開發人員指南

AWS Panorama

AWS Panorama: 開發人員指南

Table of Contents

什麼是 AWS Panorama？

AWS Panorama是為您的內部部署攝影機網路帶來電腦視覺的服務。您可以將AWS Panorama設備或數據中心中的其他兼容設備，請通過以下方式註冊AWS Panorama，並從雲端部署電腦視覺應用程式。AWS Panorama可與您現有的即時串流通訊協定 (RTSP) 網路攝影機搭配使用。該設備執行安全的電腦視覺應用程式AWS合作夥伴，或您自行建置的應用程式AWS Panorama應用程式 SDK。

所以此AWS Panorama設備是使用功能強大的緊湊型邊緣設備 system-on-module 針對機器學習工作負載最佳化的 (SOM)。該設備可以對多個視頻流並行運 parallel 多個計算機視覺模型，並實時輸出結果。此元件專為在商業和工業環境中使用而設計，並獲得防塵和液體防護 (IP-62) 等級。

所以此AWS Panorama設備可讓您在邊緣執行獨立的電腦視覺應用程式，而無需將映像傳送到 AWS 雲端。透過使用 AWS 開發套件，您可以與其他 AWS 服務整合，並使用這些服務追蹤來自應用程式一段時間的資料。透過將與其他 AWS 服務整合，您可以使用AWS Panorama執行下列動作：

- 分析流量模式— 使用 AWS 開發套件記錄資料，以便在 Amazon DynamoDB 中進行零售分析。使用無伺服器應用程式來分析一段時間內收集的資料、偵測資料中的異常情況，以及預測 future 的行為。
- 接收工地安全警報— 監控在工業現場禁區. 當您的應用程式偵測到潛在不安全的情況時，請將影像上傳到 Amazon Simple Storage Service (Amazon S3) 並向 Amazon Simple Notification Service (Amazon SNS)) 主題的 Amazon Simple Notification Service (Amazon SNS)。
- 改善品質控制— 監控裝配線的輸出，以識別不符合要求的零件。使用文字和邊框反白顯示不符合零件的影像，並將其顯示在監視器上，以供品質控制團隊檢閱。
- 收集訓練與測試資料— 上傳您的計算機視覺模型無法識別的對象的圖像，或者模型對其猜測的信心是邊界線的地方。使用無伺服器應用程式建立需要標記的映像佇列。標記圖像並使用它們在亞馬遜重新訓練模型 SageMaker。

AWS Panorama使用其他 AWS 服務來管理AWS Panorama應用裝置、存取模型和程式碼，以及部署應用程式。AWS Panorama在不需要您與其他服務互動的情況下盡可能地執行，但對以下服務的了解可以幫助您了解如何使用AWS Panorama工作。

- SageMaker— 您可以使用 SageMaker 從攝影機或感測器收集訓練資料、建立機器學習模型，並針對電腦視覺進行訓練。AWS Panorama使用 SageMaker Neo 優化模型上運行AWS Panorama設備。

- Simple Storage Service (Amazon Simple Storage Service (Amazon S3))— 您可以使用 Amazon S3 存取點來暫存應用程式程式碼、模型和組態檔案，以便部署到AWS Panorama設備。
- AWS IoT–AWS Panorama使用AWS IoT監視狀態的服務AWS Panorama應用裝置、管理軟體更新和部署應用程式。您不需要使用AWS IoT直接。

開始使用AWS Panorama設備並進一步了解此服務，繼續AWS Panorama 入門。

AWS Panorama 入門

若要開始使用AWS Panorama，請先瞭解本指南中使用的服務概念和術語。然後，您可以使用AWS Panorama主控台註冊您的AWS Panorama設備並建立應用程式。大約一個小時後，您就可以設定裝置、更新其軟體，以及部署範例應用程式。若要完成本節中的自學課程，請使用AWS Panorama設備和透過區域網路串流視訊的攝影機。

> **Note**
>
> 若要購買設AWS Panorama備，請造訪主AWS Panorama控台。

示AWS Panorama例應用程序演示了AWS Panorama功能的使用。其中包含經過訓練的模型，以 SageMaker 及使用AWS Panorama應用程式 SDK 執行推論和輸出視訊的範例程式碼。範例應用程式包含AWS CloudFormation範本和指令碼，顯示如何透過命令列自動化開發和部署工作流程。

本章最後兩個主題詳細說明型號和相機的需求，以及AWS Panorama設備的硬體規格。如果您尚未取得設備和相機，或計劃開發自己的電腦視覺模型，請先參閱這些主題以取得詳細資訊。

主題

- AWS 全景概念
- 設定 AWS Panorama 設備
- 部署 AWS Panorama 範例應用程式
- 開發 AWS Panorama 應用程式
- 支援的電腦視覺模型和相機
- AWS Panorama 設備規格
- Service Quotas

AWS 全景概念

在 AWS Panorama 中，您可以建立電腦視覺應用程式，並將其部署到 AWS Panorama 設備或相容裝置，以分析來自網路攝影機的影片串流。您可以使用 Python 撰寫應用程式程式碼，並使用 Docker 建置應用程式容 您可以使用 AWS 全景應用程式 CLI 在本機匯入機器學習模型，或從亞馬遜簡單儲存服務 (Amazon S3) 匯入機器學習模型。應用程式使用 AWS Panorama 應用程式開發套件接收來自攝影機的視訊輸入，並與模型互動。

概念

- AWS 全景設備
- 相容裝置
- 應用程式
- 節點
- 模型

AWS 全景設備

AWS 全景設備是執行應用程式的硬體。您可以使用 AWS Panorama 主控台註冊設備、更新其軟體，以及在其中部署應用程式。AWS Panorama 設備上的軟體會連線到攝影機串流、將影片框傳送到您的應用程式，並在連接的顯示器上顯示影片輸出。

AWS 全景設備是一個邊緣裝置 搭載 NVIDIA 傑特森 AGX 澤維爾。而不是將圖像發送到AWS雲進行處理，它在優化的硬件本地運行應用程序。這使您可以實時分析視頻並在本地處理結果。設備需要網際網路連線才能報告其狀態、上傳記錄以及執行軟體更新和部署。

如需詳細資訊，請參閱管理AWS Panorama設備。

相容裝置

除了 AWS 全景設備之外，AWS 全景還支援以下裝置的相容裝置：AWS合作夥伴。相容裝置支援與 AWS 全景設備相同的功能。您可以使用 AWS Panorama 主控台和 API 註冊和管理相容的裝置，並以相同的方式建置和部署應用程式。

- Lenovo ThinkEdge® 第 70 集— 技術由 Nvidia 傑特森·澤維爾 NX

本指南中的內容和範例應用程式是使用 AWS 全景設備開發的。如需有關裝置特定硬體和軟體功能的詳細資訊，請參閱製造商的說明文件。

應用程式

應用程式可在 AWS Panorama 設備上執行，以便在影片串流上執行電腦視覺任務。您可以結合 Python 程式碼和機器學習模型來建置電腦視覺應用程式，並透過網際網路將它們部署到 AWS Panorama 設備。應用程式可以將影片傳送到顯示器，或使用 AWS 開發套件將結果傳送到 AWS 服務。

若要建置和部署應用程式，請使用 AWS 全景應用程式 CLI。AWS Panorama 應用程式 CLI 是一種命令列工具，可產生預設應用程式資料夾和組態檔、使用 Docker 建置容器，以及上傳資產。您可以在一台設備上運行多個應用程序。

如需詳細資訊，請參閱管理AWS Panorama應用。

節點

應用程序包括稱為多個組件節點，代表輸入、輸出、模型和程式碼。節點可以是僅設定 (輸入和輸出)，或包含成品 (模型和程式碼)。應用程序的代碼節點捆綁在節點套件您可以上傳到 Amazon S3 存取點，AWS 全景設備可在其中存取這些存取點。一個應用清單是定義節點之間連接的配置文件。

如需詳細資訊，請參閱應用程序節點。

模型

電腦視覺模型是經過訓練來處理影像的機器學習網路。電腦視覺模型可以執行各種工作，例如分類、偵測、分割和追蹤。電腦視覺模型會將影像做為輸入，並輸出影像中有關影像或物件的資訊。

AWS 全景支援使用建置的模型 PyTorch, 阿帕奇 TensorFlow。您可以使用亞馬遜建立模型 SageMaker 或在您的開發環境中。如需詳細資訊，請參閱 ???。

設定 AWS Panorama 設備

若要開始使用 AWS Panorama 設備或相容裝置，請在 AWS Panorama 主控台註冊並更新其軟體。在設定過程中，您可以在 AWS Panorama 中建立代表實體設備的設備資源，然後使用 USB 磁碟機將檔案複製到設備。設備使用這些憑證和組態檔連線到 AWS Panorama 服務。然後，您可以使用 AWS Panorama 主控台更新設備的軟體並註冊攝影機。

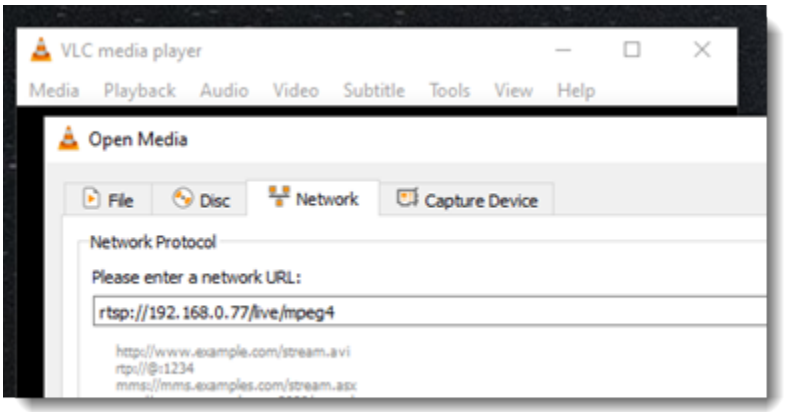
章節

先決條件

若要遵循此教學課程，您必須於此教學 AWS Panorama 必須於此教學課程或相容裝置和下列硬體

- 顯示 — 具有 HDMI 輸入的顯示器，用於檢視範例應用程式輸出。
- USB 磁碟機 (隨附於 AWS Panorama 設備) — 具有至少 1 GB 儲存空間的 FAT32 格式 USB 3.0 快閃記憶體磁碟機，用於將包含組態檔和憑證的存檔傳輸到 AWS Panorama 設備。
- 攝影機 — 輸出 RTSP 視訊串流的 IP 攝影機。

使用攝影機製造商提供的工具和說明來識別攝影機的 IP 位址和串流路徑。您可以使用視頻播放器（例如 VLC）來驗證流 URL，方法是將其作為網絡媒體源打開：



AWS Panorama 主控台使用其他 AWS 服務來組合應用程式元件、管理許可和驗證設定。若要註冊應用裝置並部署範例應用程式，您需要以下權限：

- AWSPanoramaFullAccess— 提供對 AWS Panorama、Amazon S3 中的 AWS Panorama 存取點、中的設備登入資料以及亞馬遜中AWS Secrets Manager的設備日誌的完整存取權CloudWatch。包括為 AWS Panorama 建立服務連結角色的權限。
- AWS Identity and Access Management(IAM) — 在第一次執行時，建立 AWS Panorama 服務和 AWS Panorama 設備使用的角色。

如果您沒有在 IAM 中建立角色的權限，請讓管理員開啟 AWS Panorama 主控台並接受建立服務角色的提示。

註冊和設定 AWS Panorama 設備

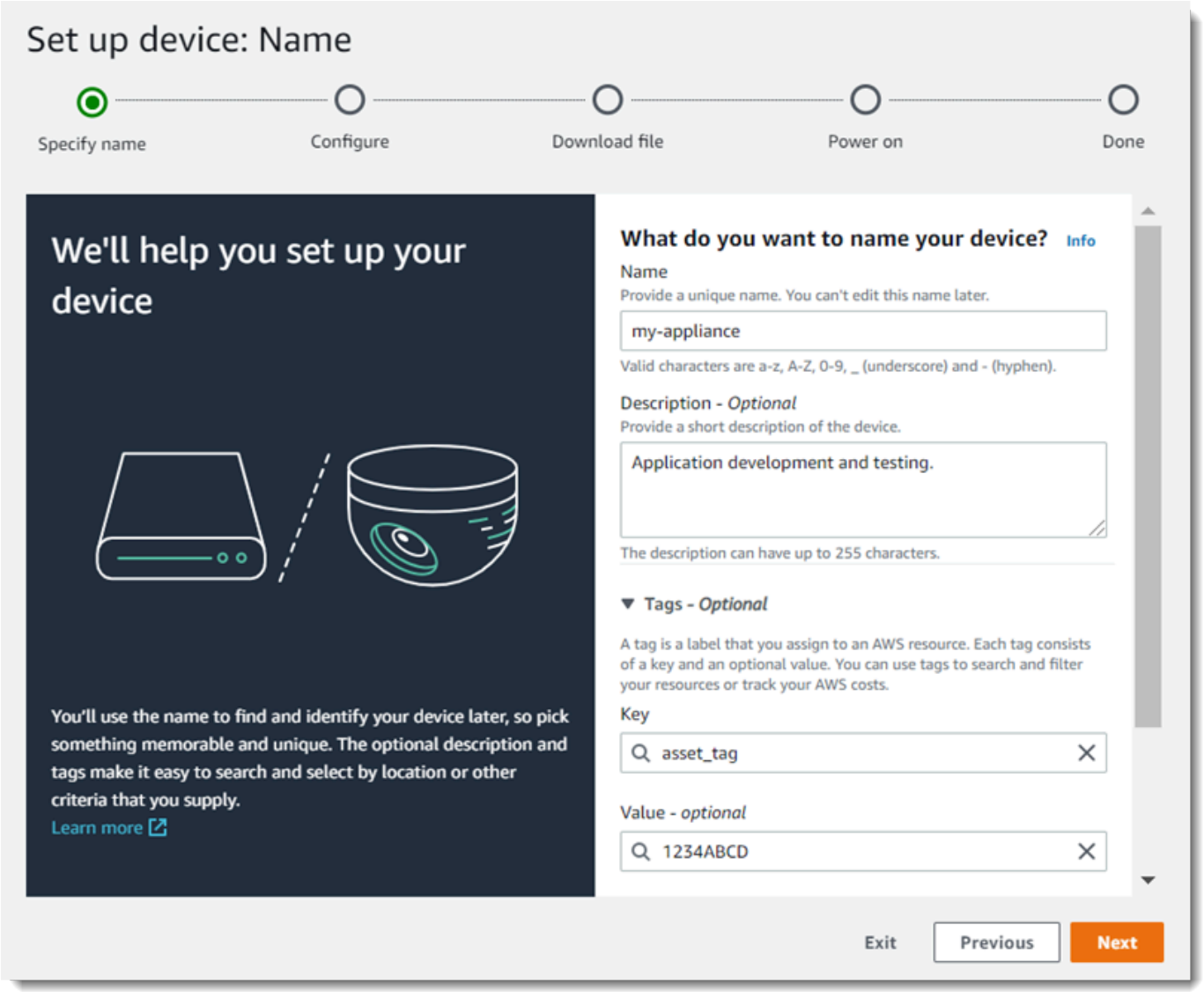
AWS Panorama 設備是一種硬體裝置，可透過本機網路連線連線連接至具有網路功能的攝影機。它使用以 Linux 為基礎的作業系統，其中包括 AWS Panorama 應用程式開發套件，以及執行電腦視覺應用程式的支援軟體。

若要連線到以AWS進行設備管理和應用程式部署，應用裝置使用裝置憑證。您可以使用 AWS Panorama 主控台產生佈建憑證。設備使用此臨時憑證完成初始設定並下載永久性裝置憑證。

> **Important**
>
> 您在此程序中產生的佈建憑證僅在 5 分鐘內有效。如果您未在此時間範圍內完成註冊程序，則必須重新開始。

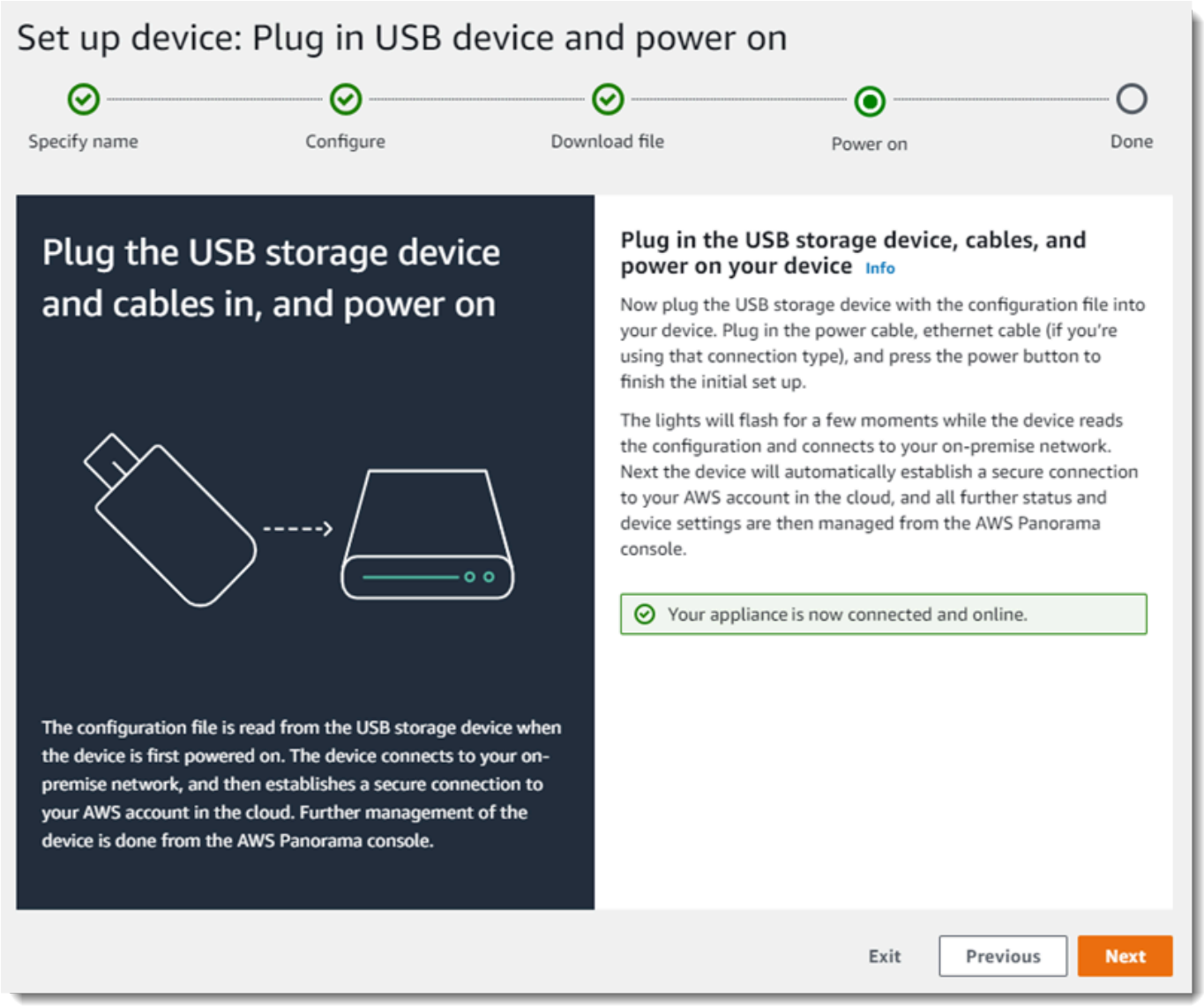
註冊設備

1. 將 USB 驅動器 Connect 到計算機。透過連接網路和電源線來準備設備。設備開啟電源並等待 USB 磁碟機連接。
2. 開啟 AWS Panorama 主控台 [開始使用] 頁面。
3. 選擇 [新增裝置]。
4. 選擇 [開始設定]。
5. 輸入能夠在 AWS Panorama 中代表設備的裝置資源的名稱和描述。選擇 Next (下一步)



6. 如果您需要手動指派 IP 位址、NTP 伺服器或 DNS 設定，請選擇 [進階網路設定]。否則請選擇 Next (下一步)。/
7. 選擇 [下載封存]。選擇 下一步。
8. 將組態歸檔複製到 USB 磁碟機的根目錄。
9. 將 USB 磁碟機 Connect 至設備正面 HDMI 連接埠旁邊的 USB 3.0 連接埠。

   當您連接 USB 磁碟機時，設備會將組態歸檔和網路組態檔複製到自身並連線至AWS雲端。設備完成連接時，設備的狀態指示燈從綠色變為藍色，然後返回綠色。
10. 若要繼續，請選擇 Next (下一步)。



11. 選擇 Done (完成)。

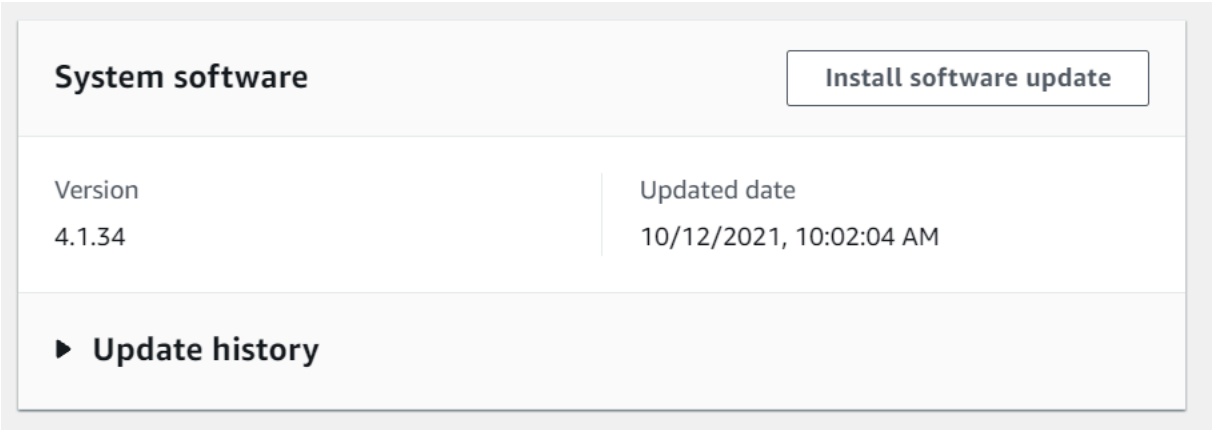
升級設備軟體

AWS Panorama 設備具有多個軟體元件，包括 Linux 作業系統、AWS Panorama 應用程式開發套件，以及支援電腦視覺程式庫和架構。為確保您可以在設備上使用最新功能和應用程式，請在安裝後以及在有更新時升級其軟體。

更新設備軟體

1. 開啟 AWS Panorama 主控台裝置頁面。
2. 選擇一個設備。

3. 選擇設定
4. 在系統軟體下，選擇安裝軟體更新。



5. 選擇新版本，然後選擇 [安裝]。

> **Important**
>
> 在繼續之前，請先從設備中移除 USB 磁碟機並進行格式化以刪除其內容。組態封存包含機密資料，不會自動刪除。

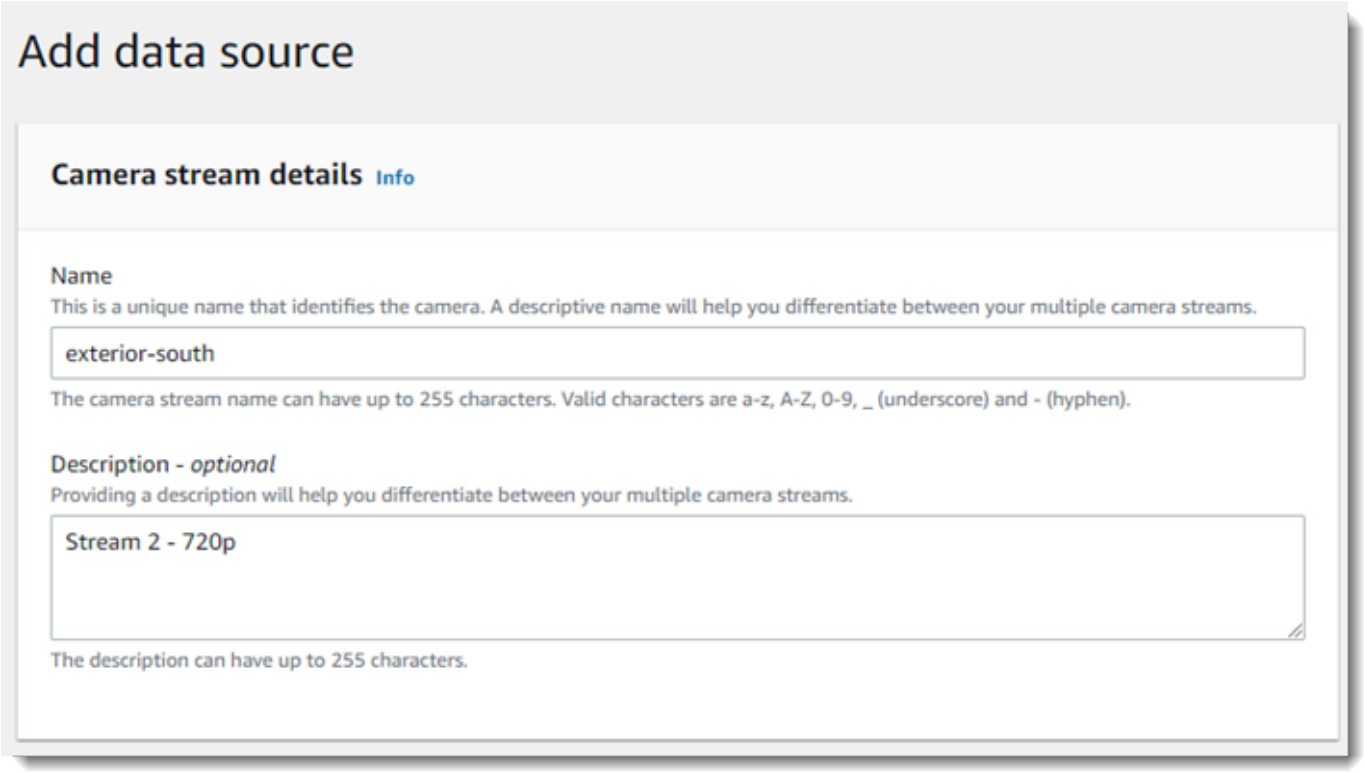
升級過程可能需要 30 分鐘或更長時間。您可以在 AWS Panorama 主控台或連線的監視器上監控其進度。程序完成後，應用裝置會重新啟動。

新增攝影機串流

接下來，使用 AWS Panorama 主控台註冊攝影機串流。

註冊攝影機串流

1. 開啟 AWS Panorama 主控台資料來源頁面。
2. 選擇 [新增資料來源]。



3. 進行下列設定。

   - 名稱 — 攝影機串流的名稱。
   - 描述 — 相機的簡短描述、其位置或其他詳細資料。
   - RTSP URL — 指定攝影機 IP 位址和串流路徑的 URL。例如：`rtsp://192.168.0.77/live/mpeg4/`
   - 認證 — 若攝影機串流受密碼保護，請指定使用者名稱和密碼。
4. 選擇 儲存 。

AWS Panorama 資料將您的相機登入資料安全地存放在AWS Secrets Manager。多個應用程序可以同時處理相同的攝像機流。

後續步驟

如果在安裝過程中遇到錯誤，請參閱疑難排解。

若要部署範例應用程式，請繼續下一個主題。

部署 AWS Panorama 範例應用程式

之後你已經設定 AWS Panorama 設備或相容裝置並升級了其軟件，部署示例應用程序。在以下各節中，您可以使用 AWS Panorama 應用程式 CLI 匯入範例應用程式，並使用 AWS Panorama 主控台進行部署。

範例應用程式使用機器學習模型，將來自網路攝影機的影片框架中的物件分類。它使用 AWS Panorama 應用程式開發套件載入模型、取得映像和執行模型。然後，應用程序將結果覆蓋在原始視頻的頂部，並將其輸出到連接的顯示器。

在零售環境中，分析人流量模式可讓您預測流量水平。透過將分析與其他資料結合在一起，您可以規劃假日和其他活動期間增加的人員配置需求、衡量廣告和促銷活動的有效性，或者最佳化展示位置和庫存管理。

章節

先決條件

為了遵循本指南的程序，您需要命令列終端機或 shell 來執行命令。在清單前，命令前會出現提示字元 (如有)。

```
~/panorama-project$ this is a command
this is output
```

對於長命令，我們使用轉義字符 (\)，將指令分割為多行。

在 Linux 和 macOS 上，使用您偏好的 shell 和套件軟體管理工具。在 Windows 10 上，您可以安裝適用於 Linux 的 Windows 子系統，以取得 Ubuntu 和 Bash 的 Windows 整合版本。如需在 Windows 中設定開發環境的說明，請參閱在 Windows 中設置開發環境。

您可以使用 Python 開發 AWS Panorama 應用程式，並使用 Python 套件管理員 pip 來安裝工具。如果您還沒有 Python，安裝最新版本。如果您有 Python 3 而不是 pip，請使用操作系統的軟件包管理器安裝 pip，或者安裝帶有 pip 的新版本的 Python。

在本教學課程中，您會使用 Docker 建置執行應用程式程式碼的容器。從碼頭網站安裝碼頭工人：取得 Docker

本教學使用 AWS Panorama 應用程式 CLI 匯入範例應用程式、建立套件和上傳成品。AWS Panorama 應用程式 CLI 使用AWS Command Line Interface(AWS CLI)，以呼叫服務 API 操作。如果您已經擁有 AWS CLI，將其升級到最新版本。安裝 AWS Panorama 應用程式 CLI 和AWS CLI，使用pip。

```
$ pip3 install --upgrade awscli panoramacli
```

下載範例應用程式，並將其解壓縮至您的工作區。

- 範例應用程式–aws-panorama-sample.zip

匯入範例應用程式

若要匯入範例應用程式以在您的帳戶中使用，請使用 AWS Panorama 應用程式 CLI。應用程式的資料夾和資訊清單包含預留位置帳號的參考。若要使用您的帳號更新這些資訊，請執行panorama-cli import-application指令。

```
aws-panorama-sample$ panorama-cli import-application
```

所以此SAMPLE_CODE套件，在packages目錄中，包含應用程序的代碼和配置，包括使用應用程序基礎映像的 Dockerfile，panorama-application。若要建置在應用裝置上執行的應用程式容器，請使用panorama-cli build-container指令。

```
aws-panorama-sample$ ACCOUNT_ID=$(aws sts get-caller-identity --output text --query
 'Account')
aws-panorama-sample$ panorama-cli build-container --container-asset-name code_asset --
package-path packages/${ACCOUNT_ID}-SAMPLE_CODE-1.0
```

AWS Panorama 應用程式 CLI 的最後一個步驟是註冊應用程式的程式碼和模型節點，然後將資產上傳到服務提供的 Amazon S3 存取點。這些資產包括程式碼的容器影像、模型，以及每個資產的描述器檔案。若要註冊節點並上傳資產，請執行panorama-cli package-application指令。

```
aws-panorama-sample$ panorama-cli package-application
Uploading package model
Registered model with patch version
 bc9c58bd6f83743f26aa347dc86bfc3dd2451b18f964a6de2cc4570cb6f891f9
Uploading package code
Registered code with patch version
 11fd7001cb31ea63df6aaed297d600a5ecf641a987044a0c273c78ceb3d5d806
```

部署應用程式

使用 AWS Panorama 主控台將應用程式部署到您的設備。

部署應用程式

1. 開啟 AWS Panorama 主控台部署應用程式頁面。
2. 選擇部署應用程式。
3. 粘貼應用程序清單的內容，`graphs/aws-panorama-sample/graph.json`，進入文字編輯器。選擇 Next (下一步)。
4. 在 Application name (應用程式名稱) 中，輸入 `aws-panorama-sample`。
5. 選擇繼續部署。
6. 選擇開始部署。
7. 選擇下一頁而不選擇角色。
8. 選擇選擇裝置，然後選擇您的設備。選擇 Next (下一步)。
9. 在「」選取資料來源步驟，選擇檢視輸入，並將您的攝影機串流新增為資料來源。選擇 Next (下一步)。
10. 在「」設定步驟，選擇下一頁。
11. 選擇部署，然後選擇已完成。
12. 在已部署的應用程式清單中，選擇aws-panorama-sample。

重新整理此頁面以取得更新，或使用下列指令碼從命令列監督部署。

Example monitor-deployment.sh

```
while true; do
  aws panorama list-application-instances --query 'ApplicationInstances[?Name==`aws-
panorama-sample`]'
```

```
  sleep 10
done
```

```
[
    {
        "Name": "aws-panorama-sample",
        "ApplicationInstanceId": "applicationInstance-x264exmpl33gq5pchc2ekoi6uu",
        "DefaultRuntimeContextDeviceName": "my-appliance",
        "Status": "DEPLOYMENT_PENDING",
        "HealthStatus": "NOT_AVAILABLE",
        "StatusDescription": "Deployment Workflow has been scheduled.",
        "CreatedTime": 1630010747.443,
        "Arn": "arn:aws:panorama:us-west-2:123456789012:applicationInstance/
applicationInstance-x264exmpl33gq5pchc2ekoi6uu",
        "Tags": {}
    }
]
[
    {
        "Name": "aws-panorama-sample",
        "ApplicationInstanceId": "applicationInstance-x264exmpl33gq5pchc2ekoi6uu",
        "DefaultRuntimeContextDeviceName": "my-appliance",
        "Status": "DEPLOYMENT_PENDING",
        "HealthStatus": "NOT_AVAILABLE",
        "StatusDescription": "Deployment Workflow has completed data validation.",
        "CreatedTime": 1630010747.443,
        "Arn": "arn:aws:panorama:us-west-2:123456789012:applicationInstance/
applicationInstance-x264exmpl33gq5pchc2ekoi6uu",
        "Tags": {}
    }
]
...
```

如果應用程式未開始執行，請檢查應用程式和裝置記錄檔在 Amazon CloudWatch 日誌。

檢視輸出

部署完成後，應用程式會開始處理視訊串流，並將記錄檔傳送至 CloudWatch。

若要檢視日誌 CloudWatch 日誌

1. 開啟的日誌群組頁面 CloudWatch 日誌主控台。

2. 在下列群組中尋找 AWS Panorama 應用程式和設備日誌：

   - 裝置記錄–/aws/panorama/devices/*device-id*
   - 應用程式記錄–/aws/panorama/devices/*device-id*/applications/*instance-id*

```
2022-08-26 17:43:39 INFO     INITIALIZING APPLICATION
2022-08-26 17:43:39 INFO     ## ENVIRONMENT VARIABLES
{'PATH': '/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin', 'TERM':
 'xterm', 'container': 'podman'...}
2022-08-26 17:43:39 INFO     Configuring parameters.
2022-08-26 17:43:39 INFO     Configuring AWS SDK for Python.
2022-08-26 17:43:39 INFO     Initialization complete.
2022-08-26 17:43:39 INFO     PROCESSING STREAMS
2022-08-26 17:46:19 INFO     epoch length: 160.183 s (0.936 FPS)
2022-08-26 17:46:19 INFO     avg inference time: 805.597 ms
2022-08-26 17:46:19 INFO     max inference time: 120023.984 ms
2022-08-26 17:46:19 INFO     avg frame processing time: 1065.129 ms
2022-08-26 17:46:19 INFO     max frame processing time: 149813.972 ms
2022-08-26 17:46:29 INFO     epoch length: 10.562 s (14.202 FPS)
2022-08-26 17:46:29 INFO     avg inference time: 7.185 ms
2022-08-26 17:46:29 INFO     max inference time: 15.693 ms
2022-08-26 17:46:29 INFO     avg frame processing time: 66.561 ms
2022-08-26 17:46:29 INFO     max frame processing time: 123.774 ms
```

若要檢視應用程式的視訊輸出，請使用 HDMI 纜線將設備連接至螢幕。根據預設，應用程式會顯示任何具有超過 20% 信賴度的分類結果。

Example 壓縮類。JSON

```
["tench", "goldfish", "great white shark", "tiger shark",
"hammerhead", "electric ray", "stingray", "cock", "hen", "ostrich",
"brambling", "goldfinch", "house finch", "junco", "indigo bunting",
"robin", "bulbul", "jay", "magpie", "chickadee", "water ouzel",
"kite", "bald eagle", "vulture", "great grey owl",
"European fire salamander", "common newt", "eft",
"spotted salamander", "axolotl", "bullfrog", "tree frog",
...
```

樣本模型有 1000 個類別，包括許多動物，食物和常見物體。嘗試將相機指向鍵盤或咖啡杯。

為了簡化，範例應用程式使用輕量型分類模型。該模型輸出與其每個類的概率的單個數組。現實世界的應用程式更頻繁地使用具有多維輸出的物體偵測模型。如需具有更複雜模型的範例應用程式，請參閱範例應用程式、指令碼和範本。

啟用適用於 Python 的軟體開發套件

範例應用程式會使用AWS SDK for Python (Boto)將指標發送到亞馬遜 CloudWatch。若要啟用此功能，請建立一個角色，以授與應用程式傳送指標的權限，然後重新部署附加角色的應用程式。

範例應用程式包括AWS CloudFormation建立具有所需權限之角色的範本。若要建立角色，請使用`aws cloudformation deploy`指令。

```
$ aws cloudformation deploy --template-file aws-panorama-sample.yml --stack-name aws-panorama-sample-runtime --capabilities CAPABILITY_NAMED_IAM
```

若要重新部署應用程式

1. 開啟 AWS Panorama 主控台部署應用程式頁面。
2. 選擇應用程式。
3. 選擇 Replace (取代)。
4. 完成部署應用程式。在指定 IAM 角色」下，選擇您建立的角色。其名稱開頭為 `aws-panorama-sample-runtime`。
5. 部署完成時，開啟CloudWatch安慰，然後檢視`AWSPanoramaApplication`命名空間。每 150 個框架，應用程式就會記錄並上傳指標，以便進行框架處理和推論時間。

清除

如果您已完成使用範例應用程式，則可以使用 AWS Panorama 主控台將其從設備中移除。

從設備中移除應用程式

1. 開啟 AWS Panorama 主控台部署應用程式頁面。
2. 選擇應用程式。
3. 選擇從裝置刪除。

後續步驟

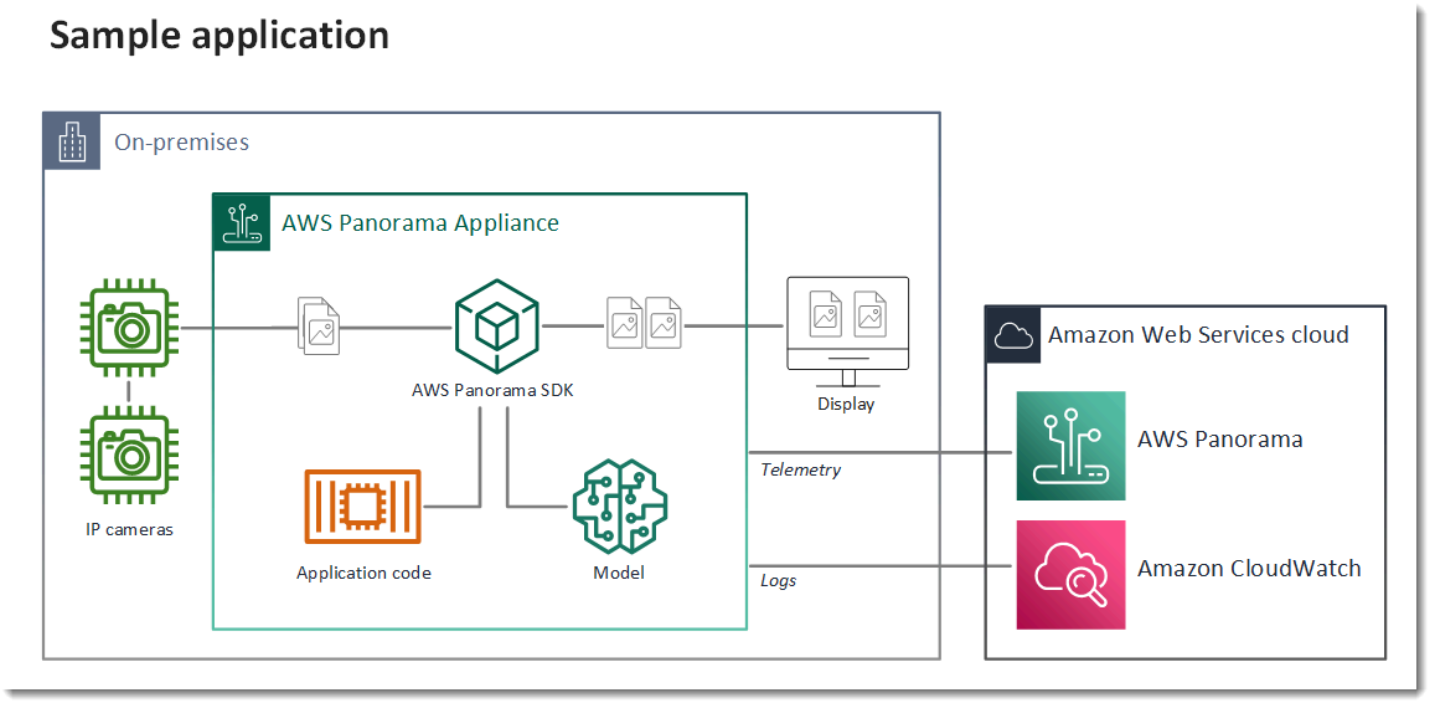
如果您在部署或執行範例應用程式時遇到錯誤，請參閱疑難排解。

若要進一步瞭解範例應用程式的功能和實作，請繼續：下一主題。

開發 AWS Panorama 應用程式

您可以使用範例應用程式來了解 AWS Panorama 應用程式結構，以及做為自有應用程式的起點。

下圖顯示在 AWS Panorama 設備上執行的應用程式的主要元件。應用程式程式碼使用 AWS Panorama 應用程式開發套件取得影像，並與模型互動，而模型無法直接存取。應用程式會將視訊輸出到連接的顯示器，但不會將影像資料傳送到區域網路之外。



在此範例中，應用程式使用 AWS Panorama 應用程式開發套件從攝影機取得影片框、預先處理影片資料，然後將資料傳送至可偵測物件的電腦視覺模型。應用程式會在連接至設備的 HDMI 顯示器上顯示結果。

章節

應用程式資訊清單

應用程序清單是一個名為的文件graph.json中的graphsfolder。清單定義了應用程序的組件，這是包，節點和邊緣。

套件是應用程式程式碼、模型、相機和顯示器的程式碼、組態和二進位檔案。範例應用程式使用 4 個套件：

Example **graphs/aws-panorama-sample/graph.json**— 套件

```
        "packages": [
            {
                "name": "123456789012::SAMPLE_CODE",
                "version": "1.0"
            },
            {
                "name": "123456789012::SQUEEZENET_PYTORCH_V1",
                "version": "1.0"
            },
            {
                "name": "panorama::abstract_rtsp_media_source",
                "version": "1.0"
            },
            {
                "name": "panorama::hdmi_data_sink",
                "version": "1.0"
            }
        ],
```

前兩個套件是在應用程式中定義的packages目錄。它們包含特定於此應用程序的代碼和模型。第二個套件是 AWS Panorama 服務所提供的一般攝影機和顯示套件。所以此abstract_rtsp_media_source套件是您在部署期間覆寫的相機的預留位置。所以此hdmi_data_sink封裝表示設備上的 HDMI 輸出連接器。

節點是套件的介面，以及非套件參數，這些參數可能具有您在部署時覆寫的預設值。代碼和模型包定義接口package.json指定輸入和輸出的檔案，可以是視訊串流或基本資料類型，例如浮點數、布林值或字串。

例如，code_node節點是指來自SAMPLE_CODE封裝。

```
        "nodes": [
            {
```

```
                "name": "code_node",
                "interface": "123456789012::SAMPLE_CODE.interface",
                "overridable": false,
                "launch": "onAppStart"
            },
```

此界面是在套件組態檔案中定義的，package.json。介面會指定套件為商務邏輯，而且需要名為的視訊串流video_in和一個名為的浮點數threshold作為輸入。該接口還指定代碼需要名為的視頻流緩衝區video_out將視訊輸出至顯示器

Example **packages/123456789012-SAMPLE_CODE-1.0/package.json**

```
{
    "nodePackage": {
        "envelopeVersion": "2021-01-01",
        "name": "SAMPLE_CODE",
        "version": "1.0",
        "description": "Computer vision application code.",
        "assets": [],
        "interfaces": [
            {
                "name": "interface",
                "category": "business_logic",
                "asset": "code_asset",
                "inputs": [
                    {
                        "name": "video_in",
                        "type": "media"
                    },
                    {
                        "name": "threshold",
                        "type": "float32"
                    }
                ],
                "outputs": [
                    {
                        "description": "Video stream output",
                        "name": "video_out",
                        "type": "media"
                    }
                ]
            }
        ]
```

```
    }
}
```

回到應用程序清單中，camera_nodenode 表示來自攝像機的視頻流。它包含在部署應用程式時出現在主控台中的裝飾器，提示您選擇攝影機串流。

Example **graphs/aws-panorama-sample/graph.json**— Camera 節點

```
            {
                "name": "camera_node",
                "interface": "panorama::abstract_rtsp_media_source.rtsp_v1_interface",
                "overridable": true,
                "launch": "onAppStart",
                "decorator": {
                    "title": "Camera",
                    "description": "Choose a camera stream."
                }
            },
```

參數節點，threshold_param，定義應用程式程式碼所使用的信賴度閾值參數。它的預設值為 60，並且可以在部署期間覆寫。

Example **graphs/aws-panorama-sample/graph.json**— 參數節點

```
            {
                "name": "threshold_param",
                "interface": "float32",
                "value": 60.0,
                "overridable": true,
                "decorator": {
                    "title": "Confidence threshold",
                    "description": "The minimum confidence for a classification to be
 recorded."
                }
            }
```

應用程序清單的最後一部分，edges，在節點之間建立連接。攝像機的視頻流和閾值參數連接到代碼節點的輸入，並且來自代碼節點的視頻輸出連接到顯示器。

Example **graphs/aws-panorama-sample/graph.json**— Edge

```
        "edges": [
```

```
        {
            "producer": "camera_node.video_out",
            "consumer": "code_node.video_in"
        },
        {
            "producer": "code_node.video_out",
            "consumer": "output_node.video_in"
        },
        {
            "producer": "threshold_param",
            "consumer": "code_node.threshold"
        }
    ]
```

使用範例應用程式建置

您可以將範例應用程式做為應用程式的起點。

每個套件的名稱在您的帳戶中必須是唯一的。如果您和您帳戶中的其他用戶都使用通用軟件包名稱，例如code或者model，您可能會在部署時收到錯誤的套件版本。將程式碼套件的名稱變更為代表應用程式的名稱。

若要重新命名程式碼套件

1. 重命名包文件夾：packages/123456789012-*SAMPLE_CODE*-1.0/。
2. 更新下列位置的套件名稱。

   - 應用程式清單–graphs/aws-panorama-sample/graph.json
   - Package 組態–packages/123456789012-SAMPLE_CODE-1.0/package.json
   - 建置指令碼–3-build-container.sh

更新應用程式碼

1. 修改應用程式程式碼packages/123456789012-SAMPLE_CODE-1.0/src/application.py。
2. 建置容器，執行3-build-container.sh。

```
aws-panorama-sample$ ./3-build-container.sh
TMPDIR=$(pwd) docker build -t code_asset packages/123456789012-SAMPLE_CODE-1.0
```

```
Sending build context to Docker daemon  61.44kB
Step 1/2 : FROM public.ecr.aws/panorama/panorama-application
 ---> 9b197f256b48
Step 2/2 : COPY src /panorama
 ---> 55c35755e9d2
Successfully built 55c35755e9d2
Successfully tagged code_asset:latest
docker export --output=code_asset.tar $(docker create code_asset:latest)
gzip -9 code_asset.tar
Updating an existing asset with the same name
{
    "name": "code_asset",
    "implementations": [
        {
            "type": "container",
            "assetUri":
 "98aaxmpl1c1ef64cde5ac13bd3be5394e5d17064beccee963b4095d83083c343.tar.gz",
            "descriptorUri":
 "1872xmpl129481ed053c52e66d6af8b030f9eb69b1168a29012f01c7034d7a8f.json"
        }
    ]
}
Container asset for the package has been succesfully built at  ~/aws-panorama-
sample-dev/
assets/98aaxmpl1c1ef64cde5ac13bd3be5394e5d17064beccee963b4095d83083c343.tar.gz
```

CLI 會自動刪除舊的容器資產`assets`文件夾並更新軟件包配置。

3. 若要上傳套件，請執行`4-package-application.py`。
4. 開啟 AWS Panorama 主控台建置應用程式頁面。
5. 選擇應用程式。
6. 選擇 Replace (取代)。
7. 完成應用程式部署的步驟。如有需要，您可以變更應用程式資訊清單、攝影機串流或參數。

變更電腦視覺模型

此範例應用程式包含電腦視覺模型。若要使用您自己的模型，請修改模型節點的組態，然後使用 AWS Panorama 應用程式 CLI 將其匯入為資產。

下列範例使用 MXNet 固態硬碟 ResNet您可以從本指南下載的 50 種型號 GitHub 回購：ssd_512_resnet50_v1_voc.tar.gz

若要變更範例應用程式的模型

1. 重命名包文件夾以匹配您的模型。例如，到packages/*123456789012*-*SSD_512_RESNET50_V1_VOC*-1.0/。
2. 更新下列位置的套件名稱。

   - 應用程式清單–graphs/aws-panorama-sample/graph.json
   - Package 組態–packages/*123456789012*-*SSD_512_RESNET50_V1_VOC*-1.0/package.json
3. 在包配置文件中（package.json。將變更assets值轉換為空白數組。

```
{
    "nodePackage": {
        "envelopeVersion": "2021-01-01",
        "name": "SSD_512_RESNET50_V1_VOC",
        "version": "1.0",
        "description": "Compact classification model",
        "assets": [],
```

4. 開啟套件描述器檔案 (descriptor.json。更新framework和shape值以符合您的模型。

```
{
    "mlModelDescriptor": {
        "envelopeVersion": "2021-01-01",
        "framework": "MXNET",
        "inputs": [
            {
                "name": "data",
                "shape": [ 1, 3, 512, 512 ]
            }
        ]
    }
}
```

   適用於的值形狀、1,3,512,512，指示模型作為輸入的影像數目 (1)、每個影像中的色版數 (3-紅、綠和藍)，以及影像的尺寸 (512 x 512)。數組的值和順序因模型而異。

5. 使用 AWS Panorama 應用程式 CLI 匯入模型。AWS Panorama 應用程式 CLI 會將模型和描述器檔案複製到assets具有唯一名稱的文件夾，並更新軟件包配置。

```
aws-panorama-sample$ panorama-cli add-raw-model --model-asset-name model-asset \
--model-local-path ssd_512_resnet50_v1_voc.tar.gz \
--descriptor-path packages/123456789012-SSD_512_RESNET50_V1_VOC-1.0/descriptor.json
 \
--packages-path packages/123456789012-SSD_512_RESNET50_V1_VOC-1.0
{
    "name": "model-asset",
    "implementations": [
        {
            "type": "model",
            "assetUri":
 "b1a1589afe449b346ff47375c284a1998c3e1522b418a7be8910414911784ce1.tar.gz",
            "descriptorUri":
 "a6a9508953f393f182f05f8beaa86b83325f4a535a5928580273e7fe26f79e78.json"
        }
    ]
}
```

6. 若要上傳模型，請執行panorama-cli package-application。

```
$ panorama-cli package-application
Uploading package SAMPLE_CODE
Patch Version 1844d5a59150d33f6054b04bac527a1771fd2365e05f990ccd8444a5ab775809
 already registered, ignoring upload
Uploading package SSD_512_RESNET50_V1_VOC
Patch version for the package
 244a63c74d01e082ad012ebf21e67eef5d81ce0de4d6ad1ae2b69d0bc498c8fd
upload: assets/
b1a1589afe449b346ff47375c284a1998c3e1522b418a7be8910414911784ce1.tar.gz to
 s3://arn:aws:s3:us-west-2:454554846382:accesspoint/panorama-123456789012-
wc66m5eishf4si4sz5jefhx
63a/123456789012/nodePackages/SSD_512_RESNET50_V1_VOC/binaries/
b1a1589afe449b346ff47375c284a1998c3e1522b418a7be8910414911784ce1.tar.gz
upload: assets/
a6a9508953f393f182f05f8beaa86b83325f4a535a5928580273e7fe26f79e78.json to
 s3://arn:aws:s3:us-west-2:454554846382:accesspoint/panorama-123456789012-
wc66m5eishf4si4sz5jefhx63
a/123456789012/nodePackages/SSD_512_RESNET50_V1_VOC/binaries/
a6a9508953f393f182f05f8beaa86b83325f4a535a5928580273e7fe26f79e78.json
{
    "ETag": "\"2381dabba34f4bc0100c478e67e9ab5e\"",
    "ServerSideEncryption": "AES256",
```

```
    "VersionId": "KbY5fpESdpYamjWZ0YyGqHo3.LQQWUC2"
}
Registered SSD_512_RESNET50_V1_VOC with patch version
 244a63c74d01e082ad012ebf21e67eef5d81ce0de4d6ad1ae2b69d0bc498c8fd
Uploading package SQUEEZENET_PYTORCH_V1
Patch Version 568138c430e0345061bb36f05a04a1458ac834cd6f93bf18fdacdffb62685530
 already registered, ignoring upload
```

7. 更新應用程式碼。大多數代碼都可以重複使用。模型回應的特定程式碼位於`process_results`方法。

```
    def process_results(self, inference_results, stream):
        """Processes output tensors from a computer vision model and annotates a
 video frame."""
        for class_tuple in inference_results:
            indexes = self.topk(class_tuple[0])
        for j in range(2):
            label = 'Class [%s], with probability %.3f.'%
 (self.classes[indexes[j]], class_tuple[0][indexes[j]])
            stream.add_label(label, 0.1, 0.25 + 0.1*j)
```

根據您的型號，您可能也需要更新`preprocess`方法。

預處理影像

在應用程式將影像傳送至模型之前，它會透過調整影像大小和標準化顏色資料來準備它進行推論。該應用程序使用的模型需要一個 224 x 224 像素圖像，具有三個顏色通道，以匹配其第一層中的輸入數。應用程式會將每個顏色值轉換為 0 到 1 之間的數字，減去該顏色的平均值，然後除以標準差來調整每個顏色值。最後，它結合了顏色通道並將其轉換為 NumPy 模型可以處理的陣列。

Example application.py— 預處理

```
    def preprocess(self, img, width):
        resized = cv2.resize(img, (width, width))
        mean = [0.485, 0.456, 0.406]
        std = [0.229, 0.224, 0.225]
        img = resized.astype(np.float32) / 255.
        img_a = img[:, :, 0]
        img_b = img[:, :, 1]
        img_c = img[:, :, 2]
        # Normalize data in each channel
```

```
        img_a = (img_a - mean[0]) / std[0]
        img_b = (img_b - mean[1]) / std[1]
        img_c = (img_c - mean[2]) / std[2]
        # Put the channels back together
        x1 = [[[], [], []]]
        x1[0][0] = img_a
        x1[0][1] = img_b
        x1[0][2] = img_c
        return np.asarray(x1)
```

此程序會在以 0 為中心的可預測範圍內提供模型值。它與訓練資料集中套用至影像的預先處理相符，這是一種標準方法，但可能會因模型而有所不同。

使用適用於 Python 的軟體開發套件上傳指標

範例應用程式使用適用於 Python 的開發套件將指標上傳到亞馬遜 CloudWatch。

Example application.py— 適用於 Python 的開發套件

```
    def process_streams(self):
        """Processes one frame of video from one or more video streams."""
        ...
            logger.info('epoch length: {:.3f} s ({:.3f} FPS)'.format(epoch_time,
epoch_fps))
            logger.info('avg inference time: {:.3f} ms'.format(avg_inference_time))
            logger.info('max inference time: {:.3f} ms'.format(max_inference_time))
            logger.info('avg frame processing time: {:.3f}
ms'.format(avg_frame_processing_time))
            logger.info('max frame processing time: {:.3f}
ms'.format(max_frame_processing_time))
            self.inference_time_ms = 0
            self.inference_time_max = 0
            self.frame_time_ms = 0
            self.frame_time_max = 0
            self.epoch_start = time.time()
            self.put_metric_data('AverageInferenceTime', avg_inference_time)
            self.put_metric_data('AverageFrameProcessingTime',
avg_frame_processing_time)

    def put_metric_data(self, metric_name, metric_value):
        """Sends a performance metric to CloudWatch."""
        namespace = 'AWSPanoramaApplication'
        dimension_name = 'Application Name'
```

```
        dimension_value = 'aws-panorama-sample'
        try:
            metric = self.cloudwatch.Metric(namespace, metric_name)
            metric.put_data(
                Namespace=namespace,
                MetricData=[{
                    'MetricName': metric_name,
                    'Value': metric_value,
                    'Unit': 'Milliseconds',
                    'Dimensions': [
                        {
                            'Name': dimension_name,
                            'Value': dimension_value
                        },
                        {
                            'Name': 'Device ID',
                            'Value': self.device_id
                        }
                    ]
                }]
            )
            logger.info("Put data for metric %s.%s", namespace, metric_name)
        except ClientError:
            logger.warning("Couldn't put data for metric %s.%s", namespace,
metric_name)
        except AttributeError:
            logger.warning("CloudWatch client is not available.")
```

它會從您在部署期間指派的執行階段角色取得權限。此角色是在中定義`aws-panorama-sample.yml` AWS CloudFormationTemplate Template

Example aws-panorama-sample. YML

```
Resources:
  runtimeRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Effect: Allow
            Principal:
```

```
            Service:
              - panorama.amazonaws.com
          Action:
            - sts:AssumeRole
      Policies:
        - PolicyName: cloudwatch-putmetrics
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action: 'cloudwatch:PutMetricData'
                Resource: '*'
      Path: /service-role/
```

示例應用程序使用 pip 安裝適用於 Python 的 SDK 和其他依賴項。當您建置應用程式容器時，`Dockerfile`運行命令以在基本映像附帶的基本圖像之上安裝庫。

Example Dockerfile

```
FROM public.ecr.aws/panorama/panorama-application
WORKDIR /panorama
COPY . .
RUN pip install --no-cache-dir --upgrade pip && \
    pip install --no-cache-dir -r requirements.txt
```

若要使用 AWS for WordPressAWSSDK 在您的應用程序代碼中，首先修改模板以為應用程序使用的所有 API 操作添加權限。更新AWS CloudFormation通過運行堆棧`1-create-role.sh`每次你做出改變。然後，將變更部署到您的應用程式程式碼。

對於修改或使用現有資源的動作，最佳做法是指定目標的名稱或模式，將此原則的範圍縮到最小。`Resource`在一個單獨的聲明中。如需每個服務支援的動作和資源的詳細資訊，請參閱動作、資源及條件索引鍵在服務授權參考中

後續步驟

如需使用 AWS Panorama 應用程式 CLI 建置應用程式並從頭開始建立套件的相關說明，請參閱 CLI 的讀我檔案。

- Gitthub.com /aw/aws-panorama-cli

如需更多範例程式碼和可在部署之前驗證應用程式程式碼的測試公用程式，請瀏覽 AWS Panorama 範例儲存庫。

- Gitthub.com /awTemplate Templateaws-panorama-samples

支援的電腦視覺模型和相機

AWS Panorama 支援使用 PyTorch Apache MXNet 和TensorFlow. 當您部署應用程式時，AWS Panorama 會在 SageMaker Neo 中編譯您的模型。只要您使用與 SageMaker Neo 相容的圖層，就可以在 Amazon SageMaker 或您的開發環境中建立模型。

為了處理影片並取得要傳送至模型的影像，AWS Panorama 設備會使用 RTSP 協定連線至 H.264 編碼的視訊串流。AWS Panorama 測試各種常見攝影機的相容性。

章節

支援的型號

當您建立 AWS Panorama 的應用程式時，您會提供應用程式用於電腦視覺的機器學習模型。您可以使用由模型架構、範例模型或您自行建置和訓練的模型所提供的預先建置和預先訓練的模型。

> **Note**
>
> 如需已透過 AWS Panorama 測試的預先建置模型清單，請參閱模型相容性。

當您部署應用程式時，AWS Panorama 會使用 SageMaker Neo 編譯器來編譯您的電腦視覺模型。SageMakerNeo 是一種編譯器，可將模型最佳化以在目標平台上有效執行，目標平台可以是 Amazon Elastic Compute Cloud (Amazon EC2) 中的執行個體，也可以是 AWS Panorama 設備之類的邊緣裝置。

AWS Panorama 支援的 PyTorch Apache MXNet 版本，而TensorFlow且由 SageMaker Neo 支援邊緣裝置。當您建立自己的模型時，您可以使用 SageMakerNeo 發行說明中列出的架構版本。在中 SageMaker，您可以使用內建的影像分類演算法。

如需在 AWS Panorama 中使用的詳細資訊，請參閱「」電腦視覺模型。

支援的相機

AWS Panorama 設備支援來自透過本機網路輸出 RTSP 的攝影機的 H.264 視訊串流。對於大於 200 萬像素的相機串流，設備將影像縮小至 1920x1080 像素或同等大小，以保留串流的外觀比例。

下列相機型號已經過與 AWS Panorama 設備的相容性測試：

- 軸 — M3057-PLVE、M3058-PLVE、P1448-LE、P3225-LV MK 二
- LaView— LV-PB3040W
- 慧豐科技 — IB9360-H
- 阿姆克雷斯特 — IP2M-841B
- 安普維茲 — IPC-B850W-S-3X, IPC-D250W-S
- WGCC — 圓頂 PoE 4 萬像素 ONVIF

如需設備的硬體規格，請參閱AWS Panorama 設備規格。

AWS Panorama 設備規格

AWS Panorama 設備具有以下硬件規格。針對其他相容的設備，請參閱製造商的文件。

元件	規格
處理器和 GPU	恩維迪亞·傑森 AGX 澤維爾具有 32 GB RAM
乙太網路	2 倍 1000 個底座-T（技嘉字節）
USB	1 個 U 盤 2.0 和 1 個 USB 3.0 型 A 型母頭
HDMI 輸出	2.0a
維度	7.75 英寸 x 9.6 英寸 x 1.6 英寸（七十毫米 x 243 毫米 x 40 毫米）
Weight	3.7 磅（1.7 千克）
進行供電源	50 赫茲交流電
進度	三引腳插座
防塵和液體保護	IP-62
EMI/EMC 法規遵從性	FCC 第 15 部分（美國）
Thermal touch	國際經濟委員會 -62368
運行温度	-20 攝氏度至 60 攝氏度
運行濕度	百分之零至 95% 的生殖器
儲存體温度	-20 攝氏度至 85 攝氏度
儲存體濕度	低温不受控制。90% 在高温下的 RH
冷卻	強制空氣熱抽排（風扇）
掛載選項	機架式或獨立式

元件	規格
進行電源線	6 英尺（1.8 米）
進度控制	按鈕
Reset	瞬時切換
狀態和網絡指示燈	可編程 3 色 RGB 指示燈

設備上存在 Wi-Fi、藍牙和 SD 卡存儲，但無法使用。

AWS Panorama 設備包括兩個用於安裝在服務器機架上的螺釘。您可以安裝兩個裝置 side-by-side 在一個 19 英寸的機架上。

Service Quotas

AWS Panorama 會對您在帳戶中建立的資源和部署的應用程式套用配額。如果您使用多個 AWS 全景AWS地區，配額會分別套用至每個區域。AWS 全景配額無法調整。

AWS Panorama 中的資源包括裝置、應用程式節點套件和應用程式執行個體。

- 设备— 每個區域最多 50 個已註冊的設備。
- 節點套件— 每個區域 50 個套件，每個套件最多 20 個版本。
- 應用實例— 每個設備最多 10 個應用程序。每個應用程序最多可以監控 8 個攝像機流。每個裝置每天最多可部署 200 部署。

當您使用 AWS 全景應用程式 CLI 時，AWS Command Line Interface，或AWS使用 AWS 全景服務的開發套件，配額適用於您發出的 API 呼叫數量。您每秒最多可以提出 5 個請求。建立或修改資源的 API 作業子集會套用每秒 1 個要求的額外限制。

如需配額的完整清單，請造訪服務配額主控台，或請參閱AWS 全景端點和配額在Amazon Web Services 一般參考。

AWS Panorama 許可

您可以使用AWS Identity and Access Management (IAM) 來管理對應和應用程式之類的AWS Panorama服務和應用程式和資源的存取。針對在您帳戶中使用的使用者AWS Panorama，您可在可套用至 IAM 角色的許可政策中管理許可。若要管理應用程式的權限，您可以建立角色並將其指派給應用程式。

若要為帳戶中的使用者管理許可，使用AWS Panorama提供的受管政策或者自己撰寫政策。您需要其他AWS服務的權限，才能取得應用程式和應用裝置記錄、檢視指標，以及指派角色給應用程式。

設AWS Panorama備也具有可授予許可以存取AWS服務和資源的角色。應用裝置的角色是服務用來代表您存取其他AWS Panorama服務的服務角色之一。

應用程式角色是您為應用程式建立的個別服務角色，可授與該應用程式搭配使用AWS服務的權限AWS SDK for Python (Boto)。若要建立應用程式角色，您需要管理權限或管理員的協助。

您可以透過一個動作可以影響的資源 (在某些狀況下，透過額外的條件) 來限制使用者的許可。例如，您可以為應用程式的 Amazon 資源名稱 (ARN) 指定模式，該應用程式會需要使用者將其使用者名稱包含在他們建立的應用程式名稱中。如需每個動作支援的資源和條件，請參閱服務授權參考AWS Panorama中的動作、資源和條件索引鍵。

如需詳細資訊，請參閱什麼是 IAM？《IAM 使用者指南》中的。

主題

- AWS Panorama 政策
- AWS Panorama 服務角色和跨服務資源
- 向應用程序授予權限

AWS Panorama 政策

若要授予帳戶中的使用者對 AWS Panorama 的存取權，請在AWS Identity and Access Management (IAM 政策)。將身分為 IAM 政策，用於與使用者相關聯的 IAM 政策。您可以授予另一個帳戶的使用者許可，讓他們可以擔任您帳戶中的角色並存取 AWS Panorama 資源。

AWS Panorama 提供受管政策，這些政策可授予對 AWS Panorama API 動作的存取權，在某些情況下會存取其他服務，用於開發和管理 AWS Panorama 資源。AWS Panorama 可視需要更新受管政策，來確保您的使用者可在新函數發行時進行存取。

- AWSPanoramaFullAccess— 提供對 AWS Panorama、Amazon S3 中的 AWS Panorama 存取點、中的設備登入資料以及亞馬遜中AWS Secrets Manager的設備日誌的完整存取權 CloudWatch。包括為 AWS Panorama 建立服務連結角色的權限。查看政策

該AWSPanoramaFullAccess政策允許您標記 AWS Panorama 資源，但是沒有 AWS Panorama 主控台使用的所有與標籤相關的許可。若要授予這些許可，請新增下列政策。

- ResourceGroupsandTagEditorFullAccess— 查看政策

該AWSPanoramaFullAccess政策不包括從 AWS Panorama 主控台購買裝置的許可。若要授予這些許可，請新增下列政策。

- ElementalAppliancesSoftwareFullAccess— 查看政策

受管政策會授予對 API 動作的許可，而不會限制使用者可以修改的資源。如需進行更精細的控制，您可以建立自己的政策，來限制使用者的許可範圍。使用完整存取原則作為原則的起點。

> **建立服務角色**
>
> 第一次使用 AWS Panorama 主控台時，您需要許可才能建立 AWS Panorama 設備使用的服務角色。服務角色會授與服務權限，以管理資源或與其他服務互動。在授予使用者存取權之前，請先建立此角色。

如需有關可用來限制 AWS Panorama 中使用者許可範圍的資源和條件的詳細資訊，請參閱服務授權參考中的 AWS Panorama 的動作、資源和條件金鑰。

AWS Panorama 服務角色和跨服務資源

AWS Panorama 使用其他 AWS 服務來管理 AWS Panorama 設備、存放資料和匯入應用程式資源。服務角色會授與服務權限，以管理資源或與其他服務互動。首次登入 AWS Panorama 主控台時，您會建立以下服務角色：

- AWSServiceRoleForAWSPanorama— 允許 AWS Panorama 管理 AWS IoT、AWS Secrets Manager 和 AWS Panorama 資源。

  受管政策：AWSPanoramaServiceLinkedRolePolicy
- AWSPanoramaApplianceServiceRole— 允許 AWS Panorama 設備將日誌上傳到 CloudWatch，並從 AWS Panorama 創建的 Amazon S3 接入點獲取對象。

  受管政策：AWSPanoramaApplianceServiceRolePolicy

若要檢視附加至每個角色的權限，請使用IAM 主控台。在可能的情況下，角色的許可僅限於符合 AWS Panorama 使用的命名模式的資源。例如，。`AWSServiceRoleForAWSPanorama`僅授予服務訪問權限AWS IoT具有的資源`panorama`以他們的名字。

章節

- 保護應用裝置角色
- 使用其他服務

保護應用裝置角色

AWS Panorama 設備使用`AWSPanoramaApplianceServiceRole`角色可存取您帳戶中的資源。設備擁有上傳日誌至的許可 CloudWatch 日誌，從中讀取攝像機流憑據AWS Secrets Manager，以及 AWS Simple Storage Service (Amazon S3) 存取點中 AWS Panorama imple Storage Service (Amazon S3) 存取點中的應用程式成品。

> **Note**
>
> 應用程式不使用設備的權限。授予您的應用程式使用權限AWS服務，建立應用程式角色。

AWS Panorama 對您帳戶中的所有設備使用相同的服務角色，且不會跨帳戶使用角色。為了增加安全層，您可以修改應用裝置角色的信任原則以明確強制執行此操作，這是當您使用角色授與服務權限以存取帳戶中資源時的最佳做法。

更新應用裝置角色信任原則

1. 在 IAM 主控台中開啟設備角色：AWSPanoramaApplianceServiceRole
2. 選擇 Edit trust relationship (編輯信任關係)。
3. 更新策略內容，然後選擇更新信任政策。

以下信任政策包括一個條件，可確保當 AWS Panorama 擔任設備角色時，它正在為您帳戶中的設備執行此操作。所以此`aws:SourceAccount`條件會將 AWS Panorama 指定的帳戶 ID 與您包含在政策中的帳戶 ID 進行比較。

Example 信任策略 — 特定帳戶

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "panorama.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        }
      }
    }
  ]
}
```

如果您想要進一步限制 AWS Panorama，並允許其僅擔任特定裝置的角色，可以透過 ARN 指定裝置。所以此`aws:SourceArn`條件會將 AWS Panorama 指定的設備的 ARN 與您包含在政策中的設備進行比較。

Example 信任原則 — 單一應用裝置

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "panorama.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:panorama:us-east-1:123456789012:device/
device-lk7exmplpvcr3heqwjmesw76ky"
        },
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        }
      }
    }
  ]
}
```

如果重設並重新佈建應用裝置，則必須暫時移除來源 ARN 條件，然後使用新裝置 ID 再次新增該條件。

如需上述條件的資訊，以及當服務使用角色以存取您帳戶中的資訊，請參閱混淆代理人問題《IAM 使用者指南》中的資訊。

使用其他服務

AWS Panorama 在下列服務中建立或存取資源：

- AWS IoT— AWS Panorama 設備的事物、政策、憑證和任務
- Simple Storage Service (Amazon Simple Storage Service (Amazon S3))— 暫存應用程式模型、程式碼和組態的存取點。
- Secrets Manager— AWS Panorama 設備的短期登入資料。

如需每個服務的 Amazon 資源名稱 (ARN) 格式或權限範圍的相關資訊，請參閱中的主題IAM User Guide在此列表中鏈接到。

向應用程序授予權限

您可以為應用程序創建一個角色，以授予其調用AWS服務。在缺省情況下，應用程序沒有任何許可。您可以在 IAM 中創建應用程序角色，並在部署期間將其分配給應用程序。要僅授予應用程序所需的權限，請為其創建一個具有特定 API 操作權限的角色。

所以此範例應用程式包含AWS CloudFormation模板和創建應用程序角色的腳本。這是一個服務角色AWS Panorama 可以假設的。此角色授予應用程序調用 CloudWatch 以上傳指標的權限。

Example As-全景-樣本 .yml— 應用程序角色

```
Resources:
  runtimeRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Effect: Allow
            Principal:
              Service:
                - panorama.amazonaws.com
            Action:
              - sts:AssumeRole
      Policies:
        - PolicyName: cloudwatch-putmetrics
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action: 'cloudwatch:PutMetricData'
                Resource: '*'
      Path: /service-role/
```

您可以擴展此腳本以授予對其他服務的權限，方法是指定 API 操作或模式的`Action`。

如需 AWS Panorama 許可的詳細資訊，請參。AWS Panorama 許可。

管理AWS Panorama設備

所以此AWS Panorama設備是運行應用程序的硬件。您可以使用AWS Panorama控制台註冊設備、更新其軟件並向其部署應用程序。上的軟件AWS Panorama設備連接到攝像機流，將視頻幀發送到您的應用程序，並在連接的顯示器上顯示視頻輸出。

設置您的設備或其他相容設備，您可以註冊相機以便與應用程序一起使用。您管理相機串流中的AWS Panorama主控台。部署應用程序時，您可以選擇設備發送給應用程序進行處理的攝像機流。

對於介紹AWS Panorama應用程序示例的設備，請參閱AWS Panorama 入門。

主題

管理 AWS Panorama 設備

您可以使用 AWS Panorama 主控台來設定、升級或取消註冊 AWS Panorama 設備和其他相容裝置。

若要設定設備，請遵循入門教學課程中的指示進行操作。安裝程序會在 AWS Panorama 中建立資源，以追蹤您的設備並協調更新和部署。

若要使用 AWS Panorama API 註冊設備，請參閱自動化設備註冊。

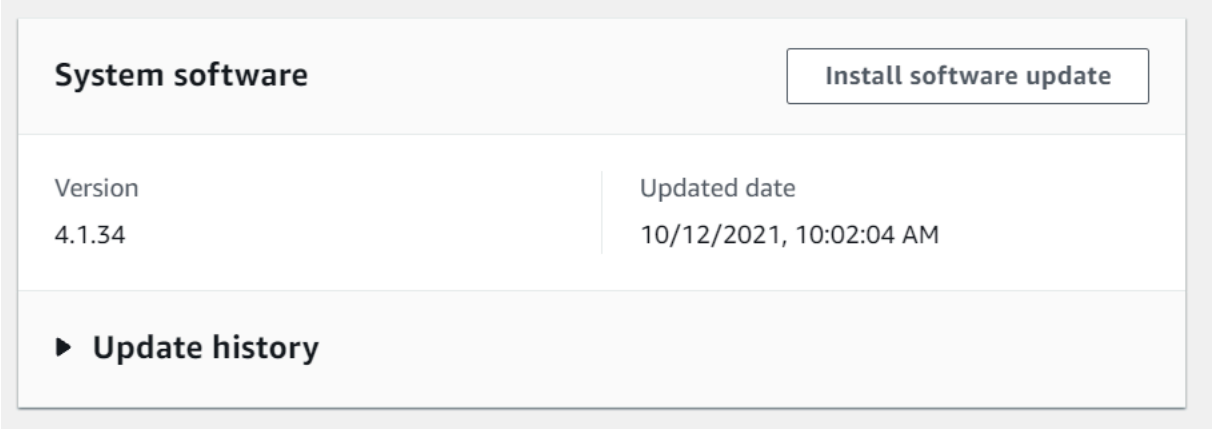
章節

更新設備軟體

您可以在 AWS Panorama 主控台中檢視和部署設備的軟體更新。更新可以是必要項目或選用項目。當必要的更新可用時，主控台會提示您套用它。您可以在應用裝置設定頁面上套用選用更新。

更新設備軟體

1. 開啟 AWS Panorama 主控台裝置頁面。
2. 選擇一個設備。
3. 選擇設定
4. 在系統軟體下，選擇安裝軟體更新。



5. 選擇新版本，然後選擇 [安裝]。

取消註冊設備

如果您已完成使用設備，則可以使用 AWS Panorama 主控台取消註冊並刪除相關AWS IoT資源。

刪除設備

1. 開啟 AWS Panorama 主控台裝置頁面。
2. 選擇設備的名稱。
3. 選擇 Delete (刪除)。
4. 輸入應用裝置的名稱，然後選擇刪除。

當您從 AWS Panorama 服務刪除設備時，不會自動刪除該設備上的資料。已取消註冊的應用裝置無法連線至AWS服務，且在重設之前無法再次註冊。

重新開機設備

您可以遠端將設備重新開機。

重新開機設備

1. 開啟 AWS Panorama 主控台裝置頁面。
2. 選擇設備的名稱。
3. 選擇 Reboot (重新啟動)。

主控台會傳送訊息給應用裝置以重新開機。要接收訊號，設備必須能夠連線至AWS IoT。若要使用 AWS Panorama API 重新啟動設備，請參閱重新開機。

重設設備

若要使用不同的區域或不同的帳戶，您必須重設並使用新的憑證重新佈建設備。重設裝置會套用最新的所需軟體版本，並刪除所有帳戶資料。

若要開始重設作業，必須插入設備並關閉電源。同時按住電源和重置按鈕五秒鐘。當您放開按鈕時，狀態指示燈會閃爍橘色。請等待狀態指示燈閃爍綠燈，然後再佈建或中斷應用裝置的連線。

您也可以重設設備軟體，而不刪除裝置中的憑證。如需詳細資訊，請參閱電源和重置按鈕。

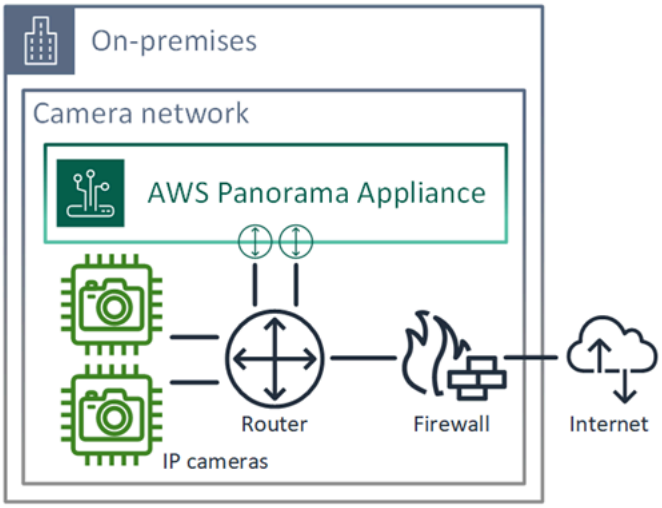
將 AWS Panorama 設備連接到您的網路

AWS Panorama 設備需要連線到AWS雲端和現場部署 IP 攝影機網路。您可以將設備連接到授予兩者存取權的單一防火牆，或將裝置的兩個網路介面中的每個網路介面連線到不同的子網路。在任何一種情況下，您都必須保護設備的網路連線，以防止未經授權存取您的攝影機串流。

章節

單一網路組態

設備具有兩個乙太網路連接埠。如果您透過單一路由器來路由裝置的所有流量，則可以在第一個連接埠的實體連線中斷時，使用第二個連接埠進行備援。將路由器設定為允許設備僅連接至攝影機串流和網際網路，並阻止攝影機串流以其他方式離開您的內部網路。

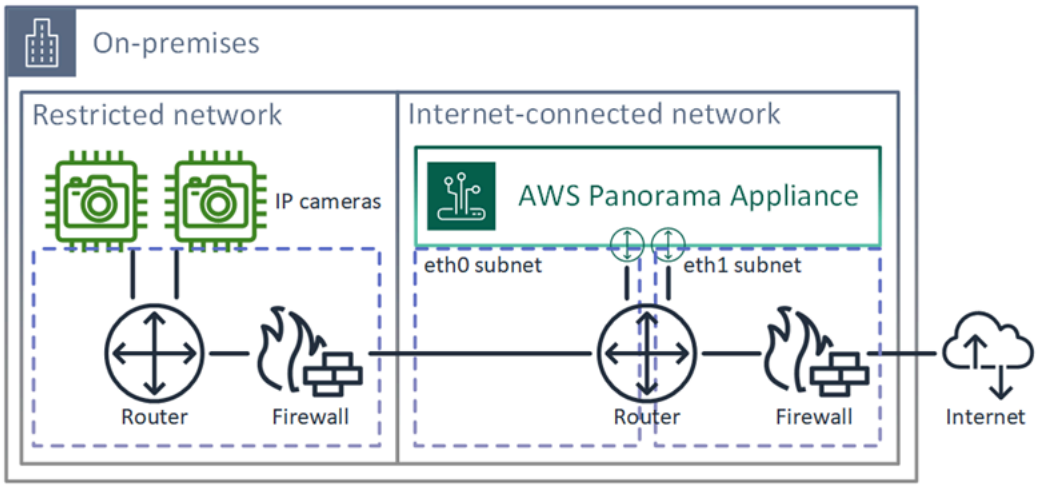


如需應用裝置需要存取的連接埠和端點的詳細資訊，請參閱設定服務存取和設定本機網路存取。

雙網路配置

為了獲得額外的安全性，您可以將設備放置在與攝影機網路分開的網路中。受限制的攝影機網路與設備網路之間的防火牆僅允許設備存取視訊串流。如果您的攝影機網路之前出於安全考量，您可能更喜歡使用此方法，而不是將攝影機網路連接到路由器，該路由器也可以存取網際網路。

以下範例顯示連線至每個連接埠上不同子網路的設備。路由器會將`eth0`介面放置在路由至攝影機網路的子網路`eth1`上，以及路由至網際網路的子網路上。



您可以在 AWS Panorama 主控台中確認每個連接埠的 IP 位址和 MAC 位址。

設定服務存取

在佈建期間，您可以將應用裝置設定為要求特定 IP 位址。提前選擇 IP 位址以簡化防火牆組態，並確保設備的位址在長時間離線時不會變更。

設備使用AWS服務來協調軟體更新和部署。將防火牆設定為允許應用裝置連線到這些端點。

網際網路存取

- AWS IoT(HTTPS 和 MQTT、連接埠 443、8443 和 8883) 以及裝置管理端點。AWS IoT Core如需詳細資訊，請參閱中的 AWS IoT Device Management 端點和配額Amazon Web Services 一般參考。
- AWS IoT認證 (HTTPS、連接埠 443) — `credentials.iot.<region>.amazonaws.com` 以及子網域。
- Amazon 彈性容器註冊表（HTTPS，端口 443）-`dkr.ecr.<region>.amazonaws.com` 和`api.ecr.<region>.amazonaws.com`子域。
- Amazon CloudWatch （HTTPS，端口 443）—`monitoring.<region>.amazonaws.com`.

- Amazon CloudWatch 日誌（HTTPS，端口 443）—`logs.<region>.amazonaws.com`.
- Amazon 簡單存儲服務（HTTPS，端口 443）-`s3-accesspoint.<region>.amazonaws.com` 和`s3.<region>.amazonaws.com`子域。

如果您的應用程式呼叫其他AWS服務，則應用裝置也需要存取這些服務的端點。如需詳細資訊，請參閱服務端點和配額。

設定本機網路存取

設備需要本機存取 RTSP 視訊串流，但不能透過網際網路存取。將防火牆設定為允許設備在內部存取連接埠 554 上的 RTSP 串流，並且不允許串流傳出或從網際網路傳入。

本機存取

- 實時流協議（RTSP，端口 554）-讀取攝像機流。
- 網路時間通訊協定 (NTP，連接埠 123) — 使設備的時鐘保持同步。如果您未在網路上執行 NTP 伺服器，則設備也可以透過網際網路連線至公用 NTP 伺服器。

私有連線

如果您將 AWS AWS Panorama 設備部署在具有 VPN 連線的私有 VPC 子網路中，則不需要網際網路存取。您可以使用 Site-to-Site VPN，或AWS Direct Connect在內部部署路由器和. AWS 在私有 VPC 子網路中，您可以建立可讓設備連接到 Amazon 簡單儲存服務和其他服務的端點。AWS IoT如需詳細資訊，請參閱 將應用裝置連接至私有子網路。

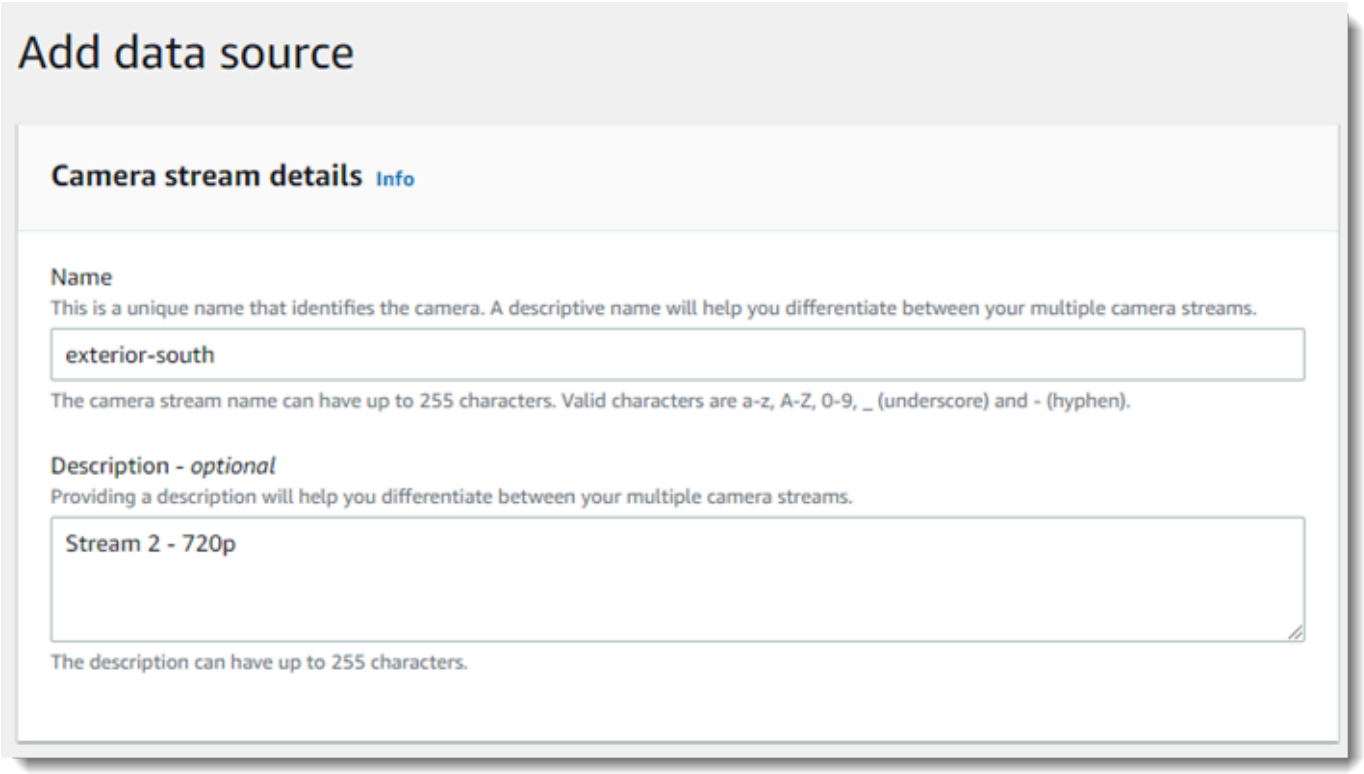
在 AWS Panorama 圖中管理相機流

要將視頻流註冊為應用程序的數據源，請使用 AWS Panorama 控制台。一個應用程序可以同時處理多個流，多個設備可以連接到同一個流。

> **Important**
>
> 應用程序可以連接到任何可從其連接的本地網絡路由的攝像機流。要保護您的視頻流，請將網絡配置為僅允許 RTSP 本地流量。如需詳細資訊，請參閱 AWS Panorama。

註冊攝像機流

1. 開啟 AWS Porama 主控台資料來源。
2. 選擇添加資料來源。



3. 進行下列設定。

   - 名稱— 攝影機流的名稱。
   - 描述— 照相機、其位置或其他詳細信息的簡短描述。

- RTSP 網址— 指定攝像機 IP 地址和流路徑的 URL。例如，`rtsp://192.168.0.77/live/mpeg4/`
- 登入資料— 如果攝影機流受到密碼保護，請指定使用者名稱和密碼。

4. 選擇 Save (儲存)。

要使用 AWS Panorama API 註冊相機流，請參閱自動化設備註冊。

有關與 AWS Panorama 設備兼容的攝像機列表，請參閱支援的電腦視覺模型和相機。

刪除流

您可以在 AWS Panorama 控制台中刪除相機流。

刪除攝像機流

1. 開啟 AWS Porama 主控台資料來源。
2. 選擇相機流。
3. 選擇刪除資料來源。

從服務中刪除攝像機流不會停止運行應用程序或從 Secrets Manager 中刪除攝像機憑據。要刪除密文，請使用Secrets Manager 主控台。

在 AWS Panorama 設備上管理應用程式

應用程序是代碼、模型和配置的組合。來自裝置頁面上，您可以管理設備上的應用程序。

在 AWS Panorama 設備上管理應用程式

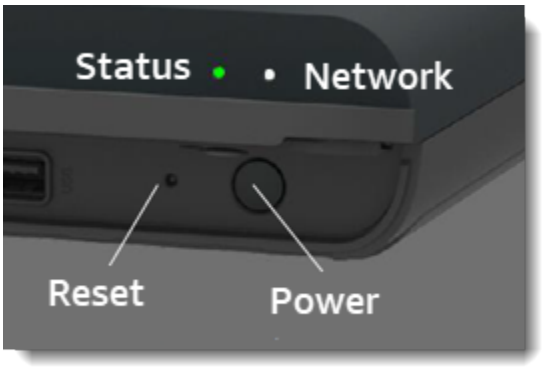
1. 打開 AWS Panorama 主控台「設備」頁面。
2. 選擇一個裝置。

所以此部署應用程式頁面顯示已部署到設備的應用程序。

使用此頁面上的選項可以從設備中刪除已部署的應用程序，或用新版本替換正在運行的應用程序。您還可以克隆應用程序（正在運行或已刪除）以部署該應用程序的新副本。

AWS Panorama 設備按鈕和指示燈

AWS Panorama 設備的電源按鈕上方有兩個 LED 指示燈，指示裝置狀態和網路連線。



Status 指示燈

LED 會變更顏色並閃爍以指示狀態。緩慢閃爍是每三秒鐘一次。快速閃爍是每秒一次。

狀態指示燈狀態

- 快速閃爍 Green— 設備正在啟動。
- 固體 Green— 設備運行正常。
- 慢速閃爍藍色— 設備正在複製設定檔並嘗試註冊AWS IoT。
- 快速閃爍藍色— 該設備是複製記錄影像到 USB 驅動器。
- 快速閃爍紅色— 設備在啟動期間遇到錯誤或過熱。
- 慢速閃爍橙色— 設備正在還原最新的軟體版本。
- 快速閃爍橘色— 設備正在還原最低軟體版本。

網路燈

網路 LED 為下列狀態：

網路 LED 狀態

- 固體 Green— 已連接乙太網路纜線。
- 閃爍 Green— 設備正在透過網路通訊。
- 固體紅色— 未連接乙太網路纜線。

電源和重置按鈕

電源和重設按鈕位於裝置正面的保護蓋下方。重置按鈕更小且凹陷。使用小螺絲刀或迴紋針將其按下。

重設設備

1. 必須插入設備並關閉電源。要關閉設備電源，請按住電源按鈕 1 秒鐘，然後等待關機順序完成。關機順序大約需要 10 秒鐘。
2. 要重設設備，請使用以下按鈕組合。短按是 1 秒。長按時間為 5 秒。對於需要多個按鈕的操作，請同時按住兩個按鈕。

   - 完全重設— 長按電源並重置。

     還原最低軟體版本，並刪除所有組態檔案和應用程式。
   - 還原最新的軟體版本— 短按重置.

     將最新的軟體更新重新套用至應用裝置。
   - 還原最低軟體版本— 長按復位.

     將所需的最新軟體更新重新套用至設備。
3. 釋放兩個按鈕。設備電源開啟，狀態指示燈閃爍橙色數分鐘。
4. 設備就緒後，狀態指示燈會閃爍綠燈。

重設設備不會將其從 AWS Panorama 服務中刪除。如需詳細資訊，請參閱取消註冊設備。

管理AWS Panorama應用

應用程式在AWS Panorama設備用於在視頻流上執行計算機視覺任務。您可以結合 Python 程式碼和機器學習模型來建置電腦視覺應用程式，並將其部署到AWS Panorama透過網際網路設備。應用程式可以將影片傳送到顯示器，或使用 AWS 開發套件將結果傳送到 AWS 服務。

主題

部署應用程式

若要部署應用程式，您可以使用 AWS Panorama 應用程式 CLI 將其匯入您的帳戶、建立容器、上傳和註冊資產，以及建立應用程式執行個體。本主題詳細介紹這些步驟中的每個步驟，並描述了背景中發生的情況。

如果您尚未部署應用程式，請參AWS Panorama 入門閱逐步解說。

如需自訂和延伸範例應用程式的詳細資訊，請參閱建置AWS Panorama應用。

章節

- 安裝 AWS 全景應用程式 CLI
- 匯入應用程式
- 建立容器映像檔
- 匯入模型
- 上傳應用程式資
- 使用 AWS 全景主控台部署應用程式
- 自動化應用部署

安裝 AWS 全景應用程式 CLI

若要安裝 AWS 全景應用程式 CLIAWS CLI，請使用 pip。

```
$ pip3 install --upgrade awscli panoramacli
```

若要使用 AWS 全景應用程式 CLI 建立應用程式映像，您需要 Docker。在 Linux 上，以qemu及相關的系統庫也是必需的。如需安裝和設定 AWS 全景應用程式 CLI 的詳細資訊，請參閱專案GitHub儲存庫中的讀我檔案。

- /aw/ aws-panorama-cli

如需使用 WSL2 在 Windows 中設定建置環境的指示，請參閱。在 Windows 中設置開發環境

匯入應用程式

如果您使用的是範例應用程式或第三方提供的應用程式，請使用 AWS Panorama 應用程式 CLI 匯入應用程式。

```
my-app$ panorama-cli import-application
```

此指令會使用您的帳戶 ID 重新命名應用程式套件。套件名稱以部署所在帳戶的帳號 ID 開頭。當您將應用程式部署到多個帳戶時，您必須為每個帳戶分別匯入和封裝應用程式。

例如，本指南的範例應用程式是程式碼套件和模型套件，每個套件都以預留位置帳戶 ID 命名。命`import-application`令會重新命名這些項目，以使用 CLI 從您的工作區認證推斷出的AWS帳戶 ID。

```
/aws-panorama-sample
### assets
### graphs
#   ### my-app
#       ### graph.json
### packages
    ### 123456789012-SAMPLE_CODE-1.0
    #   ### Dockerfile
    #   ### application.py
    #   ### descriptor.json
    #   ### package.json
    #   ### requirements.txt
    #   ### squeezenet_classes.json
    ### 123456789012-SQUEEZENET_PYTORCH-1.0
        ### descriptor.json
        ### package.json
```

`123456789012`在包目錄名稱和應用程序 manifest（`graph.json`）中引用它們的帳戶 ID 替換為您的帳戶 ID。您可以撥打電話`aws sts get-caller-identity`來確認您的帳戶 ID AWS CLI。

```
$ aws sts get-caller-identity
{
    "UserId": "AIDAXMPL7W66UC3GFXMPL",
    "Account": "210987654321",
    "Arn": "arn:aws:iam::210987654321:user/devenv"
}
```

建立容器映像檔

您的應用程式程式碼會封裝在 Docker 容器映像中，其中包含您在 Dockerfile 中安裝的應用程式程式碼和程式庫。使用 AWS 全景應用程式 CLI `build-container` 命令建立 Docker 映像檔並匯出檔案系統映像檔。

```
my-app$ panorama-cli build-container --container-asset-name code_asset --package-path
 packages/210987654321-SAMPLE_CODE-1.0
{
    "name": "code_asset",
    "implementations": [
        {
            "type": "container",
            "assetUri":
 "5fa5xmplbc8c16bf8182a5cb97d626767868d3f4d9958a4e49830e1551d227c5.tar.gz",
            "descriptorUri":
 "1872xmpl129481ed053c52e66d6af8b030f9eb69b1168a29012f01c7034d7a8f.json"
        }
    ]
}
Container asset for the package has been succesfully built at
 assets/5fa5xmplbc8c16bf8182a5cb97d626767868d3f4d9958a4e49830e1551d227c5.tar.gz
```

此指令會建立名為的 Docker 映像檔，`code_asset`並將檔案系統匯出至`assets`資料夾中的`.tar.gz`歸檔。CLI 從亞馬遜彈性容器註冊表（亞馬遜 ECR）中提取應用程序基礎映像，如應用程序的碼頭文件中的指定。

除了容器歸檔之外，CLI 還會為套件描述元 (`descriptor.json`) 建立資產。這兩個檔案都會以反映原始檔案雜湊的唯一識別碼來重新命名。AWS 全景應用程式 CLI 也會在套件組態中新增一個區塊，以記錄這兩個資產的名稱。在部署程序期間，應用裝置會使用這些名稱。

Example 套件 /123456789012-範例代碼 -1.0/封裝檔 — 含資產區塊

```
{
    "nodePackage": {
        "envelopeVersion": "2021-01-01",
        "name": "SAMPLE_CODE",
        "version": "1.0",
        "description": "Computer vision application code.",
        "assets": [
            {
                "name": "code_asset",
```

```
                "implementations": [
                    {
                        "type": "container",
                        "assetUri":
"5fa5xmplbc8c16bf8182a5cb97d626767868d3f4d9958a4e49830e1551d227c5.tar.gz",
                        "descriptorUri":
"1872xmpl129481ed053c52e66d6af8b030f9eb69b1168a29012f01c7034d7a8f.json"
                    }
                ]
            }
        ],
        "interfaces": [
            {
                "name": "interface",
                "category": "business_logic",
                "asset": "code_asset",
                "inputs": [
                    {
                        "name": "video_in",
                        "type": "media"
                    },
```

在build-container命令中指定的程式碼資產名稱必須與套件組態中的asset欄位值相符。在前面的例子中，兩個值都是code_asset。

匯入模型

您的應用程式可能在其 assets 資料夾中有模型歸檔，或者您分別下載。如果您有新模型、更新的模型或更新的模型描述元檔案，請使用add-raw-model指令匯入它。

```
my-app$ panorama-cli add-raw-model --model-asset-name model_asset \
      --model-local-path my-model.tar.gz \
      --descriptor-path packages/210987654321-SQUEEZENET_PYTORCH-1.0/descriptor.json \
      --packages-path packages/210987654321-SQUEEZENET_PYTORCH-1.0
```

如果您只需要更新描述符文件，則可以重複使用 assets 目錄中的現有模型。您可能需要更新描述器檔案，以配置諸如浮點精確度模式之類的功能。例如，下列指令碼顯示如何使用範例應用程式執行此操作。

Example 實用程序腳本/.sh update-model-config

```
#!/bin/bash
```

```
set -eo pipefail
MODEL_ASSET=fd1axmplacc3350a5c2673adacffab06af54c3f14da6fe4a8be24cac687a386e
MODEL_PACKAGE=SQUEEZENET_PYTORCH
ACCOUNT_ID=$(ls packages | grep -Eo '[0-9]{12}' | head -1)
panorama-cli add-raw-model --model-asset-name model_asset --model-local-path assets/
${MODEL_ASSET}.tar.gz --descriptor-path packages/${ACCOUNT_ID}-${MODEL_PACKAGE}-1.0/
descriptor.json --packages-path packages/${ACCOUNT_ID}-${MODEL_PACKAGE}-1.0
cp packages/${ACCOUNT_ID}-${MODEL_PACKAGE}-1.0/package.json packages/${ACCOUNT_ID}-
${MODEL_PACKAGE}-1.0/package.json.bup
```

在使用 CLI 重新匯入模型套件目錄中的描述元檔案之前，不會套用對該描述元檔案的變更。CLI 會使用新的資產名稱就地更新模型套件組態，類似於當您重建容器時更新應用程式程式碼套件的組態。

上傳應用程式資

要上傳和註冊應用程序的資產，其中包括模型歸檔，容器文件系統歸檔和它們的描述符文件，請使用package-application命令。

```
my-app$ panorama-cli package-application
Uploading package SQUEEZENET_PYTORCH
Patch version for the package
 5d3cxmplb7113faa1d130f97f619655d8ca12787c751851a0e155e50eb5e3e96
Deregistering previous patch version
 e845xmpl8ea0361eb345c313a8dded30294b3a46b486dc8e7c174ee7aab29362
Asset fd1axmplacc3350a5c2673adacffab06af54c3f14da6fe4a8be24cac687a386e.tar.gz already
 exists, ignoring upload
upload: assets/87fbxmpl6f18aeae4d1e3ff8bbc6147390feaf47d85b5da34f8374974ecc4aaf.json
 to s3://arn:aws:s3:us-east-2:212345678901:accesspoint/
panorama-210987654321-6k75xmpl2jypelgzst7uux62ye/210987654321/nodePackages/
SQUEEZENET_PYTORCH/
binaries/87fbxmpl6f18aeae4d1e3ff8bbc6147390feaf47d85b5da34f8374974ecc4aaf.json
Called register package version for SQUEEZENET_PYTORCH with patch version
 5d3cxmplb7113faa1d130f97f619655d8ca12787c751851a0e155e50eb5e3e96
...
```

如果資產檔案或套件組態沒有任何變更，CLI 會略過它。

```
Uploading package SAMPLE_CODE
Patch Version ca91xmplca526fe3f07821fb0c514f70ed0c444f34cb9bd3a20e153730b35d70 already
 registered, ignoring upload
Register patch version complete for SQUEEZENET_PYTORCH with patch version
 5d3cxmplb7113faa1d130f97f619655d8ca12787c751851a0e155e50eb5e3e96
```

```
Register patch version complete for SAMPLE_CODE with patch version
 ca91xmplca526fe3f07821fb0c514f70ed0c444f34cb9bd3a20e153730b35d70
All packages uploaded and registered successfully
```

CLI 會將每個套件的資產上傳到您帳戶專屬的 Amazon S3 存取點。AWS 全景會為您管理存取點，並透過 DescribePackageAPI 提供存取點的相關資訊。CLI 會將每個套件的資產上傳到為該套件提供的位置，並使用套件組態描述的設定向 AWS Panorama 服務註冊這些資產。

使用 AWS 全景主控台部署應用程式

您可以使用 AWS 全景主控台部署應用程式。在部署過程中，您可以選擇要傳遞給應用程式程式碼的攝影機串流，並設定應用程式開發人員提供的選項。

若要部署應用程式

1. 開啟 AWS 全景主控台已部署的應用程式頁面。
2. 選擇部署應用程式。
3. 將應用程式資訊清單的內容貼到文字編輯器中。`graph.json`選擇 下一步。
4. 輸入名稱和解除選項。
5. 選擇繼續進行部署。
6. 選擇 [開始部署]。
7. 如果您的應用程式使用角色，請從下拉式功能表中選擇角色。選擇 下一步。
8. 選擇 [選取裝置]，然後選擇您的設備。選擇 下一步。
9. 在 [選取資料來源] 步驟中，選擇 [檢視輸入]，然後將您的攝影機串流新增為資料來源。選擇 下一步。
10. 在 [設定] 步驟中，設定開發人員定義的任何應用程式特定設定。選擇 下一步。
11. 選擇 [部署]，然後選擇 [完成]。
12. 在已部署的應用程式清單中，選擇要監視其狀態的應用程式。

部署過程需要 15-20 分鐘。應用程式啟動時，設備的輸出可以長時間保持空白。如果遇到錯誤，請參閱疑難排解。

自動化應用部署

您可以使用 CreateApplicationInstanceAPI 自動化應用程式部署程序。該 API 需要兩個配置文件作為輸入。應用程序清單指定使用的包及其關係。第二個檔案是覆寫檔案，指定應用程式資訊清單中值的部署

時間覆寫。使用覆寫檔案可讓您使用相同的應用程式資訊清單，以不同的攝影機串流部署應用程式，以及設定其他應用程式特定的設定。

如需本主題中每個步驟的詳細資訊和範例指令碼，請參閱自動化應用部署。

在 AWS Panorama 主控台中管理應用程式

使用 AWS Panorama 主控台管理部署的應用程式。

章節

更新或複製應用程式

若要更新應用程式，請使用取代選項。取代應用程式時，您可以更新其程式碼或模型。

更新應用程式

1. 開啟 AWS Panorama 主控台部署應用程式頁面。
2. 選擇應用程式。
3. 選擇 Replace (取代)。
4. 請按照操作說明以建立新版本或應用程式。

還有一個複製選項，其行為類似取代，但不會移除舊版本的應用程式。您可以使用此選項在不停止執行中的版本的情況下測試應用程式的變更，或重新部署已刪除的版本。

刪除版本和應用程式

若要清除未使用的應用程式版本，請將其從設備中刪除。

欲刪除應用程式

1. 開啟 AWS Panorama 主控台部署應用程式頁面。
2. 選擇應用程式。
3. 選擇從裝置刪除。

封裝組態

使用 AWS Panorama 應用程序 CLI 命令時`panorama-cli package-application`，CLI 會將您的應用程序的資產上傳到 Amazon S3 並將其註冊到 AWS Panorama 圖。資產包括二進制文件（容器映像和模型）和描述符文件，AWS Panorama 設備會在部署期間下載這些文件。要註冊軟件包的資產，請提供一個單獨的包配置文件，用於定義軟件包、其資產及其接口。

以下示例顯示了具有一個輸入和一個輸出的代碼節點的軟件包配置。視頻輸入提供從攝像機流訪問圖像數據。輸出節點將處理過的圖像發送到顯示器。

Example 包裝 /1234567890-樣品代碼 -1.0/ 包裝。

```
{
    "nodePackage": {
        "envelopeVersion": "2021-01-01",
        "name": "SAMPLE_CODE",
        "version": "1.0",
        "description": "Computer vision application code.",
        "assets": [
            {
                "name": "code_asset",
                "implementations": [
                    {
                        "type": "container",
                        "assetUri":
 "3d9bxmplbdb67a3c9730abb19e48d78780b507f3340ec3871201903d8805328a.tar.gz",
                        "descriptorUri":
 "1872xmpl129481ed053c52e66d6af8b030f9eb69b1168a29012f01c7034d7a8f.json"
                    }
                ]
            }
        ],
        "interfaces": [
            {
                "name": "interface",
                "category": "business_logic",
                "asset": "code_asset",
                "inputs": [
                    {
                        "name": "video_in",
                        "type": "media"
                    }
                ],
```

```
                "outputs": [
                    {
                        "description": "Video stream output",
                        "name": "video_out",
                        "type": "media"
                    }
                ]
            }
        ]
    }
}
```

所以此assets部分指定 AWS Panorama 應用程序 CLI 上傳到 Amazon S3 的工件名稱。如果您從其他用户導入示例應用程序或應用程序，則此部分可能為空，也可以引用不在您的帳户中的資產。當你運行panorama-cli package-application中，AWS Panorama 應用程序 CLI 會使用正確的值填充本部分。

AWS Panorama 應用程序清單

部署應用程序時，將提供一個稱為應用程序清單的配置文件。此文件將應用程序定義為帶有節點和邊的圖形。應用程式清單是應用程式原始碼的一部分，儲存在graphs目錄。

Example 圖表aws-panorama-sample/圖形 .json

```
{
    "nodeGraph": {
        "envelopeVersion": "2021-01-01",
        "packages": [
            {
                "name": "123456789012::SAMPLE_CODE",
                "version": "1.0"
            },
            {
                "name": "123456789012::SQUEEZENET_PYTORCH_V1",
                "version": "1.0"
            },
            {
                "name": "panorama::abstract_rtsp_media_source",
                "version": "1.0"
            },
            {
                "name": "panorama::hdmi_data_sink",
                "version": "1.0"
            }
        ],
        "nodes": [
            {
                "name": "code_node",
                "interface": "123456789012::SAMPLE_CODE.interface"
            }
            {
                "name": "model_node",
                "interface": "123456789012::SQUEEZENET_PYTORCH_V1.interface"
            },
            {
                "name": "camera_node",
                "interface": "panorama::abstract_rtsp_media_source.rtsp_v1_interface",
                "overridable": true,
                "overrideMandatory": true,
                "decorator": {
```

```
                    "title": "IP camera",
                    "description": "Choose a camera stream."
                }
            },
            {
                "name": "output_node",
                "interface": "panorama::hdmi_data_sink.hdmi0"
            },
            {
                "name": "log_level",
                "interface": "string",
                "value": "INFO",
                "overridable": true,
                "decorator": {
                    "title": "Logging level",
                    "description": "DEBUG, INFO, WARNING, ERROR, or CRITICAL."
                }
            }
            ...
        ],
        "edges": [
            {
                "producer": "camera_node.video_out",
                "consumer": "code_node.video_in"
            },
            {
                "producer": "code_node.video_out",
                "consumer": "output_node.video_in"
            },
            {
                "producer": "log_level",
                "consumer": "code_node.log_level"
            }
        ]
    }
}
```

節點通過邊連接，邊指定節點輸入和輸出之間的映射。一個節點的輸出連接到另一個節點的輸入，形成一個圖形。

JSON 結構描述

應用程序清單和覆蓋文檔的格式是在 JSON 架構中定義的。您可以在部署前使用 JSON 架構驗證配置文檔。JSON 架構可在本指南的GitHub儲存庫。

- JSON 結構描述–aws-panorama-developer-指南/資源

應用程序節點

節點是模型、代碼、攝像機流、輸出和參數。節點有一個界面，用於定義其輸入和輸出。接口可以在您賬户中的軟件包、AWS Panorama 提供的軟件包或內置類型中定義。

在以下範例中，code_node和model_node請參閱示例應用程序附帶的示例代碼和模型包。camera_node使用 AWS Panorama 提供的軟件包為您在部署期間指定的攝像機流創建佔位符。

Example 圖形 .json — 節點

```
        "nodes": [
            {
                "name": "code_node",
                "interface": "123456789012::SAMPLE_CODE.interface"
            },
            {
                "name": "model_node",
                "interface": "123456789012::SQUEEZENET_PYTORCH_V1.interface"
            },
            {
                "name": "camera_node",
                "interface": "panorama::abstract_rtsp_media_source.rtsp_v1_interface",
                "overridable": true,
                "overrideMandatory": true,
                "decorator": {
                    "title": "IP camera",
                    "description": "Choose a camera stream."
                }
            }
    ]
```

Edges (邊)

邊緣會將一個節點的輸出映射到另一個節點的輸入。在以下示例中，第一條邊將從攝像機流節點的輸出映射到應用程序代碼節點的輸入。這些名字video_in和video_out是在節點包的界面中定義的。

Example 圖形 .json-邊

```
        "edges": [
            {
                "producer": "camera_node.video_out",
```

```
            "consumer": "code_node.video_in"
        },
        {
            "producer": "code_node.video_out",
            "consumer": "output_node.video_in"
        },
```

在您的應用程序代碼中，您可以使用inputs和outputs屬性從輸入串流獲取圖像，並將圖像發送到輸出串流。

Example application.py — 視頻輸入和輸出

```
    def process_streams(self):
        """Processes one frame of video from one or more video streams."""
        frame_start = time.time()
        self.frame_num += 1
        logger.debug(self.frame_num)
        # Loop through attached video streams
        streams = self.inputs.video_in.get()
        for stream in streams:
            self.process_media(stream)
        ...
        self.outputs.video_out.put(streams)
```

抽象節點

在應用程序清單中，抽象節點是指由 AWS Panorama 定義的包，您可以在應用程序清單中用作佔位符。AWS Panorama 提供兩種類型的抽象節點。

- 相機串流— 選擇應用程序在部署期間使用的攝像機流。

  套件名稱–panorama::abstract_rtsp_media_source

  介面名稱–rtsp_v1_interface
- HDMI 輸出— 表示應用程序輸出視頻。

  套件名稱–panorama::hdmi_data_sink

  介面名稱–hdmi0

以下示例顯示了處理相機流並將視頻輸出到顯示器的應用程序的一組基本軟件包、節點和邊緣。攝像機節點，它使用abstract_rtsp_media_source軟件包，可以接受多個攝像機流作為輸入。輸出節點，它引用hdmi_data_sink，允許應用程序代碼訪問從裝置的 HDMI 端口輸出的視頻緩衝區。

Example 圖形 .json — 抽象節點

```
{
    "nodeGraph": {
        "envelopeVersion": "2021-01-01",
        "packages": [
            {
                "name": "123456789012::SAMPLE_CODE",
                "version": "1.0"
            },
            {
                "name": "123456789012::SQUEEZENET_PYTORCH_V1",
                "version": "1.0"
            },
            {
                "name": "panorama::abstract_rtsp_media_source",
                "version": "1.0"
            },
            {
                "name": "panorama::hdmi_data_sink",
                "version": "1.0"
            }
        ],
        "nodes": [
            {
                "name": "camera_node",
                "interface": "panorama::abstract_rtsp_media_source.rtsp_v1_interface",
                "overridable": true,
                "decorator": {
                    "title": "IP camera",
                    "description": "Choose a camera stream."
                }
            },
            {
                "name": "output_node",
                "interface": "panorama::hdmi_data_sink.hdmi0"
            }
        ],
        "edges": [
```

```
            {
                "producer": "camera_node.video_out",
                "consumer": "code_node.video_in"
            },
            {
                "producer": "code_node.video_out",
                "consumer": "output_node.video_in"
            }
        ]
    }
}
```

應用程式參數

參數是具有基本類型的節點，可以在部署過程中被覆蓋。參數可以具有默認值和裝潢者，它指示應用程序的用户如何配置它。

參數類型

- string— 字串。例如： DEBUG。
- int32— 整數。例如 20
- float32— 浮點數。例如 47.5
- boolean–true或者false。

下面的示例顯示了兩個參數，一個字符串和一個數字，它們作為輸入發送到代碼節點。

Example 圖形 .json — 參數

```
       "nodes": [
           {
               "name": "detection_threshold",
               "interface": "float32",
               "value": 20.0,
               "overridable": true,
               "decorator": {
                   "title": "Threshold",
                   "description": "The minimum confidence percentage for a positive
classification."
               }
           },
           {
               "name": "log_level",
               "interface": "string",
               "value": "INFO",
               "overridable": true,
               "decorator": {
                   "title": "Logging level",
                   "description": "DEBUG, INFO, WARNING, ERROR, or CRITICAL."
               }
           }
           ...
       ],
```

```
        "edges": [
            {
                "producer": "detection_threshold",
                "consumer": "code_node.threshold"
            },
            {
                "producer": "log_level",
                "consumer": "code_node.log_level"
            }
            ...
        ]
    }
```

您可以直接在應用程序清單中修改參數，或在部署時使用覆蓋提供新值。如需詳細資訊，請參閱[帶覆蓋的部署時配置](#)。

帶覆蓋的部署時配置

您可以在部署過程中配置參數和抽象節點。如果您使用 AWS Panorama 控制台進行部署，則可以為每個參數指定一個值並選擇攝像機流作為輸入。如果您使用 AWS Panorama API 部署應用程序，則可以使用覆蓋文檔指定這些設置。

覆蓋文檔的結構類似於應用程序清單。對於具有基本類型的參數，您可以定義一個節點。對於攝像機流，您可以定義映射到已註冊攝像機流的節點和包。然後，您可以為每個節點定義一個覆蓋，該節點從它替換的應用程序清單中指定節點。

Example 覆蓋 .json

```
{
    "nodeGraphOverrides": {
        "nodes": [
            {
                "name": "my_camera",
                "interface": "123456789012::exterior-south.exterior-south"
            },
            {
                "name": "my_region",
                "interface": "string",
                "value": "us-east-1"
            }
        ],
        "packages": [
            {
                "name": "123456789012::exterior-south",
                "version": "1.0"
            }
        ],
        "nodeOverrides": [
            {
                "replace": "camera_node",
                "with": [
                    {
                        "name": "my_camera"
                    }
                ]
            },
            {
                "replace": "region",
                "with": [
```

```
                    {
                        "name": "my_region"
                    }
                ]
            }
        ],
        "envelopeVersion": "2021-01-01"
    }
}
```

在上面的示例中，文檔定義了一個字符串參數和一個抽象攝像機節點的覆蓋。所以此nodeOverrides告訴 AWS Panorama 此文檔中的哪些節點覆蓋應用程序清單中的哪個節點。

建置AWS Panorama應用

應用程式在AWS Panorama設備可在視訊串流上執行電腦視覺工作。您可以結合 Python 程式碼和機器學習模型來建置電腦視覺應用程式，並將其部署到AWS Panorama透過網際網路設備。應用程式可以將影片傳送到顯示器，或使用 AWS 開發套件將結果傳送到 AWS 服務。

一個模型分析影像以偵測人物、車輛和其他物體。根據它在訓練過程中看到的圖像，該模型告訴您它認為某些東西是什麼，以及它在猜測中的信心。您可以使用自己的影像資料訓練模型，也可以開始使用範例。

該應用程序的代碼處理攝影機串流中的靜態影像，將其傳送至模型，然後處理結果。模型可能會偵測多個物件，並傳回其形狀和位置。代碼可以使用此信息向視頻添加文本或圖形，或將結果發送到AWS用於存儲或進一步處理的服務。

要從流中獲取圖像，與模型交互並輸出視頻，應用程序代碼使用該AWS Panorama應用程式 SDK。應用程序 SDK 是一個 Python 庫，支持使用生成的模型 PyTorch, MXNet 奇 TensorFlow。

主題

電腦視覺模型

計算機視覺模型是經過訓練以檢測圖像中的對象的軟件程序。模型通過訓練首先分析這些對象的圖像來學會識別一組對象。電腦視覺模型會將影像做為輸入，並輸出其偵測到的物件的相關資訊，例如物件類型及其位置。AWS 全景支援使用 Apache MXNet 和PyTorch建置的電腦視覺模型。TensorFlow

> **Note**
>
> 如需已透過 AWS Panorama 測試的預先建置模型清單，請參閱模型相容性。

章節

在程式碼中使用模型

模型會傳回一或多個結果，其中可包含偵測到之類別、位置資訊及其他資料的機率。下列範例顯示如何針對影像串流執行推論，並將模型的輸出傳送至處理函數。

Example application.py — 推論

```
    def process_media(self, stream):
        """Runs inference on a frame of video."""
        image_data = preprocess(stream.image,self.MODEL_DIM)
        logger.debug('Image data: {}'.format(image_data))
        # Run inference
        inference_start = time.time()
        inference_results = self.call({"data":image_data}, self.MODEL_NODE)
         # Log metrics
        inference_time = (time.time() - inference_start) * 1000
        if inference_time > self.inference_time_max:
            self.inference_time_max = inference_time
        self.inference_time_ms += inference_time
        # Process results (classification)
        self.process_results(inference_results, stream)
```

下列範例顯示處理基本分類模型結果的函數。示例模型返回概率的數組，這是結果數組中的第一個也是唯一的值。

Example application.py — 處理結果

```
    def process_results(self, inference_results, stream):
        """Processes output tensors from a computer vision model and annotates a video
frame."""
        if inference_results is None:
            logger.warning("Inference results are None.")
            return
        max_results = 5
        logger.debug('Inference results: {}'.format(inference_results))
        class_tuple = inference_results[0]
        enum_vals = [(i, val) for i, val in enumerate(class_tuple[0])]
        sorted_vals = sorted(enum_vals, key=lambda tup: tup[1])
        top_k = sorted_vals[::-1][:max_results]
        indexes =  [tup[0] for tup in top_k]

        for j in range(max_results):
            label = 'Class [%s], with probability %.3f.'% (self.classes[indexes[j]],
class_tuple[0][indexes[j]])
            stream.add_label(label, 0.1, 0.1 + 0.1*j)
```

應用程式程式碼會尋找機率最高的值，並將它們對應至初始化期間載入的資源檔案中的標籤。

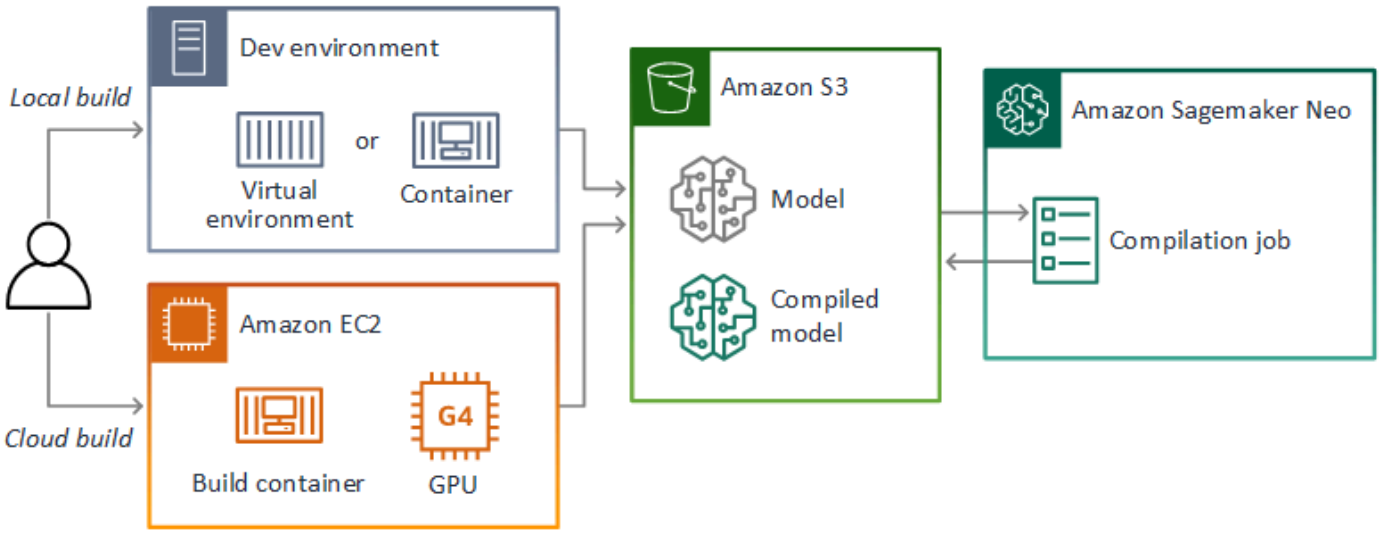
建立自訂模型

您可以使用在中建置的模型PyTorch、Apache MXNet 和 AWS 全景應用程式TensorFlow中。除了在中建置和訓練模型SageMaker，您可以使用訓練有素的模型，或使用支援的架構建立和訓練自己的模型，然後將其匯出到本機環境或 Amazon EC2 中。

> **Note**
>
> 如需 SageMaker Neo 支援的架構版本和檔案格式的詳細資訊，請參閱 Amazon SageMaker 開發人員指南中的支援架構。

本指南的儲存庫提供範例應用程式，以TensorFlowSavedModel格式示範 Keras 模型的此工作流程。它使用 TensorFlow 2，並且可以在虛擬環境或 Docker 容器中本地運行。範例應用程式還包括用於在 Amazon EC2 執行個體上建立模型的範本和指令碼。

- 自訂模型範例應用



AWS 全景使用 SageMaker Neo 編譯模型，以便在 AWS 全景設備上使用。對於每個框架，請使用 SageMakerNeo 支持的格式，並將模型打包到`.tar.gz`歸檔中。

如需詳細資訊，請參閱 Amazon SageMaker 開發人員指南中的使用 Neo 編譯和部署模型。

封裝模型

一個模型包包括一個描述符，包配置和模型歸檔。就像在應用程式映像套件中一樣，套件組態會告訴 AWS 全景服務，模型和描述元存放在 Amazon S3 中的位置。

Example 套件 /123456789012-擠壓器 _ 比爾托 -1.0/ 描述符

```
{
    "mlModelDescriptor": {
        "envelopeVersion": "2021-01-01",
        "framework": "PYTORCH",
        "frameworkVersion": "1.8",
        "precisionMode": "FP16",
        "inputs": [
            {
                "name": "data",
                "shape": [
                    1,
                    3,
```

```
                    224,
                    224
                ]
            }
        ]
    }
}
```

> **Note**
>
> 僅指定框架版本的主要和次要版本。如需支援PyTorch的 Apache MXNet 和TensorFlow版本的清單，請參閱支援的架構。

若要匯入模型，請使用 AWS 全景應用程式 CLI `import-raw-model` 命令。如果您對模型或其描述符進行任何更改，則必須重新運行此命令以更新應用程序的資產。如需詳細資訊，請參閱變更電腦視覺模型。

如需描述元檔案的 JSON 結構描述，請參閱資產描述器 .Schema.json。

訓練模型

訓練模型時，請使用來自目標環境或與目標環境非常類似的測試環境中的影像。請考慮下列可能會影響模型效能的因素：

- 照明 — 主體反射的光量決定了模型必須分析多少細節。使用光線充足的主體影像訓練的模型，在低光源或背光環境下可能無法正常運作。
- 解析度 — 模型的輸入大小通常以方形長寬比為 224 到 512 像素寬的解析度固定。在將視訊影格傳遞給模型之前，您可以縮小或裁剪它以符合所需的大小。
- 影像失真 — 相機的焦距和鏡頭形狀可能會導致影像遠離畫面中心的失真。相機的位置也決定了主體的哪些功能是可見的。例如，配備廣角鏡頭的頭頂攝影機會在畫面中央顯示主體的頂部，而當主體偏離中心時，主體側面的傾斜視圖則會顯示出來。

若要解決這些問題，您可以在將影像傳送至模型之前預先處理影像，並在反映真實環境中變化的各種影像上訓練模型。如果模型需要在照明情況下操作，並且需要使用各種相機，則需要更多資料進行訓練。除了收集更多圖像之外，您還可以通過創建偏斜或具有不同照明的現有圖像的變化來獲得更多訓練數據。

建立應用程式影像

AWS Panorama 設備會以容器檔案系統的形式執行應用程式，從您建立的映像匯出。您可以在使用 AWS Panorama 應用程式基礎映像作為起點的 Dockerfile 中指定應用程式的相依性和資源。

若要建立應用程式映像，請使用 Docker 和 AWS 全景應用程式 CLI。本指南範例應用程式的下列範例示範了這些使用案例。

Example 套件 /123456789012-樣本 _ 程式碼頭檔案

```
FROM public.ecr.aws/panorama/panorama-application
WORKDIR /panorama
COPY . .
RUN pip install --no-cache-dir --upgrade pip && \
    pip install --no-cache-dir -r requirements.txt
```

下面的碼頭文件指令被使用。

- FROM— 載入應用程式基礎影像 (public.ecr.aws/panorama/panorama-application)。
- WORKDIR— 設定影像上的工作目錄。 /panorama用於應用程序代碼和相關文件。此設定只會在建置期間持續存在，不會影響執行階段應用程式的工作目錄 (/)。
- COPY— 將檔案從本機路徑複製到影像上的路徑。 COPY . .將目前目錄 (套件目錄) 中的檔案複製到影像上的工作目錄中。例如，應用程式程式碼會從複製packages/123456789012-SAMPLE_CODE-1.0/application.py到/panorama/application.py。
- RUN-在構建過程中在映像上運行 shell 命令。單個RUN操作可以通過在命令&&之間使用順序運行多個命令。此範例會更新pip套件管理員，然後安裝中列出的程式庫requirements.txt。

您可以使用在構建時非常有用的其他指令ARG，例如ADD和。將執行階段資訊新增至容器的指示 (例如ENV，不適用於 AWS Panorama)。AWS 全景圖不會從映像執行容器。它僅使用影像匯出檔案系統，該檔案系統會傳輸至設備。

指定相依性

requirements.txt是指定應用程式所使用的程式庫的 Python 需求檔案。範例應用程式使用 Open CV 和AWS SDK for Python (Boto3).

Example 套裝軟體包 /樣本 _ 程式碼 -1.0 要求

```
boto3==1.24.*
opencv-python==4.6.*
```

Dockerfile 中的`pip install`命令將這些庫安裝到下面的 Python `dist-packages` 目錄中`/usr/local/lib`，以便可以通過應用程序代碼導入它們。

本機儲存

AWS 全景圖會保留應用程式儲存的`/opt/aws/panorama/storage`目錄。您的應用程式可以在此路徑上建立和修改檔案。在儲存目錄中建立的檔案會在重新開機時持續存在。開機時會清除其他暫存檔案位置。

建立影像資產

當您使用 AWS 全景應用程式 CLI 為應用程式套件建立映像時，CLI 會`docker build`在套件目錄中執行。這會建立包含應用程式程式碼的應用程式映像檔。然後 CLI 會建立容器、匯出其檔案系統、壓縮，並將其儲存在`assets`資料夾中。

```
$ panorama-cli build-container --container-asset-name code_asset --package-path
 packages/123456789012-SAMPLE_CODE-1.0
docker build -t code_asset packages/123456789012-SAMPLE_CODE-1.0 --pull
docker export --output=code_asset.tar $(docker create code_asset:latest)
gzip -1 code_asset.tar
{
    "name": "code_asset",
    "implementations": [
        {
            "type": "container",
            "assetUri":
 "6f67xmpl32743ed0e60c151a02f2f0da1bf70a4ab9d83fe236fa32a6f9b9f808.tar.gz",
            "descriptorUri":
 "1872xmpl129481ed053c52e66d6af8b030f9eb69b1168a29012f01c7034d7a8f.json"
        }
    ]
}
Container asset for the package has been succesfully built at  /home/
user/aws-panorama-developer-guide/sample-apps/aws-panorama-sample/
assets/6f67xmpl32743ed0e60c151a02f2f0da1bf70a4ab9d83fe236fa32a6f9b9f808.tar.gz
```

輸出中的 JSON 區塊是 CLI 新增至套件組態 (`package.json`) 並向 AWS 全景服務註冊的資產定義。CLI 也會複製描述器檔案，該檔案會指定應用程式指令碼 (應用程式的入口點) 的路徑。

Example 套件/程式碼

```
{
    "runtimeDescriptor":
    {
        "envelopeVersion": "2021-01-01",
        "entry":
        {
            "path": "python3",
            "name": "/panorama/application.py"
        }
    }
}
```

在 assets 資料夾中，描述器和應用程式影像會以其 SHA-256 總和檢查碼命名。此名稱在 Amazon S3 存放資產時，會用作資產的唯一識別碼。

從應用程式程式碼呼叫 AWS 服務

您可以使用從應AWS SDK for Python (Boto)用程式程式碼呼叫 AWS 服務。例如，如果您的模型偵測到異常情況，您可以將指標張貼到 Amazon CloudWatch、使用 Amazon SNS 傳送通知、將影像儲存到 Amazon S3，或叫用 Lambda 函數以進行進一步處理。大多數 AWS 服務都有可與 AWS 開發套件搭配使用的公用 API。

預設情況下，設備沒有存取得任何 AWS 服務。若要授與權限，請為應用程式建立角色，並在部署期間將其指派給應用程式執行個體。

章節

- 使用 Amazon S3
- 使用AWS IoT MQTT 主題

使用 Amazon S3

您可以使用 Amazon S3 來存放處理結果和其他應用程式資料。

```
import boto3
s3_client=boto3.client("s3")
s3_clients3.upload_file(data_file,
                    s3_bucket_name,
                    os.path.basename(data_file))
```

使用AWS IoT MQTT 主題

您可以使用適用 SDK for Python (Boto3) 的發件取得適用於 Python (Boto3) 的發件取AWS IoT得 在下列範例中，應用程式會張貼到以設備物件名稱命名的主題，您可以在主AWS IoT控台中找到該主題。

```
import boto3
iot_client=boto3.client('iot-data')
topic = "panorama/panorama_my-appliance_Thing_a01e373b"
iot_client.publish(topic=topic, payload="my message")
```

選擇一個名稱，表示您選擇的設備 ID 或其他標識符。若要發佈訊息，應用程式需要呼叫的權限`iot:Publish`。

若要監視 MQTT 佇列

1. 開啟AWS IoT主控台 [測試] 頁面。
2. 對於訂用的主題，請輸入主題名稱。例如：`panorama/panorama_my-appliance_Thing_a01e373b`。
3. 請選擇 Subscribe to topic (訂閱主題)。

AWS Panorama 應用程序軟件開發工具包

AWS Panorama 應用程序開發工具包是一個用於開發 AWS Panorama 應用程序的 Python 庫。在您的程式碼，您可以使用 AWS Panorama 應用程序軟件開發工具包加載計算機視覺模型、運行推理並將視頻輸出到顯示器。

> **Note**
>
> 為確保您可以使用 AWS Panorama 應用程式的程式碼開發套件的最新功能，升級裝置軟件。

有關應用程序 SDK 定義的類及其方法的詳細信息，請參閱製作者開發套件參考。

章節

- 添加文本和框以輸出視頻

添加文本和框以輸出視頻

藉助 AWS Panorama 軟件開發工具包，您可以將視頻流輸出到顯示器。視頻可以包含顯示模型輸出、應用程序當前狀態或其他數據的文本和框。

中的每個對象`video_in`陣列是來自連接到設備的照相機流的圖像。此物件的類型為`panoramasdk.media`。它具有向圖像添加文本和矩形框的方法，然後您可以將其分配給`video_out`陣列。

在以下示例中，示例應用程序為每個結果添加一個標籤。每個結果都定位在同一左側位置，但高度不同。

```
        for j in range(max_results):
            label = 'Class [%s], with probability %.3f.'% (self.classes[indexes[j]],
class_tuple[0][indexes[j]])
            stream.add_label(label, 0.1, 0.1 + 0.1*j)
```

要向輸出圖像添加框，請使用`add_rect`。此方法採用 0 到 1 之間的 4 個值，指示框左上角和右下角的位置。

```
        w,h,c = stream.image.shape
        stream.add_rect(x1/w, y1/h, x2/w, y2/h)
```

執行多執行緒

您可以在處理執行緒上執行應用程式邏輯，並將其他執行緒用於其他背景處理序。例如，您可以建立執行緒檢查 HTTP 流量時用於調試，或監視推論結果並將數據發送到的線程AWS。

若要執行多個執行緒，請使用執行緒模組從 Python 標準庫為每個進程創建一個線程。下列範例顯示除錯伺服器範例應用程式的主要迴圈，該應用程式會建立應用程式物件並使用它來執行三執行緒。

Example 套件 /123456789012-除錯 _ 伺服器 -1.0/應用程式.PY— 主要迴圈

```
def main():
    panorama = panoramasdk.node()
    while True:
        try:
            # Instantiate application
            logger.info('INITIALIZING APPLICATION')
            app = Application(panorama)
            # Create threads for stream processing, debugger, and client
            app.run_thread = threading.Thread(target=app.run_cv)
            app.server_thread = threading.Thread(target=app.run_debugger)
            app.client_thread = threading.Thread(target=app.run_client)
            # Start threads
            logger.info('RUNNING APPLICATION')
            app.run_thread.start()
            logger.info('RUNNING SERVER')
            app.server_thread.start()
            logger.info('RUNNING CLIENT')
            app.client_thread.start()
            # Wait for threads to exit
            app.run_thread.join()
            app.server_thread.join()
            app.client_thread.join()
            logger.info('RESTARTING APPLICATION')
        except:
            logger.exception('Exception during processing loop.')
```

當所有執行緒結束時，應用程式會自行重新啟動。所以此run_cv循環處理來自攝像機流的圖像。如果收到要停止的信號，它會關閉調試器進程，該調試器進程運行 HTTP 服務器並且無法自行關閉。每個線程都必須處理自己的錯誤。如果未捕捉到並記錄錯誤，則執行緒會以無訊息方式結束。

Example 套件 /123456789012-除錯 _ 伺服器 -1.0/應用程式.PY— 處理迴圈

```
    # Processing loop
    def run_cv(self):
        """Run computer vision workflow in a loop."""
        logger.info("PROCESSING STREAMS")
        while not self.terminate:
            try:
                self.process_streams()
                # turn off debug logging after 15 loops
                if logger.getEffectiveLevel() == logging.DEBUG and self.frame_num ==
15:
                    logger.setLevel(logging.INFO)
            except:
                logger.exception('Exception on processing thread.')
        # Stop signal received
        logger.info("SHUTTING DOWN SERVER")
        self.server.shutdown()
        self.server.server_close()
        logger.info("EXITING RUN THREAD")
```

線程通過應用程序的通信self物件。若要重新啟動應用程式處理迴圈，除錯程式執行緒會呼叫stop方法。此方法設置terminate屬性，這表示其他線程關閉。

Example 套件 /123456789012-除錯 _ 伺服器 -1.0/應用程式.PY— 停止方法

```
    # Interrupt processing loop
    def stop(self):
        """Signal application to stop processing."""
        logger.info("STOPPING APPLICATION")
        # Signal processes to stop
        self.terminate = True
    # HTTP debug server
    def run_debugger(self):
        """Process debug commands from local network."""
        class ServerHandler(SimpleHTTPRequestHandler):
            # Store reference to application
            application = self
            # Get status
            def do_GET(self):
                """Process GET requests."""
                logger.info('Get request to {}'.format(self.path))
                if self.path == "/status":
```

```
                self.send_200('OK')
            else:
                self.send_error(400)
        # Restart application
        def do_POST(self):
            """Process POST requests."""
            logger.info('Post request to {}'.format(self.path))
            if self.path == '/restart':
                self.send_200('OK')
                ServerHandler.application.stop()
            else:
                self.send_error(400)
```

服務傳入流量

您可以透過在應用程式程式碼旁邊執行 HTTP 伺服器，在本機監視或偵錯應用程式。為了提供外部流量，您可以將 AWS Panorama 設備上的連接埠對應到應用程式容器上的連接埠。

> **Important**
>
> 根據預設，AWS Panorama 設備不接受任何連接埠上的傳入流量。在應用裝置上開啟連接埠存在隱含的安全風險。使用此功能時，您必須採取其他步驟保護您的設備免受外部流量的影響以及授權用戶端與設備之間的安全通訊。
> 本指南隨附的範例程式碼僅用於示範目的，不會實作驗證、授權或加密。

您可以在設備上開啟範圍為 8000-9000 的連接埠。這些連接埠在開啟時，可以接收來自任何可路由用戶端的流量。部署應用程式時，您可以指定要開啟的連接埠，並將應用裝置上的連接埠對應至應用程式容器上的連接埠。設備軟體會將流量轉送至容器，並將回應傳送回應給要求者。要求會在您指定的設備連接埠上接收，而回應則會在隨機暫時連接埠上傳出。

設定傳入連接埠

您可以在應用程式組態中的三個位置指定連接埠對應。該代碼包的`package.json`時，您可以指定程式碼節點偵聽的連接埠`network`區塊。下列範例宣告節點在連接埠 80 上偵聽。

Example 套件 /123456789012-除錯 _ 伺服器

```
            "outputs": [
                {
                    "description": "Video stream output",
                    "name": "video_out",
                    "type": "media"
                }
            ],
            "network": {
                "inboundPorts": [
                    {
                        "port": 80,
                        "description": "http"
                    }
                ]
            }
```

在應用程式資訊清單中，您可以宣告將應用裝置上的連接埠對應至應用程式程式碼容器上的連接埠的路由規則。下列範例會新增將裝置上的連接埠 8080 對應至code_node容器。

Example 圖形/我的應用程序/圖形.json

```
            {
                "producer": "model_input_width",
                "consumer": "code_node.model_input_width"
            },
            {
                "producer": "model_input_order",
                "consumer": "code_node.model_input_order"
            }
        ],
        "networkRoutingRules": [
            {
                "node": "code_node",
                "containerPort": 80,
                "hostPort": 8080,
                "decorator": {
                    "title": "Listener port 8080",
                    "description": "Container monitoring and debug."
                }
            }
        ]
```

部署應用程式時，您可以在 AWS Panorama 主控台中指定相同的規則，或是使用覆寫文件傳遞給CreateApplicationInstanceAPI。您必須在部署時提供此組態，以確認要在應用裝置上開啟連接埠。

Example 圖形/我的應用程序/覆蓋.json

```
            {
                "replace": "camera_node",
                "with": [
                    {
                        "name": "exterior-north"
                    }
                ]
            }
        ],
        "networkRoutingRules":[
            {
                "node": "code_node",
```

```
                "containerPort": 80,
                "hostPort": 8080
            }
        ],
        "envelopeVersion": "2021-01-01"
    }
}
```

如果應用程式資訊清單中指定的裝置連接埠正在被其他應用程式使用，您可以使用覆寫文件來選擇不同的連接埠。

服務流量

在容器上開啟連接埠的情況下，您可以開啟通訊端或執行伺服器來處理傳入的要求。所以此debug-server範例顯示與電腦視覺應用程式程式碼一起執行的 HTTP 伺服器的基本實作。

> **Important**
>
> 示例實現對於生產使用不安全。為避免讓您的裝置容易受到攻擊，您必須在程式碼和網路設定中實作適當的安全性控制。

Example 套件 /123456789012-除錯 _ 伺服器 -1.0/應用程式.PY— HTTP 伺服器

```
    # HTTP debug server
    def run_debugger(self):
        """Process debug commands from local network."""
        class ServerHandler(SimpleHTTPRequestHandler):
            # Store reference to application
            application = self
            # Get status
            def do_GET(self):
                """Process GET requests."""
                logger.info('Get request to {}'.format(self.path))
                if self.path == '/status':
                    self.send_200('OK')
                else:
                    self.send_error(400)
            # Restart application
            def do_POST(self):
                """Process POST requests."""
```

```
                logger.info('Post request to {}'.format(self.path))
                if self.path == '/restart':
                    self.send_200('OK')
                    ServerHandler.application.stop()
                else:
                    self.send_error(400)
            # Send response
            def send_200(self, msg):
                """Send 200 (success) response with message."""
                self.send_response(200)
                self.send_header('Content-Type', 'text/plain')
                self.end_headers()
                self.wfile.write(msg.encode('utf-8'))
        try:
            # Run HTTP server
            self.server = HTTPServer(("", self.CONTAINER_PORT), ServerHandler)
            self.server.serve_forever(1)
            # Server shut down by run_cv loop
            logger.info("EXITING SERVER THREAD")
        except:
            logger.exception('Exception on server thread.')
```

服務器接受 GET 請求/status路徑來檢索有關應用程序的一些信息。它也接受 POST 請求/restart以重新啟動應用程式。

為了示範此功能，範例應用程式會在個別執行緒上執行 HTTP 用戶端。用戶端呼叫/status啟動後不久通過本地網絡的路徑，並在幾分鐘後重新啟動應用程序。

Example 套件 /123456789012-除錯 _ 伺服器 -1.0/應用程式.PY— HTTP 用戶端

```
    # HTTP test client
    def run_client(self):
        """Send HTTP requests to device port to demnostrate debug server functions."""
        def client_get():
            """Get container status"""
            r = requests.get('http://{}:{}/status'.format(self.device_ip,
self.DEVICE_PORT))
            logger.info('Response: {}'.format(r.text))
            return
        def client_post():
            """Restart application"""
            r = requests.post('http://{}:{}/restart'.format(self.device_ip,
self.DEVICE_PORT))
```

```
            logger.info('Response: {}'.format(r.text))
            return
        # Call debug server
        while not self.terminate:
            try:
                time.sleep(30)
                client_get()
                time.sleep(300)
                client_post()
            except:
                logger.exception('Exception on client thread.')
        # stop signal received
        logger.info("EXITING CLIENT THREAD")
```

主循環管理線程並在退出時重新啟動應用程序。

Example 套件 /123456789012-除錯 _ 伺服器 -1.0/應用程式.PY— 主要迴圈

```
def main():
    panorama = panoramasdk.node()
    while True:
        try:
            # Instantiate application
            logger.info('INITIALIZING APPLICATION')
            app = Application(panorama)
            # Create threads for stream processing, debugger, and client
            app.run_thread = threading.Thread(target=app.run_cv)
            app.server_thread = threading.Thread(target=app.run_debugger)
            app.client_thread = threading.Thread(target=app.run_client)
            # Start threads
            logger.info('RUNNING APPLICATION')
            app.run_thread.start()
            logger.info('RUNNING SERVER')
            app.server_thread.start()
            logger.info('RUNNING CLIENT')
            app.client_thread.start()
            # Wait for threads to exit
            app.run_thread.join()
            app.server_thread.join()
            app.client_thread.join()
            logger.info('RESTARTING APPLICATION')
        except:
            logger.exception('Exception during processing loop.')
```

若要部署範例應用程式，請參閱本指南中的說明 GitHub 儲存庫。

使用顯示卡

您可以存取 AWS 全景設備上的圖形處理器 (GPU) 以使用 GPU 加速程式庫，或在應用程式程式碼中執行機器學習模型。若要開啟 GPU 存取，請在建立應用程式程式碼容器之後，將 GPU 存取新增為套件組態的必要條件。

> **Important**
>
> 如果啟用 GPU 存取，則無法在設備上的任何應用程式中執行模型節點。基於安全考量，當設備執行以 SageMaker Neo 編譯的模型時，GPU 存取會受到限制。透過 GPU 存取，您必須在應用程式程式碼節點中執行模型，而裝置上的所有應用程式都會共用 GPU 的存取權。

若要為應用程式開啟 GPU 存取功能，請在使用 AWS 全景應用程式 CLI 建立套件之後更新套件組態。下列範例顯示將 GPU 存取新增至應用程式程式碼節點的`requirements`區塊。

Example 包含要求區塊的封裝

```
{
    "nodePackage": {
        "envelopeVersion": "2021-01-01",
        "name": "SAMPLE_CODE",
        "version": "1.0",
        "description": "Computer vision application code.",
        "assets": [
            {
                "name": "code_asset",
                "implementations": [
                    {
                        "type": "container",
                        "assetUri":
 "eba3xmpl71aa387e8f89be9a8c396416cdb80a717bb32103c957a8bf41440b12.tar.gz",
                        "descriptorUri":
 "4abdxmpl5a6f047d2b3047adde44704759d13f0126c00ed9b4309726f6bb43400ba9.json",
                        "requirements": [
                            {
                                "type": "hardware_access",
                                "inferenceAccelerators": [
                                    {
                                        "deviceType": "nvhost_gpu",
                                        "sharedResourcePolicy": {
```

```
                                        "policy" : "allow_all"
                                    }
                                }
                            ]
                        }
                    ]
                }
            ],
            "interfaces": [
            ...
```

更新開發工作流程中建置和封裝步驟之間的套件組態。

部署具有 GPU 存取權的應用程式

1. 若要建置應用程式容器，請使用build-container指令。

```
$ panorama-cli build-container --container-asset-name code_asset --package-path
 packages/123456789012-SAMPLE_CODE-1.0
```

2. 將requirements區塊新增至套件組態。
3. 若要上傳容器資產和套件組態，請使用package-application指令。

```
$ panorama-cli package-application
```

4. 部署應用程式。

如需使用 GPU 存取的範例應用程式，請造訪aws-panorama-samplesGitHub存放庫。

在 Windows 中設置開發環境

要構建 AWS Panorama 應用程序，請使用 Docker、命令行工具和 Python。在 Windows 中，您可以通過使用 Docker 桌面和適用於 Linux 和 Ubuntu 的 Windows 子系統來設置開發環境。本教程將引導您完成已使用 AWS Panorama 工具和示例應用程序測試的開發環境的設置過程。

章節

先決條件

若要遵循本教程，您需要一個支持 Windows 子系統的版本，適用於 Linux 2（WSL 2）。

- 視窗 10 1903 及更高版本（建立 18362 及更高版本）或視窗 11
- 視窗功能
  - 適用於 Linux 的 Windows 子系統
  - Hyper-V：
  - 虛擬機器平台

本教程使用以下軟件版本開發。

- Ubuntu 20.04
- Python 3.8.5
- 碼頭

安裝 WSL 2 和烏本圖

如果您的 Windows 10 版本 2004 年及更高版本（版本 19041 及更高版本），則可以使用以下 PowerShell 命令安裝 WSL 2 和 Ubuntu 20.04。

```
> wsl --install -d Ubuntu-20.04
```

若要使用較舊版本的 Windows，請依照 WSL 2 文件中的指示進行：舊版本的手動安裝步驟

安裝 Docker

若要安裝 Docker 桌面，請從集線器。如果遇到問題，請按照 Docker 網站上的說明操作：碼頭桌面 WSL 2 後端。

運行 Docker 桌面並按照第一次運行的教程構建示例容器。

> **Note**
>
> Docker 桌面僅在默認發行版中啟用 Docker。如果在運行本教程之前安裝了其他 Linux 發行版，請在新安裝的 Ubuntu 發行版中啟用 Docker 桌面設置菜單資源、WSL 集成。

配置 Ubuntu

您現在可以在 Ubuntu 虛擬機中運行 Docker 命令。要打開命令行終端，請從開始菜單運行分發。第一次運行它時，您需要配置用户名和密碼，您可以使用該用户名和密碼來運行管理員命令。

要完成開發環境的配置，請更新虛擬機的軟件並安裝工具。

若要配置虛擬機器

1. 更新 Ubuntu 附帶的軟件。

   ```
   $ sudo apt update && sudo apt upgrade -y && sudo apt autoremove
   ```

2. 使用 apt 安裝開發工具。

   ```
   $ sudo apt install unzip python3-pip
   ```

3. 使用 pip 安裝 Python 程式庫。

   ```
   $ pip3 install awscli panoramacli
   ```

4. 打開新的終端機，然後運行`aws configure`配置AWS CLI。

```
$ aws configure
```

如果您沒有存取金鑰，可以在IAM 主控台。

最後，下載並導入示例應用程序。

若要獲取範例應用程式

1. 下載並解壓縮範例應用程式。

```
$ wget https://github.com/awsdocs/aws-panorama-developer-guide/releases/download/
v1.0-ga/aws-panorama-sample.zip
$ unzip aws-panorama-sample.zip
$ cd aws-panorama-sample
```

2. 運行包含的腳本來測試編譯、構建應用程序容器並將軟件包上傳到 AWS Panorama。

```
aws-panorama-sample$ ./0-test-compile.sh
aws-panorama-sample$ ./1-create-role.sh
aws-panorama-sample$ ./2-import-app.sh
aws-panorama-sample$ ./3-build-container.sh
aws-panorama-sample$ ./4-package-app.sh
```

AWS Panorama 應用程序 CLI 上傳軟件包並將其註冊到 AWS Panorama 服務中。您現在可以部署範例應用程式與 AWS Panorama 控制台一起使用。

下一步驟

要瀏覽和編輯項目文件，可以使用文件資源管理器或支持 WSL 的集成開發環境 (IDE)。

要訪問虛擬機的文件系統，請打開「文件資源管理器」並輸入\\wsl$在導覽列中。此目錄包含一個指向虛擬機文件系統的鏈接（Ubuntu-20.04）以及 Docker 數據的文件系統。在Ubuntu-20.04，您的用户目錄位於home*username*。

> **Note**
>
> 要從 Ubuntu 中訪問 Windows 安裝中的文件，請導航到/mnt/c目錄中。例如，您可以在下載目錄中運行ls /mnt/c/Users/*windows-username*/Downloads。

使用 Visual Studio 代碼，您可以在開發環境中編輯應用程序代碼，並使用集成終端運行命令。若要安裝可視工作室代碼，請訪問可視化工作代碼。安裝完成後，添加遠端 WSL延伸。

Windows 終端是您一直在運行命令的標準 Ubuntu 終端的替代方案。它支持多個選項卡，並且可以為您安裝的任何其他各種 Linux 運行 PowerShell、命令提示符和終端。它支持使用Ctrl+C和Ctrl+V、可點擊 URL 以及其他有用的改進。若要安裝 Windows 終端，請訪問微軟。

AWS Panorama

您可以使用 AWS Panorama 服務的公有 API 來自動化裝置和應用程式管理工作流程。使用AWS Command Line Interface或AWS SDK，您可以開發用於管理資源和部署的指令碼或應用程式。本指南的 GitHub 儲存庫包含指令碼，您可以使用這些腳本做為自己程式碼的起點。

- aws-panorama-developer-guide/實用程序腳本

章節

自動化設備註冊

要置備設備，請使用ProvisionDeviceAPI。響應包括一個 ZIP 文件，其中包含設備的配置和臨時證書。對文件進行解碼並將其保存在帶前綴的檔案`certificates-omni_`。

Example provision-device.sh

```
if [[ $# -eq 1 ]] ; then
    DEVICE_NAME=$1
else
    echo "Usage: ./provision-device.sh <device-name>"
    exit 1
fi
CERTIFICATE_BUNDLE=certificates-omni_${DEVICE_NAME}.zip
aws panorama provision-device --name ${DEVICE_NAME} --output text --query Certificates
 | base64 --decode > ${CERTIFICATE_BUNDLE}
echo "Created certificate bundle ${CERTIFICATE_BUNDLE}"
```

配置歸檔文件中的憑據將在 5 分鐘後過期。使用隨附的 USB 驅動器將歸檔文件傳輸到您的裝置。

要註冊攝像機，請使用從模板作業創建節點API。此 API 將獲取相機用户名、密碼和 URL 的模板參數映射。您可以使用 Bash 字符串操作將此映射格式化為 JSON 文檔。

Example register-camera.sh

```
if [[ $# -eq 3 ]] ; then
    NAME=$1
    USERNAME=$2
    URL=$3
else
    echo "Usage: ./register-camera.sh <stream-name> <username> <rtsp-url>"
    exit 1
fi
echo "Enter camera stream password: "
read PASSWORD
TEMPLATE='{"Username":"MY_USERNAME","Password":"MY_PASSWORD","StreamUrl": "MY_URL"}'
TEMPLATE=${TEMPLATE/MY_USERNAME/$USERNAME}
TEMPLATE=${TEMPLATE/MY_PASSWORD/$PASSWORD}
TEMPLATE=${TEMPLATE/MY_URL/$URL}
echo ${TEMPLATE}
```

```
JOB_ID=$(aws panorama create-node-from-template-job --template-type RTSP_CAMERA_STREAM
 --output-package-name ${NAME} --output-package-version "1.0" --node-name ${NAME} --
template-parameters "${TEMPLATE}" --output text)
```

或者，您也可以從文件加載 JSON 配置。

```
--template-parameters file://camera-template.json
```

使用 AWS Panorama API 管理設備

您可以使用 AWS Panorama API 自動執行設備管理任務。

檢視裝置

若要取得具有裝置 ID 的設備清單，請使用 ListDevicesAPI。

```
$ aws panorama list-devices
    "Devices": [
        {
            "DeviceId": "device-4tafxmplhtmzabv5lsacba4ere",
            "Name": "my-appliance",
            "CreatedTime": 1652409973.613,
            "ProvisioningStatus": "SUCCEEDED",
            "LastUpdatedTime": 1652410973.052,
            "LeaseExpirationTime": 1652842940.0
        }
    ]
}
```

若要取得有關設備的更多詳細資訊，請使用 DescribeDeviceAPI。

```
$ aws panorama describe-device --device-id device-4tafxmplhtmzabv5lsacba4ere
{
    "DeviceId": "device-4tafxmplhtmzabv5lsacba4ere",
    "Name": "my-appliance",
    "Arn": "arn:aws:panorama:us-west-2:123456789012:device/
device-4tafxmplhtmzabv5lsacba4ere",
    "Type": "PANORAMA_APPLIANCE",
    "DeviceConnectionStatus": "ONLINE",
    "CreatedTime": 1648232043.421,
    "ProvisioningStatus": "SUCCEEDED",
    "LatestSoftware": "4.3.55",
    "CurrentSoftware": "4.3.45",
    "SerialNumber": "GFXMPL0013023708",
    "Tags": {},
    "CurrentNetworkingStatus": {
        "Ethernet0Status": {
            "IpAddress": "192.168.0.1/24",
            "ConnectionStatus": "CONNECTED",
            "HwAddress": "8C:XM:PL:60:C5:88"
```

```
        },
        "Ethernet1Status": {
            "IpAddress": "--",
            "ConnectionStatus": "NOT_CONNECTED",
            "HwAddress": "8C:XM:PL:60:C5:89"
        }
    },
    "LeaseExpirationTime": 1652746098.0
}
```

升級設備軟體

如果LatestSoftware版本比新CurrentSoftware，您可以升級裝置。使用 CreateJobForDevicesAPI 建立 over-the-air (OTA) 更新工作。

```
$ aws panorama create-job-for-devices --device-ids device-4tafxmplhtmzabv5lsacba4ere \
  --device-job-config '{"OTAJobConfig": {"ImageVersion": "4.3.55"}}' --job-type OTA
{
    "Jobs": [
        {
            "JobId": "device-4tafxmplhtmzabv5lsacba4ere-0",
            "DeviceId": "device-4tafxmplhtmzabv5lsacba4ere"
        }
    ]
}
```

在腳本中，您可以使用 Bash 字符串操作在作業配置文件中填入映像版本字段。

Example check-updates.sh

```
apply_update() {
    DEVICE_ID=$1
    NEW_VERSION=$2
    CONFIG='{"OTAJobConfig": {"ImageVersion": "NEW_VERSION"}}'
    CONFIG=${CONFIG/NEW_VERSION/$NEW_VERSION}
    aws panorama create-job-for-devices --device-ids ${DEVICE_ID} --device-job-config
 "${CONFIG}" --job-type OTA
}
```

設備會下載指定的軟體版本並自行更新。使用 DescribeDeviceJobAPI 觀看更新的進度。

```
$ aws panorama describe-device-job --job-id device-4tafxmplhtmzabv5lsacba4ere-0
```

```
{
    "JobId": "device-4tafxmplhtmzabv5lsacba4ere-0",
    "DeviceId": "device-4tafxmplhtmzabv5lsacba4ere",
    "DeviceArn": "arn:aws:panorama:us-west-2:559823168634:device/
device-4tafxmplhtmzabv5lsacba4ere",
    "DeviceName": "my-appliance",
    "DeviceType": "PANORAMA_APPLIANCE",
    "ImageVersion": "4.3.55",
    "Status": "REBOOTING",
    "CreatedTime": 1652410232.465
}
```

若要取得中任務的清單，請使用ListDevicesJobs。

```
$ aws panorama list-devices-jobs
{
    "DeviceJobs": [
        {
            "DeviceName": "my-appliance",
            "DeviceId": "device-4tafxmplhtmzabv5lsacba4ere",
            "JobId": "device-4tafxmplhtmzabv5lsacba4ere-0",
            "CreatedTime": 1652410232.465
        }
    ]
}
```

如需檢查並套用更新的範例指令碼，請參閱本指南 GitHub 儲存庫中的 check-updates.sh。

重新開機

若要重新啟動設備，請使用 CreateJobForDevicesAPI。

```
$ aws panorama create-job-for-devices --device-ids device-4tafxmplhtmzabv5lsacba4ere --
job-type REBOOT
{
    "Jobs": [
        {
            "JobId": "device-4tafxmplhtmzabv5lsacba4ere-0",
            "DeviceId": "device-4tafxmplhtmzabv5lsacba4ere"
        }
    ]
}
```

在腳本中，您可以獲取設備列表，然後選擇一個以交互方式重新啟動。

Example reboot-device.sh — 用法

```
$ ./reboot-device.sh
Getting devices...
0: device-53amxmplyn3gmj72epzanacniy     my-se70-1
1: device-6talxmpl5mmik6qh5moba6jium     my-manh-24
Choose a device
1
Reboot device device-6talxmpl5mmik6qh5moba6jium? (y/n)y
{
    "Jobs": [
        {
            "DeviceId": "device-6talxmpl5mmik6qh5moba6jium",
            "JobId": "device-6talxmpl5mmik6qh5moba6jium-8"
        }
    ]
}
```

自動化應用部署

若要部署應用程式，您可以同時使用 AWS 全景應用程式 CLI 和AWS Command Line Interface。建立應用程式容器之後，您可以將它和其他資產上傳到 Amazon S3 存取點。然後，您可以使用CreateApplicationInstanceAPI。

如需使用所示指令碼的詳細內容和指示，請遵循應用程式讀我檔。

章節

- 建置容器
- 上傳容器並註冊節點
- 部署應用程式
- 監控部署

建置容器

若要建置應用程式容器，請使用`build-container`指令。此命令會建置 Docker 容器，並將其儲存為壓縮檔案系統`assets`資料夾。

Example 3-build-container.sh

```
CODE_PACKAGE=SAMPLE_CODE
ACCOUNT_ID=$(aws sts get-caller-identity --output text --query 'Account')
panorama-cli build-container --container-asset-name code_asset --package-path packages/
${ACCOUNT_ID}-${CODE_PACKAGE}-1.0
```

您也可以使用命令列完成來填入 path 引數，方法是輸入部分路徑，然後按TAB。

```
$ panorama-cli build-container --package-path packages/TAB
```

上傳容器並註冊節點

若要上傳應用程式，請使用`package-application`指令。此指令會從`assets`資料夾至 AWS 全景管理的亞馬遜 S3 存取點。

Example 4-package-app.sh

```
panorama-cli package-application
```

AWS 全景應用程式 CLI 會上傳套件組態所參照的容器和描述元資產 (package.json) 在每個套件中，並將套件註冊為 AWS 全景中的節點。然後，您在應用程序清單中引用這些節點（graph.json）以部署應用程式。

部署應用程式

若要部署應用程式，請使用CreateApplicationInstanceAPI。此操作採用以下參數，其中包括。

- ManifestPayload-應用程序清單（graph.json），定義應用程式的節點、套件、邊緣和參數。
- ManifestOverridesPayload-覆蓋第一個參數的第二個清單。應用程式資訊清單可視為應用程式來源中的靜態資源，覆寫資訊清單會提供自訂部署的部署時間設定。
- DefaultRuntimeContextDevice— 目標設備。
- RuntimeRoleArn— 應用程式用來存取 AWS 服務和資源的 IAM 角色的 ARN。
- ApplicationInstanceIdToReplace— 要從裝置移除的現有應用程式執行個體 ID。

清單和覆寫有效載荷是 JSON 文件，必須以嵌套在另一個文件內的字串值形式提供。若要這麼做，指令碼會將檔案中的資訊清單載入為字串，並使用jq 工具以建構巢狀文件。

Example 5-deploy.sh— 撰寫清單

```
GRAPH_PATH="graphs/my-app/graph.json"
OVERRIDE_PATH="graphs/my-app/override.json"
# application manifest
GRAPH=$(cat ${GRAPH_PATH} | tr -d '\n' | tr -d '[:blank:]')
MANIFEST="$(jq --arg value "${GRAPH}" '.PayloadData="\($value)"' <<< {})"
# manifest override
OVERRIDE=$(cat ${OVERRIDE_PATH} | tr -d '\n' | tr -d '[:blank:]')
MANIFEST_OVERRIDE="$(jq --arg value "${OVERRIDE}" '.PayloadData="\($value)"' <<< {})"
```

部署指令碼使用ListDevicesAPI 可取得目前區域中已註冊裝置的清單，並將使用者選擇儲存至本機檔案，以供後續部署使用。

Example 5-deploy.sh— 查找設備

```
    echo "Getting devices..."
    DEVICES=$(aws panorama list-devices)
```

```
    DEVICE_NAMES=($((echo ${DEVICES} | jq -r '.Devices |=sort_by(.LastUpdatedTime) |
 [.Devices[].Name] | @sh') | tr -d \'\"))
    DEVICE_IDS=($((echo ${DEVICES} | jq -r '.Devices |=sort_by(.LastUpdatedTime) |
 [.Devices[].DeviceId] | @sh') | tr -d \'\"))
    for (( c=0; c<${#DEVICE_NAMES[@]}; c++ ))
    do
        echo "${c}: ${DEVICE_IDS[${c}]}     ${DEVICE_NAMES[${c}]}"
    done
    echo "Choose a device"
    read D_INDEX
    echo "Deploying to device ${DEVICE_IDS[${D_INDEX}]}"
    echo -n ${DEVICE_IDS[${D_INDEX}]} > device-id.txt
    DEVICE_ID=$(cat device-id.txt)
```

應用程式角色是由另一個指令碼建立 (1-create-role.sh). 部署指令碼會從中取得此角色的 ARNAWS CloudFormation。如果應用程式已部署至裝置，則指令碼會從本機檔案取得該應用程式執行個體的 ID。

Example 5-deploy.sh-角色 ARN 和替換參數

```
# application role
STACK_NAME=panorama-${NAME}
ROLE_ARN=$(aws cloudformation describe-stacks --stack-name panorama-${PWD##*/} --query
 'Stacks[0].Outputs[?OutputKey==`roleArn`].OutputValue' --output text)
ROLE_ARG="--runtime-role-arn=${ROLE_ARN}"

# existing application instance id
if [ -f "application-id.txt" ]; then
    EXISTING_APPLICATION=$(cat application-id.txt)
    REPLACE_ARG="--application-instance-id-to-replace=${EXISTING_APPLICATION}"
    echo "Replacing application instance ${EXISTING_APPLICATION}"
fi
```

最後，該腳本將所有部分放在一起以創建應用程序實例並將應用程序部署到設備。服務會以指令碼儲存的執行個體 ID 來回應，以供日後使用。

Example 5-deploy.sh— 部署應用程式

```
APPLICATION_ID=$(aws panorama create-application-instance ${REPLACE_ARG} --manifest-
payload="${MANIFEST}" --default-runtime-context-device=${DEVICE_ID} --name=${NAME}
 --description="command-line deploy" --tags client=sample --manifest-overrides-
payload="${MANIFEST_OVERRIDE}" ${ROLE_ARG} --output text)
```

```
echo "New application instance ${APPLICATION_ID}"
echo -n $APPLICATION_ID > application-id.txt
```

監控部署

若要監視部署，請使用ListApplicationInstancesAPI。監視器指令碼會從應用程式目錄中的檔案取得裝置 ID 和應用程式執行個體 ID，並使用它們建構 CLI 命令。然後它在一個循環中調用。

Example 6-monitor-deployment.sh

```
APPLICATION_ID=$(cat application-id.txt)
DEVICE_ID=$(cat device-id.txt)
QUERY="ApplicationInstances[?ApplicationInstanceId==\`APPLICATION_ID\`]"
QUERY=${QUERY/APPLICATION_ID/$APPLICATION_ID}
MONITOR_CMD="aws panorama list-application-instances --device-id ${DEVICE_ID} --query
 ${QUERY}"
MONITOR_CMD=${MONITOR_CMD/QUERY/$QUERY}
while true; do
    $MONITOR_CMD
    sleep 60
done
```

部署完成後，您可以通過調用亞馬遜查看日誌CloudWatch記錄檔 API。檢視記錄指令碼使用CloudWatch日誌GetLogEventsAPI。

Example view-logs.sh

```
GROUP="/aws/panorama/devices/MY_DEVICE_ID/applications/MY_APPLICATION_ID"
GROUP=${GROUP/MY_DEVICE_ID/$DEVICE_ID}
GROUP=${GROUP/MY_APPLICATION_ID/$APPLICATION_ID}
echo "Getting logs for group ${GROUP}."
#set -x
while true
do
    LOGS=$(aws logs get-log-events --log-group-name ${GROUP} --log-stream-name
 code_node --limit 150)
    readarray -t ENTRIES < <(echo $LOGS | jq -c '.events[].message')
    for ENTRY in "${ENTRIES[@]}"; do
        echo "$ENTRY" | tr -d \"
    done
    sleep 20
done
```

使用 AWS Panorama API 管理應用程式

您可以使用 AWS Panorama 圖像來監控和管理應用程式。

檢視

若要取得在設備上執行的應用程式清單，請使用 ListApplicationInstancesAPI。

```
$ aws panorama list-application-instances
    "ApplicationInstances": [
        {
            "Name": "aws-panorama-sample",
            "ApplicationInstanceId": "applicationInstance-ddaxxmpl2z7bg74ywutd7byxuq",
            "DefaultRuntimeContextDevice": "device-4tafxmplhtmzabv5lsacba4ere",
            "DefaultRuntimeContextDeviceName": "my-appliance",
            "Description": "command-line deploy",
            "Status": "DEPLOYMENT_SUCCEEDED",
            "HealthStatus": "RUNNING",
            "StatusDescription": "Application deployed successfully.",
            "CreatedTime": 1661902051.925,
            "Arn": "arn:aws:panorama:us-east-2:123456789012:applicationInstance/
applicationInstance-ddaxxmpl2z7bg74ywutd7byxuq",
            "Tags": {
                "client": "sample"
            }
        },
    ]
}
```

要獲取有關應用程序實例節點的更多詳細信息，請使用 ListApplicationInstanceNodeInstancesAPI。

```
$ aws panorama list-application-instance-node-instances --application-instance-id
 applicationInstance-ddaxxmpl2z7bg74ywutd7byxuq
{
    "NodeInstances": [
        {
            "NodeInstanceId": "code_node",
            "NodeId": "SAMPLE_CODE-1.0-fd3dxmpl-interface",
            "PackageName": "SAMPLE_CODE",
            "PackageVersion": "1.0",
            "PackagePatchVersion":
 "fd3dxmpl2bdfa41e6fe1be290a79dd2c29cf014eadf7416d861ce7715ad5e8a8",
            "NodeName": "interface",
```

```
            "CurrentStatus": "RUNNING"
        },
        {
            "NodeInstanceId": "camera_node_override",
            "NodeId": "warehouse-floor-1.0-9eabxmpl-warehouse-floor",
            "PackageName": "warehouse-floor",
            "PackageVersion": "1.0",
            "PackagePatchVersion":
 "9eabxmple89f0f8b2f2852cca2a6e7971aa38f1629a210d069045e83697e42a7",
            "NodeName": "warehouse-floor",
            "CurrentStatus": "RUNNING"
        },
        {
            "NodeInstanceId": "output_node",
            "NodeId": "hdmi_data_sink-1.0-9c23xmpl-hdmi0",
            "PackageName": "hdmi_data_sink",
            "PackageVersion": "1.0",
            "PackagePatchVersion":
 "9c23xmplc4c98b92baea4af676c8b16063d17945a3f6bd8f83f4ff5aa0d0b394",
            "NodeName": "hdmi0",
            "CurrentStatus": "RUNNING"
        },
        {
            "NodeInstanceId": "model_node",
            "NodeId": "SQUEEZENET_PYTORCH-1.0-5d3cabda-interface",
            "PackageName": "SQUEEZENET_PYTORCH",
            "PackageVersion": "1.0",
            "PackagePatchVersion":
 "5d3cxmplb7113faa1d130f97f619655d8ca12787c751851a0e155e50eb5e3e96",
            "NodeName": "interface",
            "CurrentStatus": "RUNNING"
        }
    ]
}
```

管理攝影機串流

您可以使用 SignalApplicationInstanceNodeInstancesAPI 暫停和繼續攝影機串流節點。

```
$ aws panorama signal-application-instance-node-instances --application-instance-id
 applicationInstance-ddaxxmpl2z7bg74ywutd7byxuq \
       --node-signals '[{"NodeInstanceId": "camera_node_override", "Signal":
 "PAUSE"}]'
```

```
{
    "ApplicationInstanceId": "applicationInstance-ddaxxmpl2z7bg74ywutd7byxuq"
}
```

在腳本中，您可以獲取節點列表，然後選擇一個節點以互動方式暫停或繼續。

Example pause-camera.sh — 用法

```
my-app$ ./pause-camera.sh

Getting nodes...
0: SAMPLE_CODE              RUNNING
1: warehouse-floor          RUNNING
2: hdmi_data_sink           RUNNING
3: entrance-north           PAUSED
4: SQUEEZENET_PYTORCH       RUNNING
Choose a node
1
Signalling node warehouse-floor
+ aws panorama signal-application-instance-node-instances --application-instance-id
 applicationInstance-r3a7xmplcbmpjqeds7vj4b6pjy --node-signals '[{"NodeInstanceId":
 "warehouse-floor", "Signal": "PAUSE"}]'
{
    "ApplicationInstanceId": "applicationInstance-r3a7xmplcbmpjqeds7vj4b6pjy"
}
```

透過暫停和恢復攝影機節點，您可以在比同時處理更多的攝影機串流中循環。為此，請將多個攝像機流映射到覆蓋清單中的相同輸入節點。

在下列範例中，覆寫資訊清單會對應兩個攝影機資料流，entrance-north以warehouse-floor及相同的輸入節點 (camera_node)。當應用程序啟動並且entrance-north節點等待信號打開時，warehouse-floor流處於活動狀態。

Example 覆蓋多點運行

```
    "nodeGraphOverrides": {
        "nodes": [
            {
                "name": "warehouse-floor",
                "interface": "123456789012::warehouse-floor.warehouse-floor",
                "launch": "onAppStart"
            },
```

```
            {
                "name": "entrance-north",
                "interface": "123456789012::entrance-north.entrance-north",
                "launch": "onSignal"
            },
        ...
        "packages": [
            {
                "name": "123456789012::warehouse-floor",
                "version": "1.0"
            },
            {
                "name": "123456789012::entrance-north",
                "version": "1.0"
            }
        ],
        "nodeOverrides": [
            {
                "replace": "camera_node",
                "with": [
                    {
                        "name": "warehouse-floor"
                    },
                    {
                        "name": "entrance-north"
                    }
                ]
            }
```

如需使用 API 部署的詳細資訊，請參閱自動化應用部署。

使用 VPC 端點

如果您在沒有網際網路存取權的 VPC 中工作，您可以建立 VPC 端點以與 AWS Panorama 搭配使用。VPC 端點可讓在私有子網路中執行的用戶端在沒有網際網路連線的情況下連線到 AWS 服務。

如需 AWS Panorama 設備使用的連接埠和端點的詳細資訊，請參閱???。

章節

- 建立一個 VPC 端點
- 將應用裝置連接至私有子網路
- AWS CloudFormation範例範本

建立一個 VPC 端點

若要在 VPC 和 AWS Panorama 之間建立私有連線，請建立 VPC 端點。使用 AWS Panorama 模式不需要 VPC 端點。只有在沒有網際網路存取權的 VPC 中工作時，才需要建立 VPC 端點。當 AWS CLI 或開發套件嘗試連線到 AWS Panorama 時，流量會透過 VPC 端點進行路由。

使用下列設定為 AWS Panorama 建立 VPC 端點：

- 服務名稱 — **com.amazonaws.*us-west-2*.panorama**
- 類型-接口

VPC 端點使用服務的 DNS 名稱從 AWS 開發套件用戶端取得流量，而無需任何其他設定。如需使用 VPC 端點的詳細資訊，請參閱 Amazon VPC 使用者指南中的介面虛擬私人雲端端點。

將應用裝置連接至私有子網路

AWS Panorama 設備可透AWS過私有 VPN 連線與AWS Site-to-Site VPN或連線AWS Direct Connect。透過這些服務，您可以建立延伸至資料中心的私有子網路。應用裝置會連線至私有子網路，並透過 VPC 端點存取AWS服務。

網 Site-to-Site VPN，AWS Direct Connect是用於將資料中心安全地連接到 Amazon VPC 的服務。透過 Site-to-Site VPN，您可以使用市售的網路裝置進行連線。 AWS Direct Connect使用設AWS備進行連接。

- 網 Site-to-Site VPN — 什麼是？AWS Site-to-Site VPN

- AWS Direct Connect— 什麼是AWS Direct Connect?

將區域網路連線至 VPC 中的私人子網路後，請為下列服務建立 VPC 端點。

- Amazon 簡單存儲服務 — AWS PrivateLink適用於 Amazon S3
- AWS IoT Core— AWS IoT Core與介面 VPC 端點 (資料平面和憑證提供者) 搭配使用
- Amazon 彈性容器註冊表 — Amazon 彈性容器註冊表界面 VPC 端點
- Amazon CloudWatch — CloudWatch 與接口 VPC 端點一起使用
- Amazon CloudWatch 日誌 — 使用 CloudWatch 日誌與界面 VPC 端點

設備不需要連線至 AWS Panorama 服務。它會透過中AWS IoT的簡訊管道與 AWS Panorama 進行通訊。

除了 VPC 端點之外，Amazon S3 還AWS IoT需要使用 Amazon 路線 53 私有託管區域。私有託管區域會將流量從子網域 (包括 Amazon S3 存取點和 MQTT 主題的子網域) 路由到正確的 VPC 端點。如需私有託管區域的相關資訊，請參閱 Amazon Route 53 開發人員指南中的使用私有託管區域。

如需具有 VPC 端點和私有託管區域的 VPC 組態範例，請參閱。AWS CloudFormation範例範本

AWS CloudFormation範例範本

本指南的 GitHub 儲存庫提供可用來建立資源以搭配 AWS Panorama 使用的AWS CloudFormation範本。這些範本會建立具有兩個私有子網路、一個公用子網路和一個 VPC 端點的 VPC。您可以使用 VPC 中的私有子網路來託管與網際網路隔離的資源。公共子網中的資源可以與私有資源進行通信，但私有資源無法從 Internet 訪問。

Example vpc-端點. Yml — 私有子網

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  vpc:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 172.31.0.0/16
      EnableDnsHostnames: true
      EnableDnsSupport: true
      Tags:
```

```
      - Key: Name
        Value: !Ref AWS::StackName
privateSubnetA:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref vpc
    AvailabilityZone:
      Fn::Select:
       - 0
       - Fn::GetAZs: ""
    CidrBlock: 172.31.3.0/24
    MapPublicIpOnLaunch: false
    Tags:
      - Key: Name
        Value: !Sub  ${AWS::StackName}-subnet-a
...
```

vpc-endpoint.yml範本顯示如何為 AWS Panorama 建立 VPC 端點。您可以使用此端點 AWS Panorama AWS 開發套件或AWS CLI.

Example 虛擬電腦端點. YML — 虛擬私人雲端端點

```
panoramaEndpoint:
  Type: AWS::EC2::VPCEndpoint
  Properties:
    ServiceName: !Sub com.amazonaws.${AWS::Region}.panorama
    VpcId: !Ref vpc
    VpcEndpointType: Interface
    SecurityGroupIds:
    - !GetAtt vpc.DefaultSecurityGroup
    PrivateDnsEnabled: true
    SubnetIds:
    - !Ref privateSubnetA
    - !Ref privateSubnetB
    PolicyDocument:
      Version: 2012-10-17
      Statement:
      - Effect: Allow
        Principal: "*"
        Action:
          - "panorama:*"
        Resource:
          - "*"
```

這PolicyDocument是以資源為基礎的權限原則，可定義可透過端點進行的 API 呼叫。您可以修改策略以限制可透過端點存取的動作和資源。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。

vpc-appliance.yml範本顯示如何為 AWS Panorama 設備使用的服務建立 VPC 端點和私有託管區域。

Example vpc 應用程式 .yml — 具有私有託管區域的 Amazon S3 存取點端點

```
  s3Endpoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: !Sub com.amazonaws.${AWS::Region}.s3
      VpcId: !Ref vpc
      VpcEndpointType: Interface
      SecurityGroupIds:
      - !GetAtt vpc.DefaultSecurityGroup
      PrivateDnsEnabled: false
      SubnetIds:
      - !Ref privateSubnetA
      - !Ref privateSubnetB
...
  s3apHostedZone:
    Type: AWS::Route53::HostedZone
    Properties:
      Name: !Sub s3-accesspoint.${AWS::Region}.amazonaws.com
      VPCs:
        - VPCId: !Ref vpc
          VPCRegion: !Ref AWS::Region
  s3apRecords:
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref s3apHostedZone
      Name: !Sub "*.s3-accesspoint.${AWS::Region}.amazonaws.com"
      Type: CNAME
      TTL: 600
      # first DNS entry, split on :, second value
      ResourceRecords:
      - !Select [1, !Split [":", !Select [0, !GetAtt s3Endpoint.DnsEntries ] ] ]
```

範例範本示範如何使用 VPC 範例建立 Amazon VPC 和 Route 53 資源。您可以移除 VPC 資源，並以資源的 ID 取代子網路、安全群組和 VPC ID 的參照，以符合您的使用案例。

範例應用程式、指令碼和範本

該 GitHub 本指南的儲存庫提供範例應用程式、指令碼和範本AWS Panorama裝置。使用這些範例來瞭解最佳實務並自動化開發工作流程。

章節

範例應用程式

示例應用程序演示使用AWS Panorama功能和常見的電腦視覺工作。這些範例應用程式包括可自動化設定和部署的指令碼和範本。只要使用最少的組態，您就可以從命令列部署和更新應用程式。

- aws-panorama-sample— 具有分類模型的基本計算機視覺。使用AWS SDK for Python (Boto)將量度上傳至 CloudWatch、儀器預處理和推論方法，以及設定記錄。
- 調試服務器—開啟輸入連接埠在設備上並將流量轉發到應用程序代碼容器。使用多執行緒同時執行應用程式程式碼、HTTP 伺服器和 HTTP 用戶端。
- 自訂模型— 從代碼導出模型並編譯 SageMaker Neo 來測試兼容性AWS Panorama設備。在 Python 開發、碼頭容器或亞馬遜 EC2 執行個體上本機建置。導出和編譯 Keras 中的所有內置應用程序模型 TensorFlow 或蟒蛇版本。

如需更多範例應用程式，請造訪aws-panorama-samples儲存庫。

實用程序腳

中的指令碼`util-scripts`目錄管理AWS Panorama資源或自動化開發工作流程。

- provision-device.sh— 提供設備。
- check-updates.sh— 檢查並套用設備軟體更新。

- reboot-device.sh— 重新啟動設備。
- register-camera.sh— 註冊相機.
- deregister-camera.sh— 刪除攝影機節點。
- view-logs.sh— 檢視應用程式執行個體的記錄檔。
- pause-camera.sh— 暫停或恢復相機流。
- push.sh— 建置、上傳和部署應用程式。
- rename-package.sh— 重命名節點封裝。更新目錄名稱、組態檔案和應用程式資訊清單。
- samplify.sh— 用示例帳戶 ID 替換您的帳戶 ID，並恢復備份配置以刪除本地配置。
- update-model-config.sh— 更新描述器檔案後，將模型重新新增至應用程式。
- cleanup-patches.sh— 取消註冊舊的修補程式版本，並從 Amazon S3 刪除其資訊清單。

有關使用詳情，請參閱自述文件。

AWS CloudFormation 範本

使用AWS CloudFormation中的範本`cloudformation-templates`要為其建立資源的目錄AWS Panorama應用程式。

- 警報應用程序 .yml— 建立警示，以監控應用程式是否有錯誤。如果應用程式執行個體發生錯誤或停止執行 5 分鐘，警示會傳送通知電子郵件。
- 警報設備。Yml— 創建監視設備連接的警報。如果裝置停止傳送指標 5 分鐘，警示會傳送通知電子郵件。
- 應用程序角色 .yml— 建立應用程式角色。角色包含將量度傳送至的權限 CloudWatch。針對您的應用程式使用的其他 API 作業，將權限新增至政策陳述式。
- 虛擬電腦應用程式. Yml— 創建具有私有子網服務訪問權限的 VPCAWS Panorama設備。若要將應用裝置連線至 VPC，請使用AWS Direct Connect或者AWS Site-to-Site VPN。
- 虛擬電腦端點. YML— 創建具有私有子網服務訪問權限的 VPCAWS Panorama服務。VPC 內部的資源可以連接到AWS Panorama監控和管理AWS Panorama無需連接到互聯網的資源。

該`create-stack.sh`此目錄中的指令碼會建立AWS CloudFormation堆疊。它需要可變數量的參數。第一個參數是模板的名稱，其餘參數是模板中參數的覆蓋。

例如，下列命令會建立應用程式角色。

```
$ ./create-stack.sh application-role
```

更多樣本和工具

該aws-panorama-samples存儲庫有更多示例應用程序和有用的工具。

- 應用— 適用於各種模型架構和使用案例的範例應用程式。
- 攝影機串流驗證— 驗證攝像機流.
- PanoJupyter— 運行 JupyterLab 在一個AWS Panorama設備。
- 側載— 更新應用程式程式碼，而不建置或部署應用程式容器。

該AWS社區還開發了工具和指導AWS Panorama。查看以下開源項目 GitHub。

- 餅乾全景。-一個餅乾模板。AWS Panorama應用程式。
- 背包— 用於訪問運行時環境詳細信息，配置文件和其他視頻輸出選項的 Python 模塊。

監控AWS Panorama資源和應用程式

您可以監控AWS Panorama中的資源AWS Panorama控制台和亞馬遜CloudWatch。所以此AWS Panorama裝置連接到AWS雲通過互聯網報告其狀態和連接攝像機的狀態。當它處於打開狀態時，設備還會將日誌發送到CloudWatch實時登錄。

設備獲得使用權限AWS IoT、CloudWatch日誌和其他 AWS 服務，您首次使用AWS Panorama主控台。如需詳細資訊，請參閱 AWS Panorama 服務角色和跨服務資源。

有關解決特定錯誤的幫助，請參閱疑難排解。

主題

- 在 AWS Panorama 控制台中進行監控
- 檢視 AWS Panorama 日誌
- 使用亞馬遜監控裝置和應用程序CloudWatch

在 AWS Panorama 控制台中進行監控

您可以使用 AWS Panorama 控制台監控您的 AWS Panorama 設備和攝像機。主控台使用AWS IoT以監視設備的狀態。

在 AWS Panorama 控制台中監視您的設備

1. 開啟AWS Panorama 主控台。
2. 打開 AWS Panorama 主控台「設備」頁面。
3. 選擇一個裝置。
4. 若要查看應用程式實例的狀態，請在列表中選擇應用程式實例的狀態。
5. 要查看裝置網絡接口的狀態，請選擇設定。

裝置的整體狀態會顯示在頁面上方。如果狀態為線上，則設備將連接到AWS並定期發送狀態更新。

檢視 AWS Panorama 日誌

AWS Panorama 向亞馬遜報告應用程式和系統事件 CloudWatch 日誌。遇到問題時，您可以使用事件日誌協助偵錯 AWS Panorama 應用程式或對應用程式的組態進行故障診斷。

若要檢視日誌 CloudWatch 日誌

1. 開啟的日誌群組頁面 CloudWatch 日誌主控台。
2. 在下列群組中尋找 AWS Panorama 應用程式和設備日誌：

   - 裝置記錄–`/aws/panorama/devices/device-id`
   - 應用程式記錄–`/aws/panorama/devices/device-id/applications/instance-id`

更新系統軟體後重新佈建設備時，也可以檢視佈建 USB 磁碟機上的日誌。

章節

檢視裝置日誌

AWS Panorama 設備會為裝置建立日誌群組，並為您部署的每個應用程式執行個體建立一個群組。裝置記錄包含應用程式狀態、軟體升級和系統組態的相關資訊。

裝置記錄 —`/aws/panorama/devices/device-id`

- `occ_log`— 來自控制器進程的輸出。此程序會協調應用程式部署，並報告每個應用程式執行個體節點的狀態。
- `ota_log`— 從坐標的過程輸出 over-the-air (OTA) 軟體升級。
- `syslog`— 從設備的 syslog 進程輸出，該過程捕獲進程之間發送的消息。
- `kern_log`— 來自設備 Linux 內核的事件。
- `logging_setup_logs`— 從配置的過程輸出 CloudWatch 日誌代理程式。

- `cloudwatch_agent_logs`— 從輸出 CloudWatch 日誌代理程式。
- `shadow_log`— 從輸出AWS IoTDevice Shadow。

檢視應用程式日誌

應用程式執行個體的日誌群組包含以節點命名的每個節點的日誌串流。

應用程序日誌-**/aws/panorama/devices/*device-id*/applications/*instance-id***

- Code— 從應用程式程式碼和 AWS Panorama 應用程式開發套件輸出。從彙總應用程式記錄檔/`opt/aws/panorama/logs`。
- 模型— 使用模型協調推論請求的程序輸出。
- 串流— 從解碼攝像機流中的視頻的過程輸出。
- 顯示— 從渲染 HDMI 端口視頻輸出的過程中輸出。
- `mds`— 記錄來自應用裝置中繼資料伺服器。
- `console_output`— 從程式碼容器擷取標準輸出和錯誤串流。

如果未在中看到日誌 CloudWatch 日誌，確認您位於正確的 AWS 區域。如果是，則設備與 AWS 的連線可能發生問題，或在許可該設備的AWS Identity and Access Management(IAM) 角色。

設定應用程式日誌

設定 Python 記錄器以將日誌寫入`/opt/aws/panorama/logs`。應用裝置會將記錄從此位置串流至 CloudWatch 日誌。若要避免使用過多的磁碟空間，請使用 10 MiB 的檔案大小上限，備份計數為 1。下列範例會示範建立日誌的方法。

Example application.py— 記錄器組態

```
def get_logger(name=__name__,level=logging.INFO):
    logger = logging.getLogger(name)
    logger.setLevel(level)
    LOG_PATH = '/opt/aws/panorama/logs'
    handler = RotatingFileHandler("{}/app.log".format(LOG_PATH), maxBytes=10000000,
 backupCount=1)
    formatter = logging.Formatter(fmt='%(asctime)s %(levelname)-8s %(message)s',
                                  datefmt='%Y-%m-%d %H:%M:%S')
    handler.setFormatter(formatter)
```

```
    logger.addHandler(handler)
    return logger
```

在全局範圍初始化記錄器，並在整個應用程序代碼中使用它。

Example application.py— 初始化記錄器

```
def main():
    try:
        logger.info("INITIALIZING APPLICATION")
        app = Application()
        logger.info("PROCESSING STREAMS")
        while True:
            app.process_streams()
            # turn off debug logging after 150 loops
            if logger.getEffectiveLevel() == logging.DEBUG and app.frame_num == 150:
                logger.setLevel(logging.INFO)
    except:
        logger.exception('Exception during processing loop.')

logger = get_logger(level=logging.INFO)
main()
```

檢視佈建日誌

在佈建期間，AWS Panorama 設備會將日誌複製到您用來將組態存檔傳輸到設備的 USB 磁碟機。使用這些記錄可疑難排解使用最新軟體版本之應用裝置的佈建問題。

> **Important**
>
> 佈建記錄檔適用於更新至軟體 4.3.23 版或更新版本的應用裝置。

應用程式記錄

- `/panorama/occ.log`— AWS Panorama 控制器軟體日誌。
- `/panorama/ota_agent.log`— AWS Panorama over-the-air 更新代理程式記錄檔。
- `/panorama/syslog.log`— 系統日誌.
- `/panorama/kern.log`— 核心記錄檔。

從裝置輸出記錄檔

如果您的裝置和應用程式記錄未出現在 CloudWatch 日誌，您可以使用 USB 驅動器從設備中獲取加密的日誌圖像。AWS Panorama 服務團隊可以代表您解密日誌並協助進行除錯。

先決條件

要按照該過程，您將需要以下硬件：

- USB 隨身碟— 具有至少 1 GB 儲存空間的 FAT32 格式 USB 快閃記憶體磁碟機，用於從 AWS Panorama 設備傳輸記錄檔。

從裝置輸出記錄

1. 準備一個帶有一個 USB 驅動器`managed_logs`文件夾, 裡面, `apanorama`資料夾。

   ```
   /
    ### panorama
        ### managed_logs
   ```

2. 將 USB 驅動器 Connect 到設備。
3. Power offAWS Panorama 設備。
4. 開啟 AWS Panorama 設備的電源。
5. 裝置會將記錄檔複製到裝置。狀態指示燈閃爍藍色雖然這正在進行。
6. 然後可以在裡面找到日誌文件`managed_logs`具有格式的目錄`panorama_device_log_`*`v1_dd_hh_mm`*`.img`

您無法自己解密日誌映像。與客戶支援、AWS Panorama 技術客戶經理或解決方案架構師合作，與服務團隊協調。

使用亞馬遜監控裝置和應用程序CloudWatch

當設備在線時，AWS Panorama 會向亞馬遜發送指標CloudWatch。您可以使用這些指標構建圖表和儀錶板，請參閱CloudWatch控制台監視裝置活動，並設置警報，以便在設備脫機或應用程序遇到錯誤時通知您。

在 CloudWatch 主控台上檢視指標

1. 開啟AWS Panorama 控制台指標頁(PanoramaDeviceMetrics命名空間）。
2. 選擇一個維度架構。
3. 選擇指標以將其新增至圖表。
4. 若要選擇不同的統計資料並自訂圖表，請使用 Graphed metrics (圖表化指標) 標籤中的選項。根據預設，圖表會針對所有指標使用 Average 統計資料。

> **定價**
>
> CloudWatch具有始終免費套餐。除了免費方案閥值之外，CloudWatch 還會收取指標、儀表板、警示、記錄和洞見的費用。如需詳細資訊，請參閱 CloudWatch 定價。

如需有關 的詳細資訊CloudWatch，請參亞馬遜CloudWatch使用者指南。

章節

使用設備指標

當裝置在線時，它會傳送指標至亞馬遜CloudWatch。您可以使用這些指標來監控設備活動，並在設備脫機時觸發警報。

- DeviceActive— 當設備處於活動狀態時定期發送。

  維度 —DeviceId和DeviceName。

檢視DeviceActive中的指標Average統計資料。

使用應用程序指標

當應用程序遇到錯誤時，它會傳送指標至亞馬遜CloudWatch。您可以利用這些指標在應用程序停止運行時觸發警示。

- ApplicationErrors— 記錄的應用程序錯誤數。

  維度 —ApplicationInstanceName和ApplicationInstanceId。

通過Sum統計資料。

設定警示

要在指標超過閾值時獲取通知，請創建警報。例如，您可以建立警示，在ApplicationErrors指標保持為 1，持續 20 分鐘。

欲建立警示

1. 開啟亞馬遜CloudWatch主控台警示頁面。
2. 選擇 Create alarm (建立警示)。
3. 選擇選擇指標並找到設備的指標，例如ApplicationErrors為了applicationInstance-gk75xmplqbqtenlnmz4ehiu7xa、my-application。
4. 按照說明配置警報的條件、操作和名稱。

如需詳細說明，請參閱。建立CloudWatch警報中的亞馬遜CloudWatch使用者指南。

疑難排解

下列主題針對您在使用主AWS Panorama控台、應用裝置或 SDK 時可能遇到的錯誤和問題提供疑難排解建議。如果您發現此處未列出的問題，請使用此頁面上的 [提供意見反應] 按鈕來回報問題。

您可以在 Amazon 日誌主控台中找到設備的 CloudWatch 日誌。設備會從您的應用程式程式碼、設備軟體以及產生AWS IoT程序時上傳記錄。如需詳細資訊，請參閱檢視 AWS Panorama 日誌。

佈建

問題：(macOS) 我的電腦無法辨識隨附的 USB 隨身碟 (搭配 USB-C 轉接器)。

如果您將 USB 磁碟機插入已連接至電腦的 USB-C 轉接器，就會發生這種情況。嘗試斷開適配器的連接，然後將其與已連接的 USB 驅動器重新連接。

問題：當我使用自己的 USB 磁碟機時，佈建失敗。

問題：使用應用裝置的 USB 2.0 連接埠時，佈建失敗。

本設AWS Panorama備與介於 1 至 32 GB 之間的 USB 快閃記憶體裝置相容，但並非所有裝置都相容。使用 USB 2.0 連接埠進行佈建時，已觀察到一些問題。若要獲得一致的結果，請使用隨附的 USB 隨身碟搭配 USB 3.0 連接埠 (HDMI 連接埠旁)。

對於聯想 ThinkEdge® SE70，設備不隨附 USB 磁碟機。使用至少具有 1 GB 儲存空間的 USB 3.0 磁碟機。

設備配置

問題：設備在開機期間顯示空白畫面。

完成初始開機順序 (大約需要一分鐘) 後，設備載入模型並啟動應用程式時，會顯示一分鐘或更長時間的空白螢幕。此外，如果設備在開啟後連接顯示器，則不會輸出視訊。

問題：當我按住電源按鈕將其關閉時，設備沒有響應。

設備最多需要 10 秒才能安全關閉。您只需按住電源按鈕 1 秒鐘即可啟動關機順序。如需按鈕作業的完整清單，請參閱AWS Panorama 設備按鈕和指示燈。

問題：我需要產生新的組態封存檔來變更設定或取代遺失的憑證。

AWS Panorama下載裝置憑證或網路組態後，不會儲存裝置憑證或網路組態，也無法重複使用設定封存。使用AWS Panorama主控台刪除設備，並使用新的組態歸檔建立新設備。

應用程式組態

問題：當我執行多個應用程式時，我無法控制使用 HDMI 輸出的應用程式。

當您部署多個具有輸出節點的應用程式時，最近啟動的應用程式會使用 HDMI 輸出。如果此應用程序停止運行，則另一個應用程序可以使用輸出。若只要提供一個應用程式存取輸出，請從其他應用程式的應用程式資訊清單中移除輸出節點和對應邊緣，然後重新部署。

問題：應用程式輸出未出現在記錄中

配置 Python 記錄器以將日誌文件寫入`/opt/aws/panorama/logs`。這些會擷取在程式碼容器節點的記錄串流中。標準輸出和錯誤串流會擷取在稱為的個別記錄資料流中`console-output`。如果您使用`print`，請使用該`flush=True`選項來防止郵件卡在輸出緩衝區中。

錯誤：You've reached the maximum number of versions for package SAMPLE_CODE. Deregister unused package versions and try again.

來源：AWS Panorama服務

每次將變更部署到應用程式時，都會註冊一個修補程式版本，該修補程式版本代表其使用的每個套件的套件組態和資產檔案。使用清理修補程式指令碼取消註冊未使用的修補程式版本

攝影機串流

錯誤：liveMedia0: Failed to get SDP description: Connection to server failed: Connection timed out (-115)

錯誤：liveMedia0: Failed to get SDP description: 404 Not Found; with the result code: 404

錯誤：liveMedia0: Failed to get SDP description: DESCRIBE send() failed: Broken pipe; with the result code: -32

來源：攝像機節點日誌

設備無法連接到應用程序的攝像頭流。發生這種情況時，當應用程式等待應用程式 SDK 中的視訊影格時，視訊輸出為空白或凍結在上次處理的AWS Panorama影格上。設備軟體嘗試連線至攝影機串流，

並在攝影機節點記錄檔中記錄逾時錯誤。確認您的攝影機串流 URL 是否正確，且 RTSP 流量可在網路中的攝影機和設備之間路由傳送。如需詳細資訊，請參閱將 AWS Panorama 設備連接到您的網路。

錯誤：ERROR finalizeInterface(35) Camera credential fetching for port [username] failed

來源：OCC 日誌

找不到相機流憑據的AWS Secrets Manager秘密。刪除相機流並重新創建它。

錯誤：Camera did not provide an H264 encoded stream

來源：攝像機節點日誌

攝像機流具有 H.264 以外的編碼，例如 H.265。使用 H.264 攝影機串流重新部署應用程式。如需支援相機的詳細資訊，請參閱支援的相機。

AWS Panorama

雲端安全是 AWS 最重視的一環。身為 AWS 客戶的您，將能從資料中心和網路架構的建置中獲益，以滿足組織最為敏感的安全要求。

安全是 AWS 與您共同的責任。共同的責任模型 將此描述為雲端 本身 的安全和雲端 內部 的安全：

- 雲端本身的安全：AWS 負責保護在 AWS Cloud 中執行 AWS 服務的基礎設施。AWS 也提供您可安全使用的服務。在 AWS 合規計劃中，第三方稽核員會定期測試並驗證我們的安全功效。若要了解適用於 AWS Panorama 的合規計劃，請參閱AWS合規計劃的服務範圍。
- 雲端內部的安全：您的責任取決於所使用的 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。

本文件有助於您了解如何在使用 AWS Panorama 時套用共同責任模型。下列主題將示範如何將 AWS Panorama 設定為滿足您的安全與合規目標。您也將了解如何使用其他 AWS 服務來協助監控並保護 AWS Panorama 資源。

主題

AWS Panorama 設備安全功能

為了保護您的應用程序, 模型和硬件抵禦惡意代碼和其他漏洞，AWS Panorama 設備實現了一系列廣泛的安全功能。其中包括但不限於下列項目。

- 全磁碟加密— 設備實施 Linux 統一密鑰設置 (LUKS2) 全磁盤加密。所有系統軟件和應用程序數據都使用特定於您的設備的密鑰進行加密。即使對設備進行物理訪問，攻擊者也無法檢查其存儲內容。
- 存儲器佈建— 為防止針對加載到內存中的可執行代碼的攻擊，AWS Panorama 設備使用地址空間佈局隨機化 (ASLR)。ASLR 隨機化操作系統代碼加載到內存中時的位置。這樣可以防止使用利用漏洞，通過預測代碼在運行時的存儲位置來覆蓋或運行代碼的特定部分。
- 可信執行環境— 設備使用基於 ARM TrustZone 的受信任執行環境 (TEE)，具有隔離的存儲、內存和處理資源。存儲在信任區域中的密鑰和其他敏感數據只能由受信任的應用程序訪問，該應用程序在 TEE 中的獨立操作系統中運行。AWS Panorama 設備軟件與應用程序代碼一起運行在不受信任的 Linux 環境中。它只能通過向安全應用程序發出請求來訪問加密操作。
- 安全佈建— 置備設備時，您傳輸到設備的憑據（密鑰、證書和其他加密材料）僅在短時間內有效。設備使用短期憑據連接到AWS IoT，併為自己請求一個有效期較長的證書。AWS Panorama 服務生成證書，並使用設備上硬編碼的密鑰對證書進行加密。只有請求證書的設備才能對其進行解密並與 AWS Panorama 進行通信。
- 安全啟動— 當設備啟動時，每個軟件組件在運行之前都會進行身份驗證。引導 ROM 是處理器中硬編碼的軟件，無法修改，它使用硬編碼的加密密鑰來解密引導加載程序，以驗證受信任的執行環境內核等。
- 簽名核心— 內核模塊使用非對稱加密密鑰進行簽名。操作系統內核使用公鑰解密簽名，並在將模塊加載到內存之前驗證它是否匹配模塊的簽名。
- DM-真實性— 與驗證內核模塊的方式類似，設備使用 Linux 設備映射器的`dm-verity`功能，先驗證其完整性，然後再次安裝設備軟件映像。如果修改了裝置軟件，則無法運行。
- 轉返組態— 當您更新裝置軟件時，裝置會在 SoC 上吹一個電子保險絲（芯片上的系統）。每個軟件版本都希望吹出越來越多的保險絲，如果更多的保險絲被吹，則無法運行。

AWS Panorama 設備安全最佳實務

使用 AWS Panorama 設備時，請記住以下最佳實踐。

- 物理上保護設備— 將設備安裝在封閉的服務器機架或安全機房中。限制授權人員對設備的物理訪問權限。
- 保護設備的網絡連接— 將設備 Connect 到限制對內部和外部資源訪問的路由器。設備需要連接到攝像機，攝像機可以位於安全的內部網絡上。它還需要連接到AWS。僅將第二個以太網端口用於物理冗餘，並將路由器配置為僅允許所需的流量。

  使用推薦的網絡配置之一來規劃網絡佈局。如需詳細資訊，請參閱 將 AWS Panorama 設備連接到您的網路。
- 格式化 USB 驅動器— 置備設備後，卸下 USB 驅動器並對其進行格式化。設備在向 AWS Panorama 服務註冊後不使用 USB 驅動器。格式化驅動器以刪除臨時憑證、配置文件和預配日誌。
- 使設備保持最新狀態— 及時應用裝置軟件更新。當您在 AWS Panorama 控制台中查看設備時，控制台將通知您是否有軟件更新。如需詳細資訊，請參閱 管理 AWS Panorama 設備。

  使用DescribeDeviceAPI 操作，您可以通過比較`LatestSoftware`和`CurrentSoftware`和 欄位之間沒有任何差異。當最新軟件版本與當前版本不同時，請使用控制台或使用為設備創建作業operation.
- 如果停止使用設備，請將其重置— 在將設備移出安全數據中心之前，請完全重置它。關閉設備電源並插入電源後，同時按住電源和復位按鈕 5 秒鐘。這會從設備中刪除帳户憑據、應用程序和日誌。

  如需詳細資訊，請參閱 AWS Panorama 設備按鈕和指示燈。
- 限制對 AWS Panorama 和其他 AWS 服務的訪問—可 AWSPanoramaFullAccess提供對所有 AWS Panorama API 操作的訪問權限，並在必要時訪問其他服務。在可能的情況下，該策略會根據命名約定限制對資源的訪問。例如，它提供了AWS Secrets Manager名稱開頭的祕密`panorama`。對於需要只讀訪問權限或訪問更具體資源集的用户，請使用託管策略作為最小權限策略的起點。

  如需詳細資訊，請參閱AWS Panorama 政策。

AWS Panorama 中的資料保護

AWS共同責任模型適用於 AWS Panorama 中的資料保護。如此模型所述，AWS 負責保護執行所有 AWS 雲端 的全球基礎設施。您負責維護在此基礎設施上託管內容的控制權。您也必須負責您所使用 AWS 服務 的安全組態和管理任務。如需有關資料隱私權的更多相關資訊，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，建議您使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 保護 AWS 帳戶 憑證，並設定個人使用者。如此一來，每個使用者都只會獲得授與完成其任務所必須的許可。我們也建議您採用下列方式保護資料：

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源通訊。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 使用 AWS CloudTrail 設定 API 和使用者活動日誌記錄。
- 使用 AWS 加密解決方案，以及 AWS 服務 內的所有預設安全控制項。
- 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 AWS 時，需要 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 Name (名稱) 欄位。這包括當您使用主控台、API 或AWS開發套件AWS 服務使用 AWS Panorama 或其他工作時。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。

章節

- 傳輸中加密
- AWS Panorama 設備
- 應用程式
- 其他服務

傳輸中加密

AWS Panorama API 端點僅支援透過 HTTPS 進行的安全連線。當您使用 AWS 開發套件或 AWS Panorama API 管理 AWS Panorama 資源時，所有通訊都會使用傳輸層安全性 (TLS) 加密。AWS Management ConsoleAWS Panorama 設備和 AWS 之間的通訊也會使用 TLS 加密。AWS Panorama 設備和透過 RTSP 的攝影機之間的通訊未加密。

如需 API 端點的完整清單，請參閱 AWS 一般參考.

AWS Panorama 設備

AWS Panorama 設備具有用於乙太網路、HDMI 視訊和 USB 儲存裝置的實體連接埠。SD 卡插槽，無線網絡連接和藍牙不可用。USB 連接埠僅在佈建期間使用，將組態歸檔傳輸至應用裝置。

組態歸檔的內容 (包括應用裝置的佈建憑證和網路組態) 未加密。AWS Panorama 不會儲存這些檔案；只有在註冊設備時才能擷取這些檔案。將設定歸檔傳輸到設備後，將其從電腦和 USB 儲存裝置中刪除。

應用裝置的整個檔案系統都已加密。此外，此應用裝置還套用多種系統層級保護，包括必要軟體更新的回復保護、已簽署的核心和開機載入程式，以及軟體完整性驗證。

停止使用設備時，請執行完全重設以刪除應用程式資料並重設設備軟體。

應用程式

您可以控制部署到應用裝置的程式碼。在部署所有應用程式程式碼之前，先驗證安全性問題，不論其來源為何。如果您在應用程式中使用協力廠商程式庫，請仔細考慮這些程式庫的授權和支援原則。

應用程式 CPU、記憶體和磁碟使用率不受設備軟體的限制。使用太多資源的應用程式可能會對其他應用程式和裝置的作業產生負面影響。在合併或部署到生產環境之前，請分別測試應用程式

應用程式資產 (代碼和模型) 不會與帳戶、設備或建置環境中的存取隔離。AWS Panorama 應用程式 CLI 產生的容器映像和模型存檔不會加密。針對生產工作負載使用個別帳戶，並且僅允許視需要進行存取。

其他服務

為了將您的模型和應用程式容器安全地存放在 Amazon S3 中，AWS Panorama 使用伺服器端加密搭配 Amazon S3 管理的金鑰。如需詳細資訊，請參閱 Amazon 簡單儲存服務使用者指南中的使用加密保護資料。

攝影機串流認證會在靜態時加密AWS Secrets Manager。設備的 IAM 角色授予其擷取密碼的權限，以便存取串流的使用者名稱和密碼。

AWS Panorama 設備會將日誌資料傳送到 Amazon CloudWatch 日誌。 CloudWatch 記錄檔預設會加密此資料，而且可設定為使用客戶管理的金鑰。如需詳細資訊，請參閱 Amazon CloudWatch 日誌使用者指南中的使用加密 CloudWatch 日誌AWS KMS中的日誌資料。

AWS Panorama 的身分和存取管理

AWS Identity and Access Management (IAM) 是一種 AWS 服務，讓管理員能夠安全地控制對 AWS 資源的存取權。IAM 管理員控制哪些人可以通過身份驗證 (登入) 和授權 (具有許可) 來使用 AWS Panorama 資源。IAM 是一種您可以免費使用的 AWS 服務。

主題

物件

您的使用方式 AWS Identity and Access Management (IAM) 會因您在 AWS Panorama 中所做的工作而有所不同。

服務使用者 — 如果您使用 AWS Panorama 服務執行工作，則管理員會為您提供所需的登入資料和許可。當您使用更多 AWS Panorama 功能完成工作時，您可能需要額外的許可。瞭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 AWS Panorama 中的功能，請參閱AWS Panorama 身分和存取疑難排解。

服務管理員 — 如果您負責公司的 AWS Panorama 資源，您可能擁有 AWS Panorama 的完整存取權。您的任務是判斷服務使用者應存取哪些 AWS Panorama 功能和資源。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。若要進一步了解貴公司如何將 IAM 與 AWS Panorama 搭配使用，請參閱AWS Panorama 如何與 IAM 搭配使用。

IAM 管理員 — 如果您是 IAM 管理員，您可能想要了解如何撰寫政策以管理 AWS Panorama 存取權的詳細資訊。若要檢視可在 IAM 中使用的 AWS Panorama 身分型政策範例，請參閱。AWS Panorama 身分型政策範例

使用身分驗證

身分驗證是使用身分憑證登入 AWS 的方式。您必須以 AWS 帳戶根使用者、IAM 使用者身分，或擔任 IAM 角色進行 驗證 (登入至 AWS)。

您可以使用透過身分來源 AWS IAM Identity Center 提供的憑證，以聯合身分登入 AWS。(IAM Identity Center) 使用者、貴公司的單一登入身分驗證和您的 Google 或 Facebook 憑證都是聯合身分的範例。您以聯合身分登入時，您的管理員先前已設定使用 IAM 角色的聯合身分。您 AWS 藉由使用聯合進行存取時，您會間接擔任角色。

根據您的使用者類型，您可以登入 AWS Management Console 或 AWS 存取入口網站。如需有關登入至 AWS 的詳細資訊，請參閱《AWS 登入 使用者指南》中的如何登入您的 AWS 帳戶。

如果您是以程式設計的方式存取 AWS，AWS 提供軟體開發套件 (SDK) 和命令列介面 (CLI)，以便使用您的憑證透過密碼編譯方式簽署您的請求。如果您不使用 AWS 工具，您必須自行簽署請求。如需使用建議的方法自行簽署請求的詳細資訊，請參閱《IAM 使用者指南》中的簽署 AWS API 請求。

無論您使用何種身分驗證方法，您可能都需要提供額外的安全性資訊。例如，AWS 建議您使用多重要素驗證 (MFA) 以提高帳戶的安全。如需更多資訊，請參閱《AWS IAM Identity Center 使用者指南》中的多重要素驗證和《IAM 使用者指南》中的在 AWS 中使用多重要素驗證 (MFA)。

AWS 帳戶 根使用者

如果是建立 AWS 帳戶，您會先有一個登入身分，可以完整存取帳戶中所有 AWS 服務 與資源。此身分稱為 AWS 帳戶 根使用者，使用建立帳戶時所使用的電子郵件地址和密碼即可登入並存取。強烈建議您不要以根使用者處理日常作業。保護您的根使用者憑證，並將其用來執行只能由根使用者執行的任務。如需這些任務的完整清單，了解需以根使用者登入的任務，請參閱《IAM 使用者指南》中的需要根使用者憑證的任務。

IAM 使用者和群組

IAM 使用者是您 AWS 帳戶 中的一種身分，具備單一人員或應用程式的特定許可。建議您盡可能依賴暫時憑證，而不是擁有建立長期憑證 (例如密碼和存取金鑰) 的 IAM 使用者。但是如果特定使用案例需要擁有長期憑證的 IAM 使用者，建議您輪換存取金鑰。如需詳細資訊，請參閱《IAM 使用者指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#rotate-credentials中的為需要長期憑證的使用案例定期輪換存取金鑰。

IAM 群組是一種指定 IAM 使用者集合的身分。您無法以群組身分登入。您可以使用群組來一次為多名使用者指定許可。群組可讓管理大量使用者許可的過程變得更為容易。例如，您可以擁有一個名為 IAMAdmins 的群組，並給予該群組管理 IAM 資源的許可。

使用者與角色不同。使用者只會與單一人員或應用程式建立關聯，但角色的目的是在由任何需要它的人員取得。使用者擁有永久的長期憑證，但角色僅提供暫時憑證。如需進一步了解，請參閱《IAM 使用者指南》中的建立 IAM 使用者 (而非角色) 的時機。

IAM 角色

IAM 角色是您 AWS 帳戶 中的一種身分，具備特定許可。它類似 IAM 使用者，但不與特定的人員相關聯。您可以在 AWS Management Console 中透過切換角色來暫時取得 IAM 角色。您可以透過呼叫 AWS CLI 或 AWS API 操作，或是使用自訂 URL 來取得角色。如需使用角色的方法詳細資訊，請參閱《IAM 使用者指南》中的使用 IAM 角色。

使用暫時憑證的 IAM 角色在下列情況中非常有用：

- 聯合身分使用者存取 — 如需向聯合身分指派許可，請建立角色，並為角色定義許可。當聯合身分進行身分驗證時，該身分會與角色建立關聯，並取得由角色定義的許可。如需有關聯合角色的詳細資訊，請參閱《IAM 使用者指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_create_for-idp.html中的為第三方身分供應商建立角色。如果您使用 IAM Identity Center，則需要設定許可集。為控制身分驗證後可以存取的內容，IAM Identity Center 將許可集與 IAM 中的角色相關聯。如需有關許可集的資訊，請參閱《AWS IAM Identity Center 使用者指南》中的許可集。
- 暫時 IAM 使用者許可 – IAM 使用者或角色可以擔任 IAM 角色來暫時針對特定任務採用不同的許可。
- 跨帳戶存取權 – 您可以使用 IAM 角色，允許不同帳戶中的某人 (信任的委託人) 存取您帳戶中的資源。角色是授予跨帳戶存取權的主要方式。但是，針對某些 AWS 服務，您可以將政策直接連接到資源 (而非使用角色作為代理)。如需了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱《IAM 使用者指南》中的 IAM 角色與資源類型政策的差異。
- 跨服務存取 – 有些 AWS 服務 會使用其他 AWS 服務 中的功能。例如，當您在服務中進行呼叫時，該服務通常會在 Amazon EC2 中執行應用程式或將物件儲存在 Amazon Simple Storage Service (Amazon S3) 中。服務可能會使用呼叫主體的許可、使用服務角色或使用服務連結角色來執行此作業。
  - 轉發存取工作階段 (FAS)：當您使用 IAM 使用者或角色在 AWS 中執行動作時，系統會將您視為主體。當您使用某些服務時，您可能會執行一個動作，而該動作之後會在不同的服務中啟動另一個動作。FAS 使用主體的許可呼叫 AWS 服務，搭配請求 AWS 服務 以向下游服務發出請求。只有在服務收到需要與其他 AWS 服務 或資源互動才能完成的請求之後，才會提出 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》。

- 服務角色：服務角色是服務擔任的 IAM 角色，可代表您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的建立角色以委派許可給 AWS 服務 服務。
- 服務連結角色 – 服務連結角色是一種連結到 AWS 服務 的服務角色類型。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 AWS 帳戶 中，並由該服務所擁有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

- 在 Amazon EC2 上執行的應用程式 – 針對在 EC2 執行個體上執行並提出 AWS CLI 和 AWS API 請求的應用程式，您可以使用 IAM 角色來管理暫時憑證。這是在 EC2 執行個體內儲存存取金鑰的較好方式。如需指派 AWS 角色給 EC2 執行個體並提供其所有應用程式使用，您可以建立連接到執行個體的執行個體設定檔。執行個體設定檔包含該角色，並且可讓 EC2 執行個體上執行的程式取得暫時憑證。如需詳細資訊，請參閱《IAM 使用者指南》中的利用 IAM 角色來授予許可給 Amazon EC2 執行個體上執行的應用程式。

如需了解是否要使用 IAM 角色或 IAM 使用者，請參閱《IAM 使用者指南》中的建立 IAM 角色 (而非使用者) 的時機。

使用政策管理存取權

您可以透過建立政策並將其附加到 AWS 身分或資源，在 AWS 中控制存取。政策是 AWS 中的一個物件，當其和身分或資源建立關聯時，便可定義其許可。AWS 會在主體 (使用者、根使用者或角色工作階段) 發出請求時評估這些政策。政策中的許可，決定是否允許或拒絕請求。大部分政策以 JSON 文件形式儲存在 AWS 中。如需 JSON 政策文件結構和內容的詳細資訊，請參閱《IAM 使用者指南》中的 JSON 政策概觀。

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。

預設情況下，使用者和角色沒有許可。若要授與使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者便能擔任這些角色。

IAM 政策定義該動作的許可，無論您使用何種方法來執行操作。例如，假設您有一個允許 `iam:GetRole` 動作的政策。具備該政策的使用者便可以從 AWS Management Console、AWS CLI 或 AWS API 取得角色資訊。

身分型政策

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。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。

身分型政策可進一步分類成內嵌政策或受管政策。內嵌政策會直接內嵌到單一使用者、群組或角色。受管政策則是獨立的政策，您可以將這些政策附加到 AWS 帳戶 中的多個使用者、群組和角色。受管政策包含 AWS 管理政策和客戶管理政策。如需瞭解如何在受管政策及內嵌政策間選擇，請參閱 IAM 使用者指南中的在受管政策和內嵌政策間選擇。

資源型政策

資源型政策是連接到資源的 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。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。主體可以包括帳戶、使用者、角色、聯合身分使用者或 AWS 服務。

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

存取控制清單 (ACL)

存取控制清單 (ACL) 可控制哪些委託人 (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。

Amazon Simple Storage Service (Amazon S3)、AWS WAF 和 Amazon VPC 是支援 ACL 的服務範例。若要進一步了解 ACL，請參閱《Amazon Simple Storage Service 開發人員指南》中的存取控制清單 (ACL) 概觀。

其他政策類型

AWS 支援其他較少見的政策類型。這些政策類型可設定較常見政策類型授與您的最大許可。

- 許可界限 – 許可範圍是一種進階功能，可供您設定身分型政策能授予 IAM 實體 (IAM 使用者或角色) 的最大許可。您可以為實體設定許可界限。所產生的許可會是實體的身分型政策和其許可界限的交集。會在 `Principal` 欄位中指定使用者或角色的資源型政策則不會受到許可界限限制。所有這類政策中的明確拒絕都會覆寫該允許。如需許可範圍的更多相關資訊，請參閱《IAM 使用者指南》中的 IAM 實體許可範圍。

- 服務控制政策 (SCP) – SCP 是 JSON 政策，可指定 AWS Organizations 中組織或組織單位 (OU) 的最大許可。AWS Organizations 服務可用來分組和集中管理您企業所擁有的多個 AWS 帳戶。若您啟用組織中的所有功能，您可以將服務控制政策 (SCP) 套用到任何或所有帳戶。SCP 會限制成員帳戶中實體的許可，包括每個 AWS 帳戶根使用者。如需組織和 SCP 的更多相關資訊，請參閱《AWS Organizations 使用者指南》中的 SCP 運作方式。
- 工作階段政策 – 工作階段政策是一種進階政策，您可以在透過編寫程式的方式建立角色或聯合使用者的暫時工作階段時，作為參數傳遞。所產生工作階段的許可會是使用者或角色的身分型政策和工作階段政策的交集。許可也可以來自資源型政策。所有這類政策中的明確拒絕都會覆寫該允許。如需更多資訊，請參閱《IAM 使用者指南》中的工作階段政策。

多種政策類型

將多種政策類型套用到請求時，其結果形成的許可會更為複雜、更加難以理解。如需瞭解 AWS 在涉及多種政策類型時如何判斷是否允許一項請求，請參閱 IAM 使用者指南中的政策評估邏輯。

AWS Panorama 如何與 IAM 搭配使用

在您使用 IAM 管理 AWS Panorama 的存取權限之前，您應該了解哪些 IAM 功能可與 AWS Panorama 搭配使用。若要深入瞭解 AWS Panorama 和其他AWS服務如何與 IAM 搭配使用，請參閱 IAM 使用者指南中的與 IAM 搭配使用的AWS服務。

如需 AWS Panorama 使用的許可、政策和角色的概觀，請參閱AWS Panorama 許可。

AWS Panorama 身分型政策範例

根據預設，IAM 使用者和角色沒有建立或修改 AWS Panorama 資源的權限。他們也無法使用 AWS Management Console、AWS CLI 或 AWS API 執行任務。IAM 管理員必須建立 IAM 政策，授予使用者和角色在指定資源上執行特定 API 操作的所需許可。管理員接著必須將這些政策連接至需要這些許可的 IAM 使用者或群組。

若要了解如何使用這些範例 JSON 政策文件建立 IAM 身分型政策，請參閱《IAM 使用者指南》中的在 JSON 標籤上建立政策。

主題

- 政策最佳實務
- 使用 AWS Panorama 主控台
- 允許使用者檢視他們自己的許可

政策最佳實務

以身分識別為基礎的政策可決定使用者是否可以在您的帳戶中建立、存取或刪除 AWS Panorama 資源。這些動作可能會讓您的 AWS 帳戶 產生費用。當您建立或編輯身分型政策時，請遵循下列準則及建議事項：

- 開始使用 AWS 受管政策並朝向最低權限許可的目標邁進：如需開始授予許可給使用者和工作負載，請使用 AWS 受管政策，這些政策會授予許可給許多常用案例。它們可在您的 AWS 帳戶 中使用。我們建議您定義特定於使用案例的 AWS 客戶管理政策，以便進一步減少許可。如需更多資訊，請參閱 IAM 使用者指南中的 AWS 受管政策或任務職能的 AWS 受管政策。
- 套用最低許可許可 – 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予對服務動作的存取權，前提是透過特定AWS 服務 (例如 AWS CloudFormation)使用條件。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。
- 需要多重要素驗證 (MFA)：如果存在需要 AWS 帳戶中 IAM 使用者或根使用者的情況，請開啟 MFA 提供額外的安全性。如需在呼叫 API 操作時請求 MFA，請將 MFA 條件新增至您的政策。如需更多資訊，請參閱 IAM 使用者指南中的設定 MFA 保護的 API 存取。

有關 IAM 中最佳實務的更多相關資訊，請參閱 IAM 使用者指南中的 IAM 最佳安全實務。

使用 AWS Panorama 主控台

若要存取 AWS Panorama 主控台，您必須擁有至少一組許可。這些許可必須允許您列出和檢視AWS帳戶中 AWS Panorama 資源的詳細資訊。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。

如需更多資訊，請參閱 AWS Panorama 政策

允許使用者檢視他們自己的許可

此範例會示範如何建立政策，允許 IAM 使用者檢視附加到他們使用者身分的內嵌及受管政策。此政策包含在主控台上，或是使用 AWS CLI 或 AWS API 透過編寫程式的方式完成此動作的許可。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS適用於 AWS 全景的受管政策

一個AWS受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，因為這些權限適用於所有使用案例AWS客戶使用。建議您透過定義進一步減少權限客戶管理的政策特定於您的使用案例。

您無法更改 AWS 受管政策中定義的許可。如果AWS更新中定義的權限AWS受管理的原則，更新會影響所附加原則的所有主體識別 (使用者、群組和角色)。AWS最有可能更新一個AWS管理策略，當一個新的AWS 服務已啟動或新的 API 操作可用於現有服務。

如需詳細資訊，請參閱《IAM 使用者指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_managed-vs-inline.html#aws-managed-policies中的 AWS 受管政策。

AWS 全景提供下列受管政策。如需每個政策的完整內容和變更歷程記錄，請參閱 IAM 主控台中的連結頁面。

- AWSPanoramaFullAccess— 提供對 AWS 全景、Amazon S3 中的 AWS 全景存取點、中的設備登入資料的完整存取權AWS Secrets Manager和亞馬遜中的設備日誌CloudWatch。包括創建一個權限服務連結角色適用於 AWS 全景資訊。
- AWSPanoramaServiceLinkedRolePolicy— 允許 AWS 全景管理 AWS 物聯網、AWS 秘密管理員和 AWS 全景資源。
- AWSPanoramaApplianceServiceRolePolicy— 允許 AWS 全景設備將日誌上傳到CloudWatch，並從 AWS 全景創建的亞馬遜 S3 接入點獲取對象。

AWS 全景更新至AWS受管政策

下表說明 AWS 全景受管政策的更新。

變更	描述	日期
AWSPanoramaFullAccess – 更新現有政策	增加了用戶策略的權限，以允許用戶查看日誌組CloudWatch日誌控制台。	2022-01-13
AWSPanoramaFullAccess – 更新現有政策	為使用者政策新增許可，以允許使用者管理 AWS 全景服務連結角色，並在其他服務中訪問 AWS 全景資源，包括 IAM，亞馬遜 S3，CloudWatch和秘密經理。	2021-10-20

變更	描述	日期
AWSPanoramaApplianceServiceRolePolicy – 新政策	AWS 全景設備服務角色的新政策	2021-10-20
AWSPanoramaServiceLinkedRolePolicy – 新政策	AWS 全景服務連結角色的新政策。	2021-10-20
AWS 全景開始追蹤變更	AWS 全景開始追蹤其變更 AWS受管理的策略。	2021-10-20

使用 AWS Panorama 的服務連結角色

AWS Panorama 使用 AWS Identity and Access Management (IAM) 服務連結的角色。服務連結角色是直接連結至 AWS Panorama 的一種特殊 IAM 角色類型。服務連結角色由 AWS Panorama 預先定義，且內含該服務代您呼叫其他 AWS 服務所需的所有許可。

服務連結的角色可讓設定 AWS Panorama 更為簡單，因為您不必手動新增必要的許可。AWS Panorama 定義其服務連結角色的許可，除非另有定義，否則僅有 AWS Panorama 可以擔任其角色。定義的許可包含信任政策和許可政策，並且該許可政策不能附加到任何其他 IAM 實體。

您必須先刪除服務連結角色的相關資源，才能將其刪除。如此可保護您 AWS Panorama 的資源，避免您不小心移除資源的存取許可。

如需關於支援服務連結角色的其他服務的資訊，請參閱可搭配 IAM 運作的 AWS 服務，尋找 Service-Linked Role (服務連結角色) 欄中顯示為 Yes (是) 的服務。選擇具有連結的 Yes (是)，以檢視該服務的服務連結角色文件。

章節

- AWS Panorama 的服務連結角色許可
- 為 AWS Panorama 建立服務連結角色
- 為 AWS Panorama 編輯服務連結角色
- 為 AWS Panorama 刪除服務連結角色
- AWS Panorama 服務連結角色的支援區域

AWS Panorama 的服務連結角色許可

AWS Panorama使用名為 的服務連結角色適用於安全服務— 允許 AWS Panorama 管理 AWS IoT、AWS Secrets Manager 和 AWS Panorama 中的資源。

AWSServiceRoleForAWSServiceRoleForAWSServiceRole 服務連結角色信任下列服務擔任該角色：

- `panorama.amazonaws.com`

角色許可政策允許 AWS Panorama 完成以下動作：

- 監控AWS Panorama資源
- Manage (管理)AWS IoT資源AWS Panorama設備
- 存取AWS Secrets Manager獲取相機憑據的祕密

如需完整的權限清單，查看自動銷售服務策略策略在 IAM 主控台。

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的服務連結角色許可。

為 AWS Panorama 建立服務連結角色

您不需要手動建立一個服務連結角色。當您在AWS Management Console，AWS CLI，或 AWSAPI,AWS Panorama會為您建立服務連結角色。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。註冊裝置時，AWS Panorama會再次為您建立服務連結角色。

為 AWS Panorama 編輯服務連結角色

AWS Panorama不允許您編輯 AWSServiceRoleForAWSServiceRoleForAWSServiceRoleForAWSServiceRole 服務連結角色。因為有各種實體可能會參考服務連結角色，所以您無法在建立角色之後變更角色名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱《IAM 使用者指南》中的編輯服務連結角色。

為 AWS Panorama 刪除服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，在手動刪除服務連結角色之前，您必須先清除資源。

刪除AWS Panorama資源時，請使用本指南以下各節的程序。

- 刪除版本和應用程式
- 取消註冊設備

> **Note**
>
> 若 AWS Panorama 服務在您試圖刪除資源時正在使用該角色，刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

若要刪除AWSServiceRoleForAWSServiceRoleForAWSServiceRoleForAWSServiceRoleForAWSServiceRole服務連結角色，請AWS CLI，或AWSAPI。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。

AWS Panorama 服務連結角色的支援區域

AWS Panorama 在所有提供服務的區域中支援使用服務連結的角色。如需詳細資訊，請參閱「」AWS 區域與端點。

預防跨服務混淆代理人

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

若要限制 AWS Panorama 為資源提供另一項服務的許可，我們建議在資源政策中使用 `aws:SourceArn` 和 `aws:SourceAccount` 全域條件內容索引鍵。如果同時使用全域條件內容索引鍵，則在相同政策陳述式中使用`aws:SourceAccount` 值和 `aws:SourceArn` 值中的帳戶時，必須使用相同的帳戶 ID。

的值`aws:SourceArn`必須是AWS Panorama設備。

防止混淆副手問題的最有效方法是使用`aws:SourceArn`全局條件上下文鍵與資源的完整 ARN。如果不知道資源的完整 ARN，或者如果要指定多個資源，請使用`aws:SourceArn`帶通配符的全局上下文條件鍵（`*`），查看 ARN 的未知部分。例如：`arn:aws:`*`servicename`*`::`*`123456789012`*`:*`。

有關保護服務角色的說明，AWS Panorama用於授予權限AWS Panorama裝置，請參閱保護應用裝置角色。

AWS Panorama 身分和存取疑難排解

使用下列資訊協助您診斷和修正使用 AWS Panorama 和 IAM 時可能會遇到的常見問題。

主題

我沒有在 AWS Panorama 中執行動作的授權

若 AWS Management Console 告知您並未獲得執行動作的授權，您必須聯絡您的管理員以取得協助。您的管理員是提供您使用者名稱和密碼的人員。

當 `mateojackson` IAM 使用者嘗試使用主控台檢視有關設備的詳細資料但沒有`panorama:DescribeAppliance`許可時，會發生下列範例錯誤。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 panorama:DescribeAppliance on resource: my-appliance
```

在此情況下，Mateo 會請求管理員更新他的政策，允許他使用 `my-appliance` 動作存取 `panorama:DescribeAppliance` 資源。

我沒有授權執行 iam：PassRole

如果您收到未獲授權執行`iam:PassRole`動作的錯誤訊息，則必須更新您的政策以允許您將角色傳遞給 AWS Panorama。

有些 AWS 服務 允許您傳遞現有的角色至該服務，而無須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 AWS Panorama 中執行動作時，會發生下列範例錯誤。但是，該動作要求服務具備服務角色授與的許可。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。

如需任何協助，請聯絡您的 AWS 管理員。您的管理員提供您的登入憑證。

我想要允許AWS帳戶以外的人員存取我的 AWS Panorama 資源

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

如需進一步了解，請參閱以下內容：

- 若要了解 AWS Panorama 是否支援這些功能，請參閱AWS Panorama 如何與 IAM 搭配使用。
- 如需了解如何存取您擁有的所有 AWS 帳戶 所提供的資源，請參閱《IAM 使用者指南》中的將存取權提供給您所擁有的另一個 AWS 帳戶 中的 IAM 使用者。
- 如需了解如何將資源的存取權提供給第三方 AWS 帳戶，請參閱《IAM 使用者指南》中的將存取權提供給第三方擁有的 AWS 帳戶。
- 如需了解如何透過聯合身分提供存取權，請參閱《IAM 使用者指南》中的將存取權提供給在外部進行身分驗證的使用者 (聯合身分)。
- 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。

AWS Panorama 的合規驗證

要瞭解 AWS 服務 是否在特定法規遵循方案範圍內，請參閱法規遵循方案範圍內的 AWS 服務，並選擇您感興趣的法規遵循方案。如需一般資訊，請參閱 AWS 法規遵循方案。

您可使用 AWS Artifact 下載第三方稽核報告。如需詳細資訊，請參閱 AWS Artifact 中的下載報告。

您使用 AWS 服務 時的法規遵循責任取決於資料的敏感度、您的公司的合規目標，以及適用的法律和法規。AWS 提供以下資源協助您處理法規遵循事宜：

- 安全與合規快速入門指南 – 這些部署指南討論在 AWS 上部署以安全及合規為重心的基準環境的架構考量和步驟。
- Amazon Web Services 的 HIPAA 安全與法規遵循架構：本白皮書說明公司可如何運用 AWS 來建立符合 HIPAA 規定的應用程式。

> **Note**
>
> 並非全部的 AWS 服務 都符合 HIPAA 資格。如需詳細資訊，請參閱 HIPAA 資格服務參照。

- AWS 合規資源：這組手冊和指南可能適用於您的產業和位置。
- AWS 客戶合規指南：透過合規的角度了解共同的責任模式。這份指南橫跨多個架構 (包含國家標準技術研究所 (NIST)、支付卡產業安全標準委員會 (PCI) 和國際標準組織 (ISO))，總結保護 AWS 服務的最佳實務並將指導方針對應至安全控制。
- AWS Config 開發人員指南中的使用規則評估資源：AWS Config 服務可評估您的資源組態對於內部實務、業界準則和法規的合規狀態。
- AWS Security Hub – 此 AWS 服務 可供您全面檢視 AWS 中的安全狀態。Security Hub 使用安全控制，可評估您的 AWS 資源並檢查您的法規遵循是否符合安全業界標準和最佳實務。如需支援的服務和控制清單，請參閱 Security Hub controls reference。
- AWS Audit Manager – 此 AWS 服務 可協助您持續稽核 AWS 使用情況，以簡化管理風險與法規與業界標準的法規遵循方式。

人們在場時的其他注意事項

以下是在使用 AWS Panorama 時應考慮的一些最佳實務：

- 確保您了解並遵守適用於您的使用案例的所有法律和法規。這可能包括與攝影機定位和視野相關的法律、放置和使用相機時的注意事項和標誌要求，以及您影片中可能出現的人士的權利，包括他們的隱私權。
- 考慮到相機對人員及其隱私的影響。除了法律要求外，請考慮在相機所在的區域放置通知是否適當，以及是否應將相機放置在清晰的視線範圍內並且沒有任何遮蔽，這樣人們就不會感到驚訝他們可能在攝像機上。
- 制定適當的政策和程序來操作攝像機，並查看從攝像機獲取的數據。
- 針對從攝影機取得的資料，考慮適當的存取控制、使用限制和保留期限。

AWS 全景中的基礎設施安全

身為受管服務，AWS 全景受到保護AWS全球網絡安全。如需有關 AWS 安全服務以及 AWS 如何保護基礎設施的詳細資訊，請參閱 AWS 雲端安全。若要使用基礎設施安全性的最佳實務來設計您的 AWS 環境，請參閱安全性支柱 AWS 架構良好的框架中的基礎設施保護。

您使用AWS已發佈的 API 呼叫，可透過網路存取 AWS 全景圖。用戶端必須支援下列項目：

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

此外，請求必須使用存取索引鍵 ID 和與 IAM 主體相關聯的私密存取索引鍵來簽署。或者，您可以使用 AWS Security Token Service (AWS STS) 來產生暫時安全憑證來簽署請求。

在您的資料中心部署 AWS 全景設備

AWS 全景設備需要存取網際網路才能與之通訊AWS服務。它還需要訪問您的內部攝像機網絡。請務必仔細考慮您的網路組態，並且僅為每個裝置提供所需的存取權。如果您的組態允許 AWS Panorama 設備做為敏感 IP 攝影機網路的橋接器，請小心。

您必須對以下事項負責：

- AWS 全景設備的實體和邏輯網路安全性。
- 當您使用 AWS 全景設備時，可以安全地操作連接網路的攝影機。
- 保持 AWS 全景設備和相機軟體的更新。
- 遵守與您從生產環境收集的影片和影像 (包括與隱私權相關的影片和影像) 內容相關的任何適用法律或法規。

AWS 全景設備使用未加密的 RTSP 攝影機串流。如需將 AWS 全景設備連接到網路的詳細資訊，請參閱將 AWS Panorama 設備連接到您的網路。如需加密的詳細資訊，請參閱AWS Panorama 中的資料保護。

AWS Panorama 中的運行時環境軟件

AWS Panorama 提供的軟件可在 AWS Panorama 設備上的基於 Ubuntu Linux 的環境中運行您的應用程序代碼。AWS Panorama 負責使設備映像中的軟件保持最新狀態。AWS Panorama 定期發佈軟件更新，您可以通過使用 AWS Panorama 控制台。

您可以在應用程序代碼中使用庫，方法是將它們安裝在應用程序的`Dockerfile`。要確保跨構建的應用程序穩定性，請選擇每個庫的特定版本。定期更新您的依賴關係以解決安全問題。

推出

下表顯示針對AWS Panorama服務、軟體和文件發行功能和軟體更新的時間。為確保您可以存取所有功能，請將AWS Panorama設備更新為最新的軟體版本。如需有關發行版本的詳細資訊，請參閱連結的主題。

變更	描述	日期
設備軟體更新	7.0.13 版是主要版本更新，可變更應用裝置管理軟體更新的方式。如果限制從應用裝置輸出的網路通訊，或將其連線至私有 VPC 子網路，則在套用更新之前，必須允許存取其他端點和連接埠。如需詳細資訊，請參閱變更記錄。	2023 年 12 月 28 日
設備軟體更新	版本 6.2.1 包括錯誤修復。如需詳細資訊，請參閱變更記錄。	2023 年 9 月 6 日
設備軟體更新	版本 6.0.8 包括錯誤修復和安全改進。如需詳細資訊，請參閱變更記錄。	2023 年 7 月 6 日
設備軟體更新	版本 5.1.7 包括錯誤修復和錯誤處理改進。如需詳細資訊，請參閱變更記錄。	2023 年 3 月 31 日
主控台更新	您現在可以從管理主控台購買AWS Panorama應用裝置。若要授與使用者購買裝置的權限，請參閱 AWS Panorama 的身分型 IAM 政策。	2023 年 2 月 2 日

設備軟體更新	版本 5.0.74 包括錯誤修復和錯誤處理改進。如需詳細資訊，請參閱變更記錄。	2023 年 1 月 23 日
API 更新	增加了使設備軟件主要版本更新AllowMajorVersionUpdate 選擇加入的選項。OTAJobConfig 如需詳細資訊，請參閱CreateJobForDevices。	2023 年 1 月 19 日
開發人員的新工具	AWS Panorama範例 GitHub 儲存庫中提供了一個新工具「側載」。您可以使用此工具更新應用程式程式碼，而無需建置和部署容器。如需詳細資訊，請參閱讀我檔案。	2022 年 11 月 16 日
應用程式庫映像更新	版本 1.2.0 添加了一個超時選項video_in.get() ，設置AWS_REGION 環境變量，並改進了錯誤處理。如需詳細資訊，請參閱變更記錄。	2022 年 11 月 16 日
設備軟體更新	版本 5.0.42 包括錯誤修復和安全更新。如需詳細資訊，請參閱變更記錄。	2022 年 11 月 16 日
設備軟體更新	版本 5.0.7 增加了對遠程重啟設備並遠程暫停攝像機流的支持。如需詳細資訊，請參閱變更記錄。	2022 年 10 月 13 日
設備軟體更新	版本 4.3.93 增加了對從離線設備檢索日誌的支持。如需詳細資訊，請參閱變更記錄。	2022 年 8 月 24 日

設備軟體更新	版本 4.3.72 包括錯誤修復和安全更新。如需詳細資訊，請參閱變更記錄。	2022 年 6 月 23 日
AWS PrivateLink 支援	AWS Panorama支援 VPC 端點，用於管理私有子網路中的AWS Panorama資源。如需詳細資訊，請參閱使用 VPC 端點。	2022 年 6 月 2 日
設備軟體更新	版本 4.3.55 提高了日誌的`console_output` 存儲利用率。如需詳細資訊，請參閱變更記錄。	2022 年 5 月 5 日
ThinkEdge联想	聯想提供了一種新的AWS Panorama設備。聯想 ThinkEdge® SE70，由英偉達傑特森澤維爾 NX 提供支持，支持相同的功能設備。AWS Panorama如需詳細資訊，請參閱相容裝置。	2022 年 4 月 6 日
應用程式庫映像更新	版本 1.1.0 提高了運行後台線程時的性能，並向媒體對象添加了一個標誌（is_cached），以指示圖像是否是新鮮的。如需詳細資訊，請參閱庫 .ecr.aws。	2022 年 3 月 29 日
設備軟體更新	版本 4.3.45 增加了對 GPU 訪問和入站端口的支持。如需詳細資訊，請參閱變更記錄。	2022 年 3 月 24 日
設備軟體更新	版本 4.3.35 提高了安全性和性能。如需詳細資訊，請參閱變更記錄。	2022 年 2 月 22 日

更新的受管政策	AWS Identity and Access Management的受管理策略AWS Panorama已更新。如需詳細資訊，請參閱 AWS 受管政策。	2022 年 1 月 13 日
佈建記錄	使用設備軟體 4.3.23，設備會在佈建期間將記錄寫入 USB 磁碟機。如需詳細資訊，請參閱記錄檔。	2022 年 1 月 13 日
NTP 伺服器組態	您現在可以將設AWS Panorama備設定為使用特定 NTP 伺服器進行時鐘同步化。在設定應用裝置期間使用其他網路設定來設定 NTP 設定。如需詳細資訊，請參閱設定。	2022 年 1 月 13 日
其他地區	AWS Panorama現已在亞太區域 (新加坡) 和亞太區域 (雪梨) 區域推出。	2022 年 1 月 13 日
設備軟體更新	版本 4.3.4 增加了對模型`precisionMode` 設置和更新日誌記錄行為的支持。如需詳細資訊，請參閱變更記錄。	2021 年 11 月 8 日
更新的受管政策	AWS Identity and Access Management的受管理策略AWS Panorama已更新。如需詳細資訊，請參閱 AWS 受管政策。	2021 年 10 月 20 日

一般可用性	AWS Panorama現在可供美國東部 (維吉尼亞北部)、美國西部 (奧勒岡)、歐洲 (愛爾蘭) 和加拿大 (中部) 區域的所有客戶使用。若要購買設AWS Panorama備，請造訪AWS Panorama。	2021 年 10 月 20 日
預覽版	AWS Panorama可在美國東部 (維吉尼亞北部) 和美國西部 (奧勒岡) 區域透過邀請提供。	2020 年 12 月 1 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。