aws

使用者指南

EC2 Image Builder

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

EC2 Image Builder: 使用者指南

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

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

Table of Contents

什麼是 EC2 Image Builder？

EC2 Image Builder 是全受管 AWS 服務 ，可協助您自動化自訂、安全和 up-to-date 伺服器映像的建立、管理和部署。您可以使用 AWS Management Console AWS Command Line Interface、或 API 在您的 AWS 帳戶.

您擁有映像產生器在帳戶中建立的自訂映像檔。您可以設定管線，為您擁有的映像自動更新和系統修補。您也可以執行獨立的命令，使用已定義的組態資源建立映像檔。

映 Image Builder 管線精靈可引導您完成建立自訂映像檔的步驟，如下所示：

1. 為您的自訂選擇一個基本映像。
2. 在基本映像檔中新增或移除軟體。
3. 使用構建組件自定義設置和腳本。
4. 執行選取的測試或建立自訂測試元件。
5. 將 AMI 發佈至 AWS 區域 和 AWS 帳戶。
6. 如果您的 Image Builder 管道建立自訂 Amazon 機器映像 (AMI) 以進行分發，您可以授權其他 AWS 帳戶、組織和 OU 從您的帳戶啟動該映像檔。系統會針對與 AMI 相關聯的費用向您的帳戶收取費用。

區段內容

EC2 Image Builder 的功能

EC2 Image Builder 提供下列功能：

提高生產力並減少建立符合規範和 up-to-date 影像的作業

Image Builder 透過自動化您的建置管道，減少大規模建立和管理映像所需的工作量。您可以提供組建執行排程偏好設定，將組建自動化。使用最新的作業系統修補程式，自動化可降低維護軟體的作業成本。

增加服務運作時

Image Builder 可讓您存取測試元件，您可以在部署之前用來測試映像檔。您還可以使用 AWS 任務協調器和執行器（AWS TOE）創建自定義測試組件，並使用它們。只有在所有已設定的測試都成功時，映像 Image Builder 生器才會散佈您的映像檔。

提高部署的安全性列

Image Builder 可讓您建立映像檔，以移除不必要的元件安全性弱點暴露。您可以套用 AWS 安全性設定來建立符合產業和內部安全性準則的安全out-of-the-box 影像。Image Builder 也為受管制產業的公司提供設定集合。您可以使用這些設定來協助您快速輕鬆地建置符合 STIG 標準的相容影像。如需可透過 Image Builder 取得之 STIG 元件的完整清單，請參閱適用於 EC2 Image Builder 的 Amazon 受管 STIG 強化元件。

集中式執行和歷程追蹤

透過 Image Builder 的內建整合 AWS Organizations，您可以強制執行原則，限制帳戶僅從已核准的 AMI 執行執行個體。

簡化各地的資源共享 AWS 帳戶

EC2 Image Builder 與 AWS Resource Access Manager (AWS RAM) 整合，可讓您與任何 AWS 帳戶或透過共用特定資源 AWS Organizations。可共用的 EC2 Image Builder 資源包括：

- 元件
- 映像
- 圖片食譜
- 容器食譜

如需詳細資訊，請參閱 共用 EC2 Image Builder 資源。

支援的作業系統

Image Builder 支援下列作業系統版本：

作業系統/分發	支援的版本
Amazon Linux	二及二二三
CentOS	七和八
CentOS Stream	8
Red Hat Enterprise Linux (RHEL)	七和八
瑞士企業伺服器	十二和十五
Ubuntu	18.04 工資教學研究所、20.04 工業教育研究所及 22.04 律師資
Windows Server	二零一六年、二零一九年及二零二

支援的影像格式

對於自訂 AMI 映像，您可以選擇現有的 AMI 作為起點。對於 Docker 容器映像，您可以選擇託管在其上的公開映像檔 DockerHub、Amazon ECR 中的現有容器映像或 Amazon 管理的容器映像。

概念

以下術語和概念是您了解和使用 EC2 Image Builder 的核心。

AMI

亞馬遜機器映像 (AMI) 是 Amazon EC2 中部署的基本單位，也是您可以使用映像產生器建立的映像類型之一。AMI 是預先設定的虛擬機器映像，其中包含用於部署 EC2 執行個體的作業系統 (OS) 和預先安裝的軟體。如需詳細資訊，請參閱 Amazon 機器映像 (AMI)。

影像管線

映像管道提供自動化架構，用於在 AWS上建置安全 AMI 和容器映像。Image Builder 映像管線與映像方案或容器方案相關聯，該方案或容器方案可定義映像建置生命週期的建置、驗證和測試階段。

映像管線可以與定義映像建置位置的基礎架構組態相關聯。您可以定義屬性，例如執行個體類型、子網路、安全性群組、記錄以及其他與基礎架構相關的組態。您也可以將映像管線與分佈組態建立關聯，以定義您要如何部署映像。

受管理映像

受管理的映像檔是映 Image Builder 中的資源，其中包含 AMI 或容器映像，以及中繼資料 (例如版本和平台)。Image Builder 管線會使用受管理的映像檔來決定要用於組建的基礎映像。在本指南中，受管理的映像有時稱為「映像」，但映像與 AMI 不同。

圖片食譜

Image Builder 映像配方是一份文件，用於定義基本影像，以及套用至基本影像的元件，以產生輸出 AMI 影像所需組態的文件。您可以使用映像配方來複製建構。您可以使用主控台精靈、或 API 來共用、分支和編輯映像 Image Builder 映像配方。 AWS CLI您可以將影像配方與版本控制軟體搭配使用，以維護可共用的版本化映像配方。

容器食譜

Image Builder 容器方案是一份文件，用於定義基本影像，以及套用至基本影像的元件，以產生輸出容器映像所需組態的文件。您可以使用容器方案來複製組建。您可以使用主控台精靈、或 API 來共用、分支和編輯映 Image Builder 映像配方。 AWS CLI您可以將容器配方與版本控制軟體搭配使用，以維護可共用的版本化容器配方。

基本影像

基本影像是您在影像或容器方案文件中使用的選取映像檔和作業系統，以及元件。結合基本影像和元件定義可產生輸出影像所需的規劃。

元件

組件定義了在映像創建之前自定義實例（構建組件）或測試從創建映像（測試組件）啟動的實例所需的步驟順序。

元件是從宣告式純文字 YAML 或 JSON 文件建立，該文件描述用於建置和驗證的執行階段設定，或測試管線所產生的執行個體。使用元件管理應用程式在執行個體上執行的元件。組件管理應用程序解析文檔並運行所需的步驟。

建立這些元件之後，會使用映像配方或容器配方將一或多個元件分組在一起，以定義建置和測試虛擬機器或容器映像的計劃。您可以使用擁有和管理的公共組件 AWS，也可以創建自己的組件。如需元件的詳細資訊，請參閱AWS 任務協調器和執行器 元件管理員。

元件文件

宣告式純文字 YAML 或 JSON 文件，說明您可以套用至映像的自訂組態。該文檔用於創建構建或測試組件。

執行階段

EC2 Image Builder 有兩個執行階段：建置和測試。每個執行階段都有一或多個階段，其中包含組件文件所定義的組態。

組態階段

下列清單顯示在建置和測試階段執行的階段：

構建階段：

組建階段

映像管線在執行時會從建置階段的建置階段開始。系統會下載基礎映像檔，而針對元件建置階段所指定的組態會套用至建置和啟動執行個體。

驗證階段

在 Image Builder 啟動執行個體並套用所有建置階段自訂之後，驗證階段就會開始。在此階段，Image Builder 會根據元件為驗證階段指定的組態，確保所有自訂都如預期般運作。如果執行個體驗證成功，Image Builder 會停止執行個體、建立映像檔，然後繼續進行測試階段。

測試階段：

測試階段

在此階段中，Image Builder 會在驗證階段順利完成後，從映像檔啟動執行個體。Image Builder 會在此階段執行測試元件，以確認執行個體狀況良好且正常運作。

容器主機測試階段

Image Builder 針對您在容器方案中選取的所有元件執行測試階段之後，Image Builder 器會針對容器工作流程執行此階段。容器主機測試階段可以運行其他測試，以驗證容器管理和自定義運行時配置。

工作流程

工作流程會定義 Image Builder 在建立新映像時執行的步驟順序。所有映像都有構建和測試工作流程。容器具有額外的散發工作流程。

工作流類型

BUILD

涵蓋建立的每個影像的建置階段組態。

TEST

涵蓋每個建立影像的測試階段組態。

DISTRIBUTION

涵蓋容器映像的發佈工作流程。

定價

使用 EC2 Image Builder 建立自訂 AMI 或容器映像無須付費。不過，標準定價適用於程序中使用的其他服務。下列清單包含在建立、建置、儲存和散發自訂 AMI 或容器映像時可能產生成本的部 AWS 服務 分使用量 (視您的組態而定)。

- 啟動 EC2 執行個體
- 在 Amazon S3 上存放日誌
- 使用 Amazon Inspector 驗證圖像
- 為您的 AMI 儲存 Amazon EBS 快照
- 在 Amazon ECR 中存儲容器映像
- 將容器映像推入和拉出 Amazon ECR
- 如果 Systems Manager 進階層已開啟，且 Amazon EC2 執行個體透過現場部署啟用執行，您可能需要透過 Systems Manager 支付資源費用

相關 AWS 服務

EC2 Image Builder 使用其他 AWS 服務 來建置映像。視您的 Image Builder 映像配方或容器配方組態而定，可能會使用下列服務。

AWS License Manager

AWS License Manager 可讓您從帳戶授權組態存放區建立和套用授權組態。對於每個 AMI，您可以使用 Image Builder 附加至您 AWS 帳戶 可以存取的既有授權組態，做為 Image Builder 工作流程的一部

分。授權組態只能套用至 AMI。Image Builder 只能使用既有的授權組態，無法直接建立或修改授權組態。License Manager 設定不會複 AWS 區域 製必須在您帳戶中啟用的設定，例如 `ap-east-1` (亞太地區：香港) 和 `me-south-1` (中東：巴林) 區域之間。

AWS Organizations

AWS Organizations 可讓您對組織中的帳戶套用服務控制原則 (SCP)。您可以建立、管理、啟用和停用個別策略。與所有其他成 AWS 品和服務類似，Image Builder 會遵循中定義的原則 AWS Organizations。 AWS 提供常見案例的範本 SCP，例如對成員帳戶強制執行限制，以啟動僅具有已核准 AMI 的執行個體。

Amazon Inspector

Image Builder 使用 Amazon Inspector 器做為預設的弱點掃描代理程式，為 Amazon Linux 2、視窗伺服器 2012 和視窗伺服器 2016 建立安全基準。如需詳細資訊，請參閱什麼是 Amazon Inspector？

AWS Resource Access Manager

AWS Resource Access Manager (AWS RAM) 可讓您與任何人 AWS 帳戶 或透過共用您的資源 AWS Organizations。如果您有多個資源 AWS 帳戶，您可以集中建立資源，並使 AWS RAM 用這些資源與其他帳戶共用。EC2 Image Builder 允許共用下列資源：元件、映像和映像配方。若要取得有關的更多資訊 AWS RAM，請參閱AWS Resource Access Manager 使用者指南。如需共用 Image Builder 資源的相關資訊，請參閱共用 EC2 Image Builder 資源。

Amazon CloudWatch 日誌

您可以使用 Amazon CloudWatch 日誌來監控、存放和存取 EC2 執行個體 AWS CloudTrail、Amazon Route 53 和其他來源的日誌檔。

Amazon Elastic Container Registry (Amazon ECR)

Amazon ECR 是安全、可擴展且可靠的受管 AWS 容器映像登錄服務。您使用映像產生器建立的容 Image Builder 會儲存在來源區域 (執行組建的位置) 的 Amazon ECR，以及您散發容器映像的任何區域。如需 Amazon ECR 的詳細資訊，請參閱 Amazon 彈性容器登錄使用者指南。

EC2 Image Builder 的運作方式

當您使用 EC2 Image Builder 管道主控台精靈建立自訂映像時，精靈會引導您完成下列步驟。

1. 指定管線詳細資訊 — 輸入管道的相關資訊，例如名稱、說明、標籤和排程，以執行自動化組建。如果您願意，您可以選擇手動構建。
2. 選擇配方 — 選擇建立 AMI 或建立容器映像檔。對於這兩種類型的輸出影像，您可以為方案輸入名稱和版本，選取基本映像，然後選擇要新增的元件以進行建置和測試。您也可以選擇自動版本控制，以確保您永遠針對基本映像使用最新可用的作業系統 (OS) 版本。容器配方另外定義碼頭檔案，以及輸出 Docker 容器映像檔的目標 Amazon ECR 儲存庫。

   > **Note**
   >
   > 元件是映像配方或容器配方所使用的建置區塊。例如，用於安裝的套件、安全性強化步驟和測試。選取的基本影像和元件構成影像配方。

3. 定義基礎設施組態 — Image Builder 會啟動帳戶中的 EC2 執行個體，以自訂映像並執行驗證測試。基礎結構組態設定會指定建置程序 AWS 帳戶 期間將在您的執行個體中執行的基礎結構詳細資料。
4. 定義發佈設定 — 在建置完成並通過所有測試之後，選擇要散發映像檔的目標 AWS 區域。管線會自動將映像分發到執行組建的區域，您可以為其他區域新增映像分發。

您從自定義基本映像構建的圖像位於您的 AWS 帳戶. 您可以輸入建置排程，設定映像管道以產生映像檔的更新和修補版本。建置完成後，您可以透過 Amazon 簡易通知服務 (SNS) 收到通知。除了產生最終映像之外，Image Builder 主控台精靈還會產生配方，可與現有版本控制系統和持續整合/持續部署 (CI/CD) 管道搭配使用，以實現可重複自動化。您可以共享和創建新版本的食譜。

區段內容

- 合規

AMI 元素

Amazon 機器映像 (AMI) 是預先設定的虛擬機器 (VM) 映像，其中包含用於部署 EC2 執行個體的作業系統和軟體。

AMI 包括以下元素：

- 虛擬機器根磁碟區的範本。當您啟動 Amazon EC2 虛擬機器時，根裝置磁碟區會包含用於啟動執行個體的映像檔。使用執行個體存放區時，根裝置是從 Amazon S3 中的範本建立的執行個體存放區磁碟區。如需詳細資訊，請參閱 Amazon EC2 根裝置磁碟區。
- 使用 Amazon EBS 時，根裝置是從 EBS 快照建立的 EBS 磁碟區。
- 啟動權限，以決定可以 AWS 帳戶 使用 AMI 啟動虛擬機器的權限。
- 區塊裝置對應資料，指定啟動後要連接至執行個體的磁碟區。
- 每個帳號之每個區域的唯一資源識別碼。
- 中繼資料承載，例如標籤和內容，例如區域、作業系統、架構、根裝置類型、提供者、啟動權限、根裝置的儲存空間以及簽署狀態。
- 適用於 Windows 映像檔的 AMI 簽章，可防止未經授權的竄改。如需詳細資訊，請參閱執行個體識別文件。

預設配額

若要檢視 Image Builder 的預設配額，請參閱 Image Builder 端點和配額。

AWS 區域和端點

若要檢視 Image Builder 的服務端點，請參閱 Image Builder 端點和配額。

元件管理

EC2 Image Builder 使用元件管理應用程式 AWS 任務協調器和執行器 (AWS TOE)，可協助您協調複雜的工作流程、修改系統組態，以及使用 YAML 指令碼元件測試系統。由 AWS TOE 於是獨立的應用程式，因此不需要任何額外的設定。它可以在任何雲基礎架構和內部部署上運行。若要開始使用 AWS TOE 作為獨立應用程式，請參閱開始使用 AWS TOE。

Image Builder 用 AWS TOE 來執行所有執行個體活動。其中包括在拍攝快照之前建立和驗證映像，以及在建立最終映像之前測試快照以確保快照能如預期般運作。如需 Image Builder 如何用 AWS TOE 來管理其元件的詳細資訊，請參閱〈〉使用 Image Builder 管理元件。如需使用建立元件的更多資訊 AWS TOE，請參閱AWS 任務協調器和執行器 元件管理員。

影像測試

在建立最終映像檔之前，您可以使用 AWS TOE 測試元件來驗證映像檔，並確保其如預期般運作。

通常，每個測試組件都包含一個 YAML 文檔，其中包含測試腳本，測試二進製文件和測試元數據。測試腳本包含用於啟動測試二進製文件的協調流程命令，可以使用操作系統支持的任何語言編寫。退出狀態代碼表示測試結果。測試元數據描述測試及其行為；例如，名稱，描述，測試二進製文件的路徑以及預期的持續時間。

語義版本控制

Image Builder 使用語意版本控制來組織資源，並確保它們具有唯一的 ID。語義版本有四個節點：

<major>。 <minor>。 <patch>/<build>

您可以為前三個節點指派數值，並且可以篩選所有數值。

語意版本控制包含在每個物件的 Amazon 資源名稱 (ARN) 中，適用於該物件的層級，如下所示：

1. 無版本 ARN 和名稱 ARN 不會在任何節點中包含特定值。節點要么完全關閉，要么將它們指定為通配符，例如：x.x.x。
2. <major>版本 ARN 只有前三個節點：。 <minor>。 <patch>
3. 構建版本 ARN 具有所有四個節點，並指向特定版本的對象的特定構建。

指派：對於前三個節點，您可以為每個節點指派任意正整數值 (包括零)，上限為 2^30-1 或 1073741823。Image Builder 會將建置編號自動指派給第四個節點。

模式：對於您可以指派的節點，您可以使用任何符合指派要求的數值模式。例如，您可以選擇 1.0.0 等數值或 2021.01.01 等日期格式的軟體版本模式。

選取：使用語意版本控制時，您可以彈性地使用萬用字元 (x) 來指定最新版本或節點，以便為您的方案選取基本影像。在任何節點中使用萬用字元時，第一個萬用字元右側的所有節點也必須是萬用字元。

例如，假設下列最新版本：2.2.4、1.7.8 和 1.6.8，使用萬用字元選取版本會產生下列結果：

- `x.x.x`= 2.2.4
- `1.x.x`= 1.7.8
- `1.6.x`= 1.6.8
- `x.2.x`無效，並產生錯誤
- `1.x.8`無效，並產生錯誤

建立的資源

建立管線時，除非符合下列條件，否則不會建立 Image Builder 外部的資源：

- 透過管線排程建立影像時
- 當您從 Image Builder 主控台的「動作」功能表中選擇「執行管線」時
- 當您從 API 或 AWS CLI：或執行下列任一指令StartImagePipelineExecution時 CreateImage

下列資源會在映像建置程序期間建立：

AMI 圖像管道

- EC2 執行個體 (暫時性)
- EC2 執行個體上的系統管理員庫存關聯 (透過系統管理員狀態管理員，如果`EnhancedImageMetadata`已啟用)
- Amazon EC2
- 與 Amazon EC2 AMI 關聯的亞馬遜 EBS 快照

容器映像管道

- 在 EC2 實例上運行的碼頭容器（臨時）
- EC2 實例上的 Systems Manager 理器庫存關聯（通過系統管理器狀態管理器）`EnhancedImageMetadata`已啟用）
- Docker 容器映像
- Dockerfile

建立映像後，會刪除所有暫存資源。

發佈

EC2 Image Builder 可以將 AMI 或容器映像散發到任何 AWS 區域。映像會複製到您在用於建置映像的帳戶中指定的每個區域。

對於 AMI 輸出映像檔，您可以定義 AMI 啟動許可，以控制允許哪 AWS 帳戶 些使用建立的 AMI 啟動 EC2 執行個體。例如，您可以將映像設為私人、公開或與特定帳戶共用。如果您同時將 AMI 散佈至其他區域，並為其他帳戶定義啟動權限，則啟動權限會傳播至所有分散 AMI 之區域中的 AMI。

您也可以使用您的 AWS Organizations 帳戶強制執行成員帳戶的限制，以便僅透過已核准且符合規範的 AMI 啟動執行個體。如需詳細資訊，請參閱AWS 帳戶 在組織中管理。

若要使用 Image Builder 主控台更新發佈設定，請依照上述步驟執行建立新的映像配方版本 (主控台)，或使用主控台建立新的容器配方版本。

分享資源

若要與其他帳戶或內部帳戶共用元件、配方或影像 AWS Organizations，請參閱共用 EC2 Image Builder 資源。

合規

對於 CIS，EC2 Image Builder 使用 Amazon Inspector 針對暴露、漏洞和與最佳實務和合規標準的偏差執行評估。例如，Image Builder 會評估非預期的網路可存取性、未修補的 CVE、公用網際網路連線，以及遠端根登入啟用。Amazon Inspector 是以測試元件的形式提供，您可以選擇將其新增至映像配方。有關 Amazon Inspector 查器的更多信息，請參閱 Amazon Inspector 用戶指南。為了強化，EC2 Image Builder 會使用 STIG 進行驗證。如需可透過 Image Builder 取得之 STIG 元件的完整清單，請參閱適用於 EC2 Image Builder 的 Amazon 受管 STIG 強化元件。如需詳細資訊，請參閱網際網路安全中心 (CIS) 基準測試。

開始使用 EC2 Image Builder

本章協助您使用 EC2 Image Builder 建立映像管道主控台精靈，設定環境並首次建立自動化映像管道或容器管道。

目錄

必要條件

驗證下列先決條件，以使用 EC2 Image Builder 建立映像管道。除非另有說明，否則所有類型的管線都需要先決條件。

EC2 Image Builder 服務連結角色

EC2 Image Builder 使用服務連結角色代表您授與其他 AWS 服務的許可。您不需要手動建立一個服務連結角色。當您在 AWS 管理主控台、或 AWS API 中建立第一個 Image Builder 生器資源時，Image Builder 會為您建立服務連結角色。 AWS CLI如需有關 Image Builder 在您的帳戶中建立的服務連結角色的詳細資訊，請參閱使用 EC2 Image Builder 的服務連結角色。

組態需求

- Image Builder 支持AWS PrivateLink。如需有關為 Image Builder 設定 VPC 端點的詳細資訊，請參閱EC2 Image Builder 和介面 VPC 端點 ()AWS PrivateLink。
- Image Builder 器用來建立容器映像的執行個體必須具有網際網路存取權，才能 AWS CLI 從 Amazon S3 下載，並從 Docker Hub 儲存庫下載基本映像 (如果適用)。Image Builder 會 AWS CLI 使用從容器配方取得 Docker 檔案，並將其儲存為資料。
- 映像產生器用來建置映像檔和執行測試的執行個體必須具有 Systems Manager 服務的存取權。安裝需求取決於您的作業系統。

  若要查看基本映像檔的安裝需求，請選擇符合您基本映像作業系統的索引標籤。

Linux

對於 Amazon EC2 Linux 執行個體，Image Builder 會在建置執行個體上安裝系統管理員代理程式 (如果尚未存在)，並在建立映像之前將其移除。

Windows

Image Builder 不會在 Amazon EC2 Windows 伺服器建置執行個體上安裝系統管理員代理程式。如果您的基本映像檔未預先安裝 Systems Manager 代理程式，您必須從來源映像檔啟動執行個體、在執行個體上手動安裝 Systems Manager，然後從執行個體建立新的基礎映像檔。

若要在 Amazon EC2 Windows 伺服器執行個體上手動安裝系統管理員代理程式，請參閱AWS Systems Manager 使用指南中的在適用於 Windows 伺服器的 EC2 執行個體上手動安裝系統管理員代理程式

容器存放庫 (容器映像管道)

對於容器映像管線，方案定義了在目標容器存放庫中產生並儲存之 Docker 映像的組態。在為 Docker 映像檔建立容器配方之前，您必須先建立目標儲存庫。

Image Builder 使用 Amazon ECR 做為容器映像的目標儲存庫。若要建立 Amazon ECR 儲存庫，請遵循 Amazon 彈性容器登錄使用者指南中建立儲存庫中所述的步驟。

AWS Identity and Access Management (IAM)

您與執行個體設定檔相關聯的 IAM 角色必須具有執行映像檔中包含的組建和測試元件的許可。下列 IAM 角色政策必須附加至與執行個體設定檔相關聯的 IAM 角色：

- EC2InstanceProfileForImageBuilder
- EC2InstanceProfileForImageBuilderECRContainerBuilds
- 亚马孙 ManagedInstanceCore

如果您設定記錄，則基礎結構設定中指定的執行個體設定檔必須具有目標儲存貯體 (`arn:aws:s3:::BucketName/*`) 的`s3:PutObject`權限。例如：

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

```
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::bucket-name/*"
        }
    ]
}
```

連接政策

以下步驟將引導您完成將 IAM 政策附加到 IAM 角色以授予先前許可的過程。

1. 登入 AWS 管理主控台，然後開啟 IAM 主控台，網址為 https://console.aws.amazon.com/iam/。
2. 在左側導覽窗格中選擇 Policies (政策)。
3. 使用以下方式篩選策略清單 EC2InstanceProfileForImageBuilder
4. 選取原則旁邊的 bullet，然後從 [原則動作] 下拉式清單中選取 [附加]。
5. 選取要附加政策的 IAM 角色名稱。
6. 選擇連接政策。
7. 針對EC2InstanceProfileForImageBuilderECRContainerBuilds和亞馬遜 SS ManagedInstanceCore M 政策重複步驟 3-6。

> **Note**
>
> 如果要將使用 Image Builder 建立的映像複製到其他帳戶，則必須在所有目標帳戶中建立`EC2ImageBuilderDistributionCrossAccountRole`角色，並將Ec2ImageBuilderCrossAccountDistributionAccess 政策受管理的策略附加到該角色。如需詳細資訊，請參閱 共用 EC2 Image Builder 資源。

存取 EC2 Image Builder

您可以從下列其中一個介面管理 EC2 Image Builder。

- EC2 Image Builder 主控台登陸頁面。從 EC2 Image Builder 主控台。
- AWS Command Line Interface (AWS CLI)。 您可以使用 AWS CLI 來存取 AWS API 作業。若要取得更多資訊，請參閱《 AWS Command Line Interface 使用指南》中的〈安裝指 AWS 令行介面〉。

- AWS 適用於開發套件的工具。您可以使用 AWS SDK 和工具，使用偏好的語言存取和管理 Image Builder。

使用 EC2 Image Builder 主控台精靈建立映像管道

本教學將逐步引導您建立自動化管道，以使用 [建立映像管道主控台] 精靈建立和維護自訂 EC2 Image Builder 映像。為了協助您有效率地完成這些步驟，預設設定可用時會使用這些設定，並略過可選區段。

建立影像管線工作流

- 步驟 1：指定管線詳細資訊
- 步驟 2：選擇食譜
- 步驟 3：定義基礎結構組態-選用
- 步驟 4：定義發佈設定-選擇性
- 步驟 5：檢視
- 步驟 6：清除

步驟 1：指定管線詳細資訊

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 若要開始建立管線，請選擇 [建立映像管線]。
3. 在「一般」區段中，輸入您的管線名稱 (必填)。

> **Tip**
>
> 增強的中繼資料收集預設為開啟。若要確保元件與基本影像之間的相容性，請保持開啟狀態。

4. 在 [建立排程] 區段中，您可以保留 [排程] 選項的預設值。請注意，預設排程所顯示的時區為世界協調時間 (UTC)。如需有關 UTC 時間的詳細資訊，並尋找您所在時區的偏移量，請參閱時區縮寫 — 全球清單。

   對於「相依性更新」設定，請選擇「如果有相依性更新，則在排程的時間執行管線」選項。此設定會讓您的管道在開始建置之前檢查是否有更新。如果沒有更新，則會略過排程的管線建置。

> **Note**
>
> 若要確保管線如預期般辨識相依性更新和組建，您必須針對基礎映像和元件使用語意版本化 (x.x.x)。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。

5. 選擇 [下一步] 繼續進行下一個步驟。

步驟 2：選擇食譜

1. Image Builder 預設為「使用方案」區段中的現有方案。第一次使用時，請選擇「建立新配方」選項。
2. 在 [映像類型] 區段中，選擇 Amazon 機器映像 (AMI) 選項以建立可產生和散發 AMI 的映像管道。
3. 在「一般」區段中，輸入下列必要的方塊：

   - 名稱 — 您的食譜名稱
   - 版本-您的食譜版本 (使用格式<major>。 <minor>。 <patch>，其中主要、次要和修補為整數值)。新食譜一般從開始`1.0.0`。

4. 在 [來源映像檔] 區段中，保留 [選取映像檔]、[映像作業系統 (OS)] 和 [映像來源] 的預設值。這會產生由 Amazon 管理的 Amazon Linux 2 AMI 列表，供您從基本映像中進行選擇。

   a. 從 [影像名稱] 下拉式清單中，選擇影像。

   b. 保留自動版本控制選項的預設值 (使用最新的可用作業系統版本)。

   > **Note**
   >
   > 此設定可確保管線對基礎映像使用語意版本控制，以偵測自動排程工作的相依性更新。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。

5. 在 [執行個體組態] 區段中，保留系統管理員代理程式的預設值。這會導致 Image Builder 在建置和測試完成後保留系統管理員代理程式，以便在新映像中包含系統管理員代理程式。

   在本教程中，將用戶數據保持為空白。您可以在其他時間使用此區域來提供命令，或在啟動組建執行個體時執行的命令指令碼。但是，它會取代 Image Builder 可能已加入的任何指令，以確保已安裝「Systems Manager」。當您使用它時，請確定系統管理員代理程式已預先安裝在您的基礎映像上，或將安裝包含在使用者資料中。

6. 在 [元件] 區段中，您必須至少選擇一個組建元件。

在建置元件 — Amazon Linux 面板中，您可以瀏覽頁面上列出的元件。使用右上角的分頁控制項，瀏覽基本映像作業系統可用的其他元件。您也可以搜尋特定元件，或使用 [元件管理員] 建立您自己的組建元件。

在此教學課程中，請選擇使用最新安全性更新來更新 Linux 的元件，如下所示：

a. 在面板頂端的搜尋列`update`中輸入文字，以篩選結果。

b. 選取`update-linux`建置元件的核取方塊。

c. 向下捲動，然後在「選取的元件」清單的右上角選擇「全部展開」。

d. 保留版本控制選項的預設值 (使用最新可用的元件版本)。

> **Note**
>
> 此設定可確保管線對所選元件使用語意版本化，以偵測自動排程工作的相依性更新。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。

如果您選取了具有輸入參數的元件，您也會在此區域中看到參數。本教學課程不介紹參數。如需有關在元件中使用輸入參數，並在方法中設定這些參數的詳細資訊，請參閱使用 EC2 Image Builder 管理 AWS TOE 元件參數。

重新排序元件 (可選)

如果您已選擇要包含在映像中的多個元件，則可以使用 drag-and-drop 動作將它們重新排列為在建置程序期間執行的順序。

> **Note**
>
> CIS 強化元件不遵循 Image Builder 配方中的標準元件排序規則。CIS 強化元件一律會持續執行，以確保基準測試會針對您的輸出影像執行。

1. 向上捲動至可用元件清單。
2. 選取`update-linux-kernel-mainline`建置元件 (或您選擇的任何其他元件) 的核取方塊。
3. 向下捲動至 [選取的元件] 清單，以查看至少有兩個結果。

4. 新增的元件可能沒有展開其版本化或輸入參數設定。若要展開「版本控制」選項或「輸入」參數的設定，您可以選擇設定名稱旁邊的箭頭。若要展開所有選取零組件的所有設定，您可以將全部展開開關切換為關閉和開啟。
5. 選擇其中一個元件，然後向上或向下拖曳以變更元件的執行順序。
6. 若要移除`update-linux-kernel-mainline`元件，請X從元件方塊的右上角選擇。
7. 重複上一個步驟，移除您可能已加入的任何其他元件，僅保留選取的`update-linux`元件。

7. 選擇 [下一步] 繼續進行下一個步驟。

步驟 3：定義基礎結構組態-選用

Image Builder 會在您的帳戶中啟動 EC2 執行個體，以自訂映像並執行驗證測試。基礎結構組態設定會指定建置程序 AWS 帳戶 期間將在您的執行個體中執行的基礎結構詳細資料。

在 [基礎結構組態] 區段中，[組態] 選項預設為`Create infrastructure configuration using service defaults`。這會為用於設定映像的 EC2 建置和測試執行個體建立 IAM 角色和關聯的執行個體設定檔。如需有關基礎設施組態設定的詳細資訊，請參閱 EC2 Image Builder API 參考CreateInfrastructureConfiguration中的。

在本教程中，我們使用的是默認設置。

> **Note**
>
> 若要指定用於私有 VPC 的子網路，您可以建立自己的自訂基礎結構組態，或使用已建立的設定。

- 選擇 [下一步] 繼續進行下一個步驟。

步驟 4：定義發佈設定-選擇性

散發組態包括輸出 AMI 名稱、加密的特定區域設定、啟動權限 AWS 帳戶，以及可啟動輸出 AMI 的組織和組織單位 (OU)，以及授權組態。

在 [發佈設定] 區段中，[組態] 選項預設為`Create distribution settings using service defaults`。此選項將輸出 AMI 分配到當前區域。如需有關設定散發設定的詳細資訊，請參閱管理 EC2 Image Builder 分發設定。

在本教程中，我們使用的是默認設置。

- 選擇 [下一步] 繼續進行下一個步驟。

步驟 5：檢視

「複查」區段會顯示您已設定的所有設定。若要編輯任何指定區段中的資訊，請選擇步驟區段右上角的「編輯」按鈕。例如，如果您要變更管線名稱，請選擇「步驟 1：管線詳細資訊」區段右上角的 「編輯」按鈕。

1. 檢閱設定後，請選擇「建立管道」以建立管道。
2. 您可以在頁面頂端看到成功或失敗訊息，因為您的資源是針對發佈設定、基礎結構組態、新方案和管道建立的。若要查看資源的詳細資訊 (包括資源識別碼)，請選擇 [檢視詳細資料]。
3. 檢視資源的詳細資料之後，您可以從導覽窗格中選擇資源類型，以檢視其他資源的詳細資訊。例如，若要查看新管道的詳細資訊，請從導覽窗格中選擇「影像管線」。如果建置成功，則新管線會顯示在「影像管線」清單中。

步驟 6：清除

您的 Image Builder 環境就像您的住家一樣，需要定期維護以協助您找到所需的內容，並完成工作，而不必涉及雜亂。請務必定期清理您為測試而建立的暫存資源。否則，您可能會忘記這些資源，然後，不記得它們用於什麼。到那時，如果您可以安全地擺脫它們，可能還不清楚。

> **Tip**
>
> 若要避免刪除資源時發生相依性錯誤，請務必依下列順序刪除資源：
>
> 1. 影像管線
> 2. 圖片食譜
> 3. 所有剩餘資源

若要清理您為此教學課程建立的資源，請依照下列步驟執行：

刪除配管

1. 若要查看在您的帳戶下建立的建置管線清單，請從導覽窗格中選擇 [映像管線]。

2. 選取「配管名稱」(Pipeline name) 旁的核取方塊，以選取要刪除的配管。
3. 在「影像管線」面板頂端的「動作」功能表上，選擇「刪除」。
4. 若要確認刪除，請Delete在方塊中輸入，然後選擇刪除。

刪除食譜

1. 若要查看您帳戶下建立的食譜清單，請從導覽窗格中選擇 [影像配方]。
2. 選取 [配方名稱] 旁的核取方塊，以選取您要刪除的食譜。
3. 在「影像配方」面板頂端的「動作」功能表上，選擇「刪除方案」。
4. 若要確認刪除，請Delete在方塊中輸入，然後選擇刪除。

刪除基礎結構組

1. 若要查看在您帳戶下建立的基礎結構組態清單，請從導覽窗格中選擇 [基礎結構設定]。
2. 選取 [組態名稱] 旁的核取方塊，以選取您要刪除的基礎結構組態。
3. 在「基礎結構設定」面板頂端，選擇「刪除」。
4. 若要確認刪除，請Delete在方塊中輸入，然後選擇刪除。

刪除散佈設定

1. 若要查看在您的帳戶下建立的發佈設定清單，請從功能窗格中選擇 [發佈設定]。
2. 選取「組態名稱」旁邊的核取方塊，以選取您為此教學課程建立的散佈設定。
3. 在「分佈」設定面板頂端，選擇「刪除」。
4. 若要確認刪除，請Delete在方塊中輸入，然後選擇刪除。

刪除圖像

請遵循下列步驟來確認您已刪除從教學課程管線建立的任何影像。本教學課程不太可能建立映像檔，除非您根據建置排程建立管線以來已經過足夠的時間。

1. 若要查看在您帳戶下建立的影像清單，請從導覽窗格中選擇 [圖片]。
2. 為您要移除的影像選擇映像版本。這會開啟 [映像組建版本] 頁面。
3. 針對您要刪除的任何映像，選取 [版本] 旁邊的核取方塊。您可以一次選取多個映像版本。

4. 在「影像建置版本」面板頂端，選擇「刪除版本」。
5. 若要確認刪除，請Delete在方塊中輸入，然後選擇刪除。

使用 EC2 Image Builder 主控台精靈建立容器映像管道

本教學將逐步引導您建立自動化管道，以使用 [建立映像管道主控台] 精靈來建置和維護自訂 EC2 Image Builder Docker 映像。為了協助您有效率地完成這些步驟，當預設設定可用時，會使用預設設定，並略過可選區段。

建立影像管線工作流

- 步驟 1：指定管線詳細資訊
- 步驟 2：選擇食譜
- 步驟 3：定義基礎結構組態-選用
- 步驟 4：定義發佈設定-選擇性
- 步驟 5：檢視
- 步驟 6：清除

步驟 1：指定管線詳細資訊

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 若要開始建立管線，請選擇 [建立映像管線]。
3. 在「一般」區段中，輸入您的管線名稱 (必填)。
4. 在 [建立排程] 區段中，您可以保留 [排程] 選項的預設值。請注意，預設排程所顯示的時區為世界協調時間 (UTC)。如需有關 UTC 時間的詳細資訊，並尋找您所在時區的偏移量，請參閱時區縮寫 — 全球清單。

   對於「相依性更新」設定，請選擇「如果有相依性更新，則在排程的時間執行管線」選項。此設定會讓您的管道在開始建置之前檢查是否有更新。如果沒有更新，則會略過排程的管線建置。

   > **Note**
   >
   > 若要確保管線如預期般辨識相依性更新和組建，您必須針對基礎映像和元件使用語意版本化 (x.x.x)。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。

5. 選擇 [下一步] 繼續進行下一個步驟。

步驟 2：選擇食譜

1. Image Builder 預設為「使用方案」區段中的現有方案。第一次使用時，請選擇「建立新配方」選項。
2. 在 [映像類型] 區段中，選擇 Docker 映像選項以建立容器管道，以產生 Docker 映像並將其散佈到目標區域中的 Amazon ECR 儲存庫。
3. 在「一般」區段中，輸入下列必要的方塊：

   - 名稱 — 您的食譜名稱
   - 版本-您的食譜版本 (使用格式<major>。 <minor>。 <patch>，其中主要、次要和修補為整數值)。新食譜一般從開始`1.0.0`。

4. 在 [來源映像檔] 區段中，保留 [選取映像檔]、[映像作業系統 (OS)] 和 [映像來源] 的預設值。這會產生由 Amazon 管理的 Amazon Linux 2 容器映像列表，供您從基本映像中選擇。

   a. 從 [影像名稱] 下拉式清單中，選擇影像。

   b. 保留自動版本控制選項的預設值 (使用最新的可用作業系統版本)。

   > **Note**
   >
   > 此設定可確保管線對基礎映像使用語意版本控制，以偵測自動排程工作的相依性更新。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。

5. 在 [元件] 區段中，您必須至少選擇一個組建元件。

   在建置元件 — Amazon Linux 面板中，您可以瀏覽頁面上列出的元件。使用右上角的分頁控制項，瀏覽基本映像作業系統可用的其他元件。您也可以搜尋特定元件，或使用 [元件管理員] 建立您自己的組建元件。

   在此教學課程中，請選擇使用最新安全性更新來更新 Linux 的元件，如下所示：

   a. 在面板頂端的搜尋列`update`中輸入文字，以篩選結果。

   b. 選取`update-linux`建置元件的核取方塊。

   c. 向下捲動，然後在「選取的元件」清單的右上角選擇「全部展開」。

   d. 保留版本控制選項的預設值 (使用最新可用的元件版本)。

> **Note**
>
> 此設定可確保管線對所選元件使用語意版本化，以偵測自動排程工作的相依性更新。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。

如果您選取了具有輸入參數的元件，您也會在此區域中看到參數。本教學課程不介紹參數。如需有關在元件中使用輸入參數，並在方法中設定這些參數的詳細資訊，請參閱使用 EC2 Image Builder 管理 AWS TOE 元件參數。

重新排序元件 (可選)

如果您已選擇要包含在映像中的多個元件，則可以使用 drag-and-drop 動作將它們重新排列為在建置程序期間執行的順序。

> **Note**
>
> CIS 強化元件不遵循 Image Builder 配方中的標準元件排序規則。CIS 強化元件一律會持續執行，以確保基準測試會針對您的輸出影像執行。

1. 向上捲動至可用元件清單。
2. 選取`update-linux-kernel-mainline`建置元件 (或您選擇的任何其他元件) 的核取方塊。
3. 向下捲動至 [選取的元件] 清單，以查看至少有兩個結果。
4. 新增的元件可能未展開其版本控制。若要展開「版本控制」選項，您可以選擇「版本控制」選項旁邊的箭頭，或者您可以將「全部展開」開關切換為關閉和開啟，以展開所有選取元件的版本控制。
5. 選擇其中一個元件，然後向上或向下拖曳以變更元件的執行順序。
6. 若要移除`update-linux-kernel-mainline`元件，請`X`從元件方塊的右上角選擇。
7. 重複上一個步驟，移除您可能已加入的任何其他元件，僅保留選取的`update-linux`元件。

6. 在「Docker 檔案範本」區段中，選取「使用範例」選項。在「內容」面板中，請注意 Image Builder 根據容器映像方案放置組建資訊或指令碼的內容變數。

   依預設，Image Builder 會在 Docker 檔案中使用下列上下文變數。

父母圖像（必填）

在建置階段，此變數會解析為方案的基礎映像檔。

範例：

```
FROM
{{{ imagebuilder:parentImage }}}
```

環境 (如果已指定元件，則需要)

此變數將解析為執行元件的指令碼。

範例：

```
{{{ imagebuilder:environments }}}
```

組件（可選）

Image Builder 會針對容器配方所包含的元件，解析組建和測試元件指令碼。此變數可以放置在 Docker 檔案中的任何位置，位於環境變數之後。

範例：

```
{{{ imagebuilder:components }}}
```

7. 在「目標儲存庫」區段中，指定您建立的 Amazon ECR 儲存庫名稱作為本教學的先決條件。此儲存庫可作為管線執行所在區域 (區域 1) 中發佈組態的預設設定。

> **Note**
>
> 目標儲存庫必須存在於所有目標區域的 Amazon ECR 中，才能發佈。

8. 選擇 [下一步] 繼續進行下一個步驟。

步驟 3：定義基礎結構組態-選用

Image Builder 會在您的帳戶中啟動 EC2 執行個體，以自訂映像並執行驗證測試。基礎結構組態設定會指定建置程序 AWS 帳戶 期間將在您的執行個體中執行的基礎結構詳細資料。

在 [基礎結構組態] 區段中，[組態] 選項預設為`Create infrastructure configuration using service defaults`。這會建立 IAM 角色和關聯的執行個體設定檔，建置執行個體會使用這些設定檔來設定容器映像檔。您也可以建立自己的自訂基礎結構組態，或使用已建立的設定。如需有關基礎設施組態設定的詳細資訊，請參閱 EC2 Image Builder API 參考CreateInfrastructureConfiguration中的。

在本教程中，我們使用的是默認設置。

- 選擇 [下一步] 繼續進行下一個步驟。

步驟 4：定義發佈設定-選擇性

發佈設定由目標區域和目標 Amazon ECR 儲存庫名稱組成。輸出泊塢視窗映像會部署到每個區域中具名的 Amazon ECR 儲存庫。

在 [發佈設定] 區段中，[組態] 選項預設為`Create distribution settings using service defaults`。此選項會將輸出 Docker 映像分發到管道執行所在區域 (區域 1) 的容器配方中指定的 Amazon ECR 儲存庫。如果您選擇`Create new distribution settings`，您可以覆寫目前「區域」的 ECR 存放庫，並新增更多「區域」以進行分佈。

在本教程中，我們使用的是默認設置。

- 選擇 [下一步] 繼續進行下一個步驟。

步驟 5：檢視

「複查」區段會顯示您已設定的所有設定。若要編輯任何指定區段中的資訊，請選擇步驟區段右上角的「編輯」按鈕。例如，如果您要變更管線名稱，請選擇「步驟 1：管線詳細資訊」區段右上角的 「編輯」按鈕。

1. 檢閱設定後，請選擇「建立管道」以建立管道。
2. 您可以在頁面頂端看到成功或失敗訊息，因為您的資源是針對發佈設定、基礎結構組態、新方案和管道建立的。若要查看資源的詳細資訊 (包括資源識別碼)，請選擇 [檢視詳細資料]。

3. 檢視資源的詳細資料之後，您可以從導覽窗格中選擇資源類型，以檢視其他資源的詳細資訊。例如，若要查看新管道的詳細資訊，請從導覽窗格中選擇「影像管線」。如果建置成功，則新管線會顯示在「影像管線」清單中。

步驟 6：清除

您的 Image Builder 環境就像您的家一樣，需要定期維護以幫助您找到所需的內容，並完成任務，而不必涉及混亂。請務必定期清理您為測試而建立的暫存資源。否則，您可能會忘記這些資源，然後，不記得它們用於什麼。到那時，如果您可以安全地擺脫它們，可能還不清楚。

> **Tip**
>
> 若要避免刪除資源時發生相依性錯誤，請務必依下列順序刪除資源：
>
> 1. 影像管線
> 2. 圖片食譜
> 3. 所有剩餘資源

若要清理您為此教學課程建立的資源，請依照下列步驟執行：

刪除配管

1. 若要查看在您的帳戶下建立的建置管線清單，請從導覽窗格中選擇 [映像管線]。
2. 選取「配管名稱」(Pipeline name) 旁的核取方塊，以選取要刪除的配管。
3. 在「影像管線」面板頂端的「動作」功能表上，選擇「刪除」。
4. 若要確認刪除，請`Delete`在方塊中輸入，然後選擇刪除。

刪除容器配方

1. 若要查看您帳戶下建立的容器配方清單，請從功能窗格中選擇 [容器配方]。
2. 選取 [配方名稱] 旁的核取方塊，以選取您要刪除的食譜。
3. 在「容器配方」面板頂端的「動作」功能表上，選擇「刪除方案」。
4. 若要確認刪除，請`Delete`在方塊中輸入，然後選擇刪除。

刪除基礎結構組

1. 若要查看在您帳戶下建立的基礎結構組態清單，請從導覽窗格中選擇 [基礎結構設定]。
2. 選取 [組態名稱] 旁的核取方塊，以選取您要刪除的基礎結構組態。
3. 在「基礎結構設定」面板頂端，選擇「刪除」。
4. 若要確認刪除，請Delete在方塊中輸入，然後選擇刪除。

刪除散佈設定

1. 若要查看在您的帳戶下建立的發佈設定清單，請從功能窗格中選擇 [發佈設定]。
2. 選取「組態名稱」旁邊的核取方塊，以選取您為此教學課程建立的散佈設定。
3. 在「分佈」設定面板頂端，選擇「刪除」。
4. 若要確認刪除，請Delete在方塊中輸入，然後選擇刪除。

刪除圖像

請遵循下列步驟來確認您已刪除從教學課程管線建立的任何影像。本教學課程不太可能建立映像檔，除非您根據建置排程建立管線以來已經過足夠的時間。

1. 若要查看在您帳戶下建立的影像清單，請從導覽窗格中選擇 [圖片]。
2. 為您要移除的影像選擇映像版本。這會開啟 [映像組建版本] 頁面。
3. 針對您要刪除的任何映像，選取 [版本] 旁邊的核取方塊。您可以一次選取多個映像版本。
4. 在「影像建置版本」面板頂端，選擇「刪除版本」。
5. 若要確認刪除，請Delete在方塊中輸入，然後選擇刪除。

AWS 任務協調器和執行器 元件管理員

EC2 Image Builder 使用 AWS 任務協調器和執行器 (AWS TOE) 應用程式協調複雜的工作流程、修改系統組態以及測試系統，而無需撰寫程式碼。此應用程式會管理並執行實作其宣告式文件結構描述的元件。

因為它是獨立的應用程式，所以不需要額外的伺服器設定。它可以在任何雲基礎架構和內部部署上運行。

目錄

AWS TOE 下載

若要安裝 AWS TOE，請選擇您架構和平台的下載連結。如果您連接到服務的 VPC 端點 (例如 Image Builder)，則其必須附加自訂端點政策，其中包含 S3 儲存貯體以供 AWS TOE 下載的存取權。否則，您的構建和測試實例將無法下載 bootstrap 腳本（`bootstrap.sh`）並安裝 AWS TOE 應用程序。如需更多資訊，請參閱為 Image Builder 建立 VPC 端點原則。

> **Important**
>
> AWS 正逐步取消對 TLS 版本 1.0 和 1.1 版的支援。若要存取 S3 儲存貯體以進行 AWS TOE 下載，您的用戶端軟體必須使用 TLS 1.2 版或更新版本。如需詳細資訊，請參閱此AWS 安全性部落格文章。

架構	平台	下載連結	範例
386	阿尔二及 RHEL 7 和 8 Ubuntu 16.04、18.04、20.04 和 CentOS 7 和 8 舒斯 12 號和第 15 號	`https://awstoe-<region>.s3.amazonaws.com/latest/linux/386/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/386/awstoe
AMD64	視窗伺服器	`https://awstoe-<region>.s3.amazonaws.com/latest/windows/amd64/awstoe.exe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/windows/amd64/awstoe.exe
AMD64	阿尔二及 RHEL 7 和 8 Ubuntu 16.04、18.04、20.04 和 CentOS 7 和 8 CentOS Stream 8 舒斯 12 號和第 15 號	`https://awstoe-<region>.s3.amazonaws.com/latest/linux/amd64/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/amd64/awstoe
ARM64	阿尔二及 RHEL 7 和 8 Ubuntu 16.04、18.04、20.04 和 CentOS 7 和 8	`https://awstoe-<region>.s3.amazonaws.com/latest/linux/arm64/awstoe`	https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/arm64/awstoe

架構	平台	下載連結	範例
	CentOS Stream 8 舒斯 12 號和第 15 號		

支援地區

AWS TOE 在下列區域中作為獨立應用程式支援。

AWS 區域 名稱	AWS 區域
美國東部 (俄亥俄)	us-east-2
美國東部 (維吉尼亞北部)	us-east-1
AWS GovCloud (美國東部)	us-gov-east-1
AWS GovCloud (美國西部)	us-gov-west-1
美國西部 (加利佛尼亞北部)	us-west-1
美國西部 (奧勒岡)	us-west-2
非洲 (開普敦)	af-south-1
亞太區域 (香港)	ap-east-1
亞太區域 (大阪)	ap-northeast-3
亞太區域 (首爾)	ap-northeast-2
亞太區域 (孟買)	ap-south-1
亞太區域 (海德拉巴)	ap-south-2
亞太區域 (新加坡)	ap-southeast-1
亞太區域 (雪梨)	ap-southeast-2

AWS 區域 名稱	AWS 區域
亞太區域 (雅加達)	ap-southeast-3
亞太區域 (東京)	ap-northeast-1
加拿大 (中部)	ca-central-1
歐洲 (法蘭克福)	eu-central-1
歐洲 (蘇黎世)	eu-central-2
歐洲 (斯德哥爾摩)	eu-north-1
歐洲 (米蘭)	eu-south-1
歐洲 (西班牙)	eu-south-2
歐洲 (愛爾蘭)	eu-west-1
歐洲 (倫敦)	eu-west-2
歐洲 (巴黎)	eu-west-3
以色列 (特拉維夫)	il-central-1
中東 (阿拉伯聯合大公國)	me-central-1
中東 (巴林)	me-south-1
南美洲 (聖保羅)	sa-east-1
中國 (北京)	cn-north-1
中國 (寧夏)	cn-northwest-1

開始使用 AWS TOE

AWS 任務協調器和執行器 (AWS TOE) 應用程式是獨立的應用程式，可在元件定義架構中建立、驗證和執行指令。 AWS 服務可用 AWS TOE 來協調工作流程、安裝軟體、修改系統組態，以及測試映像檔組建。

請按照以下步驟安裝 AWS TOE 應用程序並首次使用它。

驗證安 AWS TOE 裝下載的簽名

本節說明在 Linux 和 Windows 作業系統 AWS TOE 上驗證安裝下載有效性的建議程序。

主題

在 Linux 上驗證 AWS TOE 安裝下載的簽名

本主題說明驗證 Linux 作業系統 AWS TOE 上安裝下載有效性的建議程序。

每當您從網際網路下載應用程式時，我們建議您驗證軟體發行者的身分。另外，請檢查應用程式是否在發佈後未遭到變更或損毀。如此可保護您，避免安裝到包含病毒或其他惡意程式碼的應用程式版本。

如果在執行本主題中的步驟之後，您判斷的軟體 AWS TOE 已變更或損毀，請勿執行安裝檔案。請聯絡 AWS Support 如需支援選項的詳細資訊，請參閱AWS Support。

AWS TOE 對於基於 Linux 的操作系統的文件使用GnuPG，一個開源實現相當好的隱私（OpenPGP）標準的安全數字簽名進行簽名。 GnuPG(也稱為GPG) 透過數位簽章提供驗證和完整性檢查。Amazon EC2 會發佈公開金鑰和簽章，供您用來驗證下載的 Amazon EC2 CLI 工具。如需 PGP 和 GnuPG (GPG) 的詳細資訊，請參閱 http://www.gnupg.org。

第一步是與軟體發佈者建立信任。下載軟體發佈者的公開金鑰，檢查公開金鑰的擁有者是否為聲稱的擁有者，然後將公開金鑰新增至您的 keyring。您的 keyring 是一組已知的公開金鑰。在您建立公開金鑰的真實性之後，即可用它來驗證應用程式的簽章。

主題

安裝 GPG 工具

如果您的作業系統是 Linux 或 Unix，那麼有可能已安裝 GPG 工具。若要測試工具是否已安裝在您的系統，請在命令提示字元中輸入 gpg。如果 GPG 工具已安裝，您會看到 GPG 命令提示字元。如果未安裝 GPG 工具，您會看到一則錯誤訊息，指出找不到命令。您可以從儲存庫安裝 GnuPG 套件。

在以 Debian 為基礎的 Linux 上安裝 GPG 工具

- 從終端機執行下列命令：apt-get install gnupg。

在以 Red Hat 為基礎的 Linux 上安裝 GPG 工具

- 從終端機執行下列命令：yum install gnupg。

驗證和匯入公開金鑰

此程序的下一個步驟是驗證 AWS TOE 公開金鑰，並將其新增為金鑰GPG圈中的受信任金鑰。

驗證及匯入 AWS TOE 公開金鑰

1. 執行以下其中一項以取得我們的公有 GPG 建置金鑰的副本：

   - 請從以下位置下載金鑰：**<region>**https://awstoe **<region>**. 亞馬遜公司/資產/值得. 例如 https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/assets/awstoe.gpg。
   - 從以下文字複製金鑰，然後貼到名為 awstoe.gpg 的檔案中。務必包含以下所有項目：

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2

mQENBF8UqwsBCACdiRF2bkZYaFSDPFC+LIkWLwFvtUCRwAHtD8KIwTJ6LVn3fHAU
GhuK0ZH9mRrqRT2bq/xJjGsnF9VqTj2AJqndGJdDjz75YCZYM+ocZ+r5HSJaeW9i
S5dykHj7Txti2zHe0G5+W0v7v5bPi2sPHsN7XWQ7+G2AMEPTz8PjxY//I0DvMQns
Sle3l9hz6wCClz1l9LbBzTyHfSm5ucTXvNe88XX5Gmt37OCDM7vfli0Ctv8WFoLN
6jbxuA/sV71yIkPm9IYp3+GvaKeT870+sn8/JOOKE/U4sJV1ppbqmuUzDfhrZUaw
8eW8IN9A1FTIuWiZED/5L83UZuQs1S7s2PjlABEBAAG0GkFXU1RPRSA8YXdzdG9l
QGFtYXpvbi5jb20+iQE5BBMBCAAjBQJfFKsLAhsDBwsJCAcDAgEGFQgCCQoLBBYC
AwECHgECF4AACgkQ3r3BVvWuvFJGiwf9EVmrBR77+Qe/DUeXZJYoaFr7If/fVDZl
6V3TC6p0J0Veme7uXleRUTFOjzbh+7e5sDX19HrnPquzCnzfMiqbp4lSoeUuNdOf
FcpuTCQH+M+sIEIgPno4PLl0Uj2uE1o++mxmonBl/Krk+hly8hB2L/9n/vW3L7BN
OMb1Ll9PmgGPbWipcT8KRdz4SUex9TXGYzjlWb3jU3uXetdaQY1M3kVKE1siRsRN
YYDtpcjmwbhjpu4xm19aFqNoAHCDctEsXJA/mkU3erwIRocPyjAZE2dnlkL9ZkFZ
z9DQkcIarbCnybDM5lemBbdhXJ6hezJE/b17VA0t1fY04MoEkn6oJg==
=oyze
-----END PGP PUBLIC KEY BLOCK-----
```

2. 在您儲存 awstoe.gpg 的目錄中的命令提示字元中，使用下列指令將 AWS TOE 公開金鑰匯入金鑰圈。

```
gpg --import awstoe.gpg
```

此命令會傳回類似以下的結果：

```
gpg: key F5AEBC52: public key "AWSTOE <awstoe@amazon.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
```

請記下金鑰值，在後續步驟您將會用到它。在前面的範例中，金鑰值為 F5AEBC52。

3. 執行以下命令以驗證指紋，請以三個步驟的值取代 key-value：

```
gpg --fingerprint key-value
```

此命令會傳回類似以下的結果：

```
pub   2048R/F5AEBC52 2020-07-19
      Key fingerprint = F6DD E01C 869F D639 15E5  5742 DEBD C156 F5AE BC52
uid       [ unknown] AWSTOE <awstoe@amazon.com>
```

此外，如以上範例所示，指紋字串應該與 F6DD E01C 869F D639 15E5 5742 DEBD C156 F5AE BC52 完全相同。比較傳回的金鑰指紋與此頁面發佈的金鑰指紋。它們應該相符。如果它們不匹配，請不要安裝安 AWS TOE 裝腳本，然後聯繫 AWS Support。

驗證套件的簽章

在您安裝 GPG 工具、驗證及匯入 AWS TOE 公有金鑰，以及驗證公有金鑰可信任之後，您就可以驗證安裝指令碼的簽章。

驗證 安裝指令碼簽章

1. 在命令提示字元中，執行下列命令以下載應用程式二進位檔：

```
curl -O https://awstoe-<region>.s3.<region>.amazonaws.com/latest/
linux/<architecture>/awstoe
```

例如：

```
curl -O https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/amd64/
awstoe
```

支援的值**architecture**可以是amd64386、和arm64。

2. 在命令提示字元中，執行下列命令，從相同的 S3 key prefix 路徑下載對應應用程式二進位檔案的簽章檔案：

```
curl -O https://awstoe-<region>.s3.<region>.amazonaws.com/latest/
linux/<architecture>/awstoe.sig
```

例如：

```
curl -O https://awstoe-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/amd64/
awstoe.sig
```

支援的值**architecture**可以是amd64386、和arm64。

3. 在您儲存的目錄awstoe.sig和 AWS TOE 安裝檔案中的命令提示字元中執行下列命令，以驗證簽章。兩個檔案都必須存在。

```
gpg --verify ./awstoe.sig ~/awstoe
```

輸出應類似以下所示：

```
gpg: Signature made Mon 20 Jul 2020 08:54:55 AM IST using RSA key ID F5AEBC52
gpg: Good signature from "AWSTOE awstoe@amazon.com" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: F6DD E01C 869F D639 15E5 5742 DEBD C156 F5AE BC52
```

如果輸出包含 Good signature from "AWSTOE <awstoe@amazon.com>" 片語，表示簽章已成功驗證，您可以繼續執行 AWS TOE 安裝指令碼。

如果輸出包含 BAD signature 片語，請檢查您是否已正確執行程序。如果您繼續收到此回應，請勿執行先前下載的安裝檔案，然後連絡 AWS Support。

以下是您可能會看到的警告的詳細資訊：

- 警告：此密鑰未通過受信任的簽名進行認證！沒有跡象表明該簽名屬於擁有者。理想情況下，您可以親自訪問 AWS 辦公室並收到鑰匙。但是，您很可能會從網站下載它。在這種情況下，該網站是一個 AWS 網站。
- gpg：沒有發現最終信任的金鑰。這意味著特定密鑰不是由您或您信任的其他人「最終信任」。

如需詳細資訊，請參閱 http://www.gnupg.org。

在 Windows 上驗證 AWS TOE 安裝下載的簽名

本主題說明在 Windows 作業系統上驗證 AWS 任務協調器和執行器 應用程式之安裝檔案有效性的建議程序。

當您從網際網路下載應用程式時，建議您驗證軟體發佈者的身分，並檢查應用程式在發佈之後未遭更改或損毀。如此可保護您，避免安裝到包含病毒或其他惡意程式碼的應用程式版本。

如果在執行本主題中的步驟之後，您判斷 AWS TOE 應用程式的軟體已變更或損毀，請勿執行安裝檔案。相反，請聯繫 AWS Support。

若要驗證 Windows 作業系統上下載的 awstoe 二進位檔案的有效性，請確定其 Amazon Services LLC 簽署者憑證的指紋等於此值：

F8 11 EE F0 4A A 91 E3 79 21 BA 6B 自動對焦 FC 19 92 12 D7

> **Note**
>
> 在新二進位檔的推出視窗期間，您的簽署者憑證可能不符合新的指紋。如果您的簽署者憑證不相符，請確認指紋值為：
> 5B 77 F4 F0 F0 C3 7A 8B 89 D9 A7 八樓 54 B6 85 11 CE 9E A3 BF 17

若要驗證這個值，請執行以下程序：

1. 在下載的 `awstoe.exe` 上按一下滑鼠右鍵，然後開啟 Properties (屬性) 視窗。
2. 選擇 數位簽章 索引標籤。
3. 從 Signature List (簽章清單) 中選擇 Amazon Services LLC，然後選擇 Details (詳細資訊)。
4. 選擇 General (一般) 索引標籤 (如果尚未選取)，然後選擇 View Certificate (檢視憑證)。
5. 選擇 [詳細資料] 索引標籤，然後在 [顯示] 下拉式清單中選擇 [全部] (如果尚未選取)。

6. 向下捲動到看見 Thumbprint (指紋) 欄位為止，然後選擇 Thumbprint (指紋)。這會在下方的視窗中顯示整個指紋值。

   - 如果下方視窗中的指紋值與以下值完全相同：

     F8 11 EE F0 4A A 91 E3 79 21 BA 6B 自動對焦 FC 19 92 12 D7

     那麼你下載的 AWS TOE 二進製文件是真實的，可以安全地安裝。

     > **Note**
     >
     > 在新二進位檔的推出視窗期間，您的簽署者憑證可能不符合新的指紋。如果您的簽署者憑證不相符，請確認指紋值為：
     > 5B 77 F4 F0 F0 C3 7A 8B 89 D9 A7 八樓 54 B6 85 11 CE 9E A3 BF 17

   - 如果下方詳細資料視窗中的指紋值與先前的值不同，請勿執行`awstoe.exe`。

開始使用步驟

- 步驟 1：安裝 AWS TOE
- 步驟 2：設定 AWS 認證
- 步驟 3：在本機開發元件文件
- 步驟 4：驗證 AWS TOE 元件
- 步驟 5：執行 AWS TOE 元件

步驟 1：安裝 AWS TOE

若要在本機開發元件，請下載並安裝 AWS TOE 應用程式。

1. 下載 AWS TOE 應用程式

   若要安裝 AWS TOE，請選擇適合您架構和平台的下載連結。如需應用程式下載連結的完整清單，請參閱 AWS TOE 下載

> **Important**
>
> AWS 正逐步取消對 TLS 版本 1.0 和 1.1 版的支援。若要存取 S3 儲存貯體以進行 AWS TOE 下載，您的用戶端軟體必須使用 TLS 1.2 版或更新版本。如需詳細資訊，請參閱此AWS 安全性部落格文章。

2. 驗證簽名

   驗證下載的步驟取決於您在安裝 AWS TOE 應用程式之後執行應用程式的伺服器平台。若要驗證您在 Linux 伺服器上的下載，請參閱在 Linux 上驗證簽名。若要確認您在 Windows 伺服器上的下載作業，請參閱驗證視窗上的簽名。

> **Important**
>
> AWS TOE 直接從其下載位置調用。不需要單獨的安裝步驟。這也意味著 AWS TOE 可以對本地環境進行更改。
> 為了確保您在元件開發期間隔離變更，建議您使用 EC2 執行個體來開發和測試 AWS TOE 元件。

步驟 2：設定 AWS 認證

AWS TOE 執行任務時 AWS 服務，需要 AWS 登入資料才能連線到其他資料 CloudWatch，例如 Amazon S3 和 Amazon，例如：

- 從使用者提供的 Amazon S3 路徑下載 AWS TOE 文件。
- 運行`S3Download`或`S3Upload`動作模塊。
- 啟用時 CloudWatch，將記錄串流至。

如果您在 EC2 執行個體 AWS TOE 上執行，則執行 AWS TOE 會使用與連接至 EC2 執行個體的 IAM 角色相同的許可。

如需適用於 EC2 的 IAM 角色的詳細資訊，請參閱適用於 Amazon EC2 的 IAM 角色。

下列範例顯示如何使用`AWS_ACCESS_KEY_ID`和`AWS_SECRET_ACCESS_KEY`環境變數設定 AWS 認證。

若要在 Linux、macOS 或 Unix 上設定這些變數，請使用`export`。

```
$ export AWS_ACCESS_KEY_ID=your_access_key_id
```

```
$ export AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

若要在 Windows 上使用設定這些變數 PowerShell，請使用`$env`。

```
C:\> $env:AWS_ACCESS_KEY_ID=your_access_key_id
```

```
C:\> $env:AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

若要使用命令提示字元在 Windows 上設定這些變數，請使用`set`。

```
C:\> set AWS_ACCESS_KEY_ID=your_access_key_id
```

```
C:\> set AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

步驟 3：在本機開發元件文件

AWS TOE 元件是使用純文字 YAML 文件建立的。如需文件語法的詳細資訊，請參閱在中使用零組件文件 AWS TOE。

以下是您可以用來在本機開發文件的範例 Hello World 元件文件。

`hello-world-windows.yml`.

```
name: Hello World
description: This is Hello World testing document for Windows.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
```

```
            - Write-Host 'Hello World from the build phase.'
  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host 'Hello World from the validate phase.'
  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host 'Hello World from the test phase.'
```

`hello-world-linux.yml`.

```
name: Hello World
description: This is hello world testing document for Linux.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo 'Hello World from the build phase.'
  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo 'Hello World from the validate phase.'
  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo 'Hello World from the test phase.'
```

步驟 4：驗證 AWS TOE 元件

您可以使 AWS TOE 用應用程式在本機驗證 AWS TOE 元件的語法。下列範例顯示 AWS TOE 應用程式`validate`命令，以驗證元件的語法而不執行元件。

> **Note**
>
> AWS TOE 應用程式只能驗證目前作業系統的元件語法。例如，`awstoe.exe`在 Windows 上執行時，您無法驗證使用`ExecuteBash`動作模組之 Linux 文件的語法。

Windows

```
C:\> awstoe.exe validate --documents C:\Users\user\Documents\hello-world.yml
```

Linux

```
$ awstoe validate --documents /home/user/hello-world.yml
```

步驟 5：執行 AWS TOE 元件

AWS TOE 應用程序可以使用`--phases`命令行參數運行指定文檔的一個或多個階段。支援的`--phases`值為`build`、`validate`、和`test`。可以將多個相位值輸入為逗號分隔值。

當您提供階段清單時，應用程式會依 AWS TOE 序執行每個文件的指定階段。例如， AWS TOE 執行的`build`和`validate`階段`document1.yaml`，然後執行的`build`和`validate`階段`document2.yaml`。

為確保您的日誌安全存放並保留以便進行故障診斷，建議您在 Amazon S3 中設定日誌儲存。在 Image Builder 中，用於發佈日誌的 Amazon S3 位置是在基礎設施組態中指定的。如需基礎結構組態的詳細資訊，請參閱 管理 EC2 Image Builder 基礎設施組

如果未提供階段清單， AWS TOE 應用程式會依照 YAML 文件中列出的順序執行所有階段。

若要在單一或多個文件中執行特定階段，請使用下列指令。

單相

```
awstoe run --documents hello-world.yml --phases build
```

多個階段

```
awstoe run --documents hello-world.yml --phases build,test
```

文件執行

在單一文件中執行所有階段

```
awstoe run --documents documentName.yaml
```

在多個文檔中運行所有階段

```
awstoe run --documents documentName1.yaml,documentName2.yaml
```

輸入 Amazon S3 資訊以從使用者定義的本機路徑上傳 AWS TOE 日誌 (建議使用)

```
awstoe run --documents documentName.yaml --log-s3-bucket-name <S3Bucket> --log-s3-key-
prefix <S3KeyPrefix> --log-s3-bucket-owner <S3BucketOwner> --log-directory <local_path>
```

在單個文檔中運行所有階段，並在控制台上顯示所有日誌

```
awstoe run --documents documentName.yaml --trace
```

範例 命令

```
awstoe run --documents s3://bucket/key/doc.yaml --phases build,validate
```

使用唯一 ID 執行文件

```
awstoe run --documents <documentName>.yaml --execution-id <user provided id> --phases
 <comma separated list of phases>
```

取得協助 AWS TOE

```
awstoe --help
```

在中使用零組件文件 AWS TOE

若要使用 AWS 任務協調器和執行器 (AWS TOE) 建置元件，您必須提供代表適用於您建立之元件的階段和步驟的 YAML 文件。 AWS 服務 在建立新的 Amazon 機器映像 (AMI) 或容器映像時使用您的元件。

主題

元件文件工作流

AWS TOE 元件文件會使用階段和步驟將相關工作分組，並將這些任務組織成元件的邏輯工作流程。

> **Tip**
>
> 使用元件建置映像檔的服務可能會實作規則，說明要用於其建置程序的階段，以及允許執行這些階段的時間。設計元件時，這一點很重要。

階段

階段代表工作流程在映像建置程序中的進度。例如，Image Builder 服務會在其建置*validate*階段針對其產生的影像使用build和階段。它在測試*container-host-test*階段使用test和階段，以確保映像快照或容器映像在創建最終 AMI 或分發容器映像之前產生預期的結果。

當元件執行時，每個階段的關聯指令將按照它們在元件文件中的顯示順序來套用。

階段規則

- 每個階段名稱在文件中都必須是唯一的。
- 您可以在文件中定義許多階段。

- 您必須至少在文件中包含下列其中一個階段：
  - build-對於 Image Builder，此階段通常在構建階段使用。
  - validate — 對於 Image Builder，此階段通常會在建置階段使用。
  - test — 對於 Image Builder，此階段通常在測試階段使用。
- 階段總是按照它們在文檔中定義的順序運行。針對中的指 AWS TOE 令指定它們的順序不 AWS CLI 起作用。

步驟

步驟是定義每個階段中工作流程的個別工作單位。步驟會循序執行。但是，一個步驟的輸入或輸出也可以進入後續步驟作為輸入。這就是所謂的「鏈接」。

步驟規則

- 階段的步驟名稱必須是唯一的。
- 步驟必須使用支援的動作 (動作模組) 來傳回結束代碼。

  如需支援的動作模組、其運作方式、輸入/輸出值和範例的完整清單，請參閱AWS TOE 元件管理員支援的動作模組。

元件記錄

AWS TOE 每次執行元件時，會在 EC2 執行個體上建立新的記錄檔資料夾，用於建置和測試新映像。對於容器映像檔，記錄檔資料夾會儲存在容器中。

若要在影像建立過程中發生問題時協助進行疑難排解，輸入文件和所有在執行元件時 AWS TOE 建立的輸出檔案都會儲存在記錄檔資料夾中。

記錄檔資料夾名稱由下列部分組成：

1. 記錄目錄 — 當服務執行 AWS TOE 元件時，它會傳入 log 目錄，以及指令的其他設定。在下列範例中，我們展示了 Image Builder 使用的記錄檔格式。
   - Linux: `/var/lib/amazon/toe/`
   - Windows: `$env:ProgramFiles\Amazon\TaskOrchestratorAndExecutor\`
2. 文件前綴-這是用於所有組件的標準前綴："`TOE_`」。
3. 執行時間 — 此時間戳記格式為 YYYY-MM-MM-MM-SS_UTC-0 格式的時間戳記。
4. 執行 ID — 這是執行一或多個元件 AWS TOE 時指派的 GUID。

範例：*/var/lib/amazon/toe/*TOE*_2021-07-01_12-34-56_UTC-0_a1bcd2e3-45f6-789a-bcde-0fa1b2c3def4*

AWS TOE 將下列核心檔案儲存在記錄檔資料夾中：

輸入文件

- 文件 .yaml — 用來做為指令輸入的文件。元件執行後，此檔案會儲存為人工因素。

輸出文件

- application.log — 應用程式記錄檔包含時間戳記的偵錯層級資訊，來自 AWS TOE 元件執行時發生的情況。
- detailedoutput.json — 此 JSON 檔案包含有關執行狀態、輸入、輸出和失敗的詳細資訊，適用於元件執行時適用的所有文件、階段和步驟。
- console.log — 主控台記錄檔包含元件執行時 AWS TOE 寫入主控台的所有標準輸出 (stdout) 和標準錯誤 (stderr) 資訊。
- 鏈接 .json — 此 JSON 文件表示 AWS TOE 應用於解析鏈接表達式的優化。

> **Note**
>
> 記錄檔資料夾也可能包含此處未涵蓋的其他暫存檔案。

輸入和輸出鏈接

AWS TOE 組態管理應用程式會以下列格式撰寫參考，提供連結輸入和輸出的功能：

```
{{ phase_name.step_name.inputs/outputs.variable }}
```

或

```
{{ phase_name.step_name.inputs/outputs[index].variable }}
```

鏈接功能允許您回收代碼並提高文檔的可維護性。

鏈接規則

- 鏈結運算式只能在每個步驟的輸入部分中使用。

- 具有鏈結運算式的陳述式必須以引號括起來。例如：
  - 無效的表示式:`echo {{ phase.step.inputs.variable }}`
  - 有效表示式："`echo {{ phase.step.inputs.variable }}`"
  - 有效表示式：'`echo {{ phase.step.inputs.variable }}`'
- 鏈結運算式可以參考相同文件中其他步驟和階段的變數。不過，呼叫服務可能有規則，需要鏈結運算式只能在單一階段的內容中運作。例如，Image Builder 不支援從建置階段到測試階段的鏈結，因為它會獨立執行每個階段。
- 鏈結運算式中的索引遵循從零開始的索引。索引從零（0）開始引用第一個元素。

範例

若要參照下列範例步驟第二個項目中的 source 變數，鏈結模式為`{{ build.SampleS3Download.inputs[1].source }}`。

```
phases:
  -
    name: 'build'
    steps:
      -
        name: SampleS3Download
        action: S3Download
        timeoutSeconds: 60
        onFailure: Abort
        maxAttempts: 3
        inputs:
          -
            source: 's3://sample-bucket/sample1.ps1'
            destination: 'C:\sample1.ps1'
          -
            source: 's3://sample-bucket/sample2.ps1'
            destination: 'C:\sample2.ps1'
```

若要參照下列範例步驟的輸出變數 (等於「Hello」)，鏈結模式為`{{ build.SamplePowerShellStep.outputs.stdout }}`。

```
phases:
  -
    name: 'build'
    steps:
```

```
      -
        name: SamplePowerShellStep
        action: ExecutePowerShell
        timeoutSeconds: 120
        onFailure: Abort
        maxAttempts: 3
        inputs:
          commands:
            - 'Write-Host "Hello"'
```

文件結構描述和定義

以下是文件的 YAML 結構描述。

```
name: (optional)
description: (optional)
schemaVersion: "string"

phases:
  - name: "string"
    steps:
        - name: "string"
          action: "string"
          timeoutSeconds: integer
          onFailure: "Abort|Continue|Ignore"
          maxAttempts: integer
          inputs:
```

文件的結構描述定義如下。

欄位	Description (描述)	Type	必要
name	文件的名稱。	字串	否
description	文件的描述。	字串	否
schemaVersion	文件的結構描述版本，目前為 1.0。	字串	是
階段	階段及其步驟的列表。	清單	是

階段的結構描述定義如下。

欄位	Description (描述)	Type	必要
name	階段的名稱。	字串	是
steps	階段中的步驟清單。	清單	是

步驟的結構描述定義如下。

欄位	Description (描述)	Type	必要	預設值
name	步驟的使用者定義名稱。	字串		
動作	與執行步驟之模組相關的關鍵字。	字串		
timeoutSeconds	失敗或重試之前，步驟執行的秒數。 此外，支持 -1 值，這表示無限超時。0 和其他負值是不允許的。	Integer	否	七百二十秒 (一百二十分鐘)
onFailure	指定步驟在失敗時應執行的動作。有效值如下： • 中止 — 嘗試次數上限後的步驟失敗，	字串	否	中止

欄位	Description (描述)	Type	必要	預設值
	並停止執行。將階段與文件的狀態設定為Failed。 • 繼續 — 在嘗試次數上限之後失敗步驟，並繼續執行剩餘步驟。將階段與文件的狀態設定為Failed。 • 略過 — 將步驟設定為超過失敗嘗試次數上限IgnoredFailure 之後的步驟，並繼續執行剩餘步驟。將階段與文件的狀態設定為SuccessWithIgnoredFailure 。			
maxAttempts	在步驟失敗之前允許的嘗試次數上限。	Integer	否	1
inputs	包含動作模組執行步驟所需的參數。	字典	是	

文件範例結構

以下是範例文件結構描述，用於安裝所有可用的 Windows 更新、執行組態指令碼、在建立 AMI 之前驗證變更，以及在建立 AMI 之後測試變更。

```
name: RunConfig_UpdateWindows
description: 'This document will install all available Windows updates and run a config
 script. It will then validate the changes before an AMI is created. Then after AMI
 creation, it will test all the changes.'
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: DownloadConfigScript
        action: S3Download
        timeoutSeconds: 60
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - source: 's3://customer-bucket/config.ps1'
            destination: 'C:\config.ps1'

      - name: RunConfigScript
        action: ExecutePowerShell
        timeoutSeconds: 120
        onFailure: Abort
        maxAttempts: 3
        inputs:
          file: '{{build.DownloadConfigScript.inputs[0].destination}}'

      - name: Cleanup
        action: DeleteFile
        onFailure: Abort
        maxAttempts: 3
        inputs:
          - path: '{{build.DownloadConfigScript.inputs[0].destination}}'

      - name: RebootAfterConfigApplied
        action: Reboot
        inputs:
          delaySeconds: 60

      - name: InstallWindowsUpdates
```

```
      action: UpdateOS

- name: validate
  steps:
    - name: DownloadTestConfigScript
      action: S3Download
      timeoutSeconds: 60
      onFailure: Abort
      maxAttempts: 3
      inputs:
        - source: 's3://customer-bucket/testConfig.ps1'
          destination: 'C:\testConfig.ps1'

    - name: ValidateConfigScript
      action: ExecutePowerShell
      timeoutSeconds: 120
      onFailure: Abort
      maxAttempts: 3
      inputs:
        file: '{{validate.DownloadTestConfigScript.inputs[0].destination}}'

    - name: Cleanup
      action: DeleteFile
      onFailure: Abort
      maxAttempts: 3
      inputs:
        - path: '{{validate.DownloadTestConfigScript.inputs[0].destination}}'

- name: test
  steps:
    - name: DownloadTestConfigScript
      action: S3Download
      timeoutSeconds: 60
      onFailure: Abort
      maxAttempts: 3
      inputs:
        - source: 's3://customer-bucket/testConfig.ps1'
          destination: 'C:\testConfig.ps1'

    - name: ValidateConfigScript
      action: ExecutePowerShell
      timeoutSeconds: 120
      onFailure: Abort
      maxAttempts: 3
```

```
        inputs:
          file: '{{test.DownloadTestConfigScript.inputs[0].destination}}'
```

以下是下載並執行自訂 Linux 二進位檔案的範例文件結構描述。

```
name: LinuxBin
description: Download and run a custom Linux binary file.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: Download
        action: S3Download
        inputs:
          - source: s3://<replaceable>mybucket</replaceable>/
<replaceable>myapplication</replaceable>
            destination: /tmp/<replaceable>myapplication</replaceable>
      - name: Enable
        action: ExecuteBash
        onFailure: Continue
        inputs:
          commands:
            - 'chmod u+x {{ build.Download.inputs[0].destination }}'
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: '{{ build.Download.inputs[0].destination }}'
          arguments:
            - '--install'
      - name: Delete
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

以下是使用安裝檔案在 Windows 執行個體 AWS CLI 上安裝的範例文件結構描述。

```
name: InstallCLISetUp
description: Install &CLI; using the setup file
schemaVersion: 1.0
phases:
  - name: build
    steps:
```

```
      - name: Download
        action: S3Download
        inputs:
          - source: s3://aws-cli/AWSCLISetup.exe
            destination: C:\Windows\temp\AWSCLISetup.exe
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: '{{ build.Download.inputs[0].destination }}'
          arguments:
            - '/install'
            - '/quiet'
            - '/norestart'
      - name: Delete
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

以下是 AWS CLI 使用 MSI 安裝程式安裝的範例文件結構描述。

```
name: InstallCLIMSI
description: Install &CLI; using the MSI installer
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: Download
        action: S3Download
        inputs:
          - source: s3://aws-cli/AWSCLI64PY3.msi
            destination: C:\Windows\temp\AWSCLI64PY3.msi
      - name: Install
        action: ExecuteBinary
        onFailure: Continue
        inputs:
          path: 'C:\Windows\System32\msiexec.exe'
          arguments:
            - '/i'
            - '{{ build.Download.inputs[0].destination }}'
            - '/quiet'
            - '/norestart'
      - name: Delete
```

```
        action: DeleteFile
        inputs:
          - path: '{{ build.Download.inputs[0].destination }}'
```

定義和引用變量 AWS TOE

變數提供了一種方法，以有意義的名稱來標記資料，這些名稱可用於整個應用程式。您可以為複雜的工作流程定義具有簡單易讀格式的自訂變數，並在元件的 YAML 應用程式元件文件中參考這些變數。AWS TOE

本節提供的資訊可協助您在 YAML 應用程式 AWS TOE 元件文件中定義元件的變數，包括語法、名稱條件約束和範例。

參數

參數是可變變數，具有呼叫應用程式可在執行階段提供的設定。您可以在 YAML 文件的`Parameters`區段中定義參數。

參數名稱的規則

- 名稱長度必須介於 3 到 128 個字元之間。
- 名稱只能包含英數字元 (a-z、A-Z、0-9)、破折號 (-) 或底線 (_)。
- 名稱在文件中必須是唯一的。
- 名稱必須指定為 YAML 字串。

語法

```
parameters:
  - <name>:
      type: <parameter type>
      default: <parameter value>
      description: <parameter description>
```

金鑰名稱	必要	描述
`name`	是	參數名稱。對於文檔必須是唯一的（它不能與任何其他參數名稱或常量相同）。

金鑰名稱	必要	描述
type	是	參數的資料類型。支援的類型包括：string。
default	否	參數的預設值。
description	否	描述參數。

參照文件中的參數值

您可以參考 YAML 文件內的步驟或迴圈輸入參數，如下所示：

- 參數參照區分大小寫，且名稱必須完全相符。
- 名稱必須用雙大括號括起來{{*MyParameter*}}。
- 大括號內允許空格，並且會自動修剪。例如，下列所有參照都是有效的：

  {{ *MyParameter* }}, {{ *MyParameter*}}, {{*MyParameter* }}, {{*MyParameter*}}
- YAML 文件中的參考必須指定為字串 (以單引號或雙引號括住)。

  例如：- {{ *MyParameter* }}無效，因為它未標識為字符串。

  但是，下列參照都是有效的：- '{{ *MyParameter* }}'和- "{{ *MyParameter* }}"。

範例

下列範例說明如何在 YAML 文件中使用參數：

- 請參照步驟輸入中的參數：

```
name: Download AWS CLI version 2
schemaVersion: 1.0
parameters:
  - Source:
      type: string
      default: 'https://awscli.amazonaws.com/AWSCLIV2.msi'
      description: The AWS CLI installer source URL.
phases:
  - name: build
    steps:
```

```
      - name: Download
        action: WebDownload
        inputs:
          - source: '{{ Source }}'
            destination: 'C:\Windows\Temp\AWSCLIV2.msi'
```

- 參考循環輸入中的參數：

```
name: PingHosts
schemaVersion: 1.0
parameters:
  - Hosts:
      type: string
      default: 127.0.0.1,amazon.com
      description: A comma separated list of hosts to ping.
phases:
  - name: build
    steps:
      - name: Ping
        action: ExecuteBash
        loop:
          forEach:
            list: '{{ Hosts }}'
            delimiter: ','
        inputs:
          commands:
            - ping -c 4 {{ loop.value }}
```

在運行時覆蓋參數

您可以使用中的`--parameters`選項搭 AWS CLI 配鍵值配對，在執行階段設定參數值。

- 將參數鍵值配對指定為名稱和值，並以等號 (<name>=<value>) 分隔。
- 多個參數必須以逗號分隔。
- 在 YAML 元件文件中找不到的參數名稱會被忽略。
- 參數名稱和值都是必需的。

> **Important**
>
> 元件參數是純文字值，並已登入 AWS CloudTrail。我們建議您使用 AWS Secrets Manager 或 AWS Systems Manager 參數存放區來儲存您的秘密。如需有關 Secrets Manager 的詳細資訊，請參閱什麼是 Secrets Manager？ 在《AWS Secrets Manager 使用者指南》中。若要取得有關 AWS Systems Manager 參數存放區的更多資訊，請AWS Systems Manager 參閱AWS Systems Manager 使用指南中的參數存放區

語法

```
--parameters name1=value1,name2=value2...
```

CLI 選項	必要	描述
-參數## = #，...	否	此選項採用鍵值對的列表，參數名稱作為關鍵字。

範例

下列範例說明如何在 YAML 文件中使用參數：

- 在此--parameter選項中指定的參數鍵值對無效：

```
--parameters ntp-server=
```

- 使用以下--parameter選項設定一個參數鍵值對： AWS CLI

```
--parameters ntp-server=ntp-server-windows-qe.us-east1.amazon.com
```

- 使用以下--parameter選項設定多個參數鍵值對： AWS CLI

```
--parameters ntp-server=ntp-server.amazon.com,http-url=https://internal-us-
east1.amazon.com
```

常數

常量是不可變的變量，一旦定義不能被修改或覆蓋。常數可以使用 AWS TOE 文檔的`constants`部分中的值來定義。

常數名稱的規則

- 名稱長度必須介於 3 到 128 個字元之間。
- 名稱只能包含英數字元 (a-z、A-Z、0-9)、破折號 (-) 或底線 (_)。
- 名稱在文件中必須是唯一的。
- 名稱必須指定為 YAML 字串。

語法

```
constants:
  - <name>:
      type: <constant type>
      value: <constant value>
```

金鑰名稱	必要	描述
`name`	是	常數的名稱。對於文檔必須是唯一的（它不能與任何其他參數名稱或常量相同）。
`value`	是	常數的值。
`type`	是	常數的類型。支援的類型為`string`。

引用文檔中的常量值

您可以在 YAML 文件內的步驟或迴圈輸入中參考常數，如下所示：

- 常量引用區分大小寫，並且名稱必須完全匹配。
- 名稱必須用雙大括號括起來`{{MyConstant}}`。
- 大括號內允許空格，並且會自動修剪。例如，下列所有參照都是有效的：

{{ *MyConstant* }}, {{ *MyConstant*}}, {{*MyConstant* }}, {{*MyConstant*}}

- YAML 文件中的參考必須指定為字串 (以單引號或雙引號括住)。

  例如：- {{ *MyConstant* }}無效，因為它未標識為字符串。

  但是，下列參照都是有效的：- '{{ *MyConstant* }}'和- "{{ *MyConstant* }}"。

範例

步驟輸入中參照的常數

```
name: Download AWS CLI version 2
schemaVersion: 1.0
constants:
  - Source:
      type: string
      value: https://awscli.amazonaws.com/AWSCLIV2.msi
phases:
  - name: build
    steps:
      - name: Download
        action: WebDownload
        inputs:
          - source: '{{ Source }}'
            destination: 'C:\Windows\Temp\AWSCLIV2.msi'
```

循環輸入中引用的常量

```
name: PingHosts
schemaVersion: 1.0
constants:
  - Hosts:
      type: string
      value: 127.0.0.1,amazon.com
phases:
  - name: build
    steps:
      - name: Ping
        action: ExecuteBash
        loop:
          forEach:
```

```
            list: '{{ Hosts }}'
            delimiter: ','
        inputs:
          commands:
            - ping -c 4 {{ loop.value }}
```

在中使用循環構造 AWS TOE

本節提供的 AWS TOE資訊可協助您在中建立迴圈建構。循環構造定義指令的重複序列。您可以在 AWS TOE中使用下列類型的迴圈建構：

- for構造 — 迭代有界的整數序列。
- forEach構造
  - forEach循環輸入列表-迭代字符串的有限集合。
  - forEach使用分隔符號清單的迴圈 — 以分隔符號連接的有限字串集合進行迭代。

> **Note**
>
> 迴圈建構只支援字串資料類型。

循環構建主題

- 引用迭代變量
- 循環構造的類型
- 步驟欄位
- 步驟和迭代輸出

引用迭代變量

若要參照目前反覆運算式變數的索引和值，{{ loop.* }}必須在包含迴圈結構之步驟的輸入主體內使用參照運算式。這個表達式不能用來引用另一個步驟的循環構造的迭代變量。

參照運算式由下列成員組成：

- {{ loop.index }}-當前迭代的序數位置，該迭代的索引位0置。
- {{ loop.value }}— 與目前反覆運算變數相關聯的值。

循環名稱

所有循環結構都有一個可選的名稱字段進行標識。如果提供了一個循環名稱，它可用於在步驟的輸入體中引用迭代變量。要引用命名循環的迭代索引和值，請{{ loop.* }}在步驟的輸入主體中使用{{ <loop_name>.* }}。這個表達式不能用來引用另一個步驟的命名循環構造。

參照運算式由下列成員組成：

- {{ <loop_name>.index }}— 具名迴圈之目前反覆運算的序數位置，其在編製索引處0。
- {{ <loop_name>.value }}— 與具名迴圈的目前反覆運算變數相關聯的值。

解析參考運算式

會 AWS TOE 解析參照運算式，如下所示：

- {{ <loop_name>.* }}— 使用下列邏輯 AWS TOE 解析此運算式：
  - 如果目前執行中步驟的迴圈與<loop_name>值相符，則參考運算式會解析為目前執行中步驟的迴圈結構。
  - <loop_name>解析為命名的循環結構（如果它出現在當前正在運行的步驟中）。
- {{ loop.* }}— 使用目前執行中步驟中定義的迴圈建構來 AWS TOE 解析運算式。

如果在不包含迴圈的步驟中使用參照運算式，則 AWS TOE 不會解析運算式，且它們會出現在步驟中，而不會取代。

> **Note**
>
> 參考運算式必須以雙引號括住，才能由 YAML 編譯器正確解譯。

循環構造的類型

本節提供有關可在中使用之迴圈建構類型的資訊和範例 AWS TOE。

循環構造類型

- for循環
- forEach循環輸入列表
- forEach循環與分隔列表

for循環

for循環迭代範圍內由變量的開始和結束概述的邊界內指定的整數。迭代值在集[start, end]合中，包括邊界值。

AWS TOE 驗證start、和updateBy值end，以確保組合不會產生無限迴圈。

for循環模式

```
  - name: "StepName"
    action: "ActionModule"
    loop:
      name: "string"
      for:
        start: int
        end: int
        updateBy: int
inputs:
  ...
```

for循環輸入

欄位	Description (描述)	Type	必要	預設
name	循環的唯一名稱。與同一階段中的其他循環名稱相比，它必須是唯一的。	字串	否	""
start	迭代的起始值。不接受鏈結運算式。	Integer	是	N/A
end	迭代的結束值。不接受鏈結運算式。	Integer	是	N/A
updateBy	迭代值通過加法更新的差異。它	Integer	是	N/A

欄位	Description (描述)	Type	必要	預設
	必須是負值或正非零值。不接受鏈結運算式。			

for循環輸入示例

```
  - name: "CalculateFileUploadLatencies"
    action: "ExecutePowerShell"
    loop:
      for:
        start: 100000
        end: 1000000
        updateBy: 100000
    inputs:
      commands:
        - |
          $f = new-object System.IO.FileStream c:\temp\test{{ loop.index }}.txt,
 Create, ReadWrite
          $f.SetLength({{ loop.value }}MB)
          $f.Close()
        - c:\users\administrator\downloads\latencyTest.exe --file c:\temp
\test{{ loop.index }}.txt
        - AWS s3 cp c:\users\administrator\downloads\latencyMetrics.json s3://bucket/
latencyMetrics.json
        - |
          Remove-Item -Path c:\temp\test{{ loop.index }}.txt
          Remove-Item -Path c:\users\administrator\downloads\latencyMetrics.json
```

forEach循環輸入列表

forEach循環迭代一個明確的值列表，它可以是字符串和鏈接表達式。

forEach循環輸入列表模式

```
  - name: "StepName"
    action: "ActionModule"
    loop:
      name: "string"
      forEach:
```

```
        - "string"
    inputs:
  ...
```

forEach循環輸入列表輸入

欄位	Description (描述)	Type	必要	預設
name	循環的唯一名稱。與同一階段中的其他循環名稱相比，它必須是唯一的。	字串	否	""
forEach循環字符串列表	迭代字符串列表。接受鏈結的表達式作為列表中的字符串。鏈結運算式必須以雙引號括起來，YAML 編譯器才能正確解譯它們。	字串清單	是	N/A

forEach循環輸入列表示例 1

```
- name: "ExecuteCustomScripts"
  action: "ExecuteBash"
  loop:
    name: BatchExecLoop
    forEach:
      - /tmp/script1.sh
      - /tmp/script2.sh
      - /tmp/script3.sh
  inputs:
    commands:
      - echo "Count {{ BatchExecLoop.index }}"
      - sh "{{ loop.value }}"
```

```
      - |
        retVal=$?
        if [ $retVal -ne 0 ]; then
          echo "Failed"
        else
          echo "Passed"
       fi
```

forEach循環輸入列表示例 2

```
- name: "RunMSIWithDifferentArgs"
  action: "ExecuteBinary"
  loop:
    name: MultiArgLoop
    forEach:
      - "ARG1=C:\Users ARG2=1"
      - "ARG1=C:\Users"
      - "ARG1=C:\Users ARG3=C:\Users\Administrator\Documents\f1.txt"
  inputs:
    commands:
      path: "c:\users\administrator\downloads\runner.exe"
      args:
        - "{{ MultiArgLoop.value }}"
```

forEach循環輸入列表示例 3

```
- name: "DownloadAllBinaries"
  action: "S3Download"
  loop:
    name: MultiArgLoop
    forEach:
      - "bin1.exe"
      - "bin10.exe"
      - "bin5.exe"
  inputs:
    - source: "s3://bucket/{{ loop.value }}"
      destination: "c:\temp\{{ loop.value }}"
```

forEach循環與分隔列表

循環迭代包含由分隔符分隔的值的字符串。要迭代字符串的組成部分，請 AWS TOE 使用分隔符將字符串拆分為適合迭代的數組。

forEach循環與分隔列表模式

```
- name: "StepName"
  action: "ActionModule"
  loop:
    name: "string"
    forEach:
      list: "string"
      delimiter: ".,;:\n\t -_"
  inputs:
...
```

forEach循環與分隔列表輸入

欄位	Description (描述)	Type	必要	預設
name	給循環的唯一名稱。與同一階段中的其他循環名稱相比，它應該是唯一的。	字串	否	""
list	由一般分隔符號字元連接的組成字串組成的字串。也接受鏈接的表達式。在鏈接表達式的情況下，請確保這些表達式用雙引號括起來，以便 YAML 編譯器正確解釋。	字串	是	N/A
delimiter	字符用於分隔塊中的字符串。預設值為逗號字元。給定列表中	字串	否	逗號：","

欄位	Description (描述)	Type	必要	預設
	只允許一個分隔符： • 點:"." • 逗號："," • 分號：";" • 冒號：":" • 新行："\n" • 標籤："\t" • 空間:" " • 連字符："-" • 下劃線："_" 無法使用鏈結運算式。			

> **Note**
>
> 的值會`list`被視為不可變的字串。如果的來源在執`list`行階段期間變更，則在執行期間將不會反映出來。

`forEach`循環與分隔列表示例 1

此範例會使用下列鏈結運算式模式來參考另一個步驟的輸出：<phase_name>.<step_name>.[inputs | outputs].<var_name>。

```
- name: "RunMSIs"
  action: "ExecuteBinary"
  loop:
    forEach:
      list: "{{ build.GetAllMSIPathsForInstallation.outputs.stdout }}"
      delimiter: "\n"
  inputs:
    commands:
      path: "{{ loop.value }}"
```

forEach循環與分隔列表示例 2

```
- name: "UploadMetricFiles"
  action: "S3Upload"
  loop:
    forEach:
      list: "/tmp/m1.txt,/tmp/m2.txt,/tmp/m3.txt,..."
  inputs:
    commands:
      - source: "{{ loop.value }}"
        destination: "s3://bucket/key/{{ loop.value }}"
```

步驟欄位

迴圈是步驟的一部分。任何與步驟執行相關的欄位都不會套用至個別版序。步驟欄位僅適用於步驟層級，如下所示：

- TimeOutSecon ds — 迴圈的所有反覆項目都必須在此欄位指定的期間內執行。如果迴圈執行逾時，則 AWS TOE 執行步驟的重試原則，並為每次新嘗試重設逾時參數。如果迴圈執行在達到重試次數上限之後超過逾時值，步驟的失敗訊息會指出迴圈執行已逾時。
- 失敗 — 失敗處理應用於步驟，如下所示：
  - 如果 onFailure 設定為*Abort*，則 AWS TOE 結束迴圈並根據重試原則重試步驟。在重試嘗試次數上限之後，將目前步驟 AWS TOE 標示為失敗，並停止執行程序。

    AWS TOE 將父階段和文件的狀態碼設定為Failed。

> **Note**
>
> 在失敗的步驟之後，不會執行其他步驟。

- 如果 onFailure 設定為*Continue*，則 AWS TOE 結束迴圈並根據重試原則重試步驟。在嘗試重試次數上限之後，將目前步驟 AWS TOE 標示為失敗，然後繼續執行下一個步驟。

  AWS TOE 將父階段和文件的狀態碼設定為`Failed`。

- 如果 onFailure 設定為*Ignore*，則 AWS TOE 結束迴圈並根據重試原則重試步驟。在重試嘗試次數上限之後，將目前步驟 AWS TOE 標示為`IgnoredFailure`，然後繼續執行下一個步驟。

  AWS TOE 將父階段和文件的狀態碼設定為`SuccessWithIgnoredFailure`。

> **Note**
>
> 這仍然被視為成功執行，但包含可讓您知道一個或多個步驟失敗且被忽略的資訊。

- maxIntries — 對於每次重試，整個步驟和所有反覆項目都會從頭開始執行。
- 狀態 — 執行步驟的整體狀態。 `status`不代表個別版序的狀況。具有迴圈的步驟狀態決定如下：
  - 如果單一版序無法執行，則步驟的狀態會指向失敗。
  - 如果所有版序都成功，步驟的狀況會指向成功。
- startTime — 步驟執行的整體開始時間。不代表個別版序的開始時間。
- endTime — 步驟執行的整體結束時間。不代表個別版序的結束時間。
- 失敗訊息 — 包括在發生非逾時錯誤時失敗的迭代索引。在超時錯誤的情況下，消息指出循環運行失敗。不會針對每個迭代提供個別錯誤訊息，以盡量減少失敗訊息的大小。

步驟和迭代輸出

每次迭代都包含一個輸出。在循環運行結束時， AWS TOE 將所有成功的迭代輸出合併到中`detailedOutput.json`。合併的輸出是屬於動作模組輸出結構描述中所定義之對應輸出金鑰的值的定序。下列範例顯示如何合併輸出：

迭代 1 **`ExecuteBash`** 的輸出

```
{
 "stdout":"Hello"
```

```
}
```

迭代 2 **ExecuteBash** 的輸出

```
{
 "stdout":"World"
}
```

步驟**ExecuteBash**的輸出

```
{
 "stdout":"Hello\nWorld"
}
```

例如，ExecuteBashExecutePowerShell、和ExecuteBinary是作STDOUT為動作模組輸出傳回的動作模組。 STDOUT訊息會與新行字元結合在一起，以產生中步驟的整體輸出detailedOutput.json。

AWS TOE 不會合併不成功迭代的輸出。

AWS TOE 元件管理員支援的動作模組

映像建置服務 (例如 EC2 Image Builder) 會使用 AWS TOE 動作模組協助設定用於建置和測試自訂機器映像的 EC2 執行個體。本節說明常用 AWS TOE 動作模組的功能，以及如何設定它們，包括範例。

AWS TOE 元件是使用純文字 YAML 文件建立的。如需文件語法的詳細資訊，請參閱在中使用零組件文件 AWS TOE。

> **Note**
>
> 所有動作模組在執行時都會使用與系統管理員代理程式相同的帳戶 (root在 Linux 和 Windows NT Authority\SYSTEM 上)。

動作模組類型

- 一般執行模組
- 文件下載和上傳模塊
- 檔案系統作業模組

- 軟體安裝動作
- 系統動作模組

一般執行模組

下節包含執行一般執行命令和指示之動作模組的詳細資訊。

一般執行動作模組

- ExecuteBash
- ExecuteBinary
- ExecuteDocument
- ExecutePowerShell

ExecuteBash

ExecuteBash動作模塊允許您使用內聯 shell 代碼/命令運行 bash 腳本。這個模組支援 Linux 系統。

您在命令塊中指定的所有命令和指令都會轉換為一個文件（例如，`input.sh`）並使用 bash shell 運行。運行 shell 文件的結果是步驟的退出代碼。

如果指令碼以結束代碼結束，則ExecuteBash模組會處理系統重新啟動194。啟動時，應用程式會執行下列其中一個動作：

- 如果系統管理員代理程式執行，應用程式會將結束代碼交給呼叫者。Systems Manager 代理程式會處理系統重新開機，並執行與起始重新啟動相同的步驟，如從指令碼重新啟動受控執行個體中所述。
- 該應用程序保存當前`executionstate`，配置重新啟動觸發器以重新運行應用程序，然後重新啟動系統。

系統重新啟動後，應用程式會執行與起始重新啟動相同的步驟。如果您需要此功能，則必須撰寫可處理相同 shell 命令的多次叫用的冪等指令碼。

輸入

原始	Description (描述)	Type	必要
`commands`	包含要按照 bash 語法運行的指令或命令	清單	是

原始	Description (描述)	Type	必要
	列表。允許使用多行 YAML。		

輸入範例：重新開機前後

```
name: ExitCode194Example
description: This shows how the exit code can be used to restart a system with
 ExecuteBash
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: RestartTrigger
        action: ExecuteBash
        inputs:
          commands:
            - |
              REBOOT_INDICATOR=/var/tmp/reboot-indicator
              if [ -f "${REBOOT_INDICATOR}" ]; then
                echo 'The reboot file exists. Deleting it and exiting with success.'
                rm "${REBOOT_INDICATOR}"
                exit 0
              fi
              echo 'The reboot file does not exist. Creating it and triggering a
 restart.'
              touch "${REBOOT_INDICATOR}"
              exit 194
```

輸出

欄位	Description (描述)	Type
stdout	命令執行的標準輸出。	string

如果您啟動重新啟動並返回退出代碼作194為操作模塊的一部分，則構建將在啟動重新啟動的相同操作模塊步驟中恢復。如果您在沒有結束代碼的情況下啟動重新開機，建置程序可能會失敗。

輸出示例：重新啟動之前（第一次通過文檔）

```
{
 “stdout”: “The reboot file does not exist. Creating it and triggering a restart."
}
```

輸出示例：重新啟動後，（第二次通過文檔）

```
{
 “stdout”: “The reboot file exists. Deleting it and exiting with success."
}
```

ExecuteBinary

ExecuteBinary動作模塊允許您運行帶有命令行參數列表的二進製文件。

如果二進位檔案以 194 (Linux) 或 3010 (Windows) 的結束代碼結束，則ExecuteBinary模組會處理系統重新啟動。發生這種情況時，應用程式會執行下列其中一個動作：

- 如果系統管理員代理程式執行，應用程式會將結束代碼交給呼叫者。Systems Manager 代理程式會處理重新啟動系統，並執行與起始重新啟動相同的步驟，如從指令碼重新啟動受控執行個體中所述。
- 該應用程序保存當前executionstate，配置重新啟動觸發器以重新運行應用程序，然後重新啟動系統。

系統重新啟動後，應用程式會執行與起始重新啟動相同的步驟。如果您需要此功能，則必須撰寫可處理相同 shell 命令的多次叫用的冪等指令碼。

輸入

原始	Description (描述)	Type	必要
path	要執行的二進位檔案的路徑。	字串	是
arguments	包含執行二進位檔時要使用的命令列引數清單。	字串清單	否

輸入範例：安裝 .NET

```
  - name: "InstallDotnet"
```

```
    action: ExecuteBinary
    inputs:
      path: C:\PathTo\dotnet_installer.exe
      arguments:
        - /qb
        - /norestart
```

輸出

欄位	Description (描述)	Type
stdout	命令執行的標準輸出。	string

輸出範例

```
{
 "stdout": "success"
}
```

ExecuteDocument

ExecuteDocument動作模組增加了對巢狀元件文件的支援，從一個文件執行多個元件文件。 AWS TOE 驗證在執行階段傳入輸入參數的文件。

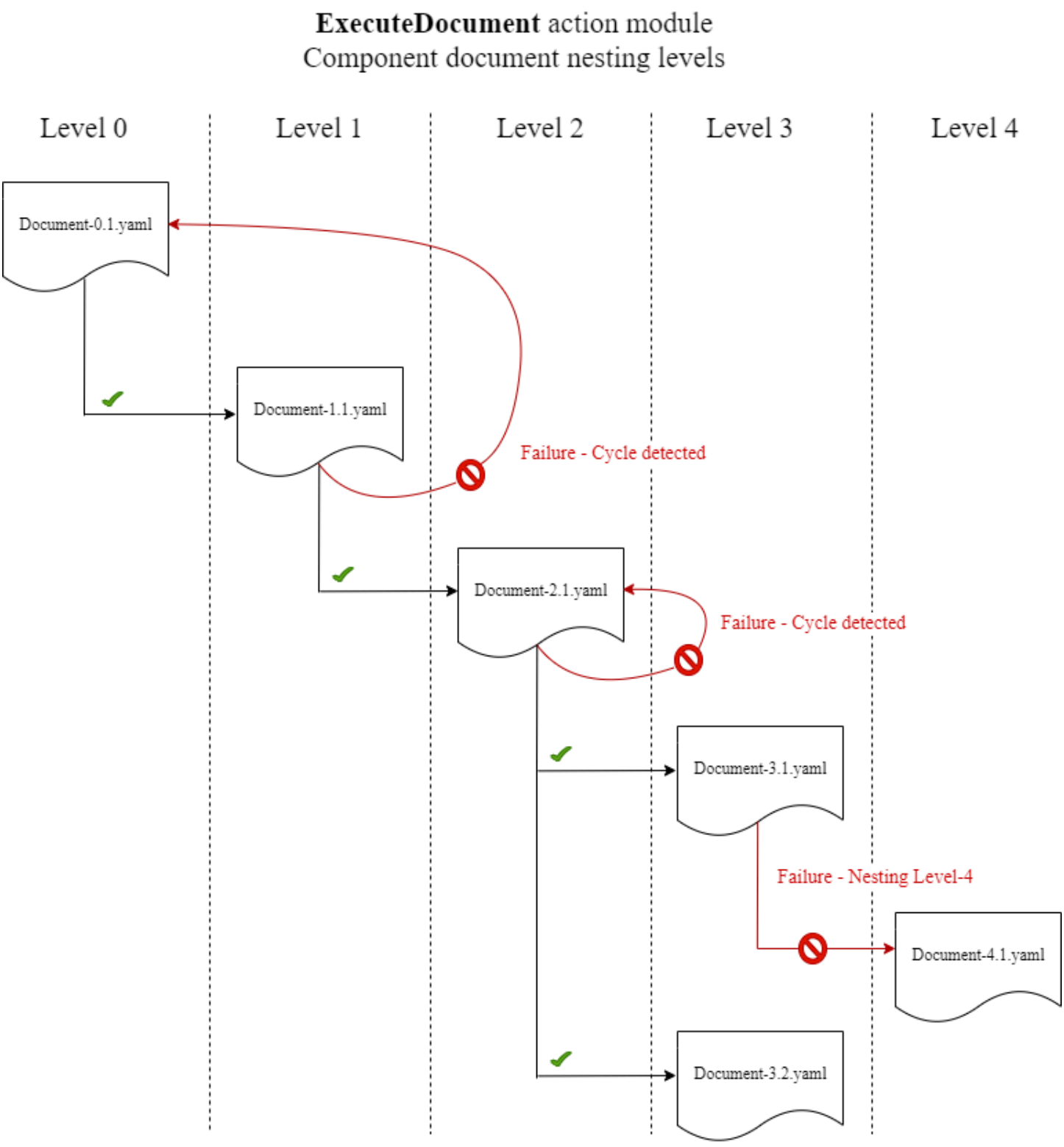
限制

- 此操作模塊運行一次，不允許重試，也沒有設置超時限制的選項。ExecuteDocument設定下列預設值，如果您嘗試變更它們，則會傳回錯誤。
  - `timeoutSeconds`：-1
  - `maxAttempts`：1

  > **Note**
  >
  > 您可以將這些值保留空白，並 AWS TOE 使用預設值。

- 允許文檔嵌套，最多三層深，但不超過此。三個巢狀層級會轉換為四個文件層級，因為頂層不是巢狀的。在這個案例中，最低層級的文件不得呼叫任何其他文件。

- 不允許循環執行元件文件。任何在迴圈結構之外呼叫本身的文件，或是在目前的執行鏈中呼叫另一個較高層級的文件，都會啟動可能導致無限迴圈的循環。當 AWS TOE 檢測到循環執行時，它會停止執行並記錄失敗。



如果元件文件嘗試自行執行，或執行目前執行鏈結中較高的任何元件文件，則執行會失敗。

輸入

原始	Description (描述)	Type	必要
document	元件文件的路徑。有效的選項包含： • 本機檔案路徑 • S3 URI • EC2 Image Builder 元件建置版本 ARN	字串	是
document-s3-bucket-owner	存放元件文件之 S3 儲存貯體的 S3 儲存貯體擁有者帳戶識別碼。(如果您在元件文件中使用 S3 URI，則建議使用此選項)。	字串	否
phases	要在元件文件中執行的階段，以逗號分隔的清單表示。如果未指定階段，則會執行所有階段。	字串	否
parameters	在執行階段作為索引鍵值配對傳入至組件文件的輸入參數。	參數對映清單	否

參數對映輸入

原始	Description (描述)	Type	必要
name		字串	是

原始	Description (描述)	Type	必要
	要傳遞至ExecuteDocument動作模組執行之元件文件的輸入參數名稱。		
value	輸入參數的值。	字串	是

輸入範例

下列範例會根據您的安裝路徑，顯示元件文件的輸入變化。

輸入範例：本機文件路徑

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        inputs:
          document: Sample-1.yaml
          phases: build
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

輸入範例：S3 URI 作為文件路徑

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
```

```
        action: ExecuteDocument
        inputs:
          document: s3://my-bucket/Sample-1.yaml
          document-s3-bucket-owner: 123456789012
          phases: build,validate
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

輸入範例：EC2 Image Builder 元件 ARN 做為文件路徑

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        inputs:
          document: arn:aws:imagebuilder:us-west-2:aws:component/Sample-Test/1.0.0
          phases: test
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

使用 ForEach 循環運行文檔

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        loop:
          name: 'myForEachLoop'
          forEach:
```

```
            - Sample-1.yaml
            - Sample-2.yaml
        inputs:
          document: "{{myForEachLoop.value}}"
          phases: test
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

使用 For 迴圈執行文件

```
# main.yaml
schemaVersion: 1.0

phases:
  - name: build
    steps:
      - name: ExecuteNestedDocument
        action: ExecuteDocument
        loop:
          name: 'myForLoop'
          for:
            start: 1
            end: 2
            updateBy: 1
        inputs:
          document: "Sample-{{myForLoop.value}}.yaml"
          phases: test
          parameters:
            - name: parameter-1
              value: value-1
            - name: parameter-2
              value: value-2
```

輸出

AWS TOE 建立`detailedoutput.json`每次執行時呼叫的輸出檔案。該文件包含有關在運行時調用的每個組件文檔的每個階段和步驟的詳細信息。對於ExecuteDocument動作模組，您可以在`outputs`欄位中找到簡短的執行階段摘要，以及它在中執行的階段、步驟和文件的詳細資訊`detailedOutput`。

```
{
 \"executedStepCount\":1,\"executionId\":\"97054e22-06cc-11ec-9b14-acde48001122\",
\"failedStepCount\":0,\"failureMessage\":\"\",\"ignoredFailedStepCount\":0,\"logUrl\":
\"\",\"status\":\"success\"
}",
```

每個組件文件的輸出摘要物件都包含下列詳細資訊，如下所示，以及範例值：

- executedStepCount「：1
- 「執行識別碼」："12345a67-89BC-01-二 f34-正版
- 「failedStepCount「：0
- 「失敗消息」：「」
- 「ignoredFailedStep計數」：0
- 「日誌網址」：「」
- 「狀態」：「成功」

輸出範例

下列範例會顯示執行巢狀執行時ExecuteDocument動作模組的輸出。在此範例中，`main.yaml`元件文件成功執行`Sample-1.yaml`元件文件。

```
{
    "executionId": "12345a67-89bc-01de-2f34-abcd56789012",
    "status": "success",
    "startTime": "2021-08-26T17:20:31-07:00",
    "endTime": "2021-08-26T17:20:31-07:00",
    "failureMessage": "",
    "documents": [
        {
            "name": "",
            "filePath": "main.yaml",
            "status": "success",
            "description": "",
            "startTime": "2021-08-26T17:20:31-07:00",
            "endTime": "2021-08-26T17:20:31-07:00",
            "failureMessage": "",
            "phases": [
                {
                    "name": "build",
```

```
                    "status": "success",
                    "startTime": "2021-08-26T17:20:31-07:00",
                    "endTime": "2021-08-26T17:20:31-07:00",
                    "failureMessage": "",
                    "steps": [
                        {
                            "name": "ExecuteNestedDocument",
                            "status": "success",
                            "failureMessage": "",
                            "timeoutSeconds": -1,
                            "onFailure": "Abort",
                            "maxAttempts": 1,
                            "action": "ExecuteDocument",
                            "startTime": "2021-08-26T17:20:31-07:00",
                            "endTime": "2021-08-26T17:20:31-07:00",
                            "inputs": "[{\"document\":\"Sample-1.yaml\",\"document-s3-
bucket-owner\":\"\",\"phases\":\"\",\"parameters\":null}]",
                            "outputs": "[{\"executedStepCount\":1,\"executionId\":
\"98765f43-21ed-09cb-8a76-fedc54321098\",\"failedStepCount\":0,\"failureMessage\":\"\",
\"ignoredFailedStepCount\":0,\"logUrl\":\"\",\"status\":\"success\"}]",
                            "loop": null,
                            "detailedOutput": [
                                {
                                    "executionId": "98765f43-21ed-09cb-8a76-
fedc54321098",
                                    "status": "success",
                                    "startTime": "2021-08-26T17:20:31-07:00",
                                    "endTime": "2021-08-26T17:20:31-07:00",
                                    "failureMessage": "",
                                    "documents": [
                                        {
                                            "name": "",
                                            "filePath": "Sample-1.yaml",
                                            "status": "success",
                                            "description": "",
                                            "startTime": "2021-08-26T17:20:31-07:00",
                                            "endTime": "2021-08-26T17:20:31-07:00",
                                            "failureMessage": "",
                                            "phases": [
                                                {
                                                    "name": "build",
                                                    "status": "success",
                                                    "startTime":
 "2021-08-26T17:20:31-07:00",
```

```
                                                    "endTime":
 "2021-08-26T17:20:31-07:00",
                                                    "failureMessage": "",
                                                    "steps": [
                                                        {
                                                            "name": "ExecuteBashStep",
                                                            "status": "success",
                                                            "failureMessage": "",
                                                            "timeoutSeconds": 7200,
                                                            "onFailure": "Abort",
                                                            "maxAttempts": 1,
                                                            "action": "ExecuteBash",
                                                            "startTime":
 "2021-08-26T17:20:31-07:00",
                                                            "endTime":
 "2021-08-26T17:20:31-07:00",
                                                            "inputs": "[{\"commands\":
[\"echo \\\"Hello World!\\\"\"]}]",
                                                            "outputs": "[{\"stdout\":
\"Hello World!\"}]",
                                                            "loop": null,
                                                            "detailedOutput": null
                                                        }]
                                                }]
                                        }]
                                }]
                        }]

                }]
        }]
}
```

ExecutePowerShell

ExecutePowerShell動作模塊允許您使用內聯 shell 代碼/命 PowerShell 令運行腳本。此模組支援視窗平台和視窗 PowerShell。

在命令塊中指定的所有命令/指令都轉換為腳本文件（例如，`input.ps1`）並使用 Windows 運行。PowerShell運行 shell 文件的結果是退出代碼。

如果 shell 命令以的結束代碼結束，則ExecutePowerShell模組會處理系統重新啟動`3010`。啟動時，應用程式會執行下列其中一個動作：

- 如果由系統管理員代理程式執行，請將結束代碼交給呼叫者。Systems Manager 代理程式會處理系統重新開機，並執行與起始重新啟動相同的步驟，如從指令碼重新啟動受控執行個體中所述。
- 儲存目前的項目executionstate、設定重新啟動觸發程式以重新執行應用程式，然後重新啟動系統。

系統重新啟動後，應用程式會執行與起始重新啟動相同的步驟。如果您需要此功能，則必須撰寫可處理相同 shell 命令的多次叫用的冪等指令碼。

輸入

原始	Description (描述)	Type	必要
commands	包含要依照PowerShell 語法執行的指示或命令清單。允許使用多行 YAML。	字串清單	是。必須指定commands或file，不能同時指定兩者。
file	包含 PowerShell 腳本檔案的路徑。PowerShell 將使用-file命令行參數對此文件運行。路徑必須指向.ps1檔案。	字串	是。必須指定commands或file，不能同時指定兩者。

輸入範例：重新開機前後

```
name: ExitCode3010Example
description: This shows how the exit code can be used to restart a system with
 ExecutePowerShell
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: RestartTrigger
        action: ExecutePowerShell
        inputs:
          commands:
            - |
```

```
              $rebootIndicator = Join-Path -Path $env:SystemDrive -ChildPath 'reboot-
indicator'
              if (Test-Path -Path $rebootIndicator) {
                Write-Host 'The reboot file exists. Deleting it and exiting with
 success.'
                Remove-Item -Path $rebootIndicator -Force | Out-Null
                [System.Environment]::Exit(0)
              }
              Write-Host 'The reboot file does not exist. Creating it and triggering a
 restart.'
              New-Item -Path $rebootIndicator -ItemType File | Out-Null
              [System.Environment]::Exit(3010)
```

輸出

欄位	Description (描述)	Type
stdout	命令執行的標準輸出。	string

如果您執行重新開機並傳回結束代碼3010做為動作模組的一部分，建置將會在啟動重新開機的相同動作模組步驟中繼續執行。如果您在沒有結束代碼的情況下執行重新開機，建置程序可能會失敗。

輸出示例：重新啟動之前（第一次通過文檔）

```
{
 “stdout”: “The reboot file does not exist. Creating it and triggering a restart."
}
```

輸出示例：重新啟動後，（第二次通過文檔）

```
{
 “stdout”: “The reboot file exists. Deleting it and exiting with success."
}
```

文件下載和上傳模塊

下節包含執行下載和上載命令和指示的動作模組的詳細資訊。

下載並上傳動作模組

- S3 下載

- 第 3 頁上傳
- WebDownload

S3 下載

使用S3Download動作模組，您可以將 Amazon S3 物件或一組物件下載到您使用destination路徑指定的本機檔案或資料夾。如果指定位置中已有任何檔案，且overwrite旗標設定為 true，則S3Download會覆寫檔案。

您的source位置可以指向 Amazon S3 中的特定物件，或者您可以使用帶有星號萬用字元 (*) 的 key prefix 置詞來下載符合 key prefix 路徑的一組物件。當您在您的source位置指定 key prefix 時，S3Download操作模塊將下載與前綴匹配的所有內容（包括的文件和文件夾）。請確定 key prefix 以正斜線結尾，後面接著星號 (/*)，以便下載符合前置詞的所有項目。例如：*s3://my-bucket/my-folder/**。

> **Note**
>
> 下載前，目標路徑中的所有資料夾都必須存在，否則下載失敗。

如果指定 key prefix 的S3Download動作在下載期間失敗，資料夾內容不會回復到失敗前的狀態。目的地資料夾會維持在失敗時的狀態。

支援的使用案例

動S3Download作模組支援下列使用案例：

- Amazon S3 物件會按照下載路徑中的指定，下載到本機資料夾。
- Amazon S3 物件 (在 Amazon S3 檔案路徑中具有 key prefix) 會下載到指定的本機資料夾，該資料夾會遞歸地將符合 key prefix 的所有 Amazon S3 物件複製到本機資料夾。

IAM 要求

您與執行個體設定檔相關聯的 IAM 角色必須具有執行S3Download動作模組的許可。下列 IAM 政策必須附加至與執行個體設定檔相關聯的 IAM 角色：

- 單一檔案：s3:GetObject針對值區/物件 (例如，)。arn:aws:s3:::**BucketName**/*
- 多個檔案：s3:ListBucket針對值區/物件 (例如，arn:aws:s3:::**BucketName**) 和s3:GetObject儲存貯體/物件 (例如，)。arn:aws:s3:::**BucketName**/*

輸入

原始	Description (描述)	Type	必要	預設
source	作為您下載的來源的 Amazon S3 存儲桶。您可以指定特定物件的路徑，或使用以正斜線結尾的 key prefix 置詞，後面接著星號萬用字元 (/*)，以下載符合金鑰前置詞的一組物件。	字串	是	N/A
destination	下載 Amazon S3 物件的本機路徑。若要下載單一檔案，您必須指定檔案名稱做為路徑的一部分。例如 */myfolder/package.zip* 。	字串	是	N/A
expectedBucketOwner	source路徑中提供之值區的預期擁有者帳號 ID。我們建議您驗證來源中指定的	字串	否	N/A

原始	Description (描述)	Type	必要	預設
	Amazon S3 儲存貯體的擁有權。			
overwrite	設定為 true 時，如果指定本機路徑的目標資料夾中已存在相同名稱的檔案，則下載檔案會覆寫本機檔案。設定為 false 時，本機系統上的現有檔案不會遭到覆寫，且動作模組會因下載錯誤而失敗。 例如：`Error: S3Download: File already exists and "overwrite" property for "destination" file is set to false. Cannot download.`	Boolean	否	true

> **Note**
>
> 在下列範例中，您可以使用 Linux 路徑取代 Windows 資料夾路徑。例如，*C:\myfolder\package.zip*可以替換為*/myfolder/package.zip*。

輸入範例：將 Amazon S3 物件複製到本機檔案

下列範例顯示如何將 Amazon S3 物件複製到本機檔案。

```
- name: DownloadMyFile
  action: S3Download
  inputs:
    - source: s3://mybucket/path/to/package.zip
      destination: C:\myfolder\package.zip
      expectedBucketOwner: 123456789022
      overwrite: false
    - source: s3://mybucket/path/to/package.zip
      destination: C:\myfolder\package.zip
      expectedBucketOwner: 123456789022
      overwrite: true
    - source: s3://mybucket/path/to/package.zip
      destination: C:\myfolder\package.zip
      expectedBucketOwner: 123456789022
```

輸入範例：將具有 key prefix 的 Amazon S3 儲存貯體中的所有 Amazon S3 物件複製到本機資料夾

下列範例顯示如何將 Amazon S3 儲存貯體中具有 key prefix 的所有 Amazon S3 物件複製到本機資料夾。Amazon S3 沒有資料夾的概念，因此會複製符合 key prefix 的所有物件。可下載的物件數目上限為 1000 個。

```
- name: MyS3DownloadKeyprefix
  action: S3Download
  maxAttempts: 3
  inputs:
    - source: s3://mybucket/path/to/*
      destination: C:\myfolder\
      expectedBucketOwner: 123456789022
      overwrite: false
    - source: s3://mybucket/path/to/*
      destination: C:\myfolder\
      expectedBucketOwner: 123456789022
```

```
        overwrite: true
      - source: s3://mybucket/path/to/*
        destination: C:\myfolder\
        expectedBucketOwner: 123456789022
```

輸出

無。

第 3 頁上傳

使用 S3Upload 動作模組，您可以將檔案從來源檔案或資料夾上傳到 Amazon S3 位置。您可以在為來源位置指定的路徑中使用萬用字元 (*)，以上傳路徑符合萬用字元模式的所有檔案。

如果遞迴 S3Upload 動作失敗，則任何已上傳的檔案都會保留在目的地 Amazon S3 儲存貯體中。

支援的使用案例

- 本地文件到 Amazon S3 對象。
- 資料夾中的本機檔案 (使用萬用字元) 至 Amazon S3 key prefix。
- 將本機資料夾 (必須`recurse`設定為`true`) 複製到 Amazon S3 key prefix。

IAM 要求

您與執行個體設定檔相關聯的 IAM 角色必須具有執行`S3Upload`動作模組的許可。下列 IAM 政策必須附加至與執行個體設定檔相關聯的 IAM 角色。該政策必須將`s3:PutObject`許可授予目標 Amazon S3 儲存貯體。例如，`arn:aws:s3:::BucketName/*`)。

輸入

原始	Description (描述)	Type	必要	預設
`source`	來源檔案/資料夾起源的本機路徑。`source`支援星號萬用字元 (*)。	字串	是	N/A

原始	Description (描述)	Type	必要	預設
`destination`	上傳來源檔案/資料夾之目的地 Amazon S3 儲存貯體的路徑。	字串	是	N/A
`recurse`	當設定為時`true`，遞迴執行 S3 上傳。	字串	否	`false`
`expectedBucketOwner`	目標路徑中指定之 Amazon S3 儲存貯體的預期擁有者帳戶 ID。我們建議您驗證目標中指定的 Amazon S3 儲存貯體的擁有權。	字串	否	N/A

輸入範例：將本機檔案複製到 Amazon S3 物件

下列範例顯示如何將本機檔案複製到 Amazon S3 物件。

```
- name: MyS3UploadFile
  action: S3Upload
  onFailure: Abort
  maxAttempts: 3
  inputs:
    - source: C:\myfolder\package.zip
      destination: s3://mybucket/path/to/package.zip
      expectedBucketOwner: 123456789022
```

輸入範例：將本機資料夾中的所有檔案複製到具有 key prefix 的 Amazon S3 儲存貯體

下列範例顯示如何將本機資料夾中的所有檔案複製到具有 key prefix 的 Amazon S3 儲存貯體。此範例不會複製子資料夾或其內容，因recurse為未指定，且預設為false。

```
- name: MyS3UploadMultipleFiles
  action: S3Upload
  onFailure: Abort
  maxAttempts: 3
  inputs:
    - source: C:\myfolder\*
      destination: s3://mybucket/path/to/
      expectedBucketOwner: 123456789022
```

輸入範例：將所有檔案和資料夾從本機資料夾遞迴複製到 Amazon S3 儲存貯體

下列範例顯示如何將所有檔案和資料夾從本機資料夾遞迴複製到具有 key prefix 的 Amazon S3 儲存貯體。

```
- name: MyS3UploadFolder
  action: S3Upload
  onFailure: Abort
  maxAttempts: 3
  inputs:
    - source: C:\myfolder\*
      destination: s3://mybucket/path/to/
      recurse: true
      expectedBucketOwner: 123456789022
```

輸出

無。

WebDownload

該WebDownload操作模塊允許您通過 HTTP/HTTPS 協議從遠程位置下載文件和資源（建議使用 HTTPS）。下載的數量或大小沒有限制。該模塊處理重試和指數輪詢邏輯。

根據使用者輸入，每個下載作業最多會分配 5 次嘗試成功。這些嘗試與文件maxAttempts欄位中指定的嘗試不同steps，這些嘗試與動作模組失敗有關。

此操作模塊隱式處理重定向。除了以外的所有 HTTP 狀態碼都會導致錯誤。200

輸入

原始	Description (描述)	Type	必要	預設
source	有效的 HTTP/HTTPS 網址 (建議使用 HTTPS)，它遵循 RFC 3986 標準。允許鏈結運算式。	字串	是	N/A
destination	本機系統上的絕對或相對檔案或資料夾路徑。資料夾路徑必須以結尾/。如果它們不以結尾/，則會將它們視為檔案路徑。該模塊創建任何成功下載所需的文件或文件夾。允許鏈結運算式。	字串	是	N/A
overwrite	啟用時，會使用下載的檔案或資源覆寫本機系統上的任何現有檔案。如果未啟用，則不會覆寫本機系統上的任何現有檔案，並且動作模組會失敗並顯示錯誤。啟用覆寫並指定總和檢查碼和演	Boolean	否	true

原始	Description (描述)	Type	必要	預設
	算法時，只有在任何預先存在檔案的總和檢查碼和雜湊不符時，動作模組才會下載檔案。			
`checksum`	當您指定總和檢查碼時，會根據提供的演算法產生的下載檔案的雜湊進行檢查。若要啟用檔案驗證，必須同時提供總和檢查碼和演算法。允許鏈結運算式。	字串	否	N/A
`algorithm`	用來計算總和檢查碼的演算法。選項包括 MD5、SHA1、SHA256 和 SHA512。若要啟用檔案驗證，必須同時提供總和檢查碼和演算法。允許鏈結運算式。	字串	否	N/A
`ignoreCertificateErrors`	啟用時會忽略 SSL 憑證驗證。	Boolean	否	`false`

輸出

原始	Description (描述)	Type				
destination	以換行字元分隔的字串，指定儲存下載檔案或資源的目標路徑。	字串				

輸入範例：將遠端檔案下載到本機目的地

```
  - name: DownloadRemoteFile
    action: WebDownload
    maxAttempts: 3
    inputs:
      - source: https://testdomain/path/to/java14.zip
        destination: C:\testfolder\package.zip
```

輸出：

```
{
 "destination": "C:\\testfolder\\package.zip"
}
```

輸入範例：將多個遠端檔案下載至多個本機目的地

```
  - name: DownloadRemoteFiles
    action: WebDownload
    maxAttempts: 3
    inputs:
      - source: https://testdomain/path/to/java14.zip
        destination: /tmp/java14_renamed.zip
      - source: https://testdomain/path/to/java14.zip
        destination: /tmp/create_new_folder_and_add_java14_as_zip/
```

輸出：

```
{
 "destination": "/tmp/create_new_folder/java14_renamed.zip\n/tmp/
create_new_folder_and_add_java14_as_zip/java14.zip"
}
```

輸入示例：下載一個遠程文件而不覆蓋本地目的地，並通過文件驗證下載另一個遠程文件

```
  - name: DownloadRemoteMultipleProperties
    action: WebDownload
    maxAttempts: 3
    inputs:
      - source: https://testdomain/path/to/java14.zip
        destination: C:\create_new_folder\java14_renamed.zip
        overwrite: false
      - source: https://testdomain/path/to/java14.zip
        destination: C:\create_new_folder_and_add_java14_as_zip\
        checksum: ac68bbf921d953d1cfab916cb6120864
        algorithm: MD5
        overwrite: true
```

輸出：

```
{
 "destination": "C:\\create_new_folder\\java14_renamed.zip\nC:\
\create_new_folder_and_add_java14_as_zip\\java14.zip"
}
```

輸入範例：下載遠端檔案並忽略 SSL 認證驗證

```
  - name: DownloadRemoteIgnoreValidation
    action: WebDownload
    maxAttempts: 3
    inputs:
      - source: https://www.bad-ssl.com/resource
        destination: /tmp/downloads/
        ignoreCertificateErrors: true
```

輸出：

```
{
 "destination": "/tmp/downloads/resource"
```

```
}
```

檔案系統作業模組

下節包含執行檔案系統作業指令和指示之動作模組的詳細資訊。

檔案系統作業動作模組

- AppendFile
- CopyFile
- CopyFolder
- CreateFile
- CreateFolder
- CreateSymlink
- DeleteFile
- DeleteFolder
- ListFiles
- MoveFile
- MoveFolder
- ReadFile
- SetFileEncoding
- SetFileOwner
- SetFolderOwner
- SetFilePermissions
- SetFolderPermissions

AppendFile

該AppendFile操作模塊將指定的內容添加到文件的預先存在的內容。

如果檔案編碼值與預設的 encoding (`utf-8`) 值不同，您可以使用`encoding`選項來指定檔案編碼值。默認情況下，`utf-16`並`utf-32`假定使用小端編碼。

當發生以下情況時，動作模塊返回一個錯誤：

- 指定的檔案在執行階段不存在。

- 您沒有修改檔案內容的寫入權限。
- 該模塊在文件操作過程中遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
path	檔案路徑。	字串	是	N/A	N/A	是
content	要附加到文件的內容。	字串	否	空字串	N/A	是
encoding	編碼標準。	字串	否	utf8	utf8、utf-LE、utf-16 LE utf16-BE、utf-16 BE 、utf32、u LEutf-32-LE 、utf32-BE、和 utf-32-BE 。編碼選項的值不區分大小寫。	是 -16、u 32-

輸入範例：不使用編碼追加檔案 (Linux)

```
- name: AppendingFileWithOutEncodingLinux
  action: AppendFile
  inputs:
    - path: ./Sample.txt
      content: "The string to be appended to the file"
```

輸入範例：不使用編碼方式追加檔案 (Windows)

```
- name: AppendingFileWithOutEncodingWindows
  action: AppendFile
  inputs:
    - path: C:\MyFolder\MyFile.txt
      content: "The string to be appended to the file"
```

輸入範例：使用編碼追加檔案 (Linux)

```
- name: AppendingFileWithEncodingLinux
  action: AppendFile
  inputs:
    - path: /FolderName/SampleFile.txt
      content: "The string to be appended to the file"
      encoding: UTF-32
```

輸入範例：使用編碼追加檔案 (Windows)

```
- name: AppendingFileWithEncodingWindows
  action: AppendFile
  inputs:
    - path: C:\MyFolderName\SampleFile.txt
      content: "The string to be appended to the file"
      encoding: UTF-32
```

輸入示例：使用空字符串附加文件（Linux）

```
- name: AppendingEmptyStringLinux
  action: AppendFile
  inputs:
    - path: /FolderName/SampleFile.txt
```

輸入範例：使用空字串附加檔案 (Windows)

```
- name: AppendingEmptyStringWindows
  action: AppendFile
  inputs:
    - path: C:\MyFolderName\SampleFile.txt
```

輸出

無。

CopyFile

該CopyFile操作模塊將文件從指定源複製到指定的目的地。默認情況下，如果模塊在運行時不存在，則遞歸地創建目標文件夾。

如果具有指定名稱的文件已經存在於指定的文件夾中，默認情況下，操作模塊將覆蓋現有文件。您可以將覆寫選項設定為來覆寫此預設行為false。當覆蓋選項設置為false，並且在指定位置已經存在具有指定名稱的文件時，操作模塊將返回一個錯誤。此選項的工作方式與 Linux 中的cp命令相同，默認情況下會覆蓋該命令。

來源檔案名稱可以包含萬用字元 (*)。只有在最後一個檔案路徑分隔符號 (/或\) 之後才接受萬用字元。如果來源檔案名稱中包含萬用字元，則所有符合萬用字元的檔案都會複製到目標資料夾。如果您要使用萬用字元移動多個檔案，則destination選項的輸入必須以檔案路徑分隔符號 (/或\) 結束，表示目的地輸入為資料夾。

如果目標檔案名稱與來源檔案名稱不同，您可以使用destination選項指定目標檔案名稱。如果您未指定目標檔案名稱，則會使用來源檔案的名稱來建立目標檔案。最後一個檔案路徑分隔符號 (/或\) 之後的任何文字都會被視為檔案名稱。如果您要使用與來源檔案相同的檔案名稱，則destination選項的輸入必須以檔案路徑分隔符號 (/或\) 結束。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有在指定資料夾中建立檔案的權限。
- 來源檔案在執行階段不存在。
- 已存在具有指定檔案名稱的資料夾，且overwrite選項設定為false。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
source	來源檔案路徑。	字串	是	N/A	N/A	是
destination	目標檔案路徑。	字串	是	N/A	N/A	是

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
overwrite	設定為 false 時，如果指定位置已存在具有指定名稱的檔案，則不會取代目標檔案。	Boolean	否	true	N/A	是

輸入範例：複製檔案 (Linux)

```
- name: CopyingAFileLinux
  action: CopyFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/destinationFile.txt
```

輸入範例：複製檔案 (視窗)

```
- name: CopyingAFileWindows
  action: CopyFile
  inputs:
    - source: C:\MyFolder\Sample.txt
      destination: C:\MyFolder\destinationFile.txt
```

輸入範例：使用來源檔案名稱複製檔案 (Linux)

```
- name: CopyingFileWithSourceFileNameLinux
  action: CopyFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/
```

輸入範例：使用來源檔案名稱複製檔案 (Windows)

```
- name: CopyingFileWithSourceFileNameWindows
```

```
    action: CopyFile
    inputs:
      - source: C:\Sample\MyFolder\Sample.txt
        destination: C:\MyFolder\
```

輸入範例：使用萬用字元 (Linux) 複製檔案

```
  - name: CopyingFilesWithWildCardLinux
    action: CopyFile
    inputs:
      - source: /Sample/MyFolder/Sample*
        destination: /MyFolder/
```

輸入範例：使用萬用字元複製檔案 (Windows)

```
  - name: CopyingFilesWithWildCardWindows
    action: CopyFile
    inputs:
      - source: C:\Sample\MyFolder\Sample*
        destination: C:\MyFolder\
```

輸入示例：複製文件而不覆蓋（Linux）

```
  - name: CopyingFilesWithoutOverwriteLinux
    action: CopyFile
    inputs:
      - source: /Sample/MyFolder/Sample.txt
        destination: /MyFolder/destinationFile.txt
        overwrite: false
```

輸入範例：複製檔案而不覆寫 (Windows)

```
  - name: CopyingFilesWithoutOverwriteWindows
    action: CopyFile
    inputs:
      - source: C:\Sample\MyFolder\Sample.txt
        destination: C:\MyFolder\destinationFile.txt
        overwrite: false
```

輸出

無。

CopyFolder

該CopyFolder操作模塊將文件夾從指定的源複製到指定的目標。source選項的輸入是要複製的資料夾，而選destination項的輸入是複製來源資料夾內容的資料夾。默認情況下，如果模塊在運行時不存在，則遞歸地創建目標文件夾。

如果指定的資料夾中已存在具有指定名稱的資料夾，動作模組預設會覆寫現有資料夾。您可以將覆寫選項設定為來覆寫此預設行為false。當覆蓋選項設置為false，並且在指定位置中已經存在具有指定名稱的文件夾時，操作模塊將返回錯誤。

來源資料夾名稱可以包含萬用字元 (*)。只有在最後一個檔案路徑分隔符號 (/或\) 之後才接受萬用字元。如果來源資料夾名稱中包含萬用字元，則符合萬用字元的所有資料夾都會複製到目標資料夾。如果要使用萬用字元複製多個資料夾，則destination選項的輸入必須以檔案路徑分隔符號 (/或\) 結束，表示目的地輸入為資料夾。

如果目標資料夾名稱與來源資料夾名稱不同，您可以使用destination選項指定目標資料夾名稱。如果您未指定目標資料夾名稱，則會使用來源資料夾的名稱來建立目標資料夾。最後一個檔案路徑分隔符號 (/或\) 之後的任何文字都會被視為資料夾名稱。如果您想要使用與來源資料夾相同的資料夾名稱，則destination選項的輸入必須以檔案路徑分隔符號 (/或\) 結束。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有在指定資料夾中建立資料夾的權限。
- 來源資料夾在執行階段不存在。
- 已有具有指定資料夾名稱的資料夾，且overwrite選項設定為false。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
source	來源資料夾路徑。	字串	是	N/A	N/A	是
destination	目標資料夾路徑。	字串	是	N/A	N/A	是

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
overwrite	設定為 false 時，如果指定位置中已存在具有指定名稱的資料夾，則不會取代目標資料夾。	Boolean	否	true	N/A	是

輸入範例：複製資料夾 (Linux)

```
- name: CopyingAFolderLinux
  action: CopyFolder
  inputs:
    - source: /Sample/MyFolder/SampleFolder
      destination: /MyFolder/destinationFolder
```

輸入範例：複製資料夾 (視窗)

```
- name: CopyingAFolderWindows
  action: CopyFolder
  inputs:
    - source: C:\Sample\MyFolder\SampleFolder
      destination: C:\MyFolder\destinationFolder
```

輸入範例：使用來源資料夾名稱複製資料夾 (Linux)

```
- name: CopyingFolderSourceFolderNameLinux
  action: CopyFolder
  inputs:
    - source: /Sample/MyFolder/SourceFolder
      destination: /MyFolder/
```

輸入範例：使用來源資料夾名稱複製資料夾 (Windows)

```
- name: CopyingFolderSourceFolderNameWindows
  action: CopyFolder
  inputs:
    - source: C:\Sample\MyFolder\SampleFolder
      destination: C:\MyFolder\
```

輸入範例：使用萬用字元 (Linux) 複製資料夾

```
- name: CopyingFoldersWithWildCardLinux
  action: CopyFolder
  inputs:
    - source: /Sample/MyFolder/Sample*
      destination: /MyFolder/
```

輸入範例：使用萬用字元複製資料夾 (Windows)

```
- name: CopyingFoldersWithWildCardWindows
  action: CopyFolder
  inputs:
    - source: C:\Sample\MyFolder\Sample*
      destination: C:\MyFolder\
```

輸入示例：複製文件夾而不覆蓋（Linux）

```
- name: CopyingFoldersWithoutOverwriteLinux
  action: CopyFolder
  inputs:
    - source: /Sample/MyFolder/SourceFolder
      destination: /MyFolder/destinationFolder
      overwrite: false
```

輸入示例：複製文件夾而不覆蓋（Windows）

```
- name: CopyingFoldersWithoutOverwrite
  action: CopyFolder
  inputs:
    - source: C:\Sample\MyFolder\SourceFolder
      destination: C:\MyFolder\destinationFolder
      overwrite: false
```

輸出

無。

CreateFile

動CreateFile作模塊在指定位置創建一個文件。默認情況下，如果需要，該模塊還遞歸創建父文件夾。

如果文件已經存在於指定的文件夾中，默認情況下，操作模塊會截斷或覆蓋現有文件。您可以將覆寫選項設定為來覆寫此預設行為false。當覆蓋選項設置為false，並且在指定位置已經存在具有指定名稱的文件時，操作模塊將返回一個錯誤。

如果檔案編碼值與預設的 encoding (utf-8) 值不同，您可以使用encoding選項來指定檔案編碼值。默認情況下，utf-16並utf-32假定使用小端編碼。

ownergroup、和permissions是選擇性輸入。的輸入permissions必須是字串值。如果未提供檔案，則會使用預設值建立檔案。這些選項在 Windows 平台上不受支援。如果、和permissions選項在 Windows 平台上使用 ownergroup，則此動作模組會驗證並傳回錯誤。

該操作模塊可以創建一個文件與操作系統的默認umask值定義的權限。如果要覆寫預設umask值，則必須設定值。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有在指定父資料夾中建立檔案或資料夾的權限。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
path	檔案路徑。	字串	是	N/A	N/A	是
content	檔案的文字內容。	字串	否	N/A	N/A	是
encoding	編碼標準。	字串	否	utf8	utf8、utf-LE、utf-16 LE utf16-	是 -16、u

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
					BE、utf-16BE 、utf32、utf-32-LEutf-32-LE 、utf32-BE、和 utf-32-BE 。編碼選項的值不區分大小寫。	32-
owner	使用者名稱或 ID。	字串	否	N/A	N/A	在視窗上不支援。
group	群組名稱或識別碼。	字串	否	目前的使用者。	N/A	在視窗上不支援。
permissions	檔案權限。	字串	否	0666	N/A	在視窗上不支援。
overwrite	如果指定檔案的名稱已存在，請設定此值以false防止依預設截斷或覆寫檔案。	Boolean	否	true	N/A	是

輸入示例：創建一個不覆蓋的文件（Linux）

```
- name: CreatingFileWithoutOverwriteLinux
  action: CreateFile
```

```
    inputs:
      - path: /home/UserName/Sample.txt
        content: The text content of the sample file.
        overwrite: false
```

輸入示例：創建一個文件而不覆蓋（Windows）

```
  - name: CreatingFileWithoutOverwriteWindows
    action: CreateFile
    inputs:
      - path: C:\Temp\Sample.txt
        content: The text content of the sample file.
        overwrite: false
```

輸入示例：創建具有文件屬性的文件

```
  - name: CreatingFileWithFileProperties
    action: CreateFile
    inputs:
      - path: SampleFolder/Sample.txt
        content: The text content of the sample file.
        encoding: UTF-16
        owner: Ubuntu
        group: UbuntuGroup
        permissions: 0777
       - path: SampleFolder/SampleFile.txt
        permissions: 755
      - path: SampleFolder/TextFile.txt
        encoding: UTF-16
        owner: root
        group: rootUserGroup
```

輸入示例：創建一個沒有文件屬性的文件

```
  - name: CreatingFileWithoutFileProperties
    action: CreateFile
    inputs:
      - path: ./Sample.txt
      - path: Sample1.txt
```

輸入示例：創建一個空文件以跳過 Linux 清理腳本中的某個部分

```
- name: CreateSkipCleanupfile
  action: CreateFile
  inputs:
    - path: <skip section file name>
```

如需更多資訊，請參閱覆寫 Linux 清理指令碼

輸出

無。

CreateFolder

CreateFolder動作模塊在指定位置創建一個文件夾。默認情況下，如果需要，該模塊還遞歸創建父文件夾。

如果該文件夾已經存在於指定的文件夾中，默認情況下，操作模塊會截斷或覆蓋現有文件夾。您可以將覆寫選項設定為來覆寫此預設行為false。當覆蓋選項設置為false，並且在指定位置中已經存在具有指定名稱的文件夾時，操作模塊將返回錯誤。

ownergroup、和permissions是選擇性輸入。的輸入permissions必須是字串值。這些選項在 Windows 平台上不受支援。如果、和permissions選項在 Windows 平台上使用 ownergroup，則此動作模組會驗證並傳回錯誤。

此操作模塊可以創建一個文件夾與操作系統的默認umask值定義的權限。如果要覆寫預設umask值，則必須設定值。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有在指定位置建立資料夾的權限。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
path	資料夾路徑。	字串	是	N/A	N/A	是

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
owner	使用者名稱或 ID。	字串	否	目前的使用者。	N/A	在視窗上不支援。
group	群組名稱或識別碼。	字串	否	目前使用者的群組。	N/A	在視窗上不支援。
permissions	資料夾權限。	字串	否	0777	N/A	在視窗上不支援。
overwrite	如果指定檔案的名稱已存在，請設定此值以false防止依預設截斷或覆寫檔案。	Boolean	否	true	N/A	是

輸入範例：建立資料夾 (Linux)

```
- name: CreatingFolderLinux
  action: CreateFolder
  inputs:
    - path: /Sample/MyFolder/
```

輸入範例：建立資料夾 (視窗)

```
- name: CreatingFolderWindows
  action: CreateFolder
  inputs:
    - path: C:\MyFolder
```

輸入示例：創建一個文件夾指定文件夾屬性

```
- name: CreatingFolderWithFolderProperties
```

```
    action: CreateFolder
    inputs:
      - path: /Sample/MyFolder/Sample/
        owner: SampleOwnerName
        group: SampleGroupName
        permissions: 0777
      - path: /Sample/MyFolder/SampleFoler/
        permissions: 777
```

輸入範例：建立覆寫現有資料夾的資料夾 (如果有的話)。

```
  - name: CreatingFolderWithOverwrite
    action: CreateFolder
    inputs:
      - path: /Sample/MyFolder/Sample/
        overwrite: true
```

輸出

無。

CreateSymlink

CreateSymlink動作模塊創建符號鏈接，或包含對另一個文件的引用的文件。此模組在視窗平台上不受支援。

path和target選項的輸入可以是絕對路徑或相對路徑。如果選path項的輸入是相對路徑，則在建立連結時會以絕對路徑取代該選項。

默認情況下，當指定的文件夾中已經存在具有指定名稱的鏈接時，操作模塊返回一個錯誤。您可以將force選項設定為來覆寫此預設行為true。當選force項設定為時true，模組將覆寫現有的連結。

如果父文件夾不存在，默認情況下，操作模塊遞歸創建該文件夾。

當發生以下情況時，動作模塊返回一個錯誤：

- 目標檔案在執行階段不存在。
- 具有指定名稱的非符號連結檔案已存在。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
`path`	檔案路徑。	字串	是	N/A	N/A	在視窗上不支援。
`target`	符號連結所指向的目標檔案路徑。	字串	是	N/A	N/A	在視窗上不支援。
`force`	當具有相同名稱的連結已存在時強制建立連結。	Boolean	否	`false`	N/A	在視窗上不支援。

輸入示例：創建強制創建鏈接的符號鏈接

```
- name: CreatingSymbolicLinkWithForce
  action: CreateSymlink
  inputs:
    - path: /Folder2/Symboliclink.txt
      target: /Folder/Sample.txt
      force: true
```

輸入範例：建立不強制建立連結的符號連結

```
- name: CreatingSymbolicLinkWithOutForce
  action: CreateSymlink
  inputs:
    - path: Symboliclink.txt
      target: /Folder/Sample.txt
```

輸出

無。

DeleteFile

DeleteFile動作模塊刪除指定位置的一個或多個文件。

的輸入path應為檔案名稱中含有萬用字元 (*) 的有效檔案路徑或檔案路徑。在檔案名稱中指定萬用字元時，同一資料夾中符合萬用字元的所有檔案都將被刪除。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有執行刪除作業的權限。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
path	檔案路徑。	字串	是	N/A	N/A	是

輸入範例：刪除單一檔案 (Linux)

```
- name: DeletingSingleFileLinux
  action: DeleteFile
  inputs:
    - path: /SampleFolder/MyFolder/Sample.txt
```

輸入範例：刪除單一檔案 (視窗)

```
- name: DeletingSingleFileWindows
  action: DeleteFile
  inputs:
    - path: C:\SampleFolder\MyFolder\Sample.txt
```

輸入示例：刪除以「日誌」（Linux）結尾的文件

```
- name: DeletingFileEndingWithLogLinux
  action: DeleteFile
  inputs:
```

```
      - path: /SampleFolder/MyFolder/*log
```

輸入示例：刪除以「日誌」結尾的文件（Windows）

```
- name: DeletingFileEndingWithLogWindows
  action: DeleteFile
  inputs:
    - path: C:\SampleFolder\MyFolder\*log
```

輸入範例：刪除指定資料夾中的所有檔案 (Linux)

```
- name: DeletingAllFilesInAFolderLinux
  action: DeleteFile
  inputs:
    - path: /SampleFolder/MyFolder/*
```

輸入範例：刪除指定資料夾中的所有檔案 (Windows)

```
- name: DeletingAllFilesInAFolderWindows
  action: DeleteFile
  inputs:
    - path: C:\SampleFolder\MyFolder\*
```

輸出

無。

DeleteFolder

DeleteFolder動作模組會刪除資料夾。

如果資料夾不是空的，您必須將選`force`項設定`true`為移除資料夾及其內容。如果您未將`force`選項設定為`true`，且您嘗試刪除的資料夾不是空的，動作模組會傳回錯誤。`force`選項的預設值為`false`。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有執行刪除作業的權限。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
`path`	資料夾路徑。	字串	是	N/A	N/A	是
`force`	移除資料夾是否為空的資料夾。	Boolean	否	`false`	N/A	是

輸入示例：使用**`force`**選項（Linux）刪除不為空的文件夾

```
- name: DeletingFolderWithForceOptionLinux
  action: DeleteFolder
  inputs:
    - path: /Sample/MyFolder/Sample/
      force: true
```

輸入範例：使用**`force`**選項刪除不為空的資料夾 (Windows)

```
- name: DeletingFolderWithForceOptionWindows
  action: DeleteFolder
  inputs:
    - path: C:\Sample\MyFolder\Sample\
      force: true
```

輸入範例：刪除資料夾 (Linux)

```
- name: DeletingFolderWithOutForceLinux
  action: DeleteFolder
  inputs:
    - path: /Sample/MyFolder/Sample/
```

輸入範例：刪除資料夾 (視窗)

```
- name: DeletingFolderWithOutForce
  action: DeleteFolder
```

```
    inputs:
      - path: C:\Sample\MyFolder\Sample\
```

輸出

無。

ListFiles

ListFiles動作模組會列出指定資料夾中的檔案。當遞迴選項設定為時`true`，它會列出子資料夾中的檔案。默認情況下，此模塊不會列出子文件夾中的文件。

若要列出名稱與指定模式相符的所有檔案，請使用`fileNamePattern`選項來提供模式。`fileNamePattern`此選項接受萬用字元 (*) 值。提供時，會傳回符合指定檔案名稱格式的所有檔案。`fileNamePattern`

當發生以下情況時，動作模塊返回一個錯誤：

- 指定的資料夾在執行階段不存在。
- 您沒有在指定父資料夾中建立檔案或資料夾的權限。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
`path`	資料夾路徑。	字串	是	N/A	N/A	是
`fileNamePattern`	要匹配的模式列出所有名稱與模式匹配的文件。	字串	否	N/A	N/A	是
`recursive`	遞迴列出資料夾中的檔案。	Boolean	否	`false`	N/A	是

輸入範例：列出指定資料夾中的檔案 (Linux)

```
- name: ListingFilesInSampleFolderLinux
  action: ListFiles
  inputs:
    - path: /Sample/MyFolder/Sample
```

輸入範例：列出指定資料夾中的檔案 (Windows)

```
- name: ListingFilesInSampleFolderWindows
  action: ListFiles
  inputs:
    - path: C:\Sample\MyFolder\Sample
```

輸入示例：列出以「日誌」結尾的文件（Linux）

```
- name: ListingFilesWithEndingWithLogLinux
  action: ListFiles
  inputs:
    - path: /Sample/MyFolder/
      fileNamePattern: *log
```

輸入示例：列出以「日誌」結尾的文件（Windows）

```
- name: ListingFilesWithEndingWithLogWindows
  action: ListFiles
  inputs:
    - path: C:\Sample\MyFolder\
      fileNamePattern: *log
```

輸入示例：遞歸列出文件

```
- name: ListingFilesRecursively
  action: ListFiles
  inputs:
    - path: /Sample/MyFolder/
      recursive: true
```

輸出

原始	Description (描述)	Type
files	文件列表。	字串

輸出範例

```
{
 "files": "/sample1.txt,/sample2.txt,/sample3.txt"
}
```

MoveFile

該MoveFile操作模塊將文件從指定的源移動到指定的目的地。

如果文件已經存在於指定的文件夾中，默認情況下，操作模塊將覆蓋現有文件。您可以將覆寫選項設定為來覆寫此預設行為false。當覆蓋選項設置為false，並且在指定位置已經存在具有指定名稱的文件時，操作模塊將返回一個錯誤。此選項的工作方式與 Linux 中的mv命令相同，默認情況下會覆蓋該命令。

來源檔案名稱可以包含萬用字元 (*)。只有在最後一個檔案路徑分隔符號 (/或\) 之後才接受萬用字元。如果來源檔案名稱中包含萬用字元，則所有符合萬用字元的檔案都會複製到目標資料夾。如果您要使用萬用字元移動多個檔案，則destination選項的輸入必須以檔案路徑分隔符號 (/或\) 結束，表示目的地輸入為資料夾。

如果目標檔案名稱與來源檔案名稱不同，您可以使用destination選項指定目標檔案名稱。如果您未指定目標檔案名稱，則會使用來源檔案的名稱來建立目標檔案。最後一個檔案路徑分隔符號 (/或\) 之後的任何文字都會被視為檔案名稱。如果您要使用與來源檔案相同的檔案名稱，則destination選項的輸入必須以檔案路徑分隔符號 (/或\) 結束。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有在指定資料夾中建立檔案的權限。
- 來源檔案在執行階段不存在。
- 已存在具有指定檔案名稱的資料夾，且overwrite選項設定為false。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
source	來源檔案路徑。	字串	是	N/A	N/A	是
destination	目標檔案路徑。	字串	是	N/A	N/A	是
overwrite	設定為 false 時，如果指定位置已存在具有指定名稱的檔案，則不會取代目標檔案。	Boolean	否	true	N/A	是

輸入範例：移動檔案 (Linux)

```
- name: MovingAFileLinux
  action: MoveFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/destinationFile.txt
```

輸入範例：移動檔案 (視窗)

```
- name: MovingAFileWindows
  action: MoveFile
  inputs:
    - source: C:\Sample\MyFolder\Sample.txt
      destination: C:\MyFolder\destinationFile.txt
```

輸入範例：使用來源檔案名稱 (Linux) 移動檔案

```
- name: MovingFileWithSourceFileNameLinux
```

```
  action: MoveFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/
```

輸入範例：使用來源檔案名稱移動檔案 (Windows)

```
- name: MovingFileWithSourceFileNameWindows
  action: MoveFile
  inputs:
    - source: C:\Sample\MyFolder\Sample.txt
      destination: C:\MyFolder
```

輸入範例：使用萬用字元 (Linux) 移動檔案

```
- name: MovingFilesWithWildCardLinux
  action: MoveFile
  inputs:
    - source: /Sample/MyFolder/Sample*
      destination: /MyFolder/
```

輸入範例：使用萬用字元移動檔案 (Windows)

```
- name: MovingFilesWithWildCardWindows
  action: MoveFile
  inputs:
    - source: C:\Sample\MyFolder\Sample*
      destination: C:\MyFolder
```

輸入範例：移動檔案而不覆寫 (Linux)

```
- name: MovingFilesWithoutOverwriteLinux
  action: MoveFile
  inputs:
    - source: /Sample/MyFolder/Sample.txt
      destination: /MyFolder/destinationFile.txt
      overwrite: false
```

輸入範例：移動檔案而不覆寫 (Windows)

```
- name: MovingFilesWithoutOverwrite
```

```
    action: MoveFile
    inputs:
      - source: C:\Sample\MyFolder\Sample.txt
        destination: C:\MyFolder\destinationFile.txt
        overwrite: false
```

輸出

無。

MoveFolder

MoveFolder動作模塊將文件夾從指定的源移動到指定的目的地。source選項的輸入是要移動的資料夾，而選destination項的輸入是移動來源資料夾內容的資料夾。

如果目標父資料夾或destination選項的輸入在執行階段不存在，則模組的預設行為是在指定的目的地遞迴建立資料夾。

如果目標資料夾中已存在與來源資料夾相同的資料夾，動作模組預設會覆寫現有資料夾。您可以將覆寫選項設定為來覆寫此預設行為false。當覆蓋選項設置為false，並且在指定位置中已經存在具有指定名稱的文件夾時，操作模塊將返回錯誤。

來源資料夾名稱可以包含萬用字元 (*)。只有在最後一個檔案路徑分隔符號 (/或\) 之後才接受萬用字元。如果來源資料夾名稱中包含萬用字元，則符合萬用字元的所有資料夾都會複製到目標資料夾。如果要使用萬用字元移動多個資料夾，則destination選項的輸入必須以檔案路徑分隔符號 (/或\) 結束，表示目的地輸入為資料夾。

如果目標資料夾名稱與來源資料夾名稱不同，您可以使用destination選項指定目標資料夾名稱。如果您未指定目標資料夾名稱，則會使用來源資料夾的名稱來建立目標資料夾。最後一個檔案路徑分隔符號 (/或\) 之後的任何文字都會被視為資料夾名稱。如果您想要使用與來源資料夾相同的資料夾名稱，則destination選項的輸入必須以檔案路徑分隔符號 (/或\) 結束。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有在目標資料夾中建立資料夾的權限。
- 來源資料夾在執行階段不存在。
- 已存在具有指定名稱的資料夾，且overwrite選項設定為false。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
source	來源資料夾路徑。	字串	是	N/A	N/A	是
destination	目標資料夾路徑。	字串	是	N/A	N/A	是
overwrite	設定為 false 時，如果指定位置中已存在具有指定名稱的資料夾，則不會取代目標資料夾。	Boolean	否	true	N/A	是

輸入範例：移動資料夾 (Linux)

```
- name: MovingAFolderLinux
  action: MoveFolder
  inputs:
    - source: /Sample/MyFolder/SourceFolder
      destination: /MyFolder/destinationFolder
```

輸入範例：移動資料夾 (視窗)

```
- name: MovingAFolderWindows
  action: MoveFolder
  inputs:
    - source: C:\Sample\MyFolder\SourceFolder
      destination: C:\MyFolder\destinationFolder
```

輸入範例：使用來源資料夾名稱移動資料夾 (Linux)

```
- name: MovingFolderWithSourceFolderNameLinux
  action: MoveFolder
  inputs:
    - source: /Sample/MyFolder/SampleFolder
      destination: /MyFolder/
```

輸入範例：使用來源資料夾名稱移動資料夾 (Windows)

```
- name: MovingFolderWithSourceFolderNameWindows
  action: MoveFolder
  inputs:
    - source: C:\Sample\MyFolder\SampleFolder
      destination: C:\MyFolder\
```

輸入範例：使用萬用字元移動資料夾 (Linux)

```
- name: MovingFoldersWithWildCardLinux
  action: MoveFolder
  inputs:
    - source: /Sample/MyFolder/Sample*
      destination: /MyFolder/
```

輸入範例：使用萬用字元移動資料夾 (Windows)

```
- name: MovingFoldersWithWildCardWindows
  action: MoveFolder
  inputs:
    - source: C:\Sample\MyFolder\Sample*
      destination: C:\MyFolder\
```

輸入範例：移動資料夾而不覆寫 (Linux)

```
- name: MovingFoldersWithoutOverwriteLinux
  action: MoveFolder
  inputs:
    - source: /Sample/MyFolder/SampleFolder
  destination: /MyFolder/destinationFolder
  overwrite: false
```

輸入範例：移動資料夾而不覆寫 (Windows)

```
- name: MovingFoldersWithoutOverwriteWindows
  action: MoveFolder
  inputs:
    - source: C:\Sample\MyFolder\SampleFolder
      destination: C:\MyFolder\destinationFolder
      overwrite: false
```

輸出

無。

ReadFile

ReadFile動作模塊讀取字符串類型的文本文件的內容。該模塊可用於讀取文件的內容，以便通過鏈接或用於將數據讀取到console.log文件的後續步驟中使用。如果指定的路徑是一個符號鏈接，該模塊返回目標文件的內容。此模塊僅支持文本文件。

如果檔案編碼值與預設的 encoding (utf-8) 值不同，您可以使用encoding選項來指定檔案編碼值。默認情況下，utf-16並utf-32假定使用小端編碼。

默認情況下，該模塊無法將文件內容打印到console.log文件中。您可以將性質設定為來取代此printFileContent設定true。

該模塊只能返回一個文件的內容。它無法剖析檔案，例如 Excel 或 JSON 檔案。

當發生以下情況時，動作模塊返回一個錯誤：

- 該文件在運行時不存在。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
path	檔案路徑。	字串	是	N/A	N/A	是
encoding	編碼標準。	字串	否	utf8	utf8、utf-LE、utf-16 LE utf16-	是 -16、u

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
					BE、utf-16BE 、utf32、uLEutf-32-LE 、utf32-BE、和 utf-32-BE 。編碼選項的值不區分大小寫。	32-
printFileContent	將檔案內容列印至console.og 檔案。	Boolean	否	false	N/A	是。

輸入範例：讀取檔案 (Linux)

```
- name: ReadingFileLinux
  action: ReadFile
  inputs:
    - path: /home/UserName/SampleFile.txt
```

輸入範例：讀取檔案 (視窗)

```
- name: ReadingFileWindows
  action: ReadFile
  inputs:
    - path: C:\Windows\WindowsUpdate.log
```

輸入範例：讀取檔案並指定編碼標準

```
- name: ReadingFileWithFileEncoding
```

```
    action: ReadFile
    inputs:
      - path: /FolderName/SampleFile.txt
        encoding: UTF-32
```

輸入範例：讀取檔案並列印至**console.log**檔案

```
  - name: ReadingFileToConsole
    action: ReadFile
    inputs:
      - path: /home/UserName/SampleFile.txt
        printFileContent: true
```

輸出

欄位	Description (描述)	Type
content	檔案內容。	string

輸出範例

```
{
 "content" : "The file content"
}
```

SetFileEncoding

動SetFileEncoding作模塊修改現有文件的編碼屬性。該模塊可以將文件編碼從轉換utf-8為指定的編碼標準。默認情況下，utf-16並utf-32假定為小端編碼。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有執行指定修改的權限。
- 該文件在運行時不存在。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
path	檔案路徑。	字串	是	N/A	N/A	是
encoding	編碼標準。	字串	否	utf8	utf8、utf-LE、utf-16LE utf16-BE、utf-16BE 、utf32、uLEutf-32-LE 、utf32-BE、和 utf-32-BE 。編碼選項的值不區分大小寫。	是 -16、u 32-

輸入範例：設定檔案編碼屬性

```
- name: SettingFileEncodingProperty
  action: SetFileEncoding
  inputs:
    - path: /home/UserName/SampleFile.txt
      encoding: UTF-16
```

輸出

無。

SetFileOwner

SetFileOwner動作模塊修改現有文件的owner和group所有者屬性。如果指定的檔案是符號連結，則模組會修改來源檔案的owner屬性。此模組在視窗平台上不受支援。

該模塊接受用戶和組名作為輸入。如果未提供群組名稱，則模組會將檔案的群組擁有者指派給使用者所屬的群組。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有執行指定修改的權限。
- 指定的使用者或群組名稱在執行階段不存在。
- 該文件在運行時不存在。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
path	檔案路徑。	字串	是	N/A	N/A	在視窗上不支援。
owner	使用者名稱。	string	是	N/A	N/A	在視窗上不支援。
group	使用者群組的名稱。	字串	否	使用者所屬群組的名稱。	N/A	在視窗上不支援。

輸入示例：設置文件所有者屬性而不指定用戶組的名稱

```
- name: SettingFileOwnerPropertyNoGroup
  action: SetFileOwner
  inputs:
    - path: /home/UserName/SampleText.txt
      owner: LinuxUser
```

輸入示例：通過指定所有者和用戶組來設置文件所有者屬性

```
- name: SettingFileOwnerProperty
  action: SetFileOwner
  inputs:
    - path: /home/UserName/SampleText.txt
      owner: LinuxUser
      group: LinuxUserGroup
```

輸出

無。

SetFolderOwner

SetFolderOwner動作模塊遞歸修改現有文件夾的owner和group所有者屬性。依預設，模組可以修改資料夾中所有內容的所有權。您可以設定recursive選項false來覆寫此行為。此模組在視窗平台上不受支援。

該模塊接受用戶和組名作為輸入。如果未提供群組名稱，則模組會將檔案的群組擁有者指派給使用者所屬的群組。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有執行指定修改的權限。
- 指定的使用者或群組名稱在執行階段不存在。
- 資料夾在執行階段不存在。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
path	資料夾路徑。	字串	是	N/A	N/A	在視窗上不支援。
owner	使用者名稱。	string	是	N/A	N/A	在視窗上不支援。

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
group	使用者群組的名稱。	字串	否	使用者所屬群組的名稱。	N/A	在視窗上不支援。
recursive	當設定為時，會取代修改資料夾所有內容之所有權的預設行為 false。	Boolean	否	true	N/A	在視窗上不支援。

輸入示例：設置文件夾所有者屬性而不指定用戶組的名稱

```
- name: SettingFolderPropertyWithOutGroup
  action: SetFolderOwner
  inputs:
    - path: /SampleFolder/
      owner: LinuxUser
```

輸入示例：設置文件夾 owner 屬性而不覆蓋文件夾中所有內容的所有權

```
- name: SettingFolderPropertyWithOutRecursively
  action: SetFolderOwner
  inputs:
    - path: /SampleFolder/
      owner: LinuxUser
      recursive: false
```

輸入示例：通過指定用戶組的名稱來設置文件所有權屬性

```
- name: SettingFolderPropertyWithGroup
  action: SetFolderOwner
  inputs:
    - path: /SampleFolder/
      owner: LinuxUser
```

```
        group: LinuxUserGroup
```

輸出

無。

SetFilePermissions

操SetFilePermissions作模塊修改現有文件`permissions`的。此模組在視窗平台上不受支援。

的輸入`permissions`必須是字串值。

該操作模塊可以創建一個文件與操作系統的默認 umask 值定義的權限。如果要覆寫預設`umask`值，則必須設定值。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有執行指定修改的權限。
- 該文件在運行時不存在。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
`path`	檔案路徑。	字串	是	N/A	N/A	在視窗上不支援。
`permissions`	檔案權限。	字串	是	N/A	N/A	在視窗上不支援。

輸入範例：修改檔案權限

```
- name: ModifyingFilePermissions
  action: SetFilePermissions
  inputs:
    - path: /home/UserName/SampleFile.txt
      permissions: 766
```

輸出

無。

SetFolderPermissions

SetFolderPermissions動作模塊遞歸修改現有文件夾及其所有子文件和子文件夾的。permissions默認情況下，該模塊可以修改指定文件夾的所有內容的權限。您可以設定recursive選項false來覆寫此行為。此模組在視窗平台上不受支援。

的輸入permissions必須是字串值。

該操作模塊可以根據操作系統的默認 umask 值修改權限。如果要覆寫預設umask值，則必須設定值。

當發生以下情況時，動作模塊返回一個錯誤：

- 您沒有執行指定修改的權限。
- 資料夾在執行階段不存在。
- 操作模塊在執行操作時遇到錯誤。

輸入

原始	Description (描述)	Type	必要	預設值	可接受值	支援所有平台
path	資料夾路徑。	字串	是	N/A	N/A	在視窗上不支援。
permissions	資料夾權限。	字串	是	N/A	N/A	在視窗上不支援。
recursive	當設定為時，會覆寫資料夾所有內容修改權限的預設行為false。	Boolean	否	true	N/A	在視窗上不支援。

輸入範例：設定資料夾權限

```
- name: SettingFolderPermissions
  action: SetFolderPermissions
  inputs:
    - path: SampleFolder/
      permissions: 0777
```

輸入示例：設置文件夾權限，而不修改文件夾的所有內容的權限

```
- name: SettingFolderPermissionsNoRecursive
  action: SetFolderPermissions
  inputs:
    - path: /home/UserName/SampleFolder/
      permissions: 777
      recursive: false
```

輸出

無。

軟體安裝動作

本節說明執行軟體安裝動作指令和指示的動作模組。

IAM 要求

如果您的安裝下載路徑是 S3 URI，則與執行個體設定檔相關聯的 IAM 角色必須具有執行S3Download動作模組的權限。若要授予所需權限，請將 S3:GetObject IAM 政策附加到與執行個體設定檔相關聯的 IAM 角色，然後指定值區的路徑。例如，*arn:aws:s3:::BucketName/**)。

複雜的 MSI 輸入

如果您的輸入字串包含雙引號字元 (")，您必須使用下列其中一種方法，以確保正確解譯這些字元：

- 您可以在字串外部使用單引號 (') 來包含它，並在字串內使用雙引號 (「)，如下列範例所示。

  ```
  properties:
    COMPANYNAME: '"Acme ""Widgets"" and ""Gizmos."""'
  ```

  在這種情況下，如果您需要在字符串中使用撇號，則必須將其轉義。這意味著在撇號之前使用另一個單引號 (') 。

- 您可以在字符串外部使用雙引號（「）來包含它。你可以轉義你的字符串內的任何雙引號，使用反斜杠字符（\），如下面的例子。

```
properties:
  COMPANYNAME: "\"Acme \"\"Widgets\"\" and \"\"Gizmos.\"\"\""
```

這兩種方法都會`COMPANYNAME="Acme ""Widgets"" and ""Gizmos."""`將值傳遞給命msiexec令。

軟體安裝動作模組

- 安裝微星
- 解除安裝 MSI

安裝微星

該`InstallMSI`操作模塊使用 MSI 文件安裝 Windows 應用程序。您可以使用本機路徑、S3 物件 URI 或網頁 URL 來指定 MSI 檔案。重新開機選項可設定系統的重新開機行為。

AWS TOE 根據動作模組的輸入參數產生msiexec指令。`path`(MSI 檔案位置) 和 `logFile` (記錄檔位置) 輸入參數的值必須以引號（「) 括住。

下列 MSI 結束代碼視為成功：

- 0 (成功案例)
- 1614 (已解除安裝的錯誤產品)
- 1641 (已啟動重新開機)
- 3010 (需要重新開機)

輸入

原始	Description (描述)	Type	必要	預設值	可接受值
`path`	使用下列其中一項指定 MSI 檔案位置：	字串	是	N/A	N/A

原始	Description (描述)	Type	必要	預設值	可接受值
	• 本機檔案路徑。路徑可以是絕對路徑或相對路徑 • 有效的 S3 物件 URI。 • 一個遵循 RFC 3986 標準的有效網頁網址 (建議使用 HTTPS)。 允許鏈結運算式。				

原始	Description (描述)	Type	必要	預設值	可接受值
reboot	設定成功執行動作模組後的系統重新開機行為。 設定： • Force— msiexec 命令成功執行後，啟動系統重新開機。 • Allow— 如果msiexec命令傳回指出需要重新開機的結束代碼，則啟動系統重新開機。 • Skip— 將資訊訊息記錄至console.log 檔案，指出已略過重新開機。此選項可防止重新啟	字串	否	Allow	Allow, Force, Skip

原始	Description (描述)	Type	必要	預設值	可接受值
	動，即使該 msiexec命令返回指示需要重新啟動的退出代碼。				
`logOptions`	指定用於 MSI 安裝記錄的選項。指定的旗標會傳遞至 MSI 安裝程式，以及啟用記錄的/L命令列參數。如果未指定任何旗標， AWS TOE 會使用預設值。 如需 MSI 記錄檔選項的詳細資訊，請參閱 Microsoft 安裝程式產品文件中的命令列選項。	字串	否	*VX	`i,w,e,a,r,u,c,m,o,p,v,x,+,!,*`

原始	Description (描述)	Type	必要	預設值	可接受值
`logFile`	記錄檔位置的絕對或相對路徑。如果記錄檔路徑不存在，則會建立該路徑。如果未提供記錄檔路徑，則 AWS TOE 不會儲存 MSI 安裝記錄。	字串	否	N/A	N/A

原始	Description (描述)	Type	必要	預設值	可接受值
properties	MSI 記錄內容鍵值對，例如：TARGETDIR: "C:\target\location" 注意：不允許修改下列屬性： • REBOOT="ReallySuppress" • REINSTALLMODE="ecmus" • REINSTALL="ALL"	映射 [字符串] 字符串	否	N/A	N/A

原始	Description (描述)	Type	必要	預設值	可接受值
`ignoreAuthenticodeSignatureErrors`	此旗標可忽略 path 中指定之安裝程式的驗證碼簽章驗證錯誤。該Get-AuthenticodeSignature命令用於驗證安裝程序。 設定： • `true`— 會忽略驗證錯誤並執行安裝程式。 • `false`— 不會忽略驗證錯誤。只有在驗證成功時，安裝程式才會執行。這是預設行為。	Boolean	否	`false`	`true`, `false`

原始	Description (描述)	Type	必要	預設值	可接受值
allowUnsignedInstaller	允許執行路徑中指定的未簽署安裝程式的旗標。該Get-AuthenticodeSignature命令用於驗證安裝程序。 設定： • true— 忽略命Get-AuthenticodeSignature令傳回的NotSigne 狀態並執行安裝程式。 • false— 需要簽署安裝程式。未簽署的安裝程式將不會執行。這是預設行為。	Boolean	否	false	true, false

範例

下列範例會根據您的安裝路徑，顯示元件文件輸入區段的變化。

輸入範例：本機文件路徑安裝

```
- name: local-path-install
  steps:
    - name: LocalPathInstaller
      action: InstallMSI
      inputs:
        path: C:\sample.msi
        logFile: C:\msilogs\local-path-install.log
        logOptions: '*VX'
        reboot: Allow
        properties:
          COMPANYNAME: '"Amazon Web Services"'
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: true
```

輸入範例：Amazon S3 路徑安裝

```
- name: s3-path-install
  steps:
    - name: S3PathInstaller
      action: InstallMSI
      inputs:
        path: s3://<bucket-name>/sample.msi
        logFile: s3-path-install.log
        reboot: Force
        ignoreAuthenticodeSignatureErrors: false
        allowUnsignedInstaller: true
```

輸入範例：Web 路徑安裝

```
- name: web-path-install
  steps:
    - name: WebPathInstaller
      action: InstallMSI
      inputs:
        path: https://<some-path>/sample.msi
        logFile: web-path-install.log
        reboot: Skip
        ignoreAuthenticodeSignatureErrors: true
```

```
        allowUnsignedInstaller: false
```

輸出

以下是動InstallMSI作模組輸出的範例。

```
{
 "logFile": "web-path-install.log",
 "msiExitCode": 0,
 "stdout": ""
}
```

解除安裝 MSI

UninstallMSI動作模組可讓您使用 MSI 檔案移除 Windows 應用程式。您可以使用本機檔案路徑、S3 物件 URI 或網頁 URL 來指定 MSI 檔案位置。重新開機選項可設定系統的重新開機行為。

AWS TOE 根據動作模組的輸入參數產生msiexec指令。產生msiexec指令時，MSI 檔案位置 (pathlogFile) 和記錄檔位置 () 會明確用雙引號 (「) 括住。

下列 MSI 結束代碼視為成功：

- 0 (成功案例)
- 1605 (錯誤 _ 未知的產品)
- 1614 (已解除安裝的錯誤產品)
- 1641 (已啟動重新開機)
- 3010 (需要重新開機)

輸入

原始	Description (描述)	Type	必要	預設值	可接受值
path	使用下列其中一項指定 MSI 檔案位置： •	字串	是	N/A	N/A

原始	Description (描述)	Type	必要	預設值	可接受值
	本機檔案路徑。路徑可以是絕對路徑或相對路徑。 • 有效的 S3 物件 URI。 • 一個遵循 RFC 3986 標準的有效網頁網址 (建議使用 HTTPS)。 允許鏈結運算式。				

原始	Description (描述)	Type	必要	預設值	可接受值
reboot	設定成功執行動作模組後的系統重新開機行為。 設定： • Force— msiexec 命令成功執行後，啟動系統重新開機。 • Allow— 如果msiexec命令傳回指出需要重新開機的結束代碼，則啟動系統重新開機。 • Skip— 將資訊訊息記錄至console.log 檔案，指出已略過重新開機。此選項可防止重新啟	字串	否	Allow	Allow, Force, Skip

原始	Description (描述)	Type	必要	預設值	可接受值
	動，即使該 msiexec命令返回指示需要重新啟動的退出代碼。				
logOptions	指定用於 MSI 安裝記錄的選項。指定的旗標會傳遞至 MSI 安裝程式，以及啟用記錄的/L命令列參數。如果未指定任何旗標， AWS TOE 會使用預設值。 如需 MSI 記錄檔選項的詳細資訊，請參閱 Microsoft 安裝程式產品文件中的命令列選項。	字串	否	*VX	i,w,e,a,r,u,c,m,o,p,v,x,+,!,*

原始	Description (描述)	Type	必要	預設值	可接受值
`logFile`	記錄檔位置的絕對或相對路徑。如果記錄檔路徑不存在，則會建立該路徑。如果未提供記錄檔路徑，則 AWS TOE 不會儲存 MSI 安裝記錄。	字串	否	N/A	N/A

原始	Description (描述)	Type	必要	預設值	可接受值
properties	MSI 記錄內容鍵值對，例如：TARGETDIR: "C:\target\location" 注意：不允許修改下列屬性： • REBOOT="ReallySuppress" • REINSTALLMODE="ecmus" • REINSTALL="ALL"	映射 [字符串] 字符串	否	N/A	N/A

原始	Description (描述)	Type	必要	預設值	可接受值
`ignoreAuthenticodeSignatureErrors`	此旗標可忽略 path 中指定之安裝程式的驗證碼簽章驗證錯誤。該Get-AuthenticodeSignature命令用於驗證安裝程序。 設定： • `true`— 會忽略驗證錯誤並執行安裝程式。 • `false`— 不會忽略驗證錯誤。只有在驗證成功時，安裝程式才會執行。這是預設行為。	Boolean	否	`false`	`true`, `false`

原始	Description (描述)	Type	必要	預設值	可接受值
allowUnsignedInstaller	允許執行路徑中指定的未簽署安裝程式的旗標。該Get-AuthenticodeSignature命令用於驗證安裝程序。 設定： • true— 忽略命Get-AuthenticodeSignature令傳回的NotSigne 狀態並執行安裝程式。 • false— 需要簽署安裝程式。未簽署的安裝程式將不會執行。這是預設行為。	Boolean	否	false	true, false

範例

下列範例會根據您的安裝路徑，顯示元件文件輸入區段的變化。

輸入範例：移除本機文件路徑安裝

```
- name: local-path-uninstall
  steps:
    - name: LocalPathUninstaller
      action: UninstallMSI
      inputs:
        path: C:\sample.msi
        logFile: C:\msilogs\local-path-uninstall.log
        logOptions: '*VX'
        reboot: Allow
        properties:
          COMPANYNAME: '"Amazon Web Services"'
        ignoreAuthenticodeSignatureErrors: true
        allowUnsignedInstaller: true
```

輸入範例：移除 Amazon S3 路徑安裝

```
- name: s3-path-uninstall
  steps:
    - name: S3PathUninstaller
      action: UninstallMSI
      inputs:
        path: s3://<bucket-name>/sample.msi
        logFile: s3-path-uninstall.log
        reboot: Force
        ignoreAuthenticodeSignatureErrors: false
        allowUnsignedInstaller: true
```

輸入示例：刪除 Web 路徑安裝

```
- name: web-path-uninstall
  steps:
    - name: WebPathUninstaller
      action: UninstallMSI
      inputs:
        path: https://<some-path>/sample.msi
        logFile: web-path-uninstall.log
        reboot: Skip
        ignoreAuthenticodeSignatureErrors: true
```

```
        allowUnsignedInstaller: false
```

輸出

以下是動UninstallMSI作模組輸出的範例。

```
{
 "logFile": "web-path-uninstall.log",
 "msiExitCode": 0,
 "stdout": ""
}
```

系統動作模組

下節說明執行檔案系統動作指令和指示的動作模組。

系統動作模組

- 重新開機
- SetRegistry
- 更新

重新開機

重新啟動動作模組會重新啟動執行個體。它有一個可配置的選項來延遲重新啟動的開始。默認情況下delaySeconds，設置為0，這意味著沒有延遲。重新啟動動作模組不支援步驟逾時，因為執行個體重新啟動時不適用。

如果應用程式是由系統管理員代理程式呼叫，它會將結束代碼 (3010適用於 Windows、194 Linux) 交給系統管理員代理程式。系統管理員代理程式會依照從指令碼重新啟動受管理執行個體中所述來處理系統

如果在主機上呼叫應用程式作為獨立處理作業，它會儲存目前的執行狀態、設定重新開機後自動執行觸發程式，以便在重新開機後重新執行應用程式，然後重新啟動系統。

重新啟動後自動運行觸發：

- 窗戶。 AWS TOE 創建一個 Windows 任務計劃程序條目，該條目具有自動運行的觸發器SystemStartup
- Linux. AWS TOE 在 crontab 中添加一個作業，該作業在系統重新啟動後自動運行。

```
@reboot /download/path/awstoe run --document s3://bucket/key/doc.yaml
```

應用程式啟動時會清除此觸發程序。

> **重試**
>
> 依預設，重試次數上限是設定為「Systems ManagerCommandRetryLimit」。如果重新啟動次數超過重試限制，則自動化會失敗。您可以編輯系統管理員代理程式設定檔 (Mds.CommandRetryLimit) 來變更限制。請參閱系統管理員代理程式開源中的執行階段組態。

若要使用「重新開機」動作模組，對於包含重新開機的步驟 exitcode (例如，3010)，您必須執行應用程式二進位檔為sudo user。

輸入

原始	Description (描述)	Type	必要	預設
delaySeconds	在啟動重新開機之前延遲特定的時間。	Integer	否	0

輸入範例：重新啟動步驟

```
- name: RebootStep
  action: Reboot
  onFailure: Abort
  maxAttempts: 2
  inputs:
    delaySeconds: 60
```

輸出

無。

當重新啟動模組完成時，Image Builder 會繼續執行組建中的下一個步驟。

SetRegistry

SetRegistry操作模塊接受輸入列表，並允許您設置指定註冊表項的值。如果登錄機碼不存在，則會在定義的路徑中建立該機碼。此功能僅適用於視窗。

輸入

原始	Description (描述)	Type	必要
path	登錄機碼的路徑。	字串	是
name	登錄機碼的名稱。	字串	是
value	登錄機碼的值。	字串/數字/陣列	是
type	登錄機碼的值類型。	字串	是

支援的路徑前置字元

- HKEY_CLASSES_ROOT / HKCR:
- HKEY_USERS / HKU:
- HKEY_LOCAL_MACHINE / HKLM:
- HKEY_CURRENT_CONFIG / HKCC:
- HKEY_CURRENT_USER / HKCU:

支援的 類型

- BINARY
- DWORD
- QWORD
- SZ
- EXPAND_SZ
- MULTI_SZ

輸入範例：設定登錄機碼值

```
  - name: SetRegistryKeyValues
```

```
    action: SetRegistry
    maxAttempts: 3
    inputs:
      - path: HKLM:\SOFTWARE\MySoftWare
        name: MyName
        value: FirstVersionSoftware
        type: SZ
      - path: HKEY_CURRENT_USER\Software\Test
        name: Version
        value: 1.1
        type: DWORD
```

輸出

無。

更新

更新操作模塊添加了對安裝視窗和 Linux 更新的支持。它默認安裝所有可用的更新。或者，您可以為要安裝的動作模組配置一個或多個特定更新的清單。您也可以指定要從安裝中排除的更新。

如果同時提供「包含」和「排除」清單，則產生的更新清單只能包含未列在「排除」清單中的「包含」清單中列出的更新清單。

> **Note**
>
> 更新操作系統不支持 Amazon 2023（AL2023）。我們建議您將基礎 AMI 更新為每個發行版本隨附的新版本。如需其他替代方案，請參閱 Amazon Linux 2023 使用者指南中的控制從主要和次要版本接收到的更新。

- 窗戶。更新會從目標電腦上設定的更新來源安裝。
- Linux. 應用程序檢查 Linux 平台中支持的軟件包管理器，並使用`yum`或`apt-get`軟件包管理器。如果兩者都不受支援，則會傳回錯誤。您應該擁有執行 UpdateOS 動作模組的`sudo`權限。如果您沒有`sudo`權限，`error.Input`則返回。

輸入

原始	Description (描述)	Type	必要
`include`		字串清單	否

原始	Description (描述)	Type	必要
	對於視窗，您可以指定下列項目： • 要包含在可能安裝的更新清單中的一或多個 Microsoft 知識庫 (KB) 文件識別碼。有效格式為 KB1234567 或 1234567。 • 使用萬用字元值 (*) 的更新名稱。有效格式為 Security* 或 *Security* 。 對於 Linux，您可以指定一或多個要包含在安裝更新清單中的套件。		

原始	Description (描述)	Type	必要
exclude	對於視窗，您可以指定下列項目： • 要包含在要從安裝之外排除的更新清單中的一或多個 Microsoft 知識庫 (KB) 文件識別碼。有效格式為 KB1234567 或 1234567。 • 使用萬用字元 (*) 值的更新名稱。有效格式為：Security* 或*Security * 。 對於 Linux，您可以指定一或多個要從安裝更新清單中排除的套件。	字串清單	否

輸入示例：添加對安裝 Linux 更新的支持

```
- name: UpdateMyLinux
  action: UpdateOS
  onFailure: Abort
  maxAttempts: 3
  inputs:
    exclude:
      - ec2-hibinit-agent
```

輸入示例：添加對安裝 Windows 更新的支持

```
- name: UpdateWindowsOperatingSystem
  action: UpdateOS
  onFailure: Abort
  maxAttempts: 3
  inputs:
    include:
      - KB1234567
      - '*Security*'
```

輸出

無。

配置 AWS TOE 運行命令的輸入

為了簡化命令的命 AWS TOE run令行輸入，您可以在 JSON 格式輸入配置檔案中包含命令參數和選項的設定，並具有副.json檔名。 AWS TOE 可以從下列其中一個位置讀取您的檔案：

- 本機檔案路徑 (*. /##.json##*
- 一個 S3 存儲桶（*S3#///##.json* <bucket-path><bucket-name>）。

當您輸入指run令時，您可以使用--config參數指定輸入規劃檔案。例如：

```
awstoe run --config <file-path>/config.json
```

輸入配置文件

輸入組態 JSON 檔案包含您可以透過run命令參數和選項直接提供之所有設定的索引鍵值配對。如果您同時在輸入組態檔案和指run令中指定設定作為參數或選項，則適用下列優先順序規則：

優先順序規則

1. 透過參數或選項直接提供給中run AWS CLI指令的設定會取代在輸入組態檔案中針對相同設定定義的任何值。
2. 輸入規劃檔案中的設定會取代元件預設值。
3. 如果沒有其他設定傳遞至元件文件，它可以套用預設值 (如果有的話)。

此規則有兩個例外狀況 — 文件和參數。這些設定在輸入組態和指令參數中的運作方式不同。如果使用輸入組態檔案，則不得將這些參數直接指定給指run令。這樣做會產生錯誤。

元件設定

輸入組態檔案包含下列設定。為了簡化您的文件，您可以省略任何不需要的可選設置。除非另有說明，否則所有設定均為選用

- cwIgnoreFailures(布林值) — 忽略記錄 CloudWatch 檔中的記錄失敗。
- cwLogGroup(字串) — 記 CloudWatch 錄檔的LogGroup名稱。
- cwLogRegion(字串) — 適用於 CloudWatch 記錄的 AWS 區域。
- cwLogStream（字符串）- CloudWatch 日誌的LogStream名稱，用於指示流式傳輸console.log文件的 AWS TOE 位置。
- 文件 3 BucketOwner (字串) — S3 URI 文件的儲存貯體擁有者帳戶 ID。
- 文件 (必要的物件陣列) — JSON 物件陣列，代表命 AWS TOE run令正在執行的 YAML 元件文件。至少必須指定一個元件文件。

  每個物件都包含下列欄位：

  - 路徑 (字串，必要) — YAML 元件文件的檔案位置。這必須是下列其中一項：
    - 本機檔案路徑 (*. /component-doc-example. ###)*.
    - 一個 S3 URI (s3://*bucket/key*)。
    - *Image Builder #### ARN ###ARN#AW########### 2#1234567890## #/2021.12.02/1##my-example-component*
  - parameters (物件陣列) — 鍵值對物件的陣列，每個物件都代表run指令執行元件文件時所傳入的元件特定參數。元件的參數是可選的。零組件文件可能已定義參數，也可能沒有定義參數。

    每個物件都包含下列欄位：

    - name (字串，必要) — 元件參數的名稱。
    - value (字串，必要) — 要傳入至具名參數之元件文件的值。

    若要深入瞭解元件參數，請參閱定義和引用變量 AWS TOE頁面中的參數段落。
- ExecutId（字符串）-這是適用於當前run命令執行的唯一 ID。此 ID 包含在輸出檔和記錄檔名稱中，以唯一識別這些檔案，並將它們連結至目前的指令執行。如果省略此設定，則 AWS TOE 會產生 GUID。

- LogDirectory (字串) — AWS TOE 儲存此命令執行所有記錄檔的目的地目錄。依預設，此目錄位於下列父目錄內：TOE_<DATETIME>_<EXECUTIONID>。如果未指定記錄目錄， AWS TOE 會使用目前的工作目錄 (.)。
- LogS3 BucketName (字串) — 如果元件日誌存放在 Amazon S3 (建議使用)，請將元件應用程式日誌 AWS TOE 上傳到此參數中指定的 S3 儲存貯體。
- LogS3 BucketOwner (字串) — 如果元件日誌存放在 Amazon S3 (建議使用)，這是 AWS TOE 寫入日誌檔的儲存貯體的擁有者帳戶 ID。
- LogS3 KeyPrefix (字串) — 如果元件日誌存放在 Amazon S3 (建議使用)，這是儲存貯體中日誌位置的 S3 物件 key prefix 置詞。
- parameters (物件陣列) — 鍵值配對物件的陣列，代表全域套用至目前run命令執行中包含的所有元件的參數。
  - name (字串，必要) — 全域參數的名稱。
  - value (字串，必要) — 要傳入至具名參數之所有元件文件的值。
- 階段 (字串) — 以逗號分隔的清單，指定要從 YAML 元件文件執行的階段。如果元件文件包含其他階段，則這些階段將不會執行。
- 狀態目錄 (字串) — 儲存狀態追蹤檔案的檔案路徑。
- trace (布林值) — 啟用主控台的詳細記錄功能。

範例

下列範例顯示一個輸入組態檔案，該檔案會針對兩個零組件文件執行build和test階段：sampledoc.yaml、和conversation-intro.yaml。每個元件文件都有一個僅適用於本身的參數，並且兩者都使用一個共用參數。project參數適用於兩個零組件文件。

```
{
   "documents": [
     {
       "path": "<file path>/awstoe/sampledoc.yaml>",
       "parameters": [
         {
           "name": "dayofweek",
           "value": "Monday"
         }
       ]
     },
     {
```

```
        "path": "<file path>/awstoe/conversation-intro.yaml>",
        "parameters": [
          {
            "name": "greeting",
            "value": "Hello, HAL."
          }
        ]
      }
    ],
    "phases": "build,test",
    "parameters": [
      {
        "name": "project",
        "value": "examples"
      }
    ],
    "cwLogGroup": "<log_group_name>",
    "cwLogStream": "<log_stream_name>",
    "documentS3BucketOwner": "<owner_aws_account_number>",
    "executionId": "<id_number>",
    "logDirectory": "<local_directory_path>",
    "logS3BucketName": "<bucket_name_for_log_files>",
    "logS3KeyPrefix": "<key_prefix_for_log_files>",
    "logS3BucketOwner": "<owner_aws_account_number>"
}
```

Windows 適用的代理商套件管理元件

AWS Systems Manager 代理商可協助您將軟體封裝並發佈至 AWS Systems Manager 受管理節點。您可以封裝並發佈自己的軟體，或使用代理商尋找並發佈 AWS提供的代理程式軟體套件。如需系統管理員代理商的詳細資訊，請參閱AWS Systems Manager 使用者指南中的「代理AWS Systems Manager 商」

代理商的受管理元件

下列 Image Builder 受管理元件使用 AWS Systems Manager 散發程式在 Windows 執行個體上安裝應用程式套件

- 受distributor-package-windows管理元件會使用 AWS Systems Manager 散發程式來安裝您在 Windows 映像建置執行個體上指定的應用程式套件。若要在方案中包含此元件時設定參數，請參閱設定distributor-package-windows為獨立元件。

- 此aws-vss-components-windows元件會使用 AWS Systems Manager 散發程式在 Windows 映像建置執行個體上安裝AwsVssComponents套件。若要在方案中包含此元件時設定參數，請參閱設定aws-vss-components-windows為獨立元件。

如需有關如何在 Image Builder 方案中使用受管理元件的建立影像配方的新版本詳細資訊，請參閱映像配方或建立容器配方的新版本容器配方。如需有關AwsVssComponents套件的詳細資訊，請參閱 Amazon EC2 Windows 執行個體使用者指南中的建立 VSS 應用程式一致性快照。

必要條件

使用依賴 Systems Manager 散發商安裝應用程式套件的 Image Builder 元件之前，您必須確定符合下列先決條件。

- 使用 Systems Manager 散發商在執行個體上安裝應用程式套件的 Image Builder 元件需要呼叫 Systems Manager API 的權限。在使用 Image Builder 方案中的元件之前，您必須建立授與權限的 IAM 政策和角色。若要設定權限，請參閱設定系統管理員代理商權。

> **Note**
>
> Image Builder 目前不支援重新啟動執行個體的 Systems Manager 散發程式套件。例如，AWSNVMeAWSPVDrivers、和AwsEnaNetworkDriver散發者套件會重新啟動執行個體，因此不允許執行個體。

設定系統管理員代理商權

使用該distributor-package-windows元件的元件和其他元件 (例如aws-vss-components-windows) 需要建置執行個體的額外權限才能執行。組建執行個體必須能夠呼叫 Systems Manager API，才能開始「代理商」安裝並輪詢結果。

請遵循中的下列程序 AWS Management Console 來建立自訂 IAM 政策和角色，以授與 Image Builder 元件從組建執行個體安裝 Systems Manager 散發者套件的權限。

步驟 1：建立策略

建立代理商許可的 IAM 政策。

1. 在 https://console.aws.amazon.com/iam/ 中開啟 IAM 主控台。

2. 在導覽窗格中，選擇 Policies (政策)，然後選擇 Create policy (建立政策)。
3. 在 [建立原則] 頁面上，選擇 [JSON] 索引標籤，然後以下列 JSON 政策取代預設內容、視需要取代分割區、區域和帳戶 ID，或使用萬用字元。

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowDistributorSendCommand",
   "Effect": "Allow",
   "Action": [
    "ssm:SendCommand"
   ],
   "Resource": [
    "arn:${AWS::Partition}:ssm:${AWS::Region}::document/AWS-ConfigureAWSPackage",
    "arn:${AWS::Partition}:ec2:${AWS::Region}:${AWS::AccountId}:instance/*"
   ]
  },
  {
   "Sid": "AllowGetCommandInvocation",
   "Effect": "Allow",
   "Action": [
    "ssm:GetCommandInvocation"
   ],
   "Resource": [
    "*"
   ]
  }
 ]
}
```

4. 選擇 Review policy (檢閱政策)。
5. 在 Name (名稱) 中，輸入名稱來識別此政策，例如 *InvokeDistributor* 或另一個您喜好的名稱。
6. (選用) 對於 Description (描述)，輸入角色目的之描述。
7. 選擇 Create policy (建立政策)。

步驟 2：建立角色

建立代理商許可的 IAM 角色。

1. 在 IAM 主控台導覽窗格中，選擇 [角色]，然後選擇 [建立角色]。
2. 在 Select type of trusted entity (選取可信任執行個體類型) 下，選擇 AWS 服務。
3. 緊接在 Choose the service that will use this role (選擇將使用此角色的服務) 下，選擇 EC2，然後選擇 Next: Permissions (下一步：許可)。
4. 在 Select your use case (選取您的使用案例) 下，選取 EC2，然後選取 Next: Permissions (下一步：許可)。
5. 在原則清單中，選取 Amazon ManagedInstanceCore SSM 旁邊的核取方塊。(如果您需要縮減清單，請在搜尋方塊中輸入 SSM。)
6. 在此政策清單中，選擇 EC2 旁邊的核取方塊InstanceProfileForImageBuilder。(如果您需要縮減清單，請在搜尋方塊中輸入 ImageBuilder。)
7. 選擇 Next: Tags (下一步：標籤)。
8. (選擇性) 新增一或多個標籤索引鍵值配對以組織、追蹤或控制此角色的存取權，然後選擇下一步：檢閱。
9. 對於 Role name (角色名稱)，輸入角色的名稱，例如 *InvokeDistributor* 或另一個您喜好的名稱。
10. (選用) 對於 Role description (角色描述)，將預設文字以此角色目的描述取代。
11. 選擇 Create role (建立角色)。系統會讓您回到 Roles (角色) 頁面。

步驟 3：將策略附加到角色

設定經銷商許可的最後一步是將 IAM 政策附加到 IAM 角色。

1. 在 IAM 主控台的 [角色] 頁面中，選擇您剛建立的角色。角色的 Summary (摘要) 頁面隨即開啟。
2. 選擇 Attach policies (連接政策)。
3. 搜尋您在上一個程序中建立的原則，然後選取名稱旁邊的核取方塊。
4. 選擇連接政策。

在 Image Builder 基礎結構組態資源中，針對包含使用 Systems Manager 散發程式之元件的任何映像檔使用此角色。如需詳細資訊，請參閱 建立基礎架構組態。

設定distributor-package-windows為獨立元件

若要在方案中使用distributor-package-windows元件，請設定下列參數來設定要安裝的套件。

> **Note**
>
> 在方案中使用distributor-package-windows元件之前，您必須確定符合所必要條件有元件。

- 動作 (必要) — 指定要安裝還是解除安裝套件。有效值包括 Install 與 Uninstall。值預設為Install。
- PackageName(必要) — 要安裝或解除安裝的散發者套件名稱。如需有效套件名稱的清單，請參閱尋找經銷商套件。
- PackageVersion(選擇性) — 要安裝的散發者套件版本。 PackageVersion 預設為建議的版本。
- AdditionalArguments(選擇性) — JSON 字串，其中包含可提供給指令碼以安裝、解除安裝或更新套件的其他參數。如需詳細資訊，請參閱「Systems Manager 命令」文件外掛程式參考頁面的「AWS：設定封裝輸入」一節中的其他引數。

設定aws-vss-components-windows為獨立元件

當您在方案中使用aws-vss-components-windows元件時，您可以選擇性地將PackageVersion參數設定為使用特定版本的AwsVssComponents套件。當您省略此參數時，元件預設會使用建議的AwsVssComponents套件版本。

> **Note**
>
> 在方案中使用aws-vss-components-windows元件之前，您必須確定符合所必要條件有元件。

尋找經銷商套件

Amazon 和第三方提供公用套件，您可以透過系統管理員代理商安裝這些套件。

若要在中檢視可用的套件 AWS Management Console，請登入AWS Systems Manager 主控台，然後從導覽窗格中選擇「散發者」。「代理商」頁面會顯示您可以使用的所有套件。若要取得有關列示可用套件的詳細資訊 AWS CLI，請參閱《使用指南》中的〈檢視套件 (命令列)AWS Systems Manager〉。

您也可以建立自己的私人系統管理員代理商套件。若要取得更多資訊，請參閱《使用指南》中的AWS Systems Manager 〈建立封裝〉。

CIS 硬化組件

互聯網安全中心（CIS）是一個社區驅動的非營利組織。他們的網絡安全專家共同努力制定 IT 安全指南，以保護公共和私人組織免受網絡威脅。他們在全球公認的一套最佳實踐，稱為 CIS 基準測試，可幫助世界各地的 IT 組織安全地配置他們的系統。如需熱門文章、部落格文章、播客、網路研討會和白皮書，請參閱網際網路安全中心網站上的 CIS 見解。

CIS 基準參考指標

CIS 會建立並維護一組稱為 CIS 基準測試的組態準則，為特定技術 (包括作業系統、雲端平台、應用程式、資料庫等) 提供組態最佳實務。獨聯體基準測試通過 PCI DSS，HIPAA，DoD 雲計算 SRG，FISMA，DFAR 和 FEDRAMP 等組織和標準認可為行業標準。若要深入瞭解，請參閱網際網路安全中心網站上的 CIS 基準測試。

CIS 硬化組件

當您在中訂閱 CIS 強化映像時 AWS Marketplace，您也可以存取相關的強化元件，該元件會執行指令碼，以針對您的組態強制執行 CIS 效能標竿第 1 級準則。CIS 組織擁有並維護 CIS 強化組件，以確保它們反映了最新的準則。

> **Note**
>
> CIS 強化元件不遵循 Image Builder 配方中的標準元件排序規則。CIS 強化元件一律會持續執行，以確保基準測試會針對您的輸出影像執行。

適用於 EC2 Image Builder 的 Amazon 受管 STIG 強化元件

安全性技術實作指南 (STIG) 是國防資訊系統機構 (DISA) 建立的組態強化標準，用於保護資訊系統與軟體的安全。若要讓您的系統符合 STIG 標準，您必須安裝、設定和測試各種安全設定。

Image Builder 提供 STIG 強化元件，協助您更有效率地建置符合 STIG 基準標準的相容映像。這些 STIG 元件會掃描設定錯誤並執行修復指令碼。使用符合 STIG 標準的元件不收取額外費用。

> **Important**
>
> 在少數例外情況下，STIG 強化元件不會安裝協力廠商套件。如果執行個體上已安裝協力廠商套件，而且 Image Builder 支援該套件的相關 STIG，則強化元件會套用這些套件。

此頁面列出 Image Builder 支援的所有 STIG，這些 STI 會套用至 Image Builder 在您建置和測試新映像時啟動的 EC2 執行個體。如果您想要將其他 STIG 設定套用至映像，您可以建立自訂元件來進行設定。如需自訂元件及其建立方式的更多資訊，請參閱〈〉使用 Image Builder 管理元件。

建立映像時，STIG 強化元件會記錄是否套用或略過支援的 STIG。建議您針對使用 STIG 強化元件的映像檔檢閱映像產生器記錄。如需如何存取和檢閱 Image Builder 記錄的詳細資訊，請參閱疑難排解管線建。

合規層級

- 高 (類別 I)

  最嚴重的風險 包含任何可能導致機密性、可用性或完整性遺失的漏洞。

- 中 (類別 II)

  包括任何可能導致機密性、可用性或完整性喪失的弱點，但可以減輕風險。

- 低 (類別 III)

  任何會降低防範機密性、可用性或完整性遺失之措施的漏洞。

主題

- 視窗 STIG 強化元件
- 視窗版本歷史記錄
- 強化元件
- STIG 版本歷史記錄日誌
- SCAP 合規性驗證程式元件

視窗 STIG 強化元件

AWS TOE Windows STIG 強化元件是專為獨立伺服器所設計，並套用本機群組原則。STIG 相容強化元件會 InstallRoot 從 Windows 基礎結構上的國防部 (DoD) 安裝，以下載、安裝及更新 DoD 憑證。他們也會移除不必要的憑證，以維持 STIG 合規性。目前，下列版本的視窗伺服器支援 STIG 基準：

本節列出每個 Windows STIG 強化元件的目前設定值，接著是版本歷史記錄記錄。

柱狀構建視窗-低版本 2022.4.x

下列清單包含強化元件套用至基礎結構的 STIG 設定。如果支援的設定不適用於您的基礎結構，強化元件會略過該設定，然後繼續進行。例如，某些 STIG 設定可能不適用於獨立伺服器。組織特定的原則也會影響強化元件套用的設定，例如系統管理員必須檢閱文件設定。

如需 Windows STIG 的完整清單，請參閱 STIGs Document Library (STIG 文件庫)。如需有關如何檢視完整清單的詳細資訊，請參閱 STIG 檢視工具。

- 視窗服務器 2022 STIG 版本 1 版本 1

  V-254335, V-254336, V-254337, V-254338, V-254351, 愿意, 渴望, 和輕 V-254357 V-254363 V-254481
- 視窗服務器 2019 時尚版本 2 發布 5

  V-205691、V-205819、V-205858、V-205859、V-205860、V-205870、V-205871 以及 V-205923
- 視窗服務器 2016 時尚版本 2 發布 5

  V-224916、V-224917、V-224918、V-224919、V-224931、V-224942 以及 V-225060
- 視窗服務器 2012 R2 MS 風格版本 3 版本 5

  V-225537、V-225536、V-225526、V-225525、V-225514、V-225511、V-225490、V-225489、V-225488 以及 V-225250
- Microsoft. NET 框架 4.0 版本 2 版本 2

  對於類別 III 漏洞，沒有 STIG 設定套用至 Microsoft .NET Framework。
- 視窗防火牆樣式版本 2 版本 1

  V-241994, V-241995, V-241996, V-241999,, 渴望, 鼎, 朗, 和熱 V-242000 V-242001 V-242006 V-242007 V-242008
- 互聯網瀏覽器 11 STIG 版本 2 發布 3

V-46477、V-46629 以及 V-97527

- Microsoft 邊緣 STIG 版本 1 版本 6 (僅限視窗伺服器 2022)

  V-235727、V-235731、V-235751、V-235752 和 V-235765

柱狀構建視窗中型版本 2022.4.x

下列清單包含強化元件套用至基礎結構的 STIG 設定。如果支援的設定不適用於您的基礎結構，強化元件會略過該設定，然後繼續進行。例如，某些 STIG 設定可能不適用於獨立伺服器。組織特定的原則也會影響強化元件套用的設定，例如系統管理員必須檢閱文件設定。

如需 Windows STIG 的完整清單，請參閱 STIGs Document Library (STIG 文件庫)。如需有關如何檢視完整清單的詳細資訊，請參閱 STIG 檢視工具。

> **Note**
>
> STIG 建置的視窗中強化元件除了專門針對類別 II 弱點列出的 STIG 設定外，還包括 AWS TOE 適用於 STIG 建置視窗低強化元件的所有 STIG 設定。

- 視窗服務器 2022 STIG 版本 1 版本 1

  包含強化元件適用於類別 III (低) 弱點的所有支援 STIG 設定，以及：

  V-254247, V, 254269, V-254269, V-254271, V-254272, V-254273, V-254274, V-254276, V-254277, V-254278, V, V-254292, V-254300, V-254302, V-254303, V-254304, V-254305, V-254306, V-254307, V-254308, V-254309, V 17, 伏特, 254319, V 型, 254320, V-254321, V-254323, V-254323, 伏特, 254325, V 型, 254326, V-254327, V-254329, VV-254344, V-254345, V-254346, V-254347, V-254349, V-254350, V-254350, V-254356, V-254358, V-254360, V-254361, V 368，254369，V-254370，V-254371，V-254372，V-254373，V-254375，伏特 -254376，V-254379，V-254382，V-254383，V 254438, V-254439, V-254442, V-254443, V-254444, V-254449, V-254449, V-254450, V-254452, V-254453, V-254454, V 型 254455, V 型 254456, V 254464,V-254468, V-254470, V-254471, V-254472,,, 湧, 球, 蘋果, 她, 節制, 節制, 幕, 蒸氣, 向量, 毫秒, 伏特, 伏特, 卷, V 型, 254495, V 型, V 型 254497, V 型 254499, V 型 254501, V 型 254501, V 型 254502, V 型 254502, V 型 254503, V 型 254504, V 型 8, V 型, V 型 254510, 電視 V-254473 V-254476 V-254477 V-254478 V-254479 V-254480 V-254482 V-254483 V-254484 V-254485 V-254486 V-254487 V-254488 V-254489 V-254490 V-254493

- 視窗服務器 2019 時尚版本 2 發布 5

  包含強化元件適用於類別 III (低) 弱點的所有支援 STIG 設定，以及：

  V-205625, V-205626, V-205627, V-205629, V-205633, V-205634, V-205635, V-205636, V-205637, V-205638, V-205639, V V-205655, V-205656, V-205659, V-205660, V-205662, V-205671, V-205672, V-205673, V-205676, V-205678, V, V -205687, V-205688, V-205689, V-205690, V-205693, V-205694, V, 電視, 電視,,V -205729, V-205730, V-205733, V-205747, 電視 -205752, V-205752, V-205756, V-205758, V. V -205770, V-205771, V-205772, V-205773, V-205774, V-205775, V, 205777, V. V-205801, V-205808, V-205809, V-205810, V-205811, V-205813, V-205814, V-205815, V-205816, V-205817, V-205821, VV-205830, V-205832, V-205833, V-205834,,, 湧, 球, 蘋果, 四季, 四千三十一, 節制, 蒸汽, 冰, 伏特, 冰壺, 風格, V-205872, V-205873, V 型 205911, V 型 205912, V 型 205915, V 型 205916, V 型 205916, V 型 205917, V 型 V 型, 和 V-205835 V-205836 V-205837 V-205838 V-205839 V-205840 V-205841 V-205861 V-205863 V-205865 V-205866 V-205867 V-205868 V-205869

- 視窗服務器 2016 時尚版本 2 發布 5

  包含強化元件適用於類別 III (低) 弱點的所有支援 STIG 設定，以及：

  V-224850, V-224853, V-224854, V-224855, V-224856, V-224857, V-224858, V-224859, V-224866, V 224873, V-224881, V-224882, V-224883, V-224885, V-224886, 電視, 電影 -224887, V-224889, V-224889, V 224896, V-224897, V-224898, V-224999, V-224901, V-224902, V, 電影 -224904, V-224904, V-224905, V-224906, V 24912, V-224913, V-二四四四十四, V-二 24915, V-二 24920, V 型電視 -224927, V-224928, V-224930, V-224935, V-224936, V-224937, V-224938, V-224939, V-224940, V, V-224948, V-224949, V-224951, V-224953, 伏特 -224955, 伏特 -224956, 電影 -224957, V-224959, V-224962, 十六、二二二五十七、二二二二五十九、二二二五二十二、二二二五十三、二二二二二五十四、二二二五十九、二五零三四、二五零三三、五五十三三、V 四十一、二百五十四、七五十四、七五十四、七五十四、四百五十五V-225051, V-225052, V-225055, V-225056,, 保證, 湧, 球, 蘋果, 節制, 節制, 蒸氣, 蒸氣, 毫秒, 伏特, 冰壺, 風格, V-225076, 格魯, V 型 225080, V 型 -225082, V 型 225083, V 型 225084, V 型 225088, V 型 V-236000 V-225057 V-225058 V-225061 V-225062 V-225063 V-225064 V-225065 V-225066 V-225067 V-225068 V-225069 V-225072 V-225073 V-225074 V-225078

- 視窗服務器 2012 R2 MS 風格版本 3 版本 5

  包含強化元件適用於類別 III (低) 弱點的所有支援 STIG 設定，以及：

  V-225574, V-225573, V-225572, V-225571, V-225569, 伏特 -225568, 伏特 -225567, V-225566, V 型 25558, 伏特 -225557, V-225554, 伏特 -225553, 伏特 -225551, 伏特 -225549, 伏特 -225549, 伏

特 -225548, V-225545, V, V-225538, V-225535, V-225534, V 型, 二五五十八, V 型,伏特 -225506, 伏特 -225503, V-225502, 伏特 -225501, 伏特 -225500, 伏特 -225494, 伏特 -225478, 伏特, V 25463, V-225461, V-225458, V-225457, V-225456, 伏特 -225455, 伏特 -225453, V-225452, V-225448, V 225411, V-225410, V-225409, V-225408, 伏特 -225406, 伏特 -225405, 伏特 -225402, V-225402, V-225393, V, V-225385, V 型,伏特 -225379, 伏特 -225378, 伏特 -225375, 伏特 -225374, 伏特 -225373, 伏特 -225372, 伏特 -225371, V, V-225349, V-225348, V-225347, 伏特 -225346, 伏特 -225345, 伏特 -225341, 伏特 -225340, V-225337, V 314, V-225305, V 型, 二二五三零三, V 型 V-225283, V-225282, V-225281, V-225280,,, 搶, 球, 蘋果, 早餐, 四季, 鑼, 蒸汽, 蘋果, 冰,,, 卷, 果汁, V-225264, V-225263, V 型 225261, V 型 225260, V 型 225259, 和 V 型 225239 V-225279 V-225278 V-225277 V-225276 V-225275 V-225273 V-225272 V-225271 V-225270 V-225269 V-225268 V-225267 V-225266 V-225265

- Microsoft. NET 框架 4.0 版本 2 版本 2

  包含強化元件適用於類別 III (低) 弱點的所有支援 STIG 設定，以及 V-225238

- 視窗防火牆樣式版本 2 版本 1

  包含強化元件適用於類別 III (低) 弱點的所有支援 STIG 設定，以及：

  V-241989、V-241990、V-241991、V-241993、V-241998 和 kin V-242003

- 互聯網瀏覽器 11 STIG 版本 2 發布 3

  包含強化元件適用於類別 III (低) 弱點的所有支援 STIG 設定，以及：

  V-46473、V-46475、V-46481、V-46483、V-46501、V-46507、V-46509、V-46511、V-46513、V-46515、以及 V-75171

- Microsoft 邊緣 STIG 版本 1 版本 6 (僅限視窗伺服器 2022)

  包含強化元件適用於類別 III (低) 弱點的所有支援 STIG 設定，以及：

  V-235720, V-235721, V-235723, V-235724,, 向, 湧, 球, 蘋果, 四季, 四千〇一年, 蒸汽, 雞, 冰, 伏特, 冰壺, 風格, V-235741, 格鬥, V 型 235743, V 型 235745, V 型 235745, V 型 235747, V 型 235747, V 型 235748, V 型 235750, V 型 235750, V 型 235750, V 型 235750, V 型 235750, V 型電視 235761, 電視 -235763, 電視 -235764, 電視 -235767, 電視 -235768, 電視 -235769, 電視 -235771, 電視 -235772, 電視 -235773, 電視 -235774, 和電視 V-235725 V-235726 V-235728 V-235729 V-235730 V-235732 V-235733 V-235734 V-235735 V-235736 V-235737 V-235738 V-235739 V-235740 V-235742

- 後衛 STIG 版本 2 版本 4 (僅限視窗服務器 2022)

包含強化元件適用於類別 III (低) 弱點的所有支援 STIG 設定，以及：

V-213427, V-213429, V-213430, V-213431,, V-213432, 向前, 球, 蘋果, 四季, 節制, 節制, 包括, 臘, 基督徒, 伏特, 冰壺, 風格, V-213446, V-213447, V 型 213448, V 型 213449, V 型 213451, V 型 213455, V 型 213464, V 型 213464, V 型 213465, V 型 213465, 和 V 型 213465, V 型 V-213433 V-213434 V-213435 V-213436 V-213437 V-213438 V-213439 V-213440 V-213441 V-213442 V-213443 V-213444 V-213445

柱狀構建視窗-高版本 2022.4.x

下列清單包含強化元件套用至基礎結構的 STIG 設定。如果支援的設定不適用於您的基礎結構，強化元件會略過該設定，然後繼續進行。例如，某些 STIG 設定可能不適用於獨立伺服器。組織特定的原則也會影響強化元件套用的設定，例如系統管理員必須檢閱文件設定。

如需 Windows STIG 的完整清單，請參閱 STIGs Document Library (STIG 文件庫)。如需有關如何檢視完整清單的詳細資訊，請參閱 STIG 檢視工具。

> **Note**
>
> STIG 建置的視窗高強化元件除了專門針對類別 I 弱點列出的 STIG 設定外，還包括 AWS TOE 適用於 STIG 建置視窗低和 STIG 內建視窗中強化元件的所有 STIG 設定。

- 視窗服務器 2022 STIG 版本 1 版本 1

  包含強化元件適用於類別 II 和 III (中和低) 弱點的所有支援 STIG 設定，以及：

  V-254293, V-254352, V-254353, V-254354,, 搶, 鼎, 蘋果, 熱潮, 電視台,, 保險, 302, 和 V-254374 V-254378 V-254381 V-254446 V-254465 V-254466 V-254467 V-254469 V-254474 V-254475 V-254500

- 視窗服務器 2019 時尚版本 2 發布 5

  包含強化元件適用於類別 II 和 III (中和低) 弱點的所有支援 STIG 設定，以及：

  V-205653、V-205654、V-205711、V-205713、V-205724、V-205725、V-205757、V-205802、V-205804 以及 V-205919

- 視窗服務器 2016 時尚版本 2 發布 5

包含強化元件適用於類別 II 和 III (中和低) 弱點的所有支援 STIG 設定，以及：

V-224874、V-224932、V-224933、V-224934、V-224954、V-224958、V-224961、V-225025、V-225044 以及 V-225079

- 視窗服務器 2012 R2 MS 風格版本 3 版本 5

  包含強化元件適用於類別 II 和 III (中和低) 弱點的所有支援 STIG 設定，以及：

  V-225556、V-225552、V-225547、V-225507、V-225505、V-225498、V-225497、V-225496、V-225493 以及 V-225274

- Microsoft. NET 框架 4.0 版本 2 版本 2

  包含強化元件適用於 Microsoft .NET 架構的第 II 和第 III 類 (中和低) 弱點的所有支援 STIG 設定。沒有其他 STIG 設定適用於類別 I 弱點。

- 視窗防火牆樣式版本 2 版本 1

  包含強化元件適用於類別 II 和 III (中和低) 弱點的所有支援 STIG 設定，以及：

  V-241992、V-241997 和 V-242002

- 互聯網瀏覽器 11 STIG 版本 2 發布 3

  包含強化元件適用的所有支援 STIG 設定，這些設定適用於 IE 11 的類別 II 和 III (中和低) 弱點。沒有其他 STIG 設定適用於類別 I 弱點。

- Microsoft 邊緣 STIG 版本 1 版本 6 (僅限視窗伺服器 2022)

  包含強化元件適用於類別 II 和 III (中和低) 弱點的所有支援 STIG 設定，以及：

  V-235758 和 V-235759

- 後衛 STIG 版本 2 版本 4 (僅限視窗服務器 2022)

  包含強化元件適用於類別 II 和 III (中和低) 弱點的所有支援 STIG 設定，以及：

  V-213426、V-213452 和 V-213453

視窗版本歷史記錄

本節會記錄每季 STIG 更新的 Windows 強化元件版本歷史記錄。若要查看季度的變更和已發佈版本，請選擇標題以展開資訊。

2024 年第一季度變更-2024 年 2 月 6 日 (沒有變更)：

對於 2024 年第一季度發行的 Windows 元件 STIG 沒有任何變更。

2023 年第四季度變動-二零二三年四月十二日 (沒有變動)：

對於 2023 年第四季度發行的 Windows 元件 STIG 沒有任何變更。

2023 年第三季變更-2023 年 4 月 10 日 (沒有變動)：

對於 2023 年第三季發行的視窗元件 STIG 沒有任何變更。

2023 年第二季度變動-二零二三年三月五日 (沒有變動)：

對於 2023 年第二季度發行的 Windows 元件 STIG 沒有任何變更。

2023 年第一季度變動-2023 年 3 月 27 日 (沒有變動)：

對於 2023 年第一季度發行的 Windows 元件 STIG 沒有任何變更。

二零二二年第四季變更-二零二三年一月二日：

針對 2022 年第四季發行版本的更新 STIG 版本和套用的 STIG，如下所示：

柱狀構建視窗-低版本 2022.4.x

- Windows Server 2022 STIG 版本 1 第 1 版
- Windows Server 2019 STIG 版本 2 第 5 版
- Windows Server 2016 STIG 版本 2 第 5 版
- Windows Server 2012 R2 MS STIG 版本 3 第 5 版
- Microsoft .NET Framework 4.0 STIG 版本 2 第 2 版
- Windows Firewall STIG 版本 2 第 1 版
- Internet Explorer 11 STIG 版本 2 第 3 版
- Microsoft 邊緣 STIG 版本 1 版本 6 (僅限視窗伺服器 2022)

柱狀構建視窗中型版本 2022.4.x

- Windows Server 2022 STIG 版本 1 第 1 版

- Windows Server 2019 STIG 版本 2 第 5 版
- Windows Server 2016 STIG 版本 2 第 5 版
- Windows Server 2012 R2 MS STIG 版本 3 第 5 版
- Microsoft .NET Framework 4.0 STIG 版本 2 第 2 版
- Windows Firewall STIG 版本 2 第 1 版
- Internet Explorer 11 STIG 版本 2 第 3 版
- Microsoft 邊緣 STIG 版本 1 版本 6 (僅限視窗伺服器 2022)
- 後衛 STIG 版本 2 版本 4 (僅限視窗服務器 2022)

柱狀構建視窗-高版本 2022.4.x

- Windows Server 2022 STIG 版本 1 第 1 版
- Windows Server 2019 STIG 版本 2 第 5 版
- Windows Server 2016 STIG 版本 2 第 5 版
- Windows Server 2012 R2 MS STIG 版本 3 第 5 版
- Microsoft .NET Framework 4.0 STIG 版本 2 第 2 版
- Windows Firewall STIG 版本 2 第 1 版
- Internet Explorer 11 STIG 版本 2 第 3 版
- Microsoft 邊緣 STIG 版本 1 版本 6 (僅限視窗伺服器 2022)
- 後衛 STIG 版本 2 版本 4 (僅限視窗服務器 2022)

2022 年第三季度變動-二零二二年九月三十日 (不變) :

對於 2022 年第三季發行的 Windows 元件 STIG 沒有任何變更。

二零二年第二季度變動：

為 2022 年第二季發行版本更新了 STIG 版本和應用的 STIG。

柱狀構建視窗低版本 1.5.x

- 視窗服務器 2019 時尚版本 2 版本 4
- 視窗服務器 2016 時尚版本 2 版本 4

- 視窗服務器 2012 R2 MS 風格版本 3 版本 3
- Microsoft. NET 框架 4.0 版本 2 版本 1
- Windows Firewall STIG 版本 2 第 1 版
- 互聯網瀏覽器 11 STIG 版本 1 版本 19

柱狀構建視窗中版本 1.5.x

- 視窗服務器 2019 時尚版本 2 版本 4
- 視窗服務器 2016 時尚版本 2 版本 4
- 視窗服務器 2012 R2 MS 風格版本 3 版本 3
- Microsoft. NET 框架 4.0 版本 2 版本 1
- Windows Firewall STIG 版本 2 第 1 版
- 互聯網瀏覽器 11 STIG 版本 1 版本 19

柱狀構建視窗高版本 1.5.x

- 視窗服務器 2019 時尚版本 2 版本 4
- 視窗服務器 2016 時尚版本 2 版本 4
- 視窗服務器 2012 R2 MS 風格版本 3 版本 3
- Microsoft. NET 框架 4.0 版本 2 版本 1
- Windows Firewall STIG 版本 2 第 1 版
- 互聯網瀏覽器 11 STIG 版本 1 版本 19

2022 年第一季度變動 (無變更)：

對於 2022 年第一季發布的 Windows 組件 STIG 沒有任何更改。

二零二一年第四季度變更：

針對 2021 年第四季發行的更新 STIG 版本和應用了 STIG。

柱狀構建視窗低版本 1.5.x

- 視窗服務器 2019 時尚版本 2 發布 3

- 視窗服務器 2016 時尚版本 2 發布 3
- 視窗服務器 2012 R2 MS 風格版本 3 版本 3
- Microsoft. NET 框架 4.0 版本 2 版本 1
- Windows Firewall STIG 版本 2 第 1 版
- 互聯網瀏覽器 11 STIG 版本 1 版本 19

柱狀構建視窗中版本 1.5.x

- 視窗服務器 2019 時尚版本 2 發布 3
- 視窗服務器 2016 時尚版本 2 發布 3
- 視窗服務器 2012 R2 MS 風格版本 3 版本 3
- Microsoft. NET 框架 4.0 版本 2 版本 1
- Windows Firewall STIG 版本 2 第 1 版
- 互聯網瀏覽器 11 STIG 版本 1 版本 19

柱狀構建視窗高版本 1.5.x

- 視窗服務器 2019 時尚版本 2 發布 3
- 視窗服務器 2016 時尚版本 2 發布 3
- 視窗服務器 2012 R2 MS 風格版本 3 版本 3
- Microsoft. NET 框架 4.0 版本 2 版本 1
- Windows Firewall STIG 版本 2 第 1 版
- 互聯網瀏覽器 11 STIG 版本 1 版本 19

二零二一年第三季變更-九月三十日：

針對 2021 年第三季發行的更新 STIG 版本和應用了 STIG。

柱狀構建視窗低版本 1.4.x

- 視窗服務器 2019 STIG 版本 2 發布 2
- 視窗服務器 2016 時間版本 2 發布 2
- 視窗服務器 2012 R2 MS 風格版本 3 版本 2

- Microsoft. NET 框架 4.0 版本 2 版本 1
- 視窗防火牆樣式版本 1 版本 7
- 互聯網瀏覽器 11 STIG 版本 1 版本 19

柱狀構建視窗中版本 1.4.x

- 視窗服務器 2019 時間版本 2 發布 2
- 視窗服務器 2016 時間版本 2 發布 2
- 視窗服務器 2012 R2 MS 風格版本 3 版本 2
- Microsoft. NET 框架 4.0 版本 2 版本 1
- 視窗防火牆樣式版本 1 版本 7
- 互聯網瀏覽器 11 STIG 版本 1 版本 19

柱狀構建視窗高版本 1.4.x

- 視窗服務器 2019 時間版本 2 發布 2
- 視窗服務器 2016 時間版本 2 發布 2
- 視窗服務器 2012 R2 MS 風格版本 3 版本 2
- Microsoft. NET 框架 4.0 版本 2 版本 1
- 視窗防火牆樣式版本 1 版本 7
- 互聯網瀏覽器 11 STIG 版本 1 版本 19

強化元件

本節包含 Linux STIG 強化元件的相關資訊，以及版本歷程記錄記錄。如果 Linux 發行版本沒有自己的 STIG 設定，強化元件會套用 RHEL 設定。強化元件會將支援的 STIG 設定套用至以 Linux 發行版為基礎的基礎結構，如下所示：

紅帽企業版 (RHEL) 7 樣式設定

- RHEL 7
- CentOS 7
- Amazon Linux 2 (AL2)

風 8 風格設置

- RHEL 8
- CentOS 8
- Amazon 亞馬遜

標籤構建-亞歷克斯-低版本 2024.1.x

下列清單包含強化元件套用至基礎結構的 STIG 設定。如果支援的設定不適用於您的基礎結構，強化元件會略過該設定，然後繼續進行。例如，某些 STIG 設定可能不適用於獨立伺服器。組織特定的原則也會影響強化元件套用的設定，例如系統管理員必須檢閱文件設定。

如需完整清單，請參閱 STIG 文件庫。如需有關如何檢視完整清單的詳細資訊，請參閱 STIG 檢視工具。

RHEL 7 風格版本 3 版本 14

- 瑞尔 7/CentOS 7

  V-204452、V-204576 和 V-204605
- AL2

  V-204452、V-204576 和 V-204605

RHEL 8 風格版本 1 版本 13

- RHEL 8 /CentOS 8

  V-230241, V-244527, V-230269, V-230270,, 搶, 壓制, 蘋果, 早餐, 卷, 鑼, 蒸氣, 北, V-230486,,,, V-230496, V-230497, V-230498, V-230499, 和 V 型 230281 V-230285 V-230253 V-230346 V-230381 V-230395 V-230468 V-230469 V-230491 V-230485 V-230494 V-230495

Ubuntu 18.04 風格版本 2 版本發布 13

V-219172, V-219173, V-219174, V-219175,, 搶, 鼎, 蘋果, 熱潮, 電梯, 鑼灣, 北歐, 和基督教 V-219210 V-219164 V-219165 V-219178 V-219180 V-219301 V-219163 V-219332 V-219327 V-219333

Ubuntu 20.04 風格版本 1 版本發布 11

V-238202, V-238234, V-238235, V-238237,, 搶, 鼎, 蘋果, 熱潮, 電梯, 鑼灣, 北歐, 和基督教 V-238323 V-238373 V-238221 V-238222 V-238223 V-238224 V-238226 V-238362 V-238357 V-238308

標籤構建-亞麻-中型版本 2024.1.x

下列清單包含強化元件套用至基礎結構的 STIG 設定。如果支援的設定不適用於您的基礎結構，強化元件會略過該設定，然後繼續進行。例如，某些 STIG 設定可能不適用於獨立伺服器。組織特定的原則也會影響強化元件套用的設定，例如系統管理員必須檢閱文件設定。

如需完整清單，請參閱 STIG 文件庫。如需有關如何檢視完整清單的詳細資訊，請參閱 STIG 檢視工具。

> **Note**
>
> STIG 建置-Linux 中強化元件除了專門針對類別 II 弱點列出的 STIG 設定外，還包括 AWS TOE 適用於 STIG 建置-Linux 低強化元件的所有 STIG 設定。

RHEL 7 風格版本 3 版本 14

包含強化元件適用於此 Linux 發行版本的「類別 III (低)」弱點的所有支援 STIG 設定，以及：

- 瑞尔 7/CentOS 7

  V-204585, V-204490, V-204491, V-, V-204418, V-204426, V-204431, V-204457, V-204466, V-204417, V-204434, V-204435, V-, V-204600, V-204602, V-204602, V-233307, V-255925, V-204578, V-204595, V-204437, V-204503, V-204507, V-204508, V 204517,V-204521, V-204524, V-204531, V-204536, V-204537, V-204538, V-204539, V-204540, V 型, V-204552, V-204553, V-204554, V-204555, V-204556, V-204557, V-204558, V-204559, V-204562, V-204563, V 型,, V-204610, V-204611, V-204612, V-204613, V-204614, V-204616, V-204617, V-204617, V.,V-204631、V-204633 和 V-256970

- 其二：

  V-204585, V-204490, V-204491, V-, V-204418, V-204426, V-204431, V-204457, V-204466, V-204417, V-204434, V-204435, V-, V-204600, V-204602, V-204602, V-233307, V-255925, V-204578, V-204595, V-204437, V-204503, V-204507, V-204508, V 204517,V-204521, V-204524, V-204531, V-204536, V-204537, V-204538, V-204539, V-204540, V 型, V-204552, V-204553, V-204554, V-204555, V-204556, V-204557, V-204558, V-204559, V-204562, V-204563, V 型,, V-204610, V-204611, V-204612, V-204613, V-204614, V-204616, V-204617, V-204617, V.,V-204631、V-204633 和 V-256970

RHEL 8 風格版本 1 版本 13

包含強化元件適用於此 Linux 發行版本的「類別 III (低)」弱點的所有支援 STIG 設定，以及：

- RHEL 8 /CentOS 8

  伏特 -230257, V-230258, V-230259, 伏特 -230248, 伏特 -230249, 伏特 -230250, V-230245, V-230228, V-230397, V 30233, V-230324, V-230324, 電影 -230378, V-230383, V-230314, 電影 -230314, 電影 -244523, V-230267, V 30532, V-230535, V-230536, V-230537, V-230539, V-230535, V-230542, V-230542, V-230543, V 型, V 型,V-250317, V-251718, V-230237, V-230356, V-230357, 伏特 -230359, V-230359, V-230361, V-230361, V V-244533, V-251713, V-251717, V-251714, V-251716, 伏特 -230332, 伏特 -230334, 伏特 -230335, V -230340, V, V-230343, V-230345, V-230240, V-230282, V-250316, V-230277, V-230277, V-230278, V-230394, V-230396, V 398, 伏特 -230402, VV-230405, V-230406, V-230407, V-230408, V-230409, V-230411, V-230412, V-230413, V-230418, V 26, V 型, 230427, V-230429, V-230429, V-230431, V-230432, V-230433, V-230434, V-230435, V, V-230448, V-230449, V-230455, V-230462, V-230463, V-230464, V-230465, V-230466, V V-244542, V-230503, V-二三四四, V 型V-230296, V-230330, V-230382, V-230526,,, 向前, 壓制, 蘋果, 四千五百五十, 四季, 前夕,,,, V-230488, V-230489, V-230559, V 型 230561, V 型 -237640, 和 V 型 256974 V-230527 V-230555 V-230556 V-244526 V-244528 V-237642 V-237643 V-251711 V-230238 V-230239 V-230273 V-230275 V-230478

Ubuntu 18.04 風格版本 2 版本發布 13

包含強化元件適用於此 Linux 發行版本的「類別 III (低)」弱點的所有支援 STIG 設定，以及：

V -219188, V -219190, V -219198, V -219199, 伏特 -219200, V -219201, V -219202, 二 19203, 二 19204, V -219205, V -219206, V -219207, V -219342, V -219189, V -219192, V -219194, V -219315, 伏特 -219195, 伏特 -219196, 伏特 -219197, 伏特, 二 19214, V -219215, V, V -219223, V-二一九二二七, V -219228, 伏特 -219229, 伏特 -219230, 伏特 -219231, 二一三三三, 二一三三四, 第二九三三四, V 19244,V-219250, V-219254, V-219257, V-219263,,, 湧, 球, 蘋果, 壺, 節制, 節制, 包括, 包括, 領導, 玻璃, 伏特, 伏特, V 型, 219287, V 型 219297, V-219298, V 型, 電信 -233780, 電信 -255906, 電子 -219338, 電子 -219344, 電視 -219184, V 型, 電子 -219156, 電子 -219156, 電子 -219160, 電子 -219306, 電子 -219149, V 型 19335 V-219264 V-219265 V-219266 V-219267 V-219268 V-219269 V-219270 V-219271 V-219272 V-219273 V-219274 V-219275 V-219276 V-219277 V-219279 V-219281

Ubuntu 20.04 風格版本 1 版本發布 11

包含強化元件適用於此 Linux 發行版本的「類別 III (低)」弱點的所有支援 STIG 設定，以及：

V-238205, V-238207, V-238329, V-238339, V-238340, V-238344, V-238345, V-238347, V-238347, V-238349, V-238349, V-238350, V V-23825, V-238330, V-23833, V-238369, V-238341, V-238342, V-238343, V-238343, V-238353, V-238228, V V-238244, V-238245, V-238246, V-238248, V-238249, V-238250, V-238251, V-238252, V-238253, V-238254, V 238255, V 277,V-238278, V-238279, V-238280, V-238281,, 保證, 湧, 球, 杯,, 節制, 蒸氣, 蒸汽, 包, 雞, 伏特, 玻璃, 伏特, 伏特, V 型 238302, V 型 238309, V 型 238309, V 型 238310, V 型 238310, V 型 238315, V 型 238315, V 型 238316, V 型 238316, V 型 238317, V 型 238319 電視 -251505, 電視 -238360, 電視機, 電視 238213, 電視機, 電視 238220, V 型, 電視 -238355, 電視, 電視 238303, 電視, 電視 238358, 電視 238356, 電視 238359, 電視 238370, 和電視 V-238282 V-238283 V-238284 V-238285 V-238286 V-238287 V-238288 V-238289 V-238290 V-238291 V-238292 V-238293 V-238294 V-238295 V-238297 V-238300

標籤構建-亞麻-高版本 2024.1.x

下列清單包含強化元件套用至基礎結構的 STIG 設定。如果支援的設定不適用於您的基礎結構，強化元件會略過該設定，然後繼續進行。例如，某些 STIG 設定可能不適用於獨立伺服器。組織特定的原則也會影響強化元件套用的設定，例如系統管理員必須檢閱文件設定。

如需完整清單，請參閱 STIG 文件庫。如需有關如何檢視完整清單的詳細資訊，請參閱 STIG 檢視工具。

> **Note**
>
> STIG 建置-Linux 高強化元件除了列出適用於第 I 類弱點的 STIG 設定之外，還包括 AWS TOE 適用於 STIG 建置-Linux 低和 STIG 建置-Linux 中強化元件的所有 STIG 設定。

RHEL 7 風格版本 3 版本 14

包含強化元件適用於此 Linux 發行版本的「類別 II」和「III」(中和低) 弱點的所有支援 STIG 設定，以及：

- 瑞尔 7/CentOS 7

  V-204425, V-204594, V-204455, V-204424,, 渴望, 鼎, 朗, 熱量, 電梯, 和 CHINGS V-204442 V-204443 V-204447 V-204448 V-204502 V-204620 V-204621

- 其二：

  V-204425, V-204594, V-204455, V-204424,, 渴望, 鼎, 朗, 熱量, 電梯, 和 CHINGS V-204442 V-204443 V-204447 V-204448 V-204502 V-204620 V-204621

RHEL 8 風格版本 1 版本 13

包含強化元件適用於此 Linux 發行版本的「類別 II」和「III」(中和低) 弱點的所有支援 STIG 設定，以及：

- RHEL 8 /CentOS 8

  V-230265, V-230529, V-230531, V-230264, V-230487, 愿意, 渴望, 和輕 V-230492 V-230533 V-230558

Ubuntu 18.04 風格版本 2 版本發布 13

包含強化元件適用於此 Linux 發行版本的「類別 II」和「III」(中和低) 弱點的所有支援 STIG 設定，以及：

V-219157, V-219158, V-219177, V-219212 V-219308, 愿意, 渴望, 和 V-251507 V-219314 V-219316

Ubuntu 20.04 風格版本 1 版本發布 11

包含強化元件適用於此 Linux 發行版本的「類別 II」和「III」(中和低) 弱點的所有支援 STIG 設定，以及：

V-238218, V-238219, V-238201, V-238326, V-238327, V-238380 和 fan V-251504

STIG 版本歷史記錄日誌

本節會記錄 Linux 元件版本歷史記錄。若要查看季度的變更和已發佈版本，請選擇標題以展開資訊。

2024 年第一季度變更-二零二四年六月二日：

針對 2024 年第一季發行版本更新 STIG 版本和套用的 STIG，如下所示：

標籤構建-亞歷克斯-低版本 2024.1.x

- RHEL 7 風格版本 3 版本 14
- RHEL 8 風格版本 1 版本 13
- Ubuntu 18.04 風格版本 2 版本發布 13
- Ubuntu 20.04 風格版本 1 版本發布 11

標籤構建-亞麻-中型版本 2024.1.x

- RHEL 7 風格版本 3 版本 14
- RHEL 8 風格版本 1 版本 13
- Ubuntu 18.04 風格版本 2 版本發布 13
- Ubuntu 20.04 風格版本 1 版本發布 11

標籤構建-亞麻-高版本 2024.1.x

- RHEL 7 風格版本 3 版本 14
- RHEL 8 風格版本 1 版本 13
- Ubuntu 18.04 風格版本 2 版本發布 13
- Ubuntu 20.04 風格版本 1 版本發布 11

二零二三年第四季變更-二零二三年七月十二日：

針對 2023 年第四季發行的 STIG 版本和套用的 STIG 更新如下：

標籤構建-亞麻-低版本 2023.4.x

- RHEL 7 風格版本 3 版本 13
- RHEL 8 風格版本 1 版本 12
- Ubuntu 18.04 風格版本 2 版本發布 12
- Ubuntu 20.04 風格版本 1 版本發布 10

標籤構建-亞麻-中型版本 2023.4.x

- RHEL 7 風格版本 3 版本 13
- RHEL 8 風格版本 1 版本 12
- Ubuntu 18.04 風格版本 2 版本發布 12
- Ubuntu 20.04 風格版本 1 版本發布 10

標籤構建-亞麻-高版本 2023.4.x

- RHEL 7 風格版本 3 版本 13

- RHEL 8 風格版本 1 版本 12
- Ubuntu 18.04 風格版本 2 版本發布 12
- Ubuntu 20.04 風格版本 1 版本發布 10

二零二三年第三季變更-二零二三年四月十日：

針對 2023 年第三季發行版本的更新 STIG 版本和套用的 STIG，如下所示：

標籤構建-亞麻-低版本 2023.3.x

- RHEL 7 風格版本 3 版本 12
- RHEL 8 風格版本 1 版本 11
- Ubuntu 18.04 風格版本 2 版本發布 11
- Ubuntu 20.04 風格版本 1 發布版本 9

標籤構建-亞麻-中型版本 2023.3.x

- RHEL 7 風格版本 3 版本 12
- RHEL 8 風格版本 1 版本 11
- Ubuntu 18.04 風格版本 2 版本發布 11
- Ubuntu 20.04 風格版本 1 發布版本 9

標籤構建-亞麻-高版本 2023.3.x

- RHEL 7 風格版本 3 版本 12
- RHEL 8 風格版本 1 版本 11
- Ubuntu 18.04 風格版本 2 版本發布 11
- Ubuntu 20.04 風格版本 1 發布版本 9

二零二三年第二季度變動-二零二三年五月三日：

針對 2023 年第二季發行的 STIG 版本和套用的 STIG 更新如下：

標籤構建-亞麻-低版本 2023.2.x

- RHEL 7 風格版本 3 版本 11

- RHEL 8 風格版本 1 發布 10
- Ubuntu 18.04 風格版本 2 版本發布 11
- Ubuntu 20.04 風格版本 1 版本發布 8

標籤構建-亞麻-中型版本 2023.2.x

- RHEL 7 風格版本 3 版本 11
- RHEL 8 風格版本 1 發布 10
- Ubuntu 18.04 風格版本 2 版本發布 11
- Ubuntu 20.04 風格版本 1 版本發布 8

標籤構建-亞麻-高版本 2023.2.x

- RHEL 7 風格版本 3 版本 11
- RHEL 8 風格版本 1 發布 10
- Ubuntu 18.04 風格版本 2 版本發布 11
- Ubuntu 20.04 風格版本 1 版本發布 8

2023 年第一季度變更-二零二三年三月二十日：

針對 2023 年第一季發行的 STIG 版本和套用的 STIG 更新，如下所示：

標籤構建-亞麻-低版本 2023.1.x

- RHEL 7 風格版本 3 發布 10
- RHEL 8 風格版本 1 版本 9
- Ubuntu 18.04 風格版本 2 版本發布 10
- Ubuntu 20.04 風格版本 1 版本發布 7

標籤構建-亞麻-中型版本 2023.1.x

- RHEL 7 風格版本 3 發布 10
- RHEL 8 風格版本 1 版本 9
- Ubuntu 18.04 風格版本 2 版本發布 10
- Ubuntu 20.04 風格版本 1 版本發布 7

標籤構建-亞麻-高版本 2023.1.x

- RHEL 7 風格版本 3 發布 10
- RHEL 8 風格版本 1 版本 9
- Ubuntu 18.04 風格版本 2 版本發布 10
- Ubuntu 20.04 風格版本 1 版本發布 7

二零二二年第四季變更-二零二三年一月二日：

針對 2022 年第四季發布的 STIG 版本和應用的 STIG 更新如下：

標籤構建-亞麻-低版本 2022.4.x

- RHEL 7 風格版本 3 版本 9
- RHEL 8 風格版本 1 版本 8
- Ubuntu 18.04 風格版本 2 版本 9
- Ubuntu 20.04 風格版本 1 發布版本 6

標籤構建-亞麻-中型版本 2022.4.x

- RHEL 7 風格版本 3 版本 9
- RHEL 8 風格版本 1 版本 8
- Ubuntu 18.04 風格版本 2 版本 9
- Ubuntu 20.04 風格版本 1 發布版本 6

標籤構建-亞麻-高版本 2022.4.x

- RHEL 7 風格版本 3 版本 9
- RHEL 8 風格版本 1 版本 8
- Ubuntu 18.04 風格版本 2 版本 9
- Ubuntu 20.04 風格版本 1 發布版本 6

2022 年第三季度變動-二零二二年九月三十日 (不變)：

對於 2022 年第三季發行的 Linux 元件 STIG，並沒有任何變更。

二零二年第二季度變動：

針對 2022 年第二季發行版推出 Ubuntu 支援、更新的 STIG 版本，以及套用 STIG，如下所示：

標籤構建-亞麻-低版本 2022.2.x

- RHEL 7 風格版本 3 版本 7
- RHEL 8 風格版本 1 版本 6
- Ubuntu 18.04 STIG 版本 2 版本 6（新）
- Ubuntu 20.04 STIG 版本 1 版本 4（新）

標籤構建-亞麻-中型版本 2022.2.x

- RHEL 7 風格版本 3 版本 7
- RHEL 8 風格版本 1 版本 6
- Ubuntu 18.04 STIG 版本 2 版本 6（新）
- Ubuntu 20.04 STIG 版本 1 版本 4（新）

標籤構建-亞麻-高版本 2022.2.x

- RHEL 7 風格版本 3 版本 7
- RHEL 8 風格版本 1 版本 6
- Ubuntu 18.04 STIG 版本 2 版本 6（新）
- Ubuntu 20.04 STIG 版本 1 版本 4（新）

二零二二年第一季度變動：

重構以包括對容器的更好支持。結合先前的 AL2 指令碼與 RHEL 7。針對 2022 年第一季發布的 STIG 版本和應用的 STIG 更新如下：

標籤構建-亞麻-低版本 3.6.x

- RHEL 7 風格版本 3 版本 6
- RHEL 8 風格版本 1 版本 5

標籤構建-亞麻-中型版本 3.6.x

- RHEL 7 風格版本 3 版本 6
- RHEL 8 風格版本 1 版本 5

標籤構建-亞麻-高版本 3.6.x

- RHEL 7 風格版本 3 版本 6
- RHEL 8 風格版本 1 版本 5

二零二一年第四季度變更：

更新了 STIG 版本，並在 2021 年第四季度發布中應用了 STIGS，如下所示：

標籤構建-亞麻低版本 3.5.x

- RHEL 7 風格版本 3 發布 5
- RHEL 8 風格版本 1 版本 4

標籤構建-亞麻-中等版本 3.5.x

- RHEL 7 風格版本 3 發布 5
- RHEL 8 風格版本 1 版本 4

標籤構建-亞麻-高版本 3.5.x

- RHEL 7 風格版本 3 發布 5
- RHEL 8 風格版本 1 版本 4

二零二一年第三季變更-九月三十日：

更新了 STIG 版本，並在 2021 年第三季度發布中應用了 STIGS，如下所示：

標籤構建-亞麻-低版本 3.4.x

- RHEL 7 風格版本 3 版本 4
- RHEL 8 風格版本 1 版本 3

標籤構建-亞麻-中型版本 3.4.x

- RHEL 7 風格版本 3 版本 4
- RHEL 8 風格版本 1 版本 3

標籤構建-亞麻-高版本 3.4.x

- RHEL 7 風格版本 3 版本 4
- RHEL 8 風格版本 1 版本 3

SCAP 合規性驗證程式元件

安全性內容自動化通訊協定 (SCAP) 是一組標準，IT 專業人員可用來識別應用程式安全性弱點以達到合規性。SCAP 合規性檢查器（SCC）是 SCAP 驗證的掃描工具，由海軍信息戰中心（NIWC）大西洋發布。如需詳細資訊，請參閱 NIWC 大西洋網站上的安全性內容自動化通訊協定 (SCAP) 符合性檢查程式 (SCC)。

AWS TOE `scap-compliance-checker-windows`和`scap-compliance-checker-linux`元件會在管線建置和測試執行個體上下載並安裝 SCC 掃描器。當掃描器執行時，它會使用 DISA SCAP 基準測試執行經過驗證的組態掃描，並提供包含下列資訊的報告。 AWS TOE 還將信息寫入您的應用程序日誌。

- 套用至執行個體的 STIG 設定。
- 執行個體的整體相容分數。

我們建議您執行 SCAP 驗證作為建置程序的最後一個步驟，以確保您報告正確的合規性驗證結果。

> **Note**
>
> 您可以使用其中一個 STIG 檢視工具來檢閱報告。這些工具可通過 DoD 網絡交易所在線獲得。

下列各節說明 SCAP 驗證元件所包含的基準測試。

scap-compliance-checker-linux 版本

`scap-compliance-checker-linux`元件會在 Image Builder 管線的建置和測試執行個體上執行。AWS TOE 記錄報告和 SCC 應用程式產生的分數。

元件會執行下列工作流程步驟：

1. 下載並安裝 SCC 應用程式。
2. 匯入合規性基準。
3. 使用 SCC 應用程式執行驗證。
4. 將符合性報告和分數儲存在組建執行個體桌面上的本機。
5. 將符合性分數從本機報告記錄到 AWS TOE 應用程式記錄檔。

> **Note**
>
> AWS TOE 目前支援視窗伺服器 2012 年 R2、2016 年和 2019 年的 SCAP 合規性驗證。

適用於 Windows 的 SCAP 相容性檢查程式元件包含下列基準測試：

SCC 版本：5.4.2

2021 年第四季度基準：

- U_MS _ 框架 _4-0_V2R1_ 柱狀結構 _ DotNet 基準
- U_MS_I11_V2R1_S 標竿
- 管理系統視窗和 _ _ _ _ _ _ _ _ _ _ 基準
- 管理系統視窗防衛 _ AV_V2R2_STIG_ 架構 _ 基準
- 管理系統視窗伺服器 _ 視窗伺服器 _2016 最佳效能標竿
- 管理系統視窗伺服器 _ 基準測試
- 管理系統視窗 _ 防火牆 _V2R1_STIGP _ 基準
- U_ 官方網站 _18-04 _ 柱狀結構 _ 基準
- 瑞海爾 7_V3R5_ 柱狀結構 _ 基準
- 瑞海爾 8_V1R3_ 標杆 _ 標杆

scap-compliance-checker-linux 版本

`scap-compliance-checker-linux`元件會在 Image Builder 管線的建置和測試執行個體上執行。AWS TOE 記錄報告和 SCC 應用程式產生的分數。

元件會執行下列工作流程步驟：

1. 下載並安裝 SCC 應用程式。
2. 匯入合規性基準。
3. 使用 SCC 應用程式執行驗證。
4. 在本機儲存符合性報告和分數，位於組建執行個體的下列位置：`/opt/scc/SCCResults`
5. 將符合性分數從本機報告記錄到 AWS TOE 應用程式記錄檔。

> **Note**
>
> AWS TOE 目前支援 RHEL 7/8 和 Ubuntu 18 的 SCAP 合規性驗證。SCC 應用程式目前支援 x86 架構進行驗證。

適用於 Linux 的 SCAP 相容性檢查程式元件包含下列基準測試：

SCC 版本：5.4.2

2021 年第四季度基準：

- U_ 官方網站 _18-04 _ 柱狀結構 _ 基準
- 瑞海爾 7_V3R5_ 柱狀結構 _ 基準
- 瑞海爾 8_V1R3_ 標杆 _ 標杆
- U_MS _ 框架 _4-0_V2R1_ 柱狀結構 _ DotNet 基準
- U_MS_I11_V2R1_S 標竿
- 管理系統視窗和 _ _ _ _ _ _ _ _ _ _ 基準
- 管理系統視窗防衛 _ AV_V2R2_STIG_ 架構 _ 基準
- 管理系統視窗伺服器 _ 視窗伺服器 _2016 最佳效能標竿
- 管理系統視窗伺服器 _ 基準測試
- 管理系統視窗 _ 防火牆 _V2R1_STIGP _ 基準

SCAP 版本歷史記錄

下表說明 SCAP 環境的重要變更，以及本文件所述的設定。

變更	描述	日期
已新增 SCAP 元件	引入下列 SCAP 元件： • 建立的 scap-compliance-checker-linux 版本：2021.04.0 (調整中心版本:5.4.2) • 建立的 scap-compliance-checker-linux 版本：2021.04.0 (調整中心版本:5.4.2)	2021 年 12 月 20 日

AWS TOE 指令參考

AWS TOE 是在中執行的元件管理應用程式 AWS CLI。

> **Note**
>
> 某些 AWS TOE 動作模組需要提升的權限才能在 Linux 伺服器上執行。若要使用提升的權限，請在命令語法前置詞sudo，或在登入時執行一次sudo su命令，然後再執行下列連結的命令。如需 AWS TOE 動作模組的詳細資訊，請參閱AWS TOE 元件管理員支援的動作模組。

run

使用命run令執行一或多個元件文件的 YAML 文件指令碼。

validate

執行命validate令以驗證一或多個元件文件的 YAML 文件語法。

要命令運行

此命令會依參數指定之組態檔案中包含的順序，或`--config`參數指定的元件文件清單，執行 YAML 元件文件指令碼。`--documents`

> **Note**
>
> 您必須指定下列其中一個參數，絕不能同時指定兩個參數：

-配置
-文件

語法

```
awstoe run [--config <file path>] [--cw-ignore-failures <?>]
      [--cw-log-group <?>] [--cw-log-region us-west-2] [--cw-log-stream <?>]
      [--document-s3-bucket-owner <owner>] [--documents <file path,file path,...>]
      [--execution-id <?>] [--log-directory <file path>]
      [--log-s3-bucket-name <name>] [--log-s3-bucket-owner <owner>]
      [--log-s3-key-prefix <?>] [--parameters name1=value1,name2=value2...]
      [--phases <phase name>] [--state-directory <directory path>] [--version <?>]
      [--help] [--trace]
```

參數和選項

參數

-配置 *./config-example.json*

簡短形式：-c *./config-example.json*

組態檔案 (條件式)。此參數包含 JSON 檔案的檔案位置，此檔案包含執行此命令之元件的組態設定。如果您在組態檔案中指定指run令設定，則不得指定--documents參數。如需輸入規劃的詳細資訊，請參閱配置 AWS TOE 運行命令的輸入。

有效地點包括：

- 本機檔案路徑 (*./config-example.json*)
- 一個 S3 URI (s3://*bucket/key*)

--cw-ignore-failures

簡短表格:N/A

忽略記錄檔中的記 CloudWatch 錄失敗。

--cw-log-group

簡短表格:N/A

CloudWatch 記錄檔的LogGroup名稱。

--cw-log-region

簡短表格:N/A

適用於 CloudWatch 日誌的 AWS 區域。

--cw-log-stream

簡短表格:N/A

CloudWatch 日誌的`LogStream`名稱，用於指示流式傳輸`console.log`文件的 AWS TOE 位置。

-文件-s3-桶所有者

簡短表格:N/A

S3 URI 文件的儲存貯體擁有者帳戶識別碼。

-#./doc-1.yaml##./doc-n.yaml

簡短形式：-d *./doc-1.yaml*，*./doc-n*

零組件文件 (條件式)。此參數包含要執行的 YAML 元件文件之檔案位置的逗號分隔清單。如果您使用`--documents`參數指定run命令的 YAML 文件，則不得指定`--config`參數。

有效地點包括：

- 本機檔案路徑 (*. /component-doc-example. ###)*.
- S3 URI (`s3://`*bucket/key*)。
- *Image Builder ###### ARN#ARN#AW###########-2#123456789012## #/2021.12.02/1##my-example-component*

> **Note**
>
> 清單中的項目之間沒有空格，只有逗號。

-執行 ID

簡短表格：-i

這是適用於執行當前run命令的唯一 ID。此 ID 包含在輸出檔和記錄檔名稱中，以唯一識別這些檔案，並將它們連結至目前的指令執行。如果省略此設定，則 AWS TOE 會產生 GUID。

-日誌目錄

簡短形式：-l

AWS TOE 儲存此命令執行中所有記錄檔的目的地目錄。根據預設，此目錄位於下列父目錄內：TOE_<DATETIME>_<EXECUTIONID>。如果未指定記錄目錄， AWS TOE 會使用目前的工作目錄 (.)。

-日誌 S3-桶名

簡短形式：-b

如果元件日誌存放在 Amazon S3 (建議使用)，請將元件應用程式日誌 AWS TOE 上傳到此參數中指定的 S3 儲存貯體。

-日誌 s3 桶所有者

簡短表格:N/A

如果元件日誌存放在 Amazon S3 (建議使用)，這是 AWS TOE 寫入日誌檔的儲存貯體的擁有者帳戶 ID。

-日誌 S3 鍵前綴

簡短形式：-k

如果元件日誌存放在 Amazon S3 (建議使用)，這是儲存貯體中日誌位置的 S3 物件 key prefix 置詞。

-#### 1 = # 1### 2 = # 2...

簡短表格:N/A

參數是在元件文件中定義的可變變數，其中包含呼叫應用程式可在執行時間提供的設定。

-階段

簡短形式：-p

以逗號分隔的清單，指定要從 YAML 元件文件執行的階段。如果元件文件包含其他階段，則這些階段將不會執行。

-狀態目錄

簡短形式：-s

儲存狀態追蹤檔案的檔案路徑。

--version

簡短形式：-v

指定元件應用程式版本。

選項

--help

簡短形式：-h

顯示使用元件管理應用程式選項的說明手冊。

-跟踪

簡短形式：-t

啟用主控台的詳細記錄功能。

要求驗證命令

當您執行這個命令時，它會驗證每個參數所指定的元件文件的 YAML 文件語法。`--documents`

語法

```
awstoe validate [--document-s3-bucket-owner <owner>]
      --documents <file path,file path,...> [--help] [--trace]
```

參數和選項

參數

-文件-s3-桶所有者

簡短表格:N/A

提供以 S3 URI 為基礎的文件的來源帳戶識別碼。

-#./doc-1.yaml##./doc-n.yaml

簡短形式：-d *./doc-1.yaml*，*./doc-n*

元件文件 (必要)。此參數包含要執行的 YAML 元件文件之檔案位置的逗號分隔清單。有效地點包括：

- 本機檔案路徑 (*. /component-doc-example. ##*爾)
- S3 URI () `s3://`*bucket/key*
- *# Image Builder #### ARN ## (ARN: AW: ######:####-2:12456789012: # #/my-example-component*

> **Note**
>
> 清單中的項目之間沒有空格，只有逗號。

選項

--help

簡短形式：-h

顯示使用元件管理應用程式選項的說明手冊。

-跟踪

簡短形式：-t

啟用主控台的詳細記錄功能。

管理 EC2 Image Builder 資源

資源是構成映像管線的建構區塊，以及這些管線產生的映像檔。本章涵蓋建立、維護和共用 Image Builder 資源，包括元件、配方和影像，以及基礎結構組態和發佈設定。

> **Note**
>
> 為了協助您管理 Image Builder 資源，您可以以標記形式將自己的中繼資料指派給每個資源。您可以使用標籤以不同的方式對 AWS 資源進行分類，例如按目的、擁有者或環境。這在您擁有許多相同類型的資源時很有用。您可以根據指派給資源的標籤，更容易識別特定資源。
> 如需有關使用「Image Builder」指令標記資源的詳細資訊 AWS CLI，請參閱本指南一標籤資源節。

目錄

使用 Image Builder 管理元件

Image Builder 使用 AWS 任務協調器和執行器 (AWS TOE) 元件管理應用程式來協調複雜的工作流程。建置和測試與 AWS TOE 應用程式搭配使用的元件是以 YAML 文件為基礎，這些文件會定義用來自訂或測試映像的指令碼。對於 AMI 映像檔，Image Builder 會在其 Amazon EC2 建置和測試執行個體上安裝 AWS TOE 元件和元件管理應用程式。對於容器映像檔，元件和 AWS TOE 元件管理應用程式會安裝在執行中的容器內。

Image Builder 用 AWS TOE 來執行所有執行個體活動。 AWS TOE 當您執行 Image Builder 命令或使用映像產生 Image Builder 主控台時，不需要額外的設定即可與之互動。

> **Note**
>
> 當 Amazon 管理的元件到達其支援壽命結束時，將不再維護該元件。在這種情況發生前大約四週，任何使用該組件的帳戶都會收到通知，並從他們的帳戶中收到受影響的配方的列表 AWS Health Dashboard。若要進一步瞭解 AWS Health，請參閱AWS Health 使用者指南。

建立新影像的工作流程階段

用於建立新 Image Builder 工作流程包括以下兩個不同的階段。

1. 建置階段 (快照前) — 在建置階段，您可以變更執行基本映像的 Amazon EC2 建置執行個體，以建立新映像的基準。例如，您的方案可以包含安裝應用程式或修改作業系統防火牆設定的元件。

   下列元件階段會在建置階段執行：

   - build
   - validate

   順利完成此階段之後，Image Builder 會建立快照或容器映像，供測試階段及其後使用。
2. 測試階段 (快照後) — 在測試階段，建立 AMI 的映像與容器映像之間會有一些差異。對於 AMI 工作流程，Image Builder 會從建置階段的最後一個步驟所建立的快照啟動 EC2 執行個體。測試會在新執行個體上執行，以驗證設定並確保執行個體如預期般運作。對於容器工作流程，測試會在用於建置的相同執行個體上執行。

   測試階段期間，會針對方案中包含的每個元件執行下列元件階段：

   - test

   此元件階段適用於「建置」和「測試」元件類型。順利完成此階段之後，Image Builder 可以從快照或容器映像建立並分發您的最終影像。

> **Note**
>
> 雖然 AWS TOE 可讓您在元件文件中定義許多階段，但 Image Builder 對其執行的階段以及執行它們的階段有嚴格的規則。若要讓組件在建置階段執行，組件文件必須至少定義下列其中一

個階段：build或validate。若要讓元件在測試階段執行，元件文件必須定義test階段，而不能定義其他階段。
由於 Image Builder 會獨立執行階段，因此元件文件中的參照鏈結無法跨越階段邊界。您無法將值從建置階段中執行的階段鏈結到測試階段中執行的階段。但是，您可以將輸入參數定義到所需的目標，並透過指令行傳遞值。如需有關在 Image Builder 方法中設定元件參數的詳細資訊，請參閱使用 EC2 Image Builder 管理 AWS TOE 元件參數。

為了協助您對構建或測試實例進行故障排除，請 AWS TOE 創建一個日誌文件夾，其中包含輸入文檔和日誌文件，以跟踪每次組件運行時發生的情況。如果您在管道組態中設定 Amazon S3 儲存貯體，則日誌也會寫入該處。如需 YAML 文件和記錄檔輸出的詳細資訊，請參閱在中使用零組件文件 AWS TOE。

Tip

當您有許多要追蹤的元件時，加標籤可協助您根據指定給該元件的標籤識別特定元件或版本。如需有關使用「Image Builder」指令標記資源的詳細資訊 AWS CLI，請參閱本指南一標籤資源節。

本節介紹如何使用 Image Builder 主控台或中的命令來列出、檢視、建立和匯入元件 AWS CLI。

目錄

建立 YAML 元件文件

若要建置元件，請提供 YAML 應用程式元件文件。這表示建立元件所需的階段和步驟。

本節中的範例會建立組建元件，以呼叫元件管理應用程式中 AWS TOE 的Update0S動作模組。該模塊更新操作系統。如需Update0S動作模組的詳細資訊，請參閱更新。如需有關 AWS TOE YAML 應用程式元件文件的階段、步驟和語法的詳細資訊，請參閱在 AWS TOE.

> **Note**
>
> Image Builder 會決定管線工作流程中的元件類型。此工作流程會對應於建置程序中的「建置」階段和「測試」階段。Image Builder 決定元件類型，如下所示：
>
> - 建置-這是預設的元件類型。任何未歸類為測試組件的內容都是構建組件。這種類型的組件在「構建」階段期間運行。如果此組建元件已定義test階段，則該階段會在「測試」階段執行。
> - 測試 — 若要符合測試元件的資格，元件文件必須只包含一個名為的階段test。對於與構建組件配置相關的測試，我們建議您不要使用獨立的測試組件。相反，請在關聯的組建元件中使用test階段。
>
> 如需 Image Builder 如何在建置程序中使用階段和階段來管理元件工作流程的詳細資訊，請參閱使用 Image Builder 管理元件。

若要為範例應用程式建立 YAML 應用程式元件文件，請依照與映像作業系統相符的索引標籤上的步驟執行。

Linux

建立 YAML 元件檔案

使用檔案編輯工具建立名為的檔案*update-linux-os.yaml*。包括以下內容：

```
# Copyright 2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: MIT-0
#
# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
# without restriction, including without limitation the rights to use, copy, modify,
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
# permit persons to whom the Software is furnished to do so.
#
```

```
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
name: update-linux-os
description: Updates Linux with the latest security updates.
schemaVersion: 1
phases:
  - name: build
    steps:
    - name: UpdateOS
      action: UpdateOS
# Document End
```

> **Tip**
>
> 使用類似這個線上 YAML 驗證程式的工具，或在程式碼環境中使用 YAML lint 延伸模組來確認您的 YAML 格式正確。

Windows

建立 YAML 元件檔案

使用檔案編輯工具建立名為的檔案*update-windows-os.yaml*。包括以下內容：

```
# Copyright 2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: MIT-0
#
# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
# without restriction, including without limitation the rights to use, copy, modify,
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
# permit persons to whom the Software is furnished to do so.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
```

```
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
name: update-windows-os
description: Updates Windows with the latest security updates.
schemaVersion: 1.0
phases:
  - name: build
    steps:
      - name: UpdateOS
        action: UpdateOS
# Document End
```

> **Tip**
>
> 使用類似這個線上 YAML 驗證程式的工具，或在程式碼環境中使用 YAML lint 延伸模組來確認您的 YAML 格式正確。

使用 EC2 Image Builder 管理 AWS TOE 元件參數

您可以直接從 EC2 Image Builder 主控台或使用 AWS CLI 命令或其中一個映像產生器 SDK 來管理元 AWS TOE 件，包括建立和設定元件參數。在本節中，我們將介紹在元件中建立和使用參數，以及透過 Image Builder 主控台和 AWS CLI 指令設定元件參數。

> **Important**
>
> 元件參數是純文字值，並已登入 AWS CloudTrail。我們建議您使用 AWS Secrets Manager 或 AWS Systems Manager 參數存放區來儲存您的秘密。如需有關 Secrets Manager 的詳細資訊，請參閱什麼是 Secrets Manager？ 在《AWS Secrets Manager 使用者指南》中。若要取得有關 AWS Systems Manager 參數存放區的更多資訊，請AWS Systems Manager 參閱AWS Systems Manager 使用指南中的參數存放區

在 YAML 元件文件中使用參數

若要建置元件，請提供 YAML 應用程式元件文件。這表示建立元件所需的階段和步驟。參照元件的配方可以設定參數以在執行階段自訂值，預設值會在參數未設定為特定值時生效。

使用輸入參數建立元件文件

本節說明如何在 YAML 元件文件中定義和使用輸入參數。

若要建立使用參數並在 Image Builder 組建或測試執行個體中執行命令的 YAML 應用程式元件文件，請遵循符合映像作業系統的步驟：

Linux

建立 YAML 元件文件

使用檔案編輯工具建立名為的檔案*hello-world-test.yaml*。包括以下內容：

```
# Document Start
#
name: "HelloWorldTestingDocument-Linux"
description: "Hello world document to demonstrate parameters."
schemaVersion: 1.0
parameters:
  - MyInputParameter:
      type: string
      default: "It's me!"
      description: This is an input parameter.
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Build phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: validate
    steps:
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Validate phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: test
    steps:
```

```
      - name: HelloWorldStep
        action: ExecuteBash
        inputs:
          commands:
            - echo "Hello World! Test phase. My input parameter value is
 {{ MyInputParameter }}"
# Document End
```

> **Tip**
>
> 使用類似這個線上 YAML 驗證程式的工具，或在程式碼環境中使用 YAML lint 延伸模組來確認您的 YAML 格式正確。

Windows

建立 YAML 元件文件

使用檔案編輯工具建立名為的檔案*hello-world-test.yaml*。包括以下內容：

```
# Document Start
#
name: "HelloWorldTestingDocument-Windows"
description: "Hello world document to demonstrate parameters."
schemaVersion: 1.0
parameters:
  - MyInputParameter:
      type: string
      default: "It's me!"
      description: This is an input parameter.
phases:
  - name: build
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host "Hello World! Build phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: validate
    steps:
      - name: HelloWorldStep
```

```
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host "Hello World! Validate phase. My input parameter value is
 {{ MyInputParameter }}"

  - name: test
    steps:
      - name: HelloWorldStep
        action: ExecutePowerShell
        inputs:
          commands:
            - Write-Host "Hello World! Test phase. My input parameter value is
 {{ MyInputParameter }}"
# Document End
```

> **Tip**
>
> 使用類似這個線上 YAML 驗證程式的工具，或在程式碼環境中使用 YAML lint 延伸模組來確認您的 YAML 格式正確。

如需有關 AWS TOE YAML 應用程式元件文件的階段、步驟和語法的詳細資訊，請參閱在 AWS TOE. 如需有關參數及其需求的詳細資訊，請參參數閱「在中定義和參考變數」 AWS TOE頁面的一節。

從 YAML 元件文件建立元件

無論您使用何種方法來建立 AWS TOE 元件，YAML 應用程式元件文件永遠都需要做為基準。

- 若要使用 Image Builder 主控台直接從 YAML 文件建立元件，請參閱使用映像產生器主控台建立元件。
- 若要使用中的 Image Builder 指令建立元件，請參閱使 AWS TOE 用 Image Builder 建立元件 AWS CLI。 AWS CLI 將這些範例中的 YAML 文件名稱取代為您的 Hello World YAML 文件 () *`hello-world-test.yaml`* 的名稱。

在 Image Builder 方案 (主控台) 中設定元件參數

對於圖像配方和容器配方，設置組件參數的作用相同。當您建立新的方案或新版方案時，您可以從 [建置元件] 和 [測試元件] 清單中選擇要包含的元件。元件清單包含適用於您為映像檔選擇之基本作業系統的元件。

選取零組件後，它會顯示在所選零組件區段中，直接位於元件清單下方。每個選取的元件都會顯示組態選項。如果您的元件已定義輸入參數，它們將顯示為稱為「輸入參數」的可展開區段。

針對您的元件定義的每個參數，會顯示下列參數設定：

- 參數名稱 (不可編輯) — 參數的名稱。
- 說明 (不可編輯) — 參數說明
- 類型 (不可編輯) — 參數值的資料類型。
- 值 — 參數的值。如果您是第一次在此方案中使用此元件，且已為輸入參數定義了預設值，則預設值會顯示在「值」(Val ue) 方塊中，並顯示灰色文字。如果未輸入其他值，「Image Builder」會使用預設值。

列出並檢視元件詳細資訊

本節說明如何尋找資訊並檢視您在 EC2 Image Builder 方法中使用的 AWS 任務協調器和執行器 (AWS TOE) 元件的詳細資訊。

元件細節

- 列出 AWS TOE 元件
- 列出元件建置版本 (AWS CLI)
- 取得元件詳細資訊 (AWS CLI)
- 取得元件原則詳細資料 (AWS CLI)

列出 AWS TOE 元件

您可以使用下列其中一種方法來列出和篩選 AWS TOE 元件。

AWS Management Console

若要在中顯示元件清單 AWS Management Console，請遵循下列步驟：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導覽窗格中選取 [元件]。依預設，Image Builder 會顯示您帳戶擁有的元件清單。
3. 您可以選擇性地篩選元件所有權。若要查看您不擁有但可存取的元件，請展開擁有者類型下拉式清單，然後選取其中一個值。擁有者類型清單位於搜尋列中，位於搜尋文字方塊旁邊。您可以選擇下列的數值：

- 快速入門 (Amazon 管理) — Amazon 建立和維護的公開可用元件。
- 由我擁有 — 您建立的元件。這是預設選項。
- 與我共用 — 其他人透過其帳戶建立並與您共用的元件。
- 第三方管理 — 您訂閱的第三方擁有的組件 AWS Marketplace。

AWS CLI

下列範例顯示如何使用list-components命令傳回您帳戶擁有的 AWS TOE 元件清單。

```
aws imagebuilder list-components
```

您可以選擇性地篩選元件所有權。owner 屬性定義誰擁有您要列出的組件。根據預設，此要求會傳回您帳戶所擁有的元件清單。若要依元件擁有者篩選結果，請在執行list-components命令時使用--owner參數指定下列其中一個值。

元件擁有者值

- 自我
- Amazon
- ThirdParty
- 共同

下列範例顯示具有用於篩選結果之--owner參數的list-components命令。

```
aws imagebuilder list-components --owner Self
{
    "requestId": "012a3456-b789-01cd-e234-fa5678b9012b",
    "componentVersionList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/sample-
component01/1.0.0",
            "name": "sample-component01",
            "version": "1.0.0",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2020-09-24T16:58:24.444Z"
```

```
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/sample-
component01/1.0.1",
            "name": "sample-component01",
            "version": "1.0.1",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2021-07-10T03:38:46.091Z"
        }
    ]
}
```

```
aws imagebuilder list-components --owner Amazon
```

```
aws imagebuilder list-components --owner Shared
```

```
aws imagebuilder list-components --owner ThirdParty
```

列出元件建置版本 (AWS CLI)

下列範例會示範如何使用命list-component-build-versions令列出具有特定語意版本的組件建置版本。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。

```
aws imagebuilder list-component-build-versions --component-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:component/example-component/1.0.1
{
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "componentSummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
examplecomponent/1.0.1/1",
            "name": "examplecomponent",
            "version": "1.0.1",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "description": "An example component that builds, validates and tests an
 image",
```

```
            "changeDescription": "Updated version.",
            "dateCreated": "2020-02-19T18:53:45.940Z",
            "tags": {
                "KeyName": "KeyValue"
            }
        }
    ]
}
```

取得元件詳細資訊 (AWS CLI)

下列範例顯示如何在指定元件的 Amazon 資源名稱 (ARN) 時使用get-component命令取得元件詳細資料。

```
aws imagebuilder get-component --component-build-version-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/example-component/1.0.1/1
   {
    "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11112",
    "component": {
        "arn": "arn:aws:imagebuilder:us-west-2:123456789012:component/
examplecomponent/1.0.1/1",
        "name": "examplecomponent",
        "version": "1.0.1",
        "type": "BUILD",
        "platform": "Linux",
        "owner": "123456789012",
        "data": "name: HelloWorldTestingDocument\ndescription: This is hello world
 testing document... etc.\"\n",
        "encrypted": true,
        "dateCreated": "2020-09-24T16:58:24.444Z",
        "tags": {}
    }
}
```

取得元件原則詳細資料 (AWS CLI)

下列範例顯示如何在指定元件的 ARN 時，使用get-component-policy命令取得元件原則的詳細資料。

```
aws imagebuilder get-component-policy --component-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/example-component/1.0.1
```

使用映像產生器主控台建立元件

若要從 Image Builder 主控台建立 AWS TOE 應用程式元件，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導覽窗格中選取 [元件]。然後選取「建立元件」。
3. 在 [建立元件] 頁面的 [元件詳細資訊] 底下，輸入下列內容：
   a. 映像作業系統 (OS)。指定元件相容的作業系統。
   b. 元件類別。從下拉式清單中，選取您要建立的組建或測試元件類型。
   c. 元件名稱。輸入元件的名稱。
   d. 元件版本。輸入元件的版本號碼。
   e. 描述。提供可選描述以協助您識別元件。
   f. 變更說明。提供選擇性描述，以協助您瞭解對此版本元件所做的變更。
4. 在「定義文件」區段中，預設選項為「定義文件內容」。元件文件會定義 Image Builder 在組建和測試執行個體上執行的動作，以建立映像檔。

   在 [內容] 方塊中，輸入您的 YAML 元件文件內容。若要從 Linux 的「你好世界」範例開始，請選擇「使用範例」選項。若要深入了解如何建立 YAML 元件文件，或從該頁面複製並貼上 UpdateOS 範例，請參閱。建立 YAML 元件文件
5. 輸入元件詳細資訊之後，請選取建立元件。

   > **Note**
   >
   > 若要在建立或更新方案時查看新元件，請將 [我擁有] 篩選器套用至組建或測試元件清單。篩選器位於元件清單頂端，位於搜尋方塊旁邊。

6. 若要刪除元件，請從「元件」頁面選取要刪除之元件旁邊的核取方塊。從動作下拉式清單中，選取刪除元件。

若要建立新的元件版本，請依照下列步驟執行：

1. 根據您從哪裡開始：
   - 從 [元件] 清單頁面 — 選取元件名稱旁的核取方塊，然後從 [動作] 功能表中選取 [建立新版本]。
   - 從元件詳細資訊頁面 — 選擇標題右上角的 [建立新版本] 按鈕。

2. 顯示「建立元件」頁面時，元件資訊已填入目前的值。遵循建立元件步驟來更新元件。這可確保您在元件版本中輸入唯一的語意版本。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。

使用建立元件 AWS CLI

本節說明如何使用 Image Builder 指令從中建立 AWS 任務協調器和執行器 (AWS TOE) 元件 AWS Command Line Interface。若要建置元件，請提供 YAML 應用程式元件文件。這表示建立元件所需的階段和步驟。若要建立新的 YAML 元件文件，請參閱建立 YAML 元件文件。

使 AWS TOE 用 Image Builder 建立元件 AWS CLI

在本節中，您將學習如何在中設定和使用 Image Builder 命令 AWS CLI 來建立 AWS TOE 應用程式元件，如下所示。

- 將 YAML 元件文件上傳到可從命令列參考的 S3 儲存貯體。
- 使用create-component指令建立 AWS TOE 應用程式元件。
- 使用list-components指令和名稱篩選列示元件版本，以查看已存在的版本。您可以使用輸出來確定下一個版本應該是什麼更新。

若要從輸入 YAML 文件建立 AWS TOE 應用程式元件，請遵循符合映像作業系統平台的步驟。

Linux

將您的應用程式元件文件存放在 Amazon S3 中

您可以使用 S3 儲存貯體做為應用 AWS TOE 程式元件來源文件的儲存庫。若要儲存元件文件，請依照下列步驟執行：

- 將文件上傳到 Amazon S3

  如果您的文件小於 64 KB，則可以略過此步驟。大小為 64 KB 或更大的文件必須存放在 Amazon S3 中。

```
aws s3 cp update-linux-os.yaml s3://my-s3-bucket/my-path/update-linux-os.yaml
```

從 YAML 文件建立元件

若要簡化您在中使用的create-component命令 AWS CLI，請建立 JSON 檔案，其中包含您要傳入指令的所有元件參數。包括您在先前步驟中建立的*update-linux-os.yaml*文件位置。uri鍵值對包含檔案參照。

> **Note**
>
> JSON 檔案中資料值的命名慣例遵循針對 Image Builder API 動作要求參數指定的模式。若要檢閱 API 命令要求參數，請參閱 EC2 Image Builder API 參考中的CreateComponent命令。
> 若要將資料值提供為指令行參數，請參考《指AWS CLI 令參考》中指定的參數名稱。

1. 建立 CLI 輸入 JSON 文件

   使用檔案編輯工具建立名為的檔案*create-update-linux-os-component.json*。包括以下內容：

```
{
 "name": "update-linux-os",
 "semanticVersion": "1.1.2",
 "description": "An example component that updates the Linux operating system",
 "changeDescription": "Initial version.",
 "platform": "Linux",
 "uri": "s3://my-s3-bucket/my-path/update-linux-os.yaml",
 "kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/98765432-
b123-456b-7f89-0123456f789c",
 "tags": {
  "MyTagKey-purpose": "security-updates"
 }
}
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

2. 建立元件

   使用下列命令建立元件，參考您在上一個步驟中建立之 JSON 檔案的檔案名稱：

   ```
   aws imagebuilder create-component --cli-input-json file://create-update-linux-os-component.json
   ```

   > **Note**
   >
   > - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
   > - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

Windows

將您的應用程式元件文件存放在 Amazon S3 中

您可以使用 S3 儲存貯體做為應用 AWS TOE 程式元件來源文件的儲存庫。若要儲存元件文件，請依照下列步驟執行：

- 將文件上傳到 Amazon S3

  如果您的文件小於 64 KB，則可以略過此步驟。大小為 64 KB 或更大的文件必須存放在 Amazon S3 中。

  ```
  aws s3 cp update-windows-os.yaml s3://my-s3-bucket/my-path/update-windows-os.yaml
  ```

從 YAML 文件建立元件

若要簡化您在中使用的create-component命令 AWS CLI，請建立 JSON 檔案，其中包含您要傳入指令的所有元件參數。包括您在先前步驟中建立的*update-windows-os.yaml*文件位置。`uri`鍵值對包含檔案參照。

> **Note**
>
> JSON 檔案中資料值的命名慣例遵循針對 Image Builder API 動作要求參數指定的模式。若要檢閱 API 命令要求參數，請參閱 EC2 Image Builder API 參考中的CreateComponent命令。
> 若要將資料值提供為指令行參數，請參閱「指AWS CLI 令參考」中指定的參數名稱。

1. 建立 CLI 輸入 JSON 文件

   使用檔案編輯工具建立名為的檔案*create-update-windows-os-component.json*。包括以下內容：

```
{
 "name": "update-windows-os",
 "semanticVersion": "1.1.2",
 "description": "An example component that updates the Windows operating
 system.",
 "changeDescription": "Initial version.",
 "platform": "Windows",
 "uri": "s3://my-s3-bucket/my-path/update-windows-os.yaml",
 "kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/98765432-
b123-456b-7f89-0123456f789c",
 "tags": {
  "MyTagKey-purpose": "security-updates"
 }
}
```

   > **Note**
   >
   > - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
   > - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

2. 建立元件

   使用下列命令建立元件，參考您在上一個步驟中建立之 JSON 檔案的檔案名稱：

```
aws imagebuilder create-component --cli-input-json file://create-update-windows-os-component.json
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

AWS TOE 更新的元件版本控制 (AWS CLI)

AWS TOE 元件名稱和版本內嵌在元件的 Amazon 資源名稱 (ARN) 中，在元件前置詞之後。元件的每個新版本都有自己唯一的 ARN。建立新版本的步驟與建立新元件的步驟完全相同，只要語意版本對該元件名稱而言是唯一的。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。

若要確保您指派下一個邏輯版本，請先取得要變更之元件的現有版本清單。將指list-components令與和篩選名稱搭配使用。 AWS CLI

在此範例中，您會篩選先前 Linux 範例中建立的元件名稱。若要列出您建立的元件，請使用您在create-component命令中使用的 JSON 檔案中的name參數值。

```
aws imagebuilder list-components --filters name="name",values="update-linux-os"
{
    "requestId": "123a4567-b890-123c-45d6-ef789ab0cd1e",
    "componentVersionList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:1234560087789012:component/update-linux-os/1.0.0",
            "name": "update-linux-os",
            "version": "1.0.0",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2020-09-24T16:58:24.444Z"
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:1234560087789012:component/update-linux-os/1.0.1",
```

```
            "name": "update-linux-os",
            "version": "1.0.1",
            "platform": "Linux",
            "type": "BUILD",
            "owner": "123456789012",
            "dateCreated": "2021-07-10T03:38:46.091Z"
        }
    ]
}
```

根據您的結果，您可以確定下一個版本應該是什麼。

匯入元件 (AWS CLI)

在某些情況下，使用預先存在的指令碼開始可能會比較容易。在這個案例中，您可以使用下列範例。

此範例假設您有一個名為 *import-component.json* (如圖所示) 的檔案。請注意，檔案會直接參考已上傳至AdminConfig.ps1的名為的 PowerShell 指令碼*my-s3-bucket*。目前SHELL，該組件支持format。

```
{
"name": "MyImportedComponent",
"semanticVersion": "1.0.0",
"description": "An example of how to import a component",
"changeDescription": "First commit message.",
"format": "SHELL",
"platform": "Windows",
"type": "BUILD",
"uri": "s3://my-s3-bucket/AdminConfig.ps1",
"kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/60763706-
b131-418b-8f85-3420912f020c"
}
```

若要匯入元件，請執行下列指令。

```
aws imagebuilder import-component --cli-input-json file://import-component.json
```

清除資源

若要避免非預期的費用，請務必清理您從本指南中的範例建立的資源和管線。如需有關在 Image Builder 中刪除資源的詳細資訊，請參閱刪除 EC2 Image Builder 資源。

管理食譜

EC2 Image Builder 方案會定義基礎映像作為建立新映像的起點，以及您新增用來自訂映像的元件集，並確認一切正常運作。Image Builder 會為每個元件提供自動版本選擇。根據預設，您最多可以將 20 個元件套用至配方。這包括構建和測試組件。

建立方案之後，就無法修改或取代它。若要在建立配方之後更新元件，您必須建立新的方案或配方版本。您始終可以將標籤應用於現有的食譜。如需有關使用「Image Builder」指令標記資源的詳細資訊 AWS CLI，請參閱本指南一標籤資源節。

> **Tip**
>
> 您可以在配方中使用 Amazon 受管元件，也可以使用 AWS 任務協調器和執行器 (AWS TOE) 應用程式開發自己的自訂元件。若要開始使用，請參閱開始使用 AWS TOE。

本節介紹如何列出，查看和創建配方。

目錄

列出並查看圖像配方詳細信息

本節說明您可以尋找 EC2 Image Builder 映像配方的資訊和檢視詳細資料的各種方式。

圖片配方詳細信

列出圖像食譜（控制台）

若要查看在 Image Builder 主控台中您帳戶下建立的影像配方清單，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導航窗格中選擇圖像配方。這將顯示在您的帳戶下創建的圖像配方的列表。
3. 若要檢視詳細資料或建立新的配方版本，請選擇配方名稱連結。這會開啟配方的詳細檢視。

> **Note**
>
> 您也可以選取「方案」名稱旁的核取方塊，然後選擇「檢視詳細資料」。

列表圖像食譜（AWS CLI）

下面的例子演示了如何列出所有的圖像配方，使用 AWS CLI.

```
aws imagebuilder list-image-recipes
```

檢視影像配方詳細資料 (主控台)

若要使用 Image Builder 主控台檢視特定影像方案的詳細資訊，請使用中所述的步驟選取要檢閱的映像配方列出圖像食譜（控制台）。

在食譜詳細資料頁面上，您可以：

- 刪除配方。如需有關在 Image Builder 中刪除資源的詳細資訊，請參閱刪除 EC2 Image Builder 資源。
- 建立新版本。
- 從配方建立管道。從此方案中選擇「建立管線」之後，即會前往管線精靈。如需有關使用管線精靈建立 Image Builder 管線的詳細資訊，請參閱 使用 EC2 Image Builder 主控台精靈建立映像管道

> **Note**
>
> 當您從現有方案建立管線時，無法使用建立新配方的選項。

獲取圖像配方詳細信息（AWS CLI）

下列範例顯示如何使用 imagebuilder CLI 命令，透過指定其 Amazon 資源名稱 (ARN) 來取得映像方案的詳細資訊。

```
aws imagebuilder get-image-recipe --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-example-recipe/2020.12.03
```

取得影像配方政策詳細資料 (AWS CLI)

下列範例顯示如何使用 imagebuilder CLI 命令，藉由指定其 ARN 來取得映像配方原則的詳細資料。

```
aws imagebuilder get-image-recipe-policy --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-example-recipe/2020.12.03
```

列出並檢視容器配方詳細資料

本節說明如何尋找 EC2 Image Builder 容器配方的資訊和檢視詳細資料。

容器配方詳情

- 在主控台中列出容器配方
- 列出容器食譜 AWS CLI
- 在主控台中檢視容器配方詳細資料
- 取得容器配方詳細資訊 AWS CLI
- 取得容器配方政策詳細資料 AWS CLI

在主控台中列出容器配方

若要查看已在 Image Builder 主控台的帳戶下建立的容器配方清單，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導航窗格中選擇容器配方。這會顯示在您的帳戶下建立的容器配方清單。
3. 若要檢視詳細資料或建立新的配方版本，請選擇配方名稱連結。這會開啟配方的詳細檢視。

> **Note**
>
> 您也可以選取「方案」名稱旁的核取方塊，然後選擇「檢視詳細資料」。

列出容器食譜 AWS CLI

下面的例子演示如何列出所有的容器配方，使用 AWS CLI.

```
aws imagebuilder list-container-recipes
```

在主控台中檢視容器配方詳細資料

若要使用 Image Builder 主控台檢視特定容器方案的詳細資訊，請選取要檢閱的容器方案，然後使用中所述的步驟在主控台中列出容器配方。

在配方詳細資料頁面上，您可以執行下列動作：

- 刪除配方。如需如何在 Image Builder 中刪除資源的詳細資訊，請參閱刪除 EC2 Image Builder 資源。
- 建立新版本。
- 從配方建立管道。在您選擇 [從此方案建立管線] 之後，便會前往管線精靈。如需如何使用管線精靈建立 Image Builder 管線的詳細資訊，請參閱 使用 EC2 Image Builder 主控台精靈建立映像管道

> **Note**
>
> 當您從現有配方建立管道時，無法使用建立新配方的選項。

取得容器配方詳細資訊 AWS CLI

下列範例顯示如何使用 imagebuilder CLI 命令，藉由指定容器方案的 ARN 來取得容器方案的詳細資料。

```
aws imagebuilder get-container-recipe --container-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:container-recipe/my-example-recipe/2020.12.03
```

取得容器配方政策詳細資料 AWS CLI

下列範例顯示如何使用 imagebuilder CLI 命令，藉由指定容器方案原則的 ARN 來取得容器方案原則的詳細資料。

```
aws imagebuilder get-container-recipe-policy --container-recipe-arn
 arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-example-
recipe/2020.12.03
```

建立影像配方的新版本

本節說明如何建立影像配方的新版本。

目錄

建立新的映像配方版本 (主控台)

當您建立新的配方版本時，它與建立新配方幾乎相同。不同之處在於，在大多數情況下，某些詳細信息被預先選擇以匹配基本配方。下列清單說明建立新方案與建立現有方案的新版本之間的差異。

新版本中的基本配方詳細信息

- 名稱 — 不可編輯。
- 版本 — 必要。此基本詳細資料未預先填入目前版本或任何類型的序列。輸入您要以格式建立的版本號碼<major>。 <minor>。 <patch>。如果版本已存在，則會遇到錯誤。
- 選擇圖像選項-預先選擇，但您可以對其進行編輯。如果您變更基本影像來源的選擇，則可能會遺失其他詳細資料，這取決於您選擇的原始選項。

  若要查看與基本影像選取項目相關聯的詳細資料，請選擇符合您選取範圍的索引標籤。

  Managed image

  - 映像作業系統 (OS) — 不可編輯。
  - 映像名稱 — 根據您為現有方案所做的基本映像選擇組合而預先選取。但是，如果您變更「選取影像」選項，則會遺失預先選取的影像名稱。
  - 自動版本控制選項 — 與您的基本配方不匹配。此映像檔選項預設為 [使用選取的作業系統版本] 選項。

> **Important**
>
> 如果您使用語義版本控制來啟動管道構建，請確保將此值更改為「使用最新的可用操作系統版本」。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。

AWS Marketplace image

- 訂閱 — 此索引標籤應該已開啟，且 AWS Marketplace 應預先選取已訂閱的映像來源，以符合您的基本配方。如果您變更方案使用的影像做為其基本影像，可能會遺失其他詳細資料，這些詳細資料取決於您選擇的原始影像。

如需更多有關 AWS Marketplace 產品的資訊，請參閱購買指南中的「AWS Marketplace 購買產品」。

Custom AMI

- AMI 識別碼 — 必要。但是，此設定不會預先填入您的原始項目。您必須輸入基本映像檔的 AMI ID。

- 執行個體組態 — 已預先選取設定，但您可以編輯它們。
  - 系統管理員代理程式 — 您可以選取或清除此核取方塊，以控制系統管理員代理程式在新映像上的安裝。依預設，會清除此核取方塊，以便在新映像中包含系統管理員代理程式。若要從最終映像中移除系統管理員代理程式，請選取核取方塊，讓代理程式不會包含在 AMI 中。
  - 使用者資料 — 當您啟動建置執行個體時，您可以使用此區域提供命令或要執行的命令指令碼。但是，此值會取代 Image Builder 可能已加入的任何指令，以確保已安裝「Systems Manager」。這些命令包括在建立新映像之前，Image Builder 通常針對 Linux 映像執行的清理指令碼。

> **Note**
>
> - 如果您輸入使用者資料，請確定系統管理員代理程式已預先安裝在您的基礎映像上，或將安裝包含在使用者資料中。
> - 對於 Linux 映像檔，請透過包含建立`perform_cleanup`在使用者資料指令碼中命名的空白檔案的命令來執行清理步驟。Image Builder 會偵測到此檔案，並在建立新映像之前執行清理程序檔。如需詳細資訊和範例指令碼，請參閱EC2 Image Builder 的安全最佳實務。

- 工作目錄 — 預先選取，但您可以對其進行編輯。

- 元件 — 已包含在方案中的元件會顯示在每個元件清單 (建置與測試) 結尾的 [選取的元件] 區段中。您可以移除或重新排序選取的元件，以符合您的需求。

  CIS 強化元件不遵循 Image Builder 配方中的標準元件排序規則。CIS 強化元件一律會持續執行，以確保基準測試會針對您的輸出影像執行。

  > **Note**
  >
  > 建置和測試元件清單會根據元件擁有者類型顯示可用的元件。若要為您的方案新增或更新元件，請為您要尋找的元件選取擁有者類型。例如，如果您要新增與您在中訂閱之基礎映像檔相關聯的元件 AWS Marketplace，請`Third party managed`從搜尋列旁邊的擁有者類型清單中選取。

  您可以為選取的元件設定下列設定：

  - 版本控制選項 — 預先選取，但您可以變更它們。我們建議您選擇 [使用最新的可用元件版本] 選項，以確保映像組建始終會選取最新版本的元件。如果您需要在方案中使用特定元件版本，您可以選擇 [指定元件版本]，然後在出現的 [元件版本] 方塊中輸入版本。
  - 輸入參數 — 顯示元件接受的輸入參數。「值」會預先填入方案先前版本的值。如果您是第一次在此方案中使用此元件，且已為輸入參數定義了預設值，則預設值會顯示在「值」(Val ue) 方塊中，並顯示灰色文字。如果未輸入其他值，Image Builder 會使用預設值。

    如果需要輸入參數，但未在組件中定義預設值，則必須提供值。如果缺少任何必要參數且未定義預設值，Image Builder 將不會建立配方版本。

    > **Important**
    >
    > 元件參數是純文字值，並已登入 AWS CloudTrail。我們建議您使用 AWS Secrets Manager 或 AWS Systems Manager 參數存放區來儲存您的秘密。如需有關 Secrets Manager 的詳細資訊，請參閱什麼是 Secrets Manager？ 在《AWS Secrets Manager 使用者指南》中。若要取得有關 AWS Systems Manager 參數存放區的更多資訊，請AWS Systems Manager 參閱AWS Systems Manager 使用指南中的參數存放區

  若要展開「版本控制」選項或「輸入」參數的設定，您可以選擇設定名稱旁邊的箭頭。若要展開所有選取零組件的所有設定，您可以將全部展開開關切換為關閉和開啟。

- 儲存 (磁碟區) — 已預先填入。無法編輯根磁碟區裝置名稱、快照和 IOPS 選項。但是，您可以變更所有剩餘的設定，例如「大小」。您也可以新增磁碟區，以及加密新磁碟區或現有磁碟區。

若要為 Image Builder 在來源區域 (執行組建的位置) 的帳戶下建立的映像加密磁碟區，您必須在映像配方中使用儲存磁碟區加密。在組建發佈階段執行的加密僅適用於散佈至其他帳戶或區域的映像。

> **Note**
>
> 如果您對磁碟區使用加密，則必須分別為每個磁碟區選取金鑰，即使金鑰與根磁碟區所使用的金鑰相同。

若要建立新的映像配方版本：

1. 在配方詳細資料頁面的頂端，選擇 [建立新版本]。這會帶您前往 [建立映像配方] 頁面。
2. 若要建立新版本，請進行變更，然後選擇 [建立映像配方]。

如需有關如何在建立映像管線時建立映像配方的詳細資訊，請參閱步驟 2：選擇食譜本指南的「入門」一節。

創建一個圖像配方 AWS CLI

若要使用中的「映像產生器」`create-image-recipe` 指令建立影像方案 AWS CLI，請依照下列步驟執行：

必要條件

在執行本節中的 Image Builder 命令以從中建立影像配方之前 AWS CLI，您必須先建立方案使用的元件。下列步驟中的影像配方範例是指本指南一使用建立元件 AWS CLI節中建立的範例元件。

建立元件後，或使用現有元件後，請注意您要包含在配方中的 ARN。

1. 建立 CLI 輸入 JSON 文件

   您可以使用內嵌命令參數為命create-image-recipe令提供所有輸入。但是，生成的命令可能很長。若要簡化指令，您可以改為提供包含所有方案設定的 JSON 檔案。

   > **Note**
   >
   > JSON 檔案中資料值的命名慣例遵循針對 Image Builder API 動作要求參數指定的模式。若要檢閱 API 命令要求參數，請參閱 EC2 Image Builder API 參考中的CreateImageRecipe命令。

若要將資料值提供為指令行參數，請參考《指AWS CLI 令參考》中指定的參數名稱。

以下是這些範例指定的參數摘要：

- name (字串，必要) — 影像配方的名稱。
- 描述（字符串）— 圖像配方的描述。
- ParentImage (字串，必要) — 影像配方用來做為自訂影像基礎的影像。該值可以是基本映像 ARN 或 AMI ID。

> **Note**
>
> Linux 範例使用 Image Builder AMI，而視窗範例則使用 ARN。

- SemanticVersion <major>(字串，必要) — 影像配方的語意版本，以下列格式表示，每個位置都有數值以表示特定版本：。 <minor>。 <patch>。例如，一個值可能是`1.0.0`。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。
- 元件 (陣列，必要) — 包含`ComponentConfiguration`物件陣列。至少必須指定一個組建元件：

> **Note**
>
> Image Builder 會依照您在方案中指定元件的順序來安裝元件。不過，CIS 強化元件一律會持續執行，以確保基準測試會針對您的輸出影像執行。

  - 元件目錄 (字串，必要) — 元件 ARN。

> **Tip**
>
> 若要使用其中一個範例來建立您自己的映像配方，您必須將範例 ARN 取代為您用於方案之元件的 ARN。

  - 參數 (物件陣列) — 包含`ComponentParameter`物件陣列。如果需要輸入參數，但未在組件中定義預設值，則必須提供值。如果缺少任何必要參數且未定義預設值，Image Builder 將不會建立配方版本。

> **Important**
>
> 元件參數是純文字值，並已登入 AWS CloudTrail。我們建議您使用 AWS Secrets Manager 或 AWS Systems Manager 參數存放區來儲存您的秘密。如需有關 Secrets Manager 的詳細資訊，請參閱什麼是 Secrets Manager？ 在《AWS Secrets Manager 使用者指南》中。若要取得有關 AWS Systems Manager 參數存放區的更多資訊，請AWS Systems Manager 參閱AWS Systems Manager 使用指南中的參數存放區

- name (字串，必要) — 要設定的元件參數名稱。
- value (字串陣列，必要) — 包含用來設定具名元件參數值的字串陣列。如果為元件定義了預設值，且未提供其他值，則 AWS TOE 會使用預設值。

- additionalInstanceConfiguration(物件) — 指定組建執行個體的其他設定和啟動指令碼。
  - systemsManagerAgent(物件) — 包含組建執行個體上系統管理員代理程式的設定。
    - uninstallAfterBuild(布林值) — 控制是否在建立新 AMI 之前從最終組建映像中移除系統管理員代理程式。如果此選項設為`true`，則會從最終映像中移除代理程式。如果選項設定為`false`，則會留下代理程式，以便將其包含在新 AMI 中。預設值為 `false`。

      > **Note**
      >
      > 如果`uninstallAfterBuild`屬性未包含在 JSON 檔案中，且符合下列條件，則 Image Builder 會從最終映像中移除系統管理員代理程式，使其無法在 AMI 中使用：
      >
      > - `userDataOverride`為空或已從 JSON 檔案中省略。
      > - Image Builder 會自動在組建執行個體上安裝系統管理員代理程式，而該作業系統並未在基本映像上預先安裝代理程式。

  - userDataOverride(字串) — 提供在啟動建置執行個體時要執行的命令或命令指令碼。

    > **Note**
    >
    > 使用者資料一律為 Base 64 編碼格式。例如，下列命令編碼為 `IyEvYmluL2Jhc2gKbWtkaXIgLXAgL3Zhci9iYi8KdG91Y2ggL3Zhcg==`：
    >
    > ```
    > #!/bin/bash
    > ```

```
mkdir -p /var/bb/
touch /var
```

Linux 範例使用此編碼值。

Linux

下列範例中的基本影像 (parentImage屬性) 為 AMI。當您使用 AMI 時，您必須能夠存取 AMI，且 AMI 必須位於來源區域 (Image Builder 執行命令的相同區域)。將檔案另存為create-image-recipe.json，並在create-image-recipe指令中使用它。

```
{
"name": "BB Ubuntu Image recipe",
"description": "Hello World image recipe for Linux.",
"parentImage": "ami-0a01b234c5de6fabc",
"semanticVersion": "1.0.0",
"components": [
 {
  "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/bb$"
 }
],
"additionalInstanceConfiguration": {
 "systemsManagerAgent": {
   "uninstallAfterBuild": true
 },
 "userDataOverride": "IyEvYmluL2Jhc2gKbWtkaXIgLXAgL3Zhci9iYi8KdG91Y2ggL3Zhcg=="
}
}
```

Windows

下面的例子是指視窗服務器 2016 年英語完整基礎圖像的最新版本。此範例中的 ARN 會根據您指定的語意版本篩選器參考 SKU 中的最新映像：arn:aws:imagebuilder:us-west-2:aws:image/windows-server-2016-english-full-base-x86/x.x.x。

```
{
"name": "MyBasicRecipe",
"description": "This example image recipe creates a Windows 2016 image.",
"parentImage": "arn:aws:imagebuilder:us-west-2:aws:image/windows-server-2016-english-full-base-x86/x.x.x",
```

```
"semanticVersion": "1.0.0",
"components": [
 {
  "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/my-example-component/2019.12.02/1"
 },
 {
  "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/my-imported-component/1.0.0/1"
 }
]
}
```

> **Note**
>
> 若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。

2. 建立食譜

使用下列指令建立配方。在參數中提供您在上一個步驟中建立的 JSON 檔案名`--cli-input-json`稱：

```
aws imagebuilder create-image-recipe --cli-input-json file://create-image-recipe.json
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

在主控台中將虛擬機器匯入為基礎映像

在本節中，我們將重點介紹如何將虛擬機器 (VM) 匯入為映像方案的基礎映像。我們不涵蓋與創建食譜或配方版本相關的其他步驟。如需使用 Image Builder 主控台中的管線建立精靈建立新映像方案的其他步驟，請參閱建立映像管線 (AMI)。如需建立新影像配方或配方版本的其他步驟，請參閱建立影像配方的新版本。

若要將虛擬機器匯入為映像產生器主控台中映像配方的基本映像檔，請遵循下列步驟以及任何其他必要步驟來建立方案或方案版本。

1. 在基本影像的「選取影像」區段中，選取「匯入基本影像」選項。
2. 像平常一樣選擇映像操作系統（OS）和操作系統版本。

VM 匯入組態

當您從虛擬化環境匯出 VM 時，該程序會建立一組或多個磁碟容器檔案，做為 VM 環境、設定和資料的快照集。您可以使用這些檔案將 VM 匯入為映像配方的基本映像檔。如需有關在 Image Builder 中匯入 VM 的詳細資訊，請參閱 匯入和匯出 VM 映像檔

若要指定匯入來源的位置，請依照下列步驟執行：

匯入來源

在 [磁碟容器 1] 區段中指定要匯入的第一個虛擬機器映像磁碟容器或快照的來源。

1. 來源 — 這可以是 S3 儲存貯體或 EBS 快照。
2. 選取磁碟 S3 位置 — 輸入 Amazon S3 中存放磁碟映像的位置。若要瀏覽位置，請選擇 [瀏覽 S3]。
3. 若要新增磁碟容器，請選擇 [新增磁碟容器]。

IAM 角色

若要將 IAM 角色與 VM 匯入設定建立關聯，請從 IAM 角色下拉式清單中選取角色，或選擇 [建立新角色] 以建立新角色。如果您建立新角色，IAM 角色主控台頁面會在單獨的索引標籤中開啟。

進階設定 — 選用

以下是可選的設定。使用這些設定，您可以為匯入所建立的基礎映像配置加密、授權、標籤等。

一般

1. 為基礎映像指定唯一的「名稱」。如果未輸入值，則基本影像會繼承配方名稱。
2. 指定基本映像的版本。使用下列格式：*<major>.<minor>.<patch>*。如果未輸入值，則基本影像會繼承配方版本。
3. 您也可以輸入基本影像的「描述」。

基本映像架構

若要指定虛擬機器匯入來源的架構，請從架構清單中選取值。

加密

如果您的虛擬機器磁碟映像檔已加密，您必須提供用於匯入程序的金鑰。若要 AWS KMS key 為匯入指定，請從加密 (KMS 金鑰) 清單中選取一個值。此清單包含您的帳戶在目前區域中可存取的 KMS 金鑰。

授權管理

匯入虛擬機器時，匯入程序會自動偵測虛擬機器作業系統，並將適當的授權套用至基礎映像。視您的作業系統平台而定，授權類型如下：

- 已包含授權 — 將適用於您平台的適當 AWS 授權套用至您的基礎映像。
- 攜帶您自己的授權 (BYOL) — 保留來自虛擬機器的授權 (如果適用)。

若要將使用建立的授權規劃附加 AWS License Manager 至基礎映像，請從 [授權規劃名稱] 清單中選取。如需有關 License Manager 的更多資訊，請參閱使用 AWS License Manager

> **Note**
>
> - 授權組態包含以企業合約條款為基礎的授權規則。
> - Linux 僅支援自攜授權。

標籤 (基本影像)

標籤使用鍵值配對，將可搜尋的文字指派給您的 Image Builder 資源。若要為匯入的基本影像指定標籤，請在「鍵」和「值」方塊中輸入鍵值配對。

若要新增標籤，請選擇 Add tag (新增標籤)。若要移除標籤，請選擇 Remove tag (移除標籤)。

建立容器配方的新版本

本節說明如何建立容器配方的新版本。

目錄

- 使用主控台建立新的容器配方版本
- 建立容器配方 AWS CLI

使用主控台建立新的容器配方版本

建立新版本的容器配方與建立新配方幾乎相同。不同之處在於，在大多數情況下，某些詳細信息被預先選擇以匹配基本配方。下列清單說明建立新方案與建立現有方案的新版本之間的差異。

食譜詳情

- 名稱 — 不可編輯。
- 版本 — 必要。此詳細資料未預先填入目前版本或任何類型的序列。輸入您要以主要 .min or .patch 格式建立的版本號碼。如果版本已存在，則會遇到錯誤。

基本影像

- 選擇圖像選項 — 預先選擇，但可編輯。如果您變更基本影像來源的選擇，可能會遺失其他詳細資料，這取決於您選擇的原始選項。

  若要查看與基本影像選取項目相關聯的詳細資料，請選擇符合您選取範圍的索引標籤。

  Managed images

  - 映像作業系統 (OS) — 不可編輯。
  - 映像名稱 — 根據您為現有方案所做的基本映像選擇組合預先選取。但是，如果您變更「選取影像」選項，則會遺失預先選取的影像名稱。
  - 自動版本控制選項 — 與您的基本配方不匹配。自動版本控制選項預設為 [使用選取的 OS 版本] 選項。

    > **Important**
    >
    > 如果您使用語義版本控制來啟動管道構建，請確保將此值更改為「使用最新的可用操作系統版本」。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。

  ECR image

  - 映像作業系統 (OS) — 預先選取，但可編輯。
  - 作業系統版本 — 預先選取，但可編輯。

- ECR 影像 ID — 預先填入但可編輯。

Docker Hub image

- 映像作業系統 (OS) — 不可編輯。
- 作業系統版本 — 預先選取，但可編輯。
- 碼頭圖像 ID — 預填充，但可編輯。

執行個體組態

- AMI ID — 預先填充，但可編輯。
- 儲存 (磁碟區)

  EBS 卷 1（AMI 根）-預填充。您無法編輯根磁碟區裝置名稱、快照或 IOPS 選項。但是，您可以變更所有剩餘的設定，例如「大小」。您也可以新增磁碟區。

> **Note**
>
> 如果您指定了從其他帳戶與您共用的基礎 AMI，則指定的任何輔助磁碟區的快照也必須與您的帳戶共用。

工作目錄

- 工作目錄路徑 — 預先填滿但可編輯。

元件

- 元件 — 已包含在方案中的元件會顯示在每個元件清單 (建置與測試) 結尾的 [選取的元件] 區段中。您可以移除或重新排序選取的元件，以符合您的需求。

  CIS 強化元件不遵循 Image Builder 配方中的標準元件排序規則。CIS 強化元件一律會持續執行，以確保基準測試會針對您的輸出影像執行。

> **Note**
>
> 建置和測試元件清單會根據元件擁有者類型顯示可用的元件。若要為您的方案新增或更新元件，請為您要尋找的元件選取擁有者類型。例如，如果您要新增與您在中訂閱之基礎映像檔

相關聯的元件 AWS Marketplace，請`Third party managed`從搜尋列旁邊的擁有者類型清單中選取。

您可以為選取的元件設定下列設定：

- 版本控制選項 — 預先選取，但您可以變更它們。我們建議您選擇 [使用最新的可用元件版本] 選項，以確保映像組建始終會選取最新版本的元件。如果您需要在方案中使用特定元件版本，您可以選擇 [指定元件版本]，然後在出現的 [元件版本] 方塊中輸入版本。
- 輸入參數 — 顯示元件接受的輸入參數。「值」會預先填入方案先前版本的值。如果您是第一次在此方案中使用此元件，且已為輸入參數定義了預設值，則預設值會顯示在「值」(Val ue) 方塊中，並顯示灰色文字。如果未輸入其他值，Image Builder 會使用預設值。

  如果需要輸入參數，但未在組件中定義預設值，則必須提供值。如果缺少任何必要參數且未定義預設值，Image Builder 將不會建立配方版本。

  > **Important**
  >
  > 元件參數是純文字值，並已登入 AWS CloudTrail。我們建議您使用 AWS Secrets Manager 或 AWS Systems Manager 參數存放區來儲存您的秘密。如需有關 Secrets Manager 的詳細資訊，請參閱什麼是 Secrets Manager？ 在《AWS Secrets Manager 使用者指南》中。若要取得有關 AWS Systems Manager 參數存放區的更多資訊，請AWS Systems Manager 參閱AWS Systems Manager 使用指南中的參數存放區

若要展開「版本控制」選項或「輸入」參數的設定，您可以選擇設定名稱旁邊的箭頭。若要展開所有選取零組件的所有設定，您可以將全部展開開關切換為關閉和開啟。

碼頭文件模板

- 碼頭文件模板-預填充，但可編輯。您可以指定下列 Image Builder 在執行階段以組建資訊取代的下列任何內容變數。

  父母圖像（必填）

  在建置階段，此變數會解析為方案的基礎映像檔。

  範例：

```
FROM
{{{ imagebuilder:parentImage }}}
```

環境 (如果已指定元件，則需要)

此變數將解析為執行元件的指令碼。

範例：

```
{{{ imagebuilder:environments }}}
```

組件（可選）

Image Builder 會針對容器配方所包含的元件，解析組建和測試元件指令碼。此變數可以放置在 Docker 檔案中的任何位置，位於環境變數之後。

範例：

```
{{{ imagebuilder:components }}}
```

目標儲存庫

- 目標存放庫名稱 — 如果在管道執行的區域 (區域 1) 的管道分發組態中未指定其他儲存庫，則儲存輸出影像的 Amazon ECR 儲存庫。

若要建立新的容器配方版本：

1. 在容器配方詳細資料頁面的頂端，選擇 [建立新版本]。您會前往容器配方的 [建立配方] 頁面。
2. 若要建立新版本，請進行變更，然後選擇 [建立方案]。

如需有關如何在建立映像管線時建立容器配方的詳細資訊，請參閱本指南的「入門」一節步驟 2：選擇食譜中的。

建立容器配方 AWS CLI

若要使用中的`imagebuilder create-container-recipe`指令建立 Image Builder 器容器方案 AWS CLI，請依照下列步驟執行：

必要條件

在執行本節中的 Image Builder 命令以建立容器方案之前 AWS CLI，您必須先建立方案將使用的元件。以下步驟中的容器配方範例是指本指南一使用建立元件 AWS CLI節中建立的範例元件。

建立元件或使用現有元件之後，請注意您要包含在配方中的 ARN。

1. 建立 CLI 輸入 JSON 文件

   您可以使用內嵌命令參數為命create-container-recipe令提供所有輸入。但是，生成的命令可能很長。若要簡化命令，您可以改為提供包含所有容器配方設定的 JSON 檔案

   > **Note**
   >
   > JSON 檔案中資料值的命名慣例遵循針對 Image Builder API 動作要求參數指定的模式。若要檢閱 API 命令要求參數，請參閱 EC2 Image Builder API 參考中的CreateContainerRecipe命令。
   > 若要將資料值提供為指令行參數，請參考《指AWS CLI 令參考》中指定的參數名稱。

   以下是此範例中參數的摘要：

   - 元件 (物件陣列，必要) — 包含`ComponentConfiguration`物件陣列。至少必須指定一個組建元件：

     > **Note**
     >
     > Image Builder 會依照您在方案中指定元件的順序來安裝元件。不過，CIS 強化元件一律會持續執行，以確保基準測試會針對您的輸出影像執行。

     - 元件目錄 (字串，必要) — 元件 ARN。

       > **Tip**
       >
       > 若要使用範例建立您自己的容器方案，請將範例 ARN 取代為您用於方案之元件的 ARN。其中包括每個項目的 AWS 區域、名稱和版本號碼。

- 參數 (物件陣列) — 包含ComponentParameter物件陣列。如果需要輸入參數，但未在組件中定義預設值，則必須提供值。如果缺少任何必要參數且未定義預設值，Image Builder 將不會建立配方版本。

  > **Important**
  >
  > 元件參數是純文字值，並已登入 AWS CloudTrail。我們建議您使用 AWS Secrets Manager 或 AWS Systems Manager 參數存放區來儲存您的秘密。如需有關 Secrets Manager 的詳細資訊，請參閱什麼是 Secrets Manager？ 在《AWS Secrets Manager 使用者指南》中。若要取得有關 AWS Systems Manager 參數存放區的更多資訊，請AWS Systems Manager 參閱AWS Systems Manager 使用指南中的參數存放區

  - name (字串，必要) — 要設定的元件參數名稱。
  - value (字串陣列，必要) — 包含用來設定具名元件參數值的字串陣列。如果為元件定義了預設值，且未提供其他值，則 AWS TOE 會使用預設值。
- 容器類型 (字串，必要) — 要建立的容器類型。有效值包括DOCKER。
- dockerfileTemplateData (字符串) -用於構建圖像的 Docker 文件模板，表示為內聯數據塊。
- name (字串，必要) — 容器配方的名稱。
- 說明 (字串) — 容器配方的說明。
- ParentImage (字串，必要) — 容器配方用來做為自訂影像基礎的影像。該值可以是基本映像 ARN 或 AMI ID。
- 平台影像 (字串) — 指定使用自訂基本映像時的作業系統平台。
- SemanticVersion <major>(字串，必要) — 以下列格式指定之容器配方的語意版本，每個位置都有數值，以表示特定版本：。 <minor>。 <patch>。例如，即改為 `1.0.0`。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。
- 標籤 (字串對應) — 附加至容器配方的標籤。
- 執行個體配置 (物件) — 可用來設定執行個體以建置和測試容器映像檔的一組選項。
  - 圖像 (字符串) -用作容器構建和測試實例的基本映像的 AMI ID。如果未指定此值，Image Builder 會使用適當的 Amazon ECS 最佳化 AMI 做為基礎映像。
  - blockDeviceMappings(物件陣列) — 定義要連接的區塊裝置，以便從image參數中指定的 Image Builder AMI 建置執行個體。
    - 裝置名稱 (字串) — 這些對應套用至的裝置。

    - ebs (物件) — 用於管理此對應的 Amazon EBS 特定組態。
      - deleteOnTermination(布林值) — 用於在關聯裝置終止時配置刪除。
      - 加密 (布林值) — 用於設定裝置加密。
      - 磁碟區大小 (整數) — 用於覆寫裝置的磁碟區大小。
      - 磁碟區類型 (字串) — 用於覆寫裝置的磁碟區類型。
- TargetRepository (物件，必要) — 如果管線執行的區域 (區域 1) 的管線發佈組態中沒有指定其他存放庫，則為容器映像檔的目標存放庫。
  - 儲存庫名稱 (字串，必要) — 儲存輸出容器映像檔的容器存放庫名稱。此名稱以儲存庫位置為字首。
  - 服務 (字串，必要) — 指定註冊此影像的服務。
- workingDirectory (字串) — 建置和測試工作流程期間使用的工作目錄。

```
{
 "components": [
   {
   "componentArn": "arn:aws:imagebuilder:us-east-1:123456789012:component/
helloworldal2/x.x.x"
   }
 ],
 "containerType": "DOCKER",
 "description": "My Linux Docker container image",
 "dockerfileTemplateData": "FROM
 {{{ imagebuilder:parentImage }}}\n{{{ imagebuilder:environments }}}\n{{{ imagebuilder:comp
 "name": "amazonlinux-container-recipe",
 "parentImage": "amazonlinux:latest",
 "platformOverride": "Linux",
 "semanticVersion": "1.0.2",
 "tags": {
   "sometag" : "Tag detail"
 },
 "instanceConfiguration": {
   "image": "ami-1234567890",
   "blockDeviceMappings": [
   {
   "deviceName": "/dev/xvda",
   "ebs": {
    "deleteOnTermination": true,
    "encrypted": false,
```

```
    "volumeSize": 8,
    "volumeType": "gp2"
    }
    }
   ]
 },
 "targetRepository": {
   "repositoryName": "myrepo",
   "service": "ECR"
 },
 "workingDirectory": "/tmp"
}
```

2. 建立食譜

   使用下列指令建立配方。在參數中提供您在上一個步驟中建立的 JSON 檔案名`--cli-input-json`稱：

   ```
   aws imagebuilder create-container-recipe --cli-input-json file://create-container-recipe.json
   ```

   > **Note**
   >
   > - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
   > - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

清除資源

若要避免非預期的費用，請務必清理您從本指南中的範例建立的資源和管線。如需有關在 Image Builder 中刪除資源的詳細資訊，請參閱刪除 EC2 Image Builder 資源。

管理 EC2 Image Builder 映像

使用映像產生器為 AMI 或容器映像建立映像資源之後，您可以使用 Image Builder 主控台、透過 Image Builder 器 API 或使用 AWS CLI. imagebuilder

> **Tip**
>
> 當您有多個相同類型的資源時，標記可協助您根據指派給該資源的標籤來識別特定資源。如需有關使用「Image Builder」指令標記資源的詳細資訊 AWS CLI，請參閱本指南一標籤資源節。

本節介紹如何列出、檢視和建立影像。如需有關影像工作流程及其管理方式的資訊，請參閱管理 EC2 Image Builder 映像的建置和測試工作流程。

目錄

列出映像檔和建置版本

在映像產生器主控台的 [映像] 頁面上，您可以看到您擁有的所有映 Image Builder 映像資源、與您共用且您有權存取的影像資源的清單。清單結果包含有關這些資源的一些重要詳細資料。

您也可以查看帳戶中具有待處理工作流程動作的所有影像。

目錄

列出圖片

本節說明您可以列出影像相關資訊的不同方式。

您可以使用下列其中一種方法來列出您有權存取的映 Image Builder 影像資源。如需 API 動作的相關資訊，請參閱 EC2 Image Builder API 參考ListImages中的。如需相關聯的 SDK 要求，請參閱相同頁面上的「另請參閱」連結。

目錄

在主控台中列出影像

若要在主控台中開啟 [映像清單] 頁面，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導航窗格中選擇圖像。

根據映像擁有權或待處理的工作流程動作，主控台中的 [映像] 頁面分為索引標籤。本節涵蓋前三個索引標籤，顯示您擁有或可存取的影像。

主控台索引標籤：我擁有

在「我擁有」索引標籤中，您可以使用下列篩選器來簡化影像清單結果。

- 您可以在搜尋列中搜尋全部或部分名稱。
- 您可以根據影像的作業系統平台 (視窗或 Linux) 篩選影像。
- 您可以根據影像產生的輸出類型 (AMI 或容器影像) 篩選影像。
- 您可以使用篩選來源尋找從具有 VMIE 的虛擬機器匯入的映像。

在篩選器控制項之後，[我擁有] 索引標籤會顯示您建立的 Image Builder 映像清單，以及所列資源的下列詳細資訊：

名稱/版本

Image Builder 影像資源名稱以配方名稱和建置的版本開頭。選取連結以查看所有相關映像組建版本。

類型

映像產生器為此影像資源 (AMI 或容器映像檔) 建立的輸出影像類型。

平台

影像資源版本的作業系統平台，例如「視窗」或「Linux」。

圖片來源

映像產生器用來建置此影像資源的基本映像的來源。這主要用於篩選從虛擬機器 (VMIE) 匯入映像的結果。

建立時間

影像產生器建立影像資源目前版本的日期和時間。

ARN

圖像資源的當前版本的 Amazon 資源名稱（ARN）。

主控台索引標籤：與我共用

在 [與我共用] 索引標籤中，您可以使用下列篩選器來簡化影像清單結果。

- 您可以在搜尋列中搜尋全部或部分名稱。
- 您可以根據影像的作業系統平台 (視窗或 Linux) 篩選影像。
- 您可以根據影像產生的輸出類型 (AMI 或容器影像) 篩選影像。
- 您可以使用篩選來源尋找從具有 VMIE 的虛擬機器匯入的映像。

在篩選器控制項之後，[與我共用] 索引標籤會顯示與您共用的 Image Builder 映像清單，其中包含所列資源的下列詳細資訊：

影像名稱

與您共用的影像資源名稱。若要在配方中使用共用映像檔，請選取「選取受管理的映像檔」選項，然後將映像來源變更為與我共享的影像。

類型

映像產生器為此影像資源 (AMI 或容器映像檔) 建立的輸出影像類型。

版本

影像資源版本的作業系統平台，例如「視窗」或「Linux」。

圖片來源

映像產生器用來建置此影像資源的基本映像的原點 (如果適用)。這主要用於篩選從虛擬機器 (VMIE) 匯入映像的結果。

平台

影像資源版本的作業系統平台，例如「視窗」或「Linux」。

建立時間

Image Builder 建立與您共用之影像資源版本的日期和時間。

擁有者

共用影像資源的擁有者。

ARN

與您共用的映像資源版本的 Amazon 資源名稱 (ARN)。

主控台標籤：由 Amazon 管理

在「由 Amazon 管理」索引標籤中，您可以使用下列篩選器來簡化影像清單結果。

- 您可以在搜尋列中搜尋全部或部分名稱。
- 您可以根據影像的作業系統平台 (視窗或 Linux) 篩選影像。
- 您可以根據影像產生的輸出類型 (AMI 或容器影像) 篩選影像。
- 您可以使用篩選來源尋找從具有 VMIE 的虛擬機器匯入的映像。

按照篩選器控制項，「由 Amazon 管理」索引標籤會顯示 Amazon 受管 Image Builder 清單，您可以將這些映像用作食譜的基本映像。Image Builder 會顯示所列資源的下列詳細資訊：

影像名稱

受管理映像的名稱。當您建立配方時，基本映像的預設值為快速入門 (Amazon 管理)。當您建立方案時，此標籤中所列的映像會填入與您為基礎映像選擇的作業系統平台相關聯的映像名稱清單。

類型

映像產生器為此影像資源 (AMI 或容器映像檔) 建立的輸出影像類型。

版本

影像資源版本的作業系統平台，例如「視窗」或「Linux」。

平台

影像資源版本的作業系統平台，例如「視窗」或「Linux」。

建立時間

Image Builder 建立與您共用之影像資源版本的日期和時間。

擁有者

Amazon 擁有受管映像。

ARN

與您共用的映像資源版本的 Amazon 資源名稱 (ARN)。

使用 AWS CLI 指令列出影像

當您在中執行list-images指令時 AWS CLI，您可以取得您擁有或有權存取的影像清單。

以下指令範例展示如何使用不含篩選條件的list-images指令來列出您擁有的所有 Image Builder 影像資源。

範例：列出所有影像

```
aws imagebuilder list-images
```

輸出：

```
{
 "requestId": "1abcd234-e567-8fa9-0123-4567b890cd12",
 "imageVersionList": [
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-name/1.0.0",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0",
    "platform": "Linux",
    "owner": "123456789012",
```

```
    "dateCreated": "2022-04-28T01:38:23.286Z"
  },
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-win/1.0.1",
    "name": "image-recipe-win",
    "type": "AMI",
    "version": "1.0.1",
    "platform": "Windows",
    "owner": "123456789012",
    "dateCreated": "2022-04-28T01:38:23.286Z"
  }
 ]
}
```

執行list-images命令時，您可以套用篩選器來簡化結果，如下列範例所示。若要取得有關如何篩選結果的更多資訊，請參閱《指令參考》中的 list 影像指AWS CLI 令。

範例：針對 Linux 映像檔進行篩選

```
aws imagebuilder list-images --filters name="platform",values="Linux"
```

輸出：

```
{
 "requestId": "1abcd234-e567-8fa9-0123-4567b890cd12",
 "imageVersionList": [
  {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-name/1.0.0",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0",
    "platform": "Linux",
    "owner": "123456789012",
    "dateCreated": "2022-04-28T01:38:23.286Z"
  }
 ]
}
```

列出等待動作的圖像

當您在影像工作流程中使用WaitForAction步驟動作時，它會暫停工作流程，直到您傳送訊號以繼續處理或工作流程失敗為止。如果您有需要在繼續之前執行的外部處理程序，則可以使用此步驟動作。然

後，您可以使SendWorkflowStepAction用將信號傳送至暫停的步驟RESUME或STOP。您也可以從主控台停止或繼續工作流程。

下列索引標籤顯示如何取得帳戶中所有影像資源的清單，其中包含目前暫停等待訊號恢復或停止的工作流程步驟。標籤涵蓋了控制台步驟和命 AWS CLI 令。

您也可以使用 API 或 SDK 來取得正在等待動作的工作流程步驟清單。如需 API 動作的相關資訊，請參閱 EC2 Image Builder API 參考ListWaitingWorkflowSteps中的。如需相關聯的 SDK 要求，請參閱相同頁面上的「另請參閱」連結。

Console

若要前往主控台中的 [等待動作] 索引標籤，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導航窗格中選擇圖像。這會開啟 [影像] 清單頁面。
3. 從清單頁面中選取 [等待動作] 索引標籤。
4. (選擇性) 若要停止或繼續步驟，請選取名稱旁的核取方塊，然後選擇 [停止步驟] 或 [繼續步驟]。您可以選取多個核取方塊，對所有選取的步驟執行相同的動作。

擱置中工作流程步驟

擱置步驟的工作流程詳細資訊包括下列項目：

- 映像名稱 — 具有擱置步驟的影像資源名稱。您可以選取名稱連結以顯示該影像的詳細資訊頁面。
- 擱置步驟名稱 — 正在等待動作的工作流程步驟名稱。
- 步驟執行 ID — 唯一識別工作流程步驟的執行階段實例。您可以選取連結的 ID，以顯示步驟的執行時期詳細資訊。
- 步驟啟動 — 工作流程步驟的執行時間執行個體啟動時的時間戳記。
- 工作流程 ARN — 具有擱置步驟的工作流程的 Amazon 資源名稱 (ARN)。
- 動作 — 處於等待狀態的步驟動作。

AWS CLI

當您在中執行list-waiting-workflow-steps命令時 AWS CLI，您會看到帳戶中所有映像檔的清單，這些映像具有在完成映像建立程序之前等待動作的工作流程步驟。

下列命令範例顯示如何使用list-waiting-workflow-steps命令，以等待動作的工作流程步驟列出帳戶中的所有影像。

示例：列出帳戶中的圖像，並等待工作流程步驟

```
aws imagebuilder list-waiting-workflow-steps
```

輸出：

此範例的輸出會顯示帳戶中的一個影像，其中包含正在等待動作的步驟。

```
{
    "steps": [
        {
            "imageBuildVersionArn": "arn:aws:imagebuilder:us-
west-2:111122223333:image/example-image/1.0.0/8",
            "name": "WaitForAction",
            "workflowExecutionId": "wf-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "stepExecutionId": "step-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "workflowBuildVersionArn": "arn:aws:imagebuilder:us-
west-2:111122223333:workflow/test/wait-for-action/1.0.0/1",
            "startTime": "2023-11-21T23:21:23.609Z",
            "action": "WaitForAction"
        }
    ]
}
```

列出映像檔建置版本

在 Image Builder 主控台的 [映像建置版本] 頁面上，您可以看到組建版本清單，以及您擁有之映像資源的其他詳細資料。您也可以 AWS CLI 將命令或動作與 Image Builder API、SDK 搭配使用，或列出映像建置版本。

您可以使用下列其中一種方法列出您擁有的映像資源的映像建置版本。如需 API 動作的相關資訊，請參閱 EC2 Image Builder API 參考ListImageBuildVersions中的。如需相關聯的 SDK 要求，請參閱相同頁面上的「另請參閱」連結。

Console

版本詳情

[映像 Image Builder 生器] 主控台中 [映像建置版本] 頁面上的詳細資料包括下列項目

- 版本 — 映像資源組建版本。在 Image Builder 主控台中，版本會連結至映像詳細資料頁面。
- 類型 — 映像產生器在建立此映像資源 (AMI 或容器映像) 時所散佈的輸出類型。
- 建立日期 — 映像產生器建立 Image Builder 立版本的日期和時間。
- 影像狀態 — 映像建置版本的目前狀態。狀態可與映像建置或處理方式相關。例如，在建置程序期間，您可能會看到`Building`或的狀態`Distributing`。對於映像的處理方式，您可能會看到`Deprecated`或`Deleted`的狀態。
- 失敗原因 — 影像狀態的原因。映 Image Builder 主控台只會顯示組建失敗時的原因 (映像狀態等於`Failed`)。
- 安全性發現項目 — 參照映像組建版本的彙總映像掃描發現項目。
- ARN — 圖像資源參考版本的 Amazon 資源名稱 (ARN)。
- 日誌流-引用映像構建版本的日誌流詳細信息的鏈接。

列表版本

若要在映像產生器主控台中列出映像建置版本，請執行下列步驟：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導航窗格中選擇圖像。依預設，影像清單會顯示您擁有的每個影像的目前版本。
3. 若要查看影像的所有版本清單，請選擇目前版本連結。此連結會開啟 [映像組建版本] 頁面，其中列出特定映像的所有組建版本。

AWS CLI

當您在中執行list-image-build-versions命令時 AWS CLI，您會取得指定映像資源的組建版本完整清單。您必須擁有該映像才能運行此命令。

下列命令範例會示範如何使用list-image-build-versions命令列出指定映像檔的所有組建版本。

範例：列出特定映像檔的建置版本

```
aws imagebuilder list-image-build-versions --image-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-name/1.0.0
```

輸出：

此範例的輸出包含指定映像方案的兩個建置版本。


```
{
  "requestId": "12f3e45d-67cb-8901-af23-45ed678c9b01",
  "imageSummaryList": [
    {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-
name/1.0.0/2",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0/2",
    "platform": "Linux",
    "osVersion": "Amazon Linux 2",
    "state": {
      "status": "AVAILABLE"
    },
    "owner": "123456789012",
    "dateCreated": "2023-03-10T01:04:40.609Z",
    "outputResources": {
      "amis": [
        {
        "region": "us-west-2",
        "image": "ami-012b3456789012c3d",
        "name": "image-recipe-name 2023-03-10T01-05-12.541Z",
        "description": "First verison of image-recipe-name",
        "accountId": "123456789012"
        }
      ]
    },
    "tags": {}
    },
    {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/image-recipe-
name/1.0.0/1",
    "name": "image-recipe-name",
    "type": "AMI",
    "version": "1.0.0/1",
    "platform": "Linux",
    "osVersion": "Amazon Linux 2",
    "state": {
      "status": "AVAILABLE"
    },
    "owner": "123456789012",
    "dateCreated": "2023-03-10T00:07:16.384Z",
```

```
    "outputResources": {
      "amis": [
        {
        "region": "us-west-2",
        "image": "ami-0d1e23456789f0a12",
        "name": "image-recipe-name 2023-03-10T00-07-18.146132Z",
        "description": "First verison of image-recipe-name",
        "accountId": "123456789012"
        }
      ]
    },
    "tags": {}
    }
  ]
}
```

> **Note**
>
> list-image-build-versions命令的輸出目前不包含安全性發現項目或記錄資料流。

檢視影像詳細資

在映像產生器主控台的映像詳細資料頁面上，您可以檢視您擁有之特定影像資源的詳細資料。您也可以將命令或動作與 Image Builder API、SDK 搭配使用，或取 AWS CLI 得影像詳細資料。

有關其他人透過 AWS Resource Access Manager (AWS RAM) 資源 AWS 帳戶 共用與您共用的資源的詳細資訊，請參閱《AWS RAM 使用指南》中的存取與您共用的 AWS 資源。

目錄

在映像產生器主控台中檢視影像詳細資

Image Builder 主控台中的影像詳細資料頁面包含摘要區段，其中包含分組成索引標籤的其他資訊。頁面標題是建立映像檔之配方的名稱和建置版本。

主控台詳細資料區段和標

- 摘要章節
- 輸出資源標籤
- 基礎結構組態標
- 發佈設定索引標
- 「工作流程
- 安全性發現項目
- 標籤索引標籤

摘要章節

摘要區段會橫跨頁面寬度，並包含下列詳細資訊。這些詳細信息始終顯示。

食譜

不包含構建版本的配方名稱和版本。例如，如果構建版本是`sample-linux-recipe | 1.0.1/2`，則配方為`sample-linux-recipe | 1.0.1`，並且構建版本為2。

Date created (建立日期)

映像產生器建立 Image Builder 置版本的日期和時間。

影像狀態

映像檔建置版本的目前狀態。狀態可與映像建置或處理方式相關。例如，在建置程序期間，您可能會看到`Building`或的狀態`Distributing`。對於映像的處理方式，您可能會看到`Deprecated`或`Deleted`的狀態。

失敗的原因

圖像狀態的原因。映 Image Builder 主控台只會顯示組建失敗時的原因 (映像狀態等於`Failed`)。

輸出資源標籤

[輸出資源] 索引標籤會列出目前顯示之影像資源的輸出和分佈詳細資訊。Image Builder 顯示的資訊取決於管線用於建立映像的配方類型，如下所示。

圖片食譜

- 區域 — 在「影像」欄中指定的輸出 Amazon 機器映像 (AMI) 的發佈區域。

- 影像 — 映 Image Builder 分配給目標的 AMI 的 ID。此識別碼會連結至 Amazon EC2 主控台中的亞馬遜機器映像 (AMI) 頁面。

> **Note**
>
> Image Builder 會在建立輸出影像資源之後，以及將 AMI 分發到目的地之前，建立 AMI。

- 名稱 — 映 Image Builder 分配給目的地的 AMI 的名稱。
- 描述 — 影像方案中用來建立輸出影像資源的管線的選擇性描述。
- 帳號 — 擁 AWS 帳戶 有目前顯示的 Image Builder 資源的帳號。

容器食譜

Image Builder 會針對從容器方案建立的輸出顯示下列詳細資訊。

- 區域 — 在 [映像 URI] 欄中指定之容器映像的發佈區域。
- 映像 URI — 映 Image Builder 分配至目標區域中 ECR 存放庫的輸出容器映像的 URI。

> **Note**
>
> Image Builder 會針對每個目的地顯示一個 輸出影像一律至少有一個項目，用於分發至建立映像的帳戶。其他目的地可以包括跨區域的分佈 AWS 帳戶、或 AWS Organizations。如需詳細資訊，請參閱 管理 EC2 Image Builder 分發設定。

基礎結構組態標

基礎設施組態索引標籤會顯示 Image Builder 用來建立和測試目前顯示的映像的 Amazon EC2 基礎設施設定。Image Builder 一律會顯示基礎設施組態資源的名稱 (組態名稱) 及其 Amazon 資源名稱 (ARN)。如果您的基礎結構組態設定值，則其他基礎結構詳細資料可能包括下列

- 執行個體類型
- 執行個體設定檔
- 網路基礎設
- 安全性群組設定
- Image Builder 存放應用程式日誌的 Amazon S3 位置

- 用於疑難排解的 Amazon EC2 key pair
- 適用於事件通知的 Amazon SNS 主題

如需詳細資訊，請參閱 管理 EC2 Image Builder 基礎設施組。

發佈設定索引標

「散佈設定」索引標籤會顯示 Image Builder 用來分發輸出影像的設定。Image Builder 一律會顯示分發組態資源的名稱 (組態名稱) 及其 Amazon 資源名稱 (ARN)。其他發佈詳細資料取決於 Image Builder 管線用於建立映像的方案類型，如下所示：

圖片食譜

如果您的配送組態資源設定了這些值，則額外的分配明細可能包括下列項目：

- 區域 — 輸出 Amazon 機器映像 (AMI) 的分佈區域。
- 輸出 AMI 名稱 — Image Builder 分配給目的地的 AMI 的名稱。
- 加密 (KMS 金鑰) — 如果已設定，映像產生器會使用 AWS KMS key 該映像來加密映像，以便分發到目標區域。
- 發佈的目標帳戶 — 如果您已設定跨帳戶分配，此欄會顯示逗號分隔清單， AWS 帳戶 以在目標區域中與其共用輸出影像。
- 具有共用權限的主參與者 — 具有啟動映像之權限的 AWS 主參與者清單 (例如， AWS Organizations 群組 AWS 帳戶 或組織單位 (OU)。

> **Note**
>
> 當您授與其他主體啟動映像的權限時，您仍然擁有該映像。 AWS 向您的帳戶收取 Amazon EC2 從映像啟動的所有執行個體的費用。

- 更快啟動配置的目標帳戶 —
- 關聯的授權配置 — 與指定區域中 AMI 相關聯的許可證 License Manager 授權配置 ARN。
- 啟動範本配置 —
- 設置啟動模板默認版本-

容器食譜

貨櫃分配一律包含下列詳細資訊：

- 區域 — 在 [映像 URI] 欄中指定之容器映像的發佈區域。
- 映像 URI — 映 Image Builder 分發到目標區域中 Amazon ECR 儲存庫的輸出容器映像的 URI。

> **Note**
>
> Image Builder 會針對每個目的地顯示一個 輸出影像一律至少有一個項目，用於分發至建立映像的帳戶。其他目的地可以包括跨區域的分佈 AWS 帳戶、或 AWS Organizations。如需詳細資訊，請參閱 管理 EC2 Image Builder 分發設定。

「工作流程

工作流程會定義 Image Builder 在建立新映像時執行的步驟順序。所有映像都有構建和測試工作流程。容器具有額外的散發工作流程。「工作流程」標籤會顯示 Image Builder 執行的適用工作流程。

篩選工作流類型

依預設，Image Builder 最初會顯示建置工作流程摘要和工作流程步驟 不過，「工作流程」篩選器會顯示影像正在進行中或已完成的所有工作流程。若要檢視其他工作流程，請從清單中選取，如下所示：

影像工作流程 (AMI 輸出)

- `build-image`
- `test-image`

容器工作流程 (容器輸出)

- `build-container`
- `test-container`
- `distribute-container`

> **Note**
>
> 如果工作流程尚未啟動，則不會顯示在清單中。例如，如果您的映像檔組建剛開始，就`build-image`是清單中出現的唯一工作流程類型。下一個工作流程開始時，`test-image`在這種情況下，Image Builder 會將其新增至清單。

在「工作流程」篩選器之後，選取的工作流程會顯示執行階段摘要，其中包含每個工作流程類型的下列

工作流狀態

此工作流程的目前執行階段狀態。值可以包括以下內容：

- 待定
- 略過
- 執行中
- 已完成
- 失敗
- R ollback-in-progress
- 倒回完成

執行識別碼

Image Builder 指派的唯一識別碼，以便在每次執行工作流程時追蹤執行階段資源。

Start (開始)

此工作流程的執行時間執行個體啟動時的時間戳記。

結束

此工作流程執行個體完成時的時間戳記。

總步數

工作流程中的總步驟數。這應該等於成功、略過和失敗的步驟計數總和。

步驟成功

工作流程中成功執行之步驟數目的執行階段計數。

步驟失敗

工作流程中失敗的步驟數目的執行階段計數。

跳過的步驟

工作流程中跳過的步驟數目的執行時期計數。

下列清單中的詳細資訊會報告此工作流程執行環境中所有步驟的目前狀態。Image Builder 會針對所有影像類型顯示相同的詳細資料

步驟 #

代表 Image Builder 執行工作流程步驟的順序的數字。

步驟識別碼

工作流程步驟的唯一識別元，在執行時間指派。

步驟狀態

指定工作流程步驟的目前執行時期狀態。

還原狀態

目前的復原狀態 (如果工作流程的執行階段執行個體失敗)。

步驟名稱

指定工作流程步驟的名稱。

Start (開始)

此工作流程執行個體的指定步驟啟動時的時間戳記。

結束

此工作流程執行個體的指定步驟完成時的時間戳記。

安全性發現項目

如果您已啟動掃描，安全性發現項目索引標籤會顯示常見弱點和入侵 (CVE) 發現項目。Amazon Inspector 會在映像產生器為建立新映像而啟動的測試執行個體上識別出這些發現項目。若要確保 Image Builder 可擷取映像的發現項目，您必須依下列方式設定掃描：

1. 為您的帳戶啟用 Amazon Inspector 掃描。如需詳細資訊，請參閱 Amazon Inspector 使用者指南中的開始使用 Amazon Inspector。
2. 針對建立此映像的管道啟動安全性發現項目。當您啟動管線的安全性發現項目時，Image Builder 會在終止測試執行個體之前儲存發現項目的快照。如需更多資訊，請參閱設定 Image Builder 檔的安全性掃描 AWS Management Console

「安全發現項目」索引標籤包含 Amazon Inspector 為您的映像識別之每個弱點的下列詳細資訊。

嚴重性

CVE 發現項目的嚴重性層級。相關值如下：

- 未分類
- 資訊
- 低
- 中
- 高
- 嚴重

問題清單 ID

Amazon Inspector 在掃描測試執行個體時針對映像偵測到的 CVE 發現項的唯一識別碼。ID 會連結至「安全性發現項目 > 依弱點」頁面。如需詳細資訊，請參閱 管理 Image Builder 的安全性發現項目 AWS Management Console。

來源

CVE 發現項目的弱點資訊來源。

年齡

自從您的影像第一次觀察到發現項目以來的天數。

Inspector 得分

Amazon Inspector 分配給 CVE 發現的分數。

標籤索引標籤

「標籤」索引標籤會顯示您為影像定義的任何標籤。

取得影像政策詳細資料 (AWS CLI)

下列範例顯示如何使用其 Amazon 資源名稱 (ARN) 取得映像政策的詳細資料。

```
aws imagebuilder get-image-policy --image-arn arn:aws:imagebuilder:us-
west-2:123456789012:image/example-image/2019.12.02
```

建立影像

本節說明如何建立 Image Builder，以及如何取消進行中的組建。

目錄

- 建立映像
- 取消影像建立 (AWS CLI)

建立映像

有幾種不同的方法可以建立新的 Image Builder 映像。例如，您可以使用下列其中一種方法建立具有 AWS Management Console 或的影像 AWS CLI。您也可以使用 CreateImageAPI 動作。對於相關的 SDK 請求，您可以參閱 EC2 Image Builder API 參考中該命令的「另請參閱」連結。

AWS Management Console

若要從現有管線建立新映像，您可以手動執行管線，如下所示。您也可以使用管線精靈從頭開始建立新映像。請參閱建立映像管線 (AMI)或建立映像管線 (泊塢視窗)，視您要建立的影像類型而定。

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導覽窗格中選擇映像管線。
3. 選取您要執行的配管名稱旁邊的核取方塊。
4. 若要建立映像，請從「動作」功能表中選取「執行管線」。這將啟動管道。

您也可以指定執行管道的排程，或使 EventBridge 用 Amazon 根據您設定的規則執行管道。

AWS CLI

在中執行create-image命令之前 AWS CLI，您必須先建立下列資源 (如果這些資源尚未存在)：

必要的資源

- 配方 — 您必須為圖像指定一個配方，如下所示：

  圖片食譜

  使用`--image-recipe-arn`參數指定映像配方資源的 Amazon 資源名稱 (ARN)。

  容器食譜

  使用`--container-recipe-arn`參數指定容器配方資源的 ARN。
- 基礎結構組態 — 使用`--infrastructure-configuration-arn`參數指定基礎結構組態資源的 ARN。

您也可以指定影像所需的下列任何資源：

可選資源和配置

- 發佈組態 — 依預設，Image Builder 會將輸出影像資源分配到您在執行create-image命令的區域中的帳戶。若要為您的發佈提供其他目的地或組態，請使用--distribution-configuration-arn參數指定發佈組態資源的 ARN。
- 映像掃描 — 若要為映像或容器測試執行個體上的 Amazon Inspector 發現項目設定快照，請使用--image-scanning-configuration參數。對於容器映像檔，您也可以指定 Amazon Inspector 用於掃描的 ECR 儲存庫。
- 影像測試 — 若要隱藏「Image Builder」測試階段，請使用--image-tests-configuration參數。或者，您可以設置它可以運行多長時間的超時。
- 影像標籤 — 使用--tags參數將標籤新增至輸出影像。
- 圖像工作流程 — 如果您沒有指定任何構建或測試工作流程，則 Image Builder 會使用其默認圖像工作流程來創建圖像。若要指定您已建立的工作流程，請使用--workflows參數。

> **Note**
>
> 如果您指定映像工作流程，則還必須提供 Image Builder 用來在--execution-role參數中執行工作流程動作的 IAM 角色的名稱或 ARN。

下列範例會示範如何使用「建立影像」指令建立 AWS CLI 影像。如需詳細資訊，請參閱 AWS CLI 命令參考。

範例：使用預設分佈建立基本映像

```
aws imagebuilder create-image --image-recipe-arn arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/simple-recipe-linux/1.0.0 --infrastructure-configuration-arn arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/simple-infra-config-linux
```

輸出：

```
{
    "requestId": "1abcd234-e567-8fa9-0123-4567b890cd12",
    "imageVersionList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/simple-recipe-linux/1.0.0",
            "name": "simple-recipe-linux",
```

```
            ...
        }
    ]
}
```

取消影像建立 (AWS CLI)

若要取消進行中的映像組建，請使用cancel-image-creation指令，執行方式如下：

```
aws imagebuilder cancel-image-creation --image-build-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-recipe/2019.12.03/1
```

匯入虛擬機器映像

Image Builder 與 Amazon EC2 虛擬機器匯入/匯出 API 整合，可讓匯入程序在背景中以非同步方式執行。Image Builder 會參考虛擬機器匯入中的工作 ID 以追蹤其進度，並建立 Image Builder 映像資源作為輸出。這可讓您在虛擬機器匯入完成之前，在方案中參考映像產生器映像資源。

匯入虛擬機器 (主控台)

若要使用映像產生器主控台匯入虛擬機器，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導航窗格中選擇圖像。
3. 選擇 Import package (匯入影像)。
4. 在 [匯入映像] 頁面上提供下列段落的詳細資訊。完成後，選擇「導入圖像」。

一般

1. 為基礎映像指定唯一的「名稱」。
2. 指定基本映像的版本。使用下列格式：*major.minor.patch*。
3. 您也可以為基本影像輸入選擇性的「描述」。

基本映像作業系統

1. 選取與您的 VM 作業系統平台相符的映像作業系統 (OS) 選項。
2. 從清單中選取與虛擬機器版本相符的作業系統版本。

VM 匯入組態

當您從虛擬化環境匯出 VM 時，該程序會建立一組或多個磁碟容器檔案。這些可做為 VM 環境、設定和資料的快照。您可以使用這些檔案將虛擬機器匯入為映像配方的基本映像檔。如需在映像產生器中匯入虛擬機器的詳細資訊，請參閱匯入和匯出 VM 映像檔。

若要指定匯入來源的位置，請依照下列步驟執行：

匯入來源

在 [磁碟容器 1] 區段中指定要匯入的第一個虛擬機器映像磁碟容器或快照的來源。

1. 來源 — 這可以是 S3 儲存貯體，也可以是 EBS 快照。
2. 選取磁碟 S3 位置 — 輸入 Amazon S3 中存放磁碟映像的位置。若要瀏覽位置，請選擇 [瀏覽 S3]。
3. 若要新增磁碟容器，請選擇 [新增磁碟容器]。

IAM 角色

若要將 IAM 角色與 VM 匯入設定建立關聯，請從 IAM 角色下拉式清單中選取角色，或選擇 [建立新角色] 以建立新角色。如果您建立新角色，IAM 角色主控台頁面會在單獨的索引標籤中開啟。

進階設定 — 選用

以下是可選的設定。使用這些設定，您可以為匯入所建立的基礎映像配置加密、授權、標籤等。

基本映像架構

若要指定虛擬機器匯入來源的架構，請從架構清單中選取一個值。

加密

如果您的虛擬機器磁碟映像檔已加密，您必須提供用於匯入程序的金鑰。若要指定用於匯入的 KMS 金鑰，請從加密 (KMS 金鑰) 清單中選取一個值。此清單包含您的帳戶在目前區域中可存取的 KMS 金鑰。

授權管理

匯入虛擬機器時，匯入程序會自動偵測虛擬機器作業系統，並將適當的授權套用至基礎映像。視您的作業系統平台而定，授權類型如下：

- 已包含授權 — 將適用於您平台的適當 AWS 授權套用至您的基礎映像。

- 攜帶您自己的授權 (BYOL) — 保留來自虛擬機器的授權 (如果適用)。

若要將使用建立的授權規劃附加 AWS License Manager 至基礎映像，請從 [授權規劃名稱] 清單中選取。若要取得有關 License Manager 的更多資訊，請參閱〈 AWS License Manager

> **Note**
>
> - 授權組態包含以企業合約條款為基礎的授權規則。
> - Linux 僅支援自攜授權。

標籤 (基本影像)

標籤使用鍵值配對，將可搜尋的文字指派給您的 Image Builder 資源。若要為匯入的基本影像指定標籤，請使用「鍵」和「值」方塊輸入鍵值配對。

若要新增標籤，請選擇 Add tag (新增標籤)。若要移除標籤，請選擇 Remove tag (移除標籤)。

匯入虛擬機器 (AWS CLI)

若要將虛擬機器從磁碟匯入 AMI，並建立可立即參考的映 Image Builder 映像資源，請依照下列步驟執行 AWS CLI：

1. 使用中的 Amazon EC2 VM Import/匯出import-image命令，啟動虛擬機器匯入 AWS CLI。記下命令回應中傳回的工作 ID。您將需要它來進行下一步。如需詳細資訊，請參閱《虛擬機器匯入/匯出使用者指南》中的使用虛擬機器匯入匯入為映像。
2. 建立 CLI 輸入 JSON 文件

   為了簡化中使用的 Image Builder import-vm-image 命令 AWS CLI，我們建立了一個 JSON 檔案，其中包含要傳入命令的所有匯入設定。

   > **Note**
   >
   > JSON 檔案中資料值的命名慣例遵循針對 Image Builder API 動作要求參數指定的模式。若要檢閱 API 命令要求參數，請參閱 EC2 Image Builder API 參考中的ImportVmImage命令。
   > 若要提供資料值做為指令行參數，請參考《指AWS CLI 令參考》中指定的參數名稱。Image Builder 指import-vm-image令做為選項。

以下是我們在此示例中指定的參數摘要：

- name (字串，必要) — 要建立為匯入輸出的影 Image Builder 影像資源的名稱。
- SemanticVersion <major>(字串，必要) — 輸出影像的語意版本，以下列格式指定版本，每個位置都有數值，以表示特定版本：。 <minor>。 <patch>。例如 `1.0.0`。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。
- 描述（字符串）— 圖像配方的描述。
- 平台 (字串，必要) — 已匯入虛擬機器的作業系統平台。
- vmImportTask識別碼 (字串，必要) — 來自 Amazon EC2 虛擬機器匯入程序的 `ImportTaskId` (AWS CLI)。Image Builder 會監控匯入程序，以便在建立並建立 Image Builder 資源的 AMI 中，以便立即用於配方中。
- ClientToken（字符串，必需）-您提供的唯一，區分大小寫的標識符，以確保請求的冪等性。如需詳細資訊，請參閱 Amazon EC2 API 參考中的確保冪等性。
- tags (字串對映) — 標籤是附加至匯入資源的鍵值配對。最多允許 50 個鍵值對。

將檔案另存為`import-vm-image.json`，以在「Image Builder」import-vm-image 指令中使用。

```
{
    "name": "example-request",
    "semanticVersion": "1.0.0",
    "description": "vm-import-test",
    "platform": "Linux",
    "vmImportTaskId": "import-ami-01ab234567890cd1e",
    "clientToken": "asz1231231234cs3z",
    "tags": {
     "Usage": "VMIE"
    }
}
```

3. 匯入影像

運行命import-vm-image令，並使用您創建的文件作為輸入：

```
aws imagebuilder import-vm-image --cli-input-json file://import-vm-image.json
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

管理 Image Builder 的安全性發現

當您使用 Amazon Inspector 啟用安全掃描時，它會持續掃描您帳戶中的機器映像和執行個體，以防範作業系統和程式設計語言漏洞。如果啟動，安全性掃描會自動執行，而 Image Builder 可以在您建立新映像時，從測試執行個體儲存發現項目的快照。Amazon Inspector 是一項付費服務。

當 Amazon Inspector 發現軟體或網路設定中的弱點時，會採取下列動作：

- 通知你有一個發現。
- 評估發現項目的嚴重性。嚴重性等級會將弱點分類，以協助您排定發現項目的優先順序，並包含下列值：
  - 未分類
  - 資訊
  - 低
  - 中
  - 高
  - 嚴重
- 提供有關發現項目的資訊，以及其他資源的連結以取得更多詳細資訊。
- 提供補救指引，協助您解決產生發現項目的問題。

設定 Image Builder 檔的安全性掃描 AWS Management Console

如果您已為帳戶啟用 Amazon Inspector，Amazon Inspector 會自動掃描 Image Builder 啟動的 EC2 執行個體，以建立和測試新映像。這些執行個體在建置和測試程序期間的使用壽命很短，而且一旦這些執行個體關閉，它們的發現通常就會過期。為了協助您調查和修復新映像的發現項目，映 Image Builder 可以選擇性地將 Amazon Inspector 在建置程序期間在測試執行個體上識別的任何發現項目儲存為快照。

步驟 1：為您的帳戶啟用 Amazon Inspector 安全掃描

若要從 Image Builder 主控台為您的帳戶啟用 Amazon Inspector 安全掃描，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導覽窗格中選擇 [安全性掃描設定]。這會開啟 [安全性掃描] 對話方塊。

   對話方塊會顯示您帳戶的掃描狀態。如果您的帳戶已經啟用 Amazon Inspector，狀態會顯示為「已啟用」。
3. 按照說明的步驟 1 和 2 激活 Amazon Inspector 掃描。

> **Note**
>
> Amazon Inspector 會收取費用。如需詳細資訊，請參閱 Amazon Inspector 定價。

如果您已針對管線啟動掃描，Image Builder 會在您建立新映像時，為您的組建執行個體擷取發現項目的快照。如此一來，您就可以在 Image Builder 終止組建執行個體之後存取發現項目。

步驟 2：設定管道以儲存發現漏洞的快照

若要為您的管道設定弱點尋找快照，請執行下列步驟：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導覽窗格中選擇 [映像管線]。
3. 選取下列其中一種方法來指定配管詳細資訊：

   建立新管線

   1. 在 [映像管線] 頁面中，選擇 [建立映像管線]。這會開啟配管精靈中的「指定配管詳細資訊」頁面。

   更新現有管線

   1. 從 [映像管線] 頁面中，選擇您要更新之管線的管線名稱連結。這會開啟配管詳細資訊頁面。

> **Note**
>
> 或者，您可以選取要更新之管線名稱旁邊的核取方塊，然後選擇 [檢視詳細資訊]。

2. 從配管詳細資訊頁面中，從「動作」功能表中選取「編輯配管」。這會帶您前往 「編輯配管」頁面。

4. 在管線精靈或編輯管線頁面的 [一般] 區段中，選取 [啟用安全性掃描] 核取方塊。

> **Note**
>
> 如果您想要稍後關閉快照，可以編輯管線以清除該核取方塊。這不會停用您帳戶的 Amazon Inspector 掃描。要停用 Amazon Inspector 掃描，請參閱 Amazon Inspector 查器用戶指南中的停用 Amazon Inspector 查器。

管理 Image Builder 的安全性發現項目 AWS Management Console

「安全性發現項目」清單頁面會顯示有關資源發現項目的高階資訊，並以您可以套用的數個不同篩選為基礎的檢視。每個視圖頂部都包括以下選項，可用於更改視圖：

- 所有安全性發現項目 — 如果您從 Image Builder 主控台的導覽窗格中選擇 [安全性發現項目] 頁面，則此為預設檢視。
- 依弱點 — 此檢視會顯示您帳戶中具有發現項目之所有影像資源的高階清單。發現項目 ID 會連結至有關發現項目的更詳細資訊。此資訊會顯示在頁面右側開啟的面板上。此面板包含下列資訊：
  - 發現項目的詳細描述。
  - 「搜尋結果明細」頁標。此索引標籤包含發現項目概觀、受影響的套件、摘要修正建議、弱點詳細資訊和相關弱點。弱點 ID 連結至國家弱點資料庫中的詳細弱點資訊。
  - 分數劃分標籤。此索引標籤包含 CVSS 和亞馬遜檢查器分數的side-by-side 比較，以便您可以查看 Amazon Inspector 修改分數的位置 (如果適用)。
- 依映像管線 — 此檢視會顯示帳戶中每個映像管道的發現項目數目。Image Builder 會顯示中等嚴重性和較高發現項目的計數，以及所有發現項目的總計。列表中的所有數據都被鏈接，如下所示：
  - 「影像管線名稱」欄會連結至指定影像管線的詳細資訊頁面。
  - 嚴重性層級欄連結會開啟 [所有安全性發現項目] 檢視，並依相關的映像管線名稱和嚴重性層級進行篩選。

  您也可以使用搜尋條件來縮小結果。
- 依影像 — 此檢視會顯示您帳戶中每個映像組建的發現項目數量。Image Builder 會顯示中等嚴重性和較高發現項目的計數，以及所有發現項目的總計。列表中的所有數據都被鏈接，如下所示：
  - [映像名稱] 欄會連結至指定映像組建的映像詳細資料頁面。如需詳細資訊，請參閱 檢視影像詳細資。

- 嚴重性層級欄連結會開啟 [所有安全性發現項目] 檢視，並依關聯的映像組建名稱和嚴重性層級進行篩選。

您也可以使用搜尋條件來縮小結果。

Image Builder 在預設「所有安全發現項目」檢視的「發現項目清單」區段中顯示下列詳細資

嚴重性

CVE 發現項目的嚴重性層級。相關值如下：

- 未分類
- 資訊
- 低
- 中
- 高
- 嚴重

問題清單 ID

Amazon Inspector 在掃描建置執行個體時針對映像偵測到的 CVE 發現的唯一識別碼。ID 會連結至「安全性發現項目 > 依弱點」頁面。

圖像 ARN

具有在「發現項目 ID」欄中指定的發現項目的映像的 Amazon 資源名稱 (ARN)。

管道

建置在「影像 ARN」資料行中指定之映像的管線。

Description

發現項目的簡短描述。

Inspector 得分

Amazon Inspector 分配給 CVE 發現的分數。

補救

連結至有關修正發現項目之建議動作方式的詳細資訊。

出版日期

此弱點首次新增至廠商資料庫的日期和時間。

清除資源

若要避免非預期的費用，請務必清理您從本指南中的範例建立的資源和管線。如需有關在 Image Builder 中刪除資源的詳細資訊，請參閱刪除 EC2 Image Builder 資源。

管理 EC2 Image Builder 基礎設施組

您可以使用基礎設施組態來指定 Image Builder 用來建立和測試 EC2 Image Builder 映像的 Amazon EC2 基礎設施。基礎結構設定包括：

- 建置和測試基礎結構的執行個體類型。建議您指定多個執行個體類型，因為這可讓 Image Builder 從具有足夠容量的集區啟動執行個體。這可以減少暫時性的建置失敗。
- 執行個體設定檔，可為您的組建和測試執行個體提供執行自訂活動所需的權限。例如，如果您有一個從 Amazon S3 擷取資源的元件，則執行個體設定檔需要存取這些檔案的許可。執行個體設定檔還需要一組最低限度的許可，EC2 Image Builder 才能成功與執行個體通訊。如需詳細資訊，請參閱 必要條件。
- 管道建置和測試執行個體的 VPC、子網路和安全群組。
- Image Builder 從您的建置和測試存放應用程式日誌的 Amazon S3 位置。如果您設定記錄，則基礎結構設定中指定的執行個體設定檔必須具有目標儲存貯體 (`arn:aws:s3:::BucketName/*`) 的`s3:PutObject`權限。
- Amazon EC2 key pair，可讓您登入執行個體，以便在建置失敗且設定為時進行疑難排`terminateInstanceOnFailure`解`false`。
- Image Builder 傳送事件通知的 SNS 主題。如需 Image Builder 如何與 Amazon SNS 整合的詳細資訊，請參閱Image Builder 中的 Amazon SNS 集成。

> **Note**
>
> 如果您的 SNS 主題已加密，則加密此主題的金鑰必須位於執行 Image Builder 服務的帳戶中。Image Builder 無法傳送通知給使用其他帳戶金鑰加密的 SNS 主題。

您可以使用 Image Builder 主控台、透過 Image Builder API 或使用 AWS CLI. imagebuilder

目錄

> **Tip**
>
> 當您有多個相同類型的資源時，標記可協助您根據指派給該資源的標籤來識別特定資源。如需有關使用「Image Builder」指令標記資源的詳細資訊 AWS CLI，請參閱本指南一標籤資源節。

列出並檢視基礎結構組態詳細

本節說明您可以尋找 EC2 Image Builder 基礎設施組態的資訊和檢視詳細資料的各種方式。

基礎結構組態詳

列出基礎架構組態 (AWS CLI)

下列範例顯示如何使用中的list-infrastructure-configurations命令列出所有基礎結構組態 AWS CLI。

```
aws imagebuilder list-infrastructure-configurations
```

取得基礎結構組態詳細資料AWS CLI(

下列範例顯示如何使用中的get-infrastructure-configuration命令，透過指定其 Amazon 資源名稱 (ARN) AWS CLI 來取得基礎設施組態的詳細資料。

```
aws imagebuilder get-infrastructure-configuration --infrastructure-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-
infrastructure-configuration
```

建立基礎架構組態

本節說明如何使用 Image Builder 主控台或中的imagebuilder命令 AWS CLI 來建立基礎結構組態。

Console

若要從 Image Builder 主控台建立基礎結構組態資源，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 在瀏覽窗格中，選擇 [基礎結構設定]。
3. 選擇建立基礎結構組態。
4. 在「一般」段落中，輸入下列必要資訊：
   - 輸入基礎結構組態資源的名稱。
   - 針對建置和測試執行個體上的元件許可，選取您要與執行個體設定檔建立關聯的 IAM 角色。Image Builder 會使用這些權限來下載和執行元件、將記錄檔上傳至 CloudWatch，以及執行方案中元件指定的任何其他動作。
5. 在AWS 基礎結構面板中，您可以設定可用的剩餘基礎結構設定。輸入下列必要資訊：
   - 執行個體類型 — 您可以指定要用於此組建的一或多個執行個體類型。服務會根據可用性挑選其中一種執行個體類型。
   - SNS 主題 (選用) — 選取要接收來自 EC2 Image Builder 的通知和警示的 SNS 主題。

   如果您未提供下列設定的值，則這些設定會在適用情況下使用服務特定的預設值。
   - VPC、子網路和安全群組 — Image Builder 使用您的預設 VPC 和子網路。如需設定 VPC 介面端點的詳細資訊，請參閱EC2 Image Builder 和介面 VPC 端點 ()AWS PrivateLink。
   - 在疑難排解設定區段中，您可以設定下列值：
     - 依預設，會選取「失敗時終止執行處理」核取方塊。不過，當組建失敗時，您可以登入 EC2 執行個體進行疑難排解。如果您希望執行個體在建置失敗後繼續執行，請清除核取方塊。
     - key pair — 如果 EC2 執行個體在建置失敗後繼續執行，您可以建立 key pair 組或使用現有金鑰組登入執行個體並進行疑難排解。
     - 日誌 — 您可以指定 S3 儲存貯體，Image Builder 可在其中撰寫應用程式日誌，以協助疑難排解您的建置和測試。如果您未指定 S3 儲存貯體，Image Builder 會將應用程式日誌寫入執行個體。

- 在執行個體中繼資料設定區段中，您可以設定下列值，以套用至 Image Builder 用來建立和測試映像的 EC2 執行個體：
  - 選取中繼資料版本，以判斷 EC2 是否需要簽署的權杖標頭來執行個體中繼資料擷取要求。
    - V1 和 V2（令牌可選）-默認值，如果你沒有選擇任何東西。
    - V2（需要令牌）

    > **Note**
    >
    > 我們建議您將 Image Builder 從管線組建啟動的所有 EC2 執行個體設定為使用 IMDSv2，以便執行個體中繼資料擷取請求需要已簽署的權杖標頭。

  - 中繼資料權杖回應躍點限制 — 中繼資料權杖可傳遞的網路躍點數目。最小躍點：1，最大躍點：64，預設為一個躍點。

6. 在基礎設施標籤區段 (選用) 中，您可以將中繼資料標籤指派給 Image Builder 在建置程序期間啟動的 Amazon EC2 執行個體。標籤作為鍵值對輸入。
7. 在 [標記] 區段 (選用) 中，您可以將中繼資料標記指派給 Image Builder 建立為輸出的基礎結構組態資源。標籤作為鍵值對輸入。

AWS CLI

下列範例顯示如何使用中的映像產生器create-infrastructure-configuration命令來設定映像的基礎結構 AWS CLI。

1. 建立 CLI 輸入 JSON 文件

   此基礎結構組態範例會指定兩種執行個體類型，`m5.large`以及 `m5.xlarge` 建議您指定多個執行個體類型，因為這可讓 Image Builder 從具有足夠容量的集區啟動執行個體。這可以減少暫時性的建置失敗。

   `instanceProfileName`指定執行個體設定檔，為執行個體提供設定檔執行自訂活動所需的權限。例如，如果您有一個從 Amazon S3 擷取資源的元件，則執行個體設定檔需要存取這些檔案的許可。執行個體設定檔還需要一組最低限度的許可，EC2 Image Builder 才能成功與執行個體通訊。如需詳細資訊，請參閱 必要條件。

   使用檔案編輯工具建立 JSON 檔案，其中包含下列範例所示的金鑰，以及適用於您環境的值。此範例使用名為 `create-infrastructure-configuration.json` 的檔案：

```
{
    "name": "MyExampleInfrastructure",
    "description": "An example that will retain instances of failed builds",
    "instanceTypes": [
        "m5.large", "m5.xlarge"
    ],
    "instanceProfileName": "myIAMInstanceProfileName",
    "securityGroupIds": [
        "sg-12345678"
    ],
    "subnetId": "sub-12345678",
    "logging": {
        "s3Logs": {
            "s3BucketName": "my-logging-bucket",
            "s3KeyPrefix": "my-path"
        }
    },
    "keyPair": "myKeyPairName",
    "terminateInstanceOnFailure": false,
    "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:MyTopic"
}
```

2. 當您執行下列命令時，請使用您建立的檔案做為輸入。

```
aws imagebuilder create-infrastructure-configuration --cli-input-json
 file://create-infrastructure-configuration.json
```

更新基礎架構組態

本節介紹如何使用 Image Builder 主控台或中的imagebuilder命令 AWS CLI 來更新基礎結構組態資源。要跟踪您的資源，您可以按如下方式應用標籤。標籤作為鍵值對輸入。

- 資源標籤會將中繼資料標籤指派給 Image Builder 在建置程序期間啟動的 Amazon EC2 執行個體。
- 標籤會將中繼資料標記指派給 Image Builder 建立為輸出的基礎結構組態資源。

Console

您可以從 Image Builder 主控台編輯下列基礎結構組態詳細資料：

- 基礎結構組態的說明。

- 要與執行個體設定檔建立關聯的 IAM 角色。
- AWS 基礎結構，包括用於通知的執行個體類型和 SNS 主題。
- VPC、子網路和安全群組。
- 疑難排解設定，包括失敗時終止執行個體、用於連線的金鑰配對，以及執行個體日誌的選用 S3 儲存貯體位置。

若要從 Image Builder 主控台更新基礎結構組態資源，請依照下列步驟執行：

選擇現有 Image Builder 基礎結構組態

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 若要查看您帳戶下的基礎結構組態資源清單，請從導覽窗格中選擇 [基礎結構設定]。
3. 若要檢視詳細資料或編輯基礎結構組態，請選擇「組態名稱」連結。這會開啟基礎結構組態的詳細資料檢視。

   > **Note**
   >
   > 您也可以選取「模型組態」名稱旁的核取方塊，然後選擇「檢視詳細資料」。

4. 從「基礎結構詳細資料」面板的右上角，選擇「編輯」。
5. 當您準備好儲存對基礎結構設定所做的更新時，請選擇 [儲存變更]。

AWS CLI

下列範例顯示如何使用中的映像產生器update-infrastructure-configuration命令更新映像的基礎結構組態 AWS CLI。

1. 建立 CLI 輸入 JSON 文件

   此基礎結構組態範例使用與建立範例相同的設定，但我們已將`terminateInstanceOnFailure`設定更新為`false`。執行update-infrastructure-configuration命令之後，使用此基礎結構設定的管線會在建置失敗時終止建置和測試執行個體。

   使用檔案編輯工具建立 JSON 檔案，其中包含下列範例所示的金鑰，以及適用於您環境的值。此範例使用名為 `update-infrastructure-configuration.json` 的檔案：

```
{
"infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
"description": "An example that will terminate instances of failed builds",
"instanceTypes": [
    "m5.large", "m5.2xlarge"
],
"instanceProfileName": "myIAMInstanceProfileName",
"securityGroupIds": [
    "sg-12345678"
],
"subnetId": "sub-12345678",
"logging": {
    "s3Logs": {
        "s3BucketName": "my-logging-bucket",
        "s3KeyPrefix": "my-path"
    }
},
"terminateInstanceOnFailure": true,
"snsTopicArn": "arn:aws:sns:us-west-2:123456789012:MyTopic"
}
```

2. 當您執行下列命令時，請使用您建立的檔案做為輸入。

```
aws imagebuilder update-infrastructure-configuration --cli-input-json
 file://update-infrastructure-configuration.json
```

EC2 Image Builder 和介面 VPC 端點 ()AWS PrivateLink

您可以建立介面 VPC 端點，在 VPC 和 EC2 Image Builder 之間建立私有連線。介面端點採用這項技術 AWS PrivateLink，可讓您在沒有網際網路閘道、NAT 裝置、VPN 連線或 AWS Direct Connect 連線的情況下私密存取 Image Builder API。VPC 中的執行個體不需要公用 IP 位址即可與 Image Builder API 通訊。您的 VPC 和 Image Builder 之間的流量不會離開 Amazon 網路。

每個介面端點都是由您子網路中的一或多個彈性網路介面表示。建立新映像時，您可以在基礎結構組態中指定 VPC 子網路 ID。

> **Note**
>
> 您從 VPC 中存取的每個服務都有自己的介面端點，並具有自己的端點策略。Image Builder 會下載 AWS TOE 元件管理員應用程式，並從 S3 儲存貯體存取受管資源以建立自訂映像。若要授與這些儲存貯體的存取權，您必須更新 S3 端點政策以允許它。如需詳細資訊，請參閱 S3 儲存貯體存取的自訂政策。

如需 VPC 端點的詳細資訊，請參閱《Amazon VPC 使用者指南》中的介面 VPC 端點 (AWS PrivateLink)。

Image Builder VPC 端點的考量事項

在為 Image Builder 設定介面 VPC 端點之前，請務必先檢閱 Amazon VPC 使用者指南中的介面端點屬性和限制。

Image Builder 支援從您的 VPC 呼叫其所有 API 動作。

為 Image Builder 建立介面 VPC 端點

若要為 Image Builder 服務建立 VPC 端點，您可以使用 Amazon VPC 主控台或 AWS Command Line Interface ()AWS CLI。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的建立介面端點。

使用下列服務名稱為 Image Builder 建立 VPC 端點：

- com.amazonaws.*region*.imagebuilder

如果您為端點啟用私有 DNS，則可以使用 Image Builder 的區域預設 DNS 名稱向映像產生器發出 API 要求，例如：`imagebuilder.us-east-1.amazonaws.com`。若要查詢適用於目標區域 EC2 Image Builder 端點，請參閱 Amazon Web Services 一般參考.

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的透過介面端點存取服務。

為 Image Builder 建立 VPC 端點原則

您可以將端點策略附加到 VPC 端點，以控制對 Image Builder 的存取。此政策會指定下列資訊：

- 可執行動作的主體。
- 可執行的動作。
- 可供執行動作的資源。

如果您在方案中使用 Amazon 管理的元件，Image Builder 的 VPC 端點必須允許存取下列服務擁有的元件程式庫：

```
arn:aws:imagebuilder:region:aws:component/*
```

> **Important**
>
> 將非預設政策套用至 EC2 Image Builder 的介面 VPC 端點時，某些失敗的 API 請求 (例如失敗者) 可能不會記錄到 AWS CloudTrail 或 Amazon。RequestLimitExceeded CloudWatch

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用 VPC 端點控制對服務的存取。

S3 儲存貯體存取的自訂政策

Image Builder 使用公開可用的 S3 儲存貯體來存放和存取受管資源，例如元件。它也會從個別的 S3 儲存貯體下載 AWS TOE 元件管理應用程式。如果您在環境中使用適用於 Amazon S3 的 VPC 端點，則需要確保 S3 VPC 端點政策允許 Image Builder 存取以下 S3 儲存貯體。值區名稱在每個 AWS 區域（地#）和應用程序環境（環#）都是唯一的。Image Builder 並 AWS TOE 支援下列應用程式環境：prodpreprod、和beta。

- AWS TOE 元件管理員值區：

  ```
  s3://ec2imagebuilder-toe-region-environment
  ```

  例如：s3://ec2 imagebuilder-toe-us-west -2-品質 /*
- Image Builder 管理的資源值區：

  ```
  s3://ec2imagebuilder-managed-resources-region-environment/components
  ```

  例如：s3://ec2 imagebuilder-managed-resources-us-西 2-產品/組件 /*

VPC 端點政策範例

本節包括自訂 VPC 端點原則的範例。

Image Builder 動作的一般 VPC 端點原則

以下 Image Builder 的範例端點策略拒絕刪除 Image Builder 和元件的權限。範例政策也授予執行所有其他 EC2 Image Builder 動作的權限。

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Action": "imagebuilder:*",
        "Effect": "Allow",
        "Resource": "*"
    },
    {
        "Action": [
            "imagebuilder: DeleteImage"
        ],
        "Effect": "Deny",
        "Resource": "*",
    },
    {
        "Action": [
            "imagebuilder: DeleteComponent"
        ],
        "Effect": "Deny",
        "Resource": "*",
    }]
}
```

依組織限制存取，允許受管理元件存取

下列範例端點原則示範如何限制存取屬於您組織的身分識別和資源，並提供對 Amazon-Managed 元 AWS TOE 件的存取權。*resource-org-id*以您組織的值取代*##principal-org-id*、和。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowRequestsByOrgsIdentitiesToOrgsResources",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "principal-org-id",
```

```
          "aws:ResourceOrgID": "resource-org-id"
        }
      }
    },
    {
      "Sid": "AllowAccessToEC2ImageBuilderComponents",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "imagebuilder:GetComponent"
      ],
      "Resource": [
        "arn:aws:imagebuilder:region:aws:component/*"
      ]
    }
  ]
}
```

適用於 Amazon S3 儲存貯體存取的 VPC 端點政策

下列 S3 端點政策範例顯示如何提供對映像產生器用來建立自訂映像之 S3 儲存貯體的存取權。以組織的價值取代##和##。根據您的應用程式需求，將任何其他必要權限新增至原則。

> **Note**
>
> 對於 Linux 映像檔，如果您未在映像配方中指定使用者資料，映 Image Builder 會新增指令碼，以便在映像檔的組建和測試執行個體上下載並安裝 Systems Manager 代理程式。若要下載代理程式，Image Builder 會存取您組建區域的 S3 儲存貯體。
> 為確保 Image Builder 可以啟動建置和測試執行個體，請將下列其他資源新增至 S3 端點政策：
> `"arn:aws:s3:::amazon-ssm-region/*"`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowImageBuilderAccessToAppAndComponentBuckets",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
```

```
      },
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::ec2imagebuilder-toe-region-environment/*",
        "arn:aws:s3:::ec2imagebuilder-managed-resources-region-environment/components/
*"
      ]
    }
  ]
}
```

管理 EC2 Image Builder 分發設定

使用 Image Builder 建立發佈設定後，您可以使用 Image Builder 主控台、映 Image Builder API 或中的imagebuilder命令來管理這些設定 AWS CLI。透過發佈設定，您可以執行下列動作：

AMI 分佈

- 指定輸出 AMI 的名稱和描述。
- 授權其他 AWS 帳戶、組織和 OU 從擁有者的帳戶啟動 AMI。擁有者帳戶會針對與 AMI 相關聯的費用計費。

> **Note**
>
> 若要公開 AMI，請將啟動許可授權帳戶設定為 `all`。請參閱在 EC2 上公開 AMI 的示例ModifyImageAttribute。

- 針對目的地區域中的每個指定目標帳戶、組織和 OU，建立輸出 AMI 的副本。目標帳戶、組織和 OU 擁有其 AMI 副本，並會針對任何相關費用計費。如需有關將 AMI 發佈到 AWS Organizations 和 OU 的詳細資訊，請參閱與組織或 OU 共用 AMI。
- 將 AMI 複製到其他擁有者的帳戶 AWS 區域。
- 將虛擬機器映像磁碟匯出至亞馬遜簡易儲存服務 (Amazon S3)。如需詳細資訊，請參閱 建立輸出虛擬機器磁碟的散佈設定 (AWS CLI)。

容器映像分佈

- 指定影像產生器將輸出影像儲存在分佈區域的 ECR 存放庫。

您可以透過下列方式使用發佈設定，一次將映像傳遞至目標區域、帳戶 AWS Organizations 和組織單位 (OU)，或在每個管道建置時傳遞映像：

- 若要自動將更新的映像傳遞至指定的區域、帳戶、Organizations 和 OU，請使用分發設定搭配按排程執行的 Image Builder 管線。
- 若要建立新映像並將其傳遞至指定的區域、帳戶、Organizations 和 OU，請使用 Image Builder 主控台執行一次的 Image Builder 管線，並使用 [動作] 功能表中的 [執行管線] 來搭配發佈設定。
- 若要建立新映像並將其傳遞至指定的區域、帳戶、Organizations 和 OU，請在下列 API 動作或 Image Builder 命令中使用發佈設定 AWS CLI：
  - Image Builder API 中的CreateImage動作。
  - 中的create-image指令 AWS CLI。
- 將虛擬機器 (VM) 映像磁碟匯出至目標區域的 S3 儲存貯體，做為一般映像建置程序的一部分。

> **Tip**
>
> 當您有多個相同類型的資源時，標記可協助您根據指派給該資源的標籤來識別特定資源。如需有關使用「Image Builder」指令標記資源的詳細資訊 AWS CLI，請參閱本指南一標籤資源節。

本主題說明如何列出、檢視和建立發佈設定。

目錄

列出並檢視散佈設定詳細資料

本節說明您可以尋找 EC2 Image Builder 分發設定的資訊和檢視詳細資料的各種方式。

發佈設定詳細資

- 列出分發配置（控制台）

- 檢視散發組態詳細資料 (主控台)
- 列表分佈（AWS CLI）
- 取得散發組態詳細資料 (AWS CLI)

列出分發配置（控制台）

若要在 Image Builder 主控台中查看在您帳戶下建立的散發組態清單，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導航窗格中選擇分發設置。這會顯示在您的帳戶下建立的散發組態清單。
3. 若要檢視詳細資料或建立新的發佈組態，請選擇 [組態名稱] 連結。這會開啟分佈設定的詳圖。

> **Note**
>
> 您也可以選取「組態」名稱旁的核取方塊，然後選擇「檢視詳細資料」。

檢視散發組態詳細資料 (主控台)

若要使用 Image Builder 主控台檢視特定散發組態的詳細資訊，請使用中所述的步驟選取要檢閱的組態列出分發配置（控制台）。

在「分佈詳細資訊」頁面上，您可以：

- 刪除發佈組態。如需有關在 Image Builder 中刪除資源的詳細資訊，請參閱刪除 EC2 Image Builder 資源。
- 編輯發佈詳細資料。

列表分佈（AWS CLI）

下列範例說明如何使用中的list-distribution-configurations命令 AWS CLI 來列出所有發行版。

```
aws imagebuilder list-distribution-configurations
```

取得散發組態詳細資料 (AWS CLI)

下列範例顯示如何使用中的get-distribution-configuration命令，透過指定其 Amazon 資源名稱 (ARN) AWS CLI 來取得分發組態的詳細資料。

```
aws imagebuilder get-distribution-configuration --distribution-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-
distribution-configuration
```

創建和更新 AMI 分發配置

本節涵蓋建立和更新 Image Builder AMI 的發佈組態。

目錄

創建 AMI 分發配置（控制台）

散發組態包括輸出 AMI 名稱、加密的特定區域設定、啟動權限 AWS 帳戶，以及可啟動輸出 AMI 的組織和組織單位 (OU)，以及授權組態。

要創建新的 AMI 分發配置：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導航窗格中選擇分發設置。這會顯示在您的帳戶下建立的散發組態清單。
3. 選擇 「分佈設定」面板頂部附近的「建立分佈設定」。
4. 在 [映像類型] 區段中，選擇 Amazon 機器映像 (AMI) 輸出類型。
5. 在 [一般] 區段中，輸入散發組態的 [名稱]，以及選用說明。
6. 在「地區設定」區段中，針對您要分發 AMI 的每個區域輸入下列詳細資訊：

    a. AMI 預設會分配到目前的區域 (區域 1)。區域 1 是分配的來源。區域 1 的某些設定無法開啟進行編輯。對於您新增的任何區域，您可以從「地區」下拉式清單中選擇「地區」。

    Kms 金鑰可識別 AWS KMS key 用來為目標區域中映像加密 EBS 磁碟區的金鑰。重要的是要注意，這不適用於構建在源區域（區域 1）的帳戶下創建的原始 AMI。在組建發佈階段執行的加密僅適用於散佈至其他帳戶或區域的映像。

若要為您的帳戶在來源區域中建立的 AMI 加密 EBS 磁碟區，您必須在映像配方區塊裝置對應 (主控台中的儲存 (磁碟區) 中設定 KMS 金鑰。

Image Builder 會將 AMI 複製到您為區域指定的目標帳戶。

> **先決條件**
>
> 若要跨帳戶複製映像，您必須在目標區域中的所有目標帳戶中建立`EC2ImageBuilderDistributionCrossAccountRole`角色，並將Ec2ImageBuilderCrossAccountDistributionAccess 政策受管理的策略附加至該角色。

輸出 AMI 名稱是可選的。如果您提供名稱，則最終輸出 AMI 名稱會包含 AMI 建置時的附加時間戳記。如果您未指定名稱，Image Builder 會將組建時間戳記附加至方案名稱。這確保了每個構建的唯一 AMI 名稱。

i. 透過 AMI 共用，您可以授與指定 AWS 主體的存取權，以便從 AMI 啟動執行個體。如果您展開 AMI 共享部分，則可以輸入以下詳細信息：

   - 啟動權限 — 如果您想要讓 AMI 保持私密，並允許特定 AWS 主參與者存取以從私人 AMI 啟動執行個體，請選取 [私人]。如果您想要公開 AMI，請選取「公開」。任何 AWS 主體都可以從您的公有 AMI 啟動執行個體。
   - 主參與者 — 您可以授與下列類型之 AWS 主參與者的存取權，以啟動實例：
     - AWS 帳戶 — 授予特定 AWS 帳戶的存取權
     - 組織單位 (OU) — 授與 OU 及其所有子系實體的存取權。子系實體包括 OU 和 AWS 帳戶。
     - 組織 — 授與您 AWS Organizations及其所有子實體的存取權。子系實體包括 OU 和 AWS 帳戶。

       首先，選取「主參與者」類型。然後在下拉式清單右側的方塊中，輸入您要授與存取權的 AWS 主參與者 ID。您可以輸入不同類型的多個 ID。

ii. 您可以展開 [授權組態] 區段，將使用建立的授權組態附加 AWS License Manager 至 Image Builder。授權組態包含以企業合約條款為基礎的授權規則。Image Builder 會自動包含與基礎 AMI 相關聯的授權組態。

iii. 您可以展開啟動範本設定區段，以指定用於從您建立的 AMI 啟動執行個體的 EC2 啟動範本。

如果您使用 EC2 啟動範本，可以指示 Image Builder 建立新版本的啟動範本，其中包含建置完成後的最新 AMI ID。若要更新啟動範本，請依照下列方式進行設定：

- 啟動範本名稱 — 選取您希望 Image Builder 更新的啟動範本名稱。
- 設定預設版本 — 選取此核取方塊可將啟動範本預設版本更新為新版本。

若要新增其他啟動範本組態，請選擇 [新增啟動範本組態]。每個區域最多可以有五個啟動範本設定。

b. 若要新增其他區域的分佈設定，請選擇 [新增地區]。

7. 完成後，選擇 [建立設定]。

建立輸出 AMI 的發佈設定 ()AWS CLI

分發配置允許您指定輸出 AMI 的名稱和描述，授權其他 AWS 帳戶 人啟動 AMI，將 AMI 複製到其他帳戶，並將 AMI 複製到其他 AWS 區域。它還允許您將 AMI 導出到 Amazon Simple Storage Service (Amazon S3)，或為輸出窗口 AMI 配置 EC2 快速啟動。若要公開 AMI，請將啟動許可授權帳戶設定為 `all`。請參閱在 EC2 上公開 AMI 的示例ModifyImageAttribute。

下列範例顯示如何使用create-distribution-configuration命令為 AMI 建立新的發佈組態，使用 AWS CLI.

1. 建立 CLI 輸入 JSON 文件

使用檔案編輯工具建立 JSON 檔案，其中包含下列其中一個範例中顯示的金鑰，以及對您環境有效的值。這些範例定義哪 AWS 帳戶些單位 AWS Organizations 或組織單位 (OU) 有權啟動您散佈至指定區域的 AMI。命名檔案`create-ami-distribution-configuration.json`，以便在下一個步驟中使用：

Accounts

此範例會將 AMI 散發至兩個區域，並指定 AWS 帳戶 在每個區域中具有啟動權限。

```
{
    "name": "MyExampleAccountDistribution",
    "description": "Copies AMI to eu-west-1, and specifies accounts that can
 launch instances in each Region.",
    "distributions": [
```

```
        {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
                "name": "Name {{imagebuilder:buildDate}}",
                "description": "An example image name with parameter
 references",
                "amiTags": {
                    "KeyName": "Some Value"
                },
                "launchPermission": {
                    "userIds": [
                        "987654321012"
                    ]
                }
            }
        },
        {
            "region": "eu-west-1",
            "amiDistributionConfiguration": {
                "name": "My {{imagebuilder:buildVersion}} image
 {{imagebuilder:buildDate}}",
                "amiTags": {
                    "KeyName": "Some value"
                },
                "launchPermission": {
                    "userIds": [
                        "100000000001"
                    ]
                }
            }
        }
    ]
}
```

Organizations and OUs

此範例會將 AMI 發佈至來源區域，並指定組織和 OU 啟動權限。

```
{
  "name": "MyExampleAWSOrganizationDistribution",
  "description": "Shares AMI with the Organization and OU",
  "distributions": [
    {
```

```
      "region": "us-west-2",
      "amiDistributionConfiguration": {
        "name": "Name {{ imagebuilder:buildDate }}",
        "launchPermission": {
          "organizationArns": [
            "arn:aws:organizations::123456789012:organization/o-
myorganization123"
          ],
          "organizationalUnitArns": [
            "arn:aws:organizations::123456789012:ou/o-123example/ou-1234-
myorganizationalunit"
          ]
        }
      }
    }
  ]
}
```

2. 使用您建立做為輸入的檔案，執行下列命令。

```
aws imagebuilder create-distribution-configuration --cli-input-json file://create-ami-distribution-configuration.json
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

若要取得更多詳細資訊，請參閱《指AWS CLI 令參考》create-distribution-configuration中的。

更新 AMI 分發設置（控制台）

您可以使用 Image Builder 主控台變更 AMI 分發設定。更新的發佈設定會用於未來所有自動和手動管線部署。但是，您所做的變更並不會套用至 Image Builder 已經散佈的任何資源。例如，如果您已將 AMI 分配到稍後從發行版中移除的區域，則已分配的 AMI 會保留在該區域中，直到您手動將其移除為止。

更新 AMI 分發配置

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導航窗格中選擇分發設置。這會顯示在您的帳戶下建立的散發組態清單。
3. 若要檢視詳細資料或更新發佈組態，請選擇 [組態名稱] 連結。這會開啟分佈設定的詳圖。

> **Note**
>
> 您也可以選取「組態」名稱旁的核取方塊，然後選擇「檢視詳細資料」。

4. 若要編輯發佈組態，請從 [發佈詳細資料] 區段右上角選擇 [編輯]。某些欄位已鎖定，例如發佈組態的「名稱」，以及顯示為「區域 1」的預設「區域」。如需有關散發組態設定的詳細資訊，請參閱創建 AMI 分發配置（控制台）。
5. 完成後，請選擇 Save changes (儲存變更)。

在啟用 EC2 快速啟動的情況下為 Windows AMI 建立發佈設定 (AWS CLI)

下列範例顯示如何使用create-distribution-configuration命令建立已為 AMI 設定 EC2 快速啟動的發佈設定，從 AWS CLI.

> **Note**
>
> 預先啟用 EC2 快速啟動時，Image Builder 不支援 AMI 的跨帳戶分發。必須從目的地帳戶啟用 EC2 快速啟動。

1. 建立 CLI 輸入 JSON 文件

   使用檔案編輯工具建立包含金鑰的 JSON 檔案，如下列範例所示，加上適用於您環境的值。

   此範例會同時啟動其所有目標資源的執行處理，因為 parallel 啟動次數上限大於目標資源計數。此檔案會`ami-dist-config-win-fast-launch.json`在下一個步驟中顯示的命令範例中命名。

```
{
"name": "WinFastLaunchDistribution",
"description": "An example of Windows AMI EC2 Fast Launch settings in the
 distribution configuration.",
"distributions": [
```

```
    {
        "region": "us-west-2",
        "amiDistributionConfiguration": {
            "name": "Name {{imagebuilder:buildDate}}",
            "description": "Includes Windows AMI EC2 Fast Launch settings.",
            "amiTags": {
                "KeyName": "Some Value"
            }
        },
        "fastLaunchConfigurations": [{
            "enabled": true,
            "snapshotConfiguration": {
                "targetResourceCount": 5
            },
            "maxParallelLaunches": 6,
            "launchTemplate": {
                "launchTemplateId": "lt-0ab1234c56d789012",
                "launchTemplateVersion": "1"
             }
        }],
        "launchTemplateConfigurations": [{
                    "launchTemplateId": "lt-0ab1234c56d789012",
                    "setDefaultVersion": true
          }]
    }]
}
```

> **Note**
>
> 您可以在launchTemplate區段launchTemplateId中指定，launchTemplateName而不是指定名稱和 ID。

2. 使用您建立做為輸入的檔案，執行下列命令。

```
aws imagebuilder create-distribution-configuration --cli-input-json file://ami-dist-config-win-fast-launch.json
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 file:// 標記。

- JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

若要取得更多詳細資訊，請參閱《指AWS CLI 令參考》create-distribution-configuration中的。

建立輸出虛擬機器磁碟的散佈設定 (AWS CLI)

下列範例顯示如何使用create-distribution-configuration命令建立分發設定，以便在每次建置映像檔時將 VM 映像磁碟匯出到 Amazon S3。

1. 建立 CLI 輸入 JSON 文件

   您可以簡化在中使用的create-distribution-configuration指令 AWS CLI。若要這麼做，請建立一個 JSON 檔案，其中包含您要傳入指令的所有匯出組態。

   > **Note**
   >
   > JSON 檔案中資料值的命名慣例遵循針對 Image Builder API 動作要求參數指定的模式。若要檢閱 API 命令要求參數，請參閱 EC2 Image Builder API 參考中的CreateDistributionConfiguration命令。
   > 若要提供資料值做為指令行參數，請參考《指AWS CLI 令參考》中指定的參數名稱。以選項形式參考指create-distribution-configuration令。

   以下是我們針對此範例在 `s3ExportConfiguration` JSON 物件中指定的參數摘要：

   - Roel@@ Name (字串，必要) — 授與虛擬機器匯入/匯出權限以將映像匯出至 S3 儲存貯體的角色名稱。
   - diskImageFormat(字串，必要) — 將更新的磁碟映像檔匯出為下列其中一種支援的格式：
     - 虛擬硬碟 (VHD) — 兼容思杰 Xen 和 Microsoft Hyper-V 虛擬化產品。
     - 串流最佳化 ESX 虛擬機器磁碟 (VMDK) — 相容於 VMware ESX 和 VMware vSphere 第 4、5 和 6 版。
     - 原始 — 原始格式。
   - S3bucket (字串，必要) — 用來存放虛擬機器輸出磁碟映像檔的 S3 儲存貯體。

儲存檔案為 `export-vm-disks.json`。在create-distribution-configuration指令中使用檔案名稱。

```
{
    "name": "example-distribution-configuration-with-vm-export",
    "description": "example",
    "distributions": [
        {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
             "description": "example-with-vm-export"

            },
            "s3ExportConfiguration": {
                "roleName": "vmimport",
                "diskImageFormat": "RAW",
                "s3Bucket": "vm-bucket-export"
            }
        }],
    "clientToken": "abc123def4567ab"
}
```

2. 使用您建立做為輸入的檔案，執行下列命令。

```
aws imagebuilder create-distribution-configuration --cli-input-json file://export-vm-disks.json
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

若要取得更多詳細資訊，請參閱《指AWS CLI 令參考》create-distribution-configuration中的。

更新 AMI 發佈設定 (AWS CLI)

下列範例顯示如何使用update-distribution-configuration命令來更新 AMI 的發佈設定，使用 AWS CLI.

1. 建立 CLI 輸入 JSON 文件

   使用您最愛的檔案編輯工具，建立 JSON 檔案，其中包含下列範例中顯示的金鑰，以及適用於您環境的值。此範例使用名為的檔案update-ami-distribution-configuration.json。

```
{
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/update-ami-distribution-
configuration.json",
    "description": "Copies AMI to eu-west-2, and specifies accounts that can launch
 instances in each Region.",
    "distributions": [
      {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
                "name": "Name {{imagebuilder:buildDate}}",
                "description": "An example image name with parameter references",
                "launchPermissions": {
                    "userIds": [
                        "987654321012"
                    ]
                }
            }
       },
       {
            "region": "eu-west-2",
            "amiDistributionConfiguration": {
                "name": "My {{imagebuilder:buildVersion}} image
 {{imagebuilder:buildDate}}",
                "tags": {
                    "KeyName": "Some value"
                },
                "launchPermissions": {
                    "userIds": [
                        "100000000001"
                    ]
                }
            }
       }
```

```
    ]
}
```

2. 使用您建立做為輸入的檔案，執行下列命令。

```
aws imagebuilder update-distribution-configuration --cli-input-json file://update-ami-distribution-configuration.json
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

若要取得更多詳細資訊，請參閱《指AWS CLI 令參考》update-distribution-configuration中的。若要更新發佈組態資源的標籤，請參閱標籤資源章節。

建立和更新容器映像的發佈設定

本節涵蓋建立和更新映像產生器容器映像的散佈設定。

目錄

建立映像產生器容器映像的發佈設定 (AWS CLI)

散發組態可讓您指定輸出容器映像的名稱和說明，並將容器映像複寫到其他 AWS 區域。您也可以將個別標籤套用至發佈組態資源，以及每個區域內的容器映像。

1. 建立 CLI 輸入 JSON 文件

   使用您最愛的檔案編輯工具，建立 JSON 檔案，其中包含下列範例中顯示的金鑰，以及適用於您環境的值。此範例使用名為 `create-container-distribution-configuration.json` 的檔案：

```
{
 "name": "distribution-configuration-name",
 "description": "Distributes container image to Amazon ECR repository in two
 regions.",
 "distributions": [
     {
         "region": "us-west-2",
         "containerDistributionConfiguration": {
             "description": "My test image.",
             "targetRepository": {
                 "service": "ECR",
                 "repositoryName": "testrepo"
             },
             "containerTags": ["west2", "image1"]
         }
     },
     {
         "region": "us-east-1",
         "containerDistributionConfiguration": {
             "description": "My test image.",
             "targetRepository": {
                 "service": "ECR",
                 "repositoryName": "testrepo"
             },
            "containerTags": ["east1", "imagedist"]
         }
     }
 ],
 "tags": {
    "DistributionConfigurationTestTagKey1":
 "DistributionConfigurationTestTagValue1",
    "DistributionConfigurationTestTagKey2":
 "DistributionConfigurationTestTagValue2"
 }
}
```

2. 使用您建立做為輸入的檔案，執行下列命令。

```
aws imagebuilder create-distribution-configuration --cli-input-json file://create-
container-distribution-configuration.json
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

若要取得更多詳細資訊，請參閱《指AWS CLI 令參考》create-distribution-configuration中的。

更新容器映像檔的發佈設定 (AWS CLI)

下列範例示範如何使用命update-distribution-configuration令來更新容器映像的發佈設定，使用 AWS CLI. 您也可以更新每個區域內容器映像檔的標籤。

1. 建立 CLI 輸入 JSON 文件

   使用您最愛的檔案編輯工具建立 JSON 檔案，其中包含下列範例中顯示的金鑰，以及適用於您環境的值。此範例使用名為 `update-container-distribution-configuration.json` 的檔案：

```
{
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/update-container-distribution-
configuration.json",
 "description": "Distributes container image to Amazon ECR repository in two
 regions.",
 "distributions": [
     {
         "region": "us-west-2",
         "containerDistributionConfiguration": {
             "description": "My test image.",
             "targetRepository": {
                 "service": "ECR",
                 "repositoryName": "testrepo"
             },
             "containerTags": ["west2", "image1"]
         }
     },
     {
```

```
            "region": "us-east-2",
            "containerDistributionConfiguration": {
                "description": "My test image.",
                "targetRepository": {
                    "service": "ECR",
                    "repositoryName": "testrepo"
                },
               "containerTags": ["east2", "imagedist"]
            }
        }
    ]
}
```

2. 使用您建立的檔案做為輸入執行下列命令：

```
aws imagebuilder update-distribution-configuration --cli-input-json file://update-container-distribution-configuration.json
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

若要取得更多詳細資訊，請參閱《指AWS CLI 令參考》update-distribution-configuration中的。若要更新發佈組態資源的標籤，請參閱標籤資源章節。

使用 Image Builder 設定跨帳戶 AMI 分配

本節說明如何設定發佈設定，將 Image Builder AMI 傳遞給您指定的其他帳戶。

然後，目的地帳戶可以根據需要啟動或修改 AMI。

> **Note**
>
> AWS CLI 本節中的命令範例假設您先前已建立映像配方和基礎結構組態 JSON 檔案。若要建立影像配方的 JSON 檔案，請參閱創建一個圖像配方 AWS CLI。若要為基礎結構組態建立 JSON 檔案，請參閱建立基礎架構組態。

必要條件

若要確保目標帳戶可以從 Image Builder 映像檔成功啟動執行個體，您必須為所有區域中的所有目標帳戶設定適當的權限。

如果使用 AWS Key Management Service (AWS KMS) 加密 AMI，則必須 AWS KMS key 為用於加密新映像的帳戶配置一個。

Image Builder 對加密 AMI 執行跨帳戶散發時，來源帳戶中的映像會解密並推送至目標區域，在此區域會使用該區域的指定金鑰重新加密該影像。由於 Image Builder 代表目標帳戶運作，並使用您在目標區域中建立的 IAM 角色，因此該帳戶必須具有來源和目標區域中金鑰的存取權。

加密金鑰

如果您的映像使用加密，則需要下列先決條件 AWS KMS。下一節將涵蓋 IAM 必要條件。

來源帳戶需求

- 在您建立和散佈 AMI 的所有區域中，在您的帳戶中建立 KMS 金鑰。您也可以使用現有的金鑰。
- 更新所有這些金鑰的金鑰原則，以允許目的地帳戶使用您的金鑰。

目的地帳戶需求

- 將內嵌原則新增至`EC2ImageBuilderDistributionCrossAccountRole`，可讓角色執行必要動作以散發加密的 AMI。如需 IAM 組態步驟，請參閱IAM 政策先決條件一節。

如需跨帳戶存取使用的詳細資訊 AWS KMS，請參閱AWS Key Management Service 開發人員指南中的允許其他帳戶中的使用者使用 KMS 金鑰。

在映像配方中指定您的加密金鑰，如下所示：

- 如果您使用的是 Image Builder 主控台，請從方案的儲存體 (磁碟區) 區段的加密 (KMS 別名) 下拉式清單中選擇您的加密金鑰。

- 如果您正在使用 CreateImageRecipe API 動作或中的create-image-recipe命令 AWS CLI，請在 JSON 輸入中的ebs區段blockDeviceMappings中設定金鑰。

  下列 JSON 程式碼片段顯示影像方案的加密設定。除了提供加密金鑰之外，您還必須將encrypted旗標設定為true。

```
{
 ...
 "blockDeviceMappings": [
 {
  "deviceName": "Example root volume",
  "ebs": {
   "deleteOnTermination": true,
   "encrypted": true,
   "iops": 100,
   "kmsKeyId": "image-owner-key-id",
   ...
  },
  ...
 }],
 ...
}
```

IAM 政策

若要在 AWS Identity and Access Management (IAM) 中設定跨帳戶分發許可，請依照下列步驟執行：

1. 若要使用跨帳戶分散的 Image Builder AMI，目標帳戶擁有者必須在名為EC2ImageBuilderDistributionCrossAccountRole的帳戶中建立新的 IAM 角色。
2. 他們必須附加Ec2ImageBuilderCrossAccountDistributionAccess 政策至角色，才能啟用跨帳戶分配。如需有關受管理策略的詳細資訊，請參閱AWS Identity and Access Management 使用指南中的受管理的策略和內嵌政策。
3. 確認來源帳戶 ID 已新增至目標帳戶 IAM 角色附加的信任政策。如需有關信任策略的詳細資訊，請參閱AWS Identity and Access Management 使用指南中的以資源為基礎的策略。
4. 如果您散佈的 AMI 已加密，則目的地帳戶擁有者必須在其帳戶EC2ImageBuilderDistributionCrossAccountRole中新增下列內嵌政策，以便他們可以使用您的 KMS 金鑰。該Principal部分包含其帳戶號碼。這可讓 Image Builder 在使用每個區域的適當金鑰 AWS KMS 加密和解密 AMI 時代表他們採取行動。

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowRoleToPerformKMSOperationsOnBehalfOfTheDestinationAccount",
   "Effect": "Allow",
   "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey",
    "kms:CreateGrant",
    "kms:ListGrants",
    "kms:RevokeGrant"
   ],
   "Resource": "*"
  }
 ]
}
```

如需有關內嵌原則的詳細資訊，請參閱AWS Identity and Access Management 使用指南中的內嵌政策。

5. 如果您使用指`launchTemplateConfigurations`定 Amazon EC2 啟動範本，則還必須`EC2ImageBuilderDistributionCrossAccountRole`在每個目的地帳戶中新增以下政策。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateLaunchTemplateVersion",
                "ec2:ModifyLaunchTemplate"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeLaunchTemplates"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*:*:launch-template/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "EC2 Image Builder"
                }
            }
        }
    ]
}
```

跨帳戶分配限制

在不同帳戶之間散佈 Image Builder 時，會有一些限制：

- 每個目的地區域的目標帳戶限制為 50 個並行 AMI 副本。
- 如果您要將半虛擬化 (PV) 虛擬化 AMI 複製到另一個區域，目的地區域必須支援 PV 虛擬化 AMI。如需詳細資訊，請參閱 Linux AMI 虛擬化類型。
- 您無法建立加密快照的未加密副本。如果您未為`KmsKeyId`參數指定 AWS Key Management Service (AWS KMS) 客戶受管金鑰，Image Builder 會使用 Amazon Elastic Block Store (Amazon EBS) 的預設金鑰。如需詳細資訊，請參閱 Amazon 彈性運算雲端使用者指南中的 Amazon EBS 加密。

如需詳細資訊，請參閱 EC2 Image Builder API 參考CreateDistributionConfiguration中的。

設定 Image Builder AMI (主控台) 的跨帳戶散發

本節說明如何使用建立和設定 Image Builder AMI 的跨帳戶散發發佈設定。 AWS Management Console設定跨帳戶分配需要特定的 IAM 許可。在繼續之前，必要條件您必須先完成本節的內容。

若要在 Image Builder 主控台中建立散發設定，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導航窗格中選擇分發設置。這會顯示在您的帳戶下建立的散佈設定清單。
3. 在 [發佈設定] 頁面頂端，選擇 [建立發佈設定]。這會帶您前往 [建立發佈設定] 頁面。
4. 在「映像類型」區段中，選擇「Amazon 機器映像 (AMI)」作為「輸出」類型。這是預設設定。
5. 在「一般」段落中，輸入您要建立的散佈設定資源的「名稱」(必要)。
6. 在 [地區設定] 區段中，在所選區域的 Target 帳戶中輸入您要將 AMI 發佈至的 12 位數帳戶 ID，然後按 Enter。這會檢查正確的格式，然後顯示您在方塊下方輸入的帳戶 ID。重複此程序以新增更多帳戶。

   若要移除您輸入的帳號，請選擇帳號 ID 右側顯示的 X。

   輸入每個區域的「輸出 AMI」名稱。
7. 繼續指定您需要的任何其他設定，然後選擇 [建立設定] 以建立新的發佈設定資源。

設定 Image Builder AMI 的跨帳戶散發 ()AWS CLI

本節說明如何設定散發設定檔案，以及如何使用中的create-image命令在 AWS CLI 不同帳戶之間建置和散發 Image Builder AMI。

設定跨帳戶分配需要特定的 IAM 許可。在執行命令之必要條件前，您必須先完成本節的create-image指令。

1. 設定發佈設定檔案

   在使用中的create-image命令建立散發 AWS CLI 至其他帳戶的 Image Builder AMI 之前，您必須建立 `DistributionConfiguration` JSON 結構，以在`AmiDistributionConfiguration`設定中指定目標帳戶 ID。您必須在來源「區域」`AmiDistributionConfiguration` 中指定至少一個。

   下列名`create-distribution-configuration.json`為的範例檔案顯示來源區域中跨帳戶映像散發的組態。

```
{
 "name": "cross-account-distribution-example",
 "description": "Cross Account Distribution Configuration Example",
 "distributions": [
  {
   "amiDistributionConfiguration": {
    "targetAccountIds": ["123456789012", "987654321098"],
    "name": "Name {{ imagebuilder:buildDate }}",
    "description": "ImageCopy Ami Copy Configuration"
   },
   "region": "us-west-2"
  }
 ]
}
```

2. 建立發佈設定

   若要使用中的create-distribution-configuration指令建立 Image Builder 分發設定資源 AWS CLI，請在命令中提供下列參數：

   - 在參數中輸入分佈的名`--name`稱。
   - 附加您在`--cli-input-json`參數中建立的散發組態 JSON 檔案。

```
aws imagebuilder create-distribution-configuration --name my distribution name --
cli-input-json file://create-distribution-configuration.json
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

您也可以使用*`--distributions`*參數直接在命令中提供 JSON。

設定 AMI 分發設定以使用 Amazon EC2 啟動範本

為了確保您的 Image Builder AMI 在目標帳戶和區域中獲得一致的啟動體驗，您可以在分發設定中指定 Amazon EC2 啟動範本，使用`launchTemplateConfigurations`。如果存`launchTemplateConfigurations`在於發佈程序期間，Image Builder 會建立新版本的啟動範本，其中包含範本中的所有原始設定，以及來自組建的新 AMI ID。如需使用啟動範本啟動 EC2 執行個體的詳細資訊，請參閱下列其中一個連結，視您的目標作業系統而定。

- 從啟動範本啟動 Linux 執行個體
- 從啟動範本啟動 Windows 執行個體

> **Note**
>
> 當您包含啟動範本以在映像中啟用 Windows 快速啟動時，啟動範本必須包含下列標記，以便 Image Builder 可以代表您啟用 Windows 快速啟動。
> `CreatedBy: EC2 Image Builder`

將 Amazon EC2 啟動範本新增至您的 AMI 分發設定 (主控台)

若要透過輸出 AMI 提供啟動範本，請在主控台中依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導航窗格中選擇分發設置。這會顯示在您的帳戶下建立的散佈設定清單。
3. 在 [發佈設定] 頁面頂端，選擇 [建立發佈設定]。這會開啟 [建立發佈設定] 頁面。
4. 在 [映像類型] 區段中，選擇 Amazon 機器映像 (AMI) 輸出類型。這是預設設定。
5. 在「一般」段落中，輸入您要建立的散佈設定資源的「名稱」(必要)。
6. 在區域設定區段中，從清單中選取 EC2 啟動範本的名稱。如果您的帳戶中沒有啟動範本，請選擇 [建立新的啟動範本]，該範本會在 EC2 儀表板中開啟啟動範本。

   選取 [設定預設版本] 核取方塊，將啟動範本預設版本更新為 Image Builder 使用輸出 AMI 建立的新版本。

   若要將其他啟動範本新增至選取的區域，請選擇 [新增啟動範本組態]。

   若要移除啟動範本，請選擇 [移除]。
7. 繼續指定您需要的任何其他設定，然後選擇 [建立設定] 以建立新的發佈設定資源。

將 Amazon EC2 啟動範本新增至您的 AMI 分發設定 (AWS CLI)

本節說明如何使用啟動範本設定發佈設定檔案，以及如何使用中的create-image指令 AWS CLI 來建置和發佈 Image Builder AMI 以及使用該範本的新版啟動範本。

1. 設定發佈設定檔案

   在您可以使用啟動範本建立 Image Builder AMI 之前 AWS CLI，您必須使用建立指定launchTemplateConfigurations設定的散發組態 JSON 結構。您必須在來源區域中指定至少一個launchTemplateConfigurations項目。

   以下名create-distribution-config-launch-template.json為的範例檔案顯示了來源區域中啟動範本組態的一些可能案例。

```
{
    "name": "NewDistributionConfiguration",
    "description": "This is just a test",
    "distributions": [
        {
            "region": "us-west-2",
            "amiDistributionConfiguration": {
                "name": "test-{{imagebuilder:buildDate}}-
{{imagebuilder:buildVersion}}",
                "description": "description"
            },
            "launchTemplateConfigurations": [
                {
                    "launchTemplateId": "lt-0a1bcde2fgh34567",
                    "accountId": "935302948087",
                    "setDefaultVersion": true
                },
                {
                    "launchTemplateId": "lt-0aaa1bcde2ff3456"
                },
                {
                    "launchTemplateId": "lt-12345678901234567",
                    "accountId": "123456789012"
                }
            ]
        }
    ],
    "clientToken": "clientToken1"
```

```
}
```

2. 建立發佈設定

   若要使用中的create-distribution-configuration指令建立 Image Builder 分發設定資源 AWS CLI，請在命令中提供下列參數：

   - 在參數中輸入分佈的名`--name`稱。
   - 附加您在`--cli-input-json`參數中建立的散發組態 JSON 檔案。

```
aws imagebuilder create-distribution-configuration --name my distribution name--
cli-input-json file://create-distribution-config-launch-template.json
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

您也可以使用*`--distributions`*參數直接在命令中提供 JSON。

管理 EC2 Image Builder 映像的生命週期政策

建立自訂映像時，您必須有計劃在這些影像過時之前淘汰這些影像。Image Builder 管線可自動套用更新和安全性修補程式。不過，每個組建都會建立映像檔的新版本，以及它所散佈的所有相關聯資源。舊版本會保留在您的帳戶中，直到您手動刪除它們，或建立指令碼來執行工作為止。

透過 Image Builder 生命週期管理政策，您可以自動化淘汰、停用和刪除過期映像檔及其相關資源的程序。關聯的資源可以包括您已分發給其他 AWS 帳戶、組織和組織單位 (OU) 的輸出影像 AWS 區域。您可以定義如何及何時採取生命週期程序中的每個步驟，以及要在原則中包含哪些步驟的規則。

自動生命週期管理的優點

自動化生命週期管理的整體優勢包括：

- 透過自動化方式淘汰影像和相關資源，簡化自訂映像檔的生命週期管理。
- 協助防止使用過期映像檔啟動新執行個體所帶來的法規遵循風險。

- 移除過時的影像，讓影像庫存保持最新狀態。
- 可以選擇性地移除已刪除影像的相關資源，藉此降低儲存和資料傳輸成本。

實現節省成本

使用 EC2 Image Builder 建立自訂 AMI 或容器映像無須付費。不過，標準定價適用於程序中使用的其他服務。當您從中移除未使用或過期的影像及其相關資源時 AWS 帳戶，您可以透過下列方式節省時間和成本：

- 當您不同時修補未使用或過時的影像時，可減少修補現有影像所需的時間。
- 對於您刪除的 AMI 映像資源，您也可以選擇移除分散式 AMI 及其相關聯的快照。這種方法可以節省儲存快照的成本。
- 對於您刪除的容器映像資源，您可以選擇刪除基礎資源。此方法可為儲存在 ECR 儲存庫中的Docker 映像節省 Amazon ECR 儲存成本和資料傳輸費率。

> **Note**
>
> Image Builder 無法評估所有可能的下游相依性 (例如 Auto Scaling 群組或啟動範本) 的潛在影響。設定原則動作時，您必須考慮映像的下游相依性。

目錄

EC2 Image Builder 映像的生命週期管理必要

您必須符合下列先決條件，才能為映像資源定義 EC2 Image Builder 生命週期管理政策和規則。

- 建立 IAM 角色，授與 Image Builder 執行生命週期政策的權限。若要建立角色，請參閱建立映像產生器生命週期管理的 IAM 角色。
- 在目標帳戶中為跨帳戶分配的關聯資源建立 IAM 角色。此角色會授與 Image Builder 在目標帳戶中針對關聯資源執行生命週期動作的權限。若要建立角色，請參閱建立 Image Builder 跨帳戶生命週期管理的 IAM 角色。

> **Note**
>
> 如果您已授與輸出 AMI 的啟動權限，則此必要條件不適用。透過啟動權限，您共用的帳戶擁有從共用 AMI 啟動的執行個體，但所有 AMI 資源仍保留在您的帳戶中。

- 對於容器映像檔，您必須將下列標籤新增至 ECR 儲存庫，才能授與 Image Builder 對儲存在存放庫中的容器映像執行生命週期動作的存取權限：`LifecycleExecutionAccess: EC2 Image Builder`

建立映像產生器生命週期管理的 IAM 角色

若要授與 Image Builder 執行生命週期政策的權限，您必須先建立其用來執行生命週期動作的 IAM 角色。請依照下列步驟建立授與權限的服務角色。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 從導覽窗格選擇 Roles (角色)。
3. 選擇建立角色。這會開啟程序中的第一個步驟：選取信任的實體以建立您的角色。
4. 選取 [信任的實體類型] 的 [自訂信任原則] 選項。
5. 複製下列 JSON 信任原則，並將其貼到 [自訂信任原則] 文字區域，取代範例文字。此信任原則可讓 Image Builder 擔任您為執行生命週期動作而建立的角色。

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

6. 從清單中選取下列受管理的策略：EC2ImageBuilderLifecycleExecutionPolicy，然後選擇下一步。這會開啟 [名稱]、[檢閱] 和 [建立] 頁面。

> **Tip**
>
> 篩選`image`以簡化結果。

7. 輸入 Role name (角色名稱)。
8. 檢閱設定後，請選擇 [建立角色]。

建立 Image Builder 跨帳戶生命週期管理的 IAM 角色

若要授予 Image Builder 在目標帳戶中針對相關資源執行生命週期動作的權限，您必須先建立 IAM 角色，以便在這些帳戶中執行生命週期動作。您必須在目的地帳戶中建立角色。

請依照下列步驟建立授與目標帳戶權限的服務角色。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 從導覽窗格選擇 Roles (角色)。
3. 選擇建立角色。這會開啟程序中的第一個步驟：選取信任的實體以建立您的角色。
4. 選取 [信任的實體類型] 的 [自訂信任原則] 選項。
5. 複製下列 JSON 信任原則，並將其貼到 [自訂信任原則] 文字區域，取代範例文字。此信任原則可讓 Image Builder 擔任您為執行生命週期動作而建立的角色。

> **Note**
>
> 當 Image Builder 在目標帳戶中使用此角色來處理跨帳戶分配的關聯資源時，它代表目標帳戶擁有者執行動作。您在 AWS 帳戶 信任策略`aws:SourceAccount`中設定的是 Image Builder 分配這些資源的帳戶。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
```

```
                    "imagebuilder.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "444455556666"
                },
                "StringLike": {
                    "aws:SourceArn": "arn:*:imagebuilder:*:*:image/*/*/*"
                }
            }
        }
    ]
}
```

6. 從清單中選取下列受管理的策略：EC2ImageBuilderLifecycleExecutionPolicy，然後選擇下一步。這會開啟 [名稱]、[檢閱] 和 [建立] 頁面。

> **Tip**
>
> 篩選image以簡化結果。

7. 輸入 Ec2ImageBuilderCrossAccountLifecycleAccess「角色」名稱。

> **Important**
>
> Ec2ImageBuilderCrossAccountLifecycleAccess必須是此角色的名稱。

8. 檢閱設定後，請選擇 [建立角色]。

EC2 Image Builder 映像資源的生命週期管理政策

透過映像生命週期政策，您可以定義資源管理策略，透過淘汰、停用和刪除過期映像檔及其相關資源的程序，淘汰過期的映像檔及其相關資源。本節說明如何列出原則、檢視原則詳細資料，以及建立 AMI 和容器映像的新原則。

目錄

- 建立生命週期原

列出映像產生 Image Builder 資源的生命週期管理

您可以取得映像生命週期管理政策的清單，其中包括中的生命週期政策清單頁面上的索引鍵詳細資料欄 AWS Management Console，或在 Image Builder API、SDK 或中使用指令或 AWS CLI動作。

您可以使用下列其中一種方法列出您的 Image Builder 映像生命週期原則資源 AWS 帳戶。如需 API 動作的相關資訊，請參閱 EC2 Image Builder API 參考ListLifecyclePolicies中的。如需相關聯的 SDK 要求，請參閱相同頁面上的「另請參閱」連結。

AWS Management Console

您現有策略的下列詳細資訊會顯示在主控台中。您可以選取任何欄來變更結果的排序順序。原則清單一開始會依策略名稱排序。目前排序順序的欄名稱為粗體。

如果您有多個頁面的結果，面板右上角的分頁箭頭會變為作用中。您可以使用搜尋列依策略名稱、策略狀態、輸出影像類型和影像資源 ARN 篩選結果。

- 策略名稱 — 策略的名稱。
- 策略狀態 — 策略是作用中還是非作用中。
- 類型 — 當您建立新映像版本 (AMI 或容器映像檔) 時，映像產生器散佈的輸出影像類型。
- 上次執行日期 — 上次執行生命週期原則的時間。
- 建立日期 — 建立生命週期原則時的時間戳記。
- ARN — 生命週期政策資源的 Amazon 資源名稱 (ARN)。

若要在中列出生命週期策略 AWS Management Console，請遵循下列步驟：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導覽窗格中選取「生命週期原則」。這會顯示您帳戶中的映像生命週期政策清單。

可用動作

您也可以從生命週期政策清單頁面針對生命週期原則執行下列動作。

若要建立新映像生命週期原則，請選擇 [建立生命週期原則] 如需如何建立策略的詳細資訊，請參閱建立生命週期原。

對於下列所有動作，您必須先選取策略。若要選取策略，您可以選取策略名稱旁邊的核取方塊。

- 若要關閉或開啟策略，請從 [動作] 功能表中選取 [停用策略] 或 [啟用策略]。
- 若要變更策略，請從 [動作] 功能表中選取 [編輯策略]。
- 若要刪除策略，請從 [動作] 功能表中選取 [刪除策略]。
- 若要建立使用您選取的基準設定原則的新原則，請從 [動作] 功能表中選取 [複製原則]。

AWS CLI

以下指令範例展示如何使用列 AWS CLI 示特定影像生命週期策略 AWS 區域。若要取得有關可與此指令配合使用的參數和選項的更多資訊，請參閱《list-lifecycle-policies指令參考》中的 AWS CLI 指令。

範例：

```
aws imagebuilder list-lifecycle-policies \
--region us-west-1
```

輸出：

```
{
    "lifecyclePolicySummaryList": [
        {
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:lifecycle-policy/
sample-lifecycle-policy1",
            "name": "sample-lifecycle-policy1",
            "status": "DISABLED",
            "executionRole": "arn:aws:iam::111122223333:role/sample-lifecycle-role",
            "resourceType": "AMI_IMAGE",
            "dateCreated": "2023-11-07T14:57:01.603000-08:00",
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:lifecycle-policy/
sample-lifecycle-policy2",
            "name": "sample-lifecycle-policy2",
            "status": "ENABLED",
            "executionRole": "arn:aws:iam::111122223333:role/sample-lifecycle-role",
            "resourceType": "AMI_IMAGE",
            "dateCreated": "2023-09-06T10:43:21.436000-07:00",
            "dateLastRun": "2023-11-13T04:43:46.106000-08:00",
```

```
            "tags": {}
        },
        {
            "arn": "arn:aws:imagebuilder:us-west-2:111122223333:lifecycle-policy/
sample-lifecycle-policy3",
            "name": "sample-lifecycle-policy3",
            "status": "ENABLED",
            "executionRole": "arn:aws:iam::111122223333:role/sample-lifecycle-role",
            "resourceType": "AMI_IMAGE",
            "dateCreated": "2023-10-19T15:16:40.046000-07:00",
            "dateUpdated": "2023-10-21T20:07:15.958000-07:00",
            "dateLastRun": "2023-11-12T09:27:45.830000-08:00"
}]}
```

> **Note**
>
> 若要使用預設值 AWS 區域，請不使用`--region`參數執行此命令。

檢視生命週期原則詳

Image Builder 主控台中的生命週期原則詳細資料頁面包含摘要區段，其中包含分組成索引標籤的其他資訊 頁面標題是策略的名稱。

在 Image Builder 主控台的生命週期原則詳細資料頁面上，您可以檢視特定生命週期原則的詳細資料。您也可以搭配 Image Builder API、SDK 使用命令或動作，或取 AWS CLI 得原則詳細資料。

目錄

在映像產生器主控台中檢視生命週期原則詳

Image Builder 主控台中的影像詳細資料頁面包含摘要區段，其中包含分組成索引標籤的其他資訊。頁面標題是建立映像檔之配方的名稱和建置版本。

主控台詳細資料區段和標

- 摘要章節
- 規則標籤
- 範圍標籤

- RunLog 標籤

摘要章節

摘要區段會橫跨頁面寬度，並包含下列詳細資訊。這些詳細信息始終顯示。

策略狀態

原則是作用中或非作用中。

類型

當您建立新映像版本 (AMI 或容器映像檔) 時，映像產生器散佈的輸出影像類型。

Date created (建立日期)

建立生命週期原則時的時間戳記。

修改日期

上次更新生命週期原則的時間。

上次執行日期

上次執行生命週期原則的時間。

IAM 角色

Image Builder 用來執行生命週期動作的 IAM 角色。

ARN

生命週期政策資源的 Amazon 資源名稱 (ARN)。

Description

生命週期原則的描述 (如果已輸入)。

規則標籤

[規則] 索引標籤會顯示您為檢視的原則設定的生命週期規則。此標籤包含下列詳細資訊：

- 名稱 — 規則的名稱。根據您可以設定的策略動作，這些名稱是靜態的。
  - `Deprecation rule`

  - Disable rule
  - Deletion rule
- 規則 — 針對規則配置之動作的簡短描述。
- 規則條件 — 列出關聯資源處理、規則例外狀況以及保留設定 (如果適用) 的組態。

如需規則組態的詳細資訊，請參閱生命週期規則的運作。

範圍標籤

[範圍] 索引標籤會顯示針對您正在檢視的策略所設定的資源選取準則。此標籤包含下列詳細資訊：

- 篩選：#### — 您用來定義範圍的篩選類型。篩選器類型可以是下列其中一種：
  - recipes— 用來建立生命週期原則所套用映像的配方。
  - tags— Image Builder 用來選取生命週期原則所套用之影像資源的一組標籤。
- 搜尋列 — 您可以依 [名稱] 篩選清單，以簡化標籤中顯示的結果。
- 名稱 — 每一列都包含您為篩選條件配置的名稱或標籤。
- 版本 — 如果您已設定配方篩選器，Image Builder 會顯示配方版本。

RunLog 標籤

每次針對已設定的資源執行原則時，Image Builder 都會儲存執行階段詳細資料。資料表中的每一列都代表單一執行階段執行個體。此標籤包含下列詳細資訊：

- 執行 ID — 識別生命週期原則執行個體。
- 執行狀態 — 報告策略動作目前是否正在執行、執行成功、失敗或已取消的執行階段狀態。
- 影響資源 — 指出執行階段執行個體是否識別生命週期動作的任何影像資源。
- 開始日期 — 執行階段執行個體啟動時的時間戳記。
- 結束日期 — 執行階段執行個體結束時的時間戳記。

建立生命週期原

當您建立新的 EC2 Image Builder 生命週期政策時，組態取決於該政策適用的映像類型。針對 AMI 映像資源和容器映像資源建立生命週期原則的 API 動作與 (CreateLifecyclePolicy) 相同。但是，映像資源和相關資源的配置不同。本節說明如何為兩者建立生命週期管理原則。

> **Note**
>
> 建立生命週期原則之前，請確定您已符合所有規定必要條件。

建立映像產生器 AMI 映像資源的生命週期管理原

您可以使用下列其中一種方法，透過 AWS Management Console 或建立 AMI 映像生命週期原則 AWS CLI。您也可以使用 CreateLifecyclePolicyAPI 動作。對於相關聯的 SDK 請求，您可以參閱 EC2 Image Builder API 參考中該命令的「另請參閱」連結。

AWS Management Console

若要在中建立 AMI 映像資源的生命週期原則 AWS Management Console，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導覽窗格中選擇「生命週期政策」。
3. 選擇建立生命週期原則。
4. 設定下列程序中所述的原則設定。
5. 若要在設定完設定後建立生命週期原則，請選擇 [建立原則]。

設定原則的一般設定。

1. 從 [原則類型] 中選取 AMI 選項。
2. 輸入策略名稱。
3. 選擇性地輸入生命週期原則的「說明」。
4. 依預設，「啟用」處於開啟狀態。預設設定會啟動生命週期原則，並立即將其新增至排程。若要建立最初停用的原則，您可以關閉 [啟用]。
5. 選取您為生命週期政策許可建立的 IAM 角色。如果您尚未建立此角色，請參閱以必要條件取得更多資訊。

設定原則的規則範圍。

本節會根據您使用的篩選器類型，設定生命週期原則的資源選取項目。

1. 篩選類型：配方 — 若要根據建立影像資源的方案將生命週期規則套用至映像資源，請為策略選取最多 50 個方案版本。

2. 篩選類型：標籤 — 若要根據資源標籤將生命週期規則套用至映像資源，請輸入最多 50 個金鑰值配對的清單，以符合策略。

開啟下列一或多個生命週期規則，以套用至生命週期原則選取的資源。如果資源在策略執行時與多個生命週期規則相符，則 Image Builder 會按以下順序執行規則動作：1) 棄用、2) 停用、3) 刪除。

取代規則

將 Image Builder 像資源狀態設定為Deprecated。Image Builder 管線仍會針對已取代的映像執行您可以選擇性地設定關聯 AMI 的棄用時間，而不會影響啟動新執行個體的能力。

- 單位計數 — 指定在建立影像資源之後必須經過之期間的整數值，然後才會將其標記為Deprecated。
- 單位 — 選取要使用的時間範圍。範圍可以是DaysWeeks、Months、或Years。
- 棄用 AMI — 選取核取方塊以將關聯的 Amazon EC2 AMI 標示為棄用日期。AMI 仍然可用，您仍然可以從中啟動新的執行個體。

停用規則

將 Image Builder 像資源狀態設定為Disabled。這樣可以防止 Image Builder 管線針對此映像執行。您可以選擇性地停用相關聯的 AMI，以防止新執行個體啟動。

- 單位計數 — 指定在建立影像資源之後必須經過之期間的整數值，然後才會將其標記為Disabled。
- 單位 — 選取要使用的時間範圍。範圍可以是DaysWeeks、Months、或Years。
- 停用 AMI — 選取核取方塊以停用相關聯的 Amazon EC2 AMI。您無法再使用 AMI 或從中啟動新執行個體。

刪除規則

按年齡或計數刪除圖像資源。您可以定義符合您需求的閾值。當 Image Builder 像資源通過臨界值時，就會移除該臨界值。您可以選擇性地取消註冊相關的 AMI 或刪除這些 AMI 的快照。您也可以為要保留超過閾值的資源指定標籤。

當您設定依年齡刪除規則時，Image Builder 會在您設定的一段時間後刪除映像資源。例如，在 6 個月後刪除影像資源。當您依計數進行設定時，Image Builder 會保留您指定的最新影像數目，或盡可能接近該數字，並刪除舊版。

- 按年齡
  - 單位計數 — 指定在建立影像資源之後必須經過之期間的整數值，然後才會刪除影像資源。
  - 單位 — 選取要使用的時間範圍。範圍可以是`Days`、`Weeks`、`Months`、或`Years`。
  - 每個方案至少保留一個映像 — 選取此核取方塊可為此規則影響的每個方案版本保留最新的可用映像資源。

  按計數
  - 影像計數 — 指定每個配方版本要保留的最近影像資源數目的整數值。
- 取消註冊 AMI — 選取核取方塊以取消註冊關聯的 Amazon EC2 AMI。您無法再使用 AMI 或從中啟動新執行個體。
- 使用相關標籤保留影像、AMI 和快照 — 選取核取方塊以輸入您要保留之影像資源的標籤清單。標籤適用於映像資源和 Amazon EC2 AMI。您最多可以輸入 50 個金鑰值對。

標籤 (選用)

將標籤新增至您的生命週期原則。

AWS CLI

若要建立新的映像產生器生命週期原則，您可以使用中的create-lifecycle-policy AWS CLI指令。

為映像產生器容器映像檔資源建立生命週期管理

您可以使用下列其中一種方法，透過 AWS Management Console 或建立容器映像生命週期原則 AWS CLI。您也可以使用 CreateLifecyclePolicyAPI 動作。對於相關聯的 SDK 請求，您可以參閱 EC2 Image Builder API 參考中該命令的「另請參閱」連結。

AWS Management Console

若要在中建立容器映像資源的生命週期政策 AWS Management Console，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導覽窗格中選擇「生命週期政策」。
3. 選擇建立生命週期原則。
4. 設定下列程序中所述的原則設定。
5. 若要在設定完設定後建立生命週期原則，請選擇 [建立原則]。

原則組態：一般設定

設定原則的一般設定。

1. 從 [原則類型] 中選取 AMI 選項。
2. 輸入策略名稱。
3. 選擇性地輸入生命週期原則的「說明」。
4. 依預設，「啟用」處於開啟狀態。預設設定會啟動生命週期原則，並立即將其新增至排程。若要建立最初停用的原則，您可以關閉 [啟用]。
5. 選取您為生命週期政策許可建立的 IAM 角色。如果您尚未建立此角色，請參閱以必要條件取得更多資訊。

設定原則的規則範圍。

本節會根據您使用的篩選器類型，設定生命週期原則的資源選取項目。

1. 篩選類型：配方 — 若要根據建立影像資源的方案將生命週期規則套用至映像資源，請為策略選取最多 50 個方案版本。
2. 篩選類型：標籤 — 若要根據資源標籤將生命週期規則套用至映像資源，請輸入最多 50 個金鑰值配對的清單，以符合策略。

刪除規則

對於容器映像檔，此規則會刪除映 Image Builder 容器映像資源。您可以選擇性地移除散佈至 ECR 儲存庫的 Docker 映像檔，以防止它們用來執行新的容器。

當您設定依年齡刪除規則時，Image Builder 會在您設定的一段時間後刪除映像資源。例如，在 6 個月後刪除影像資源。當您依計數進行設定時，Image Builder 會保留您指定的最新影像數目，或盡可能接近該數字，並刪除舊版。

- 按年齡
  - 單位計數 — 指定在建立影像資源之後必須經過之期間的整數值，然後才會刪除影像資源。
  - 單位 — 選取要使用的時間範圍。範圍可以是`Days`、`Weeks`、`Months`、或`Years`。
  - 保留至少一個映像 — 選取核取方塊以僅保留此規則影響之每個方案版本的最新可用映像資源。

按計數

- 影像計數 — 指定每個配方版本要保留的最近影像資源數目的整數值。

- 刪除 ECR 容器映像 — 選取核取方塊可刪除儲存在 ECR 存放庫中的關聯容器映像。您不能再使用容器映像檔做為建立新映像檔的基礎，或執行新的容器。
- 保留具有關聯標籤的影像 — 選取核取方塊以輸入您要保留之影像資源的標籤清單。

標籤 (選用)

將標籤新增至您的生命週期原則。

AWS CLI

若要建立新的映像產生器生命週期原則，您可以使用中的create-lifecycle-policy AWS CLI指令。

EC2 Image Builder 映像資源的生命週期管理規則如何運作

映像生命週期原則會使用您定義的生命週期規則來實作整體資源管理策略。您定義的規則有助於確保可用映像檔的新鮮度，並將基礎架構的成本降至最低，例如輸出 AMI 的快照儲存、ECR 儲存庫儲存和容器映像的資料傳輸率。

您可以為策略配置以下類型的規則。

取代規則

將 Image Builder 像資源狀態設定為`Deprecated`。Image Builder 管線仍會針對已取代的映像執行您可以選擇性地設定關聯 AMI 的棄用時間，而不會影響啟動新執行個體的能力。

當 AMI 被棄用時，它被一般搜索忽略。例如，如果您在中執行 Amazon EC2 describe-images 命令 AWS CLI，它將不會在結果集中傳回已停用的 AMI。但是，您仍然可以使用其 AMI ID 找到已過時的 AMI。

此規則不適用於容器映像。

停用規則

將 Image Builder 像資源狀態設定為`Disabled`。這樣可以防止 Image Builder 管線針對此映像執行。您可以選擇性地停用相關聯的 AMI，以防止新執行個體啟動。

當 AMI 被禁用時，它將變為私有，並且無法用於啟動新的實例。如果您與任何帳戶、組織或組織單位共用 AMI，則當 AMI 變為私人時，他們將無法存取您的 AMI。

此規則不適用於容器映像。

刪除規則

按年齡或計數刪除圖像資源。您可以定義符合您需求的閾值。當 Image Builder 像資源通過臨界值時，就會移除該臨界值。您可以選擇性地取消註冊相關的 AMI 或刪除這些 AMI 的快照。您也可以為要保留超過閾值的資源指定標籤。

對於容器映像，此規則會刪除 Image Builder 容器映像資源。您可以選擇性地移除散佈至 ECR 儲存庫的容器映像檔，以防止它們用來執行新容器。

目錄

排除規則 (API/SDK /CLI)

下列排除規則定義 AMI 生命週期規則的例外。符合排除規則指定條件的 AMI 會從生命週期動作中排除。中不提供排除規則 AWS Management Console。

下列術語使用LifecyclePolicyDetailExclusionRules資料類型中的 API 標記法。

排除規則

阿米斯

包含下列清單中LifecyclePolicyDetailExclusionRulesAmis所示的設定。

標籤映射

您可以提供最多 50 個標籤的清單，這些標籤可針對任何類型的資源略過生命週期動作。

下列術語使用LifecyclePolicyDetailExclusionRulesAmis資料類型中的 API 標記法。

AMI 排除規則

isPublic

設定是否將公用 AMI 排除在生命週期動作之外。

上次推出

指定 Image Builder 的組態詳細資料，以便從生命週期動作中排除最新的資源。

區域

從生命週期動作中排除的配置 AWS 區域 。

共享賬戶

指定 AWS 帳戶 從生命週期動作中排除的資源。

標籤映射

列示應從具有這些標籤的 AMI 的生命週期動作中排除的標籤。

檢視原則的生命週期管理規則詳細資料

規則是在您為映像產生 Image Builder 資源建立的生命週期管理原則中定義的。在主控台中，生命週期原則詳細資料頁面會顯示您為策略設定的規則詳細資料。規則標籤

若要取得中的原則詳細資訊 AWS CLI，您可以執行get-lifecycle-policy命令。回應中的策略詳細資料包含您為策略定義的動作 (規則) 清單，其中包括您所設定的所有設定。

管理 EC2 Image Builder 映像的建置和測試工作流程

映像工作流程定義 EC2 Image Builder 在映像建立程序的建置和測試階段期間執行的步驟順序。這是整體 Image Builder 工作流程架構的一部分。

影像工作流程優

- 透過影像工作流程，您可以更靈活地掌握影像建立程序，以及更多的可視性和控制權。
- 您可以在定義工作流程文件時新增自訂的工作流程步驟，也可以選擇使用 Image Builder 預設工作流程。
- 您可以排除預設影像工作流程中包含的工作流程步驟。
- 您可以建立完全略過建置程序的僅測試工作流程。您也可以執行相同的動作來建立僅建置的工作流程。

> **Note**
>
> 您無法修改現有的工作流程，但可以複製或建立新版本。

工作流程框架：階段

若要自訂影像工作流程，請務必瞭解構成映像建立工作流程架構的工作流程階段。

映像建立工作流程架構包含下列兩個不同階段。

1. 建置階段 (快照前) — 在建置階段，您可以變更執行基本映像的 Amazon EC2 建置執行個體，以建立新映像的基準。例如，您的方案可以包含安裝應用程式或修改作業系統防火牆設定的元件。

   順利完成此階段之後，Image Builder 會建立快照或容器映像，供測試階段及其後使用。

2. 測試階段 (快照後) — 在測試階段，建立 AMI 的映像與容器映像之間會有一些差異。對於 AMI 工作流程，Image Builder 會從建置階段的最後一個步驟所建立的快照啟動 EC2 執行個體。測試會在新執行個體上執行，以驗證設定並確保執行個體如預期般運作。對於容器工作流程，測試會在用於建置的相同執行個體上執行。

工作流程架構也包含發佈階段。不過，Image Builder 會處理該階段的工作流程。

服務存取

若要執行映像工作流程，Image Builder 需要執行工作流程動作的權限 您可以指定AWSServiceRoleForImageBuilder服務連結角色，也可以指定自己的服務存取角色，如下所示。

- 主控台 — 在管線精靈步驟 3 定義映像建立程序中，從「服務存取」面板的 IAM 角色清單中選取服務連結角色或您自己的自訂角色。
- I@@ mage Builder API — 在CreateImage動作請求中，指定服務連結角色或您自己的自訂角色做為`executionRole`參數的值。

若要進一步瞭解如何建立服務角色，請參閱《AWS Identity and Access Management 使用指南》中的建立角色以將權限委派給 AWS 服務。

目錄

列出影像工作流

在 Image Builder 主控台的 [映像工作流程清單] 頁面上，您可以取得您擁有或可存取的映像工作流程資源清單，以及有關這些資源的一些重要詳細資料。您也可以 AWS CLI 將指令或動作與 Image Builder API、SDK 搭配使用，或列出帳戶中的影像工作流程。

您可以使用下列其中一種方法列出您擁有或可存取的影像工作流程資源。如需 API 動作的相關資訊，請參閱 EC2 Image Builder API 參考ListWorkflows中的。如需相關聯的 SDK 要求，請參閱同一頁面上的「另請參閱」連結。

Console

工作流詳細

[映像 Image Builder 生器] 主控台中 [映像工作流程清單] 頁面上的詳細資料包括：

- 工作流程 — 影像工作流程資源的最新版本名稱。在 Image Builder 主控台中，工作流程欄會連結至工作流程詳細資料頁面。
- 版本 — 影像工作流程資源的最新版本。
- 「類型」— 工作流程類型：`BUILD`或`TEST`。
- 「所有者」— 工作流程資源的所有者。
- 建立時間 — Image Builder 建立最新版本的影像工作流程資源時的日期和時間。
- ARN — 圖像工作流程資源目前版本的 Amazon 資源名稱 (ARN)。

列出影像工作流

若要在映像產生器主控台中列出影像工作流程資源，請執行下列步驟：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導覽窗格中選擇 [影像] 工作流程。

篩選結果

在 [影像工作流程清單] 頁面上，您可以搜尋特定的影像工作流程來篩選結果。下列篩選器適用於影像工作流程：

`Workflow`

您可以輸入全部或部分工作流程名稱以簡化結果。默認設置是顯示列表中的所有工作流程。

`Version`

您可以輸入全部或部分版本號碼以簡化結果。預設為顯示清單中的所有版本。

`Type`

您可以按工作流程類型進行篩選，也可以檢視所有類型。默認設置是顯示列表中的所有工作流程類型。

- BUILD
- 測試

`Owner`

當您從搜尋列選取擁有者篩選器時，Image Builder 會顯示帳戶中影像工作流程的擁有者清單。您可以從清單中選取擁有者以簡化結果。預設為顯示清單中的所有擁有者。

- AWS 帳戶— 擁有工作流程資源的帳號。
- Amazon — Amazon 擁有和管理的工作流程資源。

AWS CLI

當您執行中的list-workflows指令時 AWS CLI，您可以取得您擁有或有權存取的影像工作流程清單。

以下指令範例展示如何使用不含篩選的list-workflows指令來列示您擁有或可存取的所有 Image Builder 影像工作流程資源。

範例：列出所有影像工作流程

```
aws imagebuilder list-workflows
```

輸出：

```
{
 "workflowVersionList": [
  {
    "name": "example-test-workflow",
```

```
    "dateCreated": "2023-11-21T22:53:14.347Z",
    "version": "1.0.0",
    "owner": "111122223333",
    "type": "TEST",
    "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/test/example-test-
workflow/1.0.0"
  },
  {
    "name": "example-build-workflow",
    "dateCreated": "2023-11-20T12:26:10.425Z",
    "version": "1.0.0",
    "owner": "111122223333",
    "type": "BUILD",
    "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/example-
build-workflow/1.0.0"
  }
 ]
}
```

執行list-workflows命令時，您可以套用篩選器來簡化結果，如下列範例所示。若要取得有關如何篩選結果的更多資訊，請參閱《指令參考》中的 list 工作流程指AWS CLI 令。

範例：組建工作流程的篩選器

```
aws imagebuilder list-workflows --filters name="type",values="BUILD"
```

輸出：

```
{
 "workflowVersionList": [
  {
    "name": "example-build-workflow",
    "dateCreated": "2023-11-20T12:26:10.425Z",
    "version": "1.0.0",
    "owner": "111122223333",
    "type": "BUILD",
    "arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/example-
build-workflow/1.0.0"
  }
 ]
}
```

建立影像工作流程

建立映像工作流程時，您可以更好地控制映像檔建立程序。您可以指定 Image Builder 建立映像時執行的工作流程，以及在測試映像時執行的工作流程。您也可以指定客戶管理的金鑰來加密工作流程資源。若要進一步瞭解工作流程資源的加密，請參閱EC2 Image Builder 中的加密和金鑰管理。

對於映像建立，您可以指定一個建置階段工作流程，以及一或多個測試階段工作流程。您甚至可以根據需要完全跳過構建或測試階段。您可以設定工作流程在工作流程使用的 YAML 定義文件中採取的動作。如需 YAML 文件語法的詳細資訊，請參閱建立 YAML 工作流程文件。

如需建立新組建或測試工作流程的步驟，請選取符合您將使用之環境的索引標籤。

AWS Management Console

您可以使用下列程序在 Image Builder 主控台中建立新的工作流程。

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 從導覽窗格中選擇 [影像] 工作流程。這會顯示您的帳戶擁有或可存取的映像工作流程清單。

   > **Note**
   >
   > 您一律會在清單中看到 Image Builder 用於其預設工作流程的 Amazon 受管工作流程資源。要查看這些工作流程的詳細信息，您可以選擇「工作流程」鏈接。

3. 若要建立新的工作流程，請選擇 [建立映像工作流程 這會顯示 [建立映像] 工作流程頁面。
4. 設定新工作流程的詳細資料。若要建立組建工作流程，請選取表單頂端附近的 [建置] 選項。若要建立測試工作流程，請選取表單頂端附近的「測試」選項。「Image Builder」會根據此選項填入「範本」清單。所有其他步驟對於構建和測試工作流程都是相同的。

   一般

   [一般] 區段包含套用至工作流程資源的設定，例如名稱和描述。一般設定包括下列項目：

   - 影像工作流程名稱 (必要) — 影像工作流程的名稱。該名稱在您的帳戶中必須是唯一的。名稱最多可包含 128 個字元。有效字元包括字母、數字-、空格和_。
   - 版本 (必要) — 要建立之工作流程資源的語意版本 (主要 .minor. patch)。
   - 描述 (選擇性) — 選擇性地新增工作流程的說明。
   - KMS 金鑰 (選用) — 您可以使用客戶受管金鑰加密工作流程資源。如需詳細資訊，請參閱 使用客戶管理的金鑰加密影像工作流程。

定義文件

YAML 工作流程文件包含工作流程的所有設定。

開始使用

- 若要以 Image Builder 預設範本作為工作流程的基準開始，請選取「從範本開始」選項。預設會選取此選項。從「範本」(T emplates) 清單中選擇要使用的範本後，這會將您選取的範本中的預設組態複製到新工作流程文件的「內容」(Content) 中，您可以在其中進行變更。
- 若要從頭開始定義工作流程文件，請選取「從頭開始」選項。這會將文件格式中某些重要部分的簡短輪廓填入「內容」，以協助您開始使用。

「內容」面板底部的狀態列會顯示 YAML 文件的警告或錯誤。如需如何建立 YAML 工作流程文件的相關資訊，請參閱建立 YAML 工作流程文件。

5. 完成工作流程後，或者如果您想要儲存進度並稍後再回來，請選擇 [建立工作流程]。

AWS CLI

在中執行命create-workflow令之前 AWS CLI，您必須建立包含工作流程之所有設定的 YAML 文件。如需詳細資訊，請參閱 建立 YAML 工作流程文件。

下列範例顯示如何使用 [建立工作流程] 指令來建立組建工作流 AWS CLI 程。此--data參數是指包含您建立之工作流程之組建組態的 YAML 文件。

範例：建立工作流程

```
aws imagebuilder create-workflow --name example-build-workflow --semantic-
version 1.0.0 --type BUILD --data file://example-build-workflow.yml
```

輸出：

```
{
"workflowBuildVersionArn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
build/example-build-workflow/1.0.0/1",
"clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

下列範例說明如何使用 [建立工作流程] 指令建立 AWS CLI 測試工作流程。此--data參數是指包含您建立之工作流程之組建組態的 YAML 文件。

範例：建立測試工作流程

```
aws imagebuilder create-workflow --name example-test-workflow --semantic-
version 1.0.0 --type TEST --data file://example-test-workflow.yml
```

輸出：

```
{
"workflowBuildVersionArn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/
test/example-test-workflow/1.0.0/1",
"clientToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

建立 YAML 工作流程文件

YAML 格式定義文件會針對映像建置程序的建置和測試階段設定輸入、輸出和工作流程步驟。您可以從包含標準化步驟的範本開始，也可以從頭開始定義自己的工作流程。無論您是使用範本還是從頭開始，都可以自訂工作流程以符合您的需求。

YAML 工作流程文件的結構

映像產生器用來執行映像建置和測試動作的 YAML 工作流程文件結構如下。

- 識別
- 輸入參數
- 步驟
- 輸出

識別

唯一識別工作流程。此區段可包含下列屬性。

欄位	Description (描述)	Type	必要

欄位	Description (描述)	Type	必要
name	工作流程文件的名稱。	字串	否
description	文件描述。	字串	否
schemaVersion	文件結構描述版本，目前為 1.0。	字串	是

範例

```
---
name: sample-test-image
description: Workflow for a sample image, with extra configuration options exposed
 through workflow parameters.
schemaVersion: 1.0
```

輸入參數

工作流程文件的這個部分定義了呼叫者可以指定的輸入參數。如果您沒有任何參數，則可以將此部分退出。如果您確實指定了參數，則每個參數都可以包含下列屬性。

欄位	Description (描述)	Type	必要	限制
name	參數名稱。	字串	是	
description	參數描述。	字串	否	
預設	如果未提供任何值，則為參數的預設值。如果您未在參數定義中	與參數資料類型相符。	否	

欄位	Description (描述)	Type	必要	限制
	包含預設值，則在執行階段需要參數值。			
type	參數的資料類型。如果您未在參數定義中包含資料類型，則參數類型會預設為執行階段所需的字串值。	字串	是	參數的資料類型必須是下列其中一種： • `string` • `integer` • `boolean` • `stringList`

範例

在工作流程文件中指定參數。

```
parameters:
  - name: waitForActionAtEnd
    type: boolean
    default: true
    description: "Wait for an external action at the end of the workflow"
```

在工作流程文件中使用參數值。

```
$.parameters.waitForActionAtEnd
```

步驟

為工作流程指定最多 15 個步驟動作。步驟會依照在工作流程文件中定義的順序執行。在失敗的情況下，復原會以相反的順序執行，從失敗的步驟開始，並向後執行先前的步驟。

每個步驟都可以參考任何先前步驟動作的輸出。這就是所謂的鏈接或引用。要引用先前步驟操作的輸出，您可以使用 JSONPath 選擇器。例如：

```
$.stepOutputs.step-name.output-name
```

如需詳細資訊，請參閱 在工作流程文件中使用動態變數。

> **Note**
>
> 即使步驟本身沒有輸出屬性，步驟動作的任何輸出都會包含在`stepOutput`步驟中。

每個步驟都可以包含下列屬性。

欄位	Description (描述)	Type	必要	預設值	限制
動作	此步驟執行的工作流程動作。	字串	是		必須是 Image Builder 工作流程文件支援的步驟動作。
`if`，後面接著一組修改`if`運算子的條件陳述式。	條件陳述式會將控制決策點的流程新增至工作流程步驟主體。	字典	否		Image Builder 支援下列條件陳述式做為`if`運算子的修飾詞： • 分支條件和修飾符：`if`、`and`、`or`、`not` 分支條件是由自己在一行上指定。 •

欄位	Description (描述)	Type	必要	預設值	限制
					比較運算子：booleanEquals numberEquals 、numberGreaterThan 、、numberGreaterThanEquals 、numberLsThan 、numberLsThanEquals 、、stringEquals 。
description	步驟說明。	字串	否		不允許使用空字串。如果包含，長度必須為 1-1024 個字元。
inputs	包含步驟動作需要執行的參數。您可以將索引鍵值指定為靜態值，或使用解析為正確資料類型的 JsonPath 變數。	字典	是		

欄位	Description (描述)	Type	必要	預設值	限制
name	步驟的名稱。此名稱在工作流程文件中必須是唯一的。	字串	是		長度必須介於 3-128 個字元之間。 可以包含英數字元和_. 沒有空格。

欄位	Description (描述)	Type	必要	預設值	限制
onFailure	設定步驟失敗時要採取的動作，如下所示。 Behavior (行為) • Abort—步驟失敗、工作流程失敗，且在失敗的步驟之後不會執行任何剩餘步驟。如果啟用倒回，倒回會從失敗的步驟開始，並繼續執行，直到所有允許復原的步驟都復原為止。 • Continue—失敗步驟，但會在失敗的步驟之後繼續執行剩餘步驟。在這種情況	字串	否	Abort	Abort | Continue

欄位	Description (描述)	Type	必要	預設值	限制
	下，沒有復原。				
啟用復原	設定如果發生失敗，是否要倒回步驟。您可以使用靜態布林值或解析為布林值的動態 JSONPath 變數。	Boolean	否	`true`	`true` | `false` | 或解析為真或假的 JSONPath 變量。
timeoutSeconds	如果重試適用，則步驟在失敗和重試之前執行的時間上限 (以秒為單位)。	Integer	否	取決於為步驟動作定義的預設值 (如果適用)。	介於 1-86400 秒之間 (最多 24 小時)

範例

```
steps:
  - name: LaunchTestInstance
    action: LaunchInstance
    onFailure: Abort
    inputs:
      waitFor: "ssmAgent"

  - name: ApplyTestComponents
    action: ExecuteComponents
    onFailure: Abort
    inputs:
      instanceId.$: "$.stepOutputs.LaunchTestInstance.instanceId"
```

```
  - name: TerminateTestInstance
    action: TerminateInstance
    onFailure: Continue
    inputs:
      instanceId.$: "$.stepOutputs.LaunchTestInstance.instanceId"

  - name: WaitForActionAtEnd
    action: WaitForAction
    if:
      booleanEquals: true
      value: "$.parameters.waitForActionAtEnd"
```

輸出

定義工作流程的輸出。每個輸出都是指定輸出名稱和值的索引鍵值配對。您可以使用輸出在執行時期匯出後續工作流程可以使用的資料。此區段為選用。

您定義的每個輸出都包含下列屬性。

欄位	Description (描述)	Type	必要
name	輸出的名稱。該名稱在您包含在管道中的工作流程中必須是唯一的。	字串	是
value	輸出的值。字串的值可以是 dyanmic 變數，例如步驟動作的輸出檔案。如需詳細資訊，請參閱 在工作流程文件中使用動態變數。	字串	是

範例

使用步驟輸出的步驟，為工作流程文件建立輸出`createProdImage`影像 ID。

```
outputs:
  - name: 'outputImageId'
    value: '$.stepOutputs.createProdImage.imageId'
```

請參閱下一個工作流程中的工作流程輸出。

```
$.workflowOutputs.outputImageId
```

工作流程文件支援的步驟動作

本節包含 Image Builder 支援之步驟動作的詳細資訊。

本節中使用的術語

AMI

Amazon Machine Image

ARN

Amazon 資源名稱

支援的動作

- BootstrapInstanceForContainer
- CollectImageMetadata
- CollectImageScanFindings
- CreateImage
- ExecuteComponents
- LaunchInstance
- RunCommand
- RunSysPrep
- SanitizeInstance
- TerminateInstance
- WaitForAction

BootstrapInstanceForContainer

此步驟動作會執行服務指令碼，以最低需求啟動執行個體，以執行容器工作流程。Image Builder 使用 Systems Manager API sendCommand 中的來執行此指令碼。如需詳細資訊，請參閱AWS Systems Manager 執行命令。

> **Note**
>
> 啟動程序指令碼會安裝 AWS CLI 和 Docker 套件，這些套件是 Image Builder 成功建置 Docker 容器的先決條件。如果您未包含此步驟動作，映像檔建置可能會失敗。

預設逾時：60 分鐘

復原：此步驟動作沒有復原。

輸入：下表包含此步驟動作的支援輸入。

輸入名稱	Description (描述)	Type	必要	預設	限制
instanceId	要啟動的執行個體識別碼。	字串	是		這必須是針對此工作流程啟動實例的工作流程步驟中的輸出實例 ID。

輸出：下表包含此步驟動作的輸出。

輸出名稱	Description (描述)	Type
runCommandId	在執行個體上執行啟動程序指令碼的 Systems Manager sendCommand 識別碼。	字串
status	Systems Manager 傳回的狀態sendCommand。	字串

輸出名稱	Description (描述)	Type
output	從 Systems Manager 傳回的輸出sendCommand。	字串

範例

在工作流程文件中指定步驟動作。

```
- name: ContainerBootstrapStep
  action: BootstrapInstanceForContainer
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

在工作流程文件中使用步驟動作值的輸出。

```
$.stepOutputs.ContainerBootstrapStep.status
```

CollectImageMetadata

此步驟動作僅適用於建置工作流程。

EC2 Image Builder 會在其啟動的 EC2 執行個體上執行 AWS Systems Manager (Systems Manager) 代理程式，以建置和測試映像。Image Builder 會透過「Systems Manager 詳細目錄」收集在建置階段使用之執行個體的其他資 此資訊包括作業系統 (OS) 名稱和版本，以及您作業系統所報告的套件清單及其個別版本。

> **Note**
>
> 此步驟動作僅適用於建立 AMI 的映像。

預設逾時：30 分鐘

復原：Image Builder 會回復在此步驟中建立的所有 Systems Manager 資源。

輸入：下表包含此步驟動作的支援輸入。

輸入名稱	Description (描述)	Type	必要	預設	限制
instanceId	要套用中繼資料設定的建置執行個體。	字串	是		這必須是針對此工作流程啟動組建實例的工作流程步驟中的輸出實例 ID。

輸出：下表包含此步驟動作的輸出。

輸出名稱	Description (描述)	Type
osVersion	從組建執行個體收集的作業系統名稱和版本。	字串
關聯識別碼	用於庫存收集的 Systems Manager 關聯 ID。	字串

範例

在工作流程文件中指定步驟動作。

```
- name: CollectMetadataStep
  action: CollectImageMetadata
  onFailure: Abort
  inputs:
      instanceId: $.stepOutputs.LaunchStep.instanceId
```

使用工作流程文件中步驟動作的輸出。

```
$.stepOutputs.CollectMetadataStep.osVersion
```

CollectImageScanFindings

如果您的帳戶已啟用 Amazon Inspector，且已為管道啟用映像掃描，則此步驟動作會收集 Amazon Inspector 針對您的測試執行個體報告的映像掃描結果。此步驟動作不適用於建置工作流程。

預設逾時時間：120 分鐘

復原：此步驟動作沒有復原。

輸入：下表包含此步驟動作的支援輸入。

輸入名稱	Description (描述)	Type	必要	預設	限制
instanceId	執行掃瞄之執行個體的 ID。	字串	是		這必須是針對此工作流程啟動實例的工作流程步驟中的輸出實例 ID。

輸出：下表包含此步驟動作的輸出。

輸出名稱	Description (描述)	Type
runCommandId	執行指令碼以收集發現項目的 Systems Manager sendCommand 識別碼。	字串
status	Systems Manager 傳回的狀態sendCommand。	字串
output	從 Systems Manager 傳回的輸出sendCommand。	字串

範例

在工作流程文件中指定步驟動作。

```
- name: CollectFindingsStep
  action: CollectImageScanFindings
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

在工作流程文件中使用步驟動作值的輸出。

```
$.stepOutputs.CollectFindingsStep.status
```

CreateImage

此步驟動作會使用 Amazon EC2 `CreateImage` API 從執行中的執行個體建立映像檔。在建立過程中，步驟動作會視需要等待，以驗證資源是否已達到正確的狀態，然後再繼續進行。

預設逾時：720 分鐘

復原：此步驟動作沒有復原。

輸入：下表包含此步驟動作的支援輸入。

輸入名稱	Description (描述)	Type	必要	預設	限制
instanceId	要從中建立新映像檔的執行個體。	字串	是		此步驟啟動時，所提供執行個體 ID 的執行個體必須處於 `running`狀態。

輸出：下表包含此步驟動作的輸出。

輸出名稱	Description (描述)	Type
imageId	所建立之影像的 AMI 識別碼。	字串

範例

在工作流程文件中指定步驟動作。

```
- name: CreateImageFromInstance
  action: CreateImage
  onFailure: Abort
  inputs:
      instanceId.$: "i-1234567890abcdef0"
```

在工作流程文件中使用步驟動作值的輸出。

```
$.stepOutputs.CreateImageFromInstance.imageId
```

ExecuteComponents

此步驟動作會執行在正在建置的目前映像的方案中指定的元件。建置工作流程會在組建執行個體上執行組建元件 測試工作流程僅在測試實例上運行測試組件。

Image Builder 使用 Systems Manager API sendCommand 中的來執行元件。如需詳細資訊，請參閱AWS Systems Manager 執行命令。

預設逾時：720 分鐘

復原：此步驟動作沒有復原。

輸入：下表包含此步驟動作的支援輸入。

輸入名稱	Description (描述)	Type	必要	預設	限制
instanceId	元件應在其上執行的執行個體識別碼。	字串	是		這必須是針對此工作流程啟動實例的工作流程步驟中的輸出實例 ID。

輸出：下表包含此步驟動作的輸出。

輸出名稱	Description (描述)	Type
runCommandId	在執行個體上執行元件的 Systems Manager sendCommand 識別碼。	字串
status	Systems Manager 傳回的狀態sendCommand。	字串
output	從 Systems Manager 傳回的輸出sendCommand。	字串

範例

在工作流程文件中指定步驟動作。

```
- name: ExecComponentsStep
  action: ExecuteComponents
  onFailure: Abort
  inputs:
      instanceId: $.stepOutputs.LaunchStep.instanceId
```

使用工作流程文件中步驟動作的輸出。

```
$.stepOutputs.ExecComponentsStep.status
```

LaunchInstance

此步驟動作會啟動您中的執行個體， AWS 帳戶 並等待系統管理員代理程式在執行個體上執行，然後再繼續下一個步驟。啟動動作會使用與映像相關聯的方案和基礎結構組態資源中的設定。例如，要啟動的執行個體類型來自基礎結構組態。輸出是啟動之執行個體的執行個體 ID。

`waitFor`輸入會設定符合步驟完成需求的條件。

預設逾時：60 分鐘

回滾：對於構建實例，回滾會執行您在基礎結構配置資源中配置的操作。根據預設，如果映像建立失敗，建置執行個體會終止。不過，基礎結構組態中有一個設定可保留建置執行個體以進行疑難排解。

輸入：下表包含此步驟動作的支援輸入。

輸入名稱	Description (描述)	Type	必要	預設	限制
等待	完成工作流程步驟並繼續下一個步驟之前要等待的條件。	字串	是		Image Builder 目前支援`ssmAgent`。

輸出：下表包含此步驟動作的輸出。

輸出名稱	Description (描述)	Type
instanceId	啟動之執行個體的執行個體 ID。	字串

範例

在工作流程文件中指定步驟動作。

```
- name: LaunchStep
  action: LaunchInstance
  onFailure: Abort
  inputs:
    waitFor: ssmAgent
```

使用工作流程文件中步驟動作的輸出。

```
$.stepOutputs.LaunchStep.instanceId
```

RunCommand

此步驟動作會為您的工作流程執行命令文件。Image Builder 會使用系 Systems Manager API sendCommand 中的來為您執行。如需詳細資訊，請參閱AWS Systems Manager 執行命令。

預設逾時：12 小時

復原：此步驟動作沒有復原。

輸入：下表包含此步驟動作的支援輸入。

輸入名稱	Description (描述)	Type	必要	預設	限制
instanceId	執行命令文件的執行個體識別碼。	字串	是		這必須是針對此工作流程啟動實例的工作流程步驟中的輸出實例 ID。
documentName	要執行的 Systems Manager 命令文件的名稱。	字串	是		
parameters	指令文件所需之任何參數的索引鍵值對清單。	<string>字典 < 字串，清單 >	有條件		
文件版本	要執行的命令文件版本。	字串	否	$DEFAULT	

輸出：下表包含此步驟動作的輸出。

輸出名稱	Description (描述)	Type
runCommandId	在執行個體上執行命令文件的 Systems Manager sendCommand 識別碼。	字串

輸出名稱	Description (描述)	Type
status	Systems Manager 傳回的狀態sendCommand。	字串
output	從 Systems Manager 傳回的輸出sendCommand。	字串清單

範例

在工作流程文件中指定步驟動作。

```
- name: RunCommandDoc
  action: RunCommand
  onFailure: Abort
  inputs:
    documentName: SampleDocument
    parameters:
        osPlatform:
          - "linux"
    instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

在工作流程文件中使用步驟動作值的輸出。

```
$.stepOutputs.RunCommandDoc.status
```

RunSysPrep

此步驟動作會使用 Systems Manager API sendCommand 中的，在建置執行個體關閉快照集之前執行 Windows 執行個體的AWSEC2-RunSysprep文件。 這些動作會遵循強化和清理影像的AWS 最佳作法。

預設逾時：60 分鐘

復原：此步驟動作沒有復原。

輸入：下表包含此步驟動作的支援輸入。

輸入名稱	Description (描述)	Type	必要	預設	限制
instanceId	執行AWSEC2-RunSysprep 文件的執行個體識別碼。	字串	是		這必須是針對此工作流程啟動實例的工作流程步驟中的輸出實例 ID。

輸出：下表包含此步驟動作的輸出。

輸出名稱	Description (描述)	Type
runCommandId	在執行個體上執行AWSEC2-RunSysprep 文件的 Systems Manager sendCommand 識別碼。	字串
status	Systems Manager 傳回的狀態sendCommand。	字串
output	從 Systems Manager 傳回的輸出sendCommand。	字串

範例

在工作流程文件中指定步驟動作。

```
- name: RunSysprep
  action: RunSysPrep
  onFailure: Abort
  inputs:
      instanceId.$: $.stepOutputs.LaunchStep.instanceId
```

在工作流程文件中使用步驟動作值的輸出。

```
$.stepOutputs.RunSysprep.status
```

SanitizeInstance

此步驟動作會在建置執行個體關閉快照集之前，執行 Linux 執行個體的建議清理指令碼。清理指令碼有助於確保最終映像檔遵循安全性最佳作法，並且會移除不應結轉至快照集的建置成品或設定。如需指令集的詳細資訊，請參閱必要的建置後清理。此步驟動作不適用於容器映像。

Image Builder 使用 Systems Manager API sendCommand 中的來執行此指令碼。如需詳細資訊，請參閱AWS Systems Manager 執行命令。

預設逾時：60 分鐘

復原：此步驟動作沒有復原。

輸入：下表包含此步驟動作的支援輸入。

輸入名稱	Description (描述)	Type	必要	預設	限制
instanceId	要清理的執行個體 ID。	字串	是		這必須是針對此工作流程啟動實例的工作流程步驟中的輸出實例 ID。

輸出：下表包含此步驟動作的輸出。

輸出名稱	Description (描述)	Type
runCommandId	在執行個體上執行清理sendCommand指令碼的系統管理員識別碼。	字串
status	Systems Manager 傳回的狀態sendCommand。	字串

輸出名稱	Description (描述)	Type
output	從 Systems Manager 傳回的輸出sendCommand。	字串

範例

在工作流程文件中指定步驟動作。

```
- name: SanitizeStep
  action: SanitizeInstance
  onFailure: Abort
  inputs:
      instanceId: $.stepOutputs.LaunchStep.instanceId
```

在工作流程文件中使用步驟動作值的輸出。

```
$.stepOutputs.SanitizeStep.status
```

TerminateInstance

此步驟動作會使用作為輸入傳入的執行個體 ID 終止執行個體。

預設逾時：30 分鐘

復原：此步驟動作沒有復原。

輸入：下表包含此步驟動作的支援輸入。

輸入名稱	Description (描述)	Type	必要	預設	限制
instanceId	要終止的執行個體識別碼。	字串	是		

輸出：此步驟動作沒有輸出。

範例

在工作流程文件中指定步驟動作。

```
- name: TerminateInstance
  action: TerminateInstance
  onFailure: Continue
  inputs:
      instanceId.$: i-1234567890abcdef0
```

WaitForAction

此步驟動作會暫停執行中的工作流程，並等待從 Image Builder SendWorkflowStepAction API 動作接收外部動作。此步驟會將 EventBridge 事件發佈至具有詳細資料類型的預設 EventBridge 事件匯流排`EC2 Image Builder Workflow Step Waiting`。如果您提供 SNS 主題 ARN，此步驟也可以傳送 SNS 通知。

預設逾時：3 天

復原：此步驟動作沒有復原。

輸入：下表包含此步驟動作的支援輸入。

輸入名稱	Description (描述)	Type	必要	預設	限制
snsTopicArn	工作流程步驟擱置時傳送通知的選用 SNS 主題 ARN。	字串	否		

輸出：下表包含此步驟動作的輸出。

輸出名稱	Description (描述)	Type
動作	SendWorkflowStepActionAPI 動作傳回的動作。	字符串（`RESUME`或`STOP`）
reason	返回操作的原因。	字串

範例

在工作流程文件中指定步驟動作。

```
- name: SendEventAndWait
  action: WaitForAction
  onFailure: Abort
  inputs:
    snsTopicArn: arn:aws:sns:us-west-2:111122223333:ExampleTopic
```

在工作流程文件中使用步驟動作值的輸出。

```
$.stepOutputs.SendEventAndWait.reason
```

在工作流程文件中使用動態變數

您可以在工作流程文件中使用動態變數，來代表影像建立程序在執行階段不同的值。動態變數值會表示為具有唯一識別目標變數之結構節點的 JSONPath 選取器。

JSONPath 動態工作流變量結構

```
$.<document structure>.[<step name>].<variable name>
```

根 ($) 之後的第一個節點是指工作流程文件結構，例如`stepOutputs`，或在 Image Builder 系統變數的情況下，`imageBuilder`。下列清單包含支援的 JSONPath 工作流程文件結構節點。

文件結構節點

- 參數-工作流程參數
- 步驟輸出-從相同工作流程文件中的步驟輸出
- 工作流程輸出-來自已經執行的工作流程文件的輸出
- Image Builder-圖像生成器系統變量

`parameters`和`stepOutputs`文件結構節點包括步驟名稱的選用節點。這有助於確保所有步驟中的唯一變數名稱。

JSONPath 中的最後一個節點是目標變量的名稱，例如。`instanceId`

每個步驟都可以使用這些 JsonPath 動態變量引用任何先前步驟操作的輸出。這也稱為鏈接或引用。若要參照先前步驟動作的輸出，您可以使用下列動態變數。

```
$.stepOutputs.step-name.output-name
```

範例

```
- name: ApplyTestComponents
  action: ExecuteComponents
  onFailure: Abort
  inputs:
    instanceId.$: "$.stepOutputs.LaunchTestInstance.instanceId"
```

使用 Image Builder 系統變數

Image Builder 提供下列系統變數，您可以在工作流程文件中使用這些變數：

變數名稱	Description (描述)	Type	範例值
cloudWatchLog集團	輸出 CloudWatch 記錄檔的「記錄」群組名稱。 格式：/aws/imagebuilder/*<recipe-name>*	字串	/aws/imagebuilder/*sampleImageRecipe*
cloudWatchLog串流	輸出 CloudWatch 記錄檔的記錄資料流名稱。	字串	*1.0.0/1*
collectImageMetadata	指示 Image Builder 是否收集執行個體中繼資料的設定。	Boolean	true | false
collectImageScan發現	可讓映像產生器收集影像掃描發現項目的設定目前值。	Boolean	true | false

變數名稱	Description (描述)	Type	範例值
imageBuildNumber	映像檔的組建版本號碼。	Integer	*1*
imageId	基本映像檔的 AMI 識別碼。	字串	*##*
今源名	影像的名稱。	字串	*####*
imageType	影像輸出類型。	字串	`AMI | Docker`
imageVersionNumber	影像的版本號碼。	字串	*1.0.0*
instanceProfileName	Image Builder 用來啟動組建和測試執行個體的執行個體設定檔角色名稱。	字串	*SampleImageBuilderInstanceProfileRole*
平台	所建置映像檔的作業系統平台。	字串	`Linux | Windows | MacOS`
記錄	包含 Image Builder 所寫入之 S3 日誌組態的 JSON 物件。	JSON 物件	*{'S3 ##'#{'s3 '#BucketName' # ## '#'s3'#'ib-logKeyPrefix'}}*
securityGroups	套用至建置和測試執行個體的安全群組 ID。	列表 [字符串]	*[Sg -1234567890 #### 1,*

變數名稱	Description (描述)	Type	範例值
來源網絡	工作流程用於建置和測試階段的 Image Builder 映像資源的 Amazon 資源名稱 (ARN)。	字串	*ARN#AW####### us-east-1#1111 2222333###/### #/1.0.0/1*
subnetId	要啟動組建和測試執行個體的子網路識別碼。	字串	*## -1234567890 ###*
terminateInstanceOn失敗	設定的目前值，指示 Image Builder 在發生故障時終止執行個體，或保留執行個體以進行疑難排解。	Boolean	`true` | `false`
工作流程階段	為工作流程執行所執行的目前階段。	字串	`Build` | `Test`
workingDirectory	工作目錄的路徑。	字串	`/tmp`

在工作流程步驟中使用條件陳述式

條件語句以語`if`句開頭的 document 屬性。`if`陳述式的最終目的是決定要執行步驟動作還是略過它。如果`if`陳述式解析為`true`，則會執行步驟動作。如果解析為`false`，Image Builder 會略過步驟動作，並在記錄檔`SKIPPED`中記錄的步驟狀態。

該語`if`句支持分支語句（`and`，`or`）和條件修飾符（`not`）。它也支援下列比較運算子，這些運算子會根據所比較的資料類型 (字串或數字) 執行值比較 (等於、小於、大於)。

支援比較運算子

- booleanEquals
- numberEquals
- numberGreaterThan
- numberGreaterThanEquals
- numberLessThan
- numberLessThanEquals
- stringEquals

分支語句和條件修飾符的規則

下列規則適用於分支陳述式 (and,or) 和條件修飾詞 (not)。

- 分支語句和條件修飾符必須自行出現在一行上。
- 分支語句和條件修飾符必須遵循級別規則。
  - 父項層級只能有一個陳述式。
  - 每個子分支或修飾符都會啟動一個新的級別。

  如需有關圖層的詳細資訊，請參閱巢狀層級。
- 每個分支陳述式必須至少有一個子條件陳述式，但不能超過十個。
- 條件修飾符只對一個子條件語句進行操作。

巢狀層級

條件語句在自己的一個部分中的幾個級別上運行。例如，if陳述式屬性會顯示在工作流程文件中與步驟名稱和動作相同的層級。這是條件語句的基礎。

您最多可以指定四個層級的條件陳述式，但父項層次只能出現一個陳述式。所有其他分支陳述式、條件修飾詞或條件運算子都會從該處縮排，每個層級有一個縮排。

下列大綱顯示條件陳述式的巢狀層級數目上限。

```
base:
  parent:
    - child (level 2)
        - child (level 3)
```

```
            child (level 4)
```

if 屬性

該if屬性指定條件語句作為一個文檔屬性。這是零級。

父級別

這是嵌套條件語句的第一級。在這個層級只能有一個陳述式。如果您不需要分支或修飾符，則可以是沒有子語句的條件運算符。除了條件運算符之外，此級別不使用破折號表示法。

兒童水平

層級二到四被視為子層級。子陳述式可以包含分支陳述式、條件修飾詞或條件運算子。

範例：巢狀圖層

下列範例會顯示條件陳述式中層級的最大數目。

```
if:
  and:                              #first level
    - stringEquals: 'my_string'     #second level
      value: 'my_string'
    - and:                          #also second level
        - numberEquals: '1'         #third level
          value: 1
        - not:                      #also third level
            stringEquals: 'second_string'      #fourth level
            value: "diff_string"
```

巢狀規則

- 子層次的每個分支或修飾符都會啟動一個新的層次。
- 每個層級都會縮排。
- 最多可以有四個層次，包括父項層次的一個敘述句、修正因子或運算子，以及最多三個額外層次。

範例

這組示例顯示了條件語句的各個方面。

分支：和

分and支陳述式會在屬於分支子項的運算式清單上運作，所有這些運算式都必須評估為true。Image Builder 會依照運算式在清單中顯示的順序來評估運算式。如果有任何運算式評估為false，則會停止處理，並考慮分支false。

下列範例會評估為true，因為兩個運算式都評估為true。

```
if:
  and:
    - stringEquals: 'test_string'
      value: 'test_string'
    - numberEquals: 1
      value: 1
```

分支：或

分or支陳述式會在做為分支子項的運算式清單上運作，其中至少有一個必須評估為true。Image Builder 會依照運算式在清單中顯示的順序來評估運算式。如果有任何運算式評估為true，則會停止處理，並考慮分支true。

下列範例會評估為true，即使第一個運算式為false。

```
if:
  or:
    - stringEquals: 'test_string'
      value: 'test_string_not_equal'
    - numberEquals: 1
      value: 1
```

條件修飾符：不

not條件修飾符會否定做為分支子系的條件陳述式。

下列範例會評估not修飾詞否定stringEquals條件陳述式的true時間。

```
if:
  not:
    - stringEquals: 'test_string'
      value: 'test_string_not_equal'
```

條件陳述式：布林等於

booleanEquals比較運算符比較布爾值，如果布爾值完全匹配返回 true。

下列範例會判斷collectImageScanFindings是否已啟用。

```
if:
  - booleanEquals: true
    value: '$.imagebuilder.collectImageScanFindings'
```

條件語句：字符串等於

stringEquals比較運算符比較兩個字符串，如果字符串是完全匹配返回 true。如果其中一個值不是字串，Image Builder 會在比較之前將其轉換為字串。

下列範例會比較平台系統變數，以判斷工作流程是否在 Linux 平台上執行。

```
if:
  - stringEquals: 'Linux'
    value: '$.imagebuilder.Platform'
```

條件陳述式：數字

numberEquals比較運算符比較兩個數字，如果數字相等則返回 true。要比較的數字必須是下列其中一種格式。

- Integer
- Float
- 匹配以下正則表達式模式的字符串：^-?[0-9]+(\.)?[0-9]+$。

下列範例會將所有評估值比較為true。

```
if:
  # Value provider as a number
  numberEquals: 1
  value: '1'

  # Comparison value provided as a string
  numberEquals: '1'
  value: 1

  # Value provided as a string
  numberEquals: 1
  value: '1'
```

```
# Floats are supported
numberEquals: 5.0
value: 5.0

# Negative values are supported
numberEquals: -1
value: -1
```

使用 EC2 Image Builder 匯入和匯出虛擬機器 (VM) 映像

當您從虛擬化環境匯出 VM 時，該程序會建立一組或多個磁碟容器檔案，做為 VM 環境、設定和資料的快照集。您可以使用這些檔案匯入 VM，並將其用作映像配方的基本映像檔。

Image Builder 支援虛擬機器磁碟容器的下列檔案格式：

- 開放虛擬化封存 (OVA)
- 虛擬機器磁碟 (VMDK)
- 虛擬硬碟 (VHD/VHDX)
- Raw

匯入會使用磁碟建立 Amazon 機器映像 (AMI) 和映 Image Builder 資源，其中任一資源都可做為自訂映像方案的基本映像檔。虛擬機器磁碟必須存放在 S3 儲存貯體中才能匯入。或者，您也可以從現有的 EBS 快照匯入。

在 Image Builder 主控台中，您可以直接匯入映像檔，然後在方案中使用輸出影像或 AMI，也可以在建立方案或配方版本時指定匯入參數。如需有關直接匯入的更多資訊，請參閱匯入虛擬機器 (主控台)。如需有關匯入為影像配方的一部分的詳細資訊，請參閱VM 匯入組態。

將虛擬機器匯入 Image Builder (AWS CLI)

若要將虛擬機器從磁碟匯入 AMI，並建立可立即參考的映 Image Builder 映像資源，請依照下列步驟執行 AWS CLI：

1. 使用中的 Amazon EC2 VM Import/匯出import-image命令來啟動虛擬機器匯入 AWS CLI。記下命令回應中傳回的工作 ID。您將需要它來進行下一步。如需詳細資訊，請參閱《虛擬機器匯入/匯出使用者指南》中的使用虛擬機器匯入匯入為映像。

2. 建立 CLI 輸入 JSON 文件

為了簡化中使用的 Image Builder import-vm-image 命令 AWS CLI，我們建立了一個 JSON 檔案，其中包含要傳入命令的所有匯入設定。

> **Note**
>
> JSON 檔案中資料值的命名慣例遵循針對 Image Builder API 動作要求參數指定的模式。若要檢閱 API 命令要求參數，請參閱 EC2 Image Builder API 參考中的ImportVmImage命令。
>
> 若要將資料值做為指令行參數提供，請參考《指AWS CLI 令參考》中指定的參數名稱。以 Image Builder 指import-vm-image令做為選項。

以下是我們在此示例中指定的參數摘要：

- name (字串，必要) — 要建立為匯入輸出的影 Image Builder 影像資源的名稱。
- SemanticVersion <major>(字串，必要) — 輸出影像的語意版本，以下列格式指定版本，每個位置都有數值，以表示特定版本：。 <minor>。 <patch>。例如 `1.0.0`。若要進一步瞭解 Image Builder 資源的語意版本控制，請參閱語義版本控制。
- 描述（字符串）— 圖像配方的描述。
- 平台 (字串，必要) — 已匯入虛擬機器的作業系統平台。
- vmImportTask識別碼 (字串，必要) — 來自 Amazon EC2 虛擬機器匯入程序的 `ImportTaskId` (AWS CLI)。Image Builder 會監控匯入程序，以便在建立並建立 Image Builder 資源的 AMI 中，以便立即用於配方中。
- ClientToken（字符串，必需）-您提供的唯一，區分大小寫的標識符，以確保請求的冪等性。如需詳細資訊，請參閱 Amazon EC2 API 參考中的確保冪等性。
- tags (字串對映) — 標籤是附加至匯入資源的鍵值配對。最多允許 50 個鍵值對。

將檔案另存為`import-vm-image.json`，以在「Image Builder」import-vm-image 指令中使用。

```
{
    "name": "example-request",
    "semanticVersion": "1.0.0",
    "description": "vm-import-test",
    "platform": "Linux",
    "vmImportTaskId": "import-ami-01ab234567890cd1e",
```

```
    "clientToken": "asz1231231234cs3z",
    "tags": {
     "Usage": "VMIE"
    }
}
```

3. 匯入影像

   運行命import-vm-image令，並使用您創建的文件作為輸入：

```
aws imagebuilder import-vm-image --cli-input-json file://import-vm-image.json
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

從映像組建中分發 VM 磁碟 (AWS CLI)

您可以使用中的映像產生器發佈組態，將支援的 VM 磁碟格式檔案設定分發到目標區域的 S3 儲存貯體，做為一般 Image Builder 置程序的一部分 AWS CLI。如需更多詳細資訊，請參閱 建立輸出虛擬機器磁碟的散佈設定 (AWS CLI)。

共用 EC2 Image Builder 資源

EC2 Image Builder 與 AWS Resource Access Manager (AWS RAM) 整合，可讓您與任何資源 AWS 帳戶 或透過 AWS Organizations。可共用的 EC2 Image Builder 資源包括：

- 元件
- 映像
- 配方

本節提供的資訊可協助您共用這些 EC2 Image Builder 資源。

區段內容

在 EC2 Image Builder 中使用共用元件、映像和方法

元件、映像和配方共用可讓資源擁有者與其他人 AWS 帳戶 或 AWS 組織內部共用軟體組態。您可以集中管理資源共用，並定義一組可與其共用設定的帳號。

在此模型中，擁有 AWS 帳戶 該組件，圖像或配方（所有者）與其他 AWS 帳戶 （消費者）共享它。消費者可以將共用元件與其映像管線建立關聯，以自動取用對共用元件、映像或方案的更新。

元件、影像或配方擁有者可與下列人員共用這些資源：

- 具體在其組織 AWS 帳戶 內部或外部 AWS Organizations。
- 其中組織內部的組織單位 (OU) AWS Organizations。
- AWS Organizations中的整個組織。
- AWS Organizations 或組織以外的 OU。

共用元件、影像和配方的先決條件

若要共用 Image Builder 元件、影像或方案：

- 您必須擁有 AWS 帳戶. 您無法共用已與您共用的資源。
- 與加密資源關聯的 AWS Key Management Service (AWS KMS) 金鑰必須明確地與目標帳戶、組織或 OU 共用。
- 若要與 AWS Organizations 和 OU 共用您的 Image Builder 資源 AWS RAM，您必須啟用共用。如需詳細資訊，請參閱《AWS RAM 使用者指南》中的透過 AWS Organizations啟用共用。

- 如果您在不同區域的帳戶 AWS KMS 之間分配使用加密的映像，則必須在每個目標區域中建立 KMS 金鑰和別名。此外，要在這些區域中啟動執行個體的人員將需要存取透過金鑰原則指定的 KMS 金鑰。

Image Builder 從管線組建建立的下列資源不會被視為映像產生器資源，而是 Image Builder 在您的帳戶中分配的外部資源，以及您在發佈組態中指定的帳戶、組織或組織單位 (OU)。 AWS 區域

- Amazon Machine Images (AMI)
- 駐留在 Amazon ECR 中的容器映像

如需 AMI 發佈設定的詳細資訊，請參閱創建和更新 AMI 分發配置。如需 Amazon ECR 中容器映像分發設定的詳細資訊，請參閱建立和更新容器映像的發佈設定。

如需與 AWS Organizations 和 OU 共用 AMI 的詳細資訊，請參閱與組織或 OU 共用 AMI。

相關服務

AWS Resource Access Manager

元件、影像和配方共用與 AWS Resource Access Manager (AWS RAM) 整合。 AWS RAM 是一項服務，可讓您與任何 AWS 帳戶或透過共用 AWS 資源 AWS Organizations。使用 AWS RAM，您可以透過建立資源共用來共用您擁有的資源。資源共用指定要共用的資源以及與之共用資源的消費者。消費者可以是中的個人 AWS 帳戶、組織單位或整個組織 AWS Organizations。

若要取得有關的更多資訊 AWS RAM，請參閱AWS RAM 使用者指南。

跨區域共用

共用的元件、影像和配方只能在指定的 AWS 區域中共用。當您共用這些資源時，這些資源不會跨區域進行複寫。

共用元件、影像或配方

若要共用 Image Builder 元件、映像或方案，您必須將其新增至資源共用。資源共用是一 AWS RAM 種可讓您跨 AWS 帳號共用資源的資源。資源共用指定要共用的資源以及與之共用的消費者。若要將元件、映像或方案新增至新的資源共用，您必須先使用 AWS RAM 主控台建立資源共用。

如果您是組織的一員， AWS Organizations 且已啟用組織內的共用功能，則組織中的取用者會自動獲得共用元件、映像或方案的存取權。否則，取用者會收到加入資源共用的邀請，並在接受邀請後授予共用資源的存取權。

您可以使用下列選項來共用您的資源：

選項 1：建立 RAM 資源共用

建立 RAM 資源共用時，只需一個步驟即可共用您擁有的元件、映像或方案。使用下列其中一種方法建立資源共用：

- 主控台

  若要使用 AWS RAM 主控台建立資源共用，請參閱《使用指南》中的「共AWS RAM 用您擁有的 AWS 資源」。
- AWS CLI

  若要使用 AWS RAM 命令列介面建立資源共用，請執行中的create-resource-share命令 AWS CLI。

選項 2：套用資源策略並升級至 RAM 資源共用

共用資源的第二個選項涉及兩個步驟，在兩者中 AWS CLI 執行指令。第一個步驟使用中的 Image Builder 指令， AWS CLI 將以資源為基礎的政策套用至共用資源。第二個步驟會使用中的promote-resource-share-created-from-policy AWS RAM 指令將資源提升至 RAM 資源共用， AWS CLI 以確保您與之共用資源的所有主參與者都可以看到該資源。

1. 套用資源策略

   若要成功套用資源策略，您必須確定要共用的帳號具有存取任何基礎資源的權限。

   針對適用的命令，選擇符合您資源類型的標籤。

   Image

   您可以將資源策略應用於圖像，以允許其他人將其用作其配方中的基本圖像。

   執行中的 put-image-policy Image Builder 指令 AWS CLI，以識別要與之共用映像的 AWS 主參與者。

   ```
   aws imagebuilder put-image-policy --image-arn arn:aws:imagebuilder:us-
   west-2:123456789012:image/my-example-image/2019.12.03/1 --policy
    '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal":
    { "AWS": [ "123456789012" ] }, "Action": ["imagebuilder:GetImage",
    "imagebuilder:ListImages"], "Resource": [ "arn:aws:imagebuilder:us-
   west-2:123456789012:image/my-example-image/2019.12.03/1" ] } ] }'
   ```

Component

您可以將資源策略套用至組建或測試元件，以啟用跨帳戶共用。此命令授予其他帳戶在其配方中使用您的組件的權限。若要成功套用資源策略，您必須確定要共用的帳號具有存取共用元件參照之任何資源的權限，例如私人存放庫上託管的檔案。

執行中的 put-component-policy Image Builder 指令 AWS CLI，以識別要與之共用元件的 AWS 主參與者。

```
aws imagebuilder put-component-policy --component-arn arn:aws:imagebuilder:us-
west-2:123456789012:component/my-example-component/2019.12.03/1 --policy
 '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal":
 { "AWS": [ "123456789012" ] }, "Action": [ "imagebuilder:GetComponent",
 "imagebuilder:ListComponents" ], "Resource": [ "arn:aws:imagebuilder:us-
west-2:123456789012:component/my-example-component/2019.12.03/1" ] } ] }'
```

Image recipe

您可以將資源策略套用至映像配方，以啟用跨帳號共用。此指令授予其他帳戶使用您的方案在其帳戶中建立影像的權限。若要成功套用資源策略，您必須確定要共用的帳號具有存取方案參照之任何資源的權限，例如基礎映像或選取的元件。

執行中的 put-image-recipe-policy Image Builder 指令 AWS CLI，以識別要與之共用映像的 AWS 主參與者。

```
aws imagebuilder put-image-recipe-policy --image-recipe-arn
 arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-
image-recipe/2019.12.03 --policy '{ "Version": "2012-10-17", "Statement":
 [ { "Effect": "Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":
 [ "imagebuilder:GetImageRecipe", "imagebuilder:ListImageRecipes" ], "Resource":
 [ "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-image-
recipe/2019.12.03" ] } ] }'
```

Container recipe

您可以將資源策略套用至容器方案，以啟用跨帳號共用。此命令授予其他帳戶使用您的方案在其帳戶中創建圖像的權限。若要成功套用資源策略，您必須確定要共用的帳號具有存取方案參照之任何資源的權限，例如基礎映像或選取的元件。

執行中的 put-container-recipe-policy Image Builder 指令 AWS CLI，以識別要與之共用映像的 AWS 主參與者。

```
aws imagebuilder put-container-recipe-policy --container-recipe-arn
 arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-example-
container-recipe/2021.12.03 --policy '{ "Version": "2012-10-17", "Statement":
 [ { "Effect": "Allow", "Principal": { "AWS": [ "123456789012" ] }, "Action":
 [ "imagebuilder:GetContainerRecipe", "imagebuilder:ListContainerRecipes" ],
 "Resource": [ "arn:aws:imagebuilder:us-west-2:123456789012:container-recipe/my-
example-container-recipe/2021.12.03" ] } ] }'
```

> **Note**
>
> 若要設定共用和取消共用資源的正確策略，資源擁有者必須具有imagebuilder:put*權限。

2. 作為 RAM 資源共享進行推廣

   若要確保與您共用資源的所有主參與者都可以看到該資源，請執行中的promote-resource-share-created-from-policy AWS RAM AWS CLI命令。

取消共用共用元件、影像或方案

若要取消共用您擁有的共用元件、映像或方案，您必須將其從資源共用中移除。您可以使用控 AWS Resource Access Manager 制台或 AWS CLI.

> **Note**
>
> 要取消共享組件，映像或配方，消費者不能對其具有任何依賴關係。在所有者可以取消共享資源之前，消費者必須刪除對共享資源的任何依賴關係。

若要取消共用您使用主控台擁有的共用元件、映像檔或配方 AWS Resource Access Manager

請參閱《AWS RAM 使用者指南》中的更新資源共享。

若要取消共用您所擁有的共用元件、映像或配方，請使用 AWS CLI

使用disassociate-resource-share指令停止共用資源。

識別共用元件、映像或方案

擁有者和取用者可以使用中的影像產生器指令來識別共用元件、影像和影像配方 AWS CLI。

識別共用元件

執行清單元件指令，以取得您擁有的元件以及與您共用的元件的清單。取得元件指令會顯示元件擁有 AWS 帳戶 者的識別碼。

識別共用影像

運行列表圖像命令以獲取您擁有的圖像和與您共享的圖像的列表。取得影像指令會顯示影像擁有 AWS 帳戶 者的 ID。

識別共用容器映像

運行列表圖像命令以獲取您擁有的圖像和與您共享的圖像的列表。取得影像指令會顯示影像擁有 AWS 帳戶 者的 ID。

識別共用的影像配方

執行此list-image-recipes指令以取得您擁有的影像配方清單，以及與您共用的影像配方。此指get-image-recipe令會顯示影像配方擁有者的 AWS 帳戶 ID。

識別共用容器配方

執行此list-container-recipes命令以取得您擁有的容器配方清單，以及與您共用的容器配方。此指get-container-recipe令會顯示容器配方擁有者的 AWS 帳戶 ID。

共用元件、映像檔和配方權限

擁有者的許可

擁有者無法刪除共用元件、映像或映像配方，直到它不再共用為止。在沒有任何取用者依賴這些資源之前，擁有者無法取消共用這些資源。

消費者的許可

消費者可以讀取元件、映像檔或映像配方，但無法以任何方式修改它們。如果這些資源由其他取用者或資源的擁有者所擁有，則無法檢視或修改這些資源。消費者可以在映像配方中使用共享組件和圖像來創建自定義映像。消費者可以使用共用影像配方來建立自己的自訂影像。

計費和計量

使用 EC2 Image Builder 無須付費。

資源限制

共用元件、影像和影像配方只會計入擁有者對應的資源限制。消費者的資源限制不會受到與他們共用的資源的影響。

標記 EC2 Image Builder 資源

標記資源對於篩選和追蹤資源成本或其他類別非常有用。您也可以根據標籤控制存取。如需標籤式授權的詳細資訊，請參閱 基於 Image Builder 標籤的授權

Image Builder 支援下列動態標籤：

- `- {{imagebuilder:buildDate}}`

  在構建時解析為構建日期/時間。
- `- {{imagebuilder:buildVersion}}`

  解析為組建版本，這是位於 Image Builder Amazon 資源名稱 (ARN) 結尾的數字。例如，將組建版本`"arn:aws:imagebuilder:us-west-2:123456789012:component/myexample-component/2019.12.02/1"`顯示為`1`。

為了協助您追蹤已分發的 Amazon 機器映像 (AMI)，Image Builder 會自動將下列標籤新增至您的輸出 AMI。

- `"CreatedBy":"EC2 Image Builder"`
- `"Ec2ImageBuilderArn":"arn:aws:imagebuilder:`*`us-west-2`*`:`*`123456789012`*`:image/`*`simple-recipe-linux`*`/`*`1.0.0/10`*`"`。 此標記包含用來建立 AMI 之映 Image Builder 映像資源的 ARN。

目錄

標記資源 (AWS CLI)

下列範例示範如何使用 imagebuilder CLI 命令在 EC2 Image Builder 中新增和標記資源。您必須提供要套用至該標籤的`resourceArn`和標籤。

示例`tag-resource.json`內容如下：

```
{
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-example-pipeline",
    "tags": {
        "KeyName": "KeyValue"
    }
}
```

運行以下命令，該命令引用前面的`tag-resource.json`文件。

```
aws imagebuilder tag-resource --cli-input-json file://tag-resource.json
```

取消標記資源 ()AWS CLI

下列範例顯示如何使用 imagebuilder CLI 命令從資源中移除標籤。您必須提供`resourceArn`和金鑰才能移除標籤。

示例`untag-resource.json`內容如下：

```
{
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-example-pipeline",
    "tagKeys": [
        "KeyName"
    ]
}
```

運行以下命令，該命令引用前面的`untag-resource.json`文件。

```
aws imagebuilder untag-resource --cli-input-json file://untag-resource.json
```

列出特定資源的所有標籤 (AWS CLI)

下列範例顯示如何使用 imagebuilder CLI 命令列出特定資源的所有標籤。

```
aws imagebuilder list-tags-for-resource --resource-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-pipeline/my-example-pipeline
```

刪除 EC2 Image Builder 資源

您的 Image Builder 環境就像您的住家一樣，需要定期維護以協助您找到所需的內容，並完成工作，而不必涉及雜亂。請務必定期清理您為測試而建立的暫存資源。否則，您可能會忘記這些資源，然後，不記得它們用於什麼。到那時，如果您可以安全地擺脫它們，可能還不清楚。

刪除資源不會刪除在映像建立程序期間建立的任何 Amazon EC2 AMI 或 Amazon ECR 容器映像檔。您必須使用適當的 Amazon EC2 或 Amazon ECR 主控台動作、API 或 AWS CLI 命令來個別清理這些項目。

> **Tip**
>
> 若要避免刪除資源時發生相依性錯誤，請務必依下列順序刪除資源：
>
> 1. 影像管線
> 2. 圖片食譜
> 3. 所有剩餘資源

使用 AWS 管理主控台刪除資源

若要刪除映像管線及其資源，請依照下列步驟執行：

刪除配管

1. 若要查看在您的帳戶下建立的建置管線清單，請從導覽窗格中選擇 [映像管線]。
2. 選取「配管名稱」(Pipeline name) 旁的核取方塊，以選取要刪除的配管。
3. 在「影像管線」面板頂端的「動作」功能表上，選擇「刪除」。
4. 若要確認刪除，請`Delete`在方塊中輸入，然後選擇刪除。

刪除食譜

1. 若要查看您帳戶下建立的食譜清單，請從導覽窗格中選擇 [影像配方]。

2. 選取 [配方名稱] 旁的核取方塊，以選取您要刪除的食譜。
3. 在「影像配方」面板頂端的「動作」功能表上，選擇「刪除方案」。
4. 若要確認刪除，請Delete在方塊中輸入，然後選擇刪除。

刪除基礎結構組

1. 若要查看在您帳戶下建立的基礎結構組態清單，請從導覽窗格中選擇 [基礎結構設定]。
2. 選取 [組態名稱] 旁的核取方塊，以選取您要刪除的基礎結構組態。
3. 在「基礎結構設定」面板頂端，選擇「刪除」。
4. 若要確認刪除，請Delete在方塊中輸入，然後選擇刪除。

刪除散佈設定

1. 若要查看在您的帳戶下建立的發佈設定清單，請從功能窗格中選擇 [發佈設定]。
2. 選取「組態名稱」旁邊的核取方塊，以選取您為此教學課程建立的散佈設定。
3. 在「分佈」設定面板頂端，選擇「刪除」。
4. 若要確認刪除，請Delete在方塊中輸入，然後選擇刪除。

刪除映像

1. 若要查看在您帳戶下建立的影像清單，請從導覽窗格中選擇 [圖片]。
2. 為您要移除的影像選擇映像版本。這會開啟 [映像組建版本] 頁面。
3. 針對您要刪除的任何映像，選取 [版本] 旁邊的核取方塊。您可以一次選取多個映像版本。
4. 在「影像建置版本」面板頂端，選擇「刪除版本」。
5. 若要確認刪除，請Delete在方塊中輸入，然後選擇刪除。

使用刪除影像管線 AWS CLI

下列範例顯示如何使用刪除 Image Builder 資源 AWS CLI。如前所述，資源必須以下列順序刪除，以避免相依性錯誤：

1. 影像管線
2. 圖片食譜
3. 所有剩餘資源

刪除影像管線 (AWS CLI)

下列範例顯示如何透過指定影像管線 ARN 來刪除影像管線。

```
aws imagebuilder delete-image-pipeline --image-pipeline-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-pipeline/my-example-pipeline
```

刪除影像配方 (AWS CLI)

下列範例顯示如何透過指定影像方案 ARN 來刪除影像方案。

```
aws imagebuilder delete-image-recipe --image-recipe-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-recipe/my-example-recipe/2019.12.03
```

刪除基礎架構組態

下列範例顯示如何透過指定基礎結構組態資源的 ARN 來刪除基礎結構組態資源。

```
aws imagebuilder delete-infrastructure-configuration --infrastructure-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-
infrastructure-configuration
```

刪除散佈設定

下列範例顯示如何透過指定散佈設定資源的 ARN 來刪除分發設定資源。

```
aws imagebuilder delete-distribution-configuration --distribution-configuration-arn
 arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-
distribution-configuration
```

刪除映像

下列範例顯示如何透過指定映像建置版本的 ARN 來刪除映像檔組建版本。

```
aws imagebuilder delete-image --image-build-version-arn arn:aws:imagebuilder:us-
west-2:123456789012:image/my-example-image/2019.12.02/1
```

刪除元件

下列範例會示範如何使用 imagebuilder CLI 命令，藉由指定元件組建版本的 ARN 來刪除元件組建版本。

```
aws imagebuilder delete-component --component-build-version-arn
 arn:aws:imagebuilder:us-west-2:123456789012:component/my-example-
component/2019.12.02/1
```

Important

在刪除組件之前，請確保沒有以任何方式引用組件構建版本的方法。不這樣做可能會導致管線故障。

使用主控台管理 EC2 Image Builder 管道

Image Builder 映像管線提供用於建立和維護自訂 AMI 和容器映像的自動化架構。管道提供下列功能：

- 組合基本映像、用於建置和測試的元件、基礎結構組態和散佈設定。
- 使用主控台精靈Schedule builder中的，或輸入 cron 運算式以重複更新映像，以便排程自動化維護程序。
- 啟用基礎映像檔和元件的變更偵測，以便在沒有變更時自動略過排程的組建。
- 透過 Amazon 啟用以規則為基礎的自動化。 EventBridge

> **Note**
>
> 如需有關使用 EventBridge API 檢視或變更規則的詳細資訊，請參閱 Amazon EventBridge API 參考資料。若要取得有關使用中的 EventBridgeevents指令 AWS CLI 來檢視或變更規則的更多資訊，請參閱《指AWS CLI 令參考》中的事件。

目錄

列出並檢視管線詳細資訊

本節說明您可以尋找 EC2 Image Builder 映像管道的資訊和檢視詳細資訊的各種方式。

管道細節

列出影像管線 (AWS CLI)

下列範例顯示如何使用中的list-image-pipelines指令 AWS CLI 來列出所有映像管線。

```
aws imagebuilder list-image-pipelines
```

取得影像管線詳細資訊 (AWS CLI)

下列範例示範如何使用中的get-image-pipeline命令，透過其 ARN 取得映像管線的詳細資訊。 AWS CLI

```
aws imagebuilder get-image-pipeline --image-pipeline-arn arn:aws:imagebuilder:us-
west-2:123456789012:image-pipeline/my-example-pipeline
```

建立和更新 AMI 映像管線

您可以從映像 Image Builder 主控台、透過 Image Builder API 或使imagebuilder用 AWS CLI. 您可以使用 [建立映像管線主控台] 精靈引導您完成下列步驟：

- 指定管線詳細資訊，例如名稱、描述和資源標籤。
- 選取 AMI 映像配方，其中包含快速啟動受管理映像中的基本映像，或是您建立或與您共用的映像檔。方案也包含在映像產生器用來建立映像的 EC2 執行個體上執行下列工作的元件：
  - 新增和移除軟體
  - 自訂設定和指令碼
  - 執行選取的測試
- 指定工作流程以設定您的管道執行的映像檔建置和測試步驟。
- 使用您自己設定的預設設定或設定，為您的管道定義基礎結構組態。組態包括用於映像檔的執行個體類型和 key pair、安全性和網路設定、記錄儲存和疑難排解設定，以及 SNS 通知。

  這是選擇性步驟。如果您未自行定義組態，Image Builder 會使用基礎結構組態的預設設定。
- 定義發佈設定，將映像傳送至目的地 AWS 區域和帳戶。您可以指定 KMS 金鑰進行加密、設定 AMI 共用或授權組態，或為您散佈的 AMI 設定啟動範本。

  這是選擇性步驟。如果您未自行定義組態，Image Builder 會使用輸出 AMI 的預設命名，並將 AMI 散佈至來源區域。來源區域是您執行管線的「區域」。

如需有關使用 [建立映像管線主控台] 精靈的詳細資訊和 step-by-step 教學課程 (如有提供)，請參閱〈〉使用 EC2 Image Builder 主控台精靈建立映像管道。

目錄

創建一個 AMI 映像管道（AWS CLI）

您可以使用 JSON 檔案建立 AMI 映像管線，該檔案包含組態詳細資訊，做為 AWS CLI. create-image-pipeline

管道建置新映像以合併來自基礎映像和元件的任何暫止更新的頻率，取決於您所設定的。scheduleA schedule 具有以下屬性：

- scheduleExpression— 設定管線執行時的排程，以評估pipelineExecutionStartCondition並決定是否應該啟動組建。排程會使用 cron 運算式設定。如需如何在 Image Builder 中格式化 cron 運算式的詳細資訊，請參閱在 EC2 Image Builder 中使用 cron 表達式。
- pipelineExecutionStartCondition— 決定您的管道是否應該啟動建置。有效值包含：
  - EXPRESSION_MATCH_ONLY— 每次 cron 運算式與目前時間相符時，您的管道都會建立新映像。
  - EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE— 除非您的基本映像或元件有暫止的變更，否則您的管道將不會啟動新的映像組建。

當您在中執行create-image-pipeline指令時 AWS CLI，許多組態資源都是選用的。但是，某些資源具有條件需求，具體取決於管線創建的映像類型。AMI 映像管道需要以下資源：

- 圖片配方
- 基礎架構組態 ARN

1. 建立 CLI 輸入 JSON 文件

   使用您最愛的檔案編輯工具，建立 JSON 檔案，其中包含下列金鑰，以及適用於您環境的值。此範例使用名為 create-image-pipeline.json 的檔案：

```
{
 "name": "MyWindows2019Pipeline",
 "description": "Builds Windows 2019 Images",
 "enhancedImageMetadataEnabled": true,
 "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-recipe/2020.12.03",
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-infrastructure-configuration",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-distribution-configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 60
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * SUN *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "ENABLED"
}
```

Note

- 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
- JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

2. 使用您建立做為輸入的檔案，執行下列命令。

```
aws imagebuilder create-image-pipeline --cli-input-json file://create-image-pipeline.json
```

更新 AMI 映像管道（控制台）

為 AMI 映像建立 Image Builder 映像管線之後，您可以從映 Image Builder 主控台變更基礎結構組態和散發設定。

若要使用新的映像方案更新映像管線，您必須使用 AWS CLI. 如需詳細資訊，請參閱本指南中的 更新 AMI 映像管道（AWS CLI）。

選擇現有的 Image Builder 管道

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 若要查看在您的帳戶下建立的映像管線清單，請從導覽窗格中選擇 [映像管線]。

> **Note**
>
> 圖像管道列表包括由管道（AMI 或 Docker）創建的輸出圖像類型的指標。

3. 若要檢視詳細資訊或編輯配管，請選擇配管名稱連結。這會開啟配管的詳細視圖。

> **Note**
>
> 您也可以選取配管名稱旁的核取方塊，然後選擇「檢視詳細資訊」。

管道細節

配管詳細資訊頁包括下列區段：

摘要

頁面頂端的區段總結了管線的主要詳細資訊，這些詳細資訊在開啟任何詳細資訊標籤時都可以看見。此區段中顯示的詳細資訊只能在其各自的詳細資訊標籤上進行編輯。

詳細資料頁

- 輸出影像 — 顯示管線產生的輸出影像。
- 圖片食譜 — 顯示配方詳細資訊。建立配方之後，就無法對其進行編輯。您必須從映像產生器主控台的 [影像配方] 頁面，或使用中的 [映像產生器] 指令，建立新版本的方案 AWS CLI。如需詳細資訊，請參閱 管理食譜。
- 基礎結構配置 — 顯示用於配置構建管道基礎結構的可編輯信息

- 發佈設定 — 顯示 AMI 發佈的可編輯資訊。
- EventBridge rules — 針對選取的「事件匯流排」，顯示以目前管線為目標的 EventBridge 規則。包括建立事件匯流排和建立連結至 EventBridge 主控台的規則動作。如需有關此頁籤的詳細資訊，請參閱使用 EventBridge 規則。

編輯管道的基礎架構組態

基礎結構組態包括下列詳細資料，您可以在建立管道後編輯這些詳細資

- 基礎結構組態的說明。
- 要與執行個體設定檔建立關聯的 IAM 角色。
- AWS 基礎結構，包括用於通知的執行個體類型和 SNS 主題。
- VPC、子網路和安全群組。
- 疑難排解設定，包括失敗時終止執行個體、用於連線的金鑰配對，以及執行個體日誌的選用 S3 儲存貯體位置。

若要從管線詳細資料頁面編輯基礎結構組態，請遵循下列步驟：

1. 選擇基礎結構組態索引標籤。
2. 從 「組態詳細資料」面板右上角選擇「編輯」。
3. 當您準備好儲存對基礎結構組態所做的更新時，請選擇 [儲存變更]。

編輯管線的發佈設定

分佈設定包括下列詳細資訊，您可以在建立配管後編輯這些詳細資訊：

- 散發組態的說明。
- 您發佈映像檔之區域的地區設定。「區域 1」(Region 1) 預設為您建立配管的「區域」。您可以使用「新增區域」按鈕來新增要分配的區域，也可以移除「區域 1」以外的所有區域。

  區域設定包括：

  - 目標地區
  - 輸出 AMI 名稱
  - 啟動權限，以及共用權限的帳戶
  - 相關授權 (關聯授權組態)

> **Note**
>
> License Manager 設定不會在您帳戶中必須啟用的 AWS 區域之間複製，例如，在 `ap-east-1` (香港) 和 `me-south-1` (巴林) 區域之間。

若要從管線詳細資訊頁面編輯分發設定，請依照下列步驟執行：

1. 選擇 [發佈設定] 索引標籤。
2. 從 「分佈詳細資料」面板右上角選擇「編輯」。
3. 當您準備好儲存更新時，請選擇 [儲存變更]。

編輯管道的建置排程

「編輯配管」 頁面包含下列詳細資訊，您可以在建立配管後編輯這些詳細資訊：

- 管道的說明。
- 增強的元數據收集。此預設為開啟。若要將其關閉，請清除「啟用增強的中繼資料收集」核取方塊。
- 管道的「建置」排程。您可以在此處變更您的排程選項和所有設定。

欲從管線詳細資訊頁面編輯管線，請遵循下列步驟：

1. 在配管詳細資訊頁面的右上角，選擇「動作」，然後選擇 「編輯管線」。
2. 當您準備好儲存更新時，請選擇 [儲存變更]。

> **Note**
>
> 如需有關使用 cron 運算式排程組建的詳細資訊，請參閱在 EC2 Image Builder 中使用 cron 表達式。

更新 AMI 映像管道（AWS CLI）

您可以使用 JSON 檔案來更新 AMI 映像管線update-image-pipeline，做為 AWS CLI. 若要設定 JSON 檔案，您必須擁有 Amazon 資源名稱 (ARN) 才能參考下列現有資源：

- 要更新的影像管線
- 圖片食譜
- 基礎架構組
- 分佈設定

您可以使用中的update-image-pipeline命令更新 AMI 映像管道， AWS CLI 如下所示：

> **Note**
>
> UpdateImagePipeline 不支援管線的選擇性更新。您必須在更新要求中指定所有必要的屬性，而不僅僅是已變更的屬性。

1. 建立 CLI 輸入 JSON 文件

   使用您最愛的檔案編輯工具，建立 JSON 檔案，其中包含下列金鑰，以及適用於您環境的值。此範例使用名為 `create-component.json` 的檔案：

```
 {
 "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-
pipeline/my-example-pipeline",
 "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-
example-recipe/2019.12.08",
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/my-example-distribution-
configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 120
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * MON *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "DISABLED"
}
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

2. 使用您建立做為輸入的檔案，執行下列命令。

```
aws imagebuilder update-image-pipeline --cli-input-json file://update-image-pipeline.json
```

建立和更新容器映像管線

您可以使用映像 Image Builder 生器主控台、透過映像產生器 API 或使用中的imagebuilder命令來設定、設定和管理容 Image Builder 管道 AWS CLI。「建立映像管線主控台」精靈會提供啟動成品，並引導您完成下列步驟：

- 從快速啟動的受管映像檔、Amazon ECR 或碼頭集線器儲存庫中選取基本映像檔
- 新增和移除軟體
- 自訂設定和指令碼
- 執行選取的測試
- 使用預先配置的構建時變量創建一個 Docker 文件。
- 將影像分發至 AWS 區域

如需使用 [建立映像管線主控台] 精靈的詳細資訊和 step-by-step 教學課程，請參閱〈〉使用 EC2 Image Builder 主控台精靈建立容器映像管道。

目錄

建立容器映像管線 (AWS CLI)

您可以使用 JSON 檔案做為中create-image-pipeline命令的輸入來建立容器映像管線 AWS CLI。

管道建置新映像以合併來自基礎映像和元件的任何暫止更新的頻率，取決於您所設定的。scheduleA schedule 具有以下屬性：

- scheduleExpression— 設定管線執行時的排程，以評估pipelineExecutionStartCondition並決定是否應該啟動組建。排程會使用 cron 運算式設定。如需如何在 Image Builder 中格式化 cron 運算式的詳細資訊，請參閱在 EC2 Image Builder 中使用 cron 表達式。
- pipelineExecutionStartCondition— 決定您的管道是否應該啟動建置。有效值包含：
  - EXPRESSION_MATCH_ONLY— 每次 cron 運算式與目前時間相符時，您的管道都會建立新映像。
  - EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE— 除非您的基本映像或元件有暫止的變更，否則您的管道將不會啟動新的映像組建。

當您在中執行create-image-pipeline指令時 AWS CLI，許多組態資源都是選用的。但是，某些資源具有條件需求，具體取決於管線創建的映像類型。容器映像管線需要下列資源：

- 容器配方
- 基礎架構組態 ARN

如果您在執行指create-image-pipeline令時未包含發佈組態資源，則輸出影像會儲存在 ECR 存放庫中，您在執行命令所在的容器配方中指定為目標存放庫的 ECR 存放庫。如果您包括管線的發佈組態資源，則會使用您為發佈中第一個「區域」指定的目標存放庫。

1. 建立 CLI 輸入 JSON 文件

   使用您最愛的檔案編輯工具，建立 JSON 檔案，其中包含下列金鑰，以及適用於您環境的值。此範例使用名為 create-image-pipeline.json 的檔案：

```
{
 "name": "MyWindows2019Pipeline",
 "description": "Builds Windows 2019 Images",
 "enhancedImageMetadataEnabled": true,
 "containerRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:container-
recipe/my-example-recipe/2020.12.03",
```

```
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/my-example-distribution-
configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 60
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * SUN *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "ENABLED"
}
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

2. 使用您建立做為輸入的檔案，執行下列命令。

```
aws imagebuilder create-image-pipeline --cli-input-json file://create-image-
pipeline.json
```

更新容器映像管道（控制台）

為 Docker 映像建立 Image Builder 容器映像管線之後，您可以從映 Image Builder 生器主控台變更基礎結構組態和散發設定。

若要使用新容器方案更新容器映像管線，您必須使用 AWS CLI. 如需詳細資訊，請參閱本指南中的 更新容器映像管線 (AWS CLI)。

選擇現有的 Image Builder Docker 映像管線

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 若要查看在您的帳戶下建立的映像管線清單，請從導覽窗格中選擇映像管線。

> **Note**
>
> 圖像管道列表包括由管道（AMI 或 Docker）創建的輸出圖像類型的指標。

3. 若要檢視詳細資訊或編輯配管，請選擇配管名稱連結。這會開啟配管的詳細視圖。

> **Note**
>
> 您也可以選取配管名稱旁的核取方塊，然後選擇「檢視詳細資訊」。

管道細節

EC2 Image Builder 管道詳細資訊頁面包含下列段落：

摘要

頁面頂端的區段總結了管線的主要詳細資訊，這些詳細資訊在開啟任何詳細資訊標籤時都可見。此區段中顯示的詳細資訊只能在其各自的詳細資訊標籤上進行編輯。

詳細資料頁

- 輸出影像 — 顯示管線產生的輸出影像。
- 容器配方 — 顯示配方詳細資訊。建立配方之後，就無法對其進行編輯。您必須從容器配方頁面建立新版本的配方。如需詳細資訊，請參閱 建立容器配方的新版本。
- 基礎結構配置 — 顯示用於配置構建管道基礎結構的可編輯信息
- 分佈設定 — 顯示 Docker 映像分發的可編輯資訊。
- EventBridge rules — 針對選取的「事件匯流排」，顯示以目前管線為目標的 EventBridge 規則。包括建立事件匯流排和建立連結至 EventBridge 主控台的規則動作。如需有關此頁籤的詳細資訊，請參閱使用 EventBridge 規則。

編輯管道的基礎架構組態

基礎結構組態包括下列詳細資料，您可以在建立管道後編輯這些詳細資

- 基礎架構組態的說明。
- 要與執行個體設定檔建立關聯的 IAM 角色。
- AWS 基礎結構，包括用於通知的執行個體類型和 SNS 主題。
- VPC、子網路和安全群組。
- 疑難排解設定，包括失敗時終止執行個體、用於連線的金鑰配對，以及執行個體日誌的選用 S3 儲存貯體位置。

若要從管線詳細資料頁面編輯基礎結構組態，請遵循下列步驟：

1. 選擇基礎結構組態索引標籤。
2. 從 「組態詳細資料」面板右上角選擇「編輯」。
3. 當您準備好儲存對基礎結構組態所做的更新時，請選擇 [儲存變更]。

編輯管線的發佈設定

分佈設定包括下列詳細資訊，您可以在建立配管後編輯這些詳細資訊：

- 發佈設定的 [說明]。
- 您發佈映像檔之區域的地區設定。「區域 1」(Region 1) 預設為您建立配管的「區域」。您可以使用「新增區域」按鈕來新增要分配的區域，也可以移除「區域 1」以外的所有區域。

  區域設定包括：

  - 目標地區
  - 服務預設為「ECR」，且不可編輯。
  - 存放庫名稱 — 目標儲存庫的名稱 (不包括 Amazon ECR 位置)。例如，包含位置的存放庫名稱看起來像下列樣式：

    ```
    <account-id>.dkr.ecr.<region>.amazonaws.com/<repository-name>
    ```

    > **Note**
    >
    > 如果您變更存放庫名稱，則只有在名稱變更後建立的映像檔才會新增到新名稱下。您管道建立的任何先前映像檔都會保留在其原始儲存庫中。

若要從管線詳細資訊頁面編輯分發設定，請依照下列步驟執行：

1. 選擇 [發佈設定] 索引標籤。
2. 從 「分佈詳細資料」面板右上角選擇「編輯」。
3. 當您準備好儲存對發佈設定所做的更新時，請選擇 [儲存變更]。

編輯管道的建置排程

「編輯配管」 頁面包含下列詳細資訊，您可以在建立配管後編輯這些詳細資訊：

- 管道的說明。
- 增強的元數據收集。此預設為開啟。若要將其關閉，請清除「啟用增強的中繼資料收集」核取方塊。
- 管道的「建置」排程。您可以變更「排程」選項以及此區段中的所有設定。

欲從管線詳細資訊頁面編輯管線，請遵循下列步驟：

1. 在配管詳細資訊頁面的右上角，選擇「動作」，然後選擇 「編輯管線」。
2. 當您準備好儲存更新時，請選擇 [儲存變更]。

> **Note**
>
> 如需有關使用 cron 運算式排程組建的詳細資訊，請參閱在 EC2 Image Builder 中使用 cron 表達式。

更新容器映像管線 (AWS CLI)

您可以使用 JSON 檔案做為中update-image-pipeline命令的輸入來更新容器映像管線 AWS CLI。若要設定 JSON 檔案，您必須擁有 Amazon 資源名稱 (ARN) 才能參考下列現有資源：

- 要更新的影像管線
- 容器食譜
- 基礎架構組
- 分佈設定 (如果包含在目前配管中)

> **Note**
>
> 如果包含發佈設定資源，則在執行命令 (Region 1) 的發佈設定中指定為目標存放庫的 ECR 存放庫的優先順序高於容器配方中指定的目標存放庫。

請遵循下列步驟，使用下列update-image-pipeline指令更新容器映像管線 AWS CLI：

> **Note**
>
> UpdateImagePipeline 不支援管線的選擇性更新。您必須在更新要求中指定所有必要的屬性，而不僅僅是已變更的屬性。

1. 建立 CLI 輸入 JSON 文件

   使用您最愛的檔案編輯工具，建立 JSON 檔案，其中包含下列金鑰，以及適用於您環境的值。此範例使用名為 `create-component.json` 的檔案：

```
{
 "imagePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-
pipeline/my-example-pipeline",
 "containerRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:container-
recipe/my-example-recipe/2020.12.08",
 "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:infrastructure-configuration/my-example-infrastructure-
configuration",
 "distributionConfigurationArn": "arn:aws:imagebuilder:us-
west-2:123456789012:distribution-configuration/my-example-distribution-
configuration",
 "imageTestsConfiguration": {
  "imageTestsEnabled": true,
  "timeoutMinutes": 120
 },
 "schedule": {
  "scheduleExpression": "cron(0 0 * * MON *)",
  "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
 },
 "status": "DISABLED"
}
```

> **Note**
>
> - 您必須在 JSON 檔案路徑的開頭包括 `file://` 標記。
> - JSON 檔案的路徑應遵循執行命令之基礎作業系統的適當慣例。例如，Windows 會使用反斜線 (\) 來參照目錄路徑，而 Linux 會使用正斜線 (/)。

2. 使用您建立做為輸入的檔案，執行下列命令。

```
aws imagebuilder update-image-pipeline --cli-input-json file://update-image-pipeline.json
```

設定 EC2 Image Builder 管道的映像工作流程

透過影像工作流程，您可以自訂管道執行的工作流程，以根據需要建置和測試影像。您定義的工作流程會在 Image Builder 工作流程架構的前後關聯內執行。如需組成工作流程架構之階段的詳細資訊，請參閱管理 EC2 Image Builder 映像的建置和測試工作流程。

建置工作流

建置工作流程會在工作流程架構的`Build`階段執行。您只能為管道指定一個建置工作流程。或者，您可以完全跳過構建以配置僅測試管道。

測試工作流

在工作流程架構`Test`階段執行測試工作流程。您最多可以為管道指定十個測試工作流程。如果您只想建立管道，也可以完全跳過測試。

定義測試工作流程的測試群組

測試工作流程是在測試組中定義的。您最多可以為管道執行十個測試工作流程。您可以決定是以特定順序執行測試工作流程，還是要同時執行盡可能多的測試工作流程。它們的運行方式取決於您如何定義測試組。下列案例示範您可以定義測試工作流程的數種方式。

> **Note**
>
> 如果您使用主控台建立工作流程，建議您在定義測試群組之前，先花一些時間規劃執行測試工作流程的方式。在主控台中，您可以新增或移除測試工作流程和群組，但無法重新排序它們。

案例 1：一次執行一個測試工作流程

要一次運行一個所有測試工作流程，您最多可以配置十個測試組，每個測試組都包含一個測試工作流程。測試群組會依照新增至管線的順序，一次執行一個測試群組。這是確保測試工作流程以特定順序一次執行一個測試工作流程的一種方法。

案例 2：同時執行多個測試工作流程

如果訂單無關緊要，並且您希望同時運行盡可能多的測試工作流程，則可以配置單個測試組，並在其中放置最大數量的測試工作流程。Image Builder 最多可同時啟動五個測試工作流程，並在其他測試工作流程完成時啟動其他測試工作流程。如果您的目標是盡可能快地執行測試工作流程，這是執行此操作的一種方法。

情景 3：混合搭配

如果您有混合的案例，有些測試工作流程可以同時執行，有些測試工作流程應該一次執行一個，您可以設定測試群組來達成此目標。設定測試群組的唯一限制是可針對管道執行的測試工作流程數目上限

在 Image Builder 管線 (主控台) 中設定工作流程參數

工作流程參數的功能與建置工作流程和測試工作流程相同。建立或更新管道時，請選取要包含的建置和測試工作流程。如果您在工作流程文件中為所選工作流程定義參數，Image Builder 會在「參數」面板中顯示參數。對於沒有定義參數的工作流程，面板會隱藏起來。

每個參數都會顯示工作流程文件所定義的下列屬性：

- 名稱 (不可編輯) — 參數的名稱。
- 類型 (不可編輯) — 參數值的資料類型。
- 值 — 參數的值。您可以編輯參數值，為配管設定它。

指定 Image Builder 用來執行工作流程動作的 IAM 服務角色

服務存取

若要執行映像工作流程，Image Builder 需要執行工作流程動作的權限 您可以指定AWSServiceRoleForImageBuilder服務連結角色，也可以指定自己的服務存取角色，如下所示。

- 主控台 — 在管線精靈步驟 3 定義映像建立程序中，從「服務存取」面板的 IAM 角色清單中選取服務連結角色或您自己的自訂角色。
- I@@ mage Builder API — 在CreateImage動作請求中，指定服務連結角色或您自己的自訂角色做為`executionRole`參數值。

若要進一步瞭解如何建立服務角色，請參閱《AWS Identity and Access Management 使用指南》中的建立角色以將權限委派給 AWS 服務。

執行映像管線

如果您為管道選擇了手動排程選項，它只會在您手動啟動組建時執行。如果您選擇其中一個自動排程選項，您也可以在定期排定的執行之間手動執行。例如，如果您的管道通常每月執行一次，但您需要在上次執行兩週後將更新合併到其中一個元件，則可以選擇手動執行管道。

Console

若要從 Image Builder 主控台的管線詳細資訊頁面執行管線，請從頁面頂端的「動作」功能表中選擇「執行管線」。頁面頂端會顯示狀態訊息，通知您管道已啟動或發生錯誤。

1. 在配管詳細資訊頁面的左上角，從「動作」功能表中選擇「執行管線」。
2. 您可以在「狀態」欄的「輸出影像」索引標籤上查看管道的目前狀態。

AWS CLI

下列範例顯示如何使用中的start-image-pipeline-execution指令 AWS CLI 來手動啟動映像管線。當您執行此命令時，管線會建置並散佈新映像。

```
aws imagebuilder start-image-pipeline-execution --image-pipeline-arn
 arn:aws:imagebuilder:us-west-2:111122223333:image-pipeline/my-example-pipeline
```

若要查看建置管線執行時建立的資源，請參閱建立的資源。

在 EC2 Image Builder 中使用 cron 表達式

使用 EC2 Image Builder 的 cron 運算式來設定時間範圍，透過套用至管道基礎映像和元件的更新來重新整理映像。管線重新整理的時間範圍從您在 cron 運算式中設定的時間開始。您可以將 cron 表達式中的時間設置為分鐘。您的管線建置可以在開始時間或之後執行。

有時可能需要幾秒鐘，或者最多可能需要一分鐘的時間才能開始運行構建。

> **Note**
>
> Cron 運算式預設會使用世界協調時間 (UTC) 時區，或者您也可以指定時區。如需有關 UTC 時間的詳細資訊，並尋找您所在時區的偏移量，請參閱時區縮寫 — 全球清單。

Image Builder 生器中的 cron 運算式支援的值

EC2 Image Builder 使用包含六個必填欄位的 cron 格式。每一個都以兩者之間的空格分隔，沒有前導或尾隨空格：

```
<Minute> <Hour> <Day> <Month> <Day of the week> <Year>
```

下表顯示支援的必要 cron 項目的值。

Cron 運算式支援的值

欄位	值	萬用字元
分鐘	0-59	, - * /
小時	0-23	, - * /
天	1-31	, - * ? / L W
月	1-12 或 jan-dec	, - * /
星期中的一天	1-7 或 sun-sat	, - * ? L #
年	1970-2199	, - * /

萬用字元

下表說明 Image Builder 如何將萬用字元用於 cron 運算式。請記住，在您指定的組建啟動時間之後，最多可能需要一分鐘的時間。

Cron 運算式支援的萬用字元

萬用字元	描述
,	, (逗號) 萬用字元包含額外的值。在「月」欄位中，`jan,feb,mar` 包括「一月」、「二月」和「三月」。
-	- (破折號) 萬用字元用於指定範圍。在月份的日期欄位中，`1-15`包含指定月份的第 1 天到第 15 天。
*	* (星號) 萬用字元包含欄位的所有有效值。
?	?(問號) 萬用字元指定欄位值取決於其他設定。如果是「日」和 ay-of-week 「D」欄位，當指定或包含所有可能的值 (*) 時，另一個欄位必須為?。您不能同時指定兩者。例如，如果您`7`在「日」欄位中輸入 a (在該月的第七天執行組建)，則 D ay-of-week 位置必須包含?.
/	/ (斜線) 萬用字元用於指定增量。例如，如果您希望組建每隔一天執行一次，請`*/2`在「天」欄位中輸入。
L	任一天欄位中的 L 萬用字元會指定最後一天：28-31 代表月份的日期，視月份的日期而定，或星期日。
W	D ay-of-month 欄位中的 W 萬用字元會指定工作日。在 D ay-of-month 欄位中，如果您輸入之前的數字`W`，則表示您要定位最接近當天的工作日。例如，如果您指定`3W`，您希望組建在最接近當月第三天的工作日執行。
#	#（散列）僅允許用於星期字段，並且後面必須跟一個介於 1 和 5 之間的數字。此數字會指定

萬用字元	描述
	特定月份中要套用哪些週來執行組建。例如，如果您希望組建在每個月的第二個星期五執行，請使`fri#2`用「星期幾」欄位。

限制

- 您無法在相同的 cron 運算式中指定 D ay-of-month 和 D ay-of-week 欄位。如果您指定值或*在其中一個欄位中指定值，則必須在另一個欄位?中使用。
- 不支援頻率多於一分鐘的 Cron 表達式。

EC2 Image Builder 中的 cron 表達式示例

對於 Image Builder 主控台，Cron 運算式的輸入方式與 API 或 CLI 的輸入方式不同。若要查看範例，請選擇適用於您的索引標籤。

Image Builder console

下列範例顯示 cron 運算式，您可以在建置排程中輸入到主控台中。UTC 時間是使用 24 小時制指定的。

每天上午 10:00 (世界標準時間) 執行

```
0 10 * * ? *
```

每天下午 12:15 (世界標準時間) 執行

```
15 12 * * ? *
```

每天午夜 (世界標準時間) 執行

```
0 0 * * ? *
```

在每個工作日早上 10:00 (世界標準時間) 執行

```
0 10 ? * 2-6 *
```

每個工作日晚上 6 點 (UTC) 執行

```
0 18 ? * mon-fri *
```

在每個月的第一天上午 8:00 (UTC) 執行

```
0 8 1 * ? *
```

在每個月的第二個星期二晚上 10:30（世界標準時間）運行

```
30 22 ? * tue#2 *
```

> **Tip**
>
> 如果您不希望管線工作在執行期間延伸到第二天，請確定在指定開始時間時考量組建的時間。

API/CLI

下列範例顯示 cron 運算式，您可以使用 CLI 命令或 API 要求為建置排程輸入這些運算式。只會顯示 cron 運算式。

每天上午 10:00 (世界標準時間) 執行

```
cron(0 10 * * ? *)
```

每天下午 12:15 (世界標準時間) 執行

```
cron(15 12 * * ? *)
```

每天午夜 (世界標準時間) 執行

```
cron(0 0 * * ? *)
```

在每個工作日早上 10:00 (世界標準時間) 執行

```
cron(0 10 ? * 2-6 *)
```

在每個工作日晚上的下午 6:00 (UTC) 執行

```
cron(0 18 ? * mon-fri *)
```

在每個月的第一天上午 8:00 (UTC) 執行

```
cron(0 8 1 * ? *)
```

在每個月的第二個星期二晚上 10:30（世界標準時間）運行

```
cron(30 22 ? * tue#2 *)
```

> **Tip**
>
> 如果您不希望管線工作在執行期間延伸到第二天，請確定在指定開始時間時考量組建的時間。

EC2 Image Builder 中的速率運算式

Rate 表達式在您建立排程事件規則時開始，然後在其定義的排程上執行。

Rate 表達式有兩個必要欄位。欄位是以空格隔開。

語法

```
rate(value unit)
```

value

正數。

單位

時間的單位。所需單位可能不同，若值為 1，則需要 `minute`；若值超過 1，則需要 `minutes`。

有效值：minute | minutes | hour | hours | day | days (分鐘、數分鐘、小時、數小時、天、數天)

限制

如果值等於 1，則單位必須為單數。同樣地，對於大於 1 的數值，單位必須為複數。例如，`rate(1 hours)` 與 `rate(5 hour)` 不是有效的，但 `rate(1 hour)` 與 `rate(5 hours)` 是有效的。

搭配 Image Builder 管線使用 EventBridge 規則

來自各種 AWS 和合作夥伴服務的活動會以近乎即時的方式串流至 Amazon EventBridge 事件匯流排。您也可以產生自訂事件，並將事件從您自己的應用程式傳送至 EventBridge。事件匯流排會使用規則來決定路由事件資料的位置。

Image Builder 管線可作為 EventBridge 規則目標使用，這表示您可以根據您建立來回應匯流排上或排程事件所建立的規則來執行 Image Builder 管線。

如需影像產生器傳送至之系統產生之事件的摘要 EventBridge，請參閱Image Builder 傳送的事件訊息。

> **Note**
>
> 活動巴士是特定於某個區域的。規則和目標必須位於相同的區域。

目錄

EventBridge 條款

本節包含術語摘要，可協助您瞭解如何 EventBridge 與 Image Builder 管線整合。

事件

描述環境中可能會影響一或多個應用程式資源的變更。環境可以是環 AWS 境、SaaS 合作夥伴服務或應用程式，或是其中一個應用程式或服務。您也可以在時間軸上設定已排程的事件。

事件匯流排

從應用程式和服務接收事件資料的管線。

來源

將事件傳送至事件匯流排的服務或應用程式。

目標

在符合規則時 EventBridge 呼叫的資源或端點，將資料從事件傳送至目標。

規則

規則會比對連入事件，並將這些事件路由到目標以進行處理。單一規則可將事件傳送至多個目標，然後再 parallel 執行。規則是以事件模式或排程為基礎。

模式

事件模式定義事件結構和規則匹配的字段，以啟動目標動作。

排程

排程規則會根據排程執行動作，例如執行 Image Builder 管線以每季重新整理映像。排程運算式有兩種類型：

- Cron 運算式 — 使用可概述簡單準則的 cron 語法來比對特定排程條件；例如，在特定日期每週執行一次。您還可以建立更複雜的條件，例如在每月的第五天上午 2 點到凌晨 4 點之間每季執行一次。
- 速率運算式 — 指定呼叫目標時的規則間隔，例如每 12 小時一次。

檢視 Image Builder 管道的 EventBridge 規則

Image Builder Image P ip eline 詳細資訊頁面中的EventBridge 規則標籤會顯示您的帳戶可存取的 EventBridge 事件匯流排，以及套用至目前管線的所選事件匯流排的規則。此索引標籤也會直接連結至主EventBridge 控台，以建立新資源。

連結至 EventBridge 主控台的動作

- 建立事件匯流排
- 建立規則

若要進一步了解 EventBridge，請參閱 Amazon EventBridge 使用者指南中的以下主題。

- 什麼是 Amazon EventBridge
- Amazon EventBridge 活動巴士
- Amazon EventBridge 活動
- Amazon EventBridge 規則

使用 EventBridge 規則來排程管線建置

在此範例中，我們使用速率運算式為預設事件匯流排建立新的排程規則。此範例中的規則每 90 天在事件匯流排上產生一個事件。此事件會啟動管線建置以重新整理映像。

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。

2. 若要查看在您的帳戶下建立的映像管線清單，請從導覽窗格中選擇 [映像管線]。

> **Note**
>
> 圖像管道列表包括由管道（AMI 或 Docker）創建的輸出圖像類型的指標。

3. 若要檢視詳細資訊或編輯配管，請選擇配管名稱連結。這會開啟配管的詳細視圖。

> **Note**
>
> 您也可以選取配管名稱旁的核取方塊，然後選擇「檢視詳細資訊」。

4. 開啟EventBridge 規則索引標籤。
5. 保留在「事件匯流排」面板中預先選取的預設事件匯流排。
6. 選擇建立規則。這會帶您前往 Amazon EventBridge 主控台中的「建立規則」頁面。
7. 輸入規則的名稱和描述。在所選區域的事件匯流排中，規則名稱必須是唯一的。
8. 在「定義樣式」面板中，選擇「明細表」選項。這樣會展開面板，並選取「固定費率每個」選項。
9. 90在第一個方塊中輸入，然後從下拉式清單中選取「天」。
10. 在「選取目標」面板中執行下列動作：
    a. `EC2 Image Builder`從「目標」下拉式清單中選取。
    b. 若要將規則套用至 Image Builder 管線，請從「映像管線」下拉式清單中選取目標管線。
    c. EventBridge 需要對所選配管初始化建構的權限。對於此範例，請保留 [為此特定資源建立新角色] 的預設選項。
    d. 選擇 Add target (新增目標)。
11. 選擇 Create (建立)

> **Note**
>
> 若要進一步了解本範例未涵蓋的費率運算式規則設定，請參閱 Amazon EventBridge 使用者指南中的 R ate 運算式。

在 EC2 Image Builder 中整合產品和服務

EC2 Image Builder AWS Marketplace 與其他應用程式整合 AWS 服務 ，可協助您建立強大、安全的自訂機器映像。

产品

Image Builder 方法可以合併來自 AWS Marketplace Image Builder 管理元件的影像產品，以提供專門的建置和測試功能，如下所示。

- AWS Marketplace 影像產品 — 使用來自的影像產品 AWS Marketplace 作為配方中的基本影像，以符合組織標準，例如 CIS 強化。 當您從映像產生器主控台建立方案時，您可以從現有的訂閱中選擇，或從中搜尋特定產品 AWS Marketplace。當您從 Image Builder API、CLI 或 SDK 建立配方時，您可以指定映像產品 Amazon 資源名稱 (ARN) 作為基礎映像檔使用。
- AWS TOE 元件 — 您在方案中指定的元件可以執行建置和測試動作，例如，安裝軟體或執行合規性驗證。您訂閱的某些圖片產品可 AWS Marketplace 能會包含您可以在食譜中使用的隨附元件。CIS 強化映像檔包含一個相符的 AWS TOE 元件，您可以在方案中使用這些元件，針對您的組態強制執行 CIS 效能標竿第 1 級準則。

> **Note**
>
> 如需法規遵循相關產品的詳細資訊，請參閱。適用於圖像產 Image Builder 的合規產品

服務

Image Builder 與下列項目整合， AWS 服務 以提供詳細的事件指標、記錄和監控。此資訊可協助您追蹤活動、疑難排解映像檔建置問題，並根據事件通知建立自動化作業。

- AWS CloudTrail— 監視傳送至的 Image Builder 事件 CloudTrail。如需有關的詳細資訊 CloudTrail，請參閱什麼是 AWS CloudTrail？ 在《AWS CloudTrail 使用者指南》中。
- Amazon CloudWatch 日誌 — 監控、存放和存取您的 Image Builder 日誌檔。或者，您可以將日誌儲存到 S3 儲存貯體。如需有關 CloudWatch 日誌的詳細資訊，請參閱什麼是 Amazon CloudWatch 日誌？ 在 Amazon CloudWatch 日誌用戶指南中。
- Amazon EventBridge — 從您帳戶中的 Image Builder 活動 Connect 到即時事件資料串流。有關更多信息 EventBridge，請參閱什麼是 Amazon EventBridge？ 在 Amazon 用 EventBridge 戶指南。

- Amazon Inspector — 針對映像產生器啟動的 EC2 測試執行個體自動掃描，以建立新映像檔，發現軟體和網路設定中的弱點。Image Builder 會儲存輸出映像資源的發現項目，以便您可以在測試執行個體終止後進行調查和修復。如需有關掃描和定價的詳細資訊，請參閱什麼是 Amazon Inspector？在 Amazon Inspector 用戶指南。

  如果您設定增強型掃描，Amazon Inspector 也可以掃描您的 ECR 儲存庫。如需詳細資訊，請參閱亞馬遜檢查器使用者指南中的掃描 Amazon EC R 容器映像。

  > **Note**
  >
  > Amazon Inspector 是一個付費功能。

- AWS Marketplace— 查看目前 AWS Marketplace 產品訂閱的清單，並直接從 Image Builder 搜尋影像產品。您也可以使用已訂閱的影像產品做為映像產生器方案的基本影像。如需管理 AWS Marketplace 訂閱的詳細資訊，請參閱《AWS Marketplace 採購指南》。
- Amazon Simple Notification Service (Amazon SNS) — 如果已設定，請將有關映像狀態的詳細訊息發佈到您訂閱的 SNS 主題。如需詳細資訊，請參閱《Amazon Simple Notification Service 開發人員指南》中的什麼是 Amazon SNS？

產品與服務整合主題

- AWS CloudTrail Image Builder 中的集成
- Amazon CloudWatch 日誌集成在 Image Builder
- Amazon EventBridge 集成在 Image Builder
- Amazon Inspector 器集成 Image Builder
- AWS Marketplace Image Builder 中的集成
- Image Builder 中的 Amazon SNS 集成
- 適用於圖像產 Image Builder 的合規產品

AWS CloudTrail Image Builder 中的集成

此服務支持 AWS CloudTrail。 CloudTrail 這是一項服務，可記錄您的 AWS 呼叫， AWS 帳戶 並將日誌檔傳送到 Amazon S3 儲存貯體。透過使用收集的資訊 CloudTrail，您可以判斷成功向哪些要求提出 AWS 服務、提出要求的人員、提出要求的時間等。如需有關與 Image Builder CloudTrail 整合的詳細資訊，請參閱使用記錄 EC2 Image Builder API 呼叫 AWS CloudTrail。

若要進一步了解 CloudTrail，包括如何開啟和尋找記錄檔，請參閱《AWS CloudTrail 使用者指南》。

Amazon CloudWatch 日誌集成在 Image Builder

CloudWatch 記錄支援預設為開啟。在建置程序期間，記錄會保留在執行個體上，並串流至 CloudWatch Logs。執行個體記錄檔會在建立映像檔之前從執行個體中移除。

組建記錄會串流至下列 Image Builder CloudWatch 記錄群組並串流處理：

LogGroup:

```
/aws/imagebuilder/ImageName
```

LogStream (x.x.x/x)：

```
ImageVersion/ImageBuildVersion
```

您可以移除下列與執行個體設定檔相關聯的權限，選擇退出 CloudWatch 記錄串流。

```
"Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:CreateLogGroup",
            "logs:PutLogEvents"
        ],
        "Resource": "arn:aws:logs:*:*:log-group:/aws/imagebuilder/*"
    }
]
```

對於進階疑難排解，您可以使用執行命令執行預先定義的命令AWS Systems Manager 和指令碼 如需詳細資訊，請參閱 疑難排解 EC2 Image Builder。

Amazon EventBridge 集成在 Image Builder

Amazon EventBridge 是一種無伺服器事件匯流排服務，您可以使用它將 Image Builder 應用程式與其他 AWS 服務應用程式的相關資料連接起來。在中 EventBridge，規則會比對內送事件，並將其傳送至目標進行處理。單一規則可將事件傳送至多個目標，然後並行執 parallel 這些事件。

您可以使用自動化您的作業 EventBridge， 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。來自的事件 AWS 服務 會以近乎即時 EventBridge 的方式傳送至。您可以設定對傳入事件做出反應的規則，以啟動動作。例如，當 EC2 執行個體的狀態從擱置變更為執行中時，將事件傳送至 Lambda 函數。這些稱為模式。若要根據事件模式建立 EventBridge 規則，請參閱 Amazon EventBridge 使用者指南中的建立可對事件做出反應的 Amazon 規則。

可以自動啟動的動作包括：

- 調用一個 AWS Lambda 函數
- 叫用 Amazon EC2 執行命令
- 將事件轉送至 Amazon Kinesis Data Streams
- 啟動 AWS Step Functions 狀態機
- 通知 Amazon SNS 主題或 Amazon SQS 隊列

您也可以設定預設事件匯流排的排程規則，以定期執行動作，例如執行 Image Builder 管線以每季重新整理映像。排程運算式有兩種類型：

- cron 運算式 — 下列 cron 運算式範例會將工作排程為每天中午 UTC+0 執行：

  ```
  cron(0 12 * * ? *)
  ```

  如需搭配使用 cron 運算式的詳細資訊 EventBridge，請參閱 Amazon 使用 EventBridge 者指南中的 Cron 運算式。
- rate 運算式 — 下列費率運算式範例會將工作排程為每 12 小時執行一次：

  ```
  rate(12 hour)
  ```

  如需將費率運算式搭配使用的詳細資訊 EventBridge，請參閱 Amazon 使用 EventBridge 者指南中的費率運算式。

如需 EventBridge 規則如何與 Image Builder 管線整合的詳細資訊，請參閱搭配 Image Builder 管線使用 EventBridge 規則。

Image Builder 傳送的事件訊息

映像產生器資源的狀態發生重大變更 EventBridge 時，Image Builder 會傳送事件訊息至。例如，當影像的狀態發生變更時。下列範例顯示 Image Builder 可能傳送的典型 JSON 事件訊息。

EC2 Image Builder 映像狀態變更

Image Builder 會在建立映像檔期間，當影像資源的狀態變更時傳送此事件。例如，當影像狀態從一個狀態變更為另一個狀態時，如下所示：

- 從 `building` 到 `testing`
- 從 `testing` 到 `distribution`
- 從 `testing` 到 `failed`
- 從 `integrating` 到 `available`

```
{
    "version": "0",
    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "detail-type": "EC2 Image Builder Image State Change",
    "source": "aws.imagebuilder",
    "account": "111122223333",
    "time": "2024-01-18T17:50:56Z",
    "region": "us-west-2",
    "resources": ["arn:aws:imagebuilder:us-west-2:111122223333:image/
cmkencryptedworkflowtest-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222/1.0.0/1"],
    "detail": {
        "previous-state": {
            "status": "TESTING"
        },
        "state": {
            "status": "AVAILABLE"
        }
    }
}
```

偵測到 EC2 Image Builder CVE

如果您已為映像啟用 CVE 偵測，則每當影像掃描完成時，映像產生器就會傳送包含結果的訊息。

```
{
    "version": "0",
    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "detail-type": "EC2 Image Builder CVE Detected",
    "source": "aws.imagebuilder",
    "account": "111122223333",
    "time": "2023-03-01T16:59:09Z",
```

```
    "region": "us-east-1",
    "resources": [
        "arn:aws:imagebuilder:us-east-1:111122223333:image/test-image/1.0.0/1",
        "arn:aws:imagebuilder:us-east-1:111122223333:image-pipeline/test-pipeline"
    ],
    "detail": {
        "resource-id": "i-1234567890abcdef0",
        "finding-severity-counts": {
            "all": 0,
            "critical": 0,
            "high": 0,
            "medium": 0
        }
    }
}
```

EC2 Image Builder 工作流程步驟等

當工作WaitForAction流程步驟暫停以等待非同步動作完成時，Image Builder 會傳送訊息。

```
{
    "version": "0",
    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "detail-type": "EC2 Image Builder Workflow Step Waiting",
    "source": "aws.imagebuilder",
    "account": "111122223333",
    "time": "2024-01-18T16:54:44Z",
    "region": "us-west-2",
    "resources": ["arn:aws:imagebuilder:us-west-2:111122223333:image/
workflowstepwaitforactionwithvalidsnstopictest-a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222/1.0.0/1", "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/
build-workflow-a1b2c3d4-5678-90ab-cdef-EXAMPLE33333/1.0.0/1"],
    "detail": {
        "workflow-execution-id": "wf-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "workflow-step-execution-id": "step-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "workflow-step-name": "TestAutoSNSStop"
    }
}
```

Amazon Inspector 器集成 Image Builder

當您使用 Amazon Inspector 啟用安全掃描時，它會持續掃描您帳戶中的機器映像和執行個體，以防範作業系統和程式設計語言漏洞。如果啟動，安全性掃描會自動執行，而 Image Builder 可以在您建立新映像時，從測試執行個體儲存發現項目的快照。Amazon Inspector 是一項付費服務。

當 Amazon Inspector 發現軟體或網路設定中的弱點時，會採取下列動作：

- 通知你有一個發現。
- 評估發現項目的嚴重性。嚴重性等級會將弱點分類，以協助您排定發現項目的優先順序，並包含下列值：
  - 未分類
  - 資訊
  - 低
  - 中
  - 高
  - 嚴重
- 提供有關發現項目的資訊，以及其他資源的連結以取得更多詳細資訊。
- 提供補救指引，協助您解決產生發現項目的問題。

設定安全性掃描

如果您已為帳戶啟用 Amazon Inspector，Amazon Inspector 會自動掃描 Image Builder 啟動的 EC2 執行個體，以建立和測試新映像。這些執行個體在建置和測試程序期間的使用壽命很短，而且一旦這些執行個體關閉，它們的發現通常就會過期。為了協助您調查和修復新映像的發現項目，映 Image Builder 可以選擇性地將 Amazon Inspector 在建置程序期間在測試執行個體上識別的任何發現項目儲存為快照。

若要設定管道的安全性掃描，請參閱設定 Image Builder 檔的安全性掃描 AWS Management Console。

檢閱安全性發現

在 Image Builder 主控台中，您可以在一個位置檢視所有 Image Builder 資源的安全性發現項目。您可以在「安全性概觀」區段的「安全性發現項目」頁面上查看所有發現項目，或者依弱點、影像管道或影像將發現項目分組。主控台預設會顯示所有安全發現項目。[所有安全性發現項目] 選項的摘要面板會顯

示您針對每個嚴重性層級的發現項目數目。如需詳細資訊，請參閱 管理 Image Builder 的安全性發現項目 AWS Management Console。

若要進一步了解 Amazon Inspector 弱點發現，請參閱亞馬遜檢查器使用者指南中的了解 Amazon Inspector 中的發現項目。

AWS Marketplace Image Builder 中的集成

AWS Marketplace 是精心策劃的數位目錄，您可以在其中尋找和訂閱協力廠商軟體、資料和服務，以協助您建置符合業務需求的解決方案。 AWS Marketplace 將經過驗證的買家和註冊賣家與來自安全性、網路、儲存裝置、機器學習等熱門類別的軟體清單匯集在一起。

AWS Marketplace 賣方可以是獨立的軟體廠商 (ISV)、經銷商，或是擁有 AWS 產品和服務可供使用的項目的個人。當賣方提交產品時 AWS Marketplace，他們定義了產品的價格，以及使用條款和條件。買家同意為此優惠設定的價格、條款和條件。若要深入瞭解 AWS Marketplace，請參閱「什麼是 AWS Marketplace？」

> **Note**
>
> 資料產品提供者必須符合 AWS Data Exchange 資格要求。如需詳細資訊，請參閱《Data Exchange 使用指南》中的〈在AWS Data Exchange 時提供資料產品〉。 AWS

AWS Marketplace 整合功能

Image Builder 與 AWS Marketplace 整合，可直接從 Image Builder 主控台提供下列功能：

- 搜尋中提供的影像產品 AWS Marketplace。
- 查看目前 AWS Marketplace 產品訂閱的清單。
- 使用影 AWS Marketplace 像產品做為影像產生器方案的基本影像。

對於包含關聯 AWS 任務協調器和執行器 (AWS TOE) 元件的產品，您可以在主控台和 API、SDK 和 CLI 中篩選產品擁有者。如需詳細資訊，請參閱 列出 AWS TOE 元件。

從 AWS Marketplace 映像產生器主控台尋找影像產品

Image Builder 與 AWS Marketplace 整合，可直接從映像產生器主控台中的AWS Marketplace區段顯示您的圖片產品訂閱。您也可以在不離開 AWS Marketplace 映像產品主控台的情況下，從「影像產品」頁面搜尋 Image Builder 產品。

若要從 AWS Marketplace 映像產生器主控台尋找影像產品，請依照下列步驟執行：

1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
2. 在導覽窗格中，選擇AWS Marketplace區段中的 [影像產品]。
3. 「圖片產品」頁面會在「訂閱」索引標籤中顯示您已訂閱的映像產品摘要，或者您也可以在索AWS Marketplace引標籤中搜尋影像產品。

   Image Builder 預先篩選 AWS Marketplace 產品，專注於您可以在映像產生器方法中使用的機器映像。如需有關與 Image Builder AWS Marketplace 整合的詳細資訊，請選擇符合您要查看內容的索引標籤。

   AWS Marketplace

   此頁籤包含兩個面板。在左側，「調整結果」面板可協助您篩選結果，以尋找您要訂閱的產品。在右側，「搜尋產品」面板會顯示符合篩選條件的產品，並提供依產品名稱進行搜尋的選項。

   細化結果

   下列清單只顯示一些您可以套用至產品搜尋的篩選條件：

   - 選取一或多個產品類別，例如基礎架構軟體或機器學習。
   - 選擇影像產品的作業系統，或選擇特定作業系統平台的所有產品，例如「所有 Linu x/Unix」。
   - 選擇一個或多個發布商以顯示其可用產品。選取「全部顯示」連結，即可顯示所有出版商的產品符合您套用的篩選器。

     > **Note**
     >
     > 出版商名稱不是按字母順序排列。如果您要尋找特定的發行者，例如`Center for Internet Security`，您可以在 [所有發佈者] 對話方塊頂端的搜尋方塊中輸入部分名稱。您應該將名稱拼出來，作為縮寫，例如`CIS`可能無法產生您正在尋找的結果。
     > 您也可以逐頁瀏覽出版商名稱。

   篩選器選擇是動態的。您所做的每個選擇都會影響其他所有類別的選項。有成千上萬種產品可供選擇 AWS Marketplace，因此您可以篩選的越多產品，就越有可能找到想要的產品。

搜索產品

要按名稱查找特定產品，您可以在此面板頂部的搜索欄中輸入名稱的一部分。每個產品結果包括以下詳細資訊：

- 產品名稱和標誌。這兩者都會連結至中的產品詳細資訊頁面 AWS Marketplace。詳細資訊頁面會在瀏覽器的新索引標籤中開啟。如果您想要在影像產生器配方中使用該圖片產品，您可以從該處訂閱該圖片產品。如需更多資訊，請參閱「購買指南」中的「AWS Marketplace 購買產品」。

  如果您在中訂閱圖片產品 AWS Marketplace，請切換回瀏覽器中的「Image Builder」索引標籤，然後重新整理訂閱的影像產品清單以查看該影像產品。

  > **Note**
  >
  > 您的新訂閱可能需要幾分鐘的時間才能使用。

- 發行者名稱。這會連結至中的發行者詳細資訊頁面 AWS Marketplace。發行者詳細資訊頁面會在瀏覽器的新索引標籤中開啟。
- 產品版本。
- 產品星級評級，並直接鏈接到中產品詳細信息頁面的評論部分 AWS Marketplace。詳細資訊頁面會在瀏覽器的新索引標籤中開啟。
- 產品說明的前幾行。

在搜尋列下方，您可以看到您的搜尋產生了多少結果，以及目前顯示的結果子集。您可以使用面板右側的其他控制項來調整設定，以便一次顯示的產品數量，以及套用至結果的排序順序。您也可以使用分頁控制項來逐頁瀏覽結果。

Subscriptions

此分頁會顯示您已訂閱的影像產品清單 AWS Marketplace。每個訂閱產品都會顯示下列詳細資料：

- 產品名稱，這會連結至中的產品詳細資訊頁面 AWS Marketplace。已訂閱產品的產品詳細資訊頁面會在瀏覽器的新分頁中開啟。
- 發行者名稱。這會連結至中的發行者詳細資訊頁面 AWS Marketplace。發行者詳細資訊頁面會在瀏覽器的新索引標籤中開啟。

- 您訂閱的產品版本。
- 如果您訂閱的產品包含關聯的元件，映像產生器會顯示 AWS TOE 元件詳細資料的連結。

在頁面頂端，您可以依名稱搜尋特定產品，也可以使用分頁控制項逐頁瀏覽結果。若要使用訂閱的產品做為新配方的基本影像，請選取已訂閱的產品，然後選擇 [建立新配方]。Image Builder 預先選擇默認情況下，列表中的第一個產品。

> **Note**
>
> 如果您正在尋找剛訂閱的產品，但在清單中沒有看到該產品，請使用索引標籤頂端的「重新整理」按鈕來重新整理結果。新的訂閱可能需要幾分鐘的時間才會顯示在清單中。

在 AWS Marketplace Image Builder 配方中使用圖像產品

在 Image Builder 主控台中，您可以透過兩種方式根據其中一個訂閱的映像產品建立新的映像配方。

1. 您可以從「影像產品」頁面開始，如下所示：
    1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
    2. 在導覽窗格中，選擇AWS Marketplace區段中的 [影像產品]。
    3. 開啟「訂閱」索引標籤。
    4. 選擇訂閱的圖片產品，用作方案中的基本圖片。
    5. 選擇 [建立新配方]。這會開啟「建立食譜」頁面，其中包含「AWS Marketplace 圖片」選項，並預先選取您訂閱的影像產品
    6. 像平常一樣為您的食譜配置剩餘設置。如需有關影像配方的更多資訊，請參閱建立影像配方的新版本。
2. 您也可以開啟「建立配方」頁面，然後選取 AWS Marketplace 要用作基本影像的圖片產品。
    1. 開啟 EC2 Image Builder 主控台，位於 https://console.aws.amazon.com/imagebuilder/。
    2. 在導覽窗格中，選擇AWS Marketplace區段中的 [影像配方]。這會顯示您已建立的影像配方清單。
    3. 選擇建立映像配方。這會開啟 [建立方案] 頁面。
    4. 像往常一樣在配方詳細信息部分輸入您的食譜名稱和版本。

5. 在「基本影像」區段中，選擇「影AWS Marketplace 像」選項。這會在「訂閱」索引標籤中顯示您已訂閱的 AWS Marketplace 映像產品清單。您可以從列表中選擇基本圖像。

   您還可以 AWS Marketplace 直接從AWS Marketplace標籤中搜索其他可用的圖像產品。選擇「新增產品」，或直接開啟分AWS Marketplace頁。如需有關如何設定篩選器和搜尋的詳細資訊 AWS Marketplace，請參閱從 AWS Marketplace 映像產生器主控台尋找影像產品。
6. 照常輸入剩餘的詳細資料，然後選擇「建立食譜」。

> **Note**
>
> 如果您的映像產品訂閱包含組 AWS TOE 建元件，您可以從 [建置元件] 清單中選取它。`Third party managed`從元件擁有者類型清單中選取以查看它。如果您的產品訂閱包含 AWS TOE 測試元件，請遵循「測試元件」清單的相同程序。

Image Builder 中的 Amazon SNS 集成

Amazon Simple Notification Service (Amazon SNS) 是一種受管服務，可提供從發佈者到訂閱者 (也稱為生產者和消費者) 的非同步訊息傳遞。

您可以在基礎結構組態中指定 SNS 主題。當您建立映像或執行管線時，Image Builder 可以將有關映像狀態的詳細訊息發佈到本主題中。當影像狀態達到下列其中一種狀態時，Image Builder 會發佈訊息：

- `AVAILABLE`
- `FAILED`

如需 Image Builder 的 SNS 訊息範例，請參閱SNS 訊息格式。如果您想要建立新的 SNS 主題，請參閱 Amazon 簡單通知服務開發人員指南中的開始使用 Amazon SNS。

加密的 SNS 主題

如果您的 SNS 主題已加密，則必須在 AWS KMS key 原則中授與 Image Builder 服務角色的權限，才能執行下列動作：

- `kms:Decrypt`
- `kms:GenerateDataKey`

> **Note**
>
> 如果您的 SNS 主題已加密，則加密此主題的金鑰必須位於執行 Image Builder 服務的帳戶中。Image Builder 無法傳送通知給使用其他帳戶金鑰加密的 SNS 主題。

KMS 金鑰原則新增範例

下列範例顯示您新增至 KMS 金鑰原則的其他區段。第一次建立映像產生器映像時，將 Amazon 資源名稱 (ARN) 用於 Image Builder 在帳戶下建立的 IAM 服務連結角色。若要深入瞭解 Image Builder 服務連結角色，請參閱使用 EC2 Image Builder 的服務連結角色。

```
{
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::123456789012:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder"
    },
    "Action": [
      "kms:GenerateDataKey*",
      "kms:Decrypt"
    ],
    "Resource": "*"
  }]
}
```

您可以使用下列方法之一來取得 ARN。

AWS Management Console

若要取得映像產生器在您帳戶下建立的服務連結角色的 ARN AWS Management Console，請依照下列步驟執行：

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇 Roles (角色)。
3. 搜尋ImageBuilder，然後從結果中選擇下列「角色」名稱：AWSServiceRoleForImageBuilder這會顯示角色詳細資訊頁面。
4. 若要將 ARN 複製到剪貼簿，請選擇 ARN 名稱旁邊的圖示。

AWS CLI

若要取得映像產生器在您的帳戶下建立之服務連結角色的 ARN AWS CLI，請使用 IAM get-role 命令，如下所示。

```
aws iam get-role --role-name AWSServiceRoleForImageBuilder
```

部分樣本輸出：

```
{
    "Role": {
        "Path": "/aws-service-role/imagebuilder.amazonaws.com/",
        "RoleName": "AWSServiceRoleForImageBuilder",
        ...
        "Arn": "arn:aws:iam::123456789012:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder",
        ...
}
```

SNS 訊息格式

Image Builder 將訊息發佈到您的 Amazon SNS 主題後，訂閱該主題的其他服務可以篩選訊息格式，並判斷其是否符合進一步動作的條件。例如，成功訊息可能會啟動更新 AWS Systems Manager 參數存放區的工作，或為輸出 AMI 啟動外部相容性測試工作流程。

下列範例顯示了 Image Builder 在管線建置執行到完成並建立 Linux 映像時發佈的典型訊息的 JSON 承載。

```
{
  "versionlessArn": "arn:aws:imagebuilder:us-west-1:123456789012:image/example-linux-
image",
  "semver": 1237940039285380274899124227,
  "arn": "arn:aws:imagebuilder:us-west-1:123456789012:image/example-linux-
image/1.0.0/3",
  "name": "example-linux-image",
  "version": "1.0.0",
  "type": "AMI",
  "buildVersion": 3,
  "state": {
    "status": "AVAILABLE"
  },
```

```
  "platform": "Linux",
  "imageRecipe": {
    "arn": "arn:aws:imagebuilder:us-west-1:123456789012:image-recipe/example-linux-
image/1.0.0",
    "name": "amjule-barebones-linux",
    "version": "1.0.0",
    "components": [
      {
        "componentArn": "arn:aws:imagebuilder:us-west-1:123456789012:component/update-
linux/1.0.2/1"
      }
    ],
    "platform": "Linux",
    "parentImage": "arn:aws:imagebuilder:us-west-1:987654321098:image/amazon-linux-2-
x86/2022.6.14/1",
    "blockDeviceMappings": [
      {
        "deviceName": "/dev/xvda",
        "ebs": {
          "encrypted": false,
          "deleteOnTermination": true,
          "volumeSize": 8,
          "volumeType": "gp2"
        }
      }
    ],
    "dateCreated": "Feb 24, 2021 12:31:54 AM",
    "tags": {
      "internalId": "1a234567-8901-2345-bcd6-ef7890123456",
      "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:image-recipe/example-
linux-image/1.0.0"
    },
    "workingDirectory": "/tmp",
    "accountId": "462045008730"
  },
  "sourcePipelineArn": "arn:aws:imagebuilder:us-west-1:123456789012:image-pipeline/
example-linux-pipeline",
  "infrastructureConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-1:123456789012:infrastructure-configuration/
example-linux-infra-config-uswest1",
    "name": "example-linux-infra-config-uswest1",
    "instanceProfileName": "example-linux-ib-baseline-admin",
    "tags": {
      "internalId": "234abc56-d789-0123-a4e5-6b789d012c34",
```

```
      "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:infrastructure-
configuration/example-linux-infra-config-uswest1"
    },
    "logging": {
      "s3Logs": {
        "s3BucketName": "12345-example-linux-testbucket-uswest1"
      }
    },
    "keyPair": "example-linux-key-pair-uswest1",
    "terminateInstanceOnFailure": true,
    "snsTopicArn": "arn:aws:sns:us-west-1:123456789012:example-linux-ibnotices-
uswest1",
    "dateCreated": "Feb 24, 2021 12:31:55 AM",
    "accountId": "123456789012"
  },
  "imageTestsConfigurationDocument": {
    "imageTestsEnabled": true,
    "timeoutMinutes": 720
  },
  "distributionConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-1:123456789012:distribution-configuration/
example-linux-distribution",
    "name": "example-linux-distribution",
    "dateCreated": "Feb 24, 2021 12:31:56 AM",
    "distributions": [
      {
        "region": "us-west-1",
        "amiDistributionConfiguration": {}
      }
    ],
    "tags": {
      "internalId": "345abc67-8910-12d3-4ef5-67a8b90c12de",
      "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:distribution-
configuration/example-linux-distribution"
    },
    "accountId": "123456789012"
  },
  "dateCreated": "Jul 28, 2022 1:13:45 AM",
  "outputResources": {
    "amis": [
      {
        "region": "us-west-1",
        "image": "ami-01a23bc4def5a6789",
        "name": "example-linux-image 2022-07-28T01-14-17.416Z",
```

```
        "accountId": "123456789012"
      }
    ]
  },
  "buildExecutionId": "ab0cd12e-34fa-5678-b901-2c3456d789e0",
  "testExecutionId": "6a7b8901-cdef-234a-56b7-8cd89ef01234",
  "distributionJobId": "1f234567-8abc-9d0e-1234-fa56b7c890de",
  "integrationJobId": "432109b8-afe7-6dc5-4321-0ba98f7654e3",
  "accountId": "123456789012",
  "osVersion": "Amazon Linux 2",
  "enhancedImageMetadataEnabled": true,
  "buildType": "USER_INITIATED",
  "tags": {
    "internalId": "901e234f-a567-89bc-0123-d4e567f89a01",
    "resourceArn": "arn:aws:imagebuilder:us-west-1:123456789012:image/example-linux-
image/1.0.0/3"
  }
}
```

下列範例顯示映像產生器針對 Linux 映像的管線建置失敗而發佈的典型訊息的 JSON 承載。

```
{
  "versionlessArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-
image",
  "semver": 12379400392853802748991242 31,
  "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-image/1.0.0/7",
  "name": "My Example Image",
  "version": "1.0.0",
  "type": "AMI",
  "buildVersion": 7,
  "state": {
    "status": "FAILED",
    "reason": "Image Failure reason."
  },
  "platform": "Linux",
  "imageRecipe": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-
image/1.0.0",
    "name": "My Example Image",
    "version": "1.0.0",
    "description": "Testing Image recipe",
    "components": [
      {
```

```
        "componentArn": "arn:aws:imagebuilder:us-west-2:123456789012:component/my-
example-image-component/1.0.0/1"
      }
    ],
    "platform": "Linux",
    "parentImage": "ami-0cd12345db678d90f",
    "dateCreated": "Jun 21, 2022 11:36:14 PM",
    "tags": {
      "internalId": "1a234567-8901-2345-bcd6-ef7890123456",
      "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-
example-image/1.0.0"
    },
    "accountId": "123456789012"
  },
  "sourcePipelineArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-pipeline/my-
example-image-pipeline",
  "infrastructureConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/
my-example-infra-config",
    "name": "SNS topic Infra config",
    "description": "An example that will retain instances of failed builds",
    "instanceTypes": [
      "t2.micro"
    ],
    "instanceProfileName": "EC2InstanceProfileForImageBuilder",
    "tags": {
      "internalId": "234abc56-d789-0123-a4e5-6b789d012c34",
      "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-
configuration/my-example-infra-config"
    },
    "terminateInstanceOnFailure": true,
    "snsTopicArn": "arn:aws:sns:us-west-2:123456789012:example-pipeline-notification-
topic",
    "dateCreated": "Jul 5, 2022 7:31:53 PM",
    "accountId": "123456789012"
  },
  "imageTestsConfigurationDocument": {
    "imageTestsEnabled": true,
    "timeoutMinutes": 720
  },
  "distributionConfiguration": {
    "arn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-
example-distribution-config",
    "name": "New distribution config",
```

```
    "dateCreated": "Dec 3, 2021 9:24:22 PM",
    "distributions": [
      {
        "region": "us-west-2",
        "amiDistributionConfiguration": {},
        "fastLaunchConfigurations": [
          {
            "enabled": true,
            "snapshotConfiguration": {
              "targetResourceCount": 2
            },
            "maxParallelLaunches": 2,
            "launchTemplate": {
              "launchTemplateId": "lt-01234567890"
            },
            "accountId": "123456789012"
          }
        ]
      }
    ],
    "tags": {
      "internalId": "1fecd23a-4f56-7f89-01e2-345678abbe90",
      "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-
configuration/my-example-distribution-config"
    },
    "accountId": "123456789012"
  },
  "dateCreated": "Jul 5, 2022 7:40:15 PM",
  "outputResources": {
    "amis": []
  },
  "accountId": "123456789012",
  "enhancedImageMetadataEnabled": true,
  "buildType": "SCHEDULED",
  "tags": {
    "internalId": "456c78b9-0e12-3f45-afb6-7e89b0f1a23b",
    "resourceArn": "arn:aws:imagebuilder:us-west-2:123456789012:image/my-example-
image/1.0.0/7"
  }
}
```

適用於圖像產 Image Builder 的合規產品

隨著不斷發展的安全標準，維持合規性並保護您的組織免受網絡威脅可能是一個挑戰。為了確保您的自訂映像檔符合規範，並在發行者發佈新版本時透過自動更新保持這種狀態，Image Builder 會與合 AWS Marketplace 規性產品和 AWS TOE 元件整合。

Image Builder 與下列法規遵循產品整合：

- 互聯網安全中心（CIS）基準加強

  您可以使用 CIS 強化映像和相關的 CIS 強化元件來建立符合最新 CIS 基準測試第 1 級準則的自訂映像檔。CIS 硬化圖像是可用在 AWS Marketplace. 若要深入了解如何設定和使用 CIS 強化映像和強化元件，請參閱 CIS 網站支援入口網站中的快速入門指南。

  > **Note**
  >
  > 當您訂閱 CIS 強化映像時，您還可以訪問相關的構建組件，該組件運行腳本以針對您的配置強制執行 CIS 基準測試級別 1 準則。如需詳細資訊，請參閱 CIS 硬化組件。

- 安全性技術實作指南 (STIG)

  對於 STIG 合規性，使用可以在 Image Builder 方法中使用亞馬遜管理 AWS 任務協調器和執行器 (AWS TOE) STIG 元件。STIG 元件會掃描您的組建執行個體是否有設定錯誤，並執行補救指令碼以更正發現的問題。我們無法保證您使用映像產生器建立的映像檔符合 STIG 規範。您必須與組織的法規遵循團隊合作，確認您的最終映像是否符合規定。如需可在 Image Builder 方法中使用的 AWS TOE STIG 元件的完整清單，請參閱適用於 EC2 Image Builder 的 Amazon 受管 STIG 強化元件。

監控 EC2 Image Builder 中的事件和日誌

為了維持 EC2 Image Builder 管道的可靠性、可用性和效能，監控事件和日誌非常重要。事件和日誌可幫助您查看大局並在 API 調用失敗時深入了解詳細信息。Image Builder 與服務整合，這些服務可在事件符合您設定的條件時傳送警示並啟動自動回應。

下列主題說明您可以透過與 Image Builder 整合的服務使用的監視技巧。

監控事件和記錄

- 使用記錄 EC2 Image Builder API 呼叫 AWS CloudTrail

使用記錄 EC2 Image Builder API 呼叫 AWS CloudTrail

EC2 Image Builder 與此服務整合 AWS CloudTrail，可提供使用者、角色或服務透過 Image Builder API 執行的所有 API 呼叫動作記錄的 AWS 服務。CloudTrail 將 Image Builder 擷取為事件。擷取的呼叫包括來自 Image Builder 主控台的呼叫，以及對 Image Builder API 作業的程式碼呼叫。

如果您建立追蹤，您可以啟用連續傳遞 CloudTrail 事件至 S3 儲存貯體，包括 Image Builder 的事件。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 Image Builder 提出的要求、提出請求的 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，請參閱使AWS CloudTrail 用者指南。

Image Builder 資訊 CloudTrail

CloudTrail 在您創建帳戶 AWS 帳戶 時啟用。在 Image Builder 中發生活動時，該活動會與 CloudTrail 事件歷史記錄中的其他 AWS 服務事件一起記錄在事件中。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。

如需您 AWS 帳戶的正在進行的事件記錄 (包括 Image Builder 的事件)，請建立追蹤。追蹤可 CloudTrail 將日誌檔傳遞至 S3 儲存貯體。依預設，當您在主控台中建立追蹤時，該追蹤會套用至所有的 AWS 區域。追蹤記錄來自 AWS 分割區中所有區域的事件，並將日誌檔傳送到您指定的 S3 儲存貯體。此外，您可以設定其他， AWS 服務 以進一步分析 CloudTrail 記錄中收集的事件資料並採取行動。如需詳細資訊，請參閱下列內容：

- 建立系統線的概述。

- CloudTrail 支援的服務和整合。
- 設定的 Amazon SNS 通知 CloudTrail。
- 接收來自多個區域的 CloudTrail 日誌文件。
- 從多個帳戶接收CloudTrail 日誌文件。

CloudTrail 記錄 EC2 Image Builder API 參考中記錄的所有映像產生器動作。例如，呼叫`CreateImagePipeline`、`UpdateInfrastructureConfiguration`和`StartImagePipelineExecution`動作會在 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：

- 該請求是否使用根或 IAM 使用者憑證提出。
- 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。
- 請求是否由其他 AWS 服務提出。

如需有關判斷誰要求事件的詳細資訊，請參閱 CloudTrail userIdentity 元素。

EC2 Image Builder 中的安全性

雲安全 AWS 是最高的優先級。身為 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 AWS 與您之間共同承擔的責任。共同責任模型將其描述為雲端的安全性和雲端中的安全性：

- 雲端的安全性 — AWS 負責保護在雲 AWS 端 AWS 服務 中執行的基礎架構。 AWS 還為您提供可以安全使用的服務。若要了解適用於 EC2 Image Builder 的合規計劃，請參閱合規計劃服務範圍AWS 服務 中的合規計劃AWS的。
- 雲端中的安全性 — 您的責任取決於您使用的 AWS 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。

本文件可協助您瞭解如何在使用 Image Builder 時套用共同的責任模型。下列主題說明如何設定 Image Builder 以符合安全性和合規性目標。您也會學到如何使用其他協助 AWS 服務 您監視和保護 Image Builder 資源的其他資源。

主題

EC2 Image Builder 中的資料保護

AWS 共同責任模型適用於 EC2 Image Builder 中的資料保護。如此模型中所述， AWS 負責保護執行所有 AWS 雲端. 您負責維護在此基礎設施上託管內容的控制權。您也同時負責所使用 AWS 服務 的安全組態和管理任務。如需資料隱私權的詳細資訊，請參閱資料隱私權常見問答集。如需有關歐洲資料保護的相關資訊，請參閱 AWS 安全性部落格上的 AWS 共同的責任模型和 GDPR 部落格文章。

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

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源進行通訊。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 使用設定 API 和使用者活動記錄 AWS CloudTrail。
- 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 Amazon S3 的敏感資料。
- 如果您在透過命令列介面或 API 存取時需要經 AWS 過 FIPS 140-2 驗證的加密模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱聯邦資訊處理標準 (FIPS) 140-2 概觀。

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

EC2 Image Builder 中的加密和金鑰管理

Image Builder 預設會使用服務擁有的 KMS 金鑰加密傳輸中和靜態資料，但下列情況除外：

- 自訂元件 — Image Builder 會使用您的預設 KMS 金鑰或服務擁有的 KMS 金鑰來加密自訂元件。
- 映像工作流程 — 如果您在建立工作流程期間指定金鑰，Image Builder 可以使用客戶管理的金鑰來加密映像工作流程 Image Builder 會使用您的金鑰來處理加密和解密，以執行您為映像檔設定的工作流程。

您可以通過管理自己的密鑰 AWS KMS。但是，您沒有管理映像產生器擁有的 Image Builder KMS 金鑰的權限。如需使用管理 KMS 金鑰的詳細資訊 AWS Key Management Service，請參閱開 AWS Key Management Service 發人員指南中的入門指南。

加密內容

為了對加密資料提供額外的完整性和真實性檢查，您可以選擇在加密資料時加入加密內容。使用加密內容加密資源時， AWS KMS 以密碼編譯方式將內容繫結至加密文字。只有在請求者為上下文提供完全相符且區分大小寫的情況下，才能解密資源。

本節中的政策範例使用類似於 Image Builder 工作流程資源之 Amazon 資源名稱 (ARN) 的加密內容。

使用客戶管理的金鑰加密影像工作流程

若要新增保護層，您可以使用自己的客戶管理金鑰來加密 Image Builder 工作流程資源。如果您使用客戶管理的金鑰來加密您建立的 Image Builder 工作流程，則必須在金鑰政策中授與存取權，Image Builder 才能在加密和解密工作流程資源時使用您的金鑰。您可以隨時撤消存取權。但是，如果您撤銷對金鑰的存取權，Image Builder 將無法存取任何已加密的工作流程。

授與 Image Builder 存取權以使用客戶管理金鑰的程序有兩個步驟，如下所示：

步驟 1：新增 Image Builder 工作流程的金鑰原則權限

若要讓 Image Builder 在建立或使用這些工作流程時加密和解密工作流程資源，您必須在 KMS 金鑰原則中指定權限。

此範例金鑰原則授予 Image Builder 管道的存取權，以便在建立程序期間加密工作流程資源，並解密工作流程資源以使用它們。此原則也會授與金鑰管理員的存取權。加密內容和資源規格使用萬用字元來涵蓋您擁有工作流程資源的所有區域。

作為使用映像工作流程的先決條件，您建立了 IAM 工作流程執行角色，該角色授予 Image Builder 執行工作流程動作的權限。此處主要政策範例中顯示的第一個陳述式的主體必須指定您的 IAM 工作流程執行角色。

如需有關客戶受管金鑰的詳細資訊，請參閱AWS Key Management Service 開發人員指南中的管理客戶受管理金鑰的存取權。

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "Allow access to build images with encrypted workflow",
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/YourImageBuilderExecutionRole"
   },
   "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
   ],
   "Resource": "*",
   "Condition": {
    "StringLike": {
```

```
      "kms:EncryptionContext:aws:imagebuilder:arn":
 "arn:aws:imagebuilder:*:111122223333:workflow/*"
    }
   }
  },
  {
   "Sid": "Allow access for key administrators",
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:root"
   },
   "Action": [
    "kms:*"
   ],
   "Resource": "arn:aws:kms:*:111122223333:key/"
  }
 ]
}
```

步驟 2：授與工作流程執行角色的金鑰存取權

Image Builder 假設用來執行工作流程的 IAM 角色需要使用客戶受管金鑰的權限。如果沒有存取您的金鑰，Image Builder 將無法使用金鑰加密或解密您的工作流程資源。

編輯工作流程執行角色的原則，以新增下列原則陳述式。

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "Allow access to the workflow key",
   "Effect": "Allow",
   "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
   ],
   "Resource": "arn:aws:kms:us-west-2:111122223333:key/key_ID",
   "Condition": {
    "StringLike": {
     "kms:EncryptionContext:aws:imagebuilder:arn":
 "arn:aws:imagebuilder:*:111122223333:workflow/*"
    }
   }
  }
```

```
 ]
}
```

AWS CloudTrail 影像工作流程的事件

下列範例顯示使用客戶管理金鑰儲存的影像工作流程加密和解密的典型 AWS CloudTrail 項目。

範例： GenerateDataKey

此範例顯示當 Image Builder API 動作叫用 AWS KMS GenerateDataKey API 動作時， CloudTrail 事件的外觀。CreateWorkflowImage Builder 必須先加密新的工作流程，才能建立工作流程資源。

```
{
 "eventVersion": "1.08",
 "userIdentity": {
  "type": "AssumedRole",
  "principalId": "PRINCIPALID1234567890:workflow-role-name",
  "arn": "arn:aws:sts::111122223333:assumed-role/Admin/workflow-role-name",
  "accountId": "111122223333",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
  "sessionContext": {
   "sessionIssuer": {
    "type": "Role",
    "principalId": "PRINCIPALID1234567890",
    "arn": "arn:aws:iam::111122223333:role/Admin",
    "accountId": "111122223333",
    "userName": "Admin"
   },
   "webIdFederationData": {},
   "attributes": {
    "creationDate": "2023-11-21T20:29:31Z",
    "mfaAuthenticated": "false"
   }
  },
  "invokedBy": "imagebuilder.amazonaws.com"
 },
 "eventTime": "2023-11-21T20:31:03Z",
 "eventSource": "kms.amazonaws.com",
 "eventName": "GenerateDataKey",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "imagebuilder.amazonaws.com",
 "userAgent": "imagebuilder.amazonaws.com",
 "requestParameters": {
```

```
  "encryptionContext": {
   "aws:imagebuilder:arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/
sample-encrypted-workflow/1.0.0/*",
   "aws-crypto-public-key": "key value"
  },
  "keyId": "arn:aws:kms:us-west-2:111122223333:alias/ExampleKMSKey",
  "numberOfBytes": 32
 },
 "responseElements": null,
 "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
 "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
 "readOnly": true,
 "resources": [
  {
   "accountId": "111122223333",
   "type": "AWS::KMS::Key",
   "ARN": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLEzzzzz"
  }
 ],
 "eventType": "AwsApiCall",
 "managementEvent": true,
 "recipientAccountId": "111122223333",
 "eventCategory": "Management"
}
```

範例：解密

此範例顯示當 Image Builder API 動作叫用 AWS KMS Decrypt API 動作時， CloudTrail 事件的外觀。GetWorkflowImage Builder 管線需要先解密工作流程資源，才能使用它。

```
{
 "eventVersion": "1.08",
 "userIdentity": {
  "type": "AssumedRole",
  "principalId": "PRINCIPALID1234567890:workflow-role-name",
  "arn": "arn:aws:sts::111122223333:assumed-role/Admin/workflow-role-name",
  "accountId": "111122223333",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
  "sessionContext": {
   "sessionIssuer": {
    "type": "Role",
    "principalId": "PRINCIPALID1234567890",
    "arn": "arn:aws:iam::111122223333:role/Admin",
```

```
    "accountId": "111122223333",
    "userName": "Admin"
   },
   "webIdFederationData": {},
   "attributes": {
    "creationDate": "2023-11-21T20:29:31Z",
    "mfaAuthenticated": "false"
   }
  },
  "invokedBy": "imagebuilder.amazonaws.com"
 },
 "eventTime": "2023-11-21T20:34:25Z",
 "eventSource": "kms.amazonaws.com",
 "eventName": "Decrypt",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "imagebuilder.amazonaws.com",
 "userAgent": "imagebuilder.amazonaws.com",
 "requestParameters": {
  "keyId": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLEzzzzz",
  "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
  "encryptionContext": {
   "aws:imagebuilder:arn": "arn:aws:imagebuilder:us-west-2:111122223333:workflow/build/
sample-encrypted-workflow/1.0.0/*",
   "aws-crypto-public-key": "ABC123def4567890abc12345678/90dE/F123abcDEF+4567890abc123D
+ef1=="
  }
 },
 "responseElements": null,
 "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEbbbbb",
 "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
 "readOnly": true,
 "resources": [
  {
   "accountId": "111122223333",
   "type": "AWS::KMS::Key",
   "ARN": "arn:aws:kms:us-west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLEzzzzz"
  }
 ],
 "eventType": "AwsApiCall",
 "managementEvent": true,
 "recipientAccountId": "111122223333",
 "eventCategory": "Management"
```

```
}
```

EC2 Image Builder 中的資料儲存

Image Builder 不會將您的任何記錄儲存在服務中。所有日誌都會儲存在用於建立映像檔的 Amazon EC2 執行個體，或儲存在 Systems Manager 自動化日誌中。

EC2 Image Builder 中的網路間流量隱私權

Image Builder 與內部部署位置之間、區域內 AZ 之間以及 AWS 區 AWS 域之間透過 HTTPS 的連線安全。帳戶之間沒有直接連接。

EC2 Image Builder 的 Identity and Access Management

主題

物件

根據您在 Image Builder 中執行的工作而定，使用方式 AWS Identity and Access Management (IAM) 會有所不同。

服務使用者 — 如果您使用 Image Builder 服務執行工作，則管理員會為您提供所需的認證和權限。當您使用更多的 Image Builder 功能來完成工作時，您可能需要額外的權限。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 Image Builder 中的功能，請參閱疑難排解 EC2 Image Builder 身分和存取。

服務管理員 — 如果您負責公司的 Image Builder 資源，您可能擁有 Image Builder 的完整存取權。決定您的服務使用者應該存取哪些 Image Builder 功能和資源是您的工作。接著，您必須將請求提交給您的

IAM 管理員，來變更您服務使用者的許可。檢閱此頁面上的資訊，了解 IAM 的基本概念。若要深入瞭解貴公司如何搭配 Image Builder 使用 IAM，請參閱EC2 Image Builder 如何與 IAM 搭配使用。

IAM 管理員 — 如果您是 IAM 管理員，您可能想要瞭解如何撰寫政策來管理 Image Builder 存取權限的詳細資訊。若要檢視可在 IAM 中使用的 Image Builder 基於身分識別的政策範例，請參閱。Image Builder 身分型原則

使用身分驗證

有關如何在您的人員和程序中提供身份驗證的詳細資訊 AWS 帳戶，請參閱 IAM 使用者指南中的身分。

EC2 Image Builder 如何與 IAM 搭配使用

在您使用 IAM 管理 Image Builder 的存取權限之前，請先了解哪些 IAM 功能可與 Image Builder 搭配使用。

若要深入瞭解 Image Builder 和其他 AWS 服務如何搭配大多數 IAM 功能使用，請參閱 IAM 使用者指南中的搭配 IAM 使用的AWS 服務。

Image Builder 的身分識別型原則

支援身分型政策	是

身分型政策是可以連接到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。若要了解如何建立身分類型政策，請參閱《IAM 使用者指南》中的建立 IAM 政策。

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

Image Builder 的身分識別型政策範例

若要檢視 Image Builder 以身分識別為基礎的原則範例，請參閱。Image Builder 身分型原則

Image Builder 中的資源型政策

支援以資源基礎的政策	是

資源型政策是附加到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。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，您可以指定在其他帳戶內的所有帳戶或 IAM 實體，作為資源型政策的主體。新增跨帳戶主體至資源型政策，只是建立信任關係的一半。當主體和資源位於不同時 AWS 帳戶，受信任帳戶中的 IAM 管理員也必須授與主體實體 (使用者或角色) 權限，才能存取資源。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 使用者指南》中的 IAM 角色與資源型政策有何差異。

Image Builder 的政策動作

支援政策動作	是

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

JSON 政策的 `Action` 元素描述您可以用來允許或拒絕政策中存取的動作。原則動作通常與關聯的 AWS API 作業具有相同的名稱。有一些例外狀況，例如沒有相符的 API 操作的僅限許可動作。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 Image Builder 動作清單，請參閱服務授權參考中由 EC2 Image Builder 定義的動作。

Image Builder 中的原則動作會在動作之前使用下列前置詞：

```
imagebuilder
```

若要在單一陳述式中指定多個動作，請用逗號分隔。

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

若要檢視 Image Builder 以身分識別為基礎的原則範例，請參閱。Image Builder 身分型原則

Image Builder 的政策資源

支援政策資源	是

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

`Resource` JSON 政策元素可指定要套用動作的物件。陳述式必須包含 `Resource` 或 `NotResource` 元素。最佳實務是使用其 Amazon Resource Name (ARN) 來指定資源。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 Image Builder 資源類型及其 ARN 的清單，請參閱服務授權參考中由 EC2 Image Builder 定義的資源。若要了解可以使用哪些動作指定每個資源的 ARN，請參閱 EC2 Image Builder 定義的動作。

若要檢視 Image Builder 以身分識別為基礎的原則範例，請參閱。Image Builder 身分型原則

Image Builder 的政策條件索引鍵

支援服務特定政策條件金鑰	是

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。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。

若您在陳述式中指定多個 Condition 元素，或是在單一 Condition 元素中指定多個索引鍵， AWS 會使用邏輯 AND 操作評估他們。如果您為單一條件索引鍵指定多個值，請使用邏輯OR運算來 AWS 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，您可以只在使用者使用其 IAM 使用者名稱標記時，將存取資源的許可授予該 IAM 使用者。如需更多資訊，請參閱 IAM 使用者指南中的 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。若要查看所有 AWS 全域條件金鑰，請參閱《IAM 使用者指南》中的AWS 全域條件內容金鑰。

若要查看 Image Builder 條件金鑰清單，請參閱服務授權參考中的 EC2 Image Builder 的條件金鑰。若要了解可以使用條件金鑰的動作和資源，請參閱 EC2 Image Builder 定義的動作。

若要檢視 Image Builder 以身分識別為基礎的原則範例，請參閱。Image Builder 身分型原則

Image Builder 中的 ACL

支援 ACL	否

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

ABAC 與 Image Builder

支援 ABAC (政策中的標籤)	部分

屬性型存取控制 (ABAC) 是一種授權策略，可根據屬性來定義許可。在中 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) 和許多 AWS 資源。為實體和資源加上標籤是 ABAC 的第一步。您接著要設計 ABAC 政策，允許在主體的標籤與其嘗試存取的資源標籤相符時操作。

ABAC 在成長快速的環境中相當有幫助，並能在政策管理變得繁瑣時提供協助。

若要根據標籤控制存取，請使用 `aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 條件金鑰，在政策的條件元素中，提供標籤資訊。

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

如需 ABAC 的詳細資訊，請參閱《IAM 使用者指南》中的什麼是 ABAC?。如要查看含有設定 ABAC 步驟的教學課程，請參閱《IAM 使用者指南》中的使用屬性型存取控制 (ABAC)。

搭配 Image Builder 使用臨時認證

支援臨時憑證	是

當您使用臨時憑據登錄時，某些 AWS 服務 不起作用。如需其他資訊，包括哪些 AWS 服務 與臨時登入資料搭配AWS 服務 使用，請參閱 IAM 使用者指南中的 IAM。

如果您使用除了使用者名稱和密碼以外的任何方法登入，則您正在 AWS Management Console 使用臨時認證。例如，當您 AWS 使用公司的單一登入 (SSO) 連結存取時，該程序會自動建立暫時認證。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。如需切換角色的詳細資訊，請參閱《IAM 使用者指南》中的切換至角色 (主控台)。

您可以使用 AWS CLI 或 AWS API 手動建立臨時登入資料。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，而不是使用長期存取金鑰。如需詳細資訊，請參閱 IAM 中的暫時性安全憑證。

Image Builder 的跨服務主體權限

支援轉寄存取工作階段 (FAS)	是

當您使用 IAM 使用者或角色在中執行動作時 AWS，您會被視為主體。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 會使用主體呼叫的權限 AWS 服務，並結合要求 AWS 服務 向下游服務發出要求。只有當服務收到需要與其 AWS 服務 他資源互動才能完成的請求時，才會發出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱《轉發存取工作階段》。

Image Builder 的服務角色

支援服務角色	是

服務角色是服務擔任的 IAM 角色，可代您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需更多資訊，請參閱 IAM 使用者指南中的建立角色以委派許可給 AWS 服務。

> **Warning**
>
> 變更服務角色的權限可能會中斷 Image Builder 功能。只有在 Image Builder 提供指導時，才編輯服務角色。

Image Builder 的服務連結角色

支援服務連結角色。	否

服務連結角色是一種連結至. AWS 服務服務可以擔任代表您執行動作的角色。服務連結角色會顯示在您的中， AWS 帳戶 且屬於服務所有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

如需有關 Image Builder 服務連結角色的詳細資訊，請參閱使用 EC2 Image Builder 的服務連結角色。

Image Builder 身分型原則

透過 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，還有在何種條件下允許或拒絕動作。Image Builder 支援特定動作、資源和條件索引鍵。有關您在 JSON 政策中使用的所有元素的詳細資訊，請參閱 IAM 使用者指南中的適用於 Amazon EC2 Image Builder 的動作、資源和條件金鑰。

動作

Image Builder 中的原則動作會在動作之前使用下列前置詞：`imagebuilder:`政策陳述式必須包含 `Action` 或 `NotAction` 元素。Image Builder 會定義自己的一組動作，描述您可以使用此服務執行的工作。

若要在單一陳述式中指定多個動作，請用逗號分隔，如下所示：

```
"Action": [
      "imagebuilder:action1",
      "imagebuilder:action2"
```

您也可以使用萬用字元 (*) 來指定多個動作。例如，若要指定開頭是 `List` 文字的所有動作，請包含以下動作：

```
"Action": "imagebuilder:List*"
```

若要查看 Image Builder 動作清單，請參閱 IAM 使用者指南 AWS 服務中的動作、資源和條件金鑰。

使用政策管理存取權

有關如何 AWS 透過建立政策並將其附加到 IAM 身分或 AWS 資源來管理存取權的詳細資訊，請參閱 IAM 使用者指南中的政策和許可。

您與執行個體設定檔相關聯的 IAM 角色必須具有執行映像檔中包含的組建和測試元件的許可。下列 IAM 角色政策必須附加至與執行個體設定檔相關聯的 IAM 角色：

- EC2InstanceProfileForImageBuilder
- EC2InstanceProfileForImageBuilderECRContainerBuilds
- AmazonSSMManagedInstanceCore

資源

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

Resource JSON 政策元素可指定要套用動作的物件。陳述式必須包含 Resource 或 NotResource 元素。最佳實務是使用其 Amazon Resource Name (ARN) 來指定資源。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": "*"
```

映 Image Builder 執行個體資源具有下列 Amazon 資源名稱 (ARN)。

```
arn:aws:imagebuilder:region:account-id:resource:resource-id
```

如需 ARN 格式的詳細資訊，請參閱 Amazon 資源名稱 (ARN) 和 AWS 服務命名空間。

例如，若要在陳述式中指定i-1234567890abcdef0執行個體，請使用下列 ARN。

```
"Resource": "arn:aws:imagebuilder:us-east-1:123456789012:instance/i-1234567890abcdef0"
```

如需指定屬於特定帳戶的所有執行個體，請使用萬用字元 (*)。

```
"Resource": "arn:aws:imagebuilder:us-east-1:123456789012:instance/*"
```

某些 Image Builder 動作 (例如用於建立資源的動作) 無法在特定資源上執行。在這些情況下，您必須使用萬用字元 (*)。

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

許多 EC2 Image Builder API 動作都涉及多個資源。若要在單一陳述式中指定多項資源，請使用逗號分隔 ARN。

```
"Resource": [
      "resource1",
      "resource2"
```

條件索引鍵

Image Builder 提供服務特定的條件金鑰，並支援使用某些全域條件金鑰。若要查看所有 AWS 全域條件金鑰，請參閱 IAM 使用者指南中的AWS 全域條件內容金鑰。提供下列服務特定條件金鑰。

圖像生成器：CreatedResourceTagKeys

使用字串運算子。

使用此鍵可根據要求中存在標籤金鑰來篩選存取。這可讓您管理映像產生器所建立的資源。

可用性 — 此金鑰僅適用於CreateInfrastrucutreConfiguration和 UpdateInfrastructureConfiguration API。

圖像生成器：/CreatedResourceTag<key>

使用字串運算子。

使用此鍵可依附加至 Image Builder 建立之資源的標籤鍵值配對篩選存取。這可讓您透過定義的標籤來管理 Image Builder 資源。

可用性 — 此金鑰僅適用於CreateInfrastrucutreConfiguration和 UpdateInfrastructureConfiguration API。

影像建置器:EC2 MetadataHttpTokens

使用字串運算子。

使用此金鑰可根據要求中指定的 EC2 執行個體中繼資料 HTTP 權杖需求篩選存取。

這個機碼的這個值可以是optional或required。

可用性 — 此金鑰僅適用於CreateInfrastrucutreConfiguration和UpdateInfrastructureConfiguration API。

圖像生成器：StatusTopicArn

使用字串運算子。

使用此金鑰可篩選要求中要發佈終端機狀態通知的 SNS 主題 ARN 的存取。

可用性 — 此金鑰僅適用於CreateInfrastrucutreConfiguration和UpdateInfrastructureConfiguration API。

範例

若要檢視 Image Builder 以身分識別為基礎的原則範例，請參閱。EC2 Image Builder 身分型政策

Image Builder 資源型原則

以資源為基礎的策略指定指定的主體可以在 Image Builder 資源以及在何種情況下執行的動作。Image Builder 支援元件、影像和影像配方的資源型權限原則。資源型政策可讓您依資源將使用許可授予至其他帳戶。您也可以使用以資源為基礎的政策，允許 AWS 服務存取您的元件、影像和映像配方。

如需如何將以資源為基礎的政策附加至元件、映像或影像方案的詳細資訊，請參閱共用 EC2 Image Builder 資源。

> **Note**
>
> 當您使用 Image Builder 更新資源策略時，更新會出現在 RAM 主控台中。

基於 Image Builder 標籤的授權

您可以將標籤附加至 Image Builder 資源，或將請求中的標籤傳遞給 Image Builder。若要根據標籤控制存取，請使用 `imagebuilder:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 條件金鑰，在政策的條件元素中，提供標籤資訊。如需標記 Image Builder 資源的更多資訊，請參閱〈〉標記資源 (AWS CLI)。

Image Builder IAM 角色

IAM 角色是您 AWS 帳戶 內部具有特定許可的實體。

搭配 Image Builder 使用臨時認證

您可以搭配聯合使用暫時憑證、擔任 IAM 角色，或是擔任跨帳戶角色。您可以透過呼叫AssumeRole或等 AWS STS API 作業來取得臨時安全登入資料GetFederationToken。

服務連結角色

服務連結角色可 AWS 服務 讓您存取其他服務中的資源，以代表您完成動作。服務連結角色會顯示在您的 IAM 帳戶中，並由該服務所擁有。具有管理存取權的使用者可以檢視但無法編輯服務連結角色的權限。

Image Builder 支援服務連結角色。如需有關建立或管理 Image Builder 服務連結角色的資訊，請參閱使用 EC2 Image Builder 的服務連結角色。

服務角色

此功能可讓服務代表您擔任服務角色。此角色可讓服務存取其他服務中的資源，以代表您完成動作。服務角色會出現在您的 IAM 帳戶中，且由該帳戶所擁有。這表示具有管理存取權的使用者可以變更此角色的權限。不過，這樣可能會破壞此服務的功能。

EC2 Image Builder 身分型政策

主題

以身分為基礎的原則最佳做法

以身分識別為基礎的原則會決定使用者是否可以在您的帳戶中建立、存取或刪除 Image Builder 資源。這些動作可能會讓您的 AWS 帳戶產生費用。當您建立或編輯身分型政策時，請遵循下列準則及建議事項：

- 開始使用 AWS 受管原則並邁向最低權限權限 — 若要開始將權限授與使用者和工作負載，請使用可授與許多常見使用案例權限的AWS 受管理原則。它們可用在您的 AWS 帳戶. 建議您透過定義特定於您使用案例的 AWS 客戶管理政策，進一步降低使用權限。如需更多資訊，請參閱 IAM 使用者指南中的 AWS 受管政策或任務職能的AWS 受管政策。

- 套用最低許可許可 – 設定 IAM 政策的許可時，請僅授予執行任務所需的權限。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。如需使用 IAM 套用許可的更多相關資訊，請參閱 IAM 使用者指南中的 IAM 中的政策和許可。
- 使用 IAM 政策中的條件進一步限制存取權 – 您可以將條件新增至政策，以限制動作和資源的存取。例如，您可以撰寫政策條件，指定必須使用 SSL 傳送所有請求。您也可以使用條件來授與對服務動作的存取權 (如透過特定) 使用這些動作 AWS 服務，例如 AWS CloudFormation。如需更多資訊，請參閱 IAM 使用者指南中的 IAM JSON 政策元素：條件。
- 使用 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作 – IAM Access Analyzer 驗證新政策和現有政策，確保這些政策遵從 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您編寫安全且實用的政策。如需更多資訊，請參閱 IAM 使用者指南中的 IAM Access Analyzer 政策驗證。
- 需要多因素身份驗證 (MFA) — 如果您的案例需要 IAM 使用者或根使用者 AWS 帳戶，請開啟 MFA 以獲得額外的安全性。若要在呼叫 API 作業時請求 MFA，請將 MFA 條件新增至您的政策。如需更多資訊，請參閱 IAM 使用者指南中的設定 MFA 保護的 API 存取。

如需 IAM 中最佳實務的相關資訊，請參閱 IAM 使用者指南中的 IAM 安全最佳實務。

使用 Image Builder 主控台

若要存取 EC2 Image Builder 主控台，您必須擁有最少一組許可。這些權限可讓您列出和檢視有關 AWS 帳戶. 如果您建立比最基本必要許可更嚴格的身分型政策，則對於具有該政策的實體 (IAM 使用者或角色) 而言，主控台就無法如預期運作。

若要確保您的 IAM 實體可以使用 Image Builder 主控台，您必須將下列其中一個 AWS 受管政策附加至這些實體：

- AWSImageBuilderReadOnlyAccess 政策
- AWSImageBuilderFullAccess 政策

如需 Image Builder 受管理的原則的詳細資訊，請參閱使用 EC2 Image Builder 的受管政策。

> **Important**
>
> 建立 Image Builder 服務連結角色需要此AWSImageBuilderFullAccess原則。將此政策附加到 IAM 實體時，還必須附加以下自訂政策，並包含您要使用的資源名稱`imagebuilder`中沒有的資源：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "sns topic arn"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetInstanceProfile"
            ],
            "Resource": "instance profile role arn"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "instance profile role arn",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": "bucket arn"
        }
    ]
}
```

您不需要為只對 AWS CLI 或 AWS API 進行呼叫的使用者允許最低主控台權限。反之，只需允許存取符合您嘗試執行之 API 作業的動作就可以了。

EC2 Image Builder 資源型政策

若要取得有關如何建立元件的資訊，請參閱使用 Image Builder 管理元件。

限制 Image Builder 元件對特定 IP 位址的存取

下列範例會授與任何使用者對元件執行任何 Image Builder 作業的權限。不過，要求必須源自於條件中所指定的 IP 地址範圍。

此陳述式中的條件會識別允許之 Internet Protocol Version 4 (IPv4) IP 地址的 54.240.143.* 範圍，但有一個例外：54.240.143.188。

Condition區塊會使用 AND NotIpAddress 條件IpAddress和條aws:SourceIp件索引鍵，這是一個 AWS寬度條件索引鍵。如需有關這些條件索引鍵的詳細資訊，請參閱指定原則中的條件。aws:sourceIp IPv4 值會使用標準 CIDR 表示法。如需詳細資訊，請參閱《IAM 使用者指南》中的 IP 地址條件運算子。

```
{
  "Version": "2012-10-17",
  "Id": "IBPolicyId1",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "imagebuilder.GetComponent:*",
      "Resource": "arn:aws:imagebuilder:::examplecomponent/*",
      "Condition": {
         "IpAddress": {"aws:SourceIp": "54.240.143.0/24"},
         "NotIpAddress": {"aws:SourceIp": "54.240.143.188/32"}
      }
    }
  ]
}
```

使用 EC2 Image Builder 的受管政策

受 AWS 管理的策略是由建立和管理的獨立策略 AWS。 AWS 受管理的策略旨在為許多常見使用案例提供權限，以便您可以開始將權限指派給使用者、群組和角色。

請記住， AWS 受管理的政策可能不會為您的特定使用案例授與最低權限權限，因為這些權限可供所有 AWS 客戶使用。我們建議您定義使用案例專屬的客戶管理政策，以便進一步減少許可。

您無法變更受 AWS 管理策略中定義的權限。如果 AWS 更新 AWS 受管理原則中定義的權限，則此更新會影響附加原則的所有主體識別 (使用者、群組和角色)。 AWS 當新的啟動或新 AWS 服務 的 API 操作可用於現有服務時，最有可能更新 AWS 受管理策略。

如需詳細資訊，請參閱《IAM 使用者指南》中的 AWS 受管政策。

AWSImageBuilderFullAccess 政策

此AWSImageBuilderFullAccess原則會針對附加的角色授與 Image Builder 資源的完整存取權，允許角色列出、描述、建立、更新和刪除 Image Builder 資源。此策略也會將目標權限授與 AWS 服務 所需的相關權限，例如，驗證資源或在中顯示帳號的目前資源 AWS Management Console。

許可詳細資訊

此政策包含以下許可：

- Image Builder — 授與管理存取權，以便角色可以列出、描述、建立、更新和刪除 Image Builder 資源。
- Amazon EC2 — 授予 Amazon EC2 的存取權限描述驗證資源是否存在或取得屬於該帳戶的資源清單所需的動作。
- IAM — 授權存取權以取得和使用名稱包含「imagebuilder」的執行個體設定檔，透過 `iam:GetRole` API 動作驗證映 Image Builder 服務連結角色是否存在，以及建立映 Image Builder 服務連結角色。
- L@@ icense Manager — 授與存取權以列出資源的授權配置或授權。
- Amazon S3 — 將存取權授予列出屬於該帳戶的儲存貯體，以及名稱中包含「imagebuilder」的映像產生器儲存貯體。
- Amazon SNS — 將寫入許可授予 Amazon SNS，以驗證包含「影像產生器」主題的主題擁有權。

政策範例

以下是AWSImageBuilderFullAccess策略的範例。

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

```
                "sns:ListTopics"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "arn:aws:sns:*:*:*imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "license-manager:ListLicenseConfigurations",
                "license-manager:ListLicenseSpecificationsForResource"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetInstanceProfile"
            ],
            "Resource": "arn:aws:iam::*:instance-profile/*imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListInstanceProfiles",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
```

```
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::*:instance-profile/*imagebuilder*",
                "arn:aws:iam::*:role/*imagebuilder*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:GetBucketLocation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": "arn:aws:s3::*:*imagebuilder*"
        },
        {
            "Action": "iam:CreateServiceLinkedRole",
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "imagebuilder.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "ec2:DescribeSnapshots",
                "ec2:DescribeVpcs",
                "ec2:DescribeRegions",
```

```
                "ec2:DescribeVolumes",
                "ec2:DescribeSubnets",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeLaunchTemplates"
            ],
            "Resource": "*"
        }
    ]
}
```

AWSImageBuilderReadOnlyAccess 政策

此原AWSImageBuilderReadOnlyAccess則提供對所有 Image Builder 資源的唯讀存取權。授與權限可透過 `iam:GetRole` API 動作驗證 Image Builder 服務連結角色是否存在。

許可詳細資訊

此政策包含以下許可：

- Image Builder — 授與對映 Image Builder 資源的唯讀存取權限。
- IAM — 授與存取權以透過 `iam:GetRole` API 動作驗證 Image Builder 服務連結角色是否存在。

政策範例

以下是AWSImageBuilderReadOnlyAccess策略的範例。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "imagebuilder:Get*",
                "imagebuilder:List*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder"
        }
    ]
}
```

AWSServiceRoleForImageBuilder 政策

此原AWSServiceRoleForImageBuilder則允許 Image Builder 代 AWS 服務 表您呼叫。

許可詳細資訊

當透過 Systems Manager 理員建立角色時，此原則會附加至 Image Builder 服務連結角色。若要檢閱授與的特定權限，請參閱本節中的原則範例。如需有關 Image Builder 服務連結角色的詳細資訊，請參閱使用 EC2 Image Builder 的服務連結角色。

該策略包括以下權限：

- CloudWatch 防護記錄 — 具有建立記 CloudWatch 錄檔並將其上傳至名稱開頭為之任何記錄群組的存取權`/aws/imagebuilder/`。
- Amazon EC2 — 只要正在建立或使用的映像、執行個體和磁碟區已標記為或，Image Builder 就可以使用相關快照、磁碟區、網路界面、子網路、安全群組、授權組態和金鑰對在帳戶中建立映像和啟動 EC2 執行個體的存`CreatedBy: EC2 Image Builder`取`CreatedBy: EC2 Fast Launch`權。

  Image Builder 可以取得 Amazon EC2 映像、執行個體屬性、執行個體狀態、您帳戶可用的執行個體類型、啟動範本、子網路、主機和 Amazon EC2 資源上標籤的相關資訊。

  Image Builder 可以更新映像設定，以啟用或停用在映像標記的帳戶中更快啟動 Windows 執行個體`CreatedBy: EC2 Image Builder`。

  此外，Image Builder 可以啟動、停止和終止帳戶中正在執行的執行個體、共用 Amazon EBS 快照、建立和更新映像以及啟動範本、取消註冊現有映像、新增標記，以及跨您透過政策授予許可的帳戶複寫映像。Ec2ImageBuilderCrossAccountDistributionAccess如前所述，所有這些動作都需要「Image Builder」標籤。
- Amazon ECR — 如果需要容器映像弱點掃描，Image Builder 會授與存取權，以便建立儲存庫，並標記其建立的資源以限制其作業範圍。Image Builder 也會授予存取權，以便在擷取弱點快照後刪除為掃描建立的容器映像檔。
- EventBridge— 授與 Image Builder 建立和管理 EventBridge 規則的存取權。

- IAM — 授予 Image Builder 的存取權，可將您帳戶中的任何角色傳遞至 Amazon EC2，以及虛擬機器匯入/匯出。
- Amazon Inspector — 授予 Image Builder 的存取權，以判斷 Amazon Inspector 何時完成建置執行個體掃描，並收集設定為允許的映像檔的發現項目。
- AWS KMS— 授予 Amazon EBS 加密、解密或重新加密 Amazon EBS 磁碟區的存取權。這對於確保加密磁碟區在 Image Builder 建立映像時能夠運作至關重要。
- L@@ icense Manager — 授予 Image Builder 的存取權，以便透過更新 License Manager 規格`license-manager:UpdateLicenseSpecificationsForResource`。
- Amazon SNS — 您帳戶中的任何 Amazon SNS 主題都會授與寫入許可。
- 系統管理員 — 授予 Image Builder 的存取權，以列出 Systems Manager 命令及其呼叫、清查項目、描述執行個體資訊和自動化執行狀態，以及取得命令叫用詳細資訊。Image Builder 也可以傳送自動化訊號，並停止帳戶中任何資源的自動化執行。

  Image Builder 可以`"CreatedBy": "EC2 Image Builder"`針對下列指令碼檔案標記的任何執行個體發出執行命令叫用：`AWS-RunPowerShellScript`AWS-RunShellScript、或。`AWSEC2-RunSysprep`Image Builder 可以在您的帳戶中啟動 Systems Manager 自動化執行，以執行名稱開頭為的自動化文件`ImageBuilder`。

  Image Builder 也能夠建立或刪除帳戶中任何執行個體的「狀態管理員」關聯，只要關聯文件是`AWS-GatherSoftwareInventory`，並在您的帳戶中建立 Systems Manager 服務連結角色。
- AWS STS— Image Builder 可將存取權限EC2ImageBuilderDistributionCrossAccountRole從您的帳戶指定到角色上信任策略允許的任何帳戶中指定的角色。這是用於跨帳戶映像分配。

政策範例

以下是AWSServiceRoleForImageBuilder策略的範例。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*::snapshot/*",
```

```
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:launch-template/*",
                "arn:aws:license-manager:*:*:license-configuration:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*:*:instance/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": [
                        "EC2 Image Builder",
                        "EC2 Fast Launch"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn",
                        "vmie.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StopInstances",
```

```
                "ec2:StartInstances",
                "ec2:TerminateInstances"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CopyImage",
                "ec2:CreateImage",
                "ec2:CreateLaunchTemplate",
                "ec2:DeregisterImage",
                "ec2:DescribeImages",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeSubnets",
                "ec2:DescribeTags",
                "ec2:ModifyImageAttribute",
                "ec2:DescribeImportImageTasks",
                "ec2:DescribeExportImageTasks",
                "ec2:DescribeSnapshots",
                "ec2:DescribeHosts"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ModifySnapshotAttribute"
            ],
            "Resource": "arn:aws:ec2:*::snapshot/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": [
                        "RunInstances",
                        "CreateImage"
                    ],
                    "aws:RequestTag/CreatedBy": [
                        "EC2 Image Builder",
                        "EC2 Fast Launch"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*:*:export-image-task/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*:*:launch-template/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": [
                        "EC2 Image Builder",
                        "EC2 Fast Launch"
```

```
                ]
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "license-manager:UpdateLicenseSpecificationsForResource"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "sns:Publish"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "ssm:ListCommands",
            "ssm:ListCommandInvocations",
            "ssm:AddTagsToResource",
            "ssm:DescribeInstanceInformation",
            "ssm:GetAutomationExecution",
            "ssm:StopAutomationExecution",
            "ssm:ListInventoryEntries",
            "ssm:SendAutomationSignal",
            "ssm:DescribeInstanceAssociationsStatus",
            "ssm:DescribeAssociationExecutions",
            "ssm:GetCommandInvocation"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": "ssm:SendCommand",
        "Resource": [
            "arn:aws:ssm:*:*:document/AWS-RunPowerShellScript",
            "arn:aws:ssm:*:*:document/AWS-RunShellScript",
            "arn:aws:ssm:*:*:document/AWSEC2-RunSysprep",
            "arn:aws:s3:::*"
        ]
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:instance/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ssm:resourceTag/CreatedBy": [
                        "EC2 Image Builder"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ssm:StartAutomationExecution",
            "Resource": "arn:aws:ssm:*:*:automation-definition/ImageBuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:CreateAssociation",
                "ssm:DeleteAssociation"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:document/AWS-GatherSoftwareInventory",
                "arn:aws:ssm:*:*:association/*",
                "arn:aws:ec2:*:*:instance/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncryptFrom",
                "kms:ReEncryptTo",
                "kms:GenerateDataKeyWithoutPlaintext"
            ],
            "Resource": "*",
```

```
            "Condition": {
                "ForAllValues:StringEquals": {
                    "kms:EncryptionContextKeys": [
                        "aws:ebs:id"
                    ]
                },
                "StringLike": {
                    "kms:ViaService": [
                        "ec2.*.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:DescribeKey"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "kms:ViaService": [
                        "ec2.*.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "kms:CreateGrant",
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": true
                },
                "StringLike": {
                    "kms:ViaService": [
                        "ec2.*.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
```

```
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::*:role/
EC2ImageBuilderDistributionCrossAccountRole"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/imagebuilder/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateLaunchTemplateVersion",
                "ec2:DescribeLaunchTemplates",
                "ec2:ModifyLaunchTemplate",
                "ec2:DescribeLaunchTemplateVersions"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ExportImage"
            ],
            "Resource": "arn:aws:ec2:*::image/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ExportImage"
            ],
            "Resource": "arn:aws:ec2:*:*:export-image-task/*"
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "ec2:CancelExportTask"
            ],
            "Resource": "arn:aws:ec2:*:*:export-image-task/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "ssm.amazonaws.com",
                        "ec2fastlaunch.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:EnableFastLaunch"
            ],
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*:*:launch-template/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "inspector2:ListCoverage",
                "inspector2:ListFindings"
            ],
```

```
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:CreateRepository"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:TagResource"
            ],
            "Resource": "arn:aws:ecr:*:*:repository/image-builder-*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchDeleteImage"
            ],
            "Resource": "arn:aws:ecr:*:*:repository/image-builder-*",
            "Condition": {
                "StringEquals": {
                    "ecr:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "events:DeleteRule",
                "events:DescribeRule",
                "events:PutRule",
```

```
                "events:PutTargets",
                "events:RemoveTargets"
            ],
            "Resource": [
                "arn:aws:events:*:*:rule/ImageBuilder-*"
            ]
        }
    ]
}
```

Ec2ImageBuilderCrossAccountDistributionAccess 政策

此Ec2ImageBuilderCrossAccountDistributionAccess原則會授與 Image Builder 在目標區域中跨帳戶散發映像的權限。此外，Image Builder 可描述、複製和套用標籤至帳戶中的任何 Amazon EC2 映像。該策略還授予透過 `ec2:ModifyImageAttribute` API 動作修改 AMI 權限的能力。

許可詳細資訊

此政策包含以下許可：

- Amazon EC2 — 授與 Amazon EC2 的存取權限可以描述、複製和修改映像的屬性，以及為帳戶中的任何 Amazon EC2 映像建立標籤。

政策範例

以下是Ec2ImageBuilderCrossAccountDistributionAccess策略的範例。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*::image/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "ec2:CopyImage",
                "ec2:ModifyImageAttribute"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

EC2ImageBuilderLifecycleExecutionPolicy 政策

此EC2ImageBuilderLifecycleExecutionPolicy原則授予 Image Builder 執行動作 (例如取代、停用或刪除 Image Builder 映像資源及其基礎資源 (AMI、快照) 的權限，以支援映像生命週期管理工作的自動化規則。

許可詳細資訊

此政策包含以下許可：

- Amazon EC2 — 授予 Amazon EC2 的訪問權限，以便在標記為的帳戶中對 Amazon 機器映像 (AMI) 執行以下操作。`CreatedBy: EC2 Image Builder`
  - 啟用和停用 AMI。
  - 啟用和禁用圖像棄用。
  - 描述並取消註冊 AMI。
  - 描述和修改 AMI 圖像屬性。
  - 刪除與 AMI 關聯的磁碟區快照。
  - 擷取資源的標籤。
  - 從 AMI 中添加或刪除標籤以進行棄用。
- Amazon ECR — 授予 Amazon ECR 存取權，以便在 ECR 儲存庫上使用標籤執行下列批次動作。`LifecycleExecutionAccess: EC2 Image Builder`Batch 動作支援自動容器映像生命週期規則。
  - `ecr:BatchGetImage`
  - `ecr:BatchDeleteImage`

  存取權會在標記為 ECR 儲存庫的存放庫層級授與`LifecycleExecutionAccess: EC2 Image Builder`。
- AWS 資源群組 — 授與 Image Builder 的存取權，以根據標籤取得資源。
- EC2 Image Builder — 授予映像產生器刪除 Image Builder 映像資源的存取權。

政策範例

以下是EC2ImageBuilderLifecycleExecutionPolicy策略的範例。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Ec2ImagePermission",
            "Effect": "Allow",
            "Action": [
                "ec2:EnableImage",
                "ec2:DeregisterImage",
                "ec2:EnableImageDeprecation",
                "ec2:DescribeImageAttribute",
                "ec2:DisableImage",
                "ec2:DisableImageDeprecation"
            ],
            "Resource": "arn:aws:ec2:*::image/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Sid": "EC2DeleteSnapshotPermission",
            "Effect": "Allow",
            "Action": "ec2:DeleteSnapshot",
            "Resource": "arn:aws:ec2:*::snapshot/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                }
            }
        },
        {
            "Sid": "EC2TagsPermission",
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteTags",
                "ec2:CreateTags"
            ],
            "Resource": [
```

```
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*::image/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/DeprecatedBy": "EC2 Image Builder",
                    "aws:ResourceTag/CreatedBy": "EC2 Image Builder"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "DeprecatedBy"
                }
            }
        },
        {
            "Sid": "ECRImagePermission",
            "Effect": "Allow",
            "Action": [
                "ecr:BatchGetImage",
                "ecr:BatchDeleteImage"
            ],
            "Resource": "arn:aws:ecr:*:*:repository/*",
            "Condition": {
                "StringEquals": {
                    "ecr:ResourceTag/LifecycleExecutionAccess": "EC2 Image Builder"
                }
            }
        },
        {
            "Sid": "ImageBuilderEC2TagServicePermission",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "tag:GetResources",
                "imagebuilder:DeleteImage"
            ],
            "Resource": "*"
        }
    ]
}
```

EC2InstanceProfileForImageBuilder 政策

該EC2InstanceProfileForImageBuilder政策授予 EC2 執行個體搭配 Image Builder 使用所需的最低許可。這不包括使用系統管理員代理程式所需的權限。

許可詳細資訊

此政策包含以下許可：

- CloudWatch 防護記錄 — 具有建立記 CloudWatch 錄檔並將其上傳至名稱開頭為之任何記錄群組的存取權`/aws/imagebuilder/`。
- Image Builder-訪問被授予獲得任何 Image Builder 組件。
- AWS KMS— 如果透過加密 Image Builder 元件，則會授與存取權以解密 AWS KMS。
- Amazon S3 — 授予存取權以取得存放在名稱開頭為之 Amazon S3 儲存貯體中的物件`ec2imagebuilder-`。

政策範例

以下是EC2InstanceProfileForImageBuilder策略的範例。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "imagebuilder:GetComponent"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "kms:EncryptionContextKeys": "aws:imagebuilder:arn",
                    "aws:CalledVia": [
```

```
                        "imagebuilder.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::ec2imagebuilder*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/imagebuilder/*"
        }
    ]
}
```

EC2InstanceProfileForImageBuilderECRContainerBuilds 政策

使用映像產生器建立 Docker 映像，然後在 Amazon ECR 容器存放庫中註冊並存放映像時，該EC2InstanceProfileForImageBuilderECRContainerBuilds政策會授予 EC2 執行個體所需的最低許可。這不包括使用系統管理員代理程式所需的權限。

許可詳細資訊

此政策包含以下許可：

- CloudWatch 防護記錄 — 具有建立記 CloudWatch 錄檔並將其上傳至名稱開頭為之任何記錄群組的存取權`/aws/imagebuilder/`。
- Amazon ECR — Amazon ECR 會授予存取權，以取得、註冊和存放容器映像，以及取得授權權杖。
- Image Builder-獲得 Image Builder 組件或容器配方的訪問權限。
- AWS KMS— 授予存取權以解密 Image Builder 元件或容器配方 (如果已透過加密) AWS KMS。
- Amazon S3 — 授予存取權以取得存放在名稱開頭為之 Amazon S3 儲存貯體中的物件`ec2imagebuilder-`。

政策範例

以下是EC2InstanceProfileForImageBuilderECRContainerBuilds策略的範例。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "imagebuilder:GetComponent",
                "imagebuilder:GetContainerRecipe",
                "ecr:GetAuthorizationToken",
                "ecr:BatchGetImage",
                "ecr:InitiateLayerUpload",
                "ecr:UploadLayerPart",
                "ecr:CompleteLayerUpload",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:PutImage"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "kms:EncryptionContextKeys": "aws:imagebuilder:arn",
                    "aws:CalledVia": [
                        "imagebuilder.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::ec2imagebuilder*"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/imagebuilder/*"
        }
    ]
}
```

AWS 受管理策略的 Image Builder 更新

本節提供有關 Image Builder AWS 受管理原則的更新資訊，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 Image Builder 文件歷程記錄頁面上的 RSS 摘要。

變更	描述	日期
EC2ImageBuilderLifecycleExecutionPolicy – 新政策	Image Builder 新增了包含映像生命週期管理權限的新`EC2ImageBuilderLifecycleExecutionPolicy` 原則。	2023 年 11 月 17 日
AWSServiceRoleForImageBuilder – 更新現有政策	Image Builder 對服務角色進行了下列變更，以提供 macOS 支援。 • 新增 ec2：DescribeHosts 啟用 Image Builder 輪詢 hostId，以確定何時處於有效狀態以啟動執行個體。 • 已新增 ssm: GetCommandInvocation、API 動作，以改善 Image Builder 用來取	2023 年 8 月 28 日

變更	描述	日期
	得命令叫用詳細資料的方法。	

變更	描述	日期
AWSServiceRoleForImageBuilder – 更新現有政策	Image Builder 對服務角色進行了下列變更，以允許 Image Builder 工作流程收集 AMI 和 ECR 容器映像組建的弱點發現項目。新的權限支援 CVE 偵測和報告功能。 • 新增檢查器 2：ListCoverage 和檢查器 2：ListFindings 允許 Image Builder 判斷 Amazon Inspector 何時完成測試執行個體掃描，並收集設定為允許執行個體掃描的映像檔的發現項目。 • 新增 ecr:CreateRepository，需要映像產生器使用 `CreatedBy: EC2 Image Builder` (tag-on-create) 標記儲存庫。還添加了 ecr:TagResource（需要 tag-on-create）具有相同標 CreatedBy 籤條件約束，以及需要存儲庫名稱開頭的其他約束。`image-builder-*` 名稱限制可防止權限提升，並防止對映 Image Builder 未建立的儲存庫進行變更。 • 已新增 ecr：BatchDeleteImage 適用於標記為的 ECR 儲存庫。`CreatedBy: EC2`	2023 年 3 月 30 日

變更	描述	日期
	Image Builder此權限需要儲存庫名稱開頭為image-builder-* 。 • 為 Image Builder 新增事件許可，以建立和管理名稱ImageBuilder-* 中包含的 Amazon EventBridge 受管規則。	
AWSServiceRoleForImageBuilder – 更新現有政策	Image Builder 對服務角色進行了下列變更： • 已新增 License Manager 授權做為 ec2 的資源：RunInstance 呼叫允許客戶使用與授權組態相關聯的基礎映像 AMI。	2022 年 3 月 22 日
AWSServiceRoleForImageBuilder – 更新現有政策	Image Builder 對服務角色進行了下列變更： • 已新增 EC2 EnableFastLaunch API 動作的許可，以啟用和停用 Windows 執行個體的更快啟動速度。 • 收緊 ec2 的範圍更多：CreateTags 動作和資源標籤條件。	2022 年 2 月 21 日

變更	描述	日期
AWSServiceRoleForImageBuilder – 更新現有政策	Image Builder 對服務角色進行了下列變更： • 已新增呼叫 VMIE 服務以匯入虛擬機器並從中建立基礎 AMI 的權限。 • 收緊 ec2 的範圍：CreateTags 動作和資源標籤條件。	2021年11月20日
AWSServiceRoleForImageBuilder – 更新現有政策	Image Builder 新增了新權限，以修正多個庫存關聯導致映像組建卡住的問題。	2021 年 8 月 11 日
AWSImageBuilderFullAccess – 更新現有政策	Image Builder 對完整存取角色進行了下列變更： • 添加了允許的權限`ec2:DescribeInstanceTypeOffereings` 。 • 已新增呼叫權限，`ec2:DescribeInstanceTypeOfferings` 以啟用 Image Builder 主控台，以準確反映帳戶中可用的執行個體類型。	2021 年 4 月 13 日
Image Builder 開始追蹤變更	Image Builder 開始追蹤其 AWS 受管理策略的變更。	2021年4月02 日

使用 EC2 Image Builder 的服務連結角色

EC2 Image Builder 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 Image Builder 的唯一 IAM 角色類型。服務連結角色由 Image Builder 預先定義，並包含服務代表您呼叫其他 AWS 服務 人所需的所有權限。

服務連結角色可讓設定 Image Builder 更有效率，因為您不需要手動新增必要的權限。Image Builder 會定義其服務連結角色的權限，除非另有定義，否則只有 Image Builder 可以擔任其角色。已定義的許可包括信任政策和許可政策。許可原則無法附加到其他任何 IAM 實體。

如需其他支援服務連結角色之服務的相關資訊 AWS 服務 ，請參閱使用 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。

Image Builder 的服務連結角色權限

Image Builder 使用AWSServiceRoleForImageBuilder服務連結角色，允許 EC2 Image Builder 代表您存取 AWS 資源。服務連結的角色會信任影像建置工具 .amazon aws.com 服務擔任該角色。

您不需要手動建立這個服務連結角色。當您在 AWS 管理主控台、或 AWS API 中建立第一個映 Image Builder 生器映像時，Image Builder 會為您建立服務連結角色。 AWS CLI

下列動作會建立新映像：

- 在 Image Builder 主控台中執行管線精靈，以建立自訂映像。
- 使用下列其中一個 API 動作或其對應的 AWS CLI 命令：
  - CreateImageAPI 動作 (create-image在中 AWS CLI)。
  - ImportVmImageAPI 動作 (import-vm-image在中 AWS CLI)。
  - StartImagePipelineExecutionAPI 動作 (start-image-pipeline-execution在中 AWS CLI)。

> **Important**
>
> 如果服務連結角色已從您的帳戶中刪除，您可以使用相同的程序重新建立該角色。當您建立第一個 EC2 Image Builder 資源時，映像產生器會再次為您建立服務連結角色。

若要查看的權限 AWSServiceRoleForImageBuilder，請參閱AWSServiceRoleForImageBuilder 政策頁面。若要進一步了解如何設定服務連結角色的許可，請參閱 IAM 使用者指南中的服務連結角色許可。

從您的帳戶移除 Image Builder 服務連結角色

您可以使用 IAM 主控台、或 AWS API AWS CLI，從您的帳戶手動移除 Image Builder 的服務連結角色。但是，在執行此操作之前，您必須確保沒有啟用任何參照它的 Image Builder 資源。

> **Note**
>
> 當您嘗試刪除資源時，如果 Image Builder 服務正在使用此角色，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。

清理由**AWSServiceRoleForImageBuilder**角色使用的 Image Builder 資源

1. 在開始之前，請確認沒有任何管線組建正在執行。若要取消執行中的組建，請使用中的`cancel-image-creation`命令 AWS CLI。

   ```
   aws imagebuilder cancel-image-creation --image-build-version-
   arn arn:aws:imagebuilder:us-east-1:123456789012:image-pipeline/sample-pipeline
   ```

2. 變更所有管線排程以使用手動建置程序，或者如果不再使用它們，則將其刪除。如需刪除資源的詳細資訊，請參閱刪除 EC2 Image Builder 資源。

使用 IAM 刪除服務連結角色

您可以使用 IAM 主控台 AWS CLI、或 AWS API 從您的帳戶刪除`AWSServiceRoleForImageBuilder`角色。如需詳細資訊，請參閱《IAM 使用者指南》中的刪除服務連結角色。

EC2 Image Builder 服務連結角色的支援區域

Image Builder 支援在所有提供服務的 AWS 區域中使用服務連結角色。如需支援的 AWS 區域清單，請參閱AWS 區域和端點。

疑難排解 EC2 Image Builder 身分和存取

主題

- 我沒有在 Image Builder 中執行動作的授權
- 我沒有授權執行 iam：PassRole

- 我想允許我以外的人訪問我 AWS 帳戶 的 Image Builder 資源

我沒有在 Image Builder 中執行動作的授權

如果您收到錯誤，告知您未獲授權執行動作，您的政策必須更新，允許您執行動作。

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

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

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

如果您需要協助，請聯絡您的 AWS 系統管理員。您的管理員提供您的簽署憑證。

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，則必須更新您的原則，以允許您將角色傳遞給 Image Builder。

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

當名為的 IAM 使用者marymajor嘗試使用主控台在 Image Builder 生器中執行動作時，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，請聯絡您的 AWS 系統管理員。您的管理員提供您的簽署憑證。

我想允許我以外的人訪問我 AWS 帳戶 的 Image Builder 資源

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

如需進一步了解，請參閱以下內容：

- 若要瞭解 Image Builder 是否支援這些功能，請參閱EC2 Image Builder 如何與 IAM 搭配使用。
- 若要了解如何提供對您所擁有資源 AWS 帳戶 的存取權，請參閱《IAM 使用者指南》中您擁有的另一 AWS 帳戶 個 IAM 使用者提供存取權限。
- 若要了解如何向第三方提供對資源的存取權 AWS 帳戶，請參閱 IAM 使用者指南中的提供第三方 AWS 帳戶 擁有的存取權。
- 若要了解如何透過聯合身分提供存取權，請參閱 IAM 使用者指南中的將存取權提供給在外部進行身分驗證的使用者 (聯合身分)。
- 若要了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱《IAM 使用者指南》中的 IAM 角色與資源型政策的差異。

EC2 Image Builder 的合規驗證

EC2 Image Builder 不在任何 AWS 合規計劃的範圍內。

如需特定規範遵循 AWS 服務 方案範圍的清單，請參閱合規性計劃範圍AWS 服務 內的AWS 服務)。如需一般資訊，請參閱AWS 規範計劃AWS。

您可以使用下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱下載 AWS 人工因素下載人工因 Ar AWS 。

使用 Image Builder 時的合規責任取決於資料的敏感度、公司的合規目標以及適用的法律和法規。AWS 提供下列資源以協助遵循法規：

- 安全與合規快速入門指南：這些部署指南討論架構考量，並提供在 AWS上部署以安全及合規為重心之基準環境的步驟。
- AWS 合規資源AWS — 此工作簿和指南集合可能適用於您的產業和所在地。
- 使用AWS Config 開發人員指南中的規則評估資源 — 此 AWS Config 服務會評估您的資源組態符合內部實務、產業準則和法規的程度。
- AWS Security Hub— 此 AWS 服務提供安全狀態的全面檢視，協助您檢查您 AWS 是否符合安全性產業標準和最佳做法。

您可以將 AWS 任務協調器和執行器 (AWS TOE) 中的合規產品 AWS Marketplace 或元件合併到 Image Builder 映像中，以協助確保您的映像符合標準。如需詳細資訊，請參閱 適用於圖像產 Image Builder 的合規產品。

EC2 Image Builder 中的彈性

AWS 全球基礎架構是圍繞區 AWS 域和可用區域建立的。 AWS 區域提供多個實體分離和隔離的可用區域，這些區域透過低延遲、高輸送量和高度備援的網路連線。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。

EC2 Image Builder 服務可讓您將在一個區域中建置的映像分發到其他區域，為這些區域提供 AMI 的多區域恢復能力。沒有機制可「備份」影像管線、配方或元件。您可以將配方和元件文件存放在 Image Builder 服務之外，例如在 Amazon S3 儲存貯體中。

EC2 Image Builder 無法設定為高可用性 (HA)。您可以將圖像分發到多個區域，以使圖像更高的可用性。

如需區域和可用區域的相關 AWS 資訊，請參閱AWS 全域基礎結構。

Image Builder 中的基礎結構

AWS 全球網路為 EC2 Image Builder 等服務提供安全功能並控制網路存取。如需有關為其服務 AWS 提供之基礎結構安全性的詳細資訊，請參閱安全性簡介白皮書中的基礎結構 AWS安全性一節。

若要透過 AWS 全球網路傳送 Image Builder API 動作的要求，您的用戶端軟體必須符合下列安全性準則：

- 若要傳送 Image Builder API 動作的要求，用戶端軟體必須使用支援的傳輸層安全性 (TLS) 版本。

> **Note**
>
> AWS 正逐步取消對 TLS 版本 1.0 和 1.1 版的支援。強烈建議您將用戶端軟體更新為使用 TLS 1.2 版或更新版本，以便您仍然可以連線。如需詳細資訊，請參閱此AWS 安全性部落格文章。

- 用戶端軟體必須支援具有完美正向保密性 (PFS) 的密碼套件，例如短暫的迪菲-赫爾曼 (DHE) 或橢圓曲線短暫迪菲-赫爾曼 (ECDHE)。大多數目前的系統 (例如 Java 7 及更新版本) 都支援這些模式。
- 您必須使用存取金鑰 ID 和與 AWS Identity and Access Management (IAM) 主體相關聯的秘密存取金鑰來簽署 API 請求。或者，您可以使用 AWS Security Token Service(AWS STS) 為您的請求生成臨時安全登入資料。

此外，映像產生器用來建立和測試映像檔的 EC2 執行個體必須具有存取權 AWS Systems Manager。

EC2 Image Builder 中的修補管理

EC2 Image Builder 提供最新的 Amazon Linux 2、Amazon Linux 2023、紅帽企業 Linux (RHEL)、CentOS、Ubuntu、SUSE 企業伺服器，以及視窗 2012 R2 及更新版本的 AMI 作為受管映像來源。您可以依照共同的責任模型維護 Amazon EC2 系統修補責任。如果可以輕鬆更換應用程式工作負載中的 EC2 執行個體，則更新基礎 AMI 並根據此映像重新部署所有運算節點可能會更有效率。

以下是您可以使 Image Builder AMI 保持在最新狀態的兩種方法。

- AWS-提供的修補元件 — EC2 Image Builder 提供兩個建置元件`update-windows`，`update-linux`其中會安裝所有擱置中的作業系統更新。這些組件使用`UpdateOS`動作模塊。如需詳細資訊，請參閱 更新。您可以從 AWS提供的元件清單中選取元件，將元件新增至映像建置管道。
- 具有修補作業的自訂建置元件 — 若要在支援 AMI 的作業系統上選擇性地安裝或更新修補程式，您可以編寫 Image Builder 元件以安裝必要的修補程式。自訂元件可以使用 shell 指令碼 (Bash 或 PowerShell) 安裝修補程式，也可以使用`UpdateOS`動作模組來指定要安裝或排除的修補程式。如需詳細資訊，請參閱 AWS TOE 元件管理員支援的動作模組。

  使用`UpdateOS`動作模組的元件 (Linux 和視窗)

```
schemaVersion: 1.0
phases:
  - name: build
 steps:
   - name: UpdateOS
  action: UpdateOS
```

  使用 Bash 來安裝 yum 更新的元件

```
schemaVersion: 1.0
phases:
  - name: build
 steps:
   - name: InstallYumUpdates
  action: ExecuteBash
  inputs:
    commands:
   - sudo yum update -y
```

EC2 Image Builder 的安全最佳實務

EC2 Image Builder 提供許多安全性功能，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。

- 請勿在 Image Builder 方法中使用過於寬鬆的安全性群組。
- 請勿與您不信任的帳戶共用影像。
- 請勿將具有私人或敏感資料的影像公開。
- 在映像檔建置期間套用所有可用的 Windows 或 Linux 安全性修補程式

強烈建議您測試映像檔，以驗證安全狀況和適用的安全性符合性層級。 像 Amazon Inspector 這樣的解決方案可協助驗證影像的安全性和合規狀態。

適用於 Image Builder 管線的 IMDSv2

當 Image Builder 管道執行時，它會傳送 HTTP 要求以啟動 Image Builder 用來建置和測試映像的 EC2 執行個體。若要設定管線用於啟動要求的 IMDS 版本，請在 Image Builder 基礎結構組態執行個體中繼資料設定中設定`httpTokens`參數。

> **Note**
>
> 我們建議您將 Image Builder 從管線組建啟動的所有 EC2 執行個體設定為使用 IMDSv2，以便執行個體中繼資料擷取請求需要已簽署的權杖標頭。

如需 Image Builder 基礎結構組態的詳細資訊，請參閱管理 EC2 Image Builder 基礎設施組。如需有關 Linux 映像的 EC2 執行個體中繼資料選項的詳細資訊，請參閱 Amazon EC2 Linux 執行個體使用者指南中的設定執行個體中繼資料選項。對於 Windows 映像檔，請參閱 Amazon EC2 Windows 執行個體使用者指南中的設定執行個體中繼資料選項。

必要的建置後清理

Image Builder 完成自訂映像檔的所有建置步驟後，映 Image Builder 會準備建置執行個體以進行測試和映像建立。在關閉組建執行個體以建立快照集之前，Image Builder 會執行下列清理，以確保映像檔的安全性：

Linux

Image Builder 管線會執行清理指令碼，以協助確保最終映像檔遵循安全性最佳作法，並移除任何不應轉移至快照集的建置成品或設定。但是，您可以跳過腳本的各個部分，或完全覆蓋用戶數據。因此，Image Builder 管線產生的映像不一定符合任何特定的法規標準。

管線完成其建置和測試階段時，Image Builder 會在建立輸出影像之前，自動執行下列清理指令碼。

> **Important**
>
> 如果您覆寫方案中的使用者資料，指令碼不會執行。在這種情況下，請確保在用戶數據中包含一個命令，該命令可以創建一個名為的空文件`perform_cleanup`。Image Builder 會偵測到此檔案，並在建立新映像之前執行清理程序檔。

```
#!/bin/bash
if [[ ! -f {{workingDirectory}}/perform_cleanup ]]; then
    echo "Skipping cleanup"
    exit 0
else
    sudo rm -f {{workingDirectory}}/perform_cleanup
fi

function cleanup() {
    FILES=("$@")
    for FILE in "${FILES[@]}"; do
        if [[ -f "$FILE" ]]; then
            echo "Deleting $FILE";
            sudo shred -zuf $FILE;
        fi;
        if [[ -f $FILE ]]; then
            echo "Failed to delete '$FILE'. Failing."
            exit 1
        fi;
    done
};


# Clean up for cloud-init files
CLOUD_INIT_FILES=(
    "/etc/sudoers.d/90-cloud-init-users"
    "/etc/locale.conf"
```

```
    "/var/log/cloud-init.log"
    "/var/log/cloud-init-output.log"
)
if [[ -f {{workingDirectory}}/skip_cleanup_cloudinit_files ]]; then
    echo "Skipping cleanup of cloud init files"
else
    echo "Cleaning up cloud init files"
    cleanup "${CLOUD_INIT_FILES[@]}"
    if [[ $( sudo find /var/lib/cloud -type f | sudo wc -l ) -gt 0 ]]; then
        echo "Deleting files within /var/lib/cloud/*"
        sudo find /var/lib/cloud -type f -exec shred -zuf {} \;
    fi;

    if [[ $( sudo ls /var/lib/cloud | sudo wc -l ) -gt 0 ]]; then
        echo "Deleting /var/lib/cloud/*"
        sudo rm -rf /var/lib/cloud/* || true
    fi;
fi;


# Clean up for temporary instance files
INSTANCE_FILES=(
    "/etc/.updated"
    "/etc/aliases.db"
    "/etc/hostname"
    "/var/lib/misc/postfix.aliasesdb-stamp"
    "/var/lib/postfix/master.lock"
    "/var/spool/postfix/pid/master.pid"
    "/var/.updated"
    "/var/cache/yum/x86_64/2/.gpgkeyschecked.yum"
)
if [[ -f {{workingDirectory}}/skip_cleanup_instance_files ]]; then
    echo "Skipping cleanup of instance files"
else
    echo "Cleaning up instance files"
    cleanup "${INSTANCE_FILES[@]}"
fi;


# Clean up for ssh files
SSH_FILES=(
    "/etc/ssh/ssh_host_rsa_key"
    "/etc/ssh/ssh_host_rsa_key.pub"
    "/etc/ssh/ssh_host_ecdsa_key"
```

```
    "/etc/ssh/ssh_host_ecdsa_key.pub"
    "/etc/ssh/ssh_host_ed25519_key"
    "/etc/ssh/ssh_host_ed25519_key.pub"
    "/root/.ssh/authorized_keys"
)
if [[ -f {{workingDirectory}}/skip_cleanup_ssh_files ]]; then
    echo "Skipping cleanup of ssh files"
else
    echo "Cleaning up ssh files"
    cleanup "${SSH_FILES[@]}"
    USERS=$(ls /home/)
    for user in $USERS; do
        echo Deleting /home/"$user"/.ssh/authorized_keys;
        sudo find /home/"$user"/.ssh/authorized_keys -type f -exec shred -zuf {} \;
    done
    for user in $USERS; do
        if [[ -f /home/"$user"/.ssh/authorized_keys ]]; then
            echo Failed to delete /home/"$user"/.ssh/authorized_keys;
            exit 1
        fi;
    done;
fi;


# Clean up for instance log files
INSTANCE_LOG_FILES=(
    "/var/log/audit/audit.log"
    "/var/log/boot.log"
    "/var/log/dmesg"
    "/var/log/cron"
)
if [[ -f {{workingDirectory}}/skip_cleanup_instance_log_files ]]; then
    echo "Skipping cleanup of instance log files"
else
    echo "Cleaning up instance log files"
    cleanup "${INSTANCE_LOG_FILES[@]}"
fi;

# Clean up for TOE files
if [[ -f {{workingDirectory}}/skip_cleanup_toe_files ]]; then
    echo "Skipping cleanup of TOE files"
else
    echo "Cleaning TOE files"
```

```
    if [[ $( sudo find {{workingDirectory}}/TOE_* -type f | sudo wc -l) -gt 0 ]];
 then
        echo "Deleting files within {{workingDirectory}}/TOE_*"
        sudo find {{workingDirectory}}/TOE_* -type f -exec shred -zuf {} \;
    fi
    if [[ $( sudo find {{workingDirectory}}/TOE_* -type f | sudo wc -l) -gt 0 ]];
 then
        echo "Failed to delete {{workingDirectory}}/TOE_*"
        exit 1
    fi
    if [[ $( sudo find {{workingDirectory}}/TOE_* -type d | sudo wc -l) -gt 0 ]];
 then
        echo "Deleting {{workingDirectory}}/TOE_*"
        sudo rm -rf {{workingDirectory}}/TOE_*
    fi
    if [[ $( sudo find {{workingDirectory}}/TOE_* -type d | sudo wc -l) -gt 0 ]];
 then
        echo "Failed to delete {{workingDirectory}}/TOE_*"
        exit 1
    fi
fi

# Clean up for ssm log files
if [[ -f {{workingDirectory}}/skip_cleanup_ssm_log_files ]]; then
    echo "Skipping cleanup of ssm log files"
else
    echo "Cleaning up ssm log files"
    if [[ $( sudo find /var/log/amazon/ssm -type f | sudo wc -l) -gt 0 ]]; then
        echo "Deleting files within /var/log/amazon/ssm/*"
        sudo find /var/log/amazon/ssm -type f -exec shred -zuf {} \;
    fi
    if [[ $( sudo find /var/log/amazon/ssm -type f | sudo wc -l) -gt 0 ]]; then
        echo "Failed to delete /var/log/amazon/ssm"
        exit 1
    fi
    if [[ -d "/var/log/amazon/ssm" ]]; then
        echo "Deleting /var/log/amazon/ssm/*"
        sudo rm -rf /var/log/amazon/ssm
    fi
    if [[ -d "/var/log/amazon/ssm" ]]; then
        echo "Failed to delete /var/log/amazon/ssm"
        exit 1
    fi
fi
```

```
if [[ $( sudo find /var/log/sa/sa* -type f | sudo wc -l ) -gt 0 ]]; then
    echo "Deleting /var/log/sa/sa*"
    sudo shred -zuf /var/log/sa/sa*
fi
if [[ $( sudo find /var/log/sa/sa* -type f | sudo wc -l ) -gt 0 ]]; then
    echo "Failed to delete /var/log/sa/sa*"
    exit 1
fi

if [[ $( sudo find /var/lib/dhclient/dhclient*.lease -type f | sudo wc -l ) -gt
 0 ]]; then
        echo "Deleting /var/lib/dhclient/dhclient*.lease"
        sudo shred -zuf /var/lib/dhclient/dhclient*.lease
fi
if [[ $( sudo find /var/lib/dhclient/dhclient*.lease -type f | sudo wc -l ) -gt
 0 ]]; then
        echo "Failed to delete /var/lib/dhclient/dhclient*.lease"
        exit 1
fi

if [[ $( sudo find /var/tmp -type f | sudo wc -l) -gt 0 ]]; then
        echo "Deleting files within /var/tmp/*"
        sudo find /var/tmp -type f -exec shred -zuf {} \;
fi
if [[ $( sudo find /var/tmp -type f | sudo wc -l) -gt 0 ]]; then
        echo "Failed to delete /var/tmp"
        exit 1
fi
if [[ $( sudo ls /var/tmp | sudo wc -l ) -gt 0 ]]; then
        echo "Deleting /var/tmp/*"
        sudo rm -rf /var/tmp/*
fi

# Shredding is not guaranteed to work well on rolling logs

if [[ -f "/var/lib/rsyslog/imjournal.state" ]]; then
        echo "Deleting /var/lib/rsyslog/imjournal.state"
        sudo shred -zuf /var/lib/rsyslog/imjournal.state
        sudo rm -f /var/lib/rsyslog/imjournal.state
fi

if [[ $( sudo ls /var/log/journal/ | sudo wc -l ) -gt 0 ]]; then
```

```
        echo "Deleting /var/log/journal/*"
        sudo find /var/log/journal/ -type f -exec shred -zuf {} \;
        sudo rm -rf /var/log/journal/*
fi

sudo touch /etc/machine-id
```

Windows

Image Builder 管線自訂視窗映像後，它會執行 Microsoft Sysprep 公用程式。 這些動作會遵循強化和清理影像的AWS 最佳作法。

覆寫 Linux 清理指令碼

Image Builder 會建立預設安全的映像檔，並遵循我們的安全性最佳做法。但是，某些更進階的使用案例可能需要您略過內建清理指令碼的一或多個區段。如果您確實需要略過某些清理，我們強烈建議您測試輸出 AMI 以確保映像的安全性。

Important

略過清理指令碼中的區段可能會導致機密資訊，例如擁有者帳戶詳細資料或安全殼層金鑰包含在最終映像檔中，以及從該映像檔啟動的任何執行個體。在不同的可用區域、區域或帳戶中啟動時，您也可能會遇到問題。

下表概述清理指令集的各個區段、在該區段中刪除的檔案，以及可用來標記 Image Builder 應略過的區段的檔案名稱。若要略過清理指令碼的特定區段，您可以使用CreateFile元件動作模組或使用者資料中的指令 (如果覆寫)，以 「略過區段檔案名稱」欄中指定的名稱建立空白檔案。

Note

您建立用來略過清理指令碼某個區段的檔案不應包含副檔名。例如，
如果您想略過指令碼CLOUD_INIT_FILES區段，但建立名為的檔
案skip_cleanup_cloudinit_files.txt，Image Builder 將無法辨識略過檔案。

輸入

清理區段	已移除檔案	略過段落檔案名稱
CLOUD_INIT_FILES	/etc/sudoers.d/90-cloud-init-users /etc/locale.conf /var/log/cloud-init.log /var/log/cloud-init-output.log	skip_cleanup_cloud init_files
INSTANCE_FILES	/etc/.updated /etc/aliases.db /etc/hostname /var/lib/misc/postfix.aliasesdb-stamp /var/lib/postfix/master.lock /var/spool/postfix/pid/master.pid /var/.updated /var/cache/yum/x86_64/2/.gpgkeyschecked.yum	skip_cleanup_instance_files
SSH_FILES	/etc/ssh/ssh_host_rsa_key /etc/ssh/ssh_host_rsa_key.pub	skip_cleanup_ssh_files

清理區段	已移除檔案	略過段落檔案名稱
	/etc/ssh/ssh_host_ecdsa_key /etc/ssh/ssh_host_ecdsa_key.pub /etc/ssh/ssh_host_ed25519_key /etc/ssh/ssh_host_ed25519_key.pub /root/.ssh/authorized_keys /home/<all users>/.ssh/authorized_keys;	
INSTANCE_LOG_FILES	/var/log/audit/audit.log /var/log/boot.log /var/log/dmesg /var/log/cron	skip_cleanup_instance_log_files
TOE_FILES	{{workingDirectory}}/TOE_*	skip_cleanup_toe_files
SSM_LOG_FILES	/var/log/amazon/ssm/*	skip_cleanup_ssm_log_files

疑難排解 EC2 Image Builder

EC2 Image Builder 與 AWS 服務 整合以進行監控和疑難排解，以協助您疑難排解映像建置問題。Image Builder 會追蹤並顯示映像建置程序中每個步驟的進度。此外，Image Builder 可以將日誌匯出到您提供的 Amazon S3 位置。

對於進階疑難排解，您可以使用執行命令執行預先定義的命令AWS Systems Manager 和指令碼

目錄

疑難排解管線建

如果 Image Builder 管線組建失敗，Image Builder 會傳回描述失敗的錯誤訊息。Image Builder 也會`workflow execution ID`在失敗訊息中傳回，例如下列範例輸出中的錯誤訊息：

```
Workflow Execution ID: wf-12345abc-6789-0123-abc4-567890123abc failed with reason: …
```

Image Builder 會透過在其標準映像建立程序中為執行階段定義的一系列步驟來排列和引導影像建立動作。流程的建置和測試階段每個階段都有相關聯的工作流程。Image Builder 執行工作流程以建立或測試新影像時，會產生工作流程中繼資料資源，以追蹤執行階段詳細資料。

容器映像具有在發佈期間執行的額外工作流程。

研究工作流程的執行階段執行個體失敗詳細

若要疑難排解工作流程的執行階段失敗，您可以使用`workflow execution ID`.
GetWorkflowExecutionListWorkflowStepExecutions

檢閱工作流程執行時

- Amazon CloudWatch 日誌

  Image Builder 會將詳細的工作流程執行記錄發佈至下列 Image Builder CloudWatch 記錄群組和串流

  LogGroup:

  ```
  /aws/imagebuilder/ImageName
  ```

LogStream (x.x.x/x)：

```
ImageVersion/ImageBuildVersion
```

使用 CloudWatch 日誌，您可以使用過濾器模式搜索日誌數據。如需詳細資訊，請參閱 Amazon CloudWatch Logs 使用者指南中的使用篩選模式搜尋日誌資料。

- AWS CloudTrail

  CloudTrail 如果在您的帳戶中激活了所有構建活動，也會登錄該活動。您可以依來源篩選 CloudTrail 事件`imagebuilder.amazonaws.com`。或者，您也可以搜尋執行日誌中傳回的 Amazon EC2 執行個體 ID，以查看有關管道執行的更多詳細資訊。

- Amazon Simple Storage Service (S3)

  如果您已在基礎設施組態中指定 S3 儲存貯體名稱和 key prefix 置詞，工作流程步驟執行階段日誌路徑會遵循以下模式：

  ```
  S3://S3BucketName/KeyPrefix/ImageName/ImageVersion/ImageBuildVersion/
  WorkflowExecutionId/StepName
  ```

  您傳送到 S3 儲存貯體的日誌會顯示映像建立程序期間 EC2 執行個體上活動的步驟和錯誤訊息。記錄檔包括來自元件管理員的記錄輸出、已執行元件的定義，以及在執行個體上執行的所有步驟的詳細輸出 (以 JSON 格式)。如果遇到問題，您應該先檢閱這些檔案`application.log`，以診斷執行個體上發生問題的原因。

依預設，Image Builder 會關閉管道發生故障時正在執行的 Amazon EC2 建置或測試執行個體。您可以變更管道所使用之基礎結構組態資源的執行個體設定，以保留組建或測試執行個體以進行疑難排解。

若要變更主控台中的執行個體設定，您必須清除位於基礎結構組態資源的 [疑難排解設定] 區段中的 [失敗時終止執行個體] 核取方塊。

您也可以使用中的update-infrastructure-configuration指令變更例證設定 AWS CLI。將命令與`--cli-input-json`參數一起參考的 JSON 檔案`false`中的`terminateInstanceOnFailure`值設定為。如需詳細資訊，請參閱 更新基礎架構組態。

故障診斷方案

本節列出下列詳細的疑難排解案例：

- 拒絕訪問-狀態碼 403
- 驗證組建執行個體上的系統管理員代理程式可用性時，建置逾時
- Windows 輔助磁碟在啟動時處於離線狀態
- 使用 CIS 強化基礎映像，構建失敗
- AssertInventoryCollection 失敗（Systems Manager 自動化）

若要查看案例的詳細資訊，請選擇案例標題以將其展開。您可以同時展開多個標題。

拒絕訪問-狀態碼 403

描述

管線建置失敗，顯示「AccessDenied: 拒絕存取狀態碼:403"。

原因

可能的原因包括：

- 執行個體設定檔沒有存取 API 或元件資源的必要權限。
- 執行個體設定檔角色缺少記錄 Amazon S3 所需的許可。最常見的情況是，當執行個體設定檔角色沒有 S3 儲存貯體的PutObject許可時，就會發生這種情況。

解決方案

根據原因，可以解決此問題，如下所示：

- 執行個體設定檔遺失受管政策 — 將遺失的政策新增至您的執行個體設定檔角色。然後再次執行管線。
- 執行個體設定檔缺少 S3 儲存貯體的寫入許可 — 將政策新增至您的執行個體設定檔角色，以授予寫入 S3 儲存貯體的PutObject權限。然後再次執行管線。

驗證組建執行個體上的系統管理員代理程式可用性時，建置逾時

描述

管線建置失敗，並顯示「狀態 = TimedOut "」和「失敗訊息 = '當步驟驗證目標執行個體上的 Systems Manager 代理程式可用性時，步驟逾時'」。

原因

可能的原因包括：

- 為執行建置作業和執行元件而啟動的執行個體無法存取 Systems Manager 端點。
- 執行個體設定檔沒有必要的權限。

解決方案

根據可能的原因，可以解決此問題，如下所示：

- 存取問題、私有子網路 — 如果您在私有子網路中建置，請確定已為 Systems Manager、Image Builder 設定PrivateLink 端點，以及 Amazon S3/(如果您想要記錄) 的端點。CloudWatch如需有關設定 PrivateLink 端點的詳細資訊，請參閱 VPC 端點概念 (AWS PrivateLink)。
- 遺失許可 — 將下列受管政策新增至 Image Builder 的 IAM 服務連結角色：
  - EC2 InstanceProfileForImageBuilder
  - EC2 InstanceProfileForImageBuilder ECR ContainerBuilds
  - 亚马孙 ManagedInstanceCore

  如需 Image Builder 服務連結角色的詳細資訊，請參閱使用 EC2 Image Builder 的服務連結角色。

Windows 輔助磁碟在啟動時處於離線狀態

描述

當用來建置 Image Builder Windows AMI 的執行個體類型與用於從 AMI 啟動的執行個體類型不符時，可能會發生非根磁碟區在啟動時離線的問題。這主要發生在構建實例使用比啟動實例更新的架構時。

下列範例示範在 EC2 Nitro 執行個體類型上建立映像產生器 AMI 並在 EC2 Xen 執行個體上啟動時會發生什麼情況：

構建實例類型：m5. 大（硝基）

啟動執行個體類型：t2. 中型 (Xen)

```
PS C:\Users\Administrator> get-disk
Number  Friendly Name  Serial Number         Health Status  Operational Status  Total
 Size  Partition Style
```

```
------  -------------  -------------         -------------  ------------------
 ----------  ---------------
0       AWS PVDISK     vol0abc12d34e567f8a9  Healthy        Online                   30
 GB  MBR
1       AWS PVDISK     vol1bcd23e45f678a9b0  Healthy        Offline                   8
 GB  MBR
```

原因

由於 Windows 預設設定，新發現的磁碟不會自動上線和格式化。在 EC2 上變更執行個體類型時，Windows 會將其視為已探索的新磁碟。這是因為基礎的驅動程序更改。

解決方案

建議您在建置要從中啟動的 Windows AMI 時，使用相同的執行個體類型系統。請勿在基礎結構組態中包含在不同系統上建置的執行個體類型。如果您指定的任何執行個體類型使用 Nitro 系統，則它們都應該使用 Nitro 系統。

如需有關在 Nitro 系統上建置的執行個體的詳細資訊，請參閱 Amazon EC2 Windows 執行個體使用者指南中的以 Nitro 系統建置的執行個體。

使用 CIS 強化基礎映像，構建失敗

描述

您正在使用 CIS 強化基礎映像，並且構建失敗。

原因

將目`/tmp`錄分類為時`noexec`，可能會導致 Image Builder 失敗。

解決方案

在影像配方`workingDirectory`欄位中為您的工作目錄選擇不同的位置。如需詳細資訊，請參閱資ImageRecipe料類型說明。

AssertInventoryCollection 失敗（Systems Manager 自動化）

描述

Systems Manager 自動化顯示`AssertInventoryCollection`自動化步驟失敗。

原因

您或您的組織可能已建立系統管理員狀態管理員關聯，以收集 EC2 執行個體的庫存資訊。如果已為 Image Builder 管線啟用增強型映像中繼資料收集 (這是預設值)，Image Builder 會嘗試為組建執行個體建立新的詳細目錄關聯。不過，Systems Manager 不允許受管理執行個體進行多個庫存關聯，如果已經存在，則會阻止新的關聯。這會導致作業失敗，並導致管線建置失敗。

解決方案

若要解決此問題，請使用下列其中一種方法關閉增強型影像中繼資料收集：

- 在主控台中更新映像管道，以清除 [啟用增強的中繼資料收集] 核取方塊。儲存變更並執行管線組建。

  如需使用 EC2 Image Builder 主控台更新 AMI 映像管道的詳細資訊，請參閱更新 AMI 映像管道（控制台）。如需使用 EC2 Image Builder 主控台更新容器映像管道的詳細資訊，請參閱更新容器映像管道（控制台）。
- 您也可以使用中的update-image-pipeline指令更新映像管線 AWS CLI。若要這麼做，請將EnhancedImageMetadataEnabled屬性包含在 JSON 檔案中，設定為false。下列範例顯示設定為的性質false。

```
{
    "name": "MyWindows2019Pipeline",
    "description": "Builds Windows 2019 Images",
    "enhancedImageMetadataEnabled": false,
    "imageRecipeArn": "arn:aws:imagebuilder:us-west-2:123456789012:image-recipe/my-example-recipe/2020.12.03",
    "infrastructureConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:infrastructure-configuration/my-example-infrastructure-configuration",
    "distributionConfigurationArn": "arn:aws:imagebuilder:us-west-2:123456789012:distribution-configuration/my-example-distribution-configuration",
    "imageTestsConfiguration": {
        "imageTestsEnabled": true,
        "timeoutMinutes": 60
    },
    "schedule": {
        "scheduleExpression": "cron(0 0 * * SUN *)",
        "pipelineExecutionStartCondition":
 "EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE"
    },
```

```
    "status": "ENABLED"
}
```

若要防止新管道發生這種情況，請在使用 EC2 Image Builder 主控台建立新管道時清除「啟用增強型中繼資料收集」核取方塊，或在使用建立管道`false`時將 JSON 檔案中的`EnhancedImageMetadataEnabled`屬性值設定為 AWS CLI。

EC2 Image Builder 使用指南的文件歷史記錄

下表依日期描述文件的重要變更。如需有關此文件更新的通知，您可以訂閱 RSS 訂閱源。

- 應用程式介面版本：

變更	描述	日期
STIG 第一季更新	為 2024 年第一季度發行版本更新了 Linux STIG 版本和應用了 STIG。視窗版本沒有變更。	2024年2月23 日
功能發佈：影像工作流程管理	透過影像工作流程，您可以更靈活地掌握影像建立程序，以及更多的可視性和控制權。您可以自訂工作流程的建置和測試步驟，也可以使用 Image Builder 預設工作流程。	2023 年 12 月 12 日
斯蒂格第 4 季更新	為 2023 年第四季度發行版本更新了 Linux STIG 版本和應用了 STIG。視窗版本沒有變更。還更新了 Linux 和視窗 SCAP 以獲取新的組件，軟件和基準測試編號。	2023 年 12 月 7 日
功能發布：映像生命週期管理	使用映像生命週期管理政策和規則，您可以定義資源管理策略，以確保過時的映像檔及其相關資源經過標記和移除程序。	2023 年 11 月 17 日
斯蒂格第 3 季更新	為 2023 年第三季發行版本更新了 STIG 版本和應用了 STIG。此外，更新的消息來澄清第三方軟件包不會自動安	2023 年 10 月 5 日

	裝，只有很少的例外。系統會記錄所有略過的 STIG。	
新的 STIG 版本	為 2023 年第二季發行版本更新了 STIG 版本和應用了 STIG。	2023 年 5 月 3 日
新的 STIG 版本	為 2023 年第一季發行版本更新了 STIG 版本和應用了 STIG。增加了對 AL2023 的支持。	2023 年 4 月 14 日
更新支援的地區 AWS TOE	新增對下列各項的 AWS TOE 支援 AWS 區域：亞太區域 (海德拉巴)、亞太區域 (雅加達)、歐洲 (蘇黎世)、歐洲 (西班牙) 和中東 (阿拉伯聯合大公國)。	2023 年 4 月 13 日
AWS TOE 應用下載更新	更新了 Windows 上 AWS TOE 安裝下載的簽名。同時更新了 TLS 注意事項，從 S3 儲存貯體下載的應用程式現在需要 TLS 1.2 版或更新版本。	2023 年 3 月 31 日
功能發布：增強的構建工作流	在映像構建版本詳細信息的新工作流程選項卡中添加了映像構建的運行時詳細信息。已改善疑難排解組建的資訊。	2023 年 3 月 30 日

功能發布：CVE 偵測和報告	對於已啟動 Amazon Inspector 掃描的帳戶，Image Builder 可以在建置程序的測試階段針對新映像的測試階段 (包括儲存在 Amazon ECR 的容器映像) 擷取 Amazon Inspector 的常見弱點和暴露 (CVE) 發現項目。Image Builder 會建立發現項目的快照，以支援詳細分析。Image Builder 也會報告可依帳戶、管線或影像篩選的發現項目計數，並可向下鑽研詳細資料。	2023 年 3 月 30 日
新增版本歷史	添加了版本歷史記錄到視窗和 Linux 部分。	2023 年 2 月 17 日
新的 STIG 版本	針對 2022 年第四季度發行的更新 STIG 版本和應用了 STIG。	2023 年 2 月 1 日
功能發布： AWS Marketplace 集成和 CIS 強化	新增 AWS Marketplace 整合功能，可輕鬆尋找並使用訂閱的映像作為新自訂映像檔的基準，包括 CIS 強化映像，以及來自網際網路安全中心的新 CIS 強化元件。	2023 年 1 月 13 日
CIS 硬化組件	增加了 CIS 擁有和維護的 CIS 硬化組件。	2023 年 1 月 13 日
新的 STIG 版本	推出 Ubuntu 支援、更新的 STIG 版本，並在 2022 年第二季度發行版中套用了 STIG。	2022 年 7 月 20 日
文件更新:瀏覽建立 YAML 元件文件頁面	將「建立 YAML」元件文件內容移至自己的頁面，並更新其他頁面以參照它。	2022 年 6 月 7 日

新的 STIG 版本	針對 2022 年第一季發行版本更新了 STIG 版本和應用的 STIG。	2022 年 4 月 25 日
新增 ExecuteDocument 動作模塊	增加了文檔下的`ExecuteDocument` 操作模塊`General execution` 。	2022 年 3 月 28 日
功能發布：Support 更快的啟動視窗 AMI	已新增散發組態設定，以支援 Windows AMI 的更快速啟動。	2022 年 2 月 21 日
維護版本：更新 AWS TOE 二進位指紋	更新 AWS TOE 簽署者憑證的二進位指紋。	2022 年 2 月 18 日
功能發布：配置輸入 AWS TOE	增加了對使用 JSON 配置文件作為 AWS TOE run命令輸入的支持。	2022 年 2 月 3 日
新的 STIG 版本	針對 2021 年第四季發行的更新 STIG 版本和應用了 STIG。此外，新增 SCAP 符合性檢查程式 (SCC) 元件的區段。	2021 年 12 月 22 日
功能發佈：虛擬機器匯入/匯出 (VMIE) 整合	增加了對通過所有通道（控制台，API/CLI 等）進行虛擬機導入的支持，並通過 API/CLI 導出虛擬機。映像產生器主控台目前無法使用虛擬機器匯出。	2021 年 12 月 20 日
功能發佈：AMI 共享 AWS Organizations 和 OU	已更新發佈組態，以新增與 OU 共用輸出 AMI AWS Organizations 的支援。	2021 年 11 月 24 日
文件更新：更新元件階段和階段	Image Builder 中元件階段的擴充內容，以及這些階段如何與 AWS TOE 元件階段互動。	2021 年 9 月 22 日
文件更新：新增 CloudTrail 整合內容	新增監控摘要和 CloudTrail 整合內容。	2021 年 9 月 17 日

新的 STIG 版本	針對 2021 年第三季發行的更新 STIG 版本和應用了 STIG。	2021 年 9 月 10 日
功能發布：Amazon EventBridge 集成	已新增 EventBridge 支援，可讓您將 Image Builder 與相關事件連線 AWS 服務，並根據中定義的規則起始事件 EventBridge。	2021 年 8 月 18 日
文件更新：重新排列 AWS TOE 頁面	為清晰起見，重新排列 AWS TOE 頁面。	2021 年 8 月 11 日
功能發布：參數化組件和其他實例配置	增加了對指定參數以自定義配方組件的支持。用於建置和測試映像的 EC2 執行個體的擴充組態，包括指定要在啟動時執行的命令的能力，以及對 Systems Manager 代理程式的安裝和移除進行更多控制。	2021 年 7 月 7 日
新的 STIG 版本	針對 2021 年第二季發行的更新 STIG 版本和應用了 STIG。	2021 年 6 月 30 日
增強功能：標記增強	改善資源標記的訊息傳遞。	2021 年 6 月 25 日
功能發布：啟動模板集成	增加了對在分發設置中使用 AMI 分發的 Amazon EC2 啟動模板的支持。	2021 年 4 月 7 日
功能發布：容器構建增強功能	已新增設定區塊裝置對應，並指定 AMI 做為容器組建的基礎映像檔的支援。	2021 年 4 月 7 日
新的 STIG 版本	更新 STIG 版本和應用 STIGS。	2021 年 3 月 5 日

更新 cron 表達式	Image Builder cron 處理已更新，可將 cron 運算式精細度提高至分鐘，並使用標準的 Cron 排程引擎。範例會更新為新格式。	2021 年 2 月 8 日
功能發布：容器支持	增加了對使用映像生成器創建 Docker 容器映像的支持，並在亞馬遜彈性容器註冊表（Amazon ECR）上註冊和存儲生成的圖像。內容已重新排列，以反映新功能並適應 future 的增長。	2020 年 12 月 17 日
重組的 cron 文件	此頁面現在會強調顯示有關 cron 如何與 Image Builder 管線組建搭配使用的詳細資訊，並包含 UTC 時間的詳細資訊。特定欄位不允許使用的萬用字元已移除。範例現在包括主控台和 CLI 的運算式範例。	2020 年 11 月 13 日
控制台版本 2.0：更新的管道編輯	入門和建立管線教學課程中的內容變更，以及「管理映像管線」頁面，以整合新的主控台功能和流程。	2020 年 11 月 13 日
新的 STIG 版本	更新 STIG 版本和應用 STIGS。附註-清單格式已變更為顯示預設套用的 STIG。	2020 年 10 月 15 日
Support 循環構造 AWS TOE	建立迴圈建構，以定義 AWS TOE 應用程式中重複的指令序列。	2020 年 7 月 29 日
Support 本地 AWS TOE 組件開發	使用應用程式在本機開發和測試影像元 AWS TOE 件。	2020 年 7 月 28 日

加密的 AMI	EC2 Image Builder 新增對加密 AMI 分發的支援。	2020 年 7 月 1 日
AutoScaling 棄用	取代啟動執行個體 AutoScaling 的使用。	2020年6月15日
通過連接 Support AWS PrivateLink	您可以建立介面 VPC 端點，在 VPC 和 EC2 Image Builder 之間建立私有連線。介面端點採用這項技術 AWS PrivateLink，可讓您在沒有網際網路閘道、NAT 裝置、VPN 連線或 AWS 直 Connect 連線的情況下，私密存取 Image Builder API。VPC 中的執行個體不需要公用 IP 位址即可與 Image Builder API 通訊。您的 VPC 和 Image Builder 之間的流量不會離開 Amazon 網路。	2020 年 6 月 10 日
新的 STIG 版本	更新 STIG 版本和應用 STIGS。	2020 年 1 月 23 日
疑難排解	新增一般疑難排解案例。	2020 年 1 月 22 日
STIG 組件	您可以使用 STIG 元件建立符合 AWS TOE STIG 標準的影像。	2020 年 1 月 22 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。