }
}'
```

若要連線到 Kinesis Data Streams，您必須設定具有VPC存取權的EMR無伺服器應用程式，並使用VPC端點來允許私人存取。或使用NAT閘道取得公開存取權。如需詳細資訊，請參閱設定VPC存取權。您也必須確定您的工作執行階段角色具有必要的讀取和寫入權限，才能存取所需的資料串流。若要進一步了解如何設定 Job 務執行時期角色，請參閱 Amazon EMR 無伺服器的任務執行階段角色。如需所有必要權限的完整清單，請參閱上的spark-sql-kinesis-connector 頁面 GitHub。

阿帕奇卡夫卡連接器

用於星火結構化流阿帕奇卡夫卡連接器是來自星火社區的開源連接器，並在 Maven 存儲庫中可用。此連接器可協助 Spark 結構化串流應用程式讀取資料，並將資料寫入自我管理的 Apache Kafka，以及適用於 Apache Kafka 的 Amazon 受管串流。如需有關連接器的詳細資訊，請參閱 Apache Spark 說明文件中的結構化串流 + 卡夫卡整合指南。

下列範例示範如何在工作執行要求中包含 Kafka 連接器。

```
aws emr-serverless start-job-run \
```

```
--application-id <APPLICATION_ID> \
--execution-role-arn <JOB_EXECUTION_ROLE> \
--mode 'STREAMING' \
--job-driver '{
    "sparkSubmit": {
        "entryPoint": "s3://<Kafka-streaming-script>",
        "entryPointArguments": ["s3://<DOC-EXAMPLE-BUCKET-OUTPUT>/output"],
        "sparkSubmitParameters": "--conf spark.executor.cores=4
                --conf spark.executor.memory=16g
                --conf spark.driver.cores=4
                --conf spark.driver.memory=16g
                --conf spark.executor.instances=3
                --packages org.apache.spark:spark-sql-
kafka-0-10_2.12:<KAFKA_CONNECTOR_VERSION>"
    }
}'
```

Apache Kafka 連接器版本取決於您的EMR無伺服器發行版本和對應的 Spark 版本。要查找正確的卡夫卡版本，請參閱結構化流 + 卡夫卡集成指南。

若要透過IAM身份驗證使用適用於 Apache Kafka 的 Amazon 受管串流，您必須包含另一個相依性，以使 Kafka 連接器能夠透過連接到 Amazon。MSK IAM如需詳細資訊，請參閱（詳見）的aws-msk-iam-auth 存放庫 GitHub。您也必須確定工作執行階段角色具有必要的IAM權限。下列範例示範如何將連接器與IAM驗證搭配使用。

```
aws emr-serverless start-job-run \
--application-id <APPLICATION_ID> \
--execution-role-arn <JOB_EXECUTION_ROLE> \
--mode 'STREAMING' \
--job-driver '{
    "sparkSubmit": {
        "entryPoint": "s3://<Kafka-streaming-script>",
        "entryPointArguments": ["s3://<DOC-EXAMPLE-BUCKET-OUTPUT>/output"],
        "sparkSubmitParameters": "--conf spark.executor.cores=4
                --conf spark.executor.memory=16g
                --conf spark.driver.cores=4
                --conf spark.driver.memory=16g
                --conf spark.executor.instances=3
                --packages org.apache.spark:spark-sql-
kafka-0-10_2.12:<KAFKA_CONNECTOR_VERSION>,software.amazon.msk:aws-msk-iam-
auth:<MSK_IAM_LIB_VERSION>"
    }
```

```
}'
```

若要使用來自 Amazon 的 Kafka 連接器和IAM身份驗證程式庫，MSK您必須設定具有存取權的EMR無伺服器應用程式。VPC您的子網必須具有 Internet 訪問權限，並使用NAT網關訪問 Maven 依賴關係。如需詳細資訊，請參閱設定VPC存取權。子網路必須具有網路連線能力才能存取 Kafka 叢集。無論您的 Kafka 叢集是自我管理還是使用適用於 Apache Kafka 的 Amazon 受管串流，都是如此。

串流工作記錄管理

日誌旋轉

串流工作支援 Spark 應用程式記錄檔和事件記錄檔的記錄輪替。記錄輪換可防止長時間串流工作產生可能會佔用所有可用磁碟空間的大型記錄檔。記錄輪替可協助您節省磁碟儲存空間，並防止因磁碟空間不足而導致工作失敗。如需詳細資訊，請參閱輪替記錄檔。

日誌壓實

只要有受管理的記錄檔可用，串流作業也支援 Spark 事件記錄的記錄壓縮。如需受管理記錄的詳細資訊，請參閱使用受管理儲存區記錄。串流作業可能會長時間執行，而且隨著時間的推移，事件資料量可能會大幅增加記錄檔大小。星火歷史記錄服務器讀取並將這些事件加載到內存中的 Spark 應用程序 UI。此程序可能會導致高延遲和成本，尤其是存放在 Amazon S3 中的事件日誌非常大時。

記錄壓縮會減少事件記錄檔的大小，因此 Spark 歷程記錄伺服器不需要載入超過 1 GB 的事件記錄檔在任何時間。如需詳細資訊，請參閱 Apache Spark 文件中的監視和檢測。

火花工作

您可以在將`type`參數設定為的應用程式上執行 Spark 工作SPARK。任務必須與與 Amazon EMR 發行版本兼容的 Spark 版本兼容。例如，當您使用 Amazon 6.6.0 EMR 版執行任務時，您的任務必須與 Apache Spark 3.2.0 相容。如需每個版本之應用程式版本的資訊，請參閱Amazon EMR 無伺服器發行版本。

火花工作參數

當您使用執`StartJobRun`API行 Spark 工作時，您可以指定下列參數。

必要參數

- Spark 工作執行階段角

- 火花工作驅動器參數
- 星火配置覆蓋參數
- 火花動態資源分配優化

Spark 工作執行階段角

用**executionRoleArn**來指定應ARN用程式用來執行 Spark 工作的IAM角色。此角色必須包含下列權限：

- 從 S3 儲存貯體或資料所在的其他資料來源讀取
- 從指令碼或JAR檔案所在的 S3 儲存貯體或首 PySpark 碼讀取
- 寫入想要編寫最終輸出的 S3 儲存貯體
- 將日誌寫入S3MonitoringConfiguration指定的 S3 儲存貯體或前綴
- 如果您使用KMS金鑰加密 S3 儲存貯體中的資料，可存取KMS金鑰
- 訪問 AWS Glue 資料目錄 (如果您使用 Spark) SQL

如果您的 Spark 工作從其他資料來源讀取或寫入資料，請在此IAM角色中指定適當的權限。如果您未提供這些權限給IAM角色，則工作可能會失敗。如需詳細資訊，請參閱 Amazon EMR 無伺服器的 Job 務執行階段角色 和 儲存記錄。

火花工作驅動器參數

用**jobDriver**於為工作提供輸入。工作驅動程式參數只接受您要執行之工作類型的一個值。對於星火工作，參數值為sparkSubmit。您可以使用此作業類型來運行斯卡拉，Java，Sparkr PySpark，並通過星火提交任何其他支持的作業。星火作業具有以下參數：

- **sparkSubmitParameters**— 這些是您要傳送至工作的其他 Spark 參數。使用此參數可覆寫預設 Spark 屬性，例如驅動程式記憶體或執行程式數目，例如--conf或--class引數中定義的屬性。
- **entryPointArguments**-這是您想要傳遞給主文件JAR或 Python 文件的參數數組。應使用 entrypoint 程式碼讀取這些參數。以逗號分隔陣列中的每個引數。
- **entryPoint**-這是 Amazon S3 中對您要運行的主文件JAR或 Python 文件的引用。如果您正在運行 Scala 或 JavaJAR，請在SparkSubmitParameters使用--class參數中指定主條目類。

如需其他資訊，請參閱透過 spark-submit 啟動應用程式。

星火配置覆蓋參數

用**configurationOverrides**於覆寫監督層次和應用程式層次組態特性。此參數接受具有下列兩個欄位的JSON物件：

- **monitoringConfiguration**-使用此欄位指定您希望EMR無伺服器任務存放 Spark 任務日誌的 Amazon S3 URL (s3MonitoringConfiguration)。確定您已使用相同的值區建立此值區 AWS 帳戶 託管您的應用程序，並且在相同 AWS 區域 您的工作正在執行的位置。
- **applicationConfiguration**— 若要覆寫應用程式的預設組態，您可以在此欄位中提供配置物件。您可以使用簡寫語法來提供配置，也可以參考檔案中的配置物件。JSON組態物件是由分類、屬性和選用的巢狀組態所組成。屬性由您想要在檔案中覆寫的設定組成。您可以在單一物件中為多個應用程式指定多個分JSON類。

> **Note**
>
> 可用的組態分類會因特定的EMR無伺服器版本而異。例如，自訂 Log4j 的分類，spark-driver-log4j2且僅spark-executor-log4j2適用於 6.8.0 及更高版本。

如果您在應用程式覆寫和 Spark 提交參數中使用相同的組態，Spark 提交參數會優先處理。組態的優先順序如下，從最高到最低：

- EMR無伺服器在建立SparkSession時提供的組態。
- 您提供作為--conf引數一部分sparkSubmitParameters的組態。
- 您在應用程式中提供的組態會在您啟動工作時覆寫。
- 您在建立應用程式runtimeConfiguration時所提供的組態。
- Amazon EMR 用於發行版本的優化組態。
- 應用程式的預設開放原始碼組態。

如需在應用程式層級宣告組態，以及在工作執行期間覆寫組態的詳細資訊，請參閱。EMR無伺服器的預設應用程式組態

火花動態資源分配優化

用dynamicAllocationOptimization於最佳化EMR無伺服器中的資源使用率。在 Spark 組態分類true中將此屬性設定為表示EMR無伺服器以最佳化執行程式資源配置，以便更好地將 Spark 要求和取消執行程式的速率與EMR無伺服器建立和釋放 Worker 的速率一致。如此一來，EMR無伺服器可以

更優化地跨階段重複使用工作者，因此在執行具有多個階段的作業時，可降低成本，同時保持相同的效能。

此屬性適用於所有 Amazon EMR 發行版本。

以下是使用的範例組態分類dynamicAllocationOptimization。

```
[
  {
    "Classification": "spark",
    "Properties": {
      "dynamicAllocationOptimization": "true"
    }
  }
]
```

如果您使用動態配置最佳化，請考慮下列事項：

- 此最佳化適用於您啟用動態資源配置的 Spark 工作。
- 為了達到最佳的成本效益，我們建議您根據工作負載，使用工作層級設定spark.dynamicAllocation.maxExecutors或應用程式層級最大容量設定，在 Worker 上設定較高的擴展限制。
- 在較簡單的工作中，您可能看不到成本改善。例如，如果您的工作在小型資料集上執行，或在一個階段中完成執行，Spark 可能不需要較多的執行程式或多個擴展事件。
- 具有大型階段、較小階段，然後再次大型階段的工作，可能會在工作執行階段中遇到迴歸。由於EMR無伺服器使用資源的效率更高，因此可能導致較大階段的可用工作者減少，從而延長執行時間。

火花工作屬性

下表列出選用的 Spark 屬性及其預設值，您可以在送出 Spark 工作時覆寫這些屬性。

金鑰	描述	預設值
spark.archives	Spark 提取到每個執行程序的工作目錄的檔案的逗號分隔列表。支援的檔案類型包括.jar .tar.gz、.tgz和.zip。若	NULL

金鑰	描述	預設值
	要指定要擷取的目錄名稱，請#在要擷取的檔案名稱之後新增。例如：file.zip#directory 。	
spark.authenticate	開啟 Spark 內部連線驗證的選項。	TRUE
spark.driver.cores	驅動程式使用的核心數目。	4
spark.driver.extraJavaOptions	為星火驅動程序額外的 Java 選項。	NULL
spark.driver.memory	驅動程式使用的記憶體容量。	14 克
spark.dynamicAllocation.enabled	開啟動態資源配置的選項。根據工作負載，此選項可擴展或減少在應用程式中註冊的執行者數量。	TRUE
spark.dynamicAllocation.executorIdleTimeout	Spark 將其刪除之前，執行程序可以保持閒置的時間長度。這只有在您開啟動態配置時才適用。	60 年代
spark.dynamicAllocation.initialExecutors	開啟動態配置時要執行的初始執行程式數目。	3
spark.dynamicAllocation.maxExecutors	如果您打開動態分配，則執行者數量的上限。	對於 6.10.0 及更高版本，infinity 對於 6.9.0 及更低版本，100
spark.dynamicAllocation.minExecutors	如果您打開動態分配，則執行者數量的下限。	0

金鑰	描述	預設值
spark.emr-serverless.allocation.batch.size	在執行人分配的每個週期中要請求的容器數量。每個配置週期之間有一秒間隔。	20
spark.emr-serverless.driver.disk	星火驅動程序盤。	20G
spark.emr-serverless.driverEnv. *[KEY]*	將環境變數新增至 Spark 驅動程式的選項。	NULL
spark.emr-serverless.executor.disk	星火執行程序磁盤。	20G
spark.emr-serverless.memoryOverheadFactor	設置內存開銷添加到驅動程序和執行程序容器內存。	0.1
spark.emr-serverless.driver.disk.type	連接到星火驅動程序的磁盤類型。	標準
spark.emr-serverless.executor.disk.type	附加到 Spark 執行程序的磁盤類型。	標準
spark.executor.cores	每個執行程序使用的內核數。	4
spark.executor.extraJavaOptions	為星火執行程序額外的 Java 選項。	NULL
spark.executor.instances	要分配的 Spark 執行程序容器的數量。	3
spark.executor.memory	每個執行程序使用的內存量。	14 克
spark.executorEnv. *[KEY]*	將環境變量添加到 Spark 執行程序的選項。	NULL

金鑰	描述	預設值
spark.files	以逗號分隔的文件列表進入每個執行程序的工作目錄。您可以使用SparkFiles.get(*fileName*)執行程序訪問這些文件的文件路徑。	NULL
spark.hadoop.hive.metastore.client.factory.class	蜂巢中繼存儲實現類。	NULL
spark.jars	要添加到驅動程序和執行程序的運行時類路徑的其他 jar。	NULL
spark.network.crypto.enabled	開啟AES基於RPC加密的選項。這包括星火 2.2.0 中添加的身份驗證協議。	FALSE
spark.sql.warehouse.dir	受管理的資料庫和資料表的預設位置。	的價值 $PWD/spark-warehouse
spark.submit.pyFiles	要在 Python 應用程式中放置的.zip.egg、或 .py檔案的逗號分隔清單。PYTHONPATH	NULL

下表列出了默認的 Spark 提交參數。

金鑰	描述	預設值
archives	Spark 提取到每個執行程序的工作目錄的檔案的逗號分隔列表。	NULL
class	應用程序的主類（用於 Java 和斯卡拉應用程序）。	NULL
conf	任意星火配置屬性。	NULL

金鑰	描述	預設值
driver-cores	驅動程式使用的核心數目。	4
driver-memory	驅動程式使用的記憶體容量。	14 克
executor-cores	每個執行程序使用的內核數。	4
executor-memory	執行程序使用的內存量。	14 克
files	以逗號分隔的文件列表放置在每個執行程序的工作目錄中。您可以使用SparkFiles.get(*fileName*)執行程序訪問這些文件的文件路徑。	NULL
jars	要包含在驅動程式和執行程式類別路徑上的 jar 的逗號分隔清單。	NULL
num-executors	要啟動的執行者的數量。	3
py-files	要放置在適用PYTHONPATH 於 Python 應用程式的.zip.egg、或.py檔案的逗號分隔清單。	NULL
verbose	開啟其他除錯輸出的選項。	NULL

火花的例子

下面的例子演示了如何使用StartJobRunAPI來運行一個 Python 腳本。如需使用此範例的 end-to-end 自學課程，請參閱開始使用 Amazon EMR 無伺服器。您可以在EMR無伺服器範例 GitHub 儲存庫中找到如何執行 PySpark 工作和新增 Python 相依性的其他範例。

```
aws emr-serverless start-job-run \
    --application-id application-id \
    --execution-role-arn job-role-arn \
    --job-driver '{
```

```
        "sparkSubmit": {
            "entryPoint": "s3://us-east-1.elasticmapreduce/emr-containers/samples/
wordcount/scripts/wordcount.py",
            "entryPointArguments": ["s3://DOC-EXAMPLE-BUCKET-OUTPUT/wordcount_output"],
            "sparkSubmitParameters": "--conf spark.executor.cores=1 --conf
 spark.executor.memory=4g --conf spark.driver.cores=1 --conf spark.driver.memory=4g --
conf spark.executor.instances=1"
        }
    }'
```

下面的例子演示了如何使StartJobRunAPI用運行一個星火JAR。

```
aws emr-serverless start-job-run \
    --application-id application-id \
    --execution-role-arn job-role-arn \
    --job-driver '{
        "sparkSubmit": {
            "entryPoint": "/usr/lib/spark/examples/jars/spark-examples.jar",
            "entryPointArguments": ["1"],
            "sparkSubmitParameters": "--class org.apache.spark.examples.SparkPi --conf
 spark.executor.cores=4 --conf spark.executor.memory=20g --conf spark.driver.cores=4 --
conf spark.driver.memory=8g --conf spark.executor.instances=1"
        }
    }'
```

蜂巢工作

您可以在將type參數設定為的應用程式上執行 Hive 作業HIVE。任務必須與與 Amazon EMR 發行版本相容的 Hive 版本相容。例如，當您在使用 Amazon 6.6.0 EMR 版的應用程式上執行任務時，您的任務必須與 Apache Hive 3.1.2 相容。如需每個版本之應用程式版本的資訊，請參閱Amazon EMR 無伺服器發行版本。

蜂巢工作參數

當您使用執StartJobRunAPI行 Hive 工作時，您必須指定下列參數。

必要參數

- 配置單元工作運行時
- 蜂巢工作驅動程式參
- 蜂巢配置覆蓋參數

配置單元工作運行時

用**executionRoleArn**來指定應ARN用程式用來執行 Hive 工作的IAM角色。此角色必須包含下列權限：

- 從 S3 儲存貯體或資料所在的其他資料來源讀取
- 從您的 Hive 查詢文件和初始化查詢文件所在的 S3 存儲桶或前綴讀取
- 讀取和寫入到您的 Hive 暫存目錄和 Hive 中繼存儲庫倉庫目錄所在的 S3 存儲桶
- 寫入想要編寫最終輸出的 S3 儲存貯體
- 將日誌寫入`S3MonitoringConfiguration`指定的 S3 儲存貯體或前綴
- 如果您使用KMS金鑰加密 S3 儲存貯體中的資料，可存取KMS金鑰
- 存取 AWS Glue Data Catalog

如果您的 Hive 作業從其他資料來源讀取或寫入資料，請在此IAM角色中指定適當的權限。如果您未提供這些權限給IAM角色，您的工作可能會失敗。如需詳細資訊，請參閱Amazon EMR 無伺服器的 Job 務執行階段角色。

蜂巢工作驅動程式參

用**jobDriver**於為工作提供輸入。工作驅動程式參數只接受您要執行之工作類型的一個值。當您指定`hive`為工作類型時，EMR無伺服器會將 Hive 查詢傳遞至`jobDriver`參數。配置單元作業具有以下參數：

- **query**— 這是 Amazon S3 中對您要執行的 Hive 查詢檔案的參考。
- **parameters**-這些是您要覆蓋的其他 Hive 配置屬性。要覆蓋屬性，請將它們作為傳遞給此參數`--hiveconf` *`property=value`*。要覆蓋變量，請將它們作為這個參數傳遞給`--hivevar` *`key=value`*。
- **initQueryFile**-這是初始化配置單元查詢文件。蜂巢在查詢之前運行此文件，並可以使用它來初始化表。

蜂巢配置覆蓋參數

用**configurationOverrides**於覆寫監督層次和應用程式層次組態特性。此參數接受具有下列兩個欄位的JSON物件：

- **monitoringConfiguration**— 使用此欄位可指定您希望EMR無伺服器任務存放 Hive 任務日誌的 Amazon S3 URL (s3MonitoringConfiguration)。請確定您使用相同的值區建立此值區 AWS 帳戶 託管您的應用程序，並且在相同 AWS 區域 您的工作正在執行的位置。
- **applicationConfiguration**— 您可以在此欄位中提供配置物件，以覆寫應用程式的預設組態。您可以使用簡寫語法來提供配置，也可以參考檔案中的配置物件。JSON組態物件是由分類、屬性和選用的巢狀組態所組成。屬性由您想要在檔案中覆寫的設定組成。您可以在單一物件中為多個應用程式指定多個分JSON類。

> **Note**
>
> 可用的組態分類會因特定的EMR無伺服器版本而異。例如，自訂 Log4j 的分類，spark-driver-log4j2且僅spark-executor-log4j2適用於 6.8.0 及更高版本。

如果您在應用程序覆蓋和蜂巢參數傳遞相同的配置，蜂房參數優先。下列清單會將組態從最高優先順序排列到最低優先順序。

- 您提供作為 Hive 參數的一部分的配置--hiveconf *property=value*。
- 您在應用程式中提供的組態會在您啟動工作時覆寫。
- 您在建立應用程式runtimeConfiguration時所提供的組態。
- Amazon 為發行版EMR指派的最佳化組態。
- 應用程式的預設開放原始碼組態。

如需在應用程式層級宣告組態，以及在工作執行期間覆寫組態的詳細資訊，請參閱。EMR無伺服器的預設應用程式組態

蜂巢工作屬性

下表列出提交 Hive 工作時必須設定的必要特性。

設定	描述
hive.exec.scratchdir	在 Hive 任務執行期間，EMR無伺服器建立暫存檔案的 Amazon S3 位置。

設定	描述
hive.metastore.warehouse.dir	在 Hive 中受管表格的資料庫的 Amazon S3 位置。

下表列出選擇性的 Hive 屬性及其預設值，您可以在提交 Hive 工作時覆寫這些屬性。

設定	描述	預設值
fs.s3.customAWSCredentialsProvider	所以此 AWS 您要使用的認證提供者。	亞馬遜公司. efaultAWS Credentials ProviderChain
fs.s3a.aws.credentials.provider	所以此 AWS 您要與 S3A 檔案系統搭配使用的認證提供者。	亞馬遜公司. efaultAWS Credentials ProviderChain
hive.auto.convert.join	根據輸入檔案大小，開啟自動將一般聯結轉換為映射連接的選項。	TRUE
hive.auto.convert.join.noconditionaltask	當 Hive 根據輸入文件大小將通用聯接轉換為 mapjoin 時打開優化的選項。	TRUE
hive.auto.convert.join.noconditionaltask.size	連接會直接轉換為此大小以下的 Mapjoin。	根據 Tez 工作記憶體計算最佳值
hive.cbo.enable	使用方解石框架啟用基於成本的優化的選項。	TRUE
hive.cli.tez.session.async	選項來啟動一個後台 Tez 會話，而你的蜂巢查詢編譯。當設置為false，TEZ AM 啟動您的蜂巢查詢編譯後。	TRUE
hive.compute.query.using.stats	激活 Hive 以使用存儲在元存儲中的統計信息來回答某些查詢的選項。對於基本	TRUE

設定	描述	預設值
	統計資料，請hive.stats.autogather 將設定為TRUE。如需更進階的查詢集合，請執行analyze table queries。	
hive.default.fileformat	CREATE TABLE陳述式的預設檔案格式。如果您在CREATE TABLE命令STORED AS [FORMAT]中指定，則可以明確覆蓋此值。	TEXTFILE
hive.driver.cores	用於 Hive 驅動程序進程的內核數。	2
hive.driver.disk	蜂巢驅動程式的磁碟大小。	20G
hive.driver.disk.type	蜂巢驅動程式的磁碟類型。	標準
hive.tez.disk.type	Tez 工作者的磁碟大小。	標準
hive.driver.memory	每個 Hive 驅動程序進程使用的內存量。配置單元CLI和 Tez 應用程序主機共享此內存與 20% 的成長空間。	6 克
hive.emr-serverless.launch.env.[*KEY*]	在所有 Hive 特定進程中設置*KEY*環境變量的選項，例如您的 Hive 驅動程序，Tez AM 和 Tez 任務。	
hive.exec.dynamic.partition	在DML/中打開動態分區的選項DDL。	TRUE

設定	描述	預設值
hive.exec.dynamic.partition.mode	指定要使用嚴謹模式還是非嚴格模式的選項。在嚴謹模式下，您必須至少指定一個靜態分割區，以防意外覆寫所有分割區。在非嚴格模式下，允許所有分區都是動態的。	strict
hive.exec.max.dynamic.partitions	Hive 總共創建的動態分區的最大數量。	1000
hive.exec.max.dynamic.partitions.pernode	Hive 在每個映射器和減速器節點中創建的動態分區的最大數量。	100
hive.exec.orc.split.strategy	需要下列其中一個值：BIETL、或HYBRID。這不是使用者層級的設定。 BI指定您希望在分割產生中花費更少的時間，而不是查詢執行。ETL指定您希望在分割產生中花費更多時間。 HYBRID根據啟發式指定上述策略的選擇。	HYBRID
hive.exec.reducers.bytes.per.reducer	每個減速器的尺寸。預設值為 256 MB。如果輸入大小為 1G，則工作使用 4 個減速器。	256000000
hive.exec.reducers.max	減速器的最大數量。	256
hive.exec.stagingdir	存儲 Hive 在表位置內部和屬性中指定的暫存目錄位置中創建的臨時文件的目錄的名hive.exec.scratchdir 稱。	.hive-staging

設定	描述	預設值
hive.fetch.task.conversion	需要下列其中一個值：NONEMINIMAL、或MORE。蜂巢可以選擇查詢轉換為單個FETCH任務。這將延遲降至最低。	MORE
hive.groupby.position.alias	導致 Hive 在GROUP BY語句中使用列位置別名的選項。	FALSE
hive.input.format	默認的輸入格式。HiveInputFormat 如果您遇到問題，請設定為CombineHiveInputFormat 。	org.apache.hadoop.hive.ql.io.CombineHiveInputFormat
hive.log.explain.output	開啟 Hive 記錄中任何查詢的延伸輸出說明的選項。	FALSE
hive.log.level	蜂巢日誌記錄級別。	INFO
hive.mapred.reduce.tasks.speculative.execution	開啟推測性推出減速器的選項。僅支援 Amazon EMR 6.10.x 及更低版本。	TRUE
hive.max-task-containers	並行容器的最大數目。設定的對應程式記憶體會乘以此值，以判斷分割運算和工作預佔使用的可用記憶體。	1000
hive.merge.mapfiles	導致小型檔案在僅地圖工作結束時合併的選項。	TRUE
hive.merge.size.per.task	工作結束時合併檔案的大小。	256000000
hive.merge.tezfiles	在 Tez DAG 結束時打開合併小文件的選項。	FALSE

設定	描述	預設值
hive.metastore.client.factory.class	生成實現IMetaStoreClient 接口的對象的工廠類的名稱。	com.amazonaws.glue.catalog.metastore.AWSGlueDataCatalogHiveClientFactory
hive.metastore.glue.catalogid	如果 AWS Glue 數據目錄充當中繼存儲，但在不同的中繼庫中運行 AWS 帳戶 比工作，的識別碼 AWS 帳戶 工作正在執行的位置。	NULL
hive.metastore.uris	元存儲客戶端用於連接到遠程中繼存儲的節儉URI。	NULL
hive.optimize.ppd	開啟謂詞下推的選項。	TRUE
hive.optimize.ppd.storage	開啟述詞下推至儲存處理常式的選項。	TRUE
hive.orderby.position.alias	導致 Hive 在ORDER BY語句中使用列位置別名的選項。	TRUE
hive.prewarm.enabled	為 Tez 開啟容器預熱的選項。	FALSE
hive.prewarm.numcontainers	Tez 預熱的容器數量。	10
hive.stats.autogather	使 Hive 在INSERT OVERWRITE 命令期間自動收集基本統計信息的選項。	TRUE
hive.stats.fetch.column.stats	關閉從中繼存放區擷取資料欄統計資料的選項。當資料行數很高時，擷取資料行統計資料可能會很昂貴。	FALSE

設定	描述	預設值
hive.stats.gather.num.threads	partialscan 和 noscan Analyze 命令用於分區資料表的執行緒數目。這僅適用於實現StatsProvidingRecordReader （如ORC）的文件格式。	10
hive.strict.checks.cartesian.product	開啟嚴格笛卡爾連接檢查的選項。這些檢查不允許笛卡爾乘積 (交叉連接)。	FALSE
hive.strict.checks.type.safety	開啟嚴格類型安全檢查並關閉bigint與string和的比較的選項double。	TRUE
hive.support.quoted.identifiers	預期值為NONE或COLUMN。NONE意味著只有字母數字和下劃線字符在標識符有效。COLUMN意味著列名可以包含任何字符。	COLUMN
hive.tez.auto.reducer.parallelism	開啟 Tez 自動縮減平行度功能的選項。Hive 仍然估計數據大小並設置並行估計值。Tez 對源頂點的輸出大小進行採樣，並根據需要在運行時調整估計值。	TRUE
hive.tez.container.size	每個 Tez 工作程序所使用的記憶體容量。	6144
hive.tez.cpu.vcores	每項 Tez 工作要使用的核心數目。	2
hive.tez.disk.size	每個工作容器的磁碟大小。	20G

設定	描述	預設值
hive.tez.input.format	在 Tez AM 分割產生的輸入格式。	org.apache.hadoop.hive.ql.io.HiveInputFormat
hive.tez.min.partition.factor	當您開啟自動減速器平行度時 Tez 指定的減速器的下限。	0.25
hive.vectorized.execution.enabled	開啟查詢執行向量化模式的選項。	TRUE
hive.vectorized.execution.reduce.enabled	開啟查詢執行縮減端的向量化模式的選項。	TRUE
javax.jdo.option.ConnectionDriverName	中JDBC繼存放區的驅動程式類別名稱。	org.apache.derby.jdbc.EmbeddedDriver
javax.jdo.option.ConnectionPassword	與中繼儲存庫資料庫相關聯的密碼。	NULL
javax.jdo.option.ConnectionURL	中JDBC繼存儲的JDBC連接字符串。	jdbc:derby:;databaseName=metastore_db;create=true
javax.jdo.option.ConnectionUserName	與中繼儲存庫資料庫相關聯的使用者名稱。	NULL
mapreduce.input.fileinputformat.split.maxsize	當您的輸入格式為時，分割計算期間的最大分割大小org.apache.hadoop.hive.ql.io.CombineHiveInputFormat 。值 0 表示沒有限制。	0
tez.am.dag.cleanup.on.completion	DAG完成時開啟清除隨機播放資料的選項。	TRUE

設定	描述	預設值
tez.am.emr-serverless.launch.env.[*KEY*]	在 Tez AM 過程中設置*KEY*環境變量的選項。對於 Tez AM，此值會取代該 hive.emr-serverless.launch.env.[*KEY*]值。	
tez.am.log.level	EMR無伺服器傳遞給 Tez 應用程式主機的根記錄層級。	INFO
tez.am.sleep.time.before.exit.millis	EMR無伺服器應在 AM 關閉要求之後的這段時間後推送ATS事件。	0
tez.am.speculation.enabled	導致推測性啟動較慢工作的選項。當某些工作執行速度較慢，因為電腦不良或速度較慢時，這有助於減少工作延遲。僅支援 Amazon EMR 6.10.x 及更低版本。	FALSE
tez.am.task.max.failed.attempts	在工作失敗之前，特定工作可能失敗的嘗試次數上限。這個數字不會計算手動終止的嘗試次數。	3
tez.am.vertex.cleanup.height	一個距離，如果所有從屬頂點都完成了，TEZ AM 將刪除頂點洗牌數據。當值為 0 時，此功能會關閉。Amazon 6.8.0 及更高EMR版本支持此功能。	0
tez.client.asynchronous-stop	導致EMR無伺服器在結束 Hive 驅ATS動程式之前推送事件的選項。	FALSE

設定	描述	預設值
tez.grouping.max-size	分組拆分的大小上限（以字節為單位）。此限制可防止過大的分割。	1073741824
tez.grouping.min-size	分組拆分的大小下限（以字節為單位）。此限制可防止太多小分割。	16777216
tez.runtime.io.sort.mb	Tez 對輸出進行排序時的軟緩衝區的大小。	根據 Tez 工作記憶體計算最佳值
tez.runtime.unordered.output.buffer.size-mb	Tez 不直接寫入磁碟時要使用的緩衝區大小。	根據 Tez 工作記憶體計算最佳值
tez.shuffle-vertex-manager.max-src-fraction	EMR無伺服器排程目前頂點的所有工作 (如果是ScatterGather 連線)，必須完成的來源工作分數。準備好在目前頂點上排程的工作數目，在min-fraction 和max-fraction 之間線性縮放。這將默認默認值或tez.shuffle-vertex-manager.min-src-fraction ，以較大者為準。	0.75
tez.shuffle-vertex-manager.min-src-fraction	EMR無伺服器為目前頂點排程工作之前必須完成的來源工作分數 (如果是ScatterGather 連線)。	0.25

設定	描述	預設值
tez.task.emr-serverless.launch.env.[*KEY*]	在 Tez 工作流程中設定*KEY*環境變數的選項。對於 Tez 工作，此值會覆寫hive.emr-serverless.launch.env.[*KEY*]值。	
tez.task.log.level	EMR無伺服器傳遞給 Tez 工作的根記錄層級。	INFO
tez.yarn.ats.event.flush.timeout.millis	AM 在關閉之前應等待事件清除的時間上限。	300000

蜂巢工作示例

下列程式碼範例示範如何使用執行 Hive 查詢StartJobRunAPI。

```
aws emr-serverless start-job-run \
    --application-id application-id \
    --execution-role-arn job-role-arn \
    --job-driver '{
        "hive": {
            "query": "s3://DOC-EXAMPLE-BUCKET/emr-serverless-hive/query/hive-query.ql",
            "parameters": "--hiveconf hive.log.explain.output=false"
        }
    }' \
    --configuration-overrides '{
        "applicationConfiguration": [{
            "classification": "hive-site",
            "properties": {
                "hive.exec.scratchdir": "s3://DOC-EXAMPLE-BUCKET/emr-serverless-hive/
hive/scratch",
                "hive.metastore.warehouse.dir": "s3://DOC-EXAMPLE-BUCKET/emr-
serverless-hive/hive/warehouse",
                "hive.driver.cores": "2",
                "hive.driver.memory": "4g",
                "hive.tez.container.size": "4096",
                "hive.tez.cpu.vcores": "1"
```

```
            }
        }]
    }'
```

您可以在EMR無伺服器範例 GitHub 儲存庫中找到如何執行 Hive 作業的其他範例。

EMR無伺服器 Job 恢復

EMR無伺服器版本 7.1.0 及更高版本包含工作恢復能力的支援，因此它會自動重試任何失敗的工作，無需您手動輸入。工作備援的另一個好處是，如果 AZ 遇到任何問題，EMR無伺服器會將工作執行移至不同的可用區域 (AZ)。

若要啟用作業的工作復原，請設定工作的重試原則。重試原則可確保EMR無伺服器會在任何時候失敗時自動重新啟動工作。批次和串流工作都支援重試原則，因此您可以根據使用案例自訂工作恢復能力。下表比較批次和串流作業之間的工作恢復能力的行為和差異。

	批次任務	串流工作
預設行為	不會重新執行工作。	當應用程式在執行工作時建立檢查點時，一律重試執行工作。
重試點	Batch 工作沒有檢查點，因此EMR無伺服器一律會從頭開始重新執行工作。	串流任務支援檢查點，因此您可以設定串流查詢以儲存執行時期狀態，並進度至 Amazon S3 中的檢查點位置。EMR無伺服器會繼續從檢查點執行的工作。如需詳細資訊，請參閱 Apache Spark 文件中的使用檢查點從失敗中復原。
重試嘗試次數上限	允許最多 10 次重試。	串流工作具有內建的鞭打防止控制功能，因此如果工作在一小時後繼續失敗，應用程式就會停止重試工作。一小時內的預設重試次數為五次。您可以將這個重試次數設定為介於 1

	批次任務	串流工作
		或 10 之間。您無法自訂最多嘗試次數。值 1 表示不重試。

當EMR無伺服器嘗試重新執行工作時，它也會使用嘗試編號為工作索引，以便您可以在各次嘗試中追蹤工作的生命週期。

您可以使用EMR無伺服器API作業或 AWS CLI 以變更工作恢復能力或查看與工作恢復能力相關的資訊。如需詳細資訊，請參閱EMR無伺服器API指南。

根據預設，EMR無伺服器不會重新執行批次工作。若要啟用批次工作的重試，請在開始批次工作執行時設定maxAttempts參數。此maxAttempts參數僅適用於批次工作。預設值為 1，表示不要重新執行工作。接受的值為 1 到 10 (含)。

下列範例示範如何在開始工作執行時指定最多 10 次嘗試次數。

```
aws emr-serverless start-job-run
 --application-id <APPLICATION_ID> \
 --execution-role-arn <JOB_EXECUTION_ROLE> \
 --mode 'BATCH' \
 --retry-policy '{
    "maxAttempts": 10
 }' \
 --job-driver '{
    "sparkSubmit": {
         "entryPoint": "/usr/lib/spark/examples/jars/spark-examples-does-not-
exist.jar",
         "entryPointArguments": ["1"],
         "sparkSubmitParameters": "--class org.apache.spark.examples.SparkPi"
     }
}'
```

EMR如果串流作業失敗，無伺服器會無限期地重試。若要防止因為重複無法復原的失敗而造成衝擊，請使用配置串流工作重試的防止衝擊控制。maxFailedAttemptsPerHour此參數可讓您指定EMR無伺服器停止重試前一小時內允許的失敗嘗試次數上限。預設值為 5。接受的值為 1 到 10 (含)。

```
aws emr-serverless start-job-run
 --application-id <APPPLICATION_ID> \
 --execution-role-arn <JOB_EXECUTION_ROLE> \
 --mode 'STREAMING' \
```

```
 --retry-policy '{
    "maxFailedAttemptsPerHour": 7
 }' \
 --job-driver '{
    "sparkSubmit": {
         "entryPoint": "/usr/lib/spark/examples/jars/spark-examples-does-not-
exist.jar",
         "entryPointArguments": ["1"],
         "sparkSubmitParameters": "--class org.apache.spark.examples.SparkPi"
     }
}'
```

您也可以使用其他作API業執行作業取得有關作業的資訊。例如，您可以將attempt參數與作業搭配使用，以取得有關特定工GetJobRun作嘗試的詳細資訊。如果未包含attempt參數，作業會傳回有關最新嘗試的資訊。

```
aws emr-serverless get-job-run \
    --job-run-id job-run-id \
    --application-id application-id \
    --attempt 1
```

此ListJobRunAttempts作業會傳回與工作執行相關之所有嘗試的相關資訊。

```
aws emr-serverless list-job-run-attempts \
  --application-id application-id \
  --job-run-id job-run-id
```

此GetDashboardForJobRun作業會建立並傳回URL可用來存取工作執行UIs的應用程式。該attempt參數可以讓你得到一URL個特定的嘗試。如果未包含attempt參數，作業會傳回有關最新嘗試的資訊。

```
aws emr-serverless get-dashboard-for-job-run \
    --application-id application-id \
    --job-run-id job-run-id \
    --attempt 1
```

使用重試政策監控作業

Job 備援支援也會新增EMR無伺服器工作執行重試的新事件。EMR無伺服器會在每次重試工作時發佈此事件。您可以使用此通知來追蹤工作的重試次數。如需有關事件的詳細資訊，請參閱 Amazon EventBridge 事件。

使用重試原則記錄

每次EMR無伺服器重試工作時，嘗試都會產生自己的記錄集。為確保EMR無伺服器可以成功地將這些日誌交付到 Amazon S3 和 Amazon， CloudWatch 而不會覆寫任何日誌，EMR無伺服器會在 S3 日誌路徑和日 CloudWatch 誌串流名稱的格式中新增前綴，以包含任務的嘗試編號。

以下是格式外觀的範例。

```
'/applications/<applicationId>/jobs/<jobId>/attempts/<attemptNumber>/'.
```

此格式可確保EMR無伺服器會將每次任務嘗試的所有日誌發佈到 Amazon S3 和 CloudWatch中的指定位置。如需詳細資訊，請參閱儲存記錄檔。

> **Note**
>
> EMR無伺服器只會在所有串流工作和任何已啟用重試的批次工作中使用此前置字元格式。

中繼存儲配置

Hive 中繼存放區是儲存資料表的結構資訊 (包括結構描述、磁碟分割名稱和資料類型) 的集中位置。使用EMR無伺服器，您可以將此資料表中繼資料保留在具有工作存取權的中繼存放區中。

Hive 中繼存儲有兩個選項：

- 所以此 AWS Glue Data Catalog
- 一個外部的阿帕奇蜂巢元存儲

使用 AWS Glue 資料目錄做為中繼存放區

您可以配置您的星火和蜂巢作業使用 AWS Glue 資料目錄做為其中繼存放區。當您需要永久性中繼存放區或不同應用程式、服務或共用的中繼存放區時，我們建議您使用此設定 AWS 帳戶。 如需有關「資料目錄」的詳細資訊，請參閱填入 AWS Glue 資料型錄。有關信息 AWS Glue 定價，請參閱 AWS Glue 定價。

您可以將EMR無伺服器工作設定為使用 AWS Glue 資料目錄在相同的 AWS 帳戶 作為您的應用程序，或在不同的 AWS 帳戶.

配置 AWS Glue Data Catalog

若要設定資料目錄，請選擇您要使用的EMR無伺服器應用程式類型。

Spark

當您使用 EMR Studio 與EMR無伺服器 Spark 應用程式來執行工作時， AWS Glue 資料目錄是預設的中繼存放區。

當您使用SDKs或 AWS CLI，您可以在工作執行的sparkSubmit參數com.amazonaws.glue.catalog.metastore.AWSGlueDataCatalogHiveClientFactory中將spark.hadoop.hive.metastore.client.factory.class組態設定為。下列範例顯示如何使用 AWS CLI.

```
aws emr-serverless start-job-run \
    --application-id application-id \
    --execution-role-arn job-role-arn \
    --job-driver '{
        "sparkSubmit": {
            "entryPoint": "s3://DOC-EXAMPLE-BUCKET/code/pyspark/extreme_weather.py",
            "sparkSubmitParameters": "--conf
 spark.hadoop.hive.metastore.client.factory.class=com.amazonaws.glue.catalog.metastore.AWSGl
 --conf spark.driver.cores=1 --conf spark.driver.memory=3g --conf
 spark.executor.cores=4 --conf spark.executor.memory=3g"
        }
    }'
```

或者，您可以SparkSession在 Spark 代碼中創建新配置時設置此配置。

```
from pyspark.sql import SparkSession

spark = (
    SparkSession.builder.appName("SparkSQL")
    .config(
        "spark.hadoop.hive.metastore.client.factory.class",
        "com.amazonaws.glue.catalog.metastore.AWSGlueDataCatalogHiveClientFactory",
    )
    .enableHiveSupport()
    .getOrCreate()
)

# we can query tables with SparkSQL
spark.sql("SHOW TABLES").show()
```

```
# we can also them with native Spark
print(spark.catalog.listTables())
```

Hive

對於EMR無伺服器 Hive 應用程式，資料目錄是預設的中繼存放區。也就是說，當您在EMR無伺服器 Hive 應用程式上執行作業時，Hive 會將中繼存放區資訊記錄在資料目錄中 AWS 帳戶 作為您的應用程序。您不需要虛擬私有雲 (VPC) 即可使用資料目錄做為中繼存放區。

要訪問 Hive 元存儲表，添加所需的 AWS 設定IAM權限中概述的 Glue 合原則 AWS Glue 。

設定EMR無伺服器和跨帳戶存取 AWS Glue Data Catalog

若要設定EMR無伺服器的跨帳戶存取，您必須先登入下列項目 AWS 帳戶:

- AccountA— 一個 AWS 帳戶 您已建立EMR無伺服器應用程式的位置。
- AccountB— 一個 AWS 帳戶 包含一個 AWS 您希望EMR無伺服器工作執行存取的 Glue 資料型錄。

1. 請確定中的管理員或其他授權身分將資源策略AccountB附加至中的資料目錄AccountB。此原則會授與AccountA特定的跨帳戶權限，以便對AccountB目錄中的資源執行作業。

```
{
  "Version" : "2012-10-17",
  "Statement" : [ {
    "Effect" : "Allow",
    "Principal": {
        "AWS": [
            "arn:aws:iam::accountA:role/job-runtime-role-A"
        ]},
    "Action" : [
        "glue:GetDatabase",
        "glue:CreateDatabase",
        "glue:GetDataBases",
        "glue:CreateTable",
        "glue:GetTable",
        "glue:UpdateTable",
        "glue:DeleteTable",
        "glue:GetTables",
        "glue:GetPartition",
        "glue:GetPartitions",
```

```
        "glue:CreatePartition",
        "glue:BatchCreatePartition",
        "glue:GetUserDefinedFunctions"
    ],
    "Resource": ["arn:aws:glue:region:AccountB:catalog"]
  } ]
}
```

2. 將IAM原則新增至中的EMR無伺服器工作執行階段角色，以AccountA便角色可以存取中AccountB的資料目錄資源。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:GetDatabase",
        "glue:CreateDatabase",
        "glue:GetDataBases",
        "glue:CreateTable",
        "glue:GetTable",
        "glue:UpdateTable",
        "glue:DeleteTable",
        "glue:GetTables",
        "glue:GetPartition",
        "glue:GetPartitions",
        "glue:CreatePartition",
        "glue:BatchCreatePartition",
        "glue:GetUserDefinedFunctions"
      ],
      "Resource": ["arn:aws:glue:region:AccountB:catalog"]
    }
  ]
}
```

3. 開始工作執行。此步驟會根據EMR無伺服器應用程式類型而略有不同。AccountA

   Spark

   如下列範例所示，在hive-site分類中設定spark.hadoop.hive.metastore.glue.catalogid屬性。Replace (取代) *##-## ID* 其中包含資料目錄的 ID AccountB。

```
aws emr-serverless start-job-run \
--application-id "application-id" \
--execution-role-arn "job-role-arn" \
--job-driver '{
    "sparkSubmit": {
        "query": "s3://DOC-EXAMPLE-BUCKET/hive/scripts/create_table.sql",
        "parameters": "--hiveconf hive.exec.scratchdir=s3://DOC-EXAMPLE-BUCKET/
hive/scratch --hiveconf hive.metastore.warehouse.dir=s3://DOC-EXAMPLE-BUCKET/
hive/warehouse"
    }
  }' \
--configuration-overrides '{
    "applicationConfiguration": [{
        "classification": "hive-site",
        "properties": {
            "spark.hadoop.hive.metastore.glue.catalogid": "AccountB-catalog-id"
        }
    }]
}'
```

Hive

如下列範例所示，在hive-site分類中設定hive.metastore.glue.catalogid屬性。Replace (取代) *##-## ID* 其中包含資料目錄的 ID AccountB。

```
aws emr-serverless start-job-run \
--application-id "application-id" \
--execution-role-arn "job-role-arn" \
--job-driver '{
    "hive": {
    "query": "s3://DOC-EXAMPLE-BUCKET/hive/scripts/create_table.sql",
    "parameters": "--hiveconf hive.exec.scratchdir=s3://DOC-EXAMPLE-BUCKET/hive/
scratch --hiveconf hive.metastore.warehouse.dir=s3://DOC-EXAMPLE-BUCKET/hive/
warehouse"
    }
}' \
--configuration-overrides '{
    "applicationConfiguration": [{
        "classification": "hive-site",
        "properties": {
            "hive.metastore.glue.catalogid": "AccountB-catalog-id"
        }
```

```
    }]
}'
```

使用時的注意事項 AWS Glue Data Catalog

您可以在 Hive 腳本ADD JAR中添加輔助JARs。如需其他考量，請參閱使用時的考量事 AWS Glue 資料型錄。

使用外部蜂巢元存儲

您可以將EMR無伺服器 Spark 和 Hive 任務設定為連接到外部 Hive 中繼存放區，例如 Amazon Aurora 或 Amazon RDS for My. SQL 本節說明如何設定 Amazon RDS Hive 中繼存放區、設定和設定VPCEMR無伺服器任務以使用外部中繼存放區。

創建一個外部蜂巢元存儲

1. 依照建立一VPC個中的指示，建立具有私有子網路的 Amazon Virtual Private Cloud (AmazonVPC)。
2. 使用新的 Amazon VPC 和私有子網路建立EMR無伺服器應用程式。當您使用設定EMR無伺服器應用程式時VPC，它會先為您指定的每個子網路佈建 elastic network interface。然後，它會將您指定的安全性群組附加至該網路介面。這使您的應用程序訪問控制。如需如何設定的詳細資訊VPC，請參閱設定VPC存取權。
3. 在 Amazon VPC 的私有子網中創建一個我的SQL或 Aurora Postgre SQL 數據庫。如需如何建立 Amazon 資RDS料庫的詳細資訊，請參閱建立 Amazon 資料RDS庫執行個體。
4. 依照修改 Amazon RDS 資料庫執行個體中的步驟，修改 My SQL 或 Aurora 資料庫的安全群組，以允許來自EMR無伺服器安全群組的JDBC連線。從其中一個EMR無伺服器RDS安全性群組新增輸入流量的規則至安全群組。

Type	通訊協定	連接埠範圍	來源
全部 TCP	TCP	3306	`emr-serverless-security-group`

配置星火選項

使用 JDBC

若要將EMR無伺服器 Spark 應用程式設定RDS為連線到以 Amazon 為基礎的我的SQL或亞馬 Amazon Aurora My SQL 執行個體的 Hive 中繼存放區，請使用連線。JDBC在作業運`--jars`行的`spark-submit`參數中傳遞`mariadb-connector-java.jar`與。

```
aws emr-serverless start-job-run \
  --application-id "application-id" \
  --execution-role-arn "job-role-arn" \
  --job-driver '{
        "sparkSubmit": {
            "entryPoint": "s3://DOC-EXAMPLE-BUCKET/scripts/spark-jdbc.py",
            "sparkSubmitParameters": "--jars s3://DOC-EXAMPLE-BUCKET/mariadb-connector-
java.jar
            --conf
 spark.hadoop.javax.jdo.option.ConnectionDriverName=org.mariadb.jdbc.Driver
            --conf spark.hadoop.javax.jdo.option.ConnectionUserName=<connection-user-
name>
            --conf spark.hadoop.javax.jdo.option.ConnectionPassword=<connection-
password>
            --conf spark.hadoop.javax.jdo.option.ConnectionURL=<JDBC-Connection-
string>
            --conf spark.driver.cores=2
            --conf spark.executor.memory=10G
            --conf spark.driver.memory=6G
            --conf spark.executor.cores=4"
        }
    }' \
    --configuration-overrides '{
        "monitoringConfiguration": {
        "s3MonitoringConfiguration": {
            "logUri": "s3://DOC-EXAMPLE-BUCKET/spark/logs/"
        }
    }
}'
```

下面的代碼示例是一個 Spark 入口點腳本，它與 Amazon 上的 Hive 中繼存儲進行交互。RDS

```
from os.path import expanduser, join, abspath
from pyspark.sql import SparkSession
```

```
from pyspark.sql import Row
# warehouse_location points to the default location for managed databases and tables
warehouse_location = abspath('spark-warehouse')
spark = SparkSession \
    .builder \
    .config("spark.sql.warehouse.dir", warehouse_location) \
    .enableHiveSupport() \
    .getOrCreate()
spark.sql("SHOW DATABASES").show()
spark.sql("CREATE EXTERNAL TABLE `sampledb`.`sparknyctaxi`(`dispatching_base_num`
 string, `pickup_datetime` string, `dropoff_datetime` string, `pulocationid` bigint,
 `dolocationid` bigint, `sr_flag` bigint) STORED AS PARQUET LOCATION 's3://<s3 prefix>/
nyctaxi_parquet/'")
spark.sql("SELECT count(*) FROM sampledb.sparknyctaxi").show()
spark.stop()
```

使用節儉服務

您可以將EMR無伺服器 Hive 應用程式設定RDS為基於我的SQL或 Amazon Amazon Aurora 我SQL的執行個體的亞馬遜連接到 Hive 中繼存放區。若要這麼做，請在現有 Amazon EMR 叢集的主節點上執行節點伺服器。如果您已經擁有一個含節儉伺服器的 Amazon EMR 叢集，您想要使用它來簡化EMR無伺服器任務組態，則此選項非常適合。

```
aws emr-serverless start-job-run \
  --application-id "application-id" \
  --execution-role-arn "job-role-arn" \
  --job-driver '{
        "sparkSubmit": {
            "entryPoint": "s3://DOC-EXAMPLE-BUCKET/thriftscript.py",
            "sparkSubmitParameters": "--jars s3://DOC-EXAMPLE-BUCKET/mariadb-connector-
java.jar
            --conf spark.driver.cores=2
            --conf spark.executor.memory=10G
            --conf spark.driver.memory=6G
            --conf spark.executor.cores=4"
        }
    }' \
    --configuration-overrides '{
        "monitoringConfiguration": {
        "s3MonitoringConfiguration": {
            "logUri": "s3://DOC-EXAMPLE-BUCKET/spark/logs/"
        }
    }
```

```
}'
```

下列程式碼範例是使用節儉通訊協定連線至 Hive 中繼存放區的入口點指令碼 (`thriftscript.py`)。請注意，該`hive.metastore.uris`屬性需要被設置為從外部 Hive 元存儲中讀取。

```
from os.path import expanduser, join, abspath
from pyspark.sql import SparkSession
from pyspark.sql import Row
# warehouse_location points to the default location for managed databases and tables
warehouse_location = abspath('spark-warehouse')
spark = SparkSession \
    .builder \
    .config("spark.sql.warehouse.dir", warehouse_location) \
    .config("hive.metastore.uris","thrift://thrift-server-host:thift-server-port") \
    .enableHiveSupport() \
    .getOrCreate()
spark.sql("SHOW DATABASES").show()
spark.sql("CREATE EXTERNAL TABLE sampledb.`sparknyctaxi`( `dispatching_base_num`
 string, `pickup_datetime` string, `dropoff_datetime` string, `pulocationid` bigint,
 `dolocationid` bigint, `sr_flag` bigint) STORED AS PARQUET LOCATION 's3://<s3 prefix>/
nyctaxi_parquet/'")
spark.sql("SELECT * FROM sampledb.sparknyctaxi").show()
spark.stop()
```

配置蜂巢選項

使用 JDBC

如果您想要在 Amazon RDS My SQL 或 Amazon Aurora 執行個體上指定外部 Hive 資料庫位置，可以覆寫預設的中繼存放區組態。

> **Note**
>
> 在 Hive 中，您可以同時對中繼存儲表執行多次寫入。如果您在兩個作業之間共用中繼儲存庫資訊，請確定您不會同時寫入相同的中繼儲存庫資料表，除非您寫入同一個中繼儲存表格的不同分割區。

在`hive-site`分類中設置以下配置以激活外部 Hive 中繼存儲。

```
{
```

```
    "classification": "hive-site",
    "properties": {
        "hive.metastore.client.factory.class":
 "org.apache.hadoop.hive.ql.metadata.SessionHiveMetaStoreClientFactory",
        "javax.jdo.option.ConnectionDriverName": "org.mariadb.jdbc.Driver",
        "javax.jdo.option.ConnectionURL": "jdbc:mysql://db-host:db-port/db-name",
        "javax.jdo.option.ConnectionUserName": "username",
        "javax.jdo.option.ConnectionPassword": "password"
    }
}
```

使用節儉伺服器

您可以將EMR無伺服器蜂巢應用程式設定為連接到以我SQL或 Amazon Amazon Aurora M 為基礎的亞馬遜RDS的 Hive 中繼存放區。ySQLinstance若要這麼做，請在現有 Amazon EMR 叢集的主節點上執行節儉伺服器。如果您已經擁有執行節儉伺服器的 Amazon EMR 叢集，並且想要使用EMR無伺服器任務組態，則此選項非常適合。

在`hive-site`分類中設定下列組態，以便EMR無伺服器可以存取遠端節儉中繼存放區。請注意，您必須將`hive.metastore.uris`屬性設置為從外部 Hive 中繼存儲中讀取。

```
{
    "classification": "hive-site",
    "properties": {
        "hive.metastore.client.factory.class":
 "org.apache.hadoop.hive.ql.metadata.SessionHiveMetaStoreClientFactory",
        "hive.metastore.uris": "thrift://thrift-server-host:thirft-server-port"
    }
}
```

使用外部中繼存放區時的考量事項

- 您可以配置與 MariaDB 兼容的數據庫JDBC作為您的中繼存儲。這些資料庫的範例適RDS用於 MariaDB SQL、我的和 Amazon Aurora。
- 元存儲不會自動初始化。如果您的元存儲未使用 Hive 版本的模式初始化，請使用配置單元架構工具。
- EMR無伺服器不支援 Kerberos 驗證。您不能將節儉中繼存放區伺服器與 Kerberos 驗證搭配EMR無伺服器 Spark 或 Hive 作業搭配使用。

在另一個存取 S3 資料 AWS 來自EMR無伺服器的帳戶

您可以從一個方式執行 Amazon EMR 無伺服器任務 AWS 帳戶並將其設定為存取屬於另一個儲存貯體的 Amazon S3 儲存貯體中的資料 AWS 帳戶。本頁說明如何設定從EMR無伺服器對 S3 的跨帳戶存取。

在EMR無伺服器上執行的任務可以使用 S3 儲存貯體政策或假定角色從不同的角色存取 Amazon S3 中的資料 AWS 帳戶。

必要條件

若要為 Amazon EMR 無伺服器設定跨帳戶存取，您必須在登入兩個任務時完成任務 AWS 帳戶：

- **AccountA**— 這就是 AWS 您已建立 Amazon EMR 無伺服器應用程式的帳戶。在設定跨帳戶存取權之前，您必須在此帳戶中準備好下列項目：
  - 您想要在其中執行任務的 Amazon EMR 無伺服器應用程式。
  - 具有在應用程式中執行工作所需權限的工作執行角色。如需詳細資訊，請參閱Amazon EMR 無伺服器的 Job 務執行階段角色。
- **AccountB**— 這就是 AWS 包含您希望 Amazon EMR 無伺服器任務存取的 S3 儲存貯體的帳戶。

使用 S3 儲存貯體政策存取跨帳戶 S3 資料

若要存取 S3 儲存貯體 account B from account A，將下列政策附加到 S3 儲存貯體 account B.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Example permissions 1",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountA:root"
         },
         "Action": [
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::bucket_name_in_AccountB"
         ]
      },
```

```
      {
         "Sid": "Example permissions 2",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountA:root"
         },
         "Action": [
            "s3:PutObject",
            "s3:GetObject",
            "s3:DeleteObject"
         ],
         "Resource": [
            "arn:aws:s3:::bucket_name_in_AccountB/*"
         ]
      }
   ]
}
```

如需使用 S3 儲存貯體政策進行 S3 跨帳戶存取的詳細資訊，請參閱 Amazon 簡單儲存貯體服務使用者指南中的範例 2：儲存貯體擁有者授予跨帳戶儲存貯體許可。

使用假定角色存取跨帳戶 S3 資料

另一種為 Amazon EMR 無伺服器設定跨帳戶存取的方法是使AssumeRole用 AWS Security Token Service (AWS STS). AWS STS 是一項全域 Web 服務，可讓您為使用者請求臨時、有限權限的憑證。您可以使用您建立的臨時安全登入資料API呼叫EMR無伺服器和 Amazon S3。AssumeRole

下列步驟說明如何使用假定角色從EMR無伺服器存取跨帳戶 S3 資料：

1. 創建一個 Amazon S3 存儲桶，*cross-account-bucket*，在中AccountB。如需詳細資訊，請參閱 Amazon 簡單儲存服務使用者指南中的建立儲存貯體。如果想要擁有對 DynamoDB 的跨帳戶存取權，也可以在 AccountB 中建立 DynamoDB 資料表。如需詳細資訊，請參閱 Amazon DynamoDB 開發人員指南中的建立 DynamoDB 表格。
2. 在中建立可AccountB存取的Cross-Account-Role-BIAM角色 *cross-account-bucket*.
   a. 登入 AWS Management Console 然後在開啟IAM主控台https://console.aws.amazon.com/iam/。
   b. 選擇角色，並建立一個新角色 Cross-Account-Role-B。如需有關如何建立IAM角色的詳細資訊，請參閱《IAM使用指南》中的〈建立IAM角色〉。
   c. 建立指定存取權限Cross-Account-Role-B的IAM策略 *cross-account-bucket* S3 儲存貯體，如下列政策聲明所示。然後將IAM原則附加至Cross-Account-Role-B。如需詳細資訊，請參閱《使用指南》中的IAM〈建立IAM策略〉。

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::cross-account-bucket",
                "arn:aws:s3:::cross-account-bucket/*"
            ]
        }
    ]
}
```

如果您需要 DynamoDB 存取權，請建立一個IAM政策，以指定存取跨帳戶 DynamoDB 表的權限。然後將IAM原則附加至Cross-Account-Role-B。如需詳細資訊，請參閱 Amazon DynamoDB：允許存取IAM使用者指南中的特定表格。

以下是允許存取 DynamoDB 表格的政策。CrossAccountTable

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "dynamodb:*",
            "Resource": "arn:aws:dynamodb:MyRegion:AccountB:table/CrossAccountTable"
        }
    ]
}
```

3. 編輯 Cross-Account-Role-B 角色的信任關係。
   a. 若要設定角色的信任關係，請為您在步驟 2 中建立的角色選擇IAM主控台中Cross-Account-Role-B的 [信任關係] 索引標籤。
   b. 選取編輯信任關係。
   c. 新增下列原則文件。這允許Job-Execution-Role-A在中AccountA擔任該Cross-Account-Role-B角色。

```
{
```

```
"Version": "2012-10-17",
  "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::AccountA:role/Job-Execution-Role-A"
        },
        "Action": "sts:AssumeRole"
    }
  ]
}
```

4. 授予 Job-Execution-Role-A AccountA AWS STS AssumeRole允許假設Cross-Account-Role-B。

   a. 在IAM控制台中 AWS 帳戶AccountA中，選取Job-Execution-Role-A。

   b. 將以下政策陳述式新增至 Job-Execution-Role-A 以允許 Cross-Account-Role-B 角色的 AssumeRole 動作。

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::AccountB:role/Cross-Account-Role-B"
        }
    ]
}
```

假定的角色示例

您可以使用單一假定角色存取帳戶中的所有 S3 資源，或者使用 Amazon EMR 6.11 及更高版本，您可以設定多個IAM角色，以在存取不同的跨帳戶 S3 儲存貯體時承擔。

主題

- 使用一個假定角色存取 S3 資源
- 存取具有多個假定角色的 S3 資源

使用一個假定角色存取 S3 資源

> **Note**
>
> 當您將任務設定為使用單一假定角色時，整個任務中的所有 S3 資源都會使用該角色，包括指entryPoint令碼。

如果您想要使用單一假定角色來存取帳戶 B 中的所有 S3 資源，請指定下列組態：

1. `fs.s3.customAWSCredentialsProvider`將EMRFS模型組態指定為`spark.hadoop.fs.s3.customAWSCredentialsProvider=com.amazonaws.emr.AssumeRole`
2. 對於 Spark，請使用`spark.emr-serverless.driverEnv.ASSUME_ROLE_CREDENTIALS_ROLE_ARN`和`spark.executorEnv.ASSUME`指定驅動程式和執行程式上的環境變數。
3. 對於 Hive，請使用`hive.emr-serverless.launch.env.ASSUME_ROLE_CREDENTIALS_ROLE_ARN`、`tez.am.emr-serverless.launch.env.ASSUME_ROLE_CREDENTIALS_ROLE_ARN`、和`tez.task.emr-serverless.launch.env.ASSUME_ROLE_CREDENTIALS_ROLE_ARN`來指定 Hive 驅動程式、Tez 應用程式主機和 Tez 工作容器上的環境變數。

下列範例顯示如何使用假定角色，以跨帳戶存取啟動EMR無伺服器工作執行。

Spark

下列範例顯示如何使用假定角色，透過跨帳戶存取 S3 來啟動EMR無伺服器 Spark 工作執行。

```
aws emr-serverless start-job-run \
    --application-id application-id \
    --execution-role-arn job-role-arn \
    --job-driver '{
        "sparkSubmit": {
            "entryPoint": "entrypoint_location",
            "entryPointArguments": [":argument_1:", ":argument_2:"],
            "sparkSubmitParameters": "--conf spark.executor.cores=4 --conf
 spark.executor.memory=20g --conf spark.driver.cores=4 --conf spark.driver.memory=8g
 --conf spark.executor.instances=1"
        }
    }' \
```

```
    --configuration-overrides '{
       "applicationConfiguration": [{
           "classification": "spark-defaults",
           "properties": {
               "spark.hadoop.fs.s3.customAWSCredentialsProvider":
 "spark.hadoop.fs.s3.customAWSCredentialsProvider=com.amazonaws.emr.AssumeRoleAWSCredentials
               "spark.emr-serverless.driverEnv.ASSUME_ROLE_CREDENTIALS_ROLE_ARN":
 "arn:aws:iam::AccountB:role/Cross-Account-Role-B",
               "spark.executorEnv.ASSUME_ROLE_CREDENTIALS_ROLE_ARN":
 "arn:aws:iam::AccountB:role/Cross-Account-Role-B"
           }
       }]
   }'
```

Hive

下列範例顯示如何使用假定角色，透過跨帳戶存取 S3 來啟動EMR無伺服器 Hive 任務執行。

```
aws emr-serverless start-job-run \
    --application-id application-id \
    --execution-role-arn job-role-arn \
    --job-driver '{
        "hive": {
            "query": "query_location",
            "parameters": "hive_parameters"
        }
    }' \
    --configuration-overrides '{
        "applicationConfiguration": [{
            "classification": "hive-site",
            "properties": {
                "fs.s3.customAWSCredentialsProvider":
 "com.amazonaws.emr.serverless.credentialsprovider.AssumeRoleAWSCredentialsProvider",
                "hive.emr-serverless.launch.env.ASSUME_ROLE_CREDENTIALS_ROLE_ARN":
 "arn:aws:iam::AccountB:role/Cross-Account-Role-B",
                "tez.am.emr-serverless.launch.env.ASSUME_ROLE_CREDENTIALS_ROLE_ARN":
 "arn:aws:iam::AccountB:role/Cross-Account-Role-B",
                "tez.task.emr-
serverless.launch.env.ASSUME_ROLE_CREDENTIALS_ROLE_ARN":
 "arn:aws:iam::AccountB:role/Cross-Account-Role-B"
            }
        }]
    }'
```

存取具有多個假定角色的 S3 資源

在EMR無伺服器版本 6.11.0 及更新版本中，您可以設定多個IAM角色，以便在存取不同的跨帳戶值區時採用。如果您想要在帳戶 B 中使用不同的假定角色存取不同的 S3 資源，請在開始任務執行時使用下列組態：

1. `fs.s3.customAWSCredentialsProvider`將EMRFS模型組態指定為`com.amazonaws.emr.serverless.credentialsprovider.BucketLevelAssumeRoleCredent`
2. 指定EMRFS組態，`fs.s3.bucketLevelAssumeRoleMapping`以定義從 S3 儲存貯體名稱到要假設的帳戶 B 中IAM角色的對應。該值的格式應為`bucket1->role1;bucket2->role2`。

例如，您可以使用存`arn:aws:iam::AccountB:role/Cross-Account-Role-B-1`取值區`bucket1`，以及用`arn:aws:iam::AccountB:role/Cross-Account-Role-B-2`來存取值區`bucket2`。下列範例顯示如何透過多個假定角色以跨帳戶存取來啟動EMR無伺服器工作執行。

Spark

下列範例顯示如何使用多個假定角色來建立EMR無伺服器 Spark 工作執行。

```
aws emr-serverless start-job-run \
    --application-id application-id \
    --execution-role-arn job-role-arn \
    --job-driver '{
        "sparkSubmit": {
            "entryPoint": "entrypoint_location",
            "entryPointArguments": [":argument_1:", ":argument_2:"],
            "sparkSubmitParameters": "--conf spark.executor.cores=4 --conf
 spark.executor.memory=20g --conf spark.driver.cores=4 --conf spark.driver.memory=8g
 --conf spark.executor.instances=1"
        }
    }' \
     --configuration-overrides '{
        "applicationConfiguration": [{
            "classification": "spark-defaults",
            "properties": {
                "spark.hadoop.fs.s3.customAWSCredentialsProvider":
 "com.amazonaws.emr.serverless.credentialsprovider.BucketLevelAssumeRoleCredentialsProvider"
                "spark.hadoop.fs.s3.bucketLevelAssumeRoleMapping":
 "bucket1->arn:aws:iam::AccountB:role/Cross-Account-Role-B-1;bucket2-
>arn:aws:iam::AccountB:role/Cross-Account-Role-B-2"
            }
```

```
        }]
    }'
```

Hive

下列範例顯示如何使用多個假定角色來建立EMR無伺服器 Hive 工作執行。

```
aws emr-serverless start-job-run \
    --application-id application-id \
    --execution-role-arn job-role-arn \
    --job-driver '{
        "hive": {
            "query": "query_location",
            "parameters": "hive_parameters"
        }
    }' \
    --configuration-overrides '{
        "applicationConfiguration": [{
            "classification": "hive-site",
            "properties": {
                "fs.s3.customAWSCredentialsProvider":
 "com.amazonaws.emr.serverless.credentialsprovider.AssumeRoleAWSCredentialsProvider",
                "fs.s3.bucketLevelAssumeRoleMapping": "bucket1-
>arn:aws:iam::AccountB:role/Cross-Account-Role-B-1;bucket2-
>arn:aws:iam::AccountB:role/Cross-Account-Role-B-2"
            }
        }]
    }'
```

疑難排解EMR無伺服器的錯誤

使用下列資訊協助診斷和修正使用 Amazon EMR 無伺服器時可能會遇到的常見問題。

主題

- 錯誤:超過允許容量上限。
- 錯誤：已超過設定的最大容量。請稍後再試。
- 錯誤：S3 存取被拒絕。請檢查所需 S3 資源上任務執行階段角色的 S3 存取權限。
- 錯誤: ModuleNotFoundError: 沒有命名的模組<module>。有關如何在EMR無服務器中使用 python 庫，請參閱用戶指南。

錯誤:超過允許容量上限。

此錯誤表示無EMR伺服器無法提交工作，因為應用程式已超過您設定的容量上限。增加應用程式的最大容量限制。

錯誤：已超過設定的最大容量。請稍後再試。

此錯誤表示無EMR伺服器無法啟動新工作，因為應用程式已超過您設定的容量上限。增加應用程式的最大容量限制。

錯誤：S3 存取被拒絕。請檢查所需 S3 資源上任務執行階段角色的 S3 存取權限。

此錯誤表示您的任務無法存取 S3 資源。確認工作執行階段角色具有存取工作需要使用的 S3 資源的權限。若要進一步瞭解執行階段角色，請參閱Amazon EMR 無伺服器的 Job 務執行階段角色。

錯誤: ModuleNotFoundError: 沒有命名的模組<module>。有關如何在EMR無服務器中使用 python 庫，請參閱用戶指南。

這個錯誤表明一個 Python 模塊不可用於星火作業。檢查相依的 Python 程式庫是否可供工作使用。如需如何封裝 Python 程式庫的詳細資訊，請參閱搭配EMR無伺服器使用 Python 程式庫。

錯誤：無法承擔執行角色，<role name>因為它不存在或未使用必要的信任關係進行設定。

此錯誤表示您為工作指定的工作執行階段角色不存在，或者該角色沒有EMR無伺服器權限的信任關係。若要驗證IAM角色是否存在，並驗證您是否已正確設定角色的信任原則，請參閱中的指示Amazon EMR 無伺服器的 Job 務執行階段角色。

透EMR過 Studio 以EMR無伺服器執行互動式工作負載

概觀

互動式應用程式是啟用互動式功能的EMR無伺服器應用程式。使用 Amazon EMR 無伺服器互動式應用程式，您可以使用 Amazon Studio 中管理的 Jupyter 筆記本執行互動式工作負載。EMR這有助於資料工程師、資料科學家和資料分析師使用 EMR Studio，透過 Amazon S3 和 Amazon DynamoDB 等資料存放區中的資料集執行互動式分析。

EMR無伺服器中互動式應用程式的使用案例包括：

- 資料工程師使用 EMR Studio 中的IDE經驗來建立ETL指令碼。指令碼會從現場部署擷取資料、轉換資料以進行分析，然後將資料存放在 Amazon S3。
- 資料科學家使用筆記本來探索資料集，並訓練機器學習 (ML) 模型，以偵測資料集中的異常情況。
- 資料分析師會探索資料集並建立可產生每日報告的指令碼，以更新商業儀表板等應用

必要條件

若要搭配EMR無伺服器使用互動式工作負載，您必須符合下列需求：

- EMRAmazon EMR 6.14.0 及更高版本支援無伺服器互動式應用程式。
- 若要存取互動式應用程式、執行您提交的工作負載，以及從 EMR Studio 執行互動式筆記本，您需要特定的權限和角色。如需詳細資訊，請參閱互動式工作負載所需權限。

互動式工作負載所需權限

除了存取EMR無伺服器所需的基本權限之外，您還必須為IAM身分或角色設定其他權限：

存取您的互動式應用程式

設定 EMR Studio 的使用者和工作區權限。如需詳細資訊，請參閱 Amazon EMR 管理指南中的設定 EMR Studio 使用者許可。

執行透過EMR無伺服器提交的工作負載

設定工作執行時期角色。如需詳細資訊，請參閱建立工作執行時期角色。

若要從 EMR Studio 執行互動式筆記本

將下列其他權限新增至 Studio 使用者的IAM原則：

- **emr-serverless:AccessInteractiveEndpoints**-授予訪問和連接到您指定為的交互式應用程序的權限Resource。需要此權限才能從 EMR Studio 工作區附加至EMR無伺服器應用程式。
- **iam:PassRole**-授與存取您計劃在附加至應用程式時使用的IAM執行角色的權限。從 EMR Studio 工作區附加至EMR無伺服器應用程式需要適當的PassRole權限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EMRServerlessInteractiveAccess",
            "Effect": "Allow",
            "Action": "emr-serverless:AccessInteractiveEndpoints",
            "Resource": "arn:aws:emr-serverless:Region:account:/applications/*"
        },
        {
            "Sid": "EMRServerlessRuntimeRoleAccess",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "interactive-execution-role-ARN",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "emr-serverless.amazonaws.com"
                }
            }
        }
    ]
}
```

配置互動式應用

使用下列高階步驟建立具有來自 Amazon EMR Studio 的互動式功能的EMR無伺服器應用程式 AWS Management Console.

1. 依照中的步驟開始使用 Amazon EMR 無伺服器建立應用程式。
2. 然後，從 EMR Studio 啟動工作區，並作為運算選項連接到EMR無伺服器應用程式。如需詳細資訊，請參閱EMR無伺服器入門文件的步驟 2 中的互動式工作負載索引標籤。

當您將應用程式附加到 Studio 工作區時，如果應用程式尚未執行，應用程式會自動啟動觸發。您也可以預先啟動應用程式，並在將應用程式附加至工作區之前準備就緒。

互動式應用程式考量

- EMRAmazon EMR 6.14.0 及更高版本支援無伺服器互動式應用程式。
- EMRStudio 是唯一與EMR無伺服器互動式應用程式整合的用戶端。EMR無伺服器互動式應用程式不支援下列 EMR Studio 功能：工作區共同作業、SQL總管，以及筆記本的程式設計執行。
- 只有 Spark 引擎才支援互動式應用程式。
- 交互式應用程序支持 Python 3 PySpark 和星火斯卡拉內核。
- 您可以在單一互動式應用程式上同時執行 25 筆記本。
- 沒有支援具有互動式應用程式的自我託管 Jupyter 筆記本的端點或API介面。
- 為了獲得最佳化的啟動體驗，我們建議您為驅動程式和執行程式設定預先初始化容量，並預先啟動應用程式。當您預先啟動應用程式時，請確保應用程式在您想要將其附加至工作區時已準備就緒。

```
aws emr-serverless start-application \
--application-id your-application-id
```

- 根據預設，`autoStopConfig`會針對應用程式啟用。這會在閒置 30 分鐘後關閉應用程式。您可以在`create-application`或`update-application`請求中變更此組態。
- 使用互動式應用程式時，建議您設定核心、驅動程式和執行程式的預先初始化容量，以執行筆記本。每個 Spark 互動式工作階段都需要一個核心和一個驅動程式，因此EMR無伺服器會為每個預先初始化的驅動程式維護預先初始化 根據預設，即使您未為驅動程式指定任何預先初始化容量，EMRServerless 仍會在整個應用程式中維護一個核心工作站的預先初始化容量。每個核心工作者都使用 4 v CPU 和 16 GB 的記憶體。如需目前的定價資訊，請參閱 Amazon EMR 定價頁面。
- 您必須擁有足夠的 v CPU 服務配額 AWS 帳戶 以執行互動式工作負載。如果您未執行啟用 Lake 格式化的工作負載，建議至少 24 v。CPU如果這樣做，我們建議至少 28 v CPU。
- EMR如果筆記型電腦閒置超過 60 分鐘，無伺服器會自動終止核心。EMR無伺服器會計算筆記型電腦工作階段期間上次完成活動的核心閒置時間。您目前無法修改核心閒置逾時設定。
- 若要啟用具有互動式工作負載的 Lake Formation，請在建立EMR無伺服器應用程式時，`spark.emr-serverless.lakeformation.enabled`將組態設定為在`runtime-configuration`物件中的`spark-defaults`分類`true`下。要了解有關在EMR無伺服器中啟用 Lake Formation 的更多信息，請參閱在 Amazon EMR 中啟用 Lake Formation。

透過 Apache Livy 端點使用EMR無伺服器執行互動式工作負載

使用 Amazon 6.14.0 及更高EMR版本，您可以建立並啟用 Apache Livy 端點，同時建立EMR無伺服器應用程式，並透過自我託管筆記本或自訂用戶端執行互動式工作負載。Apache 利維端點提供以下優點：

- 您可以透過 Jupyter 筆記本安全地連線到 Apache Livy 端點，並透過 Apache Livy 的介面管理 Apache Spark 工作負載。REST
- 針對使用來自 Apache Spark 工RESTAPI作負載資料的互動式 Web 應用程式，使用 Apache Livy 作業。

必要條件

若要搭配EMR無伺服器使用 Apache Livy 端點，您必須符合下列需求：

- 完成開始使用 Amazon EMR 無伺服器中的步驟。
- 若要透過 Apache Livy 端點執行互動式工作負載，您需要特定的權限和角色。如需詳細資訊，請參閱互動式工作負載的必要權限。

所需的許可

除了存取EMR無伺服器的必要權限之外，您還必須將下列權限新增至您的IAM角色，才能存取 Apache Livy 端點並執行應用程式：

- `emr-serverless:AccessLivyEndpoints`— 授予訪問權限，並連接到您指定為啟用 LIVE 的應用程序。`Resource`您需要此權限才能執行可從 Apache Livy 端點執行可用的RESTAPI作業。
- `iam:PassRole`— 授予在建立 Apache Livy 工作階段時存取IAM執行角色的權限。EMR無伺服器將使用此角色來執行您的工作負載。
- `emr-serverless:GetDashboardForJobRun`— 授予產生 Spark Live UI 和驅動程式記錄連結的權限，並提供存取記錄，做為 Apache Livy 工作階段結果的一部分。

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Sid": "EMRServerlessInteractiveAccess",
            "Effect": "Allow",
```

```
            "Action": "emr-serverless:AccessLivyEndpoints",
            "Resource": "arn:aws:emr-serverless:<AWS_REGION>:account:/applications/*"
        },
        {
            "Sid": "EMRServerlessRuntimeRoleAccess",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "execution-role-ARN",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "emr-serverless.amazonaws.com"
                }
            }
        },
        {
            "Sid": "EMRServerlessDashboardAccess",
            "Effect": "Allow",
            "Action": "emr-serverless:GetDashboardForJobRun",
            "Resource": "arn:aws:emr-serverless:<AWS_REGION>:account:/applications/*"
        }
    ]
}
```

開始使用

1. 要創建一個 Apache 的 Livy 啟用的應用程序，運行以下命令。

```
 aws emr-serverless create-application \
--name my-application-name \
--type 'application-type' \
 --release-label <Amazon EMR-release-version>
--interactive-configuration '{"livyEndpointEnabled": true}'
```

2. EMR無伺服器建立應用程式後，啟動應用程式以使 Apache Livy 端點可用。

```
 aws emr-serverless start-application \
 --application-id application-id
```

使用下面的命令來檢查你的應用程序的狀態。一旦狀態變成STARTED，您就可以存取 Apache Livy 端點。

```
aws emr-serverless get-application \
```

```
--region <AWS_REGION> --application-id >application_id>
```

3. 使用以下URL指令存取端點：

```
https://_<application-id>_.livy.emr-serverless-
services._<AWS_REGION>_.amazonaws.com
```

端點準備就緒後，您可以根據您的使用案例提交工作負載。您必須使用SIGv4協議將每個請求簽署到端點，並傳入授權標頭。您可以使用下列方法來執行工作負載：

- HTTP用戶端 — 您必須使用自訂HTTP用戶端提交 Apache Livy 端點API作業。
- Sparkmagic 核心 — 您必須在本機執行 Sparkmagic 核心，並使用 Jupyter 筆記本提交互式查詢。

HTTP客戶端

若要建立 Apache Livy 工作階段，您必須`emr-serverless.session.executionRoleArn`在要求主體的`conf`參數中提交。下列範例是要`POST /sessions`求範例。

```
{
    "kind": "pyspark",
    "heartbeatTimeoutInSecond": 60,
    "conf": {
        "emr-serverless.session.executionRoleArn": "<executionRoleArn>"
    }
}
```

下表描述了所有可用的 Apache 利維API操作。

API操作	描述
GET/會話	傳回所有作用中互動式工作階段的清單。
POST/會話	通過火花或 pyspark 創建一個新的交互式會話。
GET/會議/ <*sessionId* >	返回會話信息。
GET/會議/ <*sessionId* >/ 州	返回會話的狀態。
DELETE/會議/ <*sessionId* >	停止並刪除工作階段。

API操作	描述
GET/會議/ <*sessionId* > /聲明	返回會話中的所有語句。
POST/會議/ <*sessionId* > /聲明	在工作階段中執行陳述式。
GET/會議/ <*sessionId* >/ 月結單 / <*statementId* >	返回會話中指定語句的詳細信息。
POST/會議/ <*sessionId* >/ 月結單 / <*statementId* >/ 取消	取消此會話中指定的語句。

將請求發送到阿帕奇利維端點

您也可以從HTTP用戶端將要求直接傳送到 Apache Livy 端點。這樣做可讓您在筆記型電腦以外的地方遠端執行使用案例的程式碼。

在開始向端點傳送要求之前，請確定您已安裝下列程式庫：

```
pip3 install botocore awscrt requests
```

以下是將HTTP請求直接發送到端點的示例 Python 腳本：

```
from botocore import crt
import requests
from botocore.awsrequest import AWSRequest
from botocore.credentials import Credentials
import botocore.session
import json, pprint, textwrap

endpoint = 'https://<application_id>.livy.emr-serverless-
services-<AWS_REGION>.amazonaws.com'
headers = {'Content-Type': 'application/json'}

session = botocore.session.Session()
signer = crt.auth.CrtS3SigV4Auth(session.get_credentials(), 'emr-serverless',
 '<AWS_REGION>')


### Create session request
```

```
data = {'kind': 'pyspark', 'heartbeatTimeoutInSecond': 60, 'conf': { 'emr-
serverless.session.executionRoleArn': 'arn:aws:iam::123456789012:role/role1'}}

request = AWSRequest(method='POST', url=endpoint + "/sessions", data=json.dumps(data),
 headers=headers)

request.context["payload_signing_enabled"] = False

signer.add_auth(request)

prepped = request.prepare()

r = requests.post(prepped.url, headers=prepped.headers, data=json.dumps(data))

pprint.pprint(r.json())


### List Sessions Request

request = AWSRequest(method='GET', url=endpoint + "/sessions", headers=headers)

request.context["payload_signing_enabled"] = False

signer.add_auth(request)

prepped = request.prepare()

r2 = requests.get(prepped.url, headers=prepped.headers)
pprint.pprint(r2.json())


### Get session state

session_url = endpoint + r.headers['location']

request = AWSRequest(method='GET', url=session_url, headers=headers)

request.context["payload_signing_enabled"] = False

signer.add_auth(request)

prepped = request.prepare()

r3 = requests.get(prepped.url, headers=prepped.headers)
```

```
pprint.pprint(r3.json())


### Submit Statement

data = {
      'code': "1 + 1"
}

statements_url = endpoint + r.headers['location'] + "/statements"

request = AWSRequest(method='POST', url=statements_url, data=json.dumps(data),
 headers=headers)

request.context["payload_signing_enabled"] = False

signer.add_auth(request)

prepped = request.prepare()

r4 = requests.post(prepped.url, headers=prepped.headers, data=json.dumps(data))

pprint.pprint(r4.json())

### Check statements results

specific_statement_url = endpoint + r4.headers['location']

request = AWSRequest(method='GET', url=specific_statement_url, headers=headers)

request.context["payload_signing_enabled"] = False

signer.add_auth(request)

prepped = request.prepare()

r5 = requests.get(prepped.url, headers=prepped.headers)

pprint.pprint(r5.json())


### Delete session
```

```
session_url = endpoint + r.headers['location']

request = AWSRequest(method='DELETE', url=session_url, headers=headers)

request.context["payload_signing_enabled"] = False

signer.add_auth(request)

prepped = request.prepare()

r6 = requests.delete(prepped.url, headers=prepped.headers)

pprint.pprint(r6.json())
```

火花核心

在安裝火花之前，請確保您已配置 AWS 您要在其中安裝 sparkmagic 的執行個體中的認證

1. 按照安裝步驟安裝火花。請注意，您只需要執行前四個步驟。
2. sparkmagic 核心支援自訂驗證器，因此您可以將驗證器與 sparkmagic 核心整合，以便簽署每個要求。SIGv4
3. 安裝EMR無伺服器自訂驗證器。

   ```
   pip install emr-serverless-customauth
   ```
4. 現在，在 Sparkmagic 設定 json 檔案URL中提供自訂驗證器和 Apache Livy 端點的路徑。使用以下命令打開配置文件。

   ```
   vim ~/.sparkmagic/config.json
   ```

   以下是範例`config.json`檔案。

   ```
   {
   "kernel_python_credentials" : {
       "username": "",
       "password": "",
       "url": "https://<application-id>.livy.emr-serverless-
   services.<AWS_REGION>.amazonaws.com",
       "auth": "Custom_Auth"
     },
   ```

```
  "kernel_scala_credentials" : {
    "username": "",
    "password": "",
    "url": "https://<application-id>.livy.emr-serverless-
services.<AWS_REGION>.amazonaws.com",
    "auth": "Custom_Auth"
  },
  "authenticators": {
    "None": "sparkmagic.auth.customauth.Authenticator",
    "Basic_Access": "sparkmagic.auth.basic.Basic",
    "Custom_Auth":
 "emr_serverless_customauth.customauthenticator.EMRServerlessCustomSigV4Signer"
  },
  "livy_session_startup_timeout_seconds": 600,
  "ignore_ssl_errors": false
}
```

5. 啟動木普特實驗室。它應該使用您在最後一步中設置的自定義身份驗證。
6. 然後，您可以運行以下筆記本命令和代碼以開始使用。

```
%%info //Returns the information about the current sessions.
```

```
%%configure -f //Configure information specific to a session. We supply
 executionRoleArn in this example. Change it for your use case.
{
    "driverMemory": "4g",
    "conf": {
          "emr-serverless.session.executionRoleArn":
 "arn:aws:iam::123456789012:role/JobExecutionRole"
    }
}
```

```
<your code>//Run your code to start the session
```

在內部，每個指令都會透過設定的 Apache Livy 端點呼叫每個 Apache Livy API 作業。URL然後，您可以根據您的用例編寫說明。

考量事項

透過 Apache Livy 端點執行互動式工作負載時，請考慮下列考量事項。

- EMR無伺服器會使用呼叫者主體來維護工作階段層級隔離。建立工作階段的呼叫者主體是唯一可以存取該工作階段的主要項目。如需更精細的隔離，您可以在採用認證時設定來源身分識別。在此情況下，EMRServerless 會根據呼叫者主體和來源識別強制執行工作階段層級隔離。如需來源身分識別的詳細資訊，請參閱監視和控制對假定角色採取的動作。
- EMR無伺服器版本 6.14.0 及更新版本支援 Apache Livy 端點。
- 阿帕奇利維端點僅支持 Apache 星火引擎。
- 阿帕奇生活端點支持斯卡拉星火和 PySpark。
- 依預設，會`autoStopConfig`在您的應用程式中啟用。這意味著應用程序在閒置 15 分鐘後關閉。您可以在`create-application`或`update-application`請求中變更此組態。
- 您可以在啟用 Apache Livy 端點的單一應用程式上執行多達 25 個並行工作階段。
- 為了獲得最佳的啟動體驗，我們建議您為驅動程式和執行程式設定預先初始化的容量。
- 您必須先手動啟動應用程式，才能連線到 Apache Livy 端點。
- 您必須擁有足夠的 v CPU 服務配額 AWS 帳戶 以使用 Apache Livy 端點執行互動式工作負載。我們建議至少 24 v CPU。
- 預設的 Apache 利維工作階段逾時為 1 小時。如果您沒有一小時執行陳述式，則 Apache Livy 會刪除工作階段並釋放驅動程式和執行程式。您無法變更此設定。
- 只有作用中的工作階段可以與 Apache Livy 端點互動。工作階段完成、取消或終止後，您將無法透過 Apache Livy 端點存取它。

日誌記錄和監控

監控是維護EMR無伺服器應用程式和工作的可靠性、可用性和效能的重要組成部分。您應該從EMR無伺服器解決方案的所有部分收集監控資料，以便在發生多點故障時更輕鬆地偵錯。

主題

儲存記錄

若要監控EMR無伺服器上的工作進度並對工作失敗進行疑難排解，您可以選擇EMR無伺服器儲存和提供應用程式記錄的方式。當您提交任務執行時，您可以指定受管儲存、Amazon S3 和 Amazon CloudWatch 做為記錄選項。

使用時 CloudWatch，您可以指定要使用的記錄類型和記錄位置，或接受預設類型和位置。如需 CloudWatch 記錄檔的詳細資訊，請參閱the section called “Amazon CloudWatch”。使用受管儲存和 S3 記錄，下表顯示如果您選擇受管儲存、Amazon S3 儲存貯體或兩者，可預期的日誌位置和 UI 可用性。

選項	事件記錄	容器日誌	應用程式 UI
受管理儲存	儲存在受管理的儲存	儲存在受管理的儲存	支援
受管儲存和 S3 儲存貯體	存儲在兩個地方	存放在 S3 儲存貯體	支援
Amazon S3 儲存貯體	存放在 S3 儲存貯體	存放在 S3 儲存貯體	不支援 [1]

[1] 我們建議您保持選取 [受管理的儲存] 選項。否則，您將無法使用內置應用程序UIs。

使用受管理儲存進行EMR無伺服器記錄

根據預設，EMR無伺服器會將應用程式日誌安全地存放在 Amazon EMR 受管儲存中，最多可保留 30 天。

> **Note**
>
> 如果您關閉預設選項，Amazon 將EMR無法代表您對任務進行疑難排解。

若要從 EMR Studio 關閉此選項，請取消選取「允許」 AWS 在 [送出工作] 頁面的 [其他設定] 區段中，保留記錄檔 30 天核取方塊。

若要關閉此選項 AWS CLI，當您提交工作執行時，請使用`managedPersistenceMonitoringConfiguration`組態。

```
{
    "monitoringConfiguration": {
        "managedPersistenceMonitoringConfiguration": {
            "enabled": false
        }
    }
}
```

使用 Amazon S3 儲存貯體進行EMR無伺服器記錄

您必須在任務執行階段角色的許可政策中包含下列許可，才能將日誌資料傳送到 Amazon S3。*DOC-EXAMPLE-BUCKET-LOGGING*以記錄值區的名稱取代。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET-LOGGING/*"
            ]
        }
```

```
    ]
}
```

若要設定 Amazon S3 儲存貯體以存放來自 AWS CLI，當您開始工作執行時，請使用`s3MonitoringConfiguration`組態。若要這麼做，請在組態`--configuration-overrides`中提供下列資訊。

```
{
    "monitoringConfiguration": {
        "s3MonitoringConfiguration": {
            "logUri": "s3://DOC-EXAMPLE-BUCKET-LOGGING/logs/"
        }
    }
}
```

對於未啟用重試的批次工作，EMR無伺服器會將記錄檔傳送至下列路徑：

```
'/applications/<applicationId>/jobs/<jobId>'
```

EMR無伺服器版本 7.1.0 及更高版本支援串流工作和批次工作的重試嘗試。如果您在啟用重試的情況下執行工作，EMRServerless 會自動將嘗試編號新增至記錄檔路徑前置詞，以便您更好地區分和追蹤記錄。

```
'/applications/<applicationId>/jobs/<jobId>/attempts/<attemptNumber>/'
```

使用 Amazon 進行EMR無伺服器記錄 CloudWatch

將任務提交至EMR無伺服器應用程式時，您可以選擇 Amazon 作 CloudWatch為存放應用程式日誌的選項。這可讓您使用日 CloudWatch 誌分析功能，例如 CloudWatch 日誌洞察和即時尾巴。您也可以將日誌從流式傳輸 CloudWatch 到其他系統， OpenSearch 例如進一步分析。

EMR無伺服器提供驅動程式記錄的即時記錄。您可以使用實時尾部功能或通過 CloudWatch CLI尾部命令 CloudWatch 實時查看日誌。

EMR無伺服器的 CloudWatch 記錄預設為停用。若要啟用它，請參閱中的配置AWS CLI。

> **Note**
>
> Amazon 會即時 CloudWatch 發佈日誌，因此會產生更多員工的資源。如果您選擇較低的背景工作者容量，對工作執行時間的影響可能會增加。如果您啟用 CloudWatch 記錄功能，建

議您選擇較大的背景工作容量。如果每秒的交易 (TPS) 速率太低，則日誌發布也可能會限制。PutLogEvents CloudWatch 節流組態適用於所有服務 (包括EMR無伺服器)。如需詳細資訊，請參閱如何判斷記錄中的 CloudWatch 節流？ 上 AWS 回復：帖子。

使用記錄所需的權限 CloudWatch

在您的任務可以傳送日誌資料到 Amazon 之前 CloudWatch，您必須在任務執行階段角色的許可政策中包含下列許可。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups"
            ],
            "Resource": [
                "arn:aws:logs:AWS ##:111122223333:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
               "logs:PutLogEvents",
               "logs:CreateLogGroup",
               "logs:CreateLogStream",
               "logs:DescribeLogStreams"
            ],
            "Resource": [
               "arn:aws:logs:AWS ##:111122223333:log-group:my-log-group-name:*"
            ]
        }
    ]
}
```

AWS CLI

若要將 Amazon 設定 CloudWatch 為存放EMR無伺服器的日誌，請從 AWS CLI，當您開始工作執行時，請使用cloudWatchLoggingConfiguration組態。若要這麼做，請提供下列組態覆寫。您也可以選擇性地提供記錄群組名稱、記錄資料流前置詞名稱、記錄類型和加密金鑰ARN。

如果您未指定選擇性值，則會使用預設記錄資料流將記錄 CloudWatch 發佈至預設記錄群組/aws/emr-serverless/applications/*applicationId*/jobs/*jobId*/*worker-type*。

EMR無伺服器版本 7.1.0 及更高版本支援串流工作和批次工作的重試嘗試。如果您啟用了工作的重試次數，EMRServerless 會自動將嘗試編號新增至記錄路徑前置詞，以便您更好地區分和追蹤記錄。

```
'/applications/<applicationId>/jobs/<jobId>/attempts/<attemptNumber>/worker-type'
```

以下顯示使用EMR無伺服器預設設定開啟 Amazon 日 CloudWatch 誌記錄所需的最低組態：

```
{
    "monitoringConfiguration": {
        "cloudWatchLoggingConfiguration": {
            "enabled": true
         }
     }
}
```

下列範例顯示開啟EMR無伺服器的 Amazon CloudWatch 日誌記錄時，您可以指定的所有必要和選用組態。支援的logTypes值也列在此範例下方。

```
{
    "monitoringConfiguration": {
        "cloudWatchLoggingConfiguration": {
            "enabled": true, // Required
            "logGroupName": "Example_logGroup", // Optional
            "logStreamNamePrefix": "Example_logStream", // Optional
            "encryptionKeyArn": "key-arn", // Optional
            "logTypes": {
                "SPARK_DRIVER": ["stdout", "stderr"] //List of values
             }
         }
     }
}
```

根據預設，EMR無伺服器只會將驅動程式標準輸出和 stderr 記錄發佈到。 CloudWatch如果您需要其他記錄檔，則可以使用logTypes欄位指定容器角色和對應的記錄類型。

下列清單顯示您可為logTypes組態指定的支援 Worker 類型：

Spark

- `SPARK_DRIVER : ["STDERR", "STDOUT"]`
- `SPARK_EXECUTOR : ["STDERR", "STDOUT"]`

Hive

- `HIVE_DRIVER : ["STDERR", "STDOUT", "HIVE_LOG", "TEZ_AM"]`
- `TEZ_TASK : ["STDERR", "STDOUT", "SYSTEM_LOGS"]`

旋轉日誌

Amazon EMR 無伺服器可輪替 Spark 應用程式日誌和事件日誌。記錄輪替有助於解決長時間執行的工作產生大型記錄檔 (可能會佔用所有磁碟空間) 的問題。輪換記錄可協助您節省磁碟儲存空間，並減少工作失敗的數量，因為您的磁碟上沒有剩餘的空間。

記錄輪換預設為啟用，且僅適用於 Spark 工作。

火花事件日誌

> **Note**
>
> Spark 事件日誌輪替適用於所有 Amazon EMR 發行標籤。

EMRServerless 不會產生單一事件記錄檔，而是以固定的時間間隔輪換事件記錄檔，並移除較舊的事件記錄檔。輪換日誌不會影響上傳到 S3 存儲桶的日誌。

火花應用程序日

> **Note**
>
> Spark 應用程式日誌輪替適用於所有 Amazon EMR 版本標籤。

EMR無伺服器也會輪換驅動程式和執行程式 (例如和檔案) 的 spark 應用程式記錄。`stdout` `stderr`您可以通過使用 Spark 歷史記錄服務器和 Live UI 鏈接選擇 Studio 中的日誌鏈接訪問最新的日誌文件。記錄檔是最新記錄檔的截斷版本。若要查看較舊的旋轉日誌，您必須在存放日誌時指定 Amazon S3 位置。如需詳細資訊，請參閱使用 Amazon S3 儲存貯體進行EMR無伺服器記錄。

您可以在下列位置找到最新的記錄檔。EMR無伺服器會每 15 秒重新整理一次檔案。這些檔案的範圍可以介於 0 MB 到 128 MB 之間。

```
<example-S3-logUri>/applications/<application-id>/jobs/<job-id>/SPARK_DRIVER/stderr.gz
```

以下位置包含較舊的旋轉檔案。每個檔案大小為 128 MB。

```
<example-S3-logUri>/applications/<application-id>/jobs/<job-id>/SPARK_DRIVER/archived/
stderr_<index>.gz
```

同樣的行為也適用於 Spark 執行程序。此變更僅適用於 S3 記錄。日誌輪換不會對上傳到 Amazon 的日誌流進行任何更改 CloudWatch。

EMR無伺服器版本 7.1.0 及更高版本支援串流和批次工作的重試嘗試。如果您對工作啟用了重試嘗試，EMRServerless 會在此類工作的記錄路徑中新增前置詞，以便您可以更好地追蹤和區分記錄。此路徑包含所有已旋轉的記錄檔。

```
'/applications/<applicationId>/jobs/<jobId>/attempts/<attemptNumber>/'.
```

加密記錄

使用受管儲存加密EMR無伺服器記錄

若要使用您自己的KMS金鑰加密受管理儲存區中的記錄檔，請在提交作業執行時使用`managedPersistenceMonitoringConfiguration`組態。

```
{
    "monitoringConfiguration": {
        "managedPersistenceMonitoringConfiguration" : {
            "encryptionKeyArn": "key-arn"
        }
    }
}
```

使用 Amazon S3 儲存EMR貯體加密無伺服器日誌

若要使用自己的KMS金鑰加密 Amazon S3 儲存貯體中的日誌，請在提交任務執行時使用`s3MonitoringConfiguration`組態。

```
{
    "monitoringConfiguration": {
        "s3MonitoringConfiguration": {
            "logUri": "s3://DOC-EXAMPLE-BUCKET-LOGGING/logs/",
            "encryptionKeyArn": "key-arn"
        }
    }
}
```

使用 Amazon 加密EMR無伺服器日誌 CloudWatch

若要使用您自己 CloudWatch 的KMS金鑰在 Amazon 中加密日誌，請在提交任務執行時使用`cloudWatchLoggingConfiguration`組態。

```
{
    "monitoringConfiguration": {
        "cloudWatchLoggingConfiguration": {
            "enabled": true,
            "encryptionKeyArn": "key-arn"
         }
     }
}
```

記錄檔加密所需的權限

本節內容

- 所需使用者權限
- Amazon S3 和受管儲存的加密金鑰許可
- Amazon 的加密金鑰許可 CloudWatch

所需使用者權限

提交工作或檢視記錄檔或應用程式的使用者UIs必須具有使用金鑰的權限。您可以在KMS金鑰原則或使用者、群組或角色的IAM原則中指定權限。如果提交工作的使用者缺少KMS金鑰權限，則EMR無伺服器會拒絕工作執行提交。

金鑰原則範例

下列金鑰原則提供`kms:GenerateDataKey`和的權限`kms:Decrypt`：

```
{
    "Effect": "Allow",
    "Principal":{
       "AWS": "arn:aws:iam::111122223333:user/user-name"
     },
     "Action": [
       "kms:GenerateDataKey",
       "kms:Decrypt"
      ],
     "Resource": "*"
 }
```

範例IAM政策

下列IAM原則提供kms:GenerateDataKey和的權限kms:Decrypt：

```
{
 "Version": "2012-10-17",
  "Statement": {
     "Effect": "Allow",
     "Action": [
           "kms:GenerateDataKey",
           "kms:Decrypt"
      ],
     "Resource": "key-arn"
   }
}
```

若要啟動 Spark 或 Tez UI，您必須授與使用者、群組或角色存取權限，emr-serverless:GetDashboardForJobRunAPI如下所示：

```
{
 "Version": "2012-10-17",
  "Statement": {
     "Effect": "Allow",
     "Action": [
           "emr-serverless:GetDashboardForJobRun"
      ]
   }
}
```

Amazon S3 和受管儲存的加密金鑰許可

當您在受管儲存或 S3 儲存貯體中使用自己的加密金鑰加密日誌時，您必須按照下列方式設定KMS金鑰許可。

`emr-serverless.amazonaws.com`主參與者在KMS金鑰的原則中必須具有下列權限：

```
{
    "Effect": "Allow",
    "Principal":{
       "Service": "emr-serverless.amazonaws.com"
     },
     "Action": [
       "kms:Decrypt",
       "kms:GenerateDataKey"
      ],
     "Resource": "*"
     "Condition": {
       "StringLike": {
         "aws:SourceArn": "arn:aws:emr-serverless:region:aws-account-id:/
applications/application-id"
       }
     }
 }
```

我們建議您在KMS金鑰原則中新增`aws:SourceArn`條件金鑰，做為安全性最佳作法。IAM全域條件金鑰`aws:SourceArn`有助於確保EMR無伺服器僅將KMS金鑰用於應用程ARN式。

工作執行階段角色在其IAM原則中必須具有下列權限：

```
{
 "Version": "2012-10-17",
  "Statement": {
     "Effect": "Allow",
     "Action": [
           "kms:GenerateDataKey",
           "kms:Decrypt"
      ],
     "Resource": "key-arn"
   }
}
```

Amazon 的加密金鑰許可 CloudWatch

若要將KMS金鑰與您ARN的記錄群組產生關聯，請針對工作執行階段角色使用下列IAM原則。

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "logs:AssociateKmsKey"
        ],
        "Resource": [
            "arn:aws:logs:AWS ##:111122223333:log-group:my-log-group-name:*"
        ]
    }
}
```

設定KMS金鑰政策以授予KMS權限給 Amazon CloudWatch：

```
{
    "Version": "2012-10-17",
    "Id": "key-default-1",
    "Statement":
     {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.AWS ##.amazonaws.com"
        },
        "Action": [
            "kms:Decrypt",
            "kms:GenerateDataKey",
        ],
        "Resource": "*",
        "Condition": {
            "ArnLike": {
                "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:AWS #
#:111122223333:*"
            }
        }
     }
}
```

為 Amazon 無伺服器設定阿帕奇 Log4j2 屬性 EMR

此頁面說明如何在以下位置設定EMR無伺服器工作的自訂 Apache Log4j 2.x 特性。StartJobRun如果要在應用程式層次設定 Log4j 分類，請參閱EMR無伺服器的預設應用程式組態。

為 Amazon 無伺服器設定星火 Log4j2 屬性 EMR

使用 Amazon 6.8.0 及更高EMR版本，您可以自訂 Apache Log4j 2.x 屬性，以指定精細的日誌組態。如此可簡化EMR無伺服器上 Spark 工作的疑難排解作業。若要配置這些屬性，請使用spark-driver-log4j2和spark-executor-log4j2分類。

主題

- 星火的 Log4j2 分類
- 星火的 Log4j2 配置示例
- Log4j2 在樣品星火作業
- 星火的 Log4j2 注意事項

星火的 Log4j2 分類

若要自訂 Spark 記錄組態，請搭配使用下列分類applicationConfiguration。若要設定 Log4j 2.x 屬性，請使用下列指令。properties

spark-driver-log4j2

此分類會設定log4j2.properties檔案中驅動程式的值。

spark-executor-log4j2

此分類設定執行程式log4j2.properties檔案中的值。

星火的 Log4j2 配置示例

下面的例子演示了如何提交一個星火作業與自定義applicationConfiguration的星火驅動程序和執行程序的 log4j2 配置。

若要在應用程式層次設定 Log4j 分類，而不是在送出工作時設定，請參閱EMR無伺服器的預設應用程式組態。

```
aws emr-serverless start-job-run \
    --application-id application-id \
    --execution-role-arn job-role-arn \
    --job-driver '{
        "sparkSubmit": {
            "entryPoint": "/usr/lib/spark/examples/jars/spark-examples.jar",
            "entryPointArguments": ["1"],
            "sparkSubmitParameters": "--class org.apache.spark.examples.SparkPi --conf
 spark.executor.cores=4 --conf spark.executor.memory=20g --conf spark.driver.cores=4 --
conf spark.driver.memory=8g --conf spark.executor.instances=1"
        }
    }'
    --configuration-overrides '{
        "applicationConfiguration": [
             {
                "classification": "spark-driver-log4j2",
                "properties": {
                    "rootLogger.level":"error", // will only display Spark error logs
                    "logger.IdentifierForClass.name": "classpath for setting logger",
                    "logger.IdentifierForClass.level": "info"

                }
            },
            {
                "classification": "spark-executor-log4j2",
                "properties": {
                    "rootLogger.level":"error", // will only display Spark error logs
                    "logger.IdentifierForClass.name": "classpath for setting logger",
                    "logger.IdentifierForClass.level": "info"
                }
            }
       ]
    }'
```

Log4j2 在樣品星火作業

下列程式碼範例示範如何在初始化應用程式的自訂 Log4j2 組態時建立 Spark 應用程式。

Python

Example -使用 Log4j2 進行火花作業與 Python

```
import os
```

```
import sys

from pyspark import SparkConf, SparkContext
from pyspark.sql import SparkSession

app_name = "PySparkApp"
if __name__ == "__main__":
    spark = SparkSession\
        .builder\
        .appName(app_name)\
        .getOrCreate()

    spark.sparkContext._conf.getAll()
    sc = spark.sparkContext
    log4jLogger = sc._jvm.org.apache.log4j
    LOGGER = log4jLogger.LogManager.getLogger(app_name)

    LOGGER.info("pyspark script logger info")
    LOGGER.warn("pyspark script logger warn")
    LOGGER.error("pyspark script logger error")

    // your code here

    spark.stop()
```

要在執行 Spark 作業時為驅動程序自定義 log4j2，可以使用以下配置：

```
{
   "classification": "spark-driver-log4j2",
      "properties": {
          "rootLogger.level":"error", // only display Spark error logs
          "logger.PySparkApp.level": "info",
          "logger.PySparkApp.name": "PySparkApp"
      }
}
```

Scala

Example -使用 Log4j2 進行斯卡拉的火花作業

```
import org.apache.log4j.Logger
import org.apache.spark.sql.SparkSession
```

```
object ExampleClass {
  def main(args: Array[String]): Unit = {
    val spark = SparkSession
    .builder
    .appName(this.getClass.getName)
    .getOrCreate()

    val logger = Logger.getLogger(this.getClass);
    logger.info("script logging info logs")
    logger.warn("script logging warn logs")
    logger.error("script logging error logs")

// your code here
    spark.stop()
  }
}
```

要在執行 Spark 作業時為驅動程序自定義 log4j2，可以使用以下配置：

```
{
   "classification": "spark-driver-log4j2",
      "properties": {
          "rootLogger.level":"error", // only display Spark error logs
          "logger.ExampleClass.level": "info",
          "logger.ExampleClass.name": "ExampleClass"
      }
}
```

星火的 Log4j2 注意事項

以下 Log4j2.x 屬性不適用於星火進程配置：

- `rootLogger.appenderRef.stdout.ref`
- `appender.console.type`
- `appender.console.name`
- `appender.console.target`
- `appender.console.layout.type`
- `appender.console.layout.pattern`

如需有關可設定之 Log4j2.x 屬性的詳細資訊，請參閱中的檔案。`log4j2.properties.template` GitHub

監控EMR無伺服器

本節介紹監控 Amazon EMR 無伺服器應用程式和任務的方式。

主題

- 監控EMR無伺服器應用程式和工作
- 使用適用於 Prometheus 的 Amazon 託管服務監控 Spark 指標
- EMR無伺服器使用量指標

監控EMR無伺服器應用程式和工作

使用適用於EMR無伺服器的 Amazon CloudWatch 指標，您可以接收 1 分鐘 CloudWatch 指標並存取 CloudWatch 儀表板以檢視EMR無伺服器應用程式的 near-real-time操作和效能。

EMR無伺服器會 CloudWatch 每分鐘傳送指標。EMR無伺服器會在應用程式層級以及工作、工作者類型和層級發出這些指標。 capacity-allocation-type

若要開始使用，請使用EMR無伺服器 GitHub 儲存庫中提供的EMR無伺服器 CloudWatch 儀表板範本並進行部署。

> **Note**
>
> EMR無伺服器互動式工作負載僅啟用應用程式層級監控，並具有新的 Worker 類型維度。`Spark_Kernel`若要監視和偵錯您的互動式工作負載，您可以從 EMRStudio 工作區內檢視記錄檔和 Apache Spark 使用者介面。

下表說明`AWS/EMRServerless`命名空間中可用的EMR無伺服器維度。

EMR無伺服器量度的維度

維度	描述	
`ApplicationId`	篩選EMR無伺服器應用程式的所有指標。	

維度	描述	
JobId	篩選EMR無伺服器工作執行的所有指標。	
WorkerType	篩選指定 Worker 類型的所有量度。例如，您可以針對 Spark 工作進SPARK_EXECUTORS 行篩選SPARK_DRIVER 和篩選。	
CapacityAllocationType	篩選指定容量配置類型的所有度量。例如，您可以篩選預先初始化的PreInitCapacity 容量以及OnDemandCapacity 其他所有項目。	

應用程式級監控

您可以使用 Amazon CloudWatch 指標在EMR無伺服器應用程式層級監控容量使用情況。您也可以設定單一檢視，以便在 CloudWatch 儀表板中監視應用程式容量使用量。

EMR無伺服器應用程式量

指標	描述	主要維度	次要維度
CPUAllocated	vCPUs 分配的總數。	ApplicationId	ApplicationId，WorkerType，CapacityAllocationType
IdleWorkerCount	閒置的工作人員總數。	ApplicationId	ApplicationId，WorkerType，CapacityAllocationType
MaxCPUAllowed	應用程式CPU允許的最大值。	ApplicationId	N/A

指標	描述	主要維度	次要維度
MaxMemoryAllowed	應用程式允許的最大記憶體 (以 GB 為單位)。	ApplicationId	N/A
MaxStorageAllowed	應用程式允許的最大儲存空間 (以 GB 為單位)。	ApplicationId	N/A
MemoryAllocated	配置的總記憶體 (以 GB 為單位)。	ApplicationId	ApplicationId ,WorkerType ,CapacityAllocationType
PendingCreationWorkerCount	擱置建立的工作程式總數。	ApplicationId	ApplicationId ,WorkerType ,CapacityAllocationType
RunningWorkerCount	應用程式正在使用的工作者總數。	ApplicationId	ApplicationId ,WorkerType ,CapacityAllocationType
StorageAllocated	配置的總磁碟儲存體 (以 GB 為單位)。	ApplicationId	ApplicationId ,WorkerType ,CapacityAllocationType
TotalWorkerCount	可用的工作人員總數。	ApplicationId	ApplicationId ,WorkerType ,CapacityAllocationType

工作級別監控

Amazon EMR 無伺服器會將下列工作層級指標傳送至 Amazon CloudWatch 每隔一分鐘 您可以依工作執行狀態檢視彙總工作執行的測量結果值。每個量度的單位都是計數。

EMR無伺服器工作層級指標

指標	描述	主要維度
SubmittedJobs	處於「已提交」狀態的工作數目。	ApplicationId
PendingJobs	處於「擱置中」狀態的工作數目。	ApplicationId
ScheduledJobs	處於「已排程」狀態的工作數目。	ApplicationId
RunningJobs	處於「執行中」狀態的工作數目。	ApplicationId
SuccessJobs	處於「成功」狀態的工作數目。	ApplicationId
FailedJobs	處於「失敗」狀態的工作數目。	ApplicationId
CancellingJobs	處於 [取消] 狀態的工作數目。	ApplicationId
CancelledJobs	處於「已取消」狀態的工作數目。	ApplicationId

您可以使用引擎特定應用程式監視執行中和已完成的EMR無伺服器工作的引擎特定指標。UIs當您檢視執行中工作的 UI 時，您會看到包含即時更新的即時應用程式 UI。當您檢視已完成工作的 UI 時，您會看到永久性應用程式 UI。

執行任務

對於執行中的EMR無伺服器工作，您可以檢視提供引擎特定指標的即時介面。您可以使用 Apache 星火使用者介面或蜂巢特茲使用者介面來監視和偵錯您的工作。要訪問這些內容UIs，請使用 EMR Studio 控制台或請求安全URL端點 AWS Command Line Interface.

已完成工作

對於已完成的EMR無伺服器工作，您可以使用 Spark 歷史記錄伺服器或永久性 Hive Tez UI 來檢視 Spark 或 Hive 作業執行的作業詳細資料、階段、工作和指標。若要存取這些UIs項目，請使用 EMR Studio 主控台，或使URL用 AWS Command Line Interface.

Job 人員層級監控

Amazon EMR 無伺服器會將AWS/EMRServerless命名空間和Job Worker Metrics指標群組中可用的下列任務工作者層級指標傳送給 Amazon CloudWatch。EMR無伺服器會在工作層級、工作者類型和層級執行期間，從個別工作者收集資料點。 capacity-allocation-type 您可以用ApplicationId作維度來監視屬於同一個應用程式的多個工作。

EMR無伺服器工作者層級指標

指標	描述	單位	主要維度	次要維度
WorkerCpuAllocated	工作執行中分配給工作者的 vCPU 核心總數。	無	JobId	ApplicationId 、WorkerType 與 CapacityAllocationType
WorkerCpuUsed	工作執行中工作者使用的 v CPU 核心總數。	無	JobId	ApplicationId 、WorkerType 與 CapacityAllocationType
WorkerMemoryAllocated	工作執行中配置給 Worker 的總記憶體 (以 GB 為單位)。	千兆字節	JobId	ApplicationId 、WorkerType 與 CapacityA

指標	描述	單位	主要維度	次要維度
				llocation Type
WorkerMemoryUsed	工作執行中工作者使用的總記憶體 (以 GB 為單位)。	千兆字節	JobId	ApplicationId 、WorkerType 與 CapacityAllocation Type
WorkerEphemeralStorageAllocated	工作執行中配置給 Worker 的暫時儲存體位元組數。	千兆字節	JobId	ApplicationId 、WorkerType 與 CapacityAllocation Type
WorkerEphemeralStorageUsed	工作執行中 Worker 使用的暫時儲存體位元組數目。	千兆字節	JobId	ApplicationId 、WorkerType 與 CapacityAllocation Type
WorkerStorageReadBytes	工作執行中 Worker 從儲存區讀取的位元組數目。	位元組	JobId	ApplicationId 、WorkerType 與 CapacityAllocation Type

指標	描述	單位	主要維度	次要維度
WorkerStorageWriteBytes	工作執行中從 Worker 寫入儲存區的位元組數。	位元組	JobId	ApplicationId 、WorkerType 與 CapacityAllocationType

下列步驟說明如何檢視各種量度類型。

Console

使用主控台存取您的應用程式 UI

1. 按照從主控台開始使EMR用中的指示，在 EMR Studio 上導覽至您的無伺服器應用程式。
2. 若要檢視引擎特定的應用程式UIs和執行中工作的記錄：
   a. 選擇具有RUNNING狀態的工作。
   b. 在「應用程式詳細資訊」頁面上選取 Job，或瀏覽工作的「工作詳細資訊」頁面。
   c. 在 [顯示 UI] 下拉式功能表下，選擇 [Spark UI] 或 [Hive Tez UI]，瀏覽至您工作類型的應用程式 UI。
   d. 若要檢視 Spark 引擎記錄檔，請瀏覽至 Spark UI 中的 [執行程式] 索引標籤，然後選擇驅動程式的 [記錄] 連結。要查看蜂巢引擎日誌，選擇在蜂巢 Tez UI DAG 中適當的日誌鏈接。
3. 若要檢視已完成工作的引擎特定應用程式UIs和記錄：
   a. 選擇具有SUCCESS狀態的工作。
   b. 在應用程式的 [應用程式詳細資訊] 頁面上選取 Job，或瀏覽至工作的 [工作詳細資訊] 頁面
   c. 在 [顯示使用者介面] 下拉式功能表下，選擇 [Spark 歷程記錄伺服器] 或 [持續性 Hive Tez UI]，瀏覽至工作類型的應用程式 UI。
   d. 若要檢視 Spark 引擎記錄檔，請瀏覽至 Spark UI 中的 [執行程式] 索引標籤，然後選擇驅動程式的 [記錄] 連結。要查看蜂巢引擎日誌，選擇在蜂巢 Tez UI DAG 中適當的日誌鏈接。

AWS CLI

若要存取您的應用程式 UI AWS CLI

- 若要產生URL可用來存取執行中和已完成工作的應用程式 UI，請呼叫 `GetDashboardForJobRunAPI`.

```
aws emr-serverless get-dashboard-for-job-run /
--application-id <application-id> /
--job-run-id <job-id>
```

您產生的URL有效期為一小時。

使用適用於 Prometheus 的 Amazon 託管服務監控 Spark 指標

使用 Amazon 7.1.0 及更高EMR版本，您可以將EMR無伺服器與適用於 Prometheus 的 Amazon 受管服務整合，以收集無伺服器任務和應用程式的 Apache Spark 指標。EMR當您提交工作或建立應用程式時，可使用此整合 AWS 主控台、EMR無伺服API器或 AWS CLI.

必要條件

您必須完成下列先決條件，才能將 Spark 指標交付給適用於 Prometheus 的 Amazon 受管服務。

- 為 Prometheus 工作區建立 Amazon 受管服務。此工作區用作擷取端點。記下「端點-遠端寫入」的URL顯示URL。您需要指定建立EMR無伺服器應用程式的URL時間。
- 若要將任務存取權授予 Prometheus 用於監控用途的 Amazon 受管服務，請將以下政策新增至您的任務執行角色。

```
{
    "Sid": "AccessToPrometheus",
    "Effect": "Allow",
    "Action": ["aps:RemoteWrite"],
    "Resource": "arn:aws:aps:<AWS_REGION>:<AWS_ACCOUNT_ID>:workspace/<WORKSPACE_ID>"
}
```

設定

若要使用 AWS 用於創建與 Prometheus 的 Amazon 託管服務集成的應用程序的控制台

1. 請參閱開始使用 Amazon EMR 無伺服器以建立應用程式。
2. 建立應用程式時，請選擇 [使用自訂設定]，然後在您要設定的欄位中指定資訊以設定應用程式。
3. 在「應用程式日誌和指標」下，選擇「為 Prometheus 傳遞引擎指標至 Amazon 受管服務」，然後指定您的遠端寫入。URL
4. 指定您想要的任何其他組態設定，然後選擇 [建立並啟動應用程式]。

使用 AWS CLI 或EMR無伺服器 API

您也可以使用 AWS CLI 或EMR無伺API服器，在您執行或指令時，將您的EMR無伺服器應用程式與 Prometheus 的 Amazon 受管服務整合。`create-application` `start-job-run`

create-application

```
aws emr-serverless create-application \
--release-label emr-7.1.0 \
--type "SPARK" \
--monitoring-configuration '{
    "prometheusMonitoringConfiguration": {
        "remoteWriteUrl": "https://aps-workspaces.<AWS_REGION>.amazonaws.com/
workspaces/<WORKSPACE_ID>/api/v1/remote_write"
    }
}'
```

start-job-run

```
aws emr-serverless start-job-run \
--application-id <APPPLICATION_ID> \
--execution-role-arn <JOB_EXECUTION_ROLE> \
--job-driver '{
    "sparkSubmit": {
        "entryPoint": "local:///usr/lib/spark/examples/src/main/python/pi.py",
        "entryPointArguments": ["10000"],
        "sparkSubmitParameters": "--conf spark.dynamicAllocation.maxExecutors=10"
    }
}' \
--configuration-overrides '{
```

```
     "monitoringConfiguration": {
        "prometheusMonitoringConfiguration": {
            "remoteWriteUrl": "https://aps-workspaces.<AWS_REGION>.amazonaws.com/
workspaces/<WORKSPACE_ID>/api/v1/remote_write"
        }
    }
}'
```

在命令prometheusMonitoringConfiguration中包含表示EMR無伺服器必須使用代理程式來執行 Spark 任務，該代理程式會收集 Spark 指標，並將其寫入 Prometheus 的 Amazon 受管服務的remoteWriteUrl端點。然後，您可以使用適用於 Prometheus 的 Amazon 受管服務中的 Spark 指標進行視覺化、警示和分析。

進階組態屬性

EMR無伺服器使用 Spark 中名為的元件PrometheusServlet來收集 Spark 指標，並將效能資料轉換為與 Prometheus 的 Amazon 受管服務相容的資料。根據預設，EMR無伺服器會在 Spark 中設定預設值，並在您使用提交工作時剖析驅動程式和執行程式度量。PrometheusMonitoringConfiguration

下表說明您在提交 Spark 任務時可以設定的所有屬性，該任務會將指標傳送至 Prometheus 的 Amazon 受管服務。

星火屬性	預設值	描述
spark.metrics.conf.*.sink.prometheusServlet.class	或. 阿帕奇. 火花. 度量. PrometheusServlet	Spark 用來將指標傳送至 Amazon Prometheus 受管服務的類別。若要覆寫預設行為，請指定您自己的自訂類別。
spark.metrics.conf.*.source.jvm.class	或者. 火花. 度量. 來源. JvmSource	Spark 用來從基礎 Java 虛擬機器收集和傳送關鍵指標的類別。若要停止收集JVM測量結果，請將此屬性設定為空字串來停用，例如""。若要覆寫預設行為，請指定您自己的自訂類別。

星火屬性	預設值	描述
spark.metrics.conf.driver.sink.prometheusServlet.path	/度量/普罗米修斯	Amazon Prometheus 託管服務用於從驅動程序收集指標的不同URL之處。若要覆寫預設行為，請指定您自己的路徑。若要停止收集驅動程式測量結果，請將此屬性設定為空字串來停用，例如""。
spark.metrics.conf.executor.sink.prometheusServlet.path	/度量/執行者/Prometheus	Amazon Prometheus 託管服務用於從執行人URL那裡收集指標的不同之處。若要覆寫預設行為，請指定您自己的路徑。若要停止收集執行程式測量結果，請將此屬性設定為空字串來停用，例如""。

如需有關星火指標的詳細資訊，請參閱 Apache 星火指標。

考量與限制

使用適用於 Prometheus 的 Amazon 受管服務從EMR無伺服器收集指標時，請考慮下列考量事項和限制。

- Support 對 Prometheus 搭配EMR無伺服器使用 Amazon 受管服務的支援僅適用於 AWS 區域 Prometheus 的 Amazon 託管服務正式提供。
- 在適用於 Prometheus 的 Amazon 受管服務上執行代理程式以收集 Spark 指標，需要工作者提供更多資源。如果您選擇較小的背景工作大小 (例如一個 v CPU worker)，您的工作執行時間可能會增加。
- 只有 Amazon 7.1.0 及更高EMR版本才能 Support 使用 Prometheus 搭配EMR無伺服器使用 Amazon 受管服務。

EMR無伺服器使用量指標

您可以使用 Amazon 使 CloudWatch 用量指標來提供您帳戶使用的資源的可見性。使用這些指標在 CloudWatch 圖形和儀表板上視覺化您的服務使用情況。

EMR無伺服器使用量度對應至「Service Quotas」。您可以設定警示，在您的用量接近服務配額時發出警示。如需詳細資訊，請參閱 Ser vice Quotas 使用指南中的 Service Quotas 和 Amazon CloudWatch 警示。

如需EMR無伺服器服務配額的詳細資訊，請參閱的端點和配額 EMR Serverless。

EMR無伺服器的服務配額使用量度

EMR無伺服器會在AWS/Usage命名空間中發佈下列服務配額使用量測量結果。

指標	描述
ResourceCount	您帳號上執行的指定資源總數。資源由與測量結果相關聯的維度定義。

EMR無伺服器服務配額使用量度的維度

您可以使用下列維度來調整EMR無伺服器發佈的使用量度。

維度	Value	描述
Service	EMR無伺服器	的名稱 AWS 服務 包含資源。
Type	資源	EMR無伺服器所報告的實體類型。
Resource	v CPU	EMR無伺服器追蹤的資源類型。
Class	無	EMR無伺服器追蹤的資源類別。

使用自動化EMR無伺服器 Amazon EventBridge

您可以使用... Amazon EventBridge 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。 EventBridge 提供近乎即時的系統事件串流，用來描述您的系統變更 AWS 的費用。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。使用EventBridge，您可以自動：

- 調用 AWS Lambda 函數
- 將事件轉送至 Amazon Kinesis Data Streams
- 啟動一個 AWS Step Functions 狀態機器
- 通知 Amazon SNS 主題或 Amazon SQS 隊列

例如，當您 EventBridge 搭配EMR無伺服器使用時，您可以啟用 AWS Lambda 在任務成功時運ETL作，或在任務失敗時通知 Amazon SNS 主題。ETL

EMR無伺服器會發出三種事件：

- 應用程式狀態變更事件 — 發出應用程式每個狀態變更的事件。如需應用程式狀態的詳細資訊，請參閱應用狀態。
- Job 執行狀態變更事件 — 發出工作執行之每個狀態變更的事件。如需有關的更多資訊，請參閱作業執行狀態。
- Job 務執行重試事件 — 從 Amazon EMR 無伺服器版本 7.1.0 及更高版本執行的每次重試時發出的事件。

EMR無伺服器 EventBridge事件範例

EMR無伺服器報告的事件值為「aws.emr-serverless指派給」source，如下列範例所示。

應用狀態變更事件

下列範例事件顯示狀CREATING態中的應用程式。

```
{
    "version": "0",
    "id": "9fd3cf79-1ff1-b633-4dd9-34508dc1e660",
    "detail-type": "EMR Serverless Application State Change",
    "source": "aws.emr-serverless",
    "account": "123456789012",
    "time": "2022-05-31T21:16:31Z",
    "region": "us-east-1",
    "resources": [],
    "detail": {
        "applicationId": "00f1cbsc6anuij25",
        "applicationName": "3965ad00-8fba-4932-a6c8-ded32786fd42",
        "arn": "arn:aws:emr-serverless:us-east-1:111122223333:/
applications/00f1cbsc6anuij25",
```

```
        "releaseLabel": "emr-6.6.0",
        "state": "CREATING",
        "type": "HIVE",
        "createdAt": "2022-05-31T21:16:31.547953Z",
        "updatedAt": "2022-05-31T21:16:31.547970Z",
        "autoStopConfig": {
            "enabled": true,
            "idleTimeout": 15
        },
        "autoStartConfig": {
            "enabled": true
        }
    }
}
```

Job 執行狀態變更事件

下列範例事件顯示從狀態移至SCHEDULED狀態的工作執行。RUNNING

```
{
    "version": "0",
    "id": "00df3ec6-5da1-36e6-ab71-20f0de68f8a0",
    "detail-type": "EMR Serverless Job Run State Change",
    "source": "aws.emr-serverless",
    "account": "123456789012",
    "time": "2022-05-31T21:07:42Z",
    "region": "us-east-1",
    "resources": [],
    "detail": {
        "jobRunId": "00f1cbn5g4bb0c01",
        "applicationId": "00f1982r1uukb925",
        "arn": "arn:aws:emr-serverless:us-east-1:123456789012:/
applications/00f1982r1uukb925/jobruns/00f1cbn5g4bb0c01",
        "releaseLabel": "emr-6.6.0",
        "state": "RUNNING",
        "previousState": "SCHEDULED",
        "createdBy": "arn:aws:sts::123456789012:assumed-role/
TestRole-402dcef3ad14993c15d28263f64381e4cda34775/6622b6233b6d42f59c25dd2637346242",
        "updatedAt": "2022-05-31T21:07:42.299487Z",
        "createdAt": "2022-05-31T21:07:25.325900Z"
    }
}
```

Job 執行重試事件

以下是工作執行重試事件的範例。

```
{
    "version": "0",
    "id": "00df3ec6-5da1-36e6-ab71-20f0de68f8a0",
    "detail-type": "EMR Serverless Job Run Retry",
    "source": "aws.emr-serverless",
    "account": "123456789012",
    "time": "2022-05-31T21:07:42Z",
    "region": "us-east-1",
    "resources": [],
    "detail": {
        "jobRunId": "00f1cbn5g4bb0c01",
        "applicationId": "00f1982r1uukb925",
        "arn": "arn:aws:emr-serverless:us-east-1:123456789012:/
applications/00f1982r1uukb925/jobruns/00f1cbn5g4bb0c01",
        "releaseLabel": "emr-6.6.0",
        "createdBy": "arn:aws:sts::123456789012:assumed-role/
TestRole-402dcef3ad14993c15d28263f64381e4cda34775/6622b6233b6d42f59c25dd2637346242",
        "updatedAt": "2022-05-31T21:07:42.299487Z",
        "createdAt": "2022-05-31T21:07:25.325900Z",
        //Attempt Details
        "previousAttempt": 1,
        "previousAttemptState": "FAILED",
        "previousAttemptCreatedAt": "2022-05-31T21:07:25.325900Z",
        "previousAttemptEndedAt": "2022-05-31T21:07:30.325900Z",
        "newAttempt": 2,
        "newAttemptCreatedAt": "2022-05-31T21:07:30.325900Z"
    }
}
```

標記 資源

您可以使用標籤為每個資源指派自己的中繼資料，以協助您管理EMR無伺服器資源。本節提供標籤功能的概觀，並說明如何建立標籤。

主題

什麼是標籤？

標籤是您指派給標籤的標籤 AWS 資源。每個標記皆包含由您定義的索引鍵和值。標籤使您可以對您的進行分類 AWS 依屬性 (例如目的、擁有者和環境) 的資源。當您有許多相同類型的資源時，您可以依據先前指派的標籤，快速識別特定的資源。例如，您可以為 Amazon EMR 無伺服器應用程式定義一組標籤，以協助您追蹤每個應用程式的擁有者和堆疊層級。建議您為每個資源類型設計一組一致的標籤金鑰。

標籤不會自動指派給您的資源。將標籤新增至資源後，您可以隨時修改標籤的值或從資源中移除標籤。標籤對 Amazon EMR 無伺服器沒有任何語義意義，並嚴格解釋為字元字串。若您將與現有標籤具有相同鍵的標籤新增到該資源，則新值會覆寫早前的值。

如果您使用IAM，您可以控制哪些使用者 AWS 帳戶具有管理標籤的權限。如需標籤型存取控制政策範例，請參閱 標籤型存取控制的政策。

標記您的 資源

您可以標記新的或現有的應用程式和工作執行。如果您使用的是 Amazon EMR 無服務器API，AWS CLI，或 AWS SDK，您可以使用相關API動作上的`tags`參數將標籤套用至新資源。您可以使用`TagResource`API動作將標籤套用至現有資源。

在建立資源時，可以使用一些資源建立動作來指定資源的標籤。在這種情況下，如果在建立資源時無法套用標籤，則無法建立資源。該機制可確保您要在建立時標記的資源是以指定的標籤建立，不然就根本不會建立。如果您在建立資源時標記資源，則不需要在建立資源後執行自訂標記指令碼。

下表說明可標記的 Amazon EMR 無伺服器資源。

資源	支援標籤	支援標籤傳播	支援建立時標記 (Amazon EMR 無伺服器API、AWS CLI和 AWS SDK)	API用於創建（標籤可以在創建過程中添加）
應用程式	是	不。 與應用程式相關聯的標記不會傳播到提交至該應用程式的工作執行。	是	`CreateApplication`
作業執行	是	否	是	`StartJobRun`

標記限制

下列基本限制適用於標籤：

- 每個資源最多可以有 50 個使用者建立的標籤。
- 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。
- 最大密鑰長度是 128 碼字符 UTF -8。
- 最大值長度為 UTF -8 中 256 個萬國碼字元。
- 允許的字符是字母，數字，UTF-8 中可表示的空格，以及以下字符：_。：/= +-@。
- 標籤金鑰不得為空白字串。標籤值可以為空白字串，但不得是 null。
- 標籤鍵與值皆區分大小寫。
- 請勿使用`AWS:`或任何大寫或小寫的組合，例如鍵或值的前綴。這些僅保留用於 AWS 使用。

使用標籤 AWS CLI 和 Amazon EMR 無服務器 API

使用以下內容 AWS CLI 用於新增、更新、列出和刪除資源標籤的命令或 Amazon EMR 無伺服器API操作。

資源	支援標籤	支援標籤傳播
新增或覆寫一或多個標籤	tag-resource	TagResource
列出資源的標籤	list-tags-for-resource	ListTagsForResource
刪除一或多個標籤	untag-resource	UntagResource

下列範例顯示如何使用標記或取消標記資源 AWS CLI.

標記現有的應用程式

以下命令標記現有的應用程序。

```
aws emr-serverless tag-resource --resource-arn resource_ARN --tags team=devs
```

取消標記現有的應用程式

以下命令刪除現有應用程序中的標籤。

```
aws emr-serverless untag-resource --resource-arn resource_ARN --tag-keys tag_key
```

列出資源的標籤

以下命令列出與現有資源相關聯的標籤。

```
aws emr-serverless list-tags-for-resource --resource-arn resource_ARN
```

EMR無伺服器教學課程

本節說明使用EMR無伺服器應用程式時的常見使用案例。

主題

將 Java 17 與 Amazon EMR 無伺服器搭配使用

在 Amazon 6.11.0 及更高EMR版本中，您可以將EMR無伺服器 Spark 任務設定為使用 Java 17 執行階段的 Java 虛擬機器 ()。JVM使用以下方法之一來配置與 Java 17 星火。

`JAVA_HOME`

若要覆寫EMR無伺服器 6.11.0 及更新版本的JVM設定，您可以將`JAVA_HOME`設定提供給其`spark.emr-serverless.driverEnv`與`spark.executorEnv`環境分類。

x86_64

設置所需的屬性以指定 Java 17 作為 Spark 驅動程序和執行程序的`JAVA_HOME`配置：

```
--conf spark.emr-serverless.driverEnv.JAVA_HOME=/usr/lib/jvm/java-17-amazon-
corretto.x86_64/
```

```
--conf spark.executorEnv.JAVA_HOME=/usr/lib/jvm/java-17-amazon-corretto.x86_64/
```

arm_64

設置所需的屬性以指定 Java 17 作為 Spark 驅動程序和執行程序的JAVA_HOME配置：

```
--conf spark.emr-serverless.driverEnv.JAVA_HOME=/usr/lib/jvm/java-17-amazon-
corretto.aarch64/
--conf spark.executorEnv.JAVA_HOME=/usr/lib/jvm/java-17-amazon-corretto.aarch64/
```

spark-defaults

或者，您可以在spark-defaults分類中指定 Java 17，以覆寫EMR無伺服器 6.11.0 及更新版本的JVM設定。

x86_64

在spark-defaults分類中指定 Java 17：

```
{
"applicationConfiguration": [
         {
            "classification": "spark-defaults",
            "properties": {
               "spark.emr-serverless.driverEnv.JAVA_HOME" : "/usr/lib/jvm/java-17-
amazon-corretto.x86_64/",
               "spark.executorEnv.JAVA_HOME": "/usr/lib/jvm/java-17-amazon-
corretto.x86_64/"
            }
         }
      ]
}
```

arm_64

在spark-defaults分類中指定 Java 17：

```
{
"applicationConfiguration": [
         {
```

```
            "classification": "spark-defaults",
            "properties": {
               "spark.emr-serverless.driverEnv.JAVA_HOME" : "/usr/lib/jvm/java-17-
amazon-corretto.aarch64/",
               "spark.executorEnv.JAVA_HOME": "/usr/lib/jvm/java-17-amazon-
corretto.aarch64/"
            }
         }
      ]
}
```

搭EMR配無伺服器使用 Apache 胡迪

若要搭配EMR無伺服器應用程式使用 Apache Hudi

1. 在相應的 Spark 作業執行中設定所需的 Spark 屬性。

```
spark.jars=/usr/lib/hudi/hudi-spark-bundle.jar
spark.serializer=org.apache.spark.serializer.KryoSerializer
```

2. 若要將 Hudi 表格同步至已設定的目錄，請指定 AWS Glue 資料目錄做為您的中繼存放區，或設定外部中繼存放區。EMR無伺服器支援hms做為 Hudi 工作負載 Hive 資料表的同步模式。EMR無伺服器會將此內容啟用為預設值。若要進一步瞭解如何設定中繼存放區，請參閱中繼存儲配置。

   > **Important**
   >
   > EMR無伺服器不支援 HIVEQL Hive 表格的同步模式選項，也不支援處理 Hudi 工JDBC作負載的同步模式選項。若要深入瞭解，請參閱同步模式。

   當您使用 AWS Glue 資料目錄做為中繼存放區，您可以為 Hudi 工作指定下列組態屬性。

```
--conf spark.jars=/usr/lib/hudi/hudi-spark-bundle.jar,
--conf spark.serializer=org.apache.spark.serializer.KryoSerializer,
--conf
 spark.hadoop.hive.metastore.client.factory.class=com.amazonaws.glue.catalog.metastore.AWSG
```

若要進一步了解 Amazon 的 Apache Hudi 發行版本EMR，請參閱 Hudi 發行歷史記錄。

將 Apache 冰山與EMR無伺服器搭配使用

將 Apache 冰山與EMR無伺服器應用程式搭配使用

1. 在相應的 Spark 作業執行中設定所需的 Spark 屬性。

```
spark.jars=/usr/share/aws/iceberg/lib/iceberg-spark3-runtime.jar
```

2. 指定 AWS Glue 資料目錄做為您的中繼存放區或設定外部中繼存放區。若要進一步瞭解如何設定中繼存放區，請參閱中繼存儲配置。

   配置要用於冰山的中繼存儲屬性。例如，如果您想要使用 AWS Glue 資料目錄，在應用程式組態中設定下列屬性。

```
spark.sql.catalog.dev.warehouse=s3://DOC-EXAMPLE-BUCKET/EXAMPLE-PREFIX/
spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions
spark.sql.catalog.dev=org.apache.iceberg.spark.SparkCatalog
spark.sql.catalog.dev.catalog-impl=org.apache.iceberg.aws.glue.GlueCatalog
spark.hadoop.hive.metastore.client.factory.class=com.amazonaws.glue.catalog.metastore.AWSGl
```

   當您使用 AWS Glue 資料目錄做為您的中繼存放區，您可以為 Iceberg 工作指定下列組態屬性。

```
--conf spark.jars=/usr/share/aws/iceberg/lib/iceberg-spark3-runtime.jar,
--conf
 spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions,
--conf spark.sql.catalog.dev=org.apache.iceberg.spark.SparkCatalog,
--conf spark.sql.catalog.dev.catalog-impl=org.apache.iceberg.aws.glue.GlueCatalog,
--conf spark.sql.catalog.dev.warehouse=s3://DOC-EXAMPLE-BUCKET/EXAMPLE-PREFIX/
--conf
 spark.hadoop.hive.metastore.client.factory.class=com.amazonaws.glue.catalog.metastore.AWSG
```

要了解有關 Apache 冰山版本的更多信息 Amazon EMR 請參閱冰山發布歷史記錄。

搭配EMR無伺服器使用 Python 程式庫

在 Amazon EMR 無伺服器應用程式上 PySpark 執行任務時，可以將各種 Python 程式庫封裝為相依性。若要這麼做，您可以使用原生 Python 功能、建置虛擬環境，或直接設定 PySpark 工作以使用 Python 程式庫。此頁面涵蓋了每種方法。

使用原生 Python 功能

當您設置以下配置時，您可 PySpark 以使用上傳 Python 文件（ `.py` ），Python 壓縮包（ `.zip` ）和雞蛋文件（ `.egg` ）到星火執行器。

```
--conf spark.submit.pyFiles=s3://DOC-EXAMPLE-BUCKET/EXAMPLE-PREFIX/<.py|.egg|.zip file>
```

如需如何針對 PySpark 工作使用 Python 虛擬環境的詳細資訊，請參閱使用 PySpark 原生功能。

建立虛 Python 環境

若要為一項 PySpark 工作封裝多個 Python 程式庫，您可以建立獨立的 Python 虛擬環境。

1. 要構建 Python 虛擬環境，請使用以下命令。所示範例會將套`scipy`件安裝`matplotlib`到虛擬環境套件中，然後將存檔複製到 Amazon S3 位置。

   > **Important**
   >
   > 您必須在類似的 Amazon Linux 2 環境中運行以下命令，使用與在EMR無服務器中使用相同的 Python 版本，即 Amazon EMR 6.6.0 版本的 Python 3.7.10。您可以在EMR無伺服器範例儲存庫中找到 Docker 檔案範例。 GitHub

   ```
   # initialize a python virtual environment
   python3 -m venv pyspark_venvsource
   source pyspark_venvsource/bin/activate

   # optionally, ensure pip is up-to-date
   pip3 install --upgrade pip

   # install the python packages
   pip3 install scipy
   pip3 install matplotlib

   # package the virtual environment into an archive
   pip3 install venv-pack
   venv-pack -f -o pyspark_venv.tar.gz

   # copy the archive to an S3 location
   aws s3 cp pyspark_venv.tar.gz s3://DOC-EXAMPLE-BUCKET/EXAMPLE-PREFIX/
   ```

```
# optionally, remove the virtual environment directory
rm -fr pyspark_venvsource
```

2. 提交 Spark 工作，並將您的屬性設置為使用 Python 虛擬環境。

```
--conf spark.archives=s3://DOC-EXAMPLE-BUCKET/EXAMPLE-PREFIX/
pyspark_venv.tar.gz#environment
--conf spark.emr-serverless.driverEnv.PYSPARK_DRIVER_PYTHON=./environment/bin/
python
--conf spark.emr-serverless.driverEnv.PYSPARK_PYTHON=./environment/bin/python
--conf spark.executorEnv.PYSPARK_PYTHON=./environment/bin/python
```

請注意，如果您不覆蓋原始的 Python 二進製文件，則前面設置序列中的第二個配置將是`--conf spark.executorEnv.PYSPARK_PYTHON=python`。

有關如何使用 Python 虛擬環境進行 PySpark 作業的更多信息，請參閱使用虛擬環境。如需如何提交 Spark 工作的更多範例，請參閱火花工作。

設定 PySpark 工作以使用 Python 程式庫

使用 Amazon 6.12.0 及更高EMR版本，您可以直接設定EMR無伺服器任 PySpark務以使用熊貓等熱門資料科學 Python 程式庫 NumPy，PyArrow而且無需任何其他設定。

下列範例說明如何封裝 PySpark工作的每個 Python 程式庫。

NumPy

NumPy 是一個用於科學計算的 Python 庫，提供數學，排序，隨機模擬和基本統計的多維數組和運算。若要使用 NumPy，請執行下列命令：

```
import numpy
```

pandas

熊貓是一個建立在之上的 NumPy Python 庫。熊貓庫為數據科學家提供DataFrame數據結構和數據分析工具。要使用熊貓，請運行以下命令：

```
import pandas
```

PyArrow

PyArrow 是一個 Python 庫，用於管理內存中的單欄數據以提高工作性能。 PyArrow 以 Apache Arrow 跨語言開發規範為基礎，這是以單欄格式表示和交換資料的標準方式。若要使用 PyArrow，請執行下列命令：

```
import pyarrow
```

搭配EMR無伺服器使用不同的 Python 版本

除了中的使用案例之外搭配EMR無伺服器使用 Python 程式庫，您還可以使用 Python 虛擬環境使用與 Amazon EMR 無伺服器應用程式在 Amazon 版本中封裝的EMR版本不同的 Python 版本。為此，您必須使用要使用的 Python 版本構建一個 Python 虛擬環境。

若要從 Python 虛擬環境提交工作

1. 使用下列範例中的指令建置虛擬環境。本範例會將 Python 3.9.9 安裝到虛擬環境套件中，並將存檔複製到 Amazon S3 位置。

> **Important**
>
> 如果您使用 Amazon 7.0.0 及更高EMR版本，則必須在 Amazon Linux 2023 環境中執行命令，類似於您用於EMR無伺服器應用程式的指令。
> 如果您使用 6.15.0 或更低版本，則必須在類似的 Amazon Linux 2 環境中執行下列命令。

```
# install Python 3.9.9 and activate the venv
yum install -y gcc openssl-devel bzip2-devel libffi-devel tar gzip wget make
wget https://www.python.org/ftp/python/3.9.9/Python-3.9.9.tgz && \
tar xzf Python-3.9.9.tgz && cd Python-3.9.9 && \
./configure --enable-optimizations && \
make altinstall

# create python venv with Python 3.9.9
python3.9 -m venv pyspark_venv_python_3.9.9 --copies
source pyspark_venv_python_3.9.9/bin/activate

# copy system python3 libraries to venv
cp -r /usr/local/lib/python3.9/* ./pyspark_venv_python_3.9.9/lib/python3.9/
```

```
# package venv to archive.
# **Note** that you have to supply --python-prefix option
# to make sure python starts with the path where your
# copied libraries are present.
# Copying the python binary to the "environment" directory.
pip3 install venv-pack
venv-pack -f -o pyspark_venv_python_3.9.9.tar.gz --python-prefix /home/hadoop/
environment

# stage the archive in S3
aws s3 cp pyspark_venv_python_3.9.9.tar.gz s3://<path>

# optionally, remove the virtual environment directory
rm -fr pyspark_venv_python_3.9.9
```

2. 將您的屬性設定為使用 Python 虛擬環境，並提交星火工作。

```
# note that the archive suffix "environment" is the same as the directory where you
 copied the Python binary.
--conf spark.archives=s3://DOC-EXAMPLE-BUCKET/EXAMPLE-PREFIX/
pyspark_venv_python_3.9.9.tar.gz#environment
--conf spark.emr-serverless.driverEnv.PYSPARK_DRIVER_PYTHON=./environment/bin/
python
--conf spark.emr-serverless.driverEnv.PYSPARK_PYTHON=./environment/bin/python
--conf spark.executorEnv.PYSPARK_PYTHON=./environment/bin/python
```

有關如何使用 Python 虛擬環境進行 PySpark 作業的更多信息，請參閱使用虛擬環境。如需如何提交 Spark 工作的更多範例，請參閱火花工作。

OSS搭配EMR無伺服器使用三角洲湖

Amazon 6.9.0 及更高EMR版本

> **Note**
>
> Amazon EMR 7.0.0 及更高版本使用三角洲湖 3.0.0，它將`delta-core.jar`文件重命名為。`delta-spark.jar`如果您使用 Amazon EMR 7.0.0 或更高版本，請務必`delta-spark.jar`在您的組態中指定。

Amazon EMR 6.9.0 及更高版本包括 Delta Lake，因此您不再需要自行包裝三角洲湖，也不需要為您的EMR無伺服器任務提供--packages旗標。

1. 當您提交EMR無伺服器工作時，請確定您具有下列組態特性，並在sparkSubmitParameters欄位中包含下列參數。

```
--conf spark.jars=/usr/share/aws/delta/lib/delta-core.jar,/usr/share/aws/delta/lib/
delta-storage.jar
    --conf spark.sql.extensions=io.delta.sql.DeltaSparkSessionExtension
    --conf
 spark.sql.catalog.spark_catalog=org.apache.spark.sql.delta.catalog.DeltaCatalog
```

2. 創建一個本地delta_sample.py來測試創建和讀取 Delta 表。

```
# delta_sample.py
    from pyspark.sql import SparkSession

    import uuid

    url = "s3://DOC-EXAMPLE-BUCKET/delta-lake/output/%s/" % str(uuid.uuid4())
    spark = SparkSession.builder.appName("DeltaSample").getOrCreate()

    ## creates a Delta table and outputs to target S3 bucket
    spark.range(5).write.format("delta").save(url)

    ## reads a Delta table and outputs to target S3 bucket
    spark.read.format("delta").load(url).show
```

3. 使用 AWS CLI，將delta_sample.py檔案上傳到您的 Amazon S3 儲存貯體。然後使用命start-job-run令將工作送出至現有的EMR無伺服器應用程式。

```
aws s3 cp delta_sample.py s3://DOC-EXAMPLE-BUCKET/code/

    aws emr-serverless start-job-run \
        --application-id application-id \
        --execution-role-arn job-role-arn \
        --name emr-delta \
        --job-driver '{
            "sparkSubmit": {
                "entryPoint": "s3://DOC-EXAMPLE-BUCKET/code/delta_sample.py",
                "sparkSubmitParameters": "--conf spark.jars=/usr/share/
aws/delta/lib/delta-core.jar,/usr/share/aws/delta/lib/delta-storage.jar --
```

```
conf spark.sql.extensions=io.delta.sql.DeltaSparkSessionExtension --conf
 spark.sql.catalog.spark_catalog=org.apache.spark.sql.delta.catalog.DeltaCatalog"
            }
       }'
```

若要搭配 Delta Lake 使用 Python 程式庫，您可以將程式`delta-core`庫封裝為相依性，或將其用作自訂映像檔來新增程式庫。

或者，您可以使`SparkContext.addPyFile`用從`delta-core`JAR檔案中新增 Python 程式庫：

```
import glob
from pyspark.sql import SparkSession

spark = SparkSession.builder.getOrCreate()
spark.sparkContext.addPyFile(glob.glob("/usr/share/aws/delta/lib/delta-core_*.jar")[0])
```

Amazon 6.8.0 及更低EMR版本

如果您使用的是 Amazon EMR 6.8.0 或更低版本，請按照以下步驟將 Delta 湖OSS與EMR無伺服器應用程式搭配使用。

1. 若要建立與 Amazon EMR 無伺服器應用程式上 Spark 版本相容的 Delta Lake 開放原始碼版本，請導覽至三角洲 GitHub並遵循指示進行。
2. 將三角洲湖程式庫上傳到您的 Amazon S3 儲存貯體 AWS 帳戶.
3. 當您在應用程式組態中提交EMR無伺服器工作時，請將目前位於值區中的 Delta Lake JAR 檔案納入。

   ```
   --conf spark.jars=s3://DOC-EXAMPLE-BUCKET/jars/delta-core_2.12-1.1.0.jar
   ```
4. 若要確保您可以讀取和寫入 Delta 資料表，請執行範例 PySpark測試。

   ```
   from pyspark import SparkConf, SparkContext
       from pyspark.sql import HiveContext, SparkSession

       import uuid

       conf = SparkConf()
       sc = SparkContext(conf=conf)
       sqlContext = HiveContext(sc)
   ```

```
    url = "s3://DOC-EXAMPLE-BUCKET/delta-lake/output/1.0.1/%s/" % str(uuid.uuid4())

    ## creates a Delta table and outputs to target S3 bucket
    session.range(5).write.format("delta").save(url)

    ## reads a Delta table and outputs to target S3 bucket
    session.read.format("delta").load(url).show
```

從 Airflow 提交EMR無伺服器工作

Apache 氣流中的 Amazon 供應商為EMR無伺服器操作員提供。如需有關操作員的詳細資訊，請參閱 Apache 氣流文件中的 Amazon EMR 無伺服器操作員。

您可以使用創EmrServerlessCreateApplicationOperator建一個星火或蜂巢應用程序。您也可以使EmrServerlessStartJobOperator用新應用程式開始一或多個工作。

若要將操作員與 Apache Airflow (MWAA) 搭配 Airflow 2.2.2 的 Amazon 受管工作流程搭配使用，請將以下行新增至您的requirements.txt檔案，並更新您的MWAA環境以使用新檔案。

```
apache-airflow-providers-amazon==6.0.0
boto3>=1.23.9
```

請注意，EMR無伺服器支援已新增至 Amazon 供應商的 5.0.0 版。版本 6.0.0 是與氣流 2.2.2 兼容的最新版本。您可以在MWAA氣流 2.4.3 的情況下使用更高版本。

下列縮寫範例顯示如何建立應用程式、執行多個 Spark 工作，然後停止應用程式。EMR無伺服器範例 GitHub 儲存庫提供完整範例。如需sparkSubmit組態的其他詳細資訊，請參閱火花工作。

```
from datetime import datetime

from airflow import DAG
from airflow.providers.amazon.aws.operators.emr import (
    EmrServerlessCreateApplicationOperator,
    EmrServerlessStartJobOperator,
    EmrServerlessDeleteApplicationOperator,
)

# Replace these with your correct values
JOB_ROLE_ARN = "arn:aws:iam::account-id:role/emr_serverless_default_role"
S3_LOGS_BUCKET = "DOC-EXAMPLE-BUCKET"
```

```
DEFAULT_MONITORING_CONFIG = {
    "monitoringConfiguration": {
        "s3MonitoringConfiguration": {"logUri": f"s3://DOC-EXAMPLE-BUCKET/logs/"}
    },
}

with DAG(
    dag_id="example_endtoend_emr_serverless_job",
    schedule_interval=None,
    start_date=datetime(2021, 1, 1),
    tags=["example"],
    catchup=False,
) as dag:
    create_app = EmrServerlessCreateApplicationOperator(
        task_id="create_spark_app",
        job_type="SPARK",
        release_label="emr-6.7.0",
        config={"name": "airflow-test"},
    )

    application_id = create_app.output

    job1 = EmrServerlessStartJobOperator(
        task_id="start_job_1",
        application_id=application_id,
        execution_role_arn=JOB_ROLE_ARN,
        job_driver={
            "sparkSubmit": {
                "entryPoint": "local:///usr/lib/spark/examples/src/main/python/
pi_fail.py",
            }
        },
        configuration_overrides=DEFAULT_MONITORING_CONFIG,
    )

    job2 = EmrServerlessStartJobOperator(
        task_id="start_job_2",
        application_id=application_id,
        execution_role_arn=JOB_ROLE_ARN,
        job_driver={
            "sparkSubmit": {
                "entryPoint": "local:///usr/lib/spark/examples/src/main/python/pi.py",
                "entryPointArguments": ["1000"]
            }
```

```
        },
        configuration_overrides=DEFAULT_MONITORING_CONFIG,
    )

    delete_app = EmrServerlessDeleteApplicationOperator(
        task_id="delete_app",
        application_id=application_id,
        trigger_rule="all_done",
    )

    (create_app >> [job1, job2] >> delete_app)
```

搭配EMR無伺服器使用 Hive 使用者定義

Hive 使用者定義函數 (UDFs) 可讓您建立自訂函數來處理記錄或記錄群組。在本教學中，您將使用範例UDF搭配預先存在的 Amazon EMR 無伺服器應用程式來執行輸出查詢結果的任務。若要瞭解如何設定應用程式，請參閱開始使用 Amazon EMR 無伺服器。

UDF搭配EMR無伺服器使用

1. 導覽至以GitHub取得範例UDF。克隆回購並切換到您要使用的 git 分支。將存放庫`pom.xml`檔案`maven-compiler-plugin`中的更新為具有來源。同時將目標 java 版本配置更新為`1.8`. 執行`mvn package -DskipTests`以建立包含範例的JAR檔案UDFs。
2. 建立JAR檔案後，請使用下列指令將檔案上傳到 S3 儲存貯體。

   ```
   aws s3 cp brickhouse-0.8.2-JS.jar s3://DOC-EXAMPLE-BUCKET/jars/
   ```

3. 建立範例檔案以使用其中一個範例UDF函數。將此查詢另存為`udf_example.q`並將其上傳到您的 S3 儲存貯體。

   ```
   add jar s3://DOC-EXAMPLE-BUCKET/jars/brickhouse-0.8.2-JS.jar;
   CREATE TEMPORARY FUNCTION from_json AS 'brickhouse.udf.json.FromJsonUDF';
   select from_json('{"key1":[0,1,2], "key2":[3,4,5,6], "key3":[7,8,9]}', map("",
    array(cast(0 as int))));
   select from_json('{"key1":[0,1,2], "key2":[3,4,5,6], "key3":[7,8,9]}', map("",
    array(cast(0 as int))))["key1"][2];
   ```

4. 提交以下蜂巢工作。

   ```
   aws emr-serverless start-job-run \
   ```

```
  --application-id application-id \
  --execution-role-arn job-role-arn \
  --job-driver '{
    "hive": {
        "query": "s3://DOC-EXAMPLE-BUCKET/queries/udf_example.q",
        "parameters": "--hiveconf hive.exec.scratchdir=s3://DOC-EXAMPLE-BUCKET/emr-
serverless-hive/scratch --hiveconf hive.metastore.warehouse.dir=s3://'$BUCKET'/emr-
serverless-hive/warehouse"
    }
}' --configuration-overrides '{
    "applicationConfiguration": [{
        "classification": "hive-site",
        "properties": {
            "hive.driver.cores": "2",
            "hive.driver.memory": "6G"
        }
    }],
    "monitoringConfiguration": {
        "s3MonitoringConfiguration": {
            "logUri": "s3://DOC-EXAMPLE-BUCKET/logs/"
        }
    }
}'
```

5. 使用get-job-run指令檢查工作的狀態。等待狀態變更為SUCCESS。

```
aws emr-serverless get-job-run --application-id application-id --job-run-id job-id
```

6. 使用以下命令下載輸出文件。

```
aws s3 cp --recursive s3://DOC-EXAMPLE-BUCKET/logs/applications/application-id/
jobs/job-id/HIVE_DRIVER/ .
```

檔stdout.gz案類似下列。

```
{"key1":[0,1,2],"key2":[3,4,5,6],"key3":[7,8,9]}
2
```

搭配EMR無伺服器使用自訂映像

主題

使用自訂的 Python 版本

您可以建立自訂映像檔以使用不同版本的 Python。例如，若要將 Python 版本 3.10 用於星火工作，請執行下列命令：

```
FROM public.ecr.aws/emr-serverless/spark/emr-6.9.0:latest

USER root

# install python 3
RUN yum install -y gcc openssl-devel bzip2-devel libffi-devel tar gzip wget make
RUN wget https://www.python.org/ftp/python/3.10.0/Python-3.10.0.tgz && \
tar xzf Python-3.10.0.tgz && cd Python-3.10.0 && \
./configure --enable-optimizations && \
make altinstall

# EMRS will run the image as hadoop
USER hadoop:hadoop
```

在您提交 Spark 工作之前，請將屬性設定為使用 Python 虛擬環境，如下所示。

```
--conf spark.emr-serverless.driverEnv.PYSPARK_DRIVER_PYTHON=/usr/local/bin/python3.10
--conf spark.emr-serverless.driverEnv.PYSPARK_PYTHON=/usr/local/bin/python3.10
--conf spark.executorEnv.PYSPARK_PYTHON=/usr/local/bin/python3.10
```

使用自訂的 Java 版本

下面的例子演示了如何構建一個自定義映像使用 Java 11 為您的 Spark 作業。

```
FROM public.ecr.aws/emr-serverless/spark/emr-6.9.0:latest

USER root

# install JDK 11
```

```
RUN sudo amazon-linux-extras install java-openjdk11

# EMRS will run the image as hadoop
USER hadoop:hadoop
```

提交星火作業之前，設置星火屬性使用 Java 11，如下所示。

```
--conf spark.executorEnv.JAVA_HOME=/usr/lib/jvm/java-11-
openjdk-11.0.16.0.8-1.amzn2.0.1.x86_64
--conf spark.emr-serverless.driverEnv.JAVA_HOME=/usr/lib/jvm/java-11-
openjdk-11.0.16.0.8-
```

建立資料科學影像

下面的例子演示了如何包括常見的，數據科學 Python 包，如熊貓和 NumPy.

```
FROM public.ecr.aws/emr-serverless/spark/emr-6.9.0:latest

USER root

# python packages
RUN pip3 install boto3 pandas numpy
RUN pip3 install -U scikit-learn==0.23.2 scipy
RUN pip3 install sk-dist
RUN pip3 install xgboost

# EMR Serverless will run the image as hadoop
USER hadoop:hadoop
```

使用 Apache 塞多納處理地理空間資料

下面的示例演示了如何構建一個圖像，以包括 Apache Sedona 進行地理空間處理。

```
FROM public.ecr.aws/emr-serverless/spark/emr-6.9.0:latest

USER root

RUN yum install -y wget
RUN wget https://repo1.maven.org/maven2/org/apache/sedona/sedona-core-3.0_2.12/1.3.0-
incubating/sedona-core-3.0_2.12-1.3.0-incubating.jar -P /usr/lib/spark/jars/
RUN pip3 install apache-sedona
```

```
# EMRS will run the image as hadoop
USER hadoop:hadoop
```

在 Amazon 無服務器上使用亞馬遜紅移集成阿帕奇星火 EMR

在 Amazon 6.9.0 及更高EMR版本中，每個版本映像都包含 Apache 星火和 Amazon Redshift 之間的連接器。有了這個連接器，您可以使用亞馬遜EMR無伺服器上的 Spark 來處理儲存在 Amazon Redshift 中的資料。整合是以 `spark-redshift` 開放原始碼連接器為基礎。對於 Amazon EMR 無服務器，阿帕奇星火的 Amazon Redshift 集成作為原生集成包括在內。

主題

- 啟動一個星火應用程序與 Amazon Redshift 集成阿帕奇星火
- 使用 Apache Spark 的 Amazon Redshift 整合進行身分驗證
- 在 Amazon Redshift 中讀取和寫入
- 使用 Spark 連接器時的考量和限制

啟動一個星火應用程序與 Amazon Redshift 集成阿帕奇星火

若要使用與EMR無伺服器 6.9.0 的整合，您必須將必要的 Spark Redshift 相依性與 Spark 工作傳遞。用`--jars`於包括與 Redshift 連接器相關的程式庫。若要查看 `--jars` 選項支援的其他檔案位置，請參閱 Apache Spark 說明文件的進階相依性管理一節。

- `spark-redshift.jar`
- `spark-avro.jar`
- `RedshiftJDBC.jar`
- `minimal-json.jar`

Amazon 6.10.0 及更高EMR版本不需要相`minimal-json.jar`依性，預設會自動將其他相依性安裝到每個叢集。下列範例說明如何為 Apache Spark 啟動與 Amazon Redshift 整合的 Spark 應用程式。

Amazon EMR 6.10.0 +

在EMR無伺服器版本 6.10.0 及更高版本上，利用 Amazon Redshift 整合，在亞馬遜EMR無伺服器上啟動 Spark 任務。

```
spark-submit my_script.py
```

Amazon EMR 6.9.0

若要在 Amazon EMR 無伺服器上使用 Amazon Redshift 整合在無伺服EMR器 6.9.0 版本上啟動 Spark 任務，請使用下列範例所示的`--jars`選項。請注意，與`--jars`選項一起列示的路徑是JAR檔案的預設路徑。

```
--jars
    /usr/share/aws/redshift/jdbc/RedshiftJDBC.jar,
    /usr/share/aws/redshift/spark-redshift/lib/spark-redshift.jar,
    /usr/share/aws/redshift/spark-redshift/lib/spark-avro.jar,
    /usr/share/aws/redshift/spark-redshift/lib/minimal-json.jar
```

```
spark-submit \
  --jars /usr/share/aws/redshift/jdbc/RedshiftJDBC.jar,/usr/share/aws/redshift/
spark-redshift/lib/spark-redshift.jar,/usr/share/aws/redshift/spark-redshift/lib/
spark-avro.jar,/usr/share/aws/redshift/spark-redshift/lib/minimal-json.jar \
  my_script.py
```

使用 Apache Spark 的 Amazon Redshift 整合進行身分驗證

使用 AWS Secrets Manager 檢索憑據並連接到 Amazon Redshift

您可以將登入資料儲存在 Secrets Manager 中，以安全地向 Amazon Redshift 進行身份驗證，並讓 Spark 任務呼叫`GetSecretValue`API來擷取它：

```
from pyspark.sql import SQLContextimport boto3

sc = # existing SparkContext
sql_context = SQLContext(sc)

secretsmanager_client = boto3.client('secretsmanager',
 region_name=os.getenv('AWS_REGION'))
secret_manager_response = secretsmanager_client.get_secret_value(
    SecretId='string',
    VersionId='string',
    VersionStage='string'
)
```

```
username = # get username from secret_manager_response
password = # get password from secret_manager_response
url = "jdbc:redshift://redshifthost:5439/database?user=" + username + "&password="
 + password

# Access to Redshift cluster using Spark
```

使用驅動程式向 Amazon Redshift 進行驗證 JDBC

設置用戶名和密碼 JDBC URL

您可以在中指定 Amazon Redshift 資料庫名稱和密碼，以向 Amazon Redshift 叢集驗證星火任務。JDBC URL

> **Note**
>
> 如果您在中傳遞資料庫認證URL，則擁有存取權的任何人也URL可以存取認證。通常不建議使用此方法，因為它不安全。

如果您的應用程式無法考慮安全性，您可以使用下列格式在中設定使用者名稱和密碼 JDBCURL：

```
jdbc:redshift://redshifthost:5439/database?user=username&password=password
```

搭配 Amazon EMR 無伺服器任務執行角色使用IAM基於身份驗證

從 Amazon EMR 無伺服器版本 6.9.0 開始，Amazon Redshift JDBC 驅動程式 2.1 或更高版本已封裝到環境中。使用 2.1 及更高版本的JDBC驅動程式，您可以指定JDBCURL且不包含原始使用者名稱和密碼。

相反地，可以指定 `jdbc:redshift:iam://` 配置。這會命令JDBC驅動程式使用您的EMR無伺服器作業執行角色來自動擷取認證。如需詳細資訊，請參閱 Amazon Redshift 管理指南中的設定JDBC或ODBC連線以使用IAM登入資料。這方面的一個例子URL是：

```
jdbc:redshift:iam://examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com:5439/
dev
```

符合提供的條件時，您的工作執行角色需要下列權限：

權限	作業執行角色所需的條件
`redshift:GetClusterCredentials`	JDBC驅動程序需要從 Amazon Redshift 獲取憑據
`redshift:DescribeCluster`	如果您指定了 Amazon Redshift 叢集和 AWS 區域 在JDBCURL而不是端點
`redshift-serverless:GetCredentials`	JDBC驅動程式需要從 Amazon Redshift 無伺服器擷取登入資料
`redshift-serverless:GetWorkgroup`	如果您使用的是 Amazon Redshift 無伺服器，且您要根據工作群組名稱和URL區域指定

在不同的地方連接到 Amazon Redshift VPC

當您在下方設定佈建的 Amazon Redshift 叢集或 Amazon Redshift 無伺服器工作群組時VPC，必須為 Amazon EMR 無伺服器應用程式設定VPC連線，才能存取資源。如需如何在EMR無伺服器應用程式上設定VPC連線的詳細資訊，請參閱設定VPC存取權。

- 如果您佈建的 Amazon Redshift 叢集或 Amazon Redshift 無伺服器工作群組可公開存取，您可以指定在建立無伺服器應用程式時已連接NAT閘道的一或多個私有子網路。EMR
- 如果您佈建的 Amazon Redshift 叢集或 Amazon Redshift 無伺服器工作群組無法公開存取，您必須依照中所述，為您的 Amazon Redshift 叢集建立 Amazon Redshift 受管VPC端點。設定VPC存取權或者，您也可以按照亞馬遜 Redshift 管理指南中的連接到 Amazon Redshift 無伺服器中的說明來建立自己的 Amazon Redshift 無伺服器工作群組。您必須將叢集或子群組與建立EMR無伺服器應用程式時指定的私人子網路建立關聯。

> **Note**
>
> 如果您使用IAM基於身份驗證，且EMR無伺服器應用程式的私有子網路沒有連接NAT閘道，則還必須在這些子網路上建立適用於 Amazon Redshift 或 Amazon Redshift 無伺服器的VPC端點。這樣，JDBC驅動程序可以獲取憑據。

在 Amazon Redshift 中讀取和寫入

下列程式碼範例用 PySpark 於從具有資料來源和 Spark SQL 的 Amazon Redshift 資料庫讀取API和寫入範例資料。

Data source API

用 PySpark 於從具有資料來源的 Amazon Redshift 資料庫讀取和寫入範例資料API。

```
import boto3
from pyspark.sql import SQLContext

sc = # existing SparkContext
sql_context = SQLContext(sc)

url = "jdbc:redshift:iam://redshifthost:5439/database"
aws_iam_role_arn = "arn:aws:iam::account-id:role/role-name"

df = sql_context.read \
    .format("io.github.spark_redshift_community.spark.redshift") \
    .option("url", url) \
    .option("dbtable", "table-name") \
    .option("tempdir", "s3://path/for/temp/data") \
    .option("aws_iam_role", "aws-iam-role-arn") \
    .load()

df.write \
    .format("io.github.spark_redshift_community.spark.redshift") \
    .option("url", url) \
    .option("dbtable", "table-name-copy") \
    .option("tempdir", "s3://path/for/temp/data") \
    .option("aws_iam_role", "aws-iam-role-arn") \
    .mode("error") \
    .save()
```

SparkSQL

用於 PySpark 使用 Spark SQL 從 Amazon Redshift 資料庫讀取和寫入範例資料。

```
import boto3
import json
import sys
import os
```

```
from pyspark.sql import SparkSession

spark = SparkSession \
    .builder \
    .enableHiveSupport() \
    .getOrCreate()

url = "jdbc:redshift:iam://redshifthost:5439/database"
aws_iam_role_arn = "arn:aws:iam::account-id:role/role-name"

bucket = "s3://path/for/temp/data"
tableName = "table-name" # Redshift table name

s = f"""CREATE TABLE IF NOT EXISTS {table-name} (country string, data string)
    USING io.github.spark_redshift_community.spark.redshift
    OPTIONS (dbtable '{table-name}', tempdir '{bucket}', url '{url}', aws_iam_role
 '{aws-iam-role-arn}' ); """

spark.sql(s)

columns = ["country" ,"data"]
data = [("test-country","test-data")]
df = spark.sparkContext.parallelize(data).toDF(columns)

# Insert data into table
df.write.insertInto(table-name, overwrite=False)
df = spark.sql(f"SELECT * FROM {table-name}")
df.show()
```

使用 Spark 連接器時的考量和限制

- 我們建議您打開從 Amazon 上SSL的星火EMR到亞馬 Amazon Redshift 的JDBC連接。
- 我們建議您在中管理 Amazon Redshift 叢集的登入資料 AWS Secrets Manager 作為最佳實踐。請參閱使用 AWS Secrets Manager 檢索用於連接到 Amazon Redshift 的憑據舉個例子。
- 我們建議您傳遞具有 Amazon Redshift 身份驗證`aws_iam_role`參數參數的IAM角色。
- 參數 `tempformat` 目前不支援 Parquet 格式。
- 這些`tempdir`URI指向 Amazon S3 位置。此暫時目錄不會自動清理，因此可能會增加額外的費用。
- 請考慮下列針對 Amazon Redshift 的建議：
  - 建議您封鎖對 Amazon Redshift 叢集的公開存取。

  - 建議您開啟 Amazon Redshift 稽核日誌。
  - 建議您開啟 Amazon Redshift 靜態加密。
- 請考慮下列針對 Amazon S3 的建議：
  - 建議您封鎖對 Amazon S3 儲存貯體的公開存取。
  - 建議您使用 Amazon S3 伺服器端加密來加密所用的 S3 儲存貯體。
  - 建議您使用 Amazon S3 生命週期政策來定義 Amazon S3 儲存貯體的保留規則。
  - Amazon EMR 一律會驗證從開放原始碼匯入映像的程式碼。出於安全考慮，我們不支援下列從 Spark 到 Amazon S3 的身分驗證方法：
    - 設定 AWS `hadoop-env`組態分類中的存取金鑰
    - 編碼 AWS 存取金鑰 `tempdir` URI

如需有關使用連接器及其支援參數的詳細資訊，請參閱下列資源：

- 《Amazon Redshift 管理指南》中的 Apache Spark 的 Amazon Redshift 整合
- Github 上的 `spark-redshift` 社群儲存庫

使用 Amazon 無伺服器連線至 DynamoDB EMR

在本教學中，您會從美國地理名稱板上傳資料子集到 Amazon S3 儲存貯體，然後使用 Amazon EMR 無伺服器上的 Hive 或 Spark 將資料複製到您可以查詢的 Amazon DynamoDB 表格。

步驟 1：將資料上傳到 Amazon S3 儲存貯體

若要建立 Amazon S3 儲存貯體，請按照 Amazon 簡單儲存服務主控台使用者指南中建立儲存貯體中的說明進行操作。將*DOC-EXAMPLE-BUCKET*參照取代為新建立值區的名稱。現在您的EMR無伺服器應用程式已準備好執行作業。

1. `features.zip`使用以下命令下載示例數據存檔。

   ```
   wget https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/samples/
   features.zip
   ```

2. 從存`features.txt`檔中提取文件並查看文件中的前幾行：

   ```
   unzip features.zip
   head features.txt
   ```

結果看起來應類似下列內容。

```
1535908|Big Run|Stream|WV|38.6370428|-80.8595469|794
875609|Constable Hook|Cape|NJ|40.657881|-74.0990309|7
1217998|Gooseberry Island|Island|RI|41.4534361|-71.3253284|10
26603|Boone Moore Spring|Spring|AZ|34.0895692|-111.410065|3681
1506738|Missouri Flat|Flat|WA|46.7634987|-117.0346113|2605
1181348|Minnow Run|Stream|PA|40.0820178|-79.3800349|1558
1288759|Hunting Creek|Stream|TN|36.343969|-83.8029682|1024
533060|Big Charles Bayou|Bay|LA|29.6046517|-91.9828654|0
829689|Greenwood Creek|Stream|NE|41.596086|-103.0499296|3671
541692|Button Willow Island|Island|LA|31.9579389|-93.0648847|98
```

此處每行中的欄位指示唯一識別碼、名稱、自然特徵類型、狀態、緯度 (以度為單位)、經度 (以度為單位) 以及以英呎為單位的高度。

3. 將您的資料上傳到 Amazon S3

```
aws s3 cp features.txt s3://DOC-EXAMPLE-BUCKET/features/
```

第 2 步：創建一個蜂巢表

使用 Apache 星火或蜂巢建立包含 Amazon S3 中上傳資料的新蜂巢資料表。

Spark

要創建一個星火蜂房表，運行以下命令。

```
import org.apache.spark.sql.SparkSession

val sparkSession = SparkSession.builder().enableHiveSupport().getOrCreate()

sparkSession.sql("CREATE TABLE hive_features \
    (feature_id BIGINT, \
    feature_name STRING, \
    feature_class STRING, \
    state_alpha STRING, \
    prim_lat_dec DOUBLE, \
    prim_long_dec DOUBLE, \
    elev_in_ft BIGINT) \
    ROW FORMAT DELIMITED \
```

```
    FIELDS TERMINATED BY '|' \
    LINES TERMINATED BY '\n' \
    LOCATION 's3://DOC-EXAMPLE_BUCKET/features';")
```

您現在有一個填充的 Hive 表從`features.txt`文件中的數據。要驗證您的數據是在表中，如下面的例子中運行 Spark SQL 查詢。

```
sparkSession.sql(
    "SELECT state_alpha, COUNT(*) FROM hive_features GROUP BY state_alpha;")
```

Hive

要創建蜂巢配置單元表，運行以下命令。

```
CREATE TABLE hive_features
    (feature_id             BIGINT,
    feature_name            STRING ,
    feature_class           STRING ,
    state_alpha             STRING,
    prim_lat_dec            DOUBLE ,
    prim_long_dec           DOUBLE ,
    elev_in_ft              BIGINT)
    ROW FORMAT DELIMITED
    FIELDS TERMINATED BY '|'
    LINES TERMINATED BY '\n'
    LOCATION 's3://DOC-EXAMPLE-BUCKET/features';
```

現在，您有一個包含`features.txt`文件中的數據的 Hive 表。若要驗證資料是否在資料表中，請執行 HiveQL 查詢，如下列範例所示。

```
SELECT state_alpha, COUNT(*) FROM hive_features GROUP BY state_alpha;
```

步驟 3：將資料複製到

使用星火或蜂巢將資料複製到新的 DynamoDB 資料表。

Spark

若要將您在上一個步驟中建立的 Hive 資料表中的資料複製到 DynamoDB，請遵循將資料複製到 DynamoDB 中的步驟 1-3。這會建立名為的新 DynamoDB 表格。Features然後，您可以直接從文字檔讀取資料，並將其複製到 DynamoDB 表格，如下列範例所示。

```
import com.amazonaws.services.dynamodbv2.model.AttributeValue
import org.apache.hadoop.dynamodb.DynamoDBItemWritable
import org.apache.hadoop.dynamodb.read.DynamoDBInputFormat
import org.apache.hadoop.io.Text
import org.apache.hadoop.mapred.JobConf
import org.apache.spark.SparkContext

import scala.collection.JavaConverters._

object EmrServerlessDynamoDbTest {

    def main(args: Array[String]): Unit = {

        jobConf.set("dynamodb.input.tableName", "Features")
        jobConf.set("dynamodb.output.tableName", "Features")
        jobConf.set("dynamodb.region", "region")

        jobConf.set("mapred.output.format.class",
 "org.apache.hadoop.dynamodb.write.DynamoDBOutputFormat")
        jobConf.set("mapred.input.format.class",
 "org.apache.hadoop.dynamodb.read.DynamoDBInputFormat")

        val rdd = sc.textFile("s3://DOC-EXAMPLE-BUCKET/ddb-connector/")
            .map(row => {
                val line = row.split("\\|")
                val item = new DynamoDBItemWritable()

                val elevInFt = if (line.length > 6) {
                    new AttributeValue().withN(line(6))
                } else {
                    new AttributeValue().withNULL(true)
                }

                item.setItem(Map(
                    "feature_id" -> new AttributeValue().withN(line(0)),
                    "feature_name" -> new AttributeValue(line(1)),
                    "feature_class" -> new AttributeValue(line(2)),
```

```
                    "state_alpha" -> new AttributeValue(line(3)),
                    "prim_lat_dec" -> new AttributeValue().withN(line(4)),
                    "prim_long_dec" -> new AttributeValue().withN(line(5)),
                    "elev_in_ft" -> elevInFt)
                    .asJava)
                (new Text(""), item)
        })
        rdd.saveAsHadoopDataset(jobConf)
    }
}
```

Hive

若要將您在上一個步驟中建立的 Hive 表格中的資料複製到 DynamoDB，請遵循將資料複製到 DynamoDB 中的指示進行操作。

步驟 4：查詢來自 DynamoDB 料

使用星火或蜂巢查詢您的 DynamoDB 資料表。

Spark

若要查詢您在上一個步驟中建立的 DynamoDB 資料表中的資料，您可以使用 Spark SQL 或 Spark。 MapReduce API

Example — 使用星火查詢您的 DynamoDB 資料表 SQL

下面的 Spark SQL 查詢返回按字母順序排列的所有功能類型的列表。

```
val dataFrame = sparkSession.sql("SELECT DISTINCT feature_class \
    FROM ddb_features \
    ORDER BY feature_class;")
```

下列 Spark SQL 查詢會傳回以字母 M 開頭的所有湖泊清單。

```
val dataFrame = sparkSession.sql("SELECT feature_name, state_alpha \
    FROM ddb_features \
    WHERE feature_class = 'Lake' \
    AND feature_name LIKE 'M%' \
    ORDER BY feature_name;")
```

下列 Spark SQL 查詢會傳回至少三個高於一英哩之特徵的所有狀態清單。

```
val dataFrame = sparkSession.dql("SELECT state_alpha, feature_class, COUNT(*) \
    FROM ddb_features \
    WHERE elev_in_ft > 5280 \
    GROUP by state_alpha, feature_class \
    HAVING COUNT(*) >= 3 \
    ORDER BY state_alpha, feature_class;")
```

Example — 使用星火查詢您的 DynamoDB 表 MapReduce API

下列 MapReduce 查詢會依字母順序傳回所有特徵類型的清單。

```
val df = sc.hadoopRDD(jobConf, classOf[DynamoDBInputFormat], classOf[Text],
 classOf[DynamoDBItemWritable])
    .map(pair => (pair._1, pair._2.getItem))
    .map(pair => pair._2.get("feature_class").getS)
    .distinct()
    .sortBy(value => value)
    .toDF("feature_class")
```

下列 MapReduce 查詢會傳回以字母 M 開頭的所有湖泊清單。

```
val df = sc.hadoopRDD(jobConf, classOf[DynamoDBInputFormat], classOf[Text],
 classOf[DynamoDBItemWritable])
    .map(pair => (pair._1, pair._2.getItem))
    .filter(pair => "Lake".equals(pair._2.get("feature_class").getS))
    .filter(pair => pair._2.get("feature_name").getS.startsWith("M"))
    .map(pair => (pair._2.get("feature_name").getS,
 pair._2.get("state_alpha").getS))
    .sortBy(_._1)
    .toDF("feature_name", "state_alpha")
```

下列 MapReduce 查詢會傳回至少三個高於 1 英哩之特徵的所有狀態清單。

```
val df = sc.hadoopRDD(jobConf, classOf[DynamoDBInputFormat], classOf[Text],
 classOf[DynamoDBItemWritable])
    .map(pair => pair._2.getItem)
    .filter(pair => pair.get("elev_in_ft").getN != null)
    .filter(pair => Integer.parseInt(pair.get("elev_in_ft").getN) > 5280)
    .groupBy(pair => (pair.get("state_alpha").getS, pair.get("feature_class").getS))
    .filter(pair => pair._2.size >= 3)
    .map(pair => (pair._1._1, pair._1._2, pair._2.size))
    .sortBy(pair => (pair._1, pair._2))
```

```
    .toDF("state_alpha", "feature_class", "count")
```

Hive

若要查詢您在上一個步驟中建立的 DynamoDB 表格中的資料，請按照查詢 DynamoDB 表格中的資料中的指示進行操作。

設定跨帳戶存取權

若要設定EMR無伺服器的跨帳戶存取，請完成以下步驟。在此範例中，AccountA是您建立 Amazon EMR 無伺服器應用程式的帳戶，AccountB也是 Amazon DynamoDB 所在的帳戶。

1. 在中建立 DynamoDB 資料表。AccountB如需詳細資訊，請參閱步驟 1：建立資料表。
2. 在中AccountB建立可存取 DynamoDB 表格的Cross-Account-Role-BIAM角色。

   a. 登入 AWS Management Console 然後在開啟IAM主控台https://console.aws.amazon.com/iam/。

   b. 選擇「角色」，然后創建一個名為的新角色Cross-Account-Role-B。有關如何建立IAM角色的詳細資訊，請參閱《使用者指南》中的〈建立IAM角色〉。

   c. 建立一個IAM政策，以授與跨帳戶 DynamoDB 表的存取權限。然後將IAM原則附加至Cross-Account-Role-B。

      以下是授與 DynamoDB 表格存取權的政策。CrossAccountTable

      ```
      {"Version": "2012-10-17",
          "Statement": [
              {"Effect": "Allow",
                  "Action": "dynamodb:*",
                  "Resource": "arn:aws:dynamodb:region:AccountB:table/
      CrossAccountTable"
              }
          ]
      }
      ```

   d. 編輯 Cross-Account-Role-B 角色的信任關係。

      若要設定角色的信任關係，請在 [步驟 2：跨帳戶-角色 B] 中建立的角色選擇IAM主控台中的 [信任關係] 索引標籤。

選取編輯信任關係，然後新增下列原則文件。本文件允許Job-Execution-Role-A在中AccountA擔任此Cross-Account-Role-B角色。

```
{"Version": "2012-10-17",
  "Statement": [
    {"Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::AccountA:role/Job-Execution-Role-A"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

e. 授予Job-Execution-Role-A要假設AccountA的- STS Assume role權限Cross-Account-Role-B。

在IAM控制台中 AWS 帳戶 AccountA」中，選取Job-Execution-Role-A。將以下政策陳述式新增至 Job-Execution-Role-A 以允許 Cross-Account-Role-B 角色的 AssumeRole 動作。

```
{"Version": "2012-10-17",
    "Statement": [
        {"Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::AccountB:role/Cross-Account-Role-B"
        }
    ]
}
```

f. 將具有值的dynamodb.customAWSCredentialsProvider屬性設定為核心-網站分類com.amazonaws.emr.AssumeRoleAWSCredentialsProvider中的內容。ASSUME_ROLE_CREDENTIALS_ROLE_ARN使用的ARN值設定環境變數Cross-Account-Role-B。

3. 使用運行星火或蜂巢作業Job-Execution-Role-A。

考量事項

將 DynamoDB 連接器與 Apache 星火搭配使用時的考量

- Spark SQL 不支持使用存儲處理程序選項創建 Hive 表。如需詳細資訊，請參閱 Apache Spark 文件中的指定 Hive 資料表的儲存格式。
- 星火SQL不支持與存儲處理程序的STORED BY操作。如果您想要透過外部 Hive 資料表與 DynamoDB 資料表互動，請先使用 Hive 建立資料表。
- 若要將查詢轉換為 DynamoDB 查詢，DynamoDB 連接器會使用述詞下推。述詞下推篩選依據對應至 DynamoDB 表之分區索引鍵的資料行來篩選資料。述詞下推只有在您將連接器與 Spark 搭配使用時才會運作SQL，而不會與. MapReduce API

將 DynamoDB 連接器與 Apache 配置單元搭配使用時的考量

調整映射器的最大數量

- 如果您使用SELECT查詢從對應至 DynamoDB 的外部 Hive 表格讀取資料，則EMR無伺服器上的對映工作數會計算為 DynamoDB 表格設定的總讀取輸送量，除以每個地圖任務的輸送量。每個地圖工作的預設輸送量為 100。
- Hive 工作可以使用超過每個EMR無伺服器應用程式所設定容器數目上限的對應工作數目，具體取決於針對 DynamoDB 設定的讀取輸送量。此外，長時間執行的 Hive 查詢可能會消耗 DynamoDB 表格的所有佈建讀取容量。這會對其他使用者產生負面影響。
- 您可以使用此dynamodb.max.map.tasks性質來設定地圖工作的上限。您也可以使用此屬性，根據工作容器大小調整每個對映工作讀取的資料量。
- 您可以在 Hive 查詢級別或start-job-run命令的hive-site分類中設置dynamodb.max.map.tasks屬性。此數值必須等於或大於 1。當 Hive 處理您的查詢時，產生的 Hive 工作使用的值不會超過從 DynamoDB 表格讀取dynamodb.max.map.tasks時的值。

調整每個工作的寫入輸送量

- EMR無伺服器上每個工作的寫入輸送量計算為 DynamoDB 表格設定的總寫入輸送量除以內容值。mapreduce.job.maps對於蜂巢，此屬性的默認值是 2。因此 Hive 作業最後階段的前兩個工作可以消耗所有寫入輸送量。這會導致限制相同工作或其他工作中其他任務的寫入。
- 若要避免寫入限制，您可以根據最後階段中的工作數目或您要為每個工作配置的寫入輸送量來設定mapreduce.job.maps屬性值。在EMR無伺服器上的start-job-run指令mapred-site分類中設定此屬性。

安全

雲端安全 AWS 是最高的優先級。作為 AWS 客戶，您可以從資料中心和網路架構中獲益，該架構專為滿足對安全性最敏感的組織的需求而打造。

安全是兩者之間的共同責任 AWS 和你。共同責任模型 將此描述為雲端的安全和雲端內的安全：

- 雲端的安全性 — AWS 負責保護運行的基礎設施 AWS 中的服務 AWS 雲端。 AWS 還為您提供可以安全使用的服務。第三方稽核員會定期測試和驗證我們安全性的有效性，作為 AWS 合規方案。若要了解適用於 Amazon EMR 無伺服器的合規計劃，請參閱 AWS 合規計劃範圍內的服務。
- 雲端中的安全性 — 您的責任取決於 AWS 您使用的服務。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 Amazon EMR 無伺服器時套用共同責任模型。本章的主題說明如何設定 Amazon EMR 無伺服器以及如何使用其他 AWS 符合您安全性與合規目標的服務。

主題

Amazon EMR 無伺服器的安全性最佳實務

Amazon EMR 無伺服器提供許多安全功能，可在您開發和實作自己的安全政策時考慮。以下最佳實務為一般準則，並不代表完整的安全解決方案。這些最佳實務可能不適用或無法滿足您的環境需求，因此請將其視為實用建議就好，而不要當作是指示。

套用最低權限準則

EMR無伺服器會為使用角色 (例如執行IAM角色) 的應用程式提供精細的存取原則。建議只授予作業所需的最低權限集，例如覆蓋應用程式和日誌目的地的存取權限。我們還建議定期以及在應用程式碼發生變更時審核作業許可。

隔離不受信任的應用程式碼

EMR無伺服器會在屬於不同無伺服器應用EMR程式的工作間建立完整的網路隔離 在需要工作層級隔離的情況下，請考慮將工作隔離到不同EMR的無伺服器應用程式中。

以角色為基礎的存取控制 (RBAC) 權限

系統管理員應嚴格控制EMR無伺服器應用程式的角色型存取控制 (RBAC) 權限。

資料保護

所以此 AWS 共同責任模式適用於 Amazon EMR 無伺服器中的資料保護。如本模型所述， AWS 負責保護運行所有的全球基礎設施 AWS 雲端。您負責維護在此基礎設施上託管內容的控制權。此內容包括安全性設定和管理工作 AWS 您使用的服務。如需有關資料隱私權的詳細資訊，請參閱資料隱私權 FAQ。如需歐洲資料保護的相關資訊，請參閱 AWS 共同責任模型和GDPR博客文章 AWS 安全部落格。

出於數據保護目的，我們建議您進行保護 AWS 帳戶憑據並設置個人帳戶 AWS Identity and Access Management (IAM)。如此一來，每個使用者都只會獲得授予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：

- 對每個帳戶使用多重要素驗證 (MFA)。
- 使用SSL/TLS與之溝通 AWS 的費用。我們建議 TLS 1.2 或更高版本。
- 設定API和使用者活動記錄 AWS CloudTrail.
- 使用 AWS 加密解決方案，以及其中的所有默認安全控制 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 Simple Storage Service (Amazon Simple Storage Service (Amazon S3)) 的個人資料。
- 使用 Amazon EMR 無伺服器加密選項來加密靜態和傳輸中的資料。
- 如果您在訪問時需要 FIPS 140-2 驗證的加密模塊 AWS 透過指令行介面或API使用FIPS端點。如需有關可用FIPS端點的詳細資訊，請參閱聯邦資訊處理標準 (FIPS) 140-2。

我們強烈建議您絕對不要將客戶帳戶號碼等敏感的識別資訊，放在自由格式的欄位中，例如Name (名稱) 欄位。這包括當您使用 Amazon 無服務器或其他EMR服務器時 AWS 使用控制台的服務API，AWS CLI，或 AWS SDKs。您輸入 Amazon EMR 無伺服器或其他服務的任何資料都可能會被拾取以包含在診斷日誌中。當您提供URL給外部伺服器時，請勿在中包含認證資訊URL以驗證您對該伺服器的要求。

靜態加密

資料加密有助於防止未經授權的使用者讀取叢集上的資料和相關的資料儲存體系統。這包括儲存到持久性媒體的資料 (稱為靜態資料)，以及透過網路傳送時可能會被攔截的資料 (稱為傳輸中資料)。

資料加密需要金鑰和憑證。您可以從多個選項中進行選擇，包括由管理的金鑰 AWS Key Management Service、由 Amazon S3 管理的金鑰，以及您提供之自訂供應商的金鑰和憑證。使用時 AWS KMS 作為您的金鑰提供者，儲存和使用加密金鑰需支付費用。如需詳細資訊，請參閱 AWS KMS 定價。

在指定加密選項之前，請先決定要使用的金鑰和憑證管理系統。然後為您指定作為加密設定一部分的自訂提供者建立金鑰和憑證。

針對 Amazon S3 中的資EMRFS料進行靜態加密

每個EMR無伺服器應用程式都使用特定的發行版本，其中包括 EMRFS (EMR檔案系統)。Amazon S3 加密可與從 Amazon S3 讀取和寫入的EMR檔案系統 (EMRFS) 物件搭配使用。啟用靜態加密時，您可以將 Amazon S3 伺服器端加密 (SSECSE) 或用戶端加密 () 指定為預設加密模式。或者，您可以使用 Per bucket encryption overrides (每個儲存貯體加密覆寫) 為個別儲存貯體指定不同的加密方法。無論是否啟用 Amazon S3 加密，傳輸層安全性 (TLS) 都會加密EMR叢集節點和 Amazon S3 之間傳輸中的EMRFS物件。如果您將 Amazon S3 CSE 與客戶管理金鑰搭配使用，則用於在EMR無伺服器應用程式中執行任務的執行角色必須具有金鑰的存取權。如需 Amazon S3 加密的深入資訊，請參閱 Amazon 簡單儲存服務開發人員指南中的使用加密保護資料。

> **Note**
>
> 當您使用 AWS KMS，儲存和使用加密金鑰需支付費用。如需詳細資訊，請參閱 AWS KMS 定價。

Amazon S3 伺服器端加密

當您設定 Amazon S3 伺服器端加密時，Amazon S3 會在將資料寫入磁碟時在物件層級加密資料，並在存取時解密資料。如需詳細資訊SSE，請參閱 Amazon 簡單儲存服務開發人員指南中的使用伺服器端加密保護資料。

SSE在 Amazon EMR 無伺服器中指定時，您可以選擇兩種不同的金鑰管理系統：

- SSE-S3-Amazon S3 會為您管理金鑰。無EMR伺服器不需要其他設定。
- SSE-KMS 您使用的是 AWS KMS key 以設定適用於EMR無伺服器的原則。無EMR伺服器不需要其他設定。

使用 AWS KMS 對寫入 Amazon S3 的資料進行加密，使用 `StartJobRunAPI`. 您可以為寫入 Amazon S3 的所有內容啟用加密，也可以為寫入特定儲存貯體的資料啟用加密。如需有關的詳細資訊 `StartJobRunAPI`，請參閱EMR無伺服器API參考。

若要開啟 AWS KMS 對您寫入 Amazon S3 的所有資料進行加密，當您呼叫 `StartJobRunAPI`.

```
--conf spark.hadoop.fs.s3.enableServerSideEncryption=true
--conf spark.hadoop.fs.s3.serverSideEncryption.kms.keyId=<kms_id>
```

若要開啟 AWS KMS 對您寫入特定值區的資料進行加密，請在呼叫 `StartJobRunAPI`.

```
--conf spark.hadoop.fs.s3.bucket.<DOC-EXAMPLE-BUCKET>.enableServerSideEncryption=true
--conf spark.hadoop.fs.s3.bucket.<DOC-EXAMPLE-
BUCKET>.serverSideEncryption.kms.keyId=<kms-id>
```

SSE使用客戶提供的金鑰 (SSE-C) 無法與EMR無伺服器搭配使用。

Amazon S3 用戶端加密

使用 Amazon S3 用EMRFS戶端加密，Amazon S3 加密和解密會在每個 Amazon EMR 版本上提供的用戶端中進行。物件在上傳至 Amazon S3 之前會先加密，並在下載後解密。您指定的提供者會提供用

戶端使用的加密金鑰。用戶端可以使用提供的金鑰 AWS KMS (CSE-KMS) 或提供用戶端根金鑰 (CSE-C) 的自訂 Java 類別。CSE-KMS 和 CSE-C 之間的加密細節略有不同，具體取決於指定的提供程序和要解密或加密的對象的元數據。如果您將 Amazon S3 CSE 與客戶管理金鑰搭配使用，則用於在EMR無伺服器應用程式中執行任務的執行角色必須具有金鑰的存取權。可能需要KMS支付額外費用。如需有關這些差異的詳細資訊，請參閱 Amazon 簡單儲存服務開發人員指南中的使用用戶端加密保護資料。

本機磁碟加密

存儲在臨時存儲中的數據使用服務擁有的密鑰使用業界標準 AES -256 加密算法進行加密。

金鑰管理

您可以設定KMS為自動旋轉KMS金鑰。這將每年輪換一次金鑰，同時無限期儲存舊金鑰，以便您的資料仍然可以解密。如需詳細資訊，請參閱輪換客戶主金鑰。

傳輸中加密

Amazon EMR 無伺服器提供下列應用程式特定的加密功能：

- Spark
  - 默認情況下，Spark 驅動程序和執行者之間的通信進行身份驗證和內部。RPC驅動程序和執行者之間的通信被加密。
- Hive
  - 之間的溝通 AWS Glue 中繼存放區和EMR無伺服器應用程式透過. TLS

您應該只允許使用 Amazon S3 儲存貯體IAM政策上的 aws: SecureTransport 條件透過 HTTPS (TLS) 加密連線。

Amazon EMR 無伺服器中的 Identity and Access Management (IAM)

AWS Identity and Access Management (IAM) 是 AWS 服務 協助系統管理員安全地控制存取 AWS 的費用。IAM管理員控制誰可以驗證 (登入) 和授權 (具有權限) 使用 Amazon EMR 無伺服器資源。IAM是一個 AWS 服務 您可以使用，無需額外費用。

主題

物件

您如何使用 AWS Identity and Access Management (IAM) 會有所不同，這取決於您在 Amazon EMR 無伺服器中所做的工作。

服務使用者 — 如果您使用 Amazon EMR 無伺服器服務執行工作，則管理員會為您提供所需的登入資料和許可。當您使用更多 Amazon EMR 無伺服器功能完成工作時，您可能需要額外的許可。了解存取許可的管理方式可協助您向管理員請求正確的許可。如果您無法存取 Amazon EMR 無伺服器中的功能，請參閱疑難排解 Amazon EMR 無伺服器身分和存取。

服務管理員 — 如果您負責公司的 Amazon EMR 無伺服器資源，您可能擁有完整的 Amazon 無EMR伺服器存取權。判斷服務使用者應存取哪些 Amazon EMR 無伺服器功能和資源是您的工作。然後，您必須向IAM管理員提交請求，才能變更服務使用者的權限。檢閱此頁面上的資訊，以瞭解的基本概念 IAM。若要進一步了解貴公司如何IAM搭配 Amazon EMR 無伺服器使用，請參閱Amazon EMR 無伺服器中的 Identity and Access Management (IAM)。

IAM管理員 — 如果您是管理IAM員，您可能想要了解如何撰寫政策以管理 Amazon EMR 無伺服器存取的詳細資訊。若要檢視可在IAM中使用的 Amazon EMR 無伺服器身分識別政策範例，請參閱。無伺服器的身分型原則範例 EMR

使用身分驗證

驗證是您登入的方式 AWS 使用您的身份證明。您必須經過驗證 (登入至 AWS) 作為 AWS 帳戶根使用者，以IAM使用者身分或假定IAM角色。

您可以登入 AWS 使用透過身分識別來源提供的認證做為聯合身分識別。 AWS IAM Identity Center (IAM身分識別中心) 使用者、貴公司的單一登入驗證，以及您的 Google 或 Facebook 認證都是聯合身分識別的範例。當您以同盟身分登入時，您的管理員先前會使用IAM角色設定聯合身分識別。當您存取 AWS 通過使用聯合，您間接擔任一個角色。

根據您的使用者類型，您可以登入 AWS Management Console 或 AWS 存取入口網站。如需登入的詳細資訊 AWS，請參閱如何登入 AWS 帳戶 中的 AWS 登入 使用者指南。

如果您訪問 AWS 編程方式， AWS 提供軟體開發套件 (SDK) 和命令列介面 (CLI)，以使用您的認證以密碼編譯方式簽署您的要求。如果你不使用 AWS 工具，您必須自己簽署請求。如需使用建議的方法自行簽署要求的詳細資訊，請參閱簽署 AWS API《IAM用戶指南》中的請求。

無論您使用何種身分驗證方法，您可能都需要提供額外的安全性資訊。例如 AWS 建議您使用多重要素驗證 (MFA) 來增加帳戶的安全性。要了解更多信息，請參閱中的多因素身份驗證 AWS IAM Identity Center 用戶指南和使用多因素身份驗證（MFA）AWS (在 IAM 使用者指南中)

AWS 帳戶 根使用者

當您創建 AWS 帳戶時，您會從一個擁有完整存取權限的登入身分開始 AWS 服務 和帳戶中的資源。這個身份被稱為 AWS 帳戶 root 使用者，並透過使用您用來建立帳戶的電子郵件地址和密碼登入來存取。強烈建議您不要以根使用者處理日常任務。保護您的根使用者憑證，並將其用來執行只能由根使用者執行的任務。如需需要您以 root 使用者身分登入的完整工作清單，請參閱《使用指南》中的〈需要 root 使用者認證的IAM工作〉。

聯合身分

最佳作法是要求人類使用者 (包括需要管理員存取權的使用者) 使用與身分識別提供者的同盟來存取 AWS 服務 通過使用臨時憑據。

聯合身分是來自您企業使用者目錄的使用者、Web 身分識別提供者、 AWS Directory Service、身分識別中心目錄或存取的任何使用者 AWS 服務 使用透過身分識別來源提供的認證。同盟身分存取時 AWS 帳戶，他們假定角色，並且角色提供臨時認證。

對於集中式存取管理，我們建議您使用 AWS IAM Identity Center。 您可以在 IAM Identity Center 中建立使用者和群組，也可以連線並同步處理至您自己身分識別來源中的一組使用者和群組，以便在您的所有身分識別來源中使用 AWS 帳戶 和應用程序。如需IAM身分識別中心的相關資訊，請參閱IAM識別中心是什麼？ 在 AWS IAM Identity Center 使用者指南。

IAM 使用者和群組

使IAM用者是您的身分 AWS 帳戶 具有單一人員或應用程式的特定權限。在可能的情況下，我們建議您仰賴臨時登入資料，而不要建立具有長期認證 (例如密碼和存取金鑰) 的IAM使用者。不過，如果您的特定使用案例需要使用IAM者的長期認證，建議您輪換存取金鑰。如需詳細資訊，請參閱《使用指南》中的「IAM定期輪換存取金鑰」以瞭解需要長期認證的使用案例。

IAM群組是指定IAM使用者集合的身分識別。您無法以群組身分簽署。您可以使用群組來一次為多名使用者指定許可。群組可讓管理大量使用者許可的程序變得更為容易。例如，您可以擁有一個名為的群組，IAMAdmins並授與該群組管理IAM資源的權限。

使用者與角色不同。使用者只會與單一人員或應用程式建立關聯，但角色的目的是在由任何需要它的人員取得。使用者擁有永久的長期憑證，但角色僅提供暫時憑證。要了解更多信息，請參閱《IAM用戶指南》中的創建用戶（而不是角色）的IAM時間。

IAM角色

IAM角色是您的身份 AWS 帳戶 具有特定權限。它類似於用IAM戶，但不與特定人員相關聯。您可以暫時擔任的IAM角色 AWS Management Console 通過切換角色。您可以通過調用一個角色 AWS CLI 或 AWS API操作或通過使用自定義URL。如需有關使用角色方法的詳細資訊，請參閱《使用指南》中的IAM〈使用IAM角色〉。

IAM具有臨時認證的角色在下列情況下很有用：

- 聯合身分使用者存取 — 如需向聯合身分指派許可，請建立角色，並為角色定義許可。當聯合身分進行身分驗證時，該身分會與角色建立關聯，並獲授予由角色定義的許可。如需聯合角色的相關資訊，請參閱《使用指南》中的〈建立第三方身分識別提供IAM者的角色〉。如果您使用IAM身分識別中心，則需要設定權限集。為了控制身分驗證後可以存取的內IAM容，IAMIdentity Center 會將權限集與中的角色相關聯。如需有關權限集的資訊，請參閱 AWS IAM Identity Center 使用者指南。
- 暫時IAM使用者權限 — IAM 使用者或角色可以假定某個IAM角色，暫時取得特定工作的不同權限。
- 跨帳戶存取 — 您可以使用IAM角色允許不同帳戶中的某個人 (受信任的主體) 存取您帳戶中的資源。角色是授予跨帳戶存取權的主要方式。但是，有一些 AWS 服務，您可以將策略直接附加到資源（而不是使用角色作為代理）。若要瞭解跨帳戶存取角色與以資源為基礎的政策之間的差異，請參閱《IAM使用指南》IAM中的〈跨帳號資源存取〉。
- 跨服務訪問 — 一些 AWS 服務 使用其他中的功能 AWS 服務。 例如，當您在服務中撥打電話時，該服務通常會在 Amazon 中執行應用程式EC2或將物件存放在 Amazon S3 中。服務可能會使用呼叫主體的許可、使用服務角色或使用服務連結角色來執行此作業。

- 轉寄存取工作階段 (FAS) — 當您使用使用IAM者或角色執行動作時 AWS，您被視為校長。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS使用主體呼叫 AWS 服務，與請求相結合 AWS 服務 向下游服務提出請求。FAS只有當服務收到需要與其他人互動的請求時才會發出請求 AWS 服務 或要完成的資源。在此情況下，您必須具有執行這兩個動作的許可。有關提出FAS請求時的策略詳細信息，請參閱轉發訪問會話。
- 服務角色 — 服務角色是指服務代表您執行動作所代表的IAM角色。IAM管理員可以從中建立、修改和刪除服務角色IAM。如需詳細資訊，請參閱建立角色以將權限委派給 AWS 服務 (在 IAM 使用者指南中)
- 服務連結角色 — 服務連結角色是連結至 AWS 服務。 服務可以扮演角色代表您執行動作。服務連結角色會出現在 AWS 帳戶 並由服務擁有。IAM管理員可以檢視 (但無法編輯服務連結角色) 的權限。

- 在 Amazon 上執行的應用程式 EC2 — 您可以使用IAM角色來管理在執行個體上EC2執行並製作的應用程式的臨時登入資料 AWS CLI 或 AWS API請求。這比在EC2實例中存儲訪問密鑰更好。若要指派 AWS EC2執行個體的角色並讓它可供其所有應用程式使用，您可以建立連接至執行個體的執行個體設定檔。執行個體設定檔包含角色，可讓執行個體上EC2執行的程式取得臨時登入資料。如需詳細資訊，請參閱使用者指南中的使用IAM角色將許可授與在 Amazon EC2 執行個體上執行的應IAM用程式。

要了解是否使用IAM角色還是用IAM戶，請參閱《用戶指南》中的「IAM創建IAM角色的時機（而不是用戶）」。

使用政策管理存取權

您可以控制存取 AWS 藉由建立原則並將其附加至 AWS 身分識別或資源。原則是中的物件 AWS 當與身分識別或資源相關聯時，會定義其權限。 AWS 當主參與者 (使用者、root 使用者或角色工作階段) 提出要求時，評估這些原則。政策中的許可決定是否允許或拒絕請求。大多數策略都儲存在 AWS 作為JSON文件。如需有關JSON原則文件結構和內容的詳細資訊，請參閱《IAM使用指南》中的策略概觀。JSON

管理員可以使用 AWS JSON策略，用於指定誰可以存取什麼內容。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。

預設情況下，使用者和角色沒有許可。若要授與使用者對所需資源執行動作的權限，IAM管理員可以建立IAM策略。然後，系統管理員可以將IAM原則新增至角色，使用者可以擔任這些角色。

IAM原則會定義動作的權限，不論您用來執行作業的方法為何。例如，假設您有一個允許 `iam:GetRole` 動作的政策。具有該策略的使用者可以從 AWS Management Console，該 AWS CLI，或 AWS API.

身分型政策

以身分識別為基礎的原則是您可以附加至身分識別 (例如使用者、使用IAM者群組或角色) 的JSON權限原則文件。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。若要瞭解如何建立以身分識別為基礎的策略，請參閱《IAM使用指南》中的〈建立IAM策略〉。

身分型政策可進一步分類成內嵌政策或受管政策。內嵌政策會直接內嵌到單一使用者、群組或角色。受管理的策略是獨立策略，您可以將其附加到您的多個使用者、群組和角色 AWS 帳戶。 受管政策包括 AWS 受管理的政策和客戶管理的政策。若要了解如何在受管策略或內嵌策略之間進行選擇，請參閱《IAM使用手冊》中的「在受管策略和內嵌策略之間進行選擇」。

資源型政策

以資源為基礎的JSON策略是您附加至資源的政策文件。以資源為基礎的政策範例包括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。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。主參與者可以包括帳戶、使用者、角色、同盟使用者或 AWS 服務.

資源型政策是位於該服務中的內嵌政策。你不能使用 AWS 在以資源為基礎的策略IAM中受管理的策略。

存取控制清單 (ACLs)

存取控制清單 (ACLs) 控制哪些主參與者 (帳戶成員、使用者或角色) 具有存取資源的權限。ACLs類似於以資源為基礎的策略，雖然它們不使用JSON政策文件格式。

Amazon S3， AWS WAF和 Amazon VPC 是支持的服務的例子ACLs。若要進一步了解ACLs，請參閱 Amazon 簡單儲存服務開發人員指南中的存取控制清單 (ACL) 概觀。

其他政策類型

AWS 支援其他較不常見的原則類型。這些政策類型可設定較常見政策類型授予您的最大許可。

- 權限界限 — 權限界限是一項進階功能，您可以在其中設定以身分識別為基礎的原則可授與給IAM實體 (IAM使用者或角色) 的最大權限。您可以為實體設定許可界限。所產生的許可會是實體的身分型政策和其許可界限的交集。會在 `Principal` 欄位中指定使用者或角色的資源型政策則不會受到許可界

限限制。所有這類政策中的明確拒絕都會覆寫該允許。如需有關權限界限的詳細資訊，請參閱《IAM使用指南》中的IAM實體的權限界限。

- 服務控制策略（SCPs）— SCPs 是指定中組織或組織單位（OU）的最大權限的JSON策略 AWS Organizations. AWS Organizations 是一種用於分組和集中管理多個服務 AWS 帳戶 您的企業擁有。如果您啟用組織中的所有功能，則可以將服務控制策略 (SCPs) 套用至您的任何或所有帳戶。SCP 限制成員帳戶中實體的權限，包括每個 AWS 帳戶根使用者。 如需有關 Organizations 的詳細資訊 SCPs，請參閱中的服務控制原則 AWS Organizations 使用者指南。
- 工作階段政策 – 工作階段政策是一種進階政策，您可以在透過編寫程式的方式建立角色或聯合使用者的暫時工作階段時，作為參數傳遞。所產生工作階段的許可會是使用者或角色的身分型政策和工作階段政策的交集。許可也可以來自資源型政策。所有這類政策中的明確拒絕都會覆寫該允許。如需詳細資訊，請參閱《IAM使用指南》中的工作階段原則。

多種政策類型

將多種政策類型套用到請求時，其結果形成的許可會更為複雜、更加難以理解。若要瞭解如何 AWS 決定當涉及多個原則類型時是否允許要求，請參閱《IAM使用指南》中的「原則評估邏輯」。

EMR無伺服器如何搭配使用 IAM

在您用IAM來管理對 Amazon EMR 無伺服器的存取之前，請先了解哪些IAM功能可與 Amazon 無EMR伺服器搭配使用。

IAM可搭配EMR無伺服器使用的功能

IAM特徵	Amazon EMR 無伺服器支援
身分型政策	是
資源型政策	否
政策動作	是
政策資源	是
政策條件索引鍵	否
ACLs	否
ABAC(策略中的標籤)	是

IAM特徵	Amazon EMR 無伺服器支援
暫時性憑證	是
主體許可	是
服務角色	否
服務連結角色	是

若要取得EMR無伺服器和其他方式的高階檢視 AWS 服務適用於大多數IAM功能，請參閱 AWS《IAM使用者指南》IAM中使用的服務。

無伺服器的身分識別原則 EMR

支援身分型政策：是

以身分識別為基礎的原則是您可以附加至身分識別 (例如使用者、使用IAM者群組或角色) 的JSON權限原則文件。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。若要瞭解如何建立以身分識別為基礎的策略，請參閱《IAM使用指南》中的〈建立IAM策略〉。

使用以IAM身分識別為基礎的策略，您可以指定允許或拒絕的動作和資源，以及允許或拒絕動作的條件。您無法在身分型政策中指定主體，因為這會套用至連接的使用者或角色。若要瞭解可在JSON策略中使用的所有元素，請參閱《使用IAM者指南》中的IAMJSON策略元素參考資料。

無伺服器的身分型原則範例 EMR

若要檢視 Amazon EMR 無伺服器身分識別型政策的範例，請參閱。無伺服器的身分識別原則範例 EMR

無伺服器內EMR的資源型政策

支援資源型政策：否

以資源為基礎的JSON策略是您附加至資源的政策文件。以資源為基礎的政策範例包括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。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。主參與者可以包括帳戶、使用者、角色、同盟使用者或 AWS 服務.

若要啟用跨帳戶存取，您可以在以資源為基礎的策略中指定一個或多個帳戶中的一個或多個帳戶中的IAM實體作為主體。新增跨帳戶主體至資源型政策，只是建立信任關係的一半。當主參與者與資源不同時 AWS 帳戶，受信任帳戶中的IAM系統管理員也必須授與主參與者實體 (使用者或角色) 存取資源的權限。其透過將身分型政策連接到實體來授與許可。不過，如果資源型政策會為相同帳戶中的主體授予存取，這時就不需要額外的身分型政策。如需詳細資訊，請參閱《IAM使用指南》IAM中的〈跨帳號資源存取〉。

EMR無伺服器的原則動作

支援政策動作：是

管理員可以使用 AWS JSON策略，用於指定誰可以存取什麼內容。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策略Action元素描述了您可以用來允許或拒絕策略中存取的動作。策略動作通常與關聯的名稱相同 AWS API操作。有一些例外情況，例如沒有匹配API操作的僅限權限的操作。也有一些作業需要政策中的多個動作。這些額外的動作稱為相依動作。

政策會使用動作來授予執行相關聯動作的許可。

若要查看EMR無伺服器動作清單，請參閱服務授權參考中適用於 Amazon EMR 無伺服器的動作、資源和條件金鑰。

EMR無伺服器中的原則動作會在動作之前使用下列前置詞。

```
emr-serverless
```

若要在單一陳述式中指定多個動作，請用逗號分隔。

```
"Action": [
      "emr-serverless:action1",
      "emr-serverless:action2"
         ]
```

若要檢視 Amazon EMR 無伺服器身分識別型政策的範例，請參閱。無伺服器的身分識別原則範例 EMR

EMR無伺服器的原則資源

支援政策資源：是

管理員可以使用 AWS JSON策略，用於指定誰可以存取什麼內容。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。

ResourceJSON原則元素會指定要套用動作的一或多個物件。陳述式必須包含 Resource 或 NotResource 元素。最佳做法是使用其 Amazon 資源名稱 (ARN) 指定資源。您可以針對支援特定資源類型的動作 (稱為資源層級許可) 來這麼做。

對於不支援資源層級許可的動作 (例如列出操作)，請使用萬用字元 (*) 來表示陳述式適用於所有資源。

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

若要查看 Amazon EMR 無伺服器資源類型及其清單ARNs，請參閱服務授權參考中由 Amazon EMR 無伺服器定義的資源。若要了解您可以為每個資源指定哪些動作，請參閱 Amazon EMR 無伺服器ARN的動作、資源和條件金鑰。

若要檢視 Amazon EMR 無伺服器身分識別型政策的範例，請參閱。無伺服器的身分識別原則範例 EMR

EMR無伺服器的原則條件金鑰

支援服務特定政策條件金鑰	否

管理員可以使用 AWS JSON策略，用於指定誰可以存取什麼內容。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 (或 Condition 區塊) 可讓您指定使陳述式生效的條件。Condition 元素是選用項目。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。

如果您在一個語句中指定多個Condition元素，或在單個Condition元素中指定多個鍵， AWS 使用邏輯AND操作評估它們。如果您為單個條件鍵指定多個值， AWS 使用邏輯OR運算評估條件。必須符合所有條件，才會授與陳述式的許可。

您也可以在指定條件時使用預留位置變數。例如，只有在IAM使用者名稱標記資源時，您才可以授與IAM使用者存取資源的權限。如需詳細資訊，請參閱《IAM使用指南》中的IAM政策元素：變數和標籤。

AWS 支援全域條件金鑰和服務特定條件金鑰。若要查看全部 AWS 全域條件索引鍵，請參閱 AWS《IAM使用指南》中的整體條件前後關聯鍵字。

若要查看 Amazon EMR 無伺服器條件金鑰清單，並了解您可以使用條件金鑰的動作和資源，請參閱服務授權參考中適用於 Amazon EMR 無伺服器的動作、資源和條件金鑰。

所有 Amazon EC2 操作都支持aws:RequestedRegion和ec2:Region條件密鑰。如需詳細資訊，請參閱範例：限制特定區域的存取權。

EMR無伺服器中的存取控制清單 (ACLs)

支持ACLs：無

存取控制清單 (ACLs) 控制哪些主參與者 (帳戶成員、使用者或角色) 具有存取資源的權限。ACLs類似於以資源為基礎的策略，雖然它們不使用JSON政策文件格式。

以屬性為基礎的存取控制 (ABAC) 與無伺服器 EMR

支援 ABAC (策略中的標記)	是

以屬性為基礎的存取控制 (ABAC) 是一種授權策略，可根據屬性定義權限。In (入) AWS，這些屬性稱為標籤。您可以將標籤附加到IAM實體 (使用者或角色) 以及許多實體 AWS 的費用。標記實體和資源是的第一步ABAC。然後，您可以設計ABAC策略，以便在主參與者的標籤與他們嘗試存取的資源上的標籤相符時允許作業。

ABAC在快速成長的環境中很有幫助，並且有助於原則管理變得繁瑣的情況。

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。

如需有關的詳細資訊ABAC，請參閱什麼是ABAC？ 在《IAM使用者指南》中。若要檢視包含設定步驟的自學課程ABAC，請參閱《使用指南》中的〈使用以屬性為基礎的存取控制 (ABAC) IAM 〉。

搭配EMR無伺服器使用臨時認證

支援臨時憑證：是

一些 AWS 服務 使用臨時憑據登錄時不起作用。有關其他信息，包括 AWS 服務 使用臨時登入資料，請參閱 AWS 服務 在《IAM使用者指南》IAM中使用。

如果您登入，您正在使用臨時認證 AWS Management Console 使用除了使用者名稱和密碼之外的任何方法。例如，當您訪問 AWS 使用貴公司的單一登入 (SSO) 連結，該程序會自動建立臨時登入資料。當您以使用者身分登入主控台，然後切換角色時，也會自動建立臨時憑證。如需有關切換角色的詳細資訊，請參閱《IAM使用者指南》中的〈切換到角色 (主控台)〉。

您可以使用手動建立臨時登入資料 AWS CLI 或 AWS API。然後，您可以使用這些臨時登入資料來存取 AWS. AWS 建議您動態產生臨時登入資料，而非使用長期存取金鑰。如需詳細資訊，請參閱IAM。

無伺服器EMR的跨服務主體權限

支援轉寄存取工作階段 (FAS)：是

當您使用使用IAM者或角色執行動作 AWS，您被視為校長。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS使用主體呼叫 AWS 服務，與請求相結合 AWS 服務 向下游服務提出請求。FAS只有當服務收到需要與其他人互動的請求時才會發出請求 AWS 服務 或要完成的資源。在此情況下，您必須具有執行這兩個動作的許可。有關提出FAS請求時的策略詳細信息，請參閱轉發訪問會話。

無伺服EMR器的服務角色

支援服務角色	否

無伺服器EMR的服務連結角色

支援服務連結角色	是

如需建立或管理服務連結角色的詳細資訊，請參閱 AWS 與之合作的服務IAM。在表格中尋找服務，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。

針EMR對無伺服器使用服務連結角色

Amazon EMR 無伺服器使用 AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是直接連結至EMR無伺服器的唯一IAM角色類型。服務連結角色由EMR無伺服器預先定義，並包含服務呼叫其他人所需的所有權限 AWS 代表您提供的服務。

服務連結角色可讓EMR無伺服器的設定更容易，因為您不需要手動新增必要的權限。EMR無伺服器會定義其服務連結角色的權限，除非另有定義，否則只有EMR無伺服器可以擔任其角色。定義的權限包括信任原則和權限原則，而且該權限原則無法附加至任何其他IAM實體。

您必須先刪除服務連結角色的相關資源，才能將其刪除。這樣可以保護您的EMR無伺服器資源，因為您無法意外移除存取資源的權限。

如需支援服務連結角色之其他服務的相關資訊，請參閱 AWS 使用的服務，IAM並在服務連結角色欄中尋找具有 [是] 的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

無伺服器EMR的服務連結角色權限

EMR無伺服器會使用名為的服務連結角色，讓其能AWSServiceRoleForAmazonEMRServerless夠呼叫AWS APIs代表您。

服 AWSServiceRoleForAmazonEMRServerless 務連結角色會信任下列服務擔任該角色：

- `ops.emr-serverless.amazonaws.com`

名為的角色權限原則`AmazonEMRServerlessServiceRolePolicy`允許EMR無伺服器對指定的資源完成下列動作。

> **Note**
>
> 受管理的原則內容會變更，因此此處顯示的政策可能已過期。檢視中最多的 up-to-date 原 mazonEMRServerlessServiceRolePolicy則 A AWS Management Console.

- 動作：`ec2:CreateNetworkInterface`
- 動作：`ec2:DeleteNetworkInterface`
- 動作：`ec2:DescribeNetworkInterfaces`
- 動作：`ec2:DescribeSecurityGroups`
- 動作：`ec2:DescribeSubnets`
- 動作：`ec2:DescribeVpcs`
- 動作：`ec2:DescribeDhcpOptions`
- 動作：`ec2:DescribeRouteTables`
- 動作：`cloudwatch:PutMetricData`

以下是完整的`AmazonEMRServerlessServiceRolePolicy`政策。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2PolicyStatement",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:DescribeDhcpOptions",
                "ec2:DescribeRouteTables"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CloudWatchPolicyStatement",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:PutMetricData"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "cloudwatch:namespace": [
                        "AWS/EMRServerless",
                        "AWS/Usage"
                    ]
                }
            }
        }
    ]
}
```

下列信任原則已附加至此角色，以允許EMR無伺服器主體擔任此角色。

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ops.emr-serverless.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

您必須設定權限，才能允許IAM實體 (例如使用者、群組或角色) 建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱IAM使用指南中的服務連結角色權限。

建立無伺服器EMR的服務連結角色

您不需要手動建立一個服務連結角色。當您在 EMR AWS Management Console (使用EMR工作室), AWS CLI，或 AWS API，EMR無伺服器會為您建立服務連結角色。您必須設定權限，才能允許IAM實體 (例如使用者、群組或角色) 建立、編輯或刪除服務連結角色。

若要使用建立 AWSServiceRoleForAmazonEMRServerless 服務連結角色 IAM

針對需要建立服務連結角色的IAM實體，將下列陳述式新增至權限原則。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:CreateServiceLinkedRole"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/ops.emr-serverless.amazonaws.com/
AWSServiceRoleForAmazonEMRServerless*",
    "Condition": {"StringLike": {"iam:AWSServiceName": "ops.emr-
serverless.amazonaws.com"}}
}
```

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。當您建立新的EMR無伺服器應用程式時，EMR無伺服器會再次為您建立服務連結角色。

您也可以使用IAM主控台建立具有EMR無伺服器使用案例的服務連結角色。在 AWS CLI 或 AWS API，建立具有ops.emr-serverless.amazonaws.com服務名稱的服務連結角色。如需詳細資訊，

請參閱IAM使用指南中的建立服務連結角色。如果您刪除此服務連結角色，您可以使用此相同的程序以再次建立該角色。

編輯無伺服器EMR的服務連結角色

EMR無伺服器不允許您編輯 AWSServiceRoleForAmazonEMRServerless 服務連結角色，因為各個實體可能會參照該角色。您無法編輯 AWS EMR無伺服器服務連結角色所使用的擁有IAM原則，因為它包含無伺服器需求的所有必要權限EMR。但是，您可以使用編輯角色的描述IAM。

若要使用編輯 AWSServiceRoleForAmazonEMRServerless 服務連結角色的說明 IAM

將下列陳述式新增至需要編輯服務連結角色描述之IAM實體的權限原則。

```
{
    "Effect": "Allow",
    "Action": [
        "iam: UpdateRoleDescription"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/ops.emr-serverless.amazonaws.com/
AWSServiceRoleForAmazonEMRServerless*",
    "Condition": {"StringLike": {"iam:AWSServiceName": "ops.emr-
serverless.amazonaws.com"}}
}
```

如需詳細資訊，請參閱IAM使用指南中的編輯服務連結角色。

刪除無伺服器EMR的服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。這樣您就不會擁有未被主動監視或維護的未使用實體。不過，您必須先刪除所有區域中的所有EMR無伺服器應用程式，才能刪除服務連結角色。

> **Note**
>
> 當您嘗試刪除與該角色相關聯的資源時，如果EMR無伺服器服務正在使用該角色，則刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

若要使用刪除 AWSServiceRoleForAmazonEMRServerless 服務連結角色 IAM

將下列陳述式新增至需要刪除服務連結角色之IAM實體的權限原則。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/ops.emr-serverless.amazonaws.com/
AWSServiceRoleForAmazonEMRServerless*",
    "Condition": {"StringLike": {"iam:AWSServiceName": "ops.emr-
serverless.amazonaws.com"}}
}
```

若要使用手動刪除服務連結角色 IAM

使用IAM控制台 AWS CLI，或 AWS API以刪除 AWSServiceRoleForAmazonEMRServerless 服務連結角色。如需詳細資訊，請參閱IAM使用指南中的刪除服務連結角色。

EMR無伺服器服務連結角色的支援區域

EMR無伺服器支援在所有提供服務的區域中使用服務連結角色。如需詳細資訊，請參閱 AWS 區域和端點。

Amazon EMR 無伺服器的 Job 務執行階段角色

您可以指定代表您呼叫其他服務時，EMR無伺服器工作執行可以承擔的IAM角色權限。這包括對任何資料來源、目標以及其他資料來源的 Amazon S3 存取 AWS 像 Amazon Redshift 集群和 DynamoDB 表這樣的資源。若要進一步瞭解如何建立角色，請參閱建立工作執行時期角色。

示例運行時策略

您可以將執行時期原則 (如下所示) 附加至工作執行階段角色。下列工作執行階段原則允許：

- 使用EMR範例讀取 Amazon S3 儲存貯體的存取權限。
- 完全存取 S3 儲存貯體。
- 建立和讀取存取權 AWS Glue 資料型錄。

若要新增存取權至其他 AWS DynamoDB 等資源，您需要在建立執行階段角色時，在原則中包含這些資源的權限。

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

```
    "Statement": [
        {
            "Sid": "ReadAccessForEMRSamples",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::*.elasticmapreduce",
                "arn:aws:s3:::*.elasticmapreduce/*"
            ]
        },
        {
            "Sid": "FullAccessToS3Bucket",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        },
        {
            "Sid": "GlueCreateAndReadDataCatalog",
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase",
                "glue:CreateDatabase",
                "glue:GetDataBases",
                "glue:CreateTable",
                "glue:GetTable",
                "glue:UpdateTable",
                "glue:DeleteTable",
                "glue:GetTables",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:CreatePartition",
                "glue:BatchCreatePartition",
                "glue:GetUserDefinedFunctions"
```

```
            ],
            "Resource": ["*"]
        }
    ]
}
```

傳遞角色權限

您可以將IAM權限原則附加至使用者的角色，以允許使用者僅傳遞核准的角色。這可讓管理員控制哪些使用者可以將特定的工作執行階段角色傳遞至EMR無伺服器工作。要了解有關設置權限的更多信息，請參閱授予用戶將角色傳遞給某個角色的權限 AWS 服務。

以下是允許將工作執行階段角色傳遞給EMR無伺服器服務主體的範例原則。

```
{
     "Effect": "Allow",
     "Action": "iam:PassRole",
     "Resource": "arn:aws:iam::1234567890:role/JobRuntimeRoleForEMRServerless",
        "Condition": {
                "StringLike": {
                    "iam:PassedToService": "emr-serverless.amazonaws.com"
                }
            }
}
```

EMR無伺服器的使用者存取原則範例

您可以根據您希望每位使用者在與EMR無伺服器應用程式互動時執行的動作，為使用者設定精細的原則。下列原則範例可能有助於為您的使用者設定正確的權限。本節僅著重於EMR無伺服器原則。如需 EMR Studio 使用者原則的範例，請參閱設定 EMR Studio 使用者權限。如需有關如何將原則附加至IAM使IAM用者 (主參與者) 的資訊，請參閱《使用指南》中的〈管理IAM策略〉。

超級使用者政策

若要授與EMR無伺服器的所有必要動作，請建立AmazonEMRServerlessFullAccess原則並將其附加至所需的IAM使用者、角色或群組。

以下是範例原則，可讓進階使用者建立和修改EMR無伺服器應用程式，以及執行其他動作，例如提交和偵錯工作。它會顯示EMR無伺服器對其他服務所需的所有動作。

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

```
    "Statement": [
        {
            "Sid": "EMRServerlessActions",
            "Effect": "Allow",
            "Action": [
                "emr-serverless:CreateApplication",
                "emr-serverless:UpdateApplication",
                "emr-serverless:DeleteApplication",
                "emr-serverless:ListApplications",
                "emr-serverless:GetApplication",
                "emr-serverless:StartApplication",
                "emr-serverless:StopApplication",
                "emr-serverless:StartJobRun",
                "emr-serverless:CancelJobRun",
                "emr-serverless:ListJobRuns",
                "emr-serverless:GetJobRun"
            ],
            "Resource": "*"
        }
    ]
}
```

當您啟用與您的網路連線時VPC，EMR無伺服器應用程式會建立 Amazon EC2 彈性網路界面 (ENIs) 以與VPC資源通訊。下列原則可確保只在EMR無伺服器應用程式的內容中建立任何新項EC2ENIs目。

> **Note**
>
> 強烈建議您設定此原則，以確保EC2ENIs除啟動無EMR伺服器應用程式之外，使用者無法建立。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEC2ENICreationWithEMRTags",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
```

```
            ],
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaLast": "ops.emr-serverless.amazonaws.com"
                }
            }
        }
}
```

如果您想要限制對特定子網路的EMR無伺服器存取，可以使用標籤條件標記每個子網路。此IAM原則可確保EMR無伺服器應用程式只能EC2ENIs在允許的子網路內建立。

```
{
    "Sid": "AllowEC2ENICreationInSubnetAndSecurityGroupWithEMRTags",
    "Effect": "Allow",
    "Action": [
        "ec2:CreateNetworkInterface"
    ],
    "Resource": [
        "arn:aws:ec2:*:*:subnet/*",
        "arn:aws:ec2:*:*:security-group/*"
    ],
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/KEY": "VALUE"
        }
    }
}
```

> **Important**
>
> 如果您是建立第一個應用程式的系統管理員或進階使用者，則必須設定權限原則，以允許您建立EMR無伺服器服務連結角色。如需進一步了解，請參閱 針EMR對無伺服器使用服務連結角色。

下列IAM政策允許您為帳戶建立EMR無伺服器服務連結角色。

```
{
   "Sid":"AllowEMRServerlessServiceLinkedRoleCreation",
   "Effect":"Allow",
```

```
   "Action":"iam:CreateServiceLinkedRole",
   "Resource":"arn:aws:iam::account-id:role/aws-service-role/ops.emr-
serverless.amazonaws.com/AWSServiceRoleForAmazonEMRServerless"
}
```

數據工程師政策

以下是範例原則，允許使用者對EMR無伺服器應用程式擁有唯讀權限，以及提交和偵錯工作的能力。請切記，因為此政策並未明確拒絕動作，不同的政策陳述式仍有可能用於授予指定動作存取權。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EMRServerlessActions",
            "Effect": "Allow",
            "Action": [
                "emr-serverless:ListApplications",
                "emr-serverless:GetApplication",
                "emr-serverless:StartApplication",
                "emr-serverless:StartJobRun",
                "emr-serverless:CancelJobRun",
                "emr-serverless:ListJobRuns",
                "emr-serverless:GetJobRun"
            ],
            "Resource": "*"
        }
    ]
}
```

使用存取控制的標籤

您可以使用標籤條件進行精細的存取控制。例如，您可以限制一個團隊的使用者，使他們只能將工作提交到標記為其團隊名稱的EMR無伺服器應用程式。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EMRServerlessActions",
            "Effect": "Allow",
            "Action": [
```

```
                "emr-serverless:ListApplications",
                "emr-serverless:GetApplication",
                "emr-serverless:StartApplication",
                "emr-serverless:StartJobRun",
                "emr-serverless:CancelJobRun",
                "emr-serverless:ListJobRuns",
                "emr-serverless:GetJobRun"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Team": "team-name"
                }
            }
        }
    ]
}
```

標籤型存取控制的政策

您可以使用以身分識別為基礎的原則中的條件，根據標記控制對應用程式和工作執行的存取。

下列範例示範在EMR無伺服器條件金鑰中使用條件運算子的不同案例和方法。這些IAM原則陳述式僅用於示範目的，不應在生產環境中使用。有多種方法可以結合政策陳述式，以根據您的需求授予和拒絕許可。如需有關規劃和測試IAM政策的詳細資訊，請參閱IAM使用者指南。

> **Important**
>
> 標記動作的明確拒絕許可是項重要的考量條件。這可防止使用者標記資源並將您無意授予的許可授予給他們。如果未拒絕資源的標記動作，使用者可以修改標籤並規避標籤型政策的意圖。如需有關可拒絕標記動作的政策範例，請參閱 拒絕新增和移除標籤的存取權。

以下範例示範以身分識別為基礎的權限原則，這些原則用來控制EMR無伺服器應用程式允許的動作。

僅在具有特定標籤值的資源上允許動作

在下列原則範例中，StringEquals條件運算子會嘗試dev與標籤部門的值相符。如果標籤部門尚未新增至應用程式，或未包含值dev，則不會套用原則，且此原則不允許採取這些動作。如果沒有其他原則陳述式允許這些動作，則使用者只能使用具有此標記且具有此值的應用程式。

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "emr-serverless:GetApplication"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "emr-serverless:ResourceTag/department": "dev"
        }
      }
    }
  ]
}
```

您也可以使用條件運算子來指定多個標籤值。例如，若要允許對department標籤包含值的應用程式執行動作test，dev或者您可以使用下列方法取代前面範例中的條件區塊。

```
"Condition": {
        "StringEquals": {
          "emr-serverless:ResourceTag/department": ["dev", "test"]
        }
      }
```

建立資源時需要進行標記

在下面的例子中，創建應用程序時需要應用標籤。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "emr-serverless:CreateApplication"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "emr-serverless:RequestTag/department": "dev"
        }
```

```
      }
    }
  ]
}
```

下列原則陳述式只有在應用程式具有可包含任何值的department標籤時，才允許使用者建立應用程式。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "emr-serverless:CreateApplication"
      ],
      "Resource": "*",
      "Condition": {
        "Null": {
          "emr-serverless:RequestTag/department": "false"
        }
      }
    }
  ]
}
```

拒絕新增和移除標籤的存取權

此原則可防止使用者在EMR無伺服器應用程式上新增或移除department標籤，且標籤值不dev是。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "emr-serverless:TagResource",
        "emr-serverless:UntagResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
```

```
          "emr-serverless:ResourceTag/department": "dev"
        }
      }
    }
  ]
}
```

無伺服器的身分識別原則範例 EMR

依預設，使用者和角色沒有建立或修改 Amazon EMR 無伺服器資源的權限。他們也無法執行任務使用 AWS Management Console, AWS Command Line Interface (AWS CLI)，或 AWS API。若要授與使用者對所需資源執行動作的權限，IAM管理員可以建立IAM策略。然後，系統管理員可以將IAM原則新增至角色，使用者可以擔任這些角色。

若要瞭解如何使用這些範例原則文件來建立以IAM身分識別為基礎的JSON策略，請參閱使用指南中的IAM建立IAM策略。

如需 Amazon EMR Serverless 定義的動作和資源類型的詳細資訊，包括每種資源類型ARNs的格式，請參閱服務授權參考中適用於 Amazon EMR 無伺服器的動作、資源和條件金鑰。

主題

- 政策最佳實務
- 允許使用者檢視他們自己的許可

政策最佳實務

> **Note**
>
> EMR無伺服器不支援受管政策，因此下列第一個做法並不適用。

以身分識別為基礎的政策決定某人是否可以在您的帳戶中建立、存取或刪除 Amazon EMR 無伺服器資源。這些動作可能會為您帶來成本 AWS 帳戶。 建立或編輯以身分識別為基礎的原則時，請遵循下列準則和建議：

- 開始使用 AWS 受管原則並朝著最低權限權限移轉 — 若要開始授與使用者和工作負載的權限，請使用 AWS 授與許多常見使用案例權限的受管理策略。他們是可用的 AWS 帳戶。 建議您透過定義進一步減少權限 AWS 針對您的使用案例特定的客戶管理政策。如需詳細資訊，請參閱 AWS 受管理的策略或 AWS 《使用者指南》 中針對工作職能的IAM管理策略。

- 套用最低權限權限 — 當您使用原則設定權限時，IAM只授與執行工作所需的權限。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。如需有關使用套用權限IAM的詳細資訊，請參閱《使用指南》IAM中的IAM《策略與權限》。
- 使用IAM策略中的條件進一步限制存取 — 您可以在策略中新增條件，以限制對動作和資源的存取。例如，您可以撰寫政策條件，以指定必須使用傳送所有要求SSL。如果服務動作是透過特定使用條件，您也可以使用條件來授與對服務動作的存取權 AWS 服務，例如， AWS CloudFormation。 如需詳細資訊，請參閱《IAM使用指南》中的IAMJSON策略元素：條件。
- 使用 IAM Access Analyzer 驗證您的原IAM則，以確保安全和功能性的權限 — IAM Access Analyzer會驗證新的和現有的原則，以便原則遵循IAM原則語言 (JSON) 和IAM最佳做法。IAMAccess Analyzer 提供超過 100 項原則檢查和可行的建議，協助您撰寫安全且功能正常的原則。如需詳細資訊，請參閱IAM使IAM用指南中的存取分析器原則驗證。
- 需要多重要素驗證 (MFA) — 如果您的案例需要使IAM用者或 root 使用者 AWS 帳戶，請開啟MFA以獲得額外的安全性。若要在呼叫API作業MFA時需要，請在原則中新增MFA條件。如需詳細資訊，請參閱《IAM使用指南》中的 < 設定MFA受保護的API存取 > 。

如需有關中最佳作法的詳細資訊IAM，請參閱《IAM使用指南》IAM中的「安全性最佳作法」。

允許使用者檢視他們自己的許可

此範例顯示如何建立原則，讓使IAM用者檢視附加至其使用者身分識別的內嵌和受管理原則。此原則包含在主控台上完成此動作的權限，或以程式設計方式使用 AWS CLI 或 AWS API.

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

```
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon EMR 無伺服器更新 AWS 受管政策

檢視有關更新的詳細資訊 AWS Amazon EMR 無伺服器的受管政策，因為此服務開始追蹤這些變更。如需有關此頁面變更的自動警示，請訂閱 Amazon EMR 無伺服器文件歷史記錄頁面上的動RSS態消息。

變更	描述	日期
A mazonEMRServerless ServiceRolePolicy -更新現有策略	Amazon EMR 無服務器添加了新EC2PolicyStatement 的SidCloudWatc hPolicyStatement 和 A mazonEMRServerless ServiceRolePolicy 政策。	2024年1月25日
A mazonEMRServerless ServiceRolePolicy -更新現有策略	Amazon EMR 無伺服器新增許可，讓 Amazon EMR 無伺服器在命名空間中發佈 v CPU 使用量的彙總帳戶指標。"AWS/Usage"	2023 年 4 月 20 日

變更	描述	日期
Amazon EMR 無服務器開始跟踪更改	Amazon EMR 無服務器開始跟踪其更改 AWS 受管理的策略。	2023 年 4 月 20 日

疑難排解 Amazon EMR 無伺服器身分和存取

使用下列資訊協助您診斷和修正使用 Amazon EMR 無伺服器和時可能會遇到的常見問題。IAM

主題

- 我沒有在 Amazon EMR 無伺服器中執行動作的授權
- 我沒有授權執行 iam：PassRole
- 我想讓我以外的人 AWS 帳戶訪問我的 Amazon EMR 無服務器資源

我沒有在 Amazon EMR 無伺服器中執行動作的授權

如果 AWS Management Console 告訴您您沒有執行動作的授權，那麼您必須聯絡您的管理員尋求協助。您的管理員是提供您使用者名稱和密碼的人員。

下列範例錯誤會在 `mateojackson` 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 `emr-serverless:`*`GetWidget`* 許可時發生。

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

在此情況下，Mateo 會請求管理員更新他的政策，允許他使用 *`my-example-widget`* 動作存取 `emr-serverless:`*`GetWidget`* 資源。

我沒有授權執行 iam：PassRole

如果您收到未獲授權執行`iam:PassRole`動作的錯誤訊息，則必須更新您的政策以允許您將角色傳遞給 Amazon EMR 無伺服器。

一些 AWS 服務 可讓您將現有角色傳遞至該服務，而非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。

當名為的使用IAM者marymajor嘗試使用主控台在 Amazon EMR 無伺服器中執行動作時，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 沒有將角色傳遞至該服務的許可。

```
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 AWS 管理員。您的管理員提供您的簽署憑證。

我想讓我以外的人 AWS 帳戶訪問我的 Amazon EMR 無服務器資源

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。對於支援以資源為基礎的政策或存取控制清單 (ACLs) 的服務，您可以使用這些政策授與人員存取您的資源。

如需進一步了解，請參閱以下內容：

- 要了解 Amazon EMR 無伺服器是否支援這些功能，請參閱Amazon EMR 無伺服器中的 Identity and Access Management (IAM)。
- 了解如何在各地提供對資源的存取權 AWS 帳戶 您擁有的，請參閱為其他IAM使用者提供存取權 AWS 帳戶 您在IAM用戶指南中擁有的。
- 瞭解如何將資源存取權提供給第三方 AWS 帳戶，請參閱提供存取 AWS 帳戶 由IAM用戶指南中的第三方擁有。
- 若要瞭解如何透過身分聯盟提供存取權，請參閱使用指南中的提供對外部驗證使用IAM者的存取權 (身分聯合)。
- 若要瞭解針對跨帳號存取使用角色與以資源為基礎的政策之間的差異，請參閱《使用IAM者指南》IAM中的〈跨帳號資源存取〉。

搭配使用EMR無伺服器 AWS Lake Formation 用於精細的訪問控制

概觀

使用 Amazon 7.2.0 及更高EMR版本，您可以充分利用 AWS Lake Formation ，對 S3 支援的資料目錄資料表套用精細的存取控制。此功能可讓您設定下列項目的表格、列、欄和儲存格層級存取控制 read 在您的 Amazon EMR 無伺服器星火任務中進行查詢。若要為 Apache Spark 批次工作和互動式工作階

段設定精細的存取控制，請使用 EMR Studio。請參閱以下各節，以了解更多有關 Lake Formation 以及如何與EMR無伺服器搭配使用的資訊。

使用 Amazon EMR 無伺服器搭配 AWS Lake Formation 會產生額外費用。如需詳細資訊，請參閱 Amazon EMR 定價。

EMR無伺服器如何搭配使用 AWS Lake Formation

搭配 Lake Formation 使用EMR無伺服器，可讓您在每個 Spark 工作上強制執行一層權限，以便在EMR無伺服器執行工作時套用 Lake Formation 權限控制。EMR無伺服器會使用 Spark 資源設定檔來建立兩個設定檔，以有效執行工作。使用者設定檔會執行使用者提供的程式碼，而系統設定檔會強制執行 Lake Formation 政策。如需詳細資訊，請參閱什麼是 AWS Lake Formation以及考量和限制。

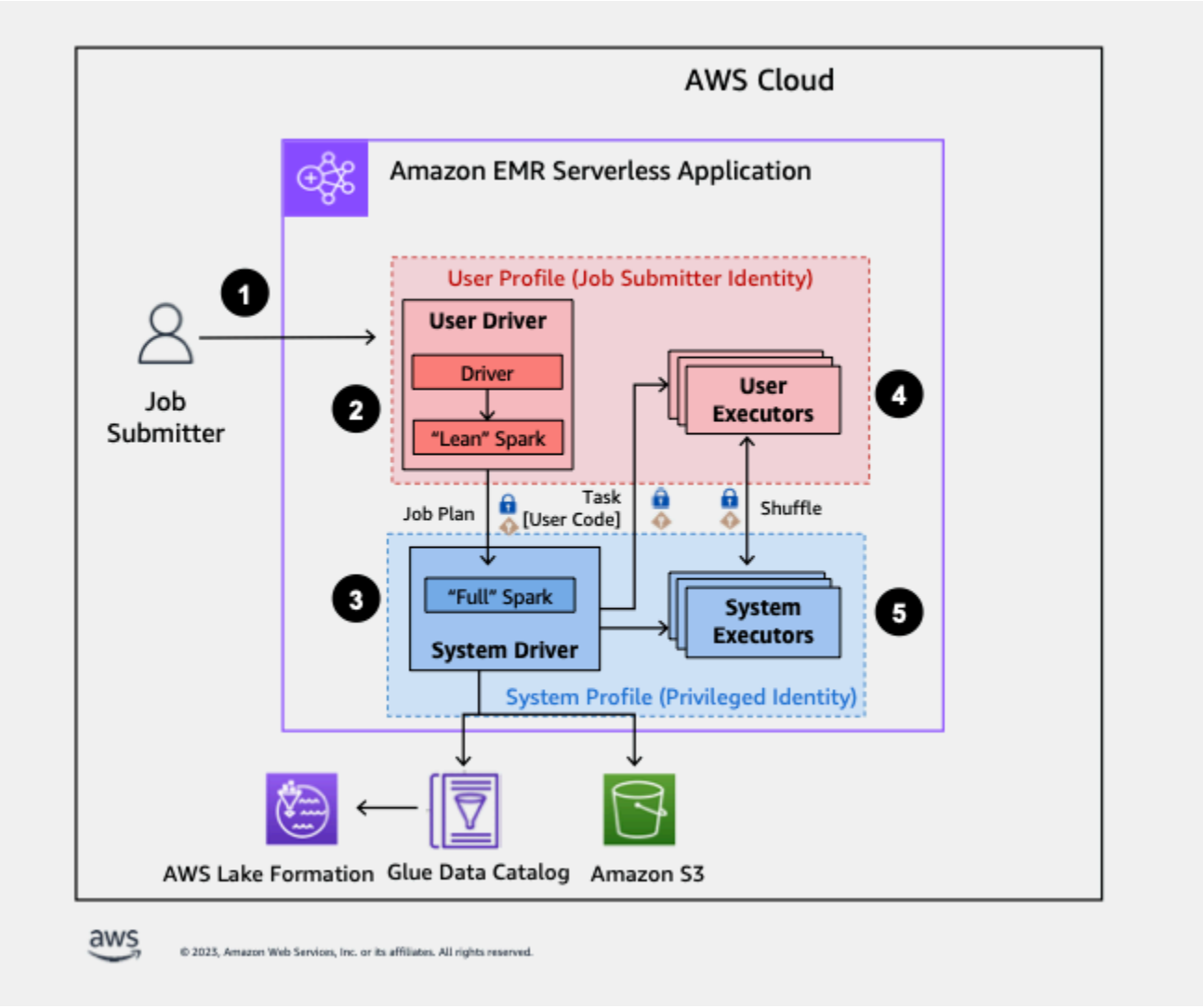
當您使用 Lake Formation 的預初始化容量時，我們建議您至少有兩個 Spark 驅動程序。每個啟用 Lake 格式化的工作都會使用兩個 Spark 驅動程式，一個用於使用者設定檔，另一個用於系統設定檔。為了獲得最佳性能，與不使用 Lake Formation 相比，對於啟用 Lake Formation 的工作，您應該使用驅動程序數量的兩倍。

當您在EMR無伺服器上執行 Spark 工作時，也必須考慮動態配置對資源管理和叢集效能的影響。每個資源配置文件`spark.dynamicAllocation.maxExecutors`的最大執行程序數量的配置適用於用戶和系統執行程序。如果您將該數字設定為等於允許的最大執行程式數目，則您的工作執行可能會因為一種使用所有可用資源的執行程式類型而卡住，這會在您執行作業工作時阻止其他執行程式。

因此，您不會耗盡資源，EMR無伺服器會將每個資源設定檔的最大執行程式數量設定為值的 `spark.dynamicAllocation.maxExecutors` 90%。當您指定介於 0 到 1 之間`spark.dynamicAllocation.maxExecutorsRatio`的值時，您可以覆寫此組態。此外，您也可以設定下列內容，以最佳化資源配置和整體效能：

- `spark.dynamicAllocation.cachedExecutorIdleTimeout`
- `spark.dynamicAllocation.shuffleTracking.timeout`
- `spark.cleaner.periodicGC.interval`

以下是EMR無伺服器如何存取受 Lake Formation 安全性原則保護之資料的高階概觀。



1. 使用者將 Spark 工作提交至 AWS Lake Formation啟用EMR無伺服器應用程式。
2. EMR無伺服器會將工作傳送至使用者驅動程式，並在使用者設定檔中執行工作。使用者驅動程式執行精簡版的 Spark，無法啟動工作、要求執行程式、存取 S3 或 Glue 目錄。它建立了一個工作計劃。
3. EMR無伺服器會設定第二個稱為系統驅動程式的驅動程式，並在系統設定檔中執行它 (具有特殊權限的身分識別)。EMR無伺服器會在兩個驅動程式之間設定加密通TLS道以進行通訊。用戶驅動程序使用通道將作業計劃發送到系統驅動程序。系統驅動程式不會執行使用者提交的程式碼。它會執行完整的 Spark 並與 S3 通訊，以及資料目錄以進行資料存取。它請求執行者和編譯 Job 計劃成執行階段的序列。
4. EMR然後，無服務器使用用戶驅動程序或系統驅動程序在執行程序上運行階段。在任何階段的用戶代碼只在用戶配置文件執行器上運行。

5. 從受保護的「資料目錄」表格讀取資料的階段 AWS Lake Formation 或者那些應用安全過濾器委託給系統執行者。

在 Amazon 中啟用 Lake Formation EMR

若要啟用 Lake Formation，您必須`true`在建立EMR無伺服器應用程式時，`spark.emr-serverless.lakeformation.enabled`將執行階段組態參數的`spark-defaults`分類設定為。

```
aws emr-serverless create-application \
    --release-label emr-7.2.0 \
    --runtime-configuration '{
     "classification": "spark-defaults",
     "properties": {
      "spark.emr-serverless.lakeformation.enabled": "true"
      }
    }' \
    --type "SPARK"
```

您也可以在 EMR Studio 中創建新的應用程序時啟用 Lake Formation。選擇使用 Lake Formation 進行精細的訪問控制，可在其他配置下使用。

當您搭配EMR無伺服器使用 Lake Formation 時，依預設會啟用 Worker 間加密，因此您不必再次明確啟用工作者間加密。

為火花工作啟用 Lake Formation

若要為個別星火工作啟用 Lake Formation，請在使用時設定`spark.emr-serverless.lakeformation.enabled`為 true `spark-submit`。

```
--conf spark.emr-serverless.lakeformation.enabled=true
```

Job 執行期角色IAM權限

Lake Formation 權限控制訪問 AWS Glue 資料型錄資源、Amazon S3 位置以及這些位置的基礎資料。IAM權限控制訪問 Lake Formation 和 AWS Glue APIs 和資源。雖然您可能擁有 Lake Formation 權限來存取資料目錄 (SELECT) 中的資料表，但是如果您沒有該作業的IAM權限，您的`glue:Get*`API作業就會失敗。

以下是如何提供存取 S3 中指令碼、將日誌上傳到 S3 的IAM權限的範例政策， AWS Glue API 權限，以及訪問 Lake Formation 的許可權。

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ScriptAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
            "arn:aws:s3:::*.DOC-EXAMPLE-BUCKET/scripts",
            "arn:aws:s3:::*.DOC-EXAMPLE-BUCKET/*" ]
        },
        {
         "Sid": "LoggingAccess",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
               "arn:aws:s3:::DOC-EXAMPLE-BUCKET/logs/*"
            ]
        },
        {
            "Sid": "GlueCatalogAccess",
            "Effect": "Allow",
            "Action": [
                 "glue:Get*",
                 "glue:Create*",
                 "glue:Update*"
            ],
            "Resource": ["*"]
        },
        {
            "Sid": "LakeFormationAccess",
            "Effect": "Allow",
            "Action": [
             "lakeformation:GetDataAccess"
             ],
            "Resource": ["*"]
        }
    ]
```

```
}
```

為工作執行階段角色設定 Lake Formation 權限

首先，使用 Lake Formation 註冊您的 Hive 桌子的位置。然後在您想要的資料表上為您的工作執行階段角色建立權限。有關 Lake Formation 的更多詳細信息，請參閱什麼是 AWS Lake Formation? 在 AWS Lake Formation 開發人員指南。

設定 Lake Formation 許可後，您可以在 Amazon EMR 無伺服器上提交 Spark 任務。如需 Spark 工作的詳細資訊，請參閱 Spark 範例。

提交工作執行

在您完 Lake Formation 補助的設定後，您可以在EMR無伺服器上提交 Spark 工作。若要執行 Iceberg 工作，您必須提供下列`spark-submit`屬性。

```
--conf spark.sql.catalog.spark_catalog=org.apache.iceberg.spark.SparkSessionCatalog
--conf spark.sql.catalog.spark_catalog.warehouse=<S3_DATA_LOCATION>
--conf spark.sql.catalog.spark_catalog.glue.account-id=<ACCOUNT_ID>
--conf spark.sql.catalog.spark_catalog.client.region=<REGION>
--conf spark.sql.catalog.spark_catalog.glue.endpoint=https://
glue.<REGION>.amazonaws.com
```

開放式表格格式支援

Amazon EMR 版本 7.2.0 包含基於 Lake Formation 的精細存取控制支援。EMR無伺服器支援 Hive 和冰山資料表類型。下表說明所有支援的作業。

作業	Hive	Iceberg
DDL命令	僅具有IAM角色權限	僅具有IAM角色權限
增量查詢	不適用	完全支援
時間歷程查詢	不適用於此表格格式	完全支援
中繼資料表	不適用於此表格格式	支援，但某些表格會隱藏。如需詳細資訊，請參閱考量與限制。

作業	Hive	Iceberg
DML INSERT	僅IAM限使用權限	僅IAM限使用權限
DML UPDATE	不適用於此表格格式	僅IAM限使用權限
DML DELETE	不適用於此表格格式	僅IAM限使用權限
讀取操作	完全支援	完全支援
預存程序	不適用	除了register_table 和之外的支援migrate。如需詳細資訊，請參閱考量與限制。

考量與限制

將 Lake Formation 與EMR無伺服器搭配使用時，請考慮下列考量和限制。

> **Note**
>
> 當您在EMR無伺服器上為 Spark 工作啟用 Lake Formation 時，工作會啟動系統驅動程式和使用者驅動程式。如果您在啟動時指定了預先初始化的容量，則驅動程式會從預先初始化的容量佈建，以及系統驅動程式數目等於您指定的使用者驅動程式數目。如果您選擇隨選容量，則除了使用者驅動程式之外，EMR無伺服器還會啟動系統驅動程式。若要估算與您的EMR無伺服器與湖泊陣型工作相關的成本，請使用 AWS Pricing Calculator.

所有受支援的EMR無伺服器區域皆提供 Amazon EMR無伺服器搭配 Lake Formation，但除外 AWS GovCloud (美國東部) 及 AWS GovCloud (美國西部).

- Amazon EMR 無伺服器僅針對 Apache Hive 和 Apache 冰山表，透過 Lake Formation 支援精細的存取控制。阿帕奇蜂巢格式包括鑲木地板ORC，和 XSV。
- 啟用 Lake 格式化的應用程式不支援使用自訂的無EMR伺服器映像。
- 你不能關閉 Lake Formation DynamicResourceAllocation 的工作。
- 你只能在火花工作中使用 Lake Formation。
- EMR含 Lake Formation 的無伺服器在整個工作中僅支援單一 Spark 工作階段。
- EMR含 Lake Formation 的無伺服器僅支援透過資源連結共用的跨帳戶表格查詢。

- 不支援下列項目：
  - 彈性分散式資料集 (RDD)
  - 火花流
  - 使用授予權限的 Lake Formation 編寫
  - 巢狀資料欄的存取控制
- EMR無伺服器會封鎖可能會破壞系統驅動程式完全隔離的功能，包括：
  - UDTsiveUDFs、H 和任何涉及自訂類別的使用者定義函數
  - 自訂資料來源
  - 供應額外的罐子用於星火擴展，連接器，或元存儲
  - ANALYZE TABLE 命令
- 為了強制執行訪問控制，以EXPLAIN PLAN及DDL諸如DESCRIBE TABLE不公開受限制信息之類的操作。
- EMR無伺服器限制對啟用 Lake 格式化之應用程式上系統驅動程式 Spark 記錄的存取。由於系統驅動程式以較多的存取權執行，因此系統驅動程式產生的事件和記錄檔可能包含敏感資訊。為了防止未經授權的使用者或程式碼存取此敏感資料，EMR無伺服器停用對系統驅動程式記錄的存取。如需疑難排解，請聯 AWS 支持。
- 如果您已使用 Lake Formation 註冊資料表位置，則無論「無EMR伺服器」工作執行階段角色的IAM權限為何，資料存取路徑都會經過 Lake Formation 儲存認證。如果您錯誤設定了使用資料表位置註冊的角色，提交使用具有 S3 IAM 權限的角色的工作將會失敗。
- 寫入 Lake Formation 表使用IAM許可，而不是授予權限的 Lake Formation。如果您的任務執行階段角色具有必要的 S3 許可，您可以使用它來執行寫入操作。

以下是使用 Apache 冰山時的注意事項和限制：

- 您只能使用 Apache 冰山與會話目錄，而不是任意命名的目錄。
- 在 Lake Formation 中註冊的冰山表僅支援中繼資料表historymetadata_log_entriessnapshots、files、manifests、和refs。Amazon EMR 隱藏了可能包含敏感數據的列partitions，例如path，和summaries。此限制不適用於尚未在 Lake Formation 中註冊的冰山桌。
- 您未在 Lake Formation 中註冊的表支持所有 Iceberg 存儲過程。register_table和程migrate序不支援任何資料表。
- 我們建議您使用冰山 DataFrameWriter V2 而不是 V1。

故障診斷

如需疑難排解解決方案，請參閱下列章

日誌

EMR無伺服器會使用 Spark 資源設定檔來分割工作執行。EMR無伺服器會使用使用者設定檔來執行您提供的程式碼，而系統設定檔會強制執行 Lake Formation 政策。您可以訪問以用戶配置文件身份運行的任務的日誌。

直播 UI 和星火歷史服務器

Live UI 和 Spark 歷史記錄服務器具有從用戶配置文件生成的所有 Spark 事件，並編輯了從系統驅動程序生成的事件。

您可以在 Executors 選項卡中查看來自用戶和系統驅動程序的所有任務。不過，記錄連結僅適用於使用者設定檔。此外，部分資訊會從 Live UI 編輯，例如輸出記錄的數目。

Job 失敗，Lake Formation 權限不足

請確定您的工作執行階段角色具有執行權限以SELECT及您正DESCRIBE在存取的資料表上。

RDD執行失敗的 Job

EMR無伺服器目前不支援啟用 Lake 格式化工作的彈性分散式資料集 (RDD) 作業。

無法訪問 Amazon S3 中的數據文件

確保您已經在湖泊形成中註冊了數據湖的位置。

安全驗證例外

EMR無伺服器偵測到安全驗證錯誤。聯絡 AWS 支持援助。

共享 AWS 跨帳戶的 Glue 資料目錄和資料表

您可以跨帳戶共享數據庫和表，並仍然使用 Lake Formation。有關更多信息，請參閱 Lake Formation 中的跨帳戶數據共享和如何共享 AWS Glue 資料目錄和表格跨帳戶使用 AWS Lake Formation?.

工作者間加密

使用 Amazon 6.15.0 及更高EMR版本，您可以在 Spark 任務執行中啟用員工之間的相互TLS加密通訊。啟用時，EMR無伺服器會自動為您的工作執行中佈建的每個 Worker 產生並分配唯一的憑證。當這

些 Worker 與交換控制郵件通訊或傳輸隨機資料時，他們會建立相互TLS連線，並使用設定的憑證來驗證彼此的身分識別。如果 Worker 無法驗證其他憑證，TLS交換會失敗，且EMR無伺服器會中止它們之間的連線。

如果您將 Lake Formation m 與EMR無伺服器搭配使用，則預設會啟用相互TLS加密。

在EMR無伺服器上啟用相互TLS加密

若要在您的 spark 應用程式上啟用相互TLS加密，請在建立EMR無伺服器應用程式時設定`spark.ssl.internode.enabled`為 true。如果您正在使用 AWS 主控台若要建立EMR無伺服器應用程式，請選擇 [使用自訂設定]，然後展開 [應用程式設定]，然後輸入 `runtimeConfiguration`

```
aws emr-serverless create-application \
--release-label emr-6.15.0 \
--runtime-configuration '{
  "classification": "spark-defaults",
  "properties": {"spark.ssl.internode.enabled": "true"}
}' \
--type "SPARK"
```

如果您想要為個別的 spark 工作執行啟用相互TLS加密，請在使用時設定`spark.ssl.internode.enabled`為 true `spark-submit`。

```
--conf spark.ssl.internode.enabled=true
```

使用EMR無伺服器進行資料保護的秘密管理

AWS Secrets Manager 是一種秘密儲存體服務，可用來保護資料庫認證、API金鑰和其他秘密資訊。然後在您的程式碼中，您可以透過API呼叫 Secrets Manager 來取代硬式編碼認證。這有助於確保密碼不會被某人檢查您的代碼而入侵，因為秘密不存在。如需概觀，請參閱 AWS Secrets Manager 用戶指南。

Secrets Manager 使用加密密碼 AWS Key Management Service 鑰匙。如需詳細資訊，請參閱 AWS Secrets Manager 用戶指南。

您可以將 Secrets Manager 設定為根據您指定的排程自動輪替您的密碼。這可讓您以短期秘密取代長期秘密，有助於大幅降低洩漏風險。如需詳細資訊，請參閱旋轉 AWS Secrets Manager在秘密 AWS Secrets Manager 用戶指南。

Amazon EMR 無伺服器與 AWS Secrets Manager 這樣您就可以將數據存儲在 Secrets Manager 中，並在配置中使用秘密 ID。

EMR無伺服器如何使用機密

當您將資料儲存在 Secrets Manager 中，並在EMR無伺服器的組態中使用秘密 ID 時，您不會以純文字格式將敏感設定資料傳遞給EMR無伺服器，而是將其公開給外部。APIs如果您指出索引鍵值配對包含您儲存在 Secrets Manager 中之密碼的秘密識別碼，則EMR無伺服器會在將組態資料傳送至 Worker 以執行工作時擷取該密碼。

若要指出組態的索引鍵值配對包含秘密儲存在 Secrets Manager 中的參照，請將`EMR.secret@`註釋新增至組態值。對於任何具有秘密 ID 註釋的組態屬性，EMR無伺服器會呼叫 Secret Manager，並在工作執行時解析密碼。

如何建立密碼

若要建立密碼，請依照建立密碼中的步驟操作 AWS Secrets Manager 秘密中的 AWS Secrets Manager 用戶指南。在步驟 3 中，選擇「明文」欄位以輸入您的敏感值。

在組態分類中提供密碼

下列範例顯示如何在的組態分類中提供密碼`StartJobRun`。如果您想要在應用程式層級設定 Secrets Manager 的分類，請參閱EMR無伺服器的預設應用程式組態。

在範例中，取代*`SecretName`*為要擷取的密碼名稱。加入連字號，後面接著 Secrets Manager 在密碼ARN結尾加入的六個字元。如需詳細資訊，請參閱如何建立密碼。

本節內容

- 指定秘密引用-星火
- 指定秘密引用-蜂巢

指定秘密引用-星火

Example — 在 Spark 的外部 Hive 中繼存儲配置中指定秘密引用

```
aws emr-serverless start-job-run \
  --application-id "application-id" \
  --execution-role-arn "job-role-arn" \
```

```
  --job-driver '{
        "sparkSubmit": {
            "entryPoint": "s3://DOC-EXAMPLE-BUCKET/scripts/spark-jdbc.py",
            "sparkSubmitParameters": "--jars s3://DOC-EXAMPLE-BUCKET/mariadb-connector-
java.jar
            --conf
 spark.hadoop.javax.jdo.option.ConnectionDriverName=org.mariadb.jdbc.Driver
            --conf spark.hadoop.javax.jdo.option.ConnectionUserName=connection-user-
name
            --conf
 spark.hadoop.javax.jdo.option.ConnectionPassword=EMR.secret@SecretName
            --conf spark.hadoop.javax.jdo.option.ConnectionURL=jdbc:mysql://db-host:db-
port/db-name
            --conf spark.driver.cores=2
            --conf spark.executor.memory=10G
            --conf spark.driver.memory=6G
            --conf spark.executor.cores=4"
        }
    }' \
    --configuration-overrides '{
        "monitoringConfiguration": {
        "s3MonitoringConfiguration": {
            "logUri": "s3://DOC-EXAMPLE-BUCKET/spark/logs/"
        }
    }
}'
```

Example — 指定分類中外部 Hive 中繼存放區組態的**spark-defaults**秘密參照

```
{
        "classification": "spark-defaults",
        "properties": {

 "spark.hadoop.javax.jdo.option.ConnectionDriverName":"org.mariadb.jdbc.Driver"
            "spark.hadoop.javax.jdo.option.ConnectionURL":"jdbc:mysql://db-host:db-
port/db-name"
            "spark.hadoop.javax.jdo.option.ConnectionUserName":"connection-user-name"
            "spark.hadoop.javax.jdo.option.ConnectionPassword":
 "EMR.secret@SecretName",
        }
    }
```

指定秘密引用-蜂巢

Example — 在 Hive 的外部 Hive 中繼存儲配置中指定秘密引用

```
aws emr-serverless start-job-run \
  --application-id "application-id" \
  --execution-role-arn "job-role-arn" \
    --job-driver '{
        "hive": {
        "query": "s3://DOC-EXAMPLE-BUCKET/emr-serverless-hive/query/hive-query.ql",
        "parameters": "--hiveconf hive.exec.scratchdir=s3://DOC-EXAMPLE-BUCKET/emr-
serverless-hive/hive/scratch
                    --hiveconf hive.metastore.warehouse.dir=s3://DOC-EXAMPLE-BUCKET/
emr-serverless-hive/hive/warehouse
                    --hiveconf javax.jdo.option.ConnectionUserName=username
                    --hiveconf
 javax.jdo.option.ConnectionPassword=EMR.secret@SecretName
                    --hiveconf
 hive.metastore.client.factory.class=org.apache.hadoop.hive.ql.metadata.SessionHiveMetaStoreCli
                    --hiveconf
 javax.jdo.option.ConnectionDriverName=org.mariadb.jdbc.Driver
                    --hiveconf javax.jdo.option.ConnectionURL=jdbc:mysql://db-host:db-
port/db-name"
        }
    }' \
    --configuration-overrides '{
        "monitoringConfiguration": {
        "s3MonitoringConfiguration": {
            "logUri": "s3://EXAMPLE-LOG-BUCKET"
        }
    }
}'
```

Example — 指定分類中外部 Hive 中繼存放區組態的**hive-site**秘密參照

```
{
    "classification": "hive-site",
    "properties": {
        "hive.metastore.client.factory.class":
 "org.apache.hadoop.hive.ql.metadata.SessionHiveMetaStoreClientFactory",
        "javax.jdo.option.ConnectionDriverName": "org.mariadb.jdbc.Driver",
        "javax.jdo.option.ConnectionURL": "jdbc:mysql://db-host:db-port/db-name",
        "javax.jdo.option.ConnectionUserName": "username",
```

```
        "javax.jdo.option.ConnectionPassword": "EMR.secret@SecretName"
    }
}
```

授予EMR無伺服器存取權以擷取密碼

若要允許EMR無伺服器從 Secrets Manager 擷取密碼值，請在您建立密碼時將下列原則陳述式新增至您的密碼。您必須使用客戶管理的KMS金鑰建立密碼，EMR無伺服器才能讀取密碼值。如需詳細資訊，請參閱 KMS AWS Secrets Manager 用戶指南。

在下列原則中，請以應用程式的 ID 取*applicationId*代。

密碼的資源策略

您必須在密碼的資源策略中包含以下權限 AWS Secrets Manager 允許EMR無伺服器擷取密碼值。若要確保只有特定應用程式可擷取此密碼，您可以選擇性地將EMR無伺服器應用程式 ID 指定為原則中的條件。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue",
        "secretsmanager:DescribeSecret"
      ],
      "Principal": {
        "Service": [
          "emr-serverless.amazonaws.com"
        ]
      },
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceArn": "arn:aws:emr-serverless:AWS ##:aws_account_id:/
applications/applicationId"
        }
      }
    }
```

```
  ]
}
```

根據客戶管理的下列政策建立您的密碼 AWS Key Management Service (AWS KMS) 鍵：

客戶管理的政策 AWS KMS 鍵

```
{
    "Sid": "Allow EMR Serverless to use the key for decrypting secrets",
    "Effect": "Allow",
    "Principal": {
        "Service": [
            "emr-serverless.amazonaws.com"
        ]
    },
    "Action": [
        "kms:Decrypt",
        "kms:DescribeKey"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "kms:ViaService": "secretsmanager.AWS ##.amazonaws.com"
        }
    }
}
```

旋轉的秘密

輪換是指您定期更新密碼的時候。您可以配置 AWS Secrets Manager ，以根據您指定的明細表自動旋轉密碼。這樣，您可以用短期的秘密替換長期秘密。這有助於降低受到妥協的風險。EMR當工作轉換為執行中狀態時，無伺服器會從已註解的組態擷取密碼值。如果您或程序更新 Secrets Manager 中的密碼值，您必須提交新工作，以便工作可以擷取更新的值。

> **Note**
>
> 已處於執行中狀態的工作無法擷取更新的密碼值。這可能會導致工作失敗。

搭配EMR無伺服器使用 Amazon S3 存取授權

EMR無伺服器的 S3 存取授權概觀

Amazon S3 存取授權使用 Amazon 6.15.0 及更高EMR版本時，提供可擴展的存取控制解決方案，您可以使用這些解決方案增強對無伺服器 Amazon S3 資料的存取。EMR如果您的 S3 資料有複雜或大型的許可組態，您可以使用 Access Grants 來擴展使用者、角色和應用程式的 S3 資料許可。

使用 S3 存取授權，擴大對 Amazon S3 資料的存取權限，超越執行時期IAM角色授予的許可或附加至具有EMR無伺服器應用程式存取權的身分的角色。

如需詳細資訊，請參閱 Amazon 管理指南中的使用適用於 Amazon EMR 的 S3 存取授權EMR管理存取和 Amazon 簡單儲存服務使用者指南中的使用 S3 存取授權管理存取。

本節說明如何啟動使用 S3 存取授與提供 Amazon S3 中資料存取權的EMR無伺服器應用程式。如需將 S3 存取授與與其他 Amazon EMR 部署搭配使用的步驟，請參閱下列文件：

- 透過 Amazon 使用 S3 存取授權 EMR
- EMR在 Amazon 上使用 S3 訪問授權 EKS

使用 S3 存取授權啟動EMR無伺服器應用程式以進行資料管理

您可以在EMR無伺服器上啟用 S3 存取授權並啟動 Spark 應用程式。當您的應用程式提出 S3 資料請求時，Amazon S3 會提供僅限於特定儲存貯體、字首或物件的臨時憑證。

1. 為EMR無伺服器應用程式設定工作執行角色。包括執行 Spark 任務和使用 S3 存取授IAM權所需的必要許可，以`s3:GetDataAccess`及`s3:GetAccessGrantsInstanceForPrefix`：

```
{
    "Effect": "Allow",
    "Action": [
    "s3:GetDataAccess",
    "s3:GetAccessGrantsInstanceForPrefix"
    ],
    "Resource": [     //LIST ALL INSTANCE ARNS THAT THE ROLE IS ALLOWED TO QUERY
         "arn:aws_partition:s3:Region:account-id1:access-grants/default",
         "arn:aws_partition:s3:Region:account-id2:access-grants/default"
    ]
}
```

> **Note**
>
> 如果您指定具有其他權限直接存取 S3 的任務執行IAM角色，則使用者將能夠存取角色允許的資料，即使他們沒有 S3 Access Grants 的許可。

2. 使用 6.15.0 或更高版本的 Amazon EMR 版本標籤和`spark-defaults`分類來啟動EMR無伺服器應用程式，如下列範例所示。使用適合您使用案例的值取代 *`red text`* 中的值。

```
aws emr-serverless start-job-run \
  --application-id application-id \
  --execution-role-arn job-role-arn \
  --job-driver '{
        "sparkSubmit": {
            "entryPoint": "s3://us-east-1.elasticmapreduce/emr-containers/samples/
wordcount/scripts/wordcount.py",
            "entryPointArguments": ["s3://DOC-EXAMPLE-BUCKET-OUTPUT/
wordcount_output"],
            "sparkSubmitParameters": "--conf spark.executor.cores=1 --conf
 spark.executor.memory=4g --conf spark.driver.cores=1 --conf spark.driver.memory=4g
 --conf spark.executor.instances=1"
        }
    }' \
  --configuration-overrides '{
    "applicationConfiguration": [{
        "classification": "spark-defaults",
        "properties": {
          "spark.hadoop.fs.s3.s3AccessGrants.enabled": "true",
          "spark.hadoop.fs.s3.s3AccessGrants.fallbackToIAM": "false"
         }
      }]
}'
```

S3 存取授與EMR無伺服器的考量

如需將 Amazon S3 存取授權與EMR無伺服器搭配使用時的重要支援、相容性和行為資訊，請參閱 Amazon EMR 管理指南EMR中的 S3 存取授與考量事項。

使用記錄 Amazon EMR 無伺服器API呼叫 AWS CloudTrail

Amazon EMR 無伺服器已與 AWS CloudTrail，提供使用者、角色或使用者所採取之動作記錄的服務 AWS EMR無伺服器中的服務。 CloudTrail 將EMR無伺服器的所有API呼叫擷取為事件。擷取的呼叫包括來自EMR無伺服器主控台的呼叫，以及對無伺EMR服API器作業的程式碼呼叫。如果您建立追蹤，您可以啟用持續交付 CloudTrail事件到 Amazon S3 儲存貯體，包括EMR無伺服器事件。如果您未設定追蹤，您仍然可以在 [事件歷程記錄] 中檢視 CloudTrail 主控台中最近的事件。使用收集的資訊 CloudTrail，您可以判斷向EMR無伺服器提出的要求、提出要求的 IP 位址、提出要求的人員、提出要求的時間，以及其他詳細資訊。

若要進一步了解 CloudTrail，請參閱 AWS CloudTrail 用戶指南。

EMR無伺服器資訊 CloudTrail

CloudTrail 已在您的 AWS 帳戶 當您創建帳戶時。當活動在EMR無伺服器中發生時，該活動會與其他活動一起記錄在 CloudTrail 事件中 AWS 事件歷史記錄中的服務事件。您可以查看，搜索和下載最近的事件 AWS 帳戶。 如需詳細資訊，請參閱使用 CloudTrail 事件歷程記錄檢視事件。

在您的事件的持續記錄 AWS 帳戶，包括EMR無伺服器的事件，建立追蹤。追蹤可 CloudTrail 將日誌檔交付到 Amazon S3 儲存貯體。根據預設，當您在主控台中建立追蹤時，追蹤會套用至所有 AWS 區域。 追蹤記錄中所有區域的事件 AWS 對日誌檔進行分割，並將其交付到您指定的 Amazon S3 儲存貯體。此外，您可以配置其他 AWS 進一步分析 CloudTrail 日誌中收集的事件數據並採取行動的服務。如需詳細資訊，請參閱下列內容：

- 建立追蹤的概觀
- CloudTrail 支援的服務與整合
- 設定 Amazon SNS 通知 CloudTrail
- 從多個區域接收 CloudTrail 日誌文件並從多個帳戶接收 CloudTrail 日誌文件

所有EMR無伺服器動作均由記錄， CloudTrail 並將其記錄在EMR無伺服器API參考中。例如，呼叫`StartJobRun`和`CancelJobRun`動作會`CreateApplication`在 CloudTrail 記錄檔中產生項目。

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：

- 是否要求是使用根或 AWS Identity and Access Management (IAM) 使用者認證。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 請求是否由另一個人提出 AWS 服務。

如需詳細資訊，請參閱CloudTrail userIdentity 元素。

瞭解EMR無伺服器記錄檔項目

追蹤是一種組態，可讓事件以日誌檔的形式傳遞到您指定的 Amazon S3 儲存貯體。 CloudTrail 記錄檔包含一或多個記錄項目。事件代表來自任何來源的單一請求，包括有關請求的操作，動作的日期和時間，請求參數等信息。 CloudTrail 日誌文件不是公共API調用的有序堆棧跟踪，因此它們不會以任何特定順序顯示。

下列範例顯示示範`CreateApplication`動作的 CloudTrail 記錄項目。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:admin",
        "arn": "arn:aws:sts::012345678910:assumed-role/Admin/admin",
        "accountId": "012345678910",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::012345678910:role/Admin",
                "accountId": "012345678910",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-06-01T23:46:52Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-06-01T23:49:28Z",
    "eventSource": "emr-serverless.amazonaws.com",
    "eventName": "CreateApplication",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "203.0.113.0",
    "userAgent": "PostmanRuntime/7.26.10",
    "requestParameters": {
        "name": "my-serverless-application",
        "releaseLabel": "emr-6.6",
```

```
        "type": "SPARK",
        "clientToken": "0a1b234c-de56-7890-1234-567890123456"
    },
    "responseElements": {
        "name": "my-serverless-application",
        "applicationId": "1234567890abcdef0",
        "arn": "arn:aws:emr-serverless:us-west-2:555555555555:/
applications/1234567890abcdef0"
    },
    "requestID": "890b8639-e51f-11e7-b038-EXAMPLE",
    "eventID": "874f89fa-70fc-4798-bc00-EXAMPLE",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "012345678910",
    "eventCategory": "Management"
}
```


適用於 Amazon EMR 無伺服器的合規驗證

第三方稽核員將EMR無伺服器的安全性與合規性評估為多個稽核人員的一部分 AWS 合規計劃，包括以下內容：

- 系統與組織控制 (SOC)
- 支付卡產業資料安全標準 (PCIDSS)
- 聯邦風險和授權管理計劃（FedRAMP）溫和
- Health 保險可攜性與責任法案 (HIPAA)

AWS 提供經常更新的清單 AWS 特定合規計劃範圍內的服務 AWS 合規計劃範圍內的服務。

第三方稽核報告可供您使用下載 AWS Artifact。 如需詳細資訊，請參閱下載報告 AWS Artifact 。

如需關於 AWS 合規方案，請參閱 AWS 合規計劃。

使用EMR無伺服器時，您的合規責任取決於資料的敏感度、組織的合規目標以及適用的法律和法規。如果您對EMR無伺服器的使用受到符合標準HIPAA，例如PCI，或 Fed RAMP 中等標準， AWS 提供資源以幫助：

- 安全性與合規性快速入門指南，討論以安全性和法規遵循為重點的基準環境部署的架構考量和步驟 AWS.

- AWS 客戶法規遵循指南可協助您透過合規的角度瞭解共同的責任模式。這些指南總結了安全性的最佳做法 AWS 服務 並在多個架構 (包括美國國家標準與技術研究所 (NIST)、支付卡產業安全標準委員會 (PCI) 和國際標準化組織 ()) 中對應安全控制指引。ISO
- AWS Config 可用來評定資源組態與內部實務、業界準則和法規的合規狀態。
- AWS 合規性資源是可能適用於您的產業和位置的工作簿和指南集合。
- AWS Security Hub 為您提供內部安全狀態的全面檢視 AWS 並協助您檢查您是否符合安全性產業標準和最佳做法。
- AWS Audit Manager— 這個 AWS 服務 協助您持續稽核 AWS 使用方式可簡化您管理風險以及遵守法規和業界標準的方式。

Amazon EMR 無伺服器的彈性

所以此 AWS 全球基礎設施是圍繞 AWS 區域和可用區域。 AWS 區域提供多個分開且隔離的實際可用區域，並以低延遲、高輸送量和高度備援聯網功能相互連結。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。

如需關於 AWS 區域和可用區域，請參閱 AWS 全球基礎設施。

除了 AWS 全球基礎設施，Amazon EMR 無伺服器提供與 Amazon S3 的整合，EMRFS以協助支援您的資料彈性和備份需求。

Amazon EMR 無伺服器中的基礎設施安全

作為託管服務，Amazon EMR 受到保護 AWS 全球網絡安全。如需相關資訊 AWS 安全服務以及如何 AWS 保護基礎架構，請參 AWS 雲端安全。若要設計您的 AWS 使用基礎架構安全性最佳做法的環境，請參閱安全性支柱中的基礎架構 AWS 架構良好的框架。

您使用 AWS 通過網絡訪問 Amazon EMR 的已發布API呼叫。使用者端必須支援下列專案：

- 傳輸層安全性 (TLS)。我們需要 TLS 1.2 並推薦 TLS 1.3。
- 具有完美前向保密（ ）的密碼套件，例如（短暫的迪菲-赫爾曼PFS）或DHE（橢圓曲線短暫迪菲-赫爾曼）。ECDHE現代系統(如 Java 7 和更新版本)大多會支援這些模式。

此外，請求必須使用存取金鑰 ID 和與IAM主體相關聯的秘密存取金鑰來簽署。或者您可以使用 AWS Security Token Service (AWS STS)，以產生用來簽署要求的臨時安全登入資料。

Amazon EMR 無伺服器中的組態和漏洞分析

AWS 處理基本安全性工作，例如客體作業系統 (OS) 和資料庫修補、防火牆組態和嚴重損壞修復。這些程序已由適當的第三方進行檢閱並認證。如需詳細資訊，請參閱以下 資源：

- 適用於 Amazon EMR 無伺服器的合規驗證
- 共同的責任模型
- Amazon Web Services：安全程序概觀 (白皮書)

的端點和配額 EMR Serverless

服務端點

若要以程式設計方式連接 AWS 服務，您使用端點。端點是URL一個進入點的 AWS 網絡服務。除了標準 AWS 端點, 一些 AWS 服務 在選取的區域中提供FIPS端點。下表列出EMR無伺服器的服務端點。如需詳細資訊，請參閱 AWS 服務 端點。

區域名稱	區域	端點	通訊協定
美國東部 (俄亥俄)	us-east-2 (限於下列可用區域：use2-az1use2-az2、和use2-az3)	emr-serverless.us-east-2.amazonaws.com	HTTPS
美國東部 (維吉尼亞北部)	us-east-1 (僅限於下列可用區域：use1-az1use1-az2、use1-az4、use1-az5、和use1-az6)	emr-serverless.us-east-1.amazonaws.com emr-serverless-fips.us-east-1.amazonaws.com	HTTPS
美國西部 (加利佛尼亞北部)	us-west-1	emr-serverless.us-west-1.amazonaws.com	HTTPS
美國西部 (奧勒岡)	us-west-2	emr-serverless.us-west-2.amazonaws.com emr-serverless-fips.us-west	HTTPS

區域名稱	區域	端點	通訊協定
		-2.amazonaws.com	
非洲 (開普敦)	af-south-1	emr-serverless.af-south-1.amazonaws.com	HTTPS
Asia Pacific (Hong Kong)	ap-east-1	emr-serverless.ap-east-1.amazonaws.com	HTTPS
亞太區域 (雅加達)	ap-southeast-3	emr-serverless.ap-southeast-3.amazonaws.com	HTTPS
亞太區域 (孟買)	ap-south-1	emr-serverless.ap-south-1.amazonaws.com	HTTPS
亞太區域 (大阪)	ap-northeast-3	emr-serverless.ap-northeast-3.amazonaws.com	HTTPS
亞太區域 (首爾)	ap-northeast-2	emr-serverless.ap-northeast-2.amazonaws.com	HTTPS

區域名稱	區域	端點	通訊協定
亞太區域 (新加坡)	ap-southeast-1	emr-serverless.ap-southeast-1.amazonaws.com	HTTPS
亞太區域 (雪梨)	ap-southeast-2	emr-serverless.ap-southeast-2.amazonaws.com	HTTPS
亞太區域 (東京)	ap-northeast-1	emr-serverless.ap-northeast-1.amazonaws.com	HTTPS
加拿大 (中部)	ca-central-1 (限於下列可用區域：cac1-az1和cac1-az2)	emr-serverless.ca-central-1.amazonaws.com	HTTPS
歐洲 (法蘭克福)	eu-central-1	emr-serverless.eu-central-1.amazonaws.com	HTTPS
歐洲 (愛爾蘭)	eu-west-1	emr-serverless.eu-west-1.amazonaws.com	HTTPS

區域名稱	區域	端點	通訊協定
歐洲 (倫敦)	eu-west-2	emr-serverless.eu-west-2.amazonaws.com	HTTPS
歐洲 (米蘭)	eu-south-1	emr-serverless.eu-south-1.amazonaws.com	HTTPS
Europe (Paris)	eu-west-3	emr-serverless.eu-west-3.amazonaws.com	HTTPS
歐洲 (西班牙)	eu-south-2	emr-serverless.eu-south-2.amazonaws.com	HTTPS
歐洲 (斯德哥爾摩)	eu-north-1	emr-serverless.eu-north-1.amazonaws.com	HTTPS
Middle East (Bahrain)	me-south-1	emr-serverless.me-south-1.amazonaws.com	HTTPS
中東 (UAE)	me-central-1	emr-serverless.me-central-1.amazonaws.com	HTTPS

區域名稱	區域	端點	通訊協定
南美洲 (聖保羅)	sa-east-1	emr-serverless.sa-east-1.amazonaws.com	HTTPS
AWS GovCloud (美國東部)	us-gov-east-1	emr-serverless.us-gov-east-1.amazonaws.com	HTTPS
AWS GovCloud (美國西部)	us-gov-west-1	emr-serverless.us-gov-west-1.amazonaws.com	HTTPS

Service Quotas

服務配額，也稱為限制，是您的服務資源或操作的最大數量 AWS 帳戶 可以使用。EMR無伺服器會每分鐘收集服務配額使用量指標，並將其發佈到AWS/Usage命名空間中。

> **Note**
>
> 新增 AWS 帳戶的初始配額可能會隨著時間而增加。Amazon EMR 無伺服器監控每個帳戶內的使用情況 AWS 區域，然後根據您的使用量自動增加配額。

下表列出EMR無伺服器的服務配額。如需詳細資訊，請參閱 AWS 服務 配額。

名稱	預設值限制	是否可調整？	描述
vCPUs 每個帳戶的最大並行	16	是	目前帳戶 vCPUs 可同時執行的最大數目 AWS 區域.

API限制

以下說明您的每個區域的API限制 AWS 帳戶.

資源	預設配額
ListApplications	每秒 10 筆交易。每秒 50 筆交易的爆發。
CreateApplication	每秒 1 次交易。每秒突發 25 筆交易。
DeleteApplication	每秒 1 次交易。每秒突發 25 筆交易。
GetApplication	每秒 10 筆交易。每秒 50 筆交易的爆發。
UpdateApplication	每秒 1 次交易。每秒突發 25 筆交易。
ListJobRuns	每秒 1 次交易。每秒突發 25 筆交易。
StartJobRun	每秒 1 次交易。每秒突發 25 筆交易。
GetDashboardForJobRun	每秒 1 次交易。每秒 2 筆交易爆發。
CancelJobRun	每秒 1 次交易。每秒突發 25 筆交易。
GetJobRun	每秒 10 筆交易。每秒 50 筆交易的爆發。
StartApplication	每秒 1 次交易。每秒突發 25 筆交易。
StopApplication	每秒 1 次交易。每秒突發 25 筆交易。

其他考量

下列清單包含EMR無伺服器的其他考量事項。

- EMR以下提供無伺服器服務 AWS 區域:
  - 美國東部 (俄亥俄)
  - 美國東部 (維吉尼亞北部)
  - 美國西部 (加利佛尼亞北部)
  - 美國西部 (奧勒岡)
  - 非洲 (開普敦)
  - Asia Pacific (Hong Kong)
  - 亞太區域 (雅加達)
  - 亞太區域 (孟買)
  - 亞太區域 (大阪)
  - 亞太區域 (首爾)
  - 亞太區域 (新加坡)
  - 亞太區域 (雪梨)
  - 亞太區域 (東京)
  - 加拿大 (中部)
  - 歐洲 (法蘭克福)
  - 歐洲 (愛爾蘭)
  - 歐洲 (倫敦)
  - 歐洲 (米蘭)
  - Europe (Paris)
  - 歐洲 (西班牙)
  - 歐洲 (斯德哥爾摩)
  - Middle East (Bahrain)
  - 中東 (UAE)
  - 南美洲 (聖保羅)
  - AWS GovCloud (美國東部)
  - AWS GovCloud (美國西部)

如需與這些區域相關聯的端點清單，請參閱服務端點。

- 工作執行的預設逾時時間為 12 小時。您可以使用`startJobRun`API或中的`executionTimeoutMinutes`性質變更此設定 AWS SDK。如果您希望工作執行永不逾時，可以將其設`executionTimeoutMinutes`定為 0。例如，如果您有串流應用程式，您可以設定`executionTimeoutMinutes`為 0 以允許串流工作持續執行。
- 中的`billedResourceUtilization`屬性會`getJobRun`API顯示彙總 v CPU、記憶體和儲存 AWS 已針對工作執行收費。計費資源包括員工最低使用 1 分鐘，以及每個工作人員超過 20 GB 的額外儲存空間。這些資源不包括閒置預先初始化 Worker 的使用情況。
- 如果沒有VPC連接，作業可以訪問一些 AWS 服務 端點在相同 AWS 區域。 這些服務包括 Amazon S3， AWS Glue，Amazon CloudWatch 日誌， AWS KMS, AWS Security Token Service, Amazon DynamoDB和 AWS Secrets Manager。 您可以啟用VPC連接以訪問其他 AWS 服務 通過 AWS PrivateLink，但您不需要這樣做。若要存取外部服務，您可以使用VPC.
- EMR無伺服器不支援HDFS。Worker 上的本機磁碟是暫時儲存體，EMR無伺服器會在工作執行期間使用它來洗牌和處理資料。
- EMR無伺服器不支援現有emr-dynamodb-connector的.

Amazon EMR 無伺服器發行版本

Amazon EMR 版本是來自大數據生態系統的一組開放原始碼應用程式。每個版本都包含大數據應用程式、元件和功能，這些功能可讓您在執行任務時部署和設定 Amazon EMR 無伺服器。

使用 Amazon EMR 6.6.0 及更高版本，您可以部署無EMR伺服器。舊EMR版 Amazon 無法使用此部署選項。當您提交工作時，您必須指定下列其中一個支援的版本。

主題

EMR Serverless 7.2.0

下表列出了可用的應用程式版本 EMR Serverless 7.2.0.

應用程式	版本
Apache Spark	3.5.1
Apache Hive	3.1.3
Apache Tez	0.10.2

EMR無伺服器 7.2.0 版本資訊

- EMR無伺服器的 Lake Formation — 您現在可以使用 AWS Lake Formation ，對 S3 支援的資料目錄資料表套用精細的存取控制。此功能可讓您針對EMR無伺服器 Spark 作業中的讀取查詢，設定表格、列、欄和儲存格層級存取控制。如需詳細資訊，請參閱 the section called “Lake Formation FGAC” 和 the section called “考量事項”。

EMR Serverless 7.1.0

下表列出了可用的應用程式版本 EMR Serverless 7.1.0.

應用程式	版本
Apache Spark	3.5.0
Apache Hive	3.1.3
Apache Tez	0.10.2

EMR Serverless 7.0.0

下表列出了可用的應用程式版本 EMR Serverless 7.0.0.

應用程式	版本
Apache Spark	3.5.0
Apache Hive	3.1.3
Apache Tez	0.10.2

EMR Serverless 6.15.0

下表列出了可用的應用程式版本 EMR Serverless 6.15.0.

應用程式	版本
Apache Spark	3.4.1
Apache Hive	3.1.3
Apache Tez	0.10.2

EMR無伺服器 6.15.0 版本資訊

- TLS支援 — 使用 Amazon EMR 無伺服器版本 6.15.0 及更高版本，您可以在 Spark 任務執行中啟用員工之間的相互TLS加密通訊。啟用時，EMRServerless 會自動為每個工作者產生唯一的憑證，這些憑證在工作執行中佈建的工作執行中，工作者會在TLS握手期間互相驗證，並建立加密通道以安全地處理資料。如需有關相互TLS加密的詳細資訊，請參閱 Worker 間加密。

EMR Serverless 6.14.0

下表列出了可用的應用程式版本 EMR Serverless 6.14.0.

應用程式	版本
Apache Spark	3.4.1
Apache Hive	3.1.3
Apache Tez	0.10.2

EMR Serverless 6.13.0

下表列出了可用的應用程式版本 EMR Serverless 6.13.0.

應用程式	版本
Apache Spark	3.4.1
Apache Hive	3.1.3

應用程式	版本
Apache Tez	0.10.2

EMR Serverless 6.12.0

下表列出了可用的應用程式版本 EMR Serverless 6.12.0.

應用程式	版本
Apache Spark	3.4.0
Apache Hive	3.1.3
Apache Tez	0.10.2

EMR Serverless 6.11.0

下表列出了可用的應用程式版本 EMR Serverless 6.11.0.

應用程式	版本
Apache Spark	3.3.2
Apache Hive	3.1.3
Apache Tez	0.10.2

EMR無伺服器 6.11.0 版本資訊

- 存取其他帳戶中的 S3 資源-使用 6.11.0 及更高版本，您可以設定多個IAM角色，以在存取不同的 Amazon S3 儲存貯體時採用 AWS 來自EMR無伺服器的帳戶。

EMR Serverless 6.10.0

下表列出了可用的應用程式版本 EMR Serverless 6.10.0.

應用程式	版本
Apache Spark	3.3.1
Apache Hive	3.1.3
Apache Tez	0.10.2

EMR無伺服器 6.10.0 版本資訊

- 對於版本 6.10.0 或更高版本的EMR無伺服器應用程式，內容的預設值為。`spark.dynamicAllocation.maxExecutors` `infinity`舊版預設為`100`。如需詳細資訊，請參閱火花工作屬性。

EMR Serverless 6.9.0

下表列出了可用的應用程式版本 EMR Serverless 6.9.0.

應用程式	版本
Apache Spark	3.3.0
Apache Hive	3.1.3
Apache Tez	0.10.2

EMR無伺服器 6.9.0 版本資訊

- Amazon Redshift 集成阿帕奇星火包含在 Amazon EMR 版本 6.9.0 及更高版本。以前是一個開放原始碼工具，本機整合是一個 Spark 連接器，可用於建置在 Amazon Redshift 和 Amazon Redshift Serverless 中讀取和寫入資料的 Apache Spark 應用程式。如需詳細資訊，請參閱在 Amazon 無服務器上使用亞馬遜紅移集成阿帕奇星火 EMR。
- EMR無伺服器版本 6.9.0 新增支援 AWS 引力 2（臂 64）架構。您可以使用`create-application`和的`architecture`參數`update-application`APIs來選擇 arm64 架構。如需詳細資訊，請參閱Amazon EMR 無伺服器架構選項。

- 您現在可以直接從EMR無伺服器 Spark 和 Hive 應用程式匯出、匯入、查詢和加入 Amazon DynamoDB 資料表。如需詳細資訊，請參閱使用 Amazon 無伺服器連線至 DynamoDB EMR。

已知問題

- 如果針對 Apache Spark 使用 Amazon Redshift 整合，並且具有 Parquet 格式的精確度為微秒的 time、timetz、timestamp 或 timestamptz，則連接器會將時間值四捨五入為最接近的微秒值。請使用文字卸載格式 `unload_s3_format` 參數作為一種解決方法。

EMR Serverless 6.8.0

下表列出了可用的應用程式版本 EMR Serverless 6.8.0.

應用程式	版本
Apache Spark	3.3.0
Apache Hive	3.1.3
Apache Tez	0.9.2

EMR Serverless 6.7.0

下表列出了可用的應用程式版本 EMR Serverless 6.7.0.

應用程式	版本
Apache Spark	3.2.1
Apache Hive	3.1.3
Apache Tez	0.9.2

引擎特定的變更、增強功能和已解決的問題

下表列出新的引擎特定功能。

變更	描述
功能	Tez 排程器現在支援 Tez 工作的預佔，而不是容器的預佔

EMR Serverless 6.6.0

下表列出了可用的應用程式版本 EMR Serverless 6.6.0.

應用程式	版本
Apache Spark	3.2.0
Apache Hive	3.1.2
Apache Tez	0.9.2

EMR無伺服器初始版本說明

- EMR無伺服器支援 Spark 組態分類`spark-defaults`。此分類會變更 Spark `spark-defaults.conf` XML 檔案中的值。組態分類可讓您自訂應用程式。如需詳細資訊，請參閱設定應用程式。
- EMR無伺服器支援 Hive 組態分類`hive-site`、`tez-site`、`emrfs-site`、和`core-site`。這種分類可以改變蜂巢的`hive-site.xml`文件，TEZ 的`tez-site.xml`文件，Amazon EMR 的EMRFS 設置，或 Hadoop 的`core-site.xml`文件，分別值。組態分類可讓您自訂應用程式。如需詳細資訊，請參閱設定應用程式。

引擎特定的變更、增強功能和已解決的問題

- 下表列出了蜂巢和 Tez 反向移植。

蜂巢和特茲變化

變更	描述
向後移植	TEZ-4430：固定的問題與財產 `tez.task.launch.cmd-opts`
向後移植	HIVE-25971：修正 Tez 工作關閉延遲，因為開啟快取的執行緒集區

文件歷史記錄

下表說明自上次發行EMR無伺服器以來，文件的重要變更。如需有關此文件更新的詳細資訊，您可以訂閱RSS摘要。

變更	描述	日期
新版本	EMR Serverless 7.2.0	2024年7月25日
新版本	EMR Serverless 7.1.0	2024年4月17日
更新為現有策略。	將新的SidCloudWatchPolicyStatement 和添加EC2PolicyStatement 到 A mazonEMRServerless ServiceRolePolicy 策略中。	2024年1月25日
新版本	EMR Serverless 7.0.0	2023 年 12 月 29 日
新版本	EMR Serverless 6.15.0	2023 年 11 月 17 日
新功能	設定當您從EMR無伺服器 (6.11 及更高版本) 存取不同帳戶中的 Amazon S3 儲存貯體時採用的多個IAM角色	2023 年 10 月 18 日
新版本	EMR Serverless 6.14.0	2023 年 10 月 17 日
新功能	EMR無伺服器的預設應用程式組態	2023 年 9 月 25 日
更新為默認蜂房屬性	更新hive.driver.disk 、、hive.tez.disk.size 和 tez.grouping.min-size Hive 工作屬性的預設值。hive.tez.auto.reducer.parallelism	2023 年 9 月 12 日

新版本	EMR Serverless 6.13.0	2023 年 9 月 11 日
新版本	EMR Serverless 6.12.0	2023 年 7 月 21 日
新版本	EMR Serverless 6.11.0	2023 年 6 月 8 日
服務連結角色原則的更新	更新AmazonEMRServerlessServiceRolePolicy_SLR角色以在命名空間中發佈帳戶層級使用情況"AWS/Usage" 。	2023 年 4 月 20 日
EMR Serverless 一般可用性 (GA)	這是EMR無伺服器的第一個公開發行版本。	2022 年 6 月 1 日

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。