開發人員指南

AWS Panorama

AWS Panorama: 開發人員指南

Table of Contents

終止支援通知：2026 年 5 月 31 日， AWS 將終止對 的支援 AWS Panorama。2026 年 5 月 31 日之後，您將無法再存取 AWS Panorama 主控台或 AWS Panorama 資源。如需詳細資訊，請參閱AWS Panorama 終止支援。

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。

什麼是 AWS Panorama？

AWS Panorama 是一項將電腦視覺帶入現場部署攝影機網路的服務。您可以在資料中心安裝 AWS Panorama Appliance 或其他相容的裝置、向 註冊 AWS Panorama，以及從雲端部署電腦視覺應用程式。 可與現有的即時串流通訊協定 (RTSP) 網路攝影機 AWS Panorama 搭配使用。設備會從 AWS 合作夥伴或您使用應用程式開發套件自行建置的應用程式執行安全的電腦視覺 AWS Panorama 應用程式。

AWS Panorama 設備是一種精簡的邊緣設備，使用針對機器學習工作負載最佳化的強大system-on-module(SOM)。設備可以平行針對多個影片串流執行多個電腦視覺模型，並即時輸出結果。其設計用於商業和工業設定，並評定為灰塵和液體保護 (IP-62)。

AWS Panorama 設備可讓您在邊緣執行獨立電腦視覺應用程式，而無需將映像傳送至 AWS 雲端。透過使用 AWS 開發套件，您可以與其他 AWS 服務整合，並使用它們來追蹤一段時間內來自應用程式的資料。透過與其他 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 Amazon DynamoDB 中零售分析的資料。使用無伺服器應用程式來分析一段時間內收集的資料、偵測資料的異常，以及預測未來的行為。
- 接收站點安全提醒 – 監控工業站點的限制外區域。當您的應用程式偵測到潛在的不安全情況時，請將映像上傳至 Amazon Simple Storage Service (Amazon S3)，並將通知傳送至 Amazon Simple Notification Service (Amazon SNS) 主題，以便收件人採取更正動作。
- 改善品質控制 – 監控組裝線的輸出，以識別不符合要求的組件。使用文字和週框方塊反白不符合部分的影像，並將其顯示在監視器上，以供您的品管團隊檢閱。
- 收集訓練和測試資料 – 上傳電腦視覺模型無法識別的物件影像，或模型對其猜測的可信度為邊界的物件影像。使用無伺服器應用程式建立需要標記的映像佇列。標記影像，並使用它們在 Amazon SageMaker AI 中重新訓練模型。

AWS Panorama 使用其他 AWS 服務來管理 AWS Panorama 設備、存取模型和程式碼，以及部署應用程式。 AWS Panorama 會盡可能執行動作，而無需您與其他 服務互動，但了解下列服務可協助您了解 AWS Panorama 的運作方式。

- SageMaker AI – 您可以使用 SageMaker AI 從攝影機或感應器收集訓練資料、建置機器學習模型，以及針對電腦視覺進行訓練。 AWS Panorama 使用 SageMaker AI Neo 最佳化模型以在 AWS Panorama 設備上執行。

- 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，以部署至 AWS Panorama 設備。
- AWS IoT – AWS Panorama 使用 AWS IoT 服務來監控 AWS Panorama 設備的狀態、管理軟體更新和部署應用程式。您不需要 AWS IoT 直接使用 。

若要開始使用 AWS Panorama 設備並進一步了解 服務，請繼續入門 AWS Panorama。

AWS Panorama 終止支援

在仔細考慮之後，我們決定終止對 AWS Panorama 的支援，自 2026 年 5 月 31 日起生效。從 2025 年 5 月 20 日起，AWS Panorama 將不再接受新客戶。身為在 2025 年 5 月 20 日之前註冊服務的現有客戶，您可以繼續使用 AWS Panorama 功能。2026 年 5 月 31 日之後，您將無法再使用 AWS Panorama。

AWS Panorama 的替代方案

如果您對 AWS Panorama 的替代方案感興趣， AWS 同時為買方和建置者提供選項。

對於out-of-the-box解決方案，AWS 合作夥伴網路提供多個合作夥伴的解決方案。您可以從許多合作夥伴瀏覽 AWS 解決方案程式庫上的解決方案。這些合作夥伴解決方案包括硬體、軟體、軟體即服務 (SaaS) 應用程式、受管解決方案或根據您的需求自訂實作的選項。此方法提供解決方案，可解決您的使用案例，而無需您在電腦視覺、AI 或應用程式開發方面擁有專業知識。這通常會利用 AWS 合作夥伴的專業知識，提供更快的價值時間。

如果您偏好建置自己的解決方案， AWS 會提供 AI 工具和服務，協助您開發 AI 型電腦視覺應用程式，並在邊緣管理應用程式和裝置。Amazon SageMaker 提供一組工具，利用全受管的基礎設施、工具和工作流程，為您的使用案例建置、訓練和部署 ML 模型。除了可讓您建置自己的模型之外，Amazon SageMaker JumpStart 還提供內建的電腦視覺演算法，可根據您的特定使用案例進行微調。

為了在邊緣管理裝置和應用程式，AWS IoT Greengrass 是一種經過驗證且安全的解決方案，可用於部署和更新 IoT 裝置的應用程式。對於以伺服器為基礎的實作，AWS Systems Manager 提供一套工具來管理伺服器，Amazon EKS Anywhere 或 ECS Anywhere 可以管理邊緣伺服器上的應用程式容器。Amazon 提供一些管理邊緣裝置的準則，以及使用 AWS 保護物聯網 (IoT) 白皮書第 4 節中的其他資源。此建置器方法為您提供加速 AI 和裝置管理開發的工具，同時提供完全彈性，以建置符合您確切需求的解決方案，並與現有的硬體和軟體基礎設施整合。這通常可以降低解決方案的營運成本。

從 AWS Panorama 遷移

若要將現有應用程式從 AWS Panorama 移至替代實作，您需要取代現有的硬體裝置、從 AWS Panorama 服務遷移應用程式，以及實作新解決方案的邊緣管理和安全性。以下將詳細說明每個領域：

硬體替換

現有的 AWS Panorama 設備是以 Nvidia Jetson Xavier 平台為基礎。可以根據符合您需求的最新一代 Nvidia Jetson 平台或邊緣伺服器，以類似的off-the-shelf裝置取代硬體。雖然大多數 AWS Panorama 部署都可以用類似的裝置取代，但我們看到一些在單一位置使用大量攝影機的客戶發現伺服器是更好的替代方案。

應用程式遷移

需要重寫 AWS Panorama 應用程式，以消除使用任何 AWS Panorama 特定的 API 呼叫。AWS Panorama 應用程式僅支援使用 H.264 透過即時串流通訊協定 (RTSP) 饋送的視訊輸入，而且這些視訊輸入是使用 AWS Panorama 裝置 SDK 中的攝影機節點提供。

若要移植現有應用程式，您需要實作類似 AWS Panorama 的應用程式類別，以便大部分地重複使用現有程式碼。範例程式碼可在 banner-code.zip 檔案中取得，其中顯示使用 PyAV 和 OpenCV 的此實作範例。

這是一種簡單的方法，具有最少的程式碼變更量，但對於支援的影片串流類型，有許多與目前 AWS Panorama 型實作相同的限制。

另一個選項是重新建構應用程式，以更好地利用系統資源並支援新的應用程式功能。對於此選項，您可以使用 GStreamer 或 DeepStream 實作從媒體來源到推論結果和商業邏輯的媒體管道，或使用功能更豐富且效能更好的機器學習 (ML) 執行期實作，例如 Nvidia Triton 推論伺服器。這種方法需要變更更多影片處理管道，但既更有效率，也允許更靈活地支援更廣泛的轉碼器、攝影機類型和其他感應器。

Edge 管理和安全性

無論媒體管道為何，您也必須實作登入資料的安全存放區，例如 RTSP 串流使用者名稱和密碼。 AWS 為應用程式提供不同的安全儲存參數方式：

- AWS IoT Device Shadow 服務用於存放傳遞給應用程式的參數，以及追蹤邊緣裝置上應用程式的狀態。
- AWS Secrets Manager 用於存放此類登入資料，以更好地保護登入資料以存取媒體串流。
- 如果您使用 Amazon EKS 或 Amazon ECS，您也可以使用安全的 AWS System Manager 參數存放區來取得登入資料和其他應用程式參數。

選擇取決於應用程式的安全需求，以及您計劃用於實作應用程式的其他 AWS 產品。

當您將 AWS Panorama 設備取代為一般邊緣裝置時，您還必須為應用程式實作必要的安全功能，並設定裝置以符合您的安全需求。 AWS 在 AWS Well-Architected Framework 的安全支柱中提供相關指導。雖然架構主要著重於雲端應用程式，但大多數原則也適用於邊緣裝置。此外，您應該使用所選解決方案的硬體安全功能，例如 AWS IoT Greengrass V2 硬體安全整合，並使用所選作業系統和/或裝置提供的安全功能，例如全磁碟加密。

Summary

雖然 AWS Panorama 計劃在 2026 年 5 月 31 日關閉，但 以 Amazon SageMaker 工具的形式 AWS 提供一組強大的 AI/ML 服務和解決方案，以建置電腦視覺模型和裝置管理服務，例如 AWS IoT Greengrass、Amazon EKS 和 Amazon ECS Anywhere 和 AWS System Manager，以支援類似解決方案的開發。如果您偏好購買而不是建置解決方案， AWS 也有來自 AWS 合作夥伴網路合作夥伴的一系列產品。提供範例程式碼和實作指引，以協助您遷移到替代解決方案。您應該探索這些選項，以判斷哪些選項最適合您的特定需求。

如需詳細資訊，請參閱下列資源：

- Amazon SageMaker 開發人員指南 – 有關如何建置模型或使用 SageMaker JumpStart 中提供之內建電腦視覺演算法的詳細文件。
- AWS IoT Core 開發人員指南 - 有關如何連接和管理 IoT 裝置的詳細文件。
- AWS IoT Greengrass V2 開發人員指南 – 有關如何在裝置上建置、部署和管理 IoT 應用程式的詳細文件。
- ECS Anywhere 開發人員指南 - 在邊緣執行 ECS 的詳細文件。
- EKS Anywhere 最佳實務指南 - 在邊緣執行 EKS 的詳細文件。
- AWS Solutions Library – 來自一系列供應商的合作夥伴產品，提供預先建置或自訂的電腦視覺解決方案。
- Panorama FAQs - 其他 Panorama 資訊。

常見問答集

Panorama 中止的時機為何？

已於 2025 年 5 月 20 日公告。在此日期之後，未在服務上處於作用中狀態的客戶將無法再存取 Panorama。作用中的客戶將能夠繼續正常使用服務，直到 2026 年 5 月 31 日為止。在這段時間之

前，客戶都可以將其應用程式移至替代解決方案，並遷移 Panorama 的應用程式。2026 年 5 月 31 日之後，任何嘗試存取 Panorama 服務的應用程式將無法再運作，且 Panorama 裝置將無法再運作。

現有客戶會受到什麼影響？

現有客戶可以繼續正常使用服務，直到 2026 年 5 月 31 日為止。之後，嘗試存取 Panorama 的應用程式將無法再運作。Panorama 裝置在該日期之後也無法再運作。

是否接受新客戶？

否。自 2025 年 5 月 20 日起，只有 Panorama 有效使用者的客戶才能存取該服務。如果客戶在服務中有先前使用的應用程式需要存取，他們可以建立客戶支援案例來請求其帳戶的存取。如果客戶先前沒有使用該服務，則不會授予他們存取權。

客戶可以探索哪些替代方案？

AWS 提供可取代 Panorama 功能的一系列服務。我們建議客戶使用off-the-shelf的硬體，並透過符合其需求的 AWS IoT Core、AWS IoT Greengrass、Amazon AKS Anywhere、Amazon ECS Anywhere 和 / 或 AWS System Manager 組合來管理裝置和應用程式。AWS 合作夥伴網路也有數個解決方案，可供具有特定電腦視覺專業知識的合作夥伴使用，客戶可以考慮。

客戶如何從 Panorama 遷移？

Panorama 應用程式需要修改，以移除 Panorama 特定 APIs 上任何主要與攝影機連線和串流相關的相依性。 AWS 提供了範例程式碼，示範如何進行這些變更。一旦移除這些相依性，應用程式就可以移至替代硬體平台。

如果我在 2025 年 5 月 20 日或之後發生問題，將會提供哪些支援？

AWS 將持續提供 Panorama 的支援，直到終止通知期間結束為止 (2026 年 5 月 31 日）。對於任何支援需求，客戶應透過其正常支援管道輸入支援案例。 AWS 將提供安全性更新、錯誤修正和可用性增強功能。

我無法在 2026 年 5 月 31 日之前遷移。是否可以延長日期？

我們確信 Panorama 可用的替代方案可讓客戶在 2026 年 5 月 31 日之前遷移至替代解決方案，而且我們沒有計劃在該日期後延長服務的可用性。

我的邊緣應用程式會在服務結束後繼續運作嗎？

否。Panorama 裝置和應用程式取決於 Panorama 雲端服務的連線能力。一旦該服務在 2026 年 5 月 31 日停止，Panorama 應用程式和 Panorama 裝置都不會繼續運作。

入門 AWS Panorama

若要開始使用 AWS Panorama，請先了解 服務的概念和本指南中使用的術語。然後，您可以使用 AWS Panorama 主控台來註冊您的 AWS Panorama 設備並建立應用程式。您可以在大約一小時內設定裝置、更新其軟體，以及部署範例應用程式。若要完成本節中的教學課程，您可以使用 AWS Panorama 裝置和透過本機網路串流視訊的攝影機。

> **Note**
>
> 若要購買 AWS Panorama 設備，請造訪 AWS Panorama 主控台。

AWS Panorama 範例應用程式示範 AWS Panorama 功能的使用方式。它包含已使用 SageMaker AI 訓練的模型，以及使用 AWS Panorama 應用程式 SDK 執行推論和輸出影片的範例程式碼。範例應用程式包含 CloudFormation 範本和指令碼，說明如何從命令列自動化開發和部署工作流程。

本章的最後兩個主題詳細說明了模型和攝影機的需求，以及 AWS Panorama 設備的硬體規格。如果您尚未取得設備與攝影機，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 Panorama 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攝影機
- AWS Panorama 設備規格
- Service Quotas

AWS Panorama 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 設備或相容的裝置，以分析網路攝影機的影片串流。您可以在 Python 中編寫應用程式程式碼，並使用 Docker 建置應用程式容器。您可以使用 AWS Panorama Application CLI，從本機或從 Amazon Simple Storage Service (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攝影機的視訊輸入，並與模型互動。

概念

- AWS Panorama 設備
- 相容裝置
- 應用程式
- 節點
- 模型

AWS Panorama 設備

AWS Panorama 設備是執行應用程式的硬體。您可以使用 AWS Panorama 主控台來註冊設備、更新其軟體，以及將應用程式部署到其中。AWS Panorama 設備上的軟體會連線至攝影機串流、將影片影格傳送至您的應用程式，並在連接的顯示器上顯示影片輸出。

AWS Panorama Appliance 是採用 Nvidia Jetson AGX Xavier 的邊緣裝置。它會在本機於最佳化硬體上執行應用程式，而不是將映像傳送至 AWS 雲端進行處理。這可讓您即時分析影片，並在本機處理結果。設備需要網際網路連線來報告其狀態、上傳日誌，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 Panorama 設備之外，AWS Panorama 還支援來自 AWS 合作夥伴的相容裝置。相容裝置支援與 AWS Panorama 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® SE70 – 採用 Nvidia Jetson Xavier NX 技術

本指南中的內容和範例應用程式是使用 AWS Panorama 設備開發。如需裝置特定硬體和軟體功能的詳細資訊，請參閱製造商的文件。

應用程式

應用程式在 AWS Panorama 設備上執行，以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，並透過網際網路將其部署到 AWS Panorama 設備。應用程式可以將影片傳送至顯示器，或使用 AWS 開發套件將結果傳送至 AWS 服務。

若要建置和部署應用程式，您可以使用 AWS Panorama 應用程式 CLI。AWS Panorama Application 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案、使用 Docker 建置容器，以及上傳資產。您可以在一個裝置上執行多個應用程式。

如需詳細資訊，請參閱管理 AWS Panorama 應用程式。

節點

應用程式包含多個稱為節點的元件，代表輸入、輸出、模型和程式碼。節點只能是組態（輸入和輸出），或包含成品（模型和程式碼）。應用程式程式碼節點會封裝在您上傳至 Amazon S3 存取點的節點套件中，AWS Panorama Appliance 可以在其中存取它們。應用程式資訊清單是定義節點之間連線的組態檔案。

如需詳細資訊，請參閱應用程式節點。

模型

電腦視覺模型是一種機器學習網路，經過訓練可處理映像。電腦視覺模型可以執行各種任務，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，並輸出影像或影像中物件的相關資訊。

AWS Panorama 支援使用 PyTorch、Apache MXNet 和 TensorFlow 建置的模型。您可以使用 Amazon SageMaker AI 或在您的開發環境中建置模型。如需詳細資訊，請參閱???。

設定 AWS Panorama 設備

若要開始使用您的 AWS Panorama 設備或相容裝置，請在 AWS Panorama 主控台中註冊它並更新其軟體。在設定程序中，您會在代表實體設備的 AWS Panorama 中建立設備資源，並使用 USB 磁碟機將檔案複製到設備。設備使用這些憑證和組態檔案來連線至 AWS Panorama 服務。然後，您可以使用 AWS Panorama 主控台來更新設備的軟體並註冊攝影機。

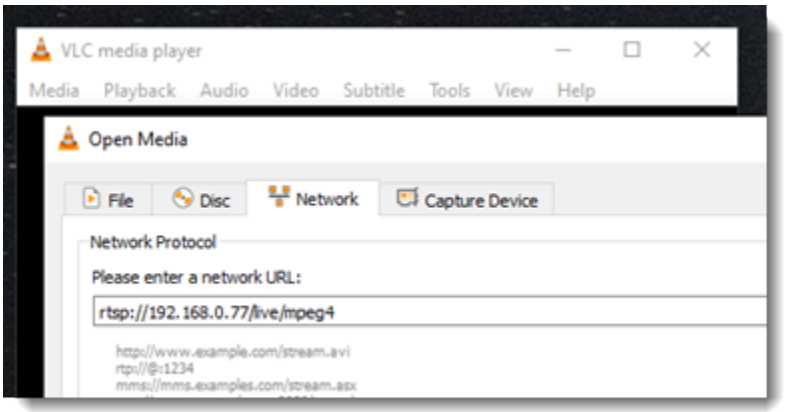
章節

先決條件

若要遵循本教學課程，您需要 AWS Panorama 設備或相容的裝置，以及下列硬體：

- 顯示 – 具有 HDMI 輸入的顯示，用於檢視範例應用程式輸出。
- USB 磁碟機（隨附於 AWS Panorama 設備）– FAT32-formatted的 USB 3.0 隨身碟，具有至少 1 GB 的儲存空間，用於將具有組態檔案和憑證的封存檔傳輸至 AWS Panorama 設備。
- 攝影機 – 輸出 RTSP 影片串流的 IP 攝影機。

使用相機製造商提供的工具和指示來識別相機的 IP 地址和串流路徑。您可以使用 VLC 等影片播放器，將其開啟為網路媒體來源來驗證串流 URL：



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。若要註冊設備並部署範例應用程式，您需要下列許可：

- AWSPanoramaFullAccess – 提供 AWS Panorama、Amazon S3 中的 AWS Panorama 存取點、 中的設備登入資料 AWS Secrets Manager，以及 Amazon CloudWatch 中的設備日誌的完整存取權。包含為 AWS Panorama 建立服務連結角色的許可。
- AWS Identity and Access Management (IAM) – 第一次執行時，建立 AWS Panorama 服務和 AWS Panorama 設備所使用的角色。

如果您沒有在 IAM 中建立角色的許可，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

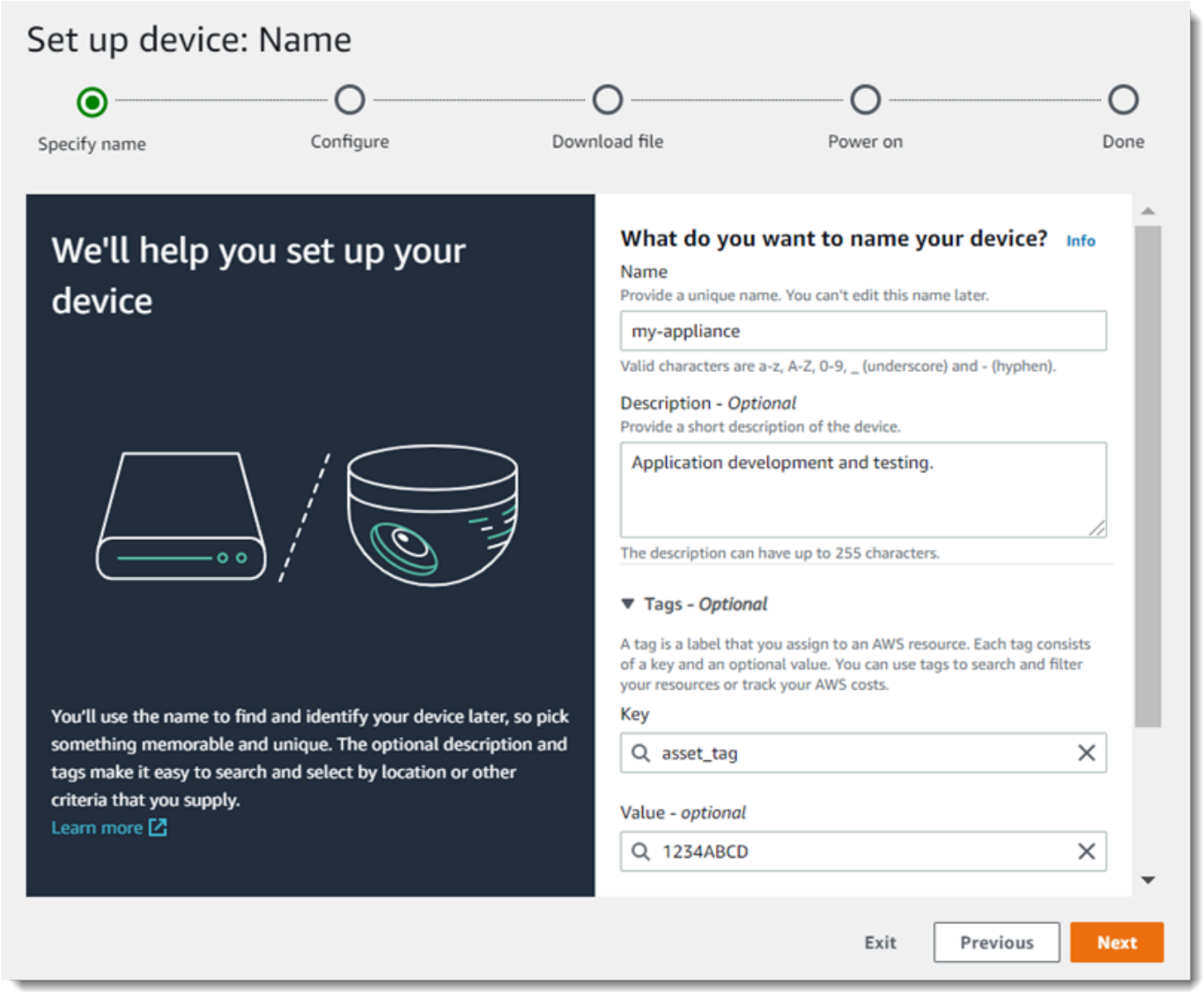
AWS Panorama Appliance 是一種硬體裝置，可透過本機網路連線連線至已啟用網路的攝影機。它使用以 Linux 為基礎的作業系統，其中包含 AWS Panorama 應用程式 SDK 和支援執行電腦視覺應用程式的軟體。

若要連線至 AWS 以進行設備管理和應用程式部署，設備會使用裝置憑證。您可以使用 AWS Panorama 主控台來產生佈建憑證。設備使用此臨時憑證來完成初始設定並下載永久裝置憑證。

> **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。

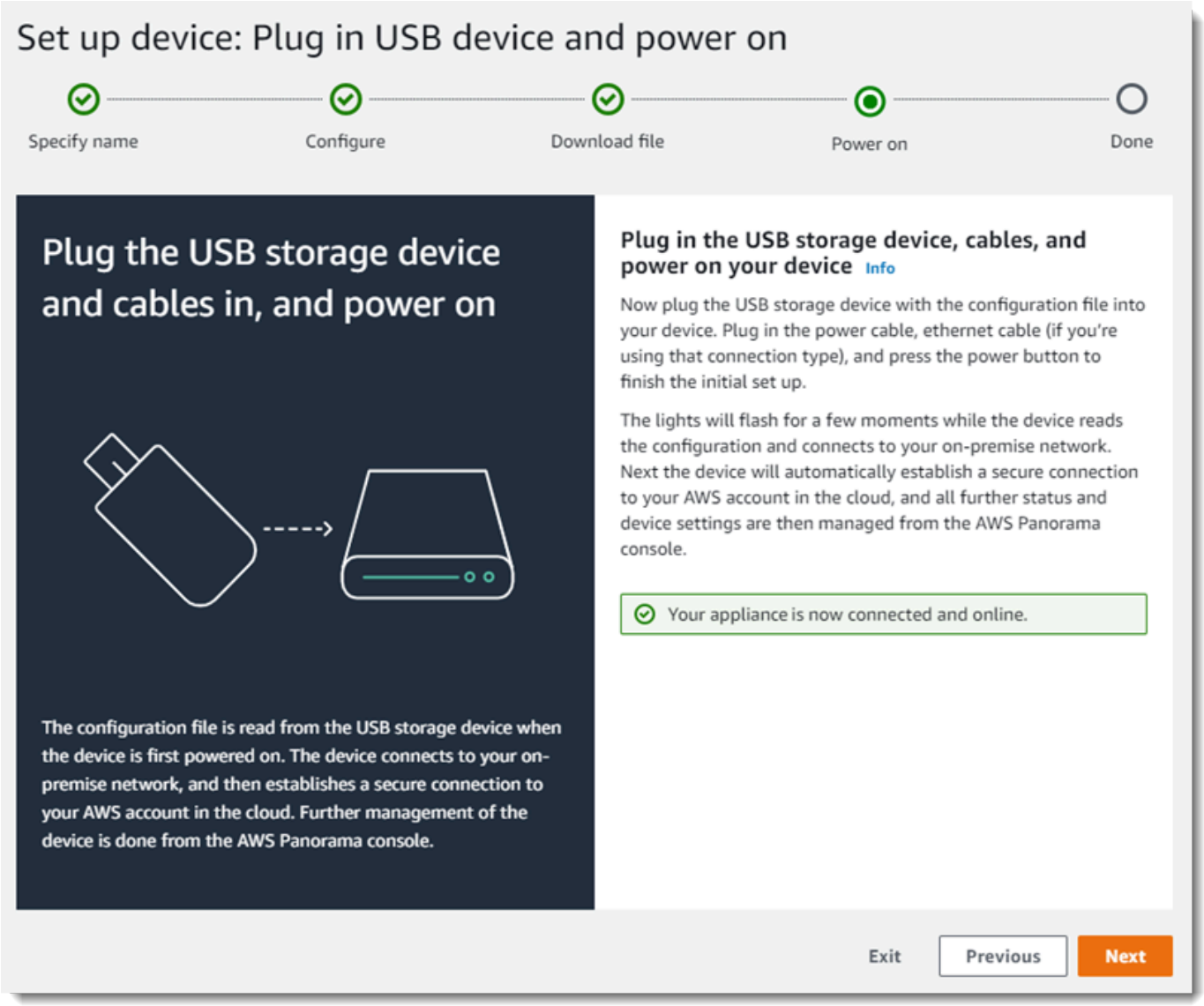
註冊設備

1. 將 USB 隨身碟連接至您的電腦。連接網路和電源線以準備設備。設備會開啟電源，並等待 USB 磁碟機連線。
2. 開啟 AWS Panorama 主控台入門頁面。
3. 選擇新增裝置。
4. 選擇開始設定。
5. 輸入代表 AWS Panorama 中設備的裝置資源名稱和描述。選擇下一步



6. 如果您需要手動指派 IP 地址、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 Next (下一步)。
8. 將組態封存複製到 USB 磁碟機的根目錄。
9. 將 USB 隨身碟連接到設備正面的 USB 3.0 連接埠，HDMI 連接埠旁。

   當您連接 USB 隨身碟時，設備會將組態封存檔和網路組態檔案複製到本身，並連接到 AWS 雲端。設備的狀態指示燈會在完成連線時從綠色變成藍色，然後回到綠色。
10. 若要繼續，請選擇 Next (下一步)。



11. 選擇完成。

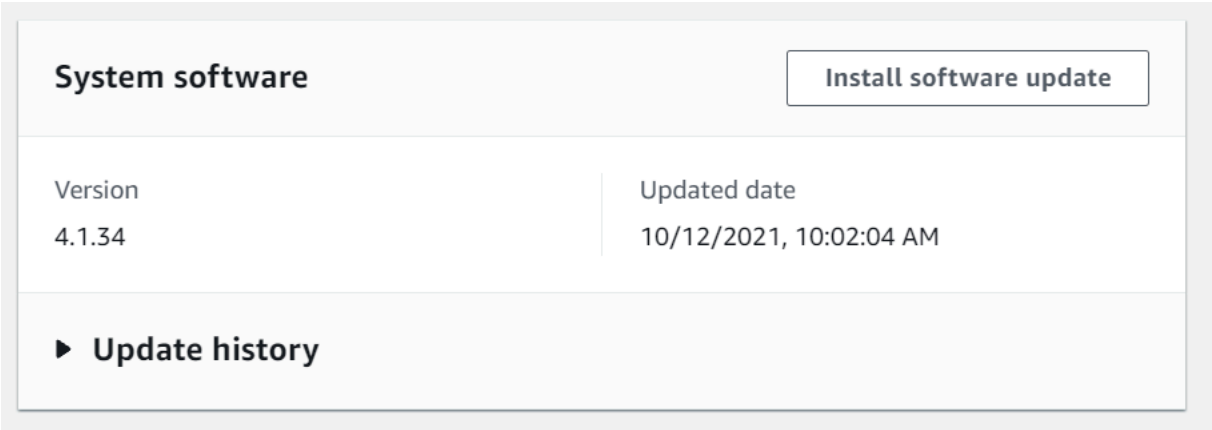
升級設備軟體

AWS Panorama Appliance 有數個軟體元件，包括 Linux 作業系統、AWS Panorama 應用程式 SDK，以及支援電腦視覺程式庫和架構。為了確保您可以將最新的功能和應用程式與設備搭配使用，請在設定後和有更新可用時升級其軟體。

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**
>
> 繼續之前，請從設備移除 USB 隨身碟，並將其格式化以刪除其內容。組態封存包含敏感資料，不會自動刪除。

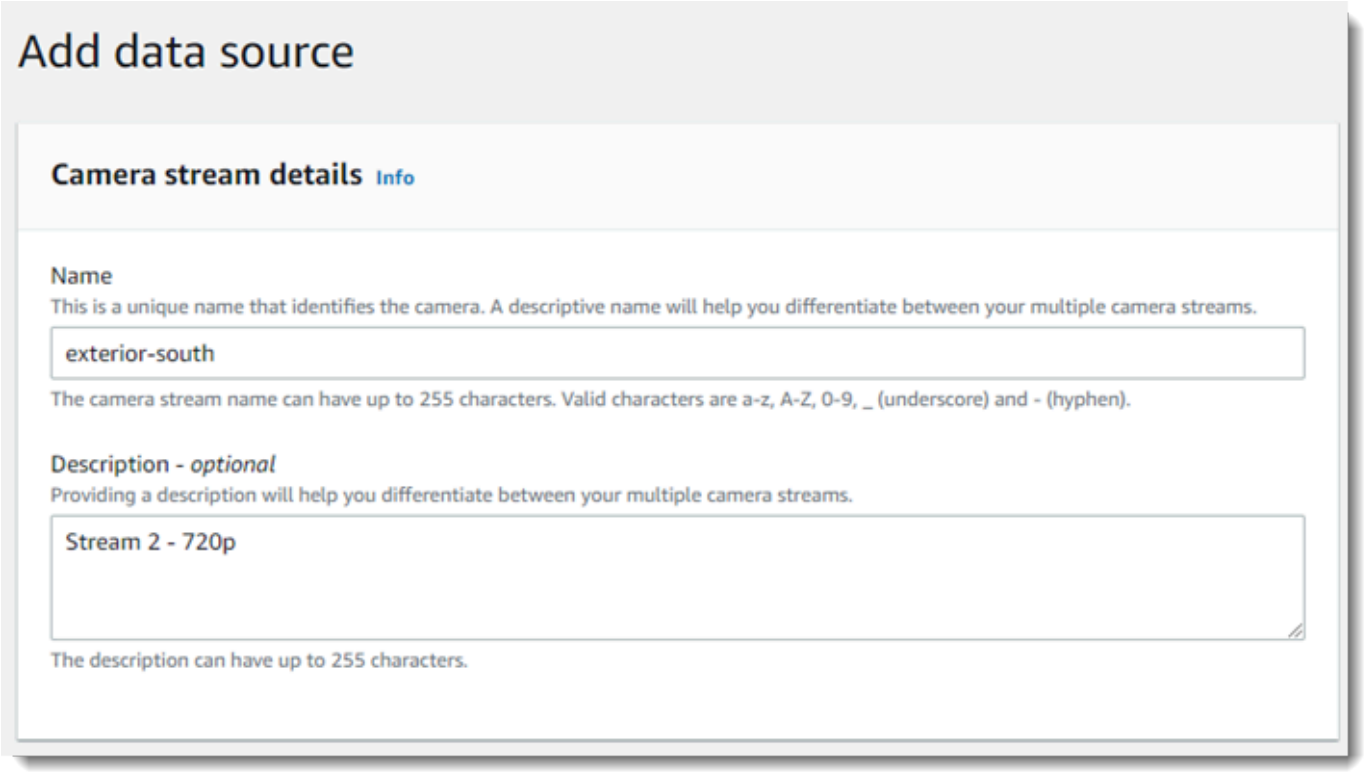
升級程序可能需要 30 分鐘或更長時間。您可以在 AWS Panorama 主控台或連接的監視器上監控其進度。程序完成時，設備會重新啟動。

新增攝影機串流

接著，向 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 – 攝影機、其位置或其他詳細資訊的簡短描述。
   - RTSP URL – 指定攝影機 IP 地址和串流路徑的 URL。例如 `rtsp://192.168.0.77/live/mpeg4/`
   - 登入資料 – 如果攝影機串流受密碼保護，請指定使用者名稱和密碼。
4. 選擇 Save (儲存)。

AWS Panorama 會安全地存放相機的登入資料 AWS Secrets Manager。多個應用程式可以同時處理相同的攝影機串流。

後續步驟

如果您在設定期間遇到錯誤，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

在您設定 AWS Panorama 設備或相容裝置並升級其軟體之後，請部署範例應用程式。在下列各節中，您可以使用 AWS Panorama Application CLI 匯入範例應用程式，並使用 AWS Panorama 主控台部署該應用程式。

範例應用程式使用機器學習模型來分類來自網路攝影機的影片影格中的物件。它使用 AWS Panorama 應用程式 SDK 載入模型、取得影像，以及執行模型。然後，應用程式會將結果疊加在原始影片上，並將其輸出到連接的顯示器。

在零售設定中，分析流量模式可讓您預測流量層級。透過將分析與其他資料結合，您可以針對假日和其他事件的更多人員配置需求進行規劃、測量廣告和銷售促銷的有效性，或最佳化顯示器配置和庫存管理。

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。在程式碼清單中，命令前面會加上提示符號 ($)，並在適當時顯示目前目錄的名稱。

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

對於長命令，我們使用逸出字元 (\) 將命令分割成多行。

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 中設定開發環境。

您可以使用 Python 開發 AWS Panorama 應用程式，並使用 Python 套件管理員 pip 安裝工具。如果您還沒有 Python，請安裝最新版本。如果您有 Python 3 但沒有 pip，請使用作業系統的套件管理員安裝 pip，或安裝隨 pip 提供的 Python 新版本。

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 Docker 網站安裝 Docker：取得 Docker

本教學課程使用 AWS Panorama 應用程式 CLI 來匯入範例應用程式、建置套件和上傳成品。AWS Panorama Application CLI 使用 AWS Command Line Interface (AWS CLI) 呼叫服務 API 操作。如果您已有 AWS CLI，請將其升級至最新版本。若要安裝 AWS Panorama 應用程式 CLI 和 AWS CLI，請使用 `pip`。

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

下載範例應用程式，並將其擷取到您的工作區。

- 應用程式範例 – aws-panorama-sample.zip

匯入範例應用程式

若要匯入範例應用程式以用於您的帳戶，請使用 AWS Panorama Application CLI。應用程式資料夾和資訊清單包含預留位置帳戶號碼的參考。若要使用您的帳戶號碼更新這些項目，請執行 `panorama-cli import-application`命令。

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

`packages` 目錄中的`SAMPLE_CODE`套件包含應用程式的程式碼和組態，包括使用應用程式基礎映像的 Dockerfile`panorama-application`。若要建置在設備上執行的應用程式容器，請使用 `panorama-cli build-container`命令。

```
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 Application CLI 的最後一步是註冊應用程式的程式碼和模型節點，並將資產上傳到服務提供的 Amazon S3 存取點。資產包含程式碼的容器映像、模型，以及每個的描述項檔案。若要註冊節點並上傳資產，請執行 `panorama-cli package-application`命令。

```
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`graphs/aws-panorama-sample/graph.json`到文字編輯器中。選擇 Next (下一步)。
4. 在應用程式名稱中，輸入 `aws-panorama-sample`。
5. 選擇繼續部署。
6. 選擇開始部署。
7. 選擇下一步，而不選取角色。
8. 選擇選取裝置，然後選擇您的設備。選擇 Next (下一步)。
9. 在選取資料來源步驟中，選擇檢視輸入，然後將攝影機串流新增為資料來源。選擇 Next (下一步)。
10. 在設定步驟中，選擇下一步。
11. 選擇部署，然後選擇完成。
12. 在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控部署。

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 Amazon CloudWatch Logs 中的應用程式和裝置日誌。

檢視輸出

部署完成時，應用程式會開始處理影片串流，並將日誌傳送至 CloudWatch。

在 CloudWatch Logs 中檢視日誌

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

- 裝置日誌 – /aws/panorama/devices/*device-id*
- 應用程式日誌 – /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影片輸出，請使用 HDMI 纜線將設備連接到監視器。根據預設，應用程式會顯示任何可信度超過 20% 的分類結果。

Example squeezenet_classes.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",
...
```

範例模型有 1000 個類別，包括許多動物、食物和常見物件。嘗試將相機指向鍵盤或咖啡馬克杯。

為了簡化，範例應用程式使用輕量型分類模型。模型會輸出單一陣列，並具有其每個類別的機率。真實世界應用程式更頻繁地使用具有多維度輸出的物件偵測模型。如需使用更複雜模型的範例應用程式，請參閱應用程式、指令碼和範本範例。

啟用適用於 Python 的 SDK

範例應用程式使用 適用於 Python (Boto) 的 AWS SDK 將指標傳送至 Amazon CloudWatch。若要啟用此功能，請建立 角色，授予應用程式傳送指標的許可，並在連接角色的情況下重新部署應用程式。

範例應用程式包含 CloudFormation 範本，可建立具有所需許可的角色。若要建立角色，請使用 `aws cloudformation deploy`命令。

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

重新部署應用程式

1. 開啟 AWS Panorama 主控台部署的應用程式頁面。
2. 選擇應用程式。
3. 選擇 Replace (取代)。
4. 完成部署應用程式的步驟。在指定 IAM 角色中，選擇您建立的角色。其名稱開頭為 `aws-panorama-sample-runtime`。
5. 部署完成時，請開啟 CloudWatch 主控台，並在AWSPanoramaApplication命名空間中檢視指標。每 150 個影格，應用程式會記錄並上傳影格處理和推論時間的指標。

清除

如果您已完成使用範例應用程式，您可以使用 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 主控台部署的應用程式頁面。
2. 選擇應用程式。
3. 選擇從裝置刪除。

後續步驟

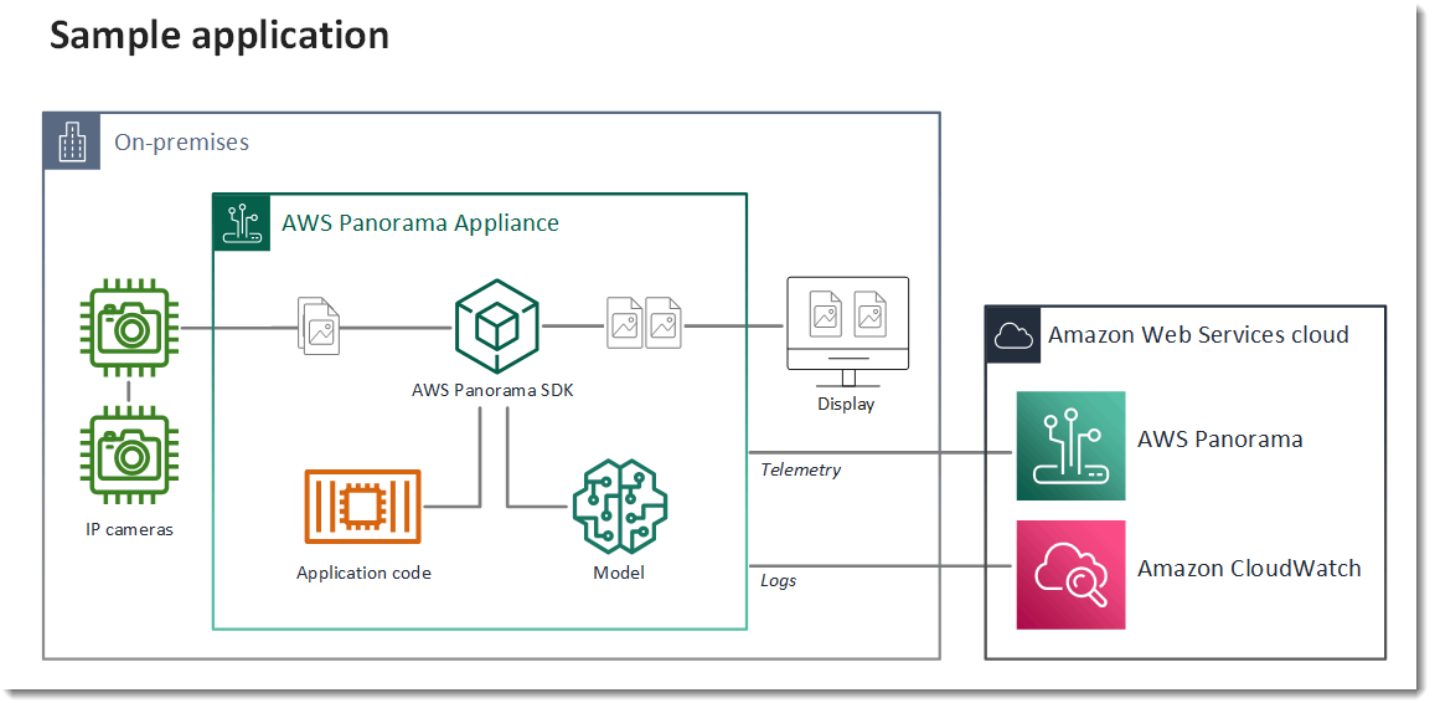
如果您在部署或執行範例應用程式時發生錯誤，請參閱 疑難排解。

若要進一步了解範例應用程式的功能和實作，請繼續下一個主題。

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，以及做為您自己應用程式的起點。

下圖顯示在 AWS Panorama 設備上執行之應用程式的主要元件。應用程式程式碼使用 AWS Panorama 應用程式 SDK 來取得映像，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至已連線的顯示器，但不會將影像資料傳送到本機網路外部。



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影格、預先處理影片資料，並將資料傳送至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

應用程式資訊清單是在 graph.json graphs 資料夾中名為 的檔案。資訊清單會定義應用程式的元件，即套件、節點和邊緣。

套件是應用程式程式碼、模型、攝影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 packages目錄中的應用程式內定義。它們包含此應用程式特定的程式碼和模型。第二個套件是 AWS Panorama 服務提供的一般攝影機和顯示套件。abstract_rtsp_media_source 套件是您在部署期間覆寫之攝影機的預留位置。hdmi_data_sink 套件代表裝置上的 HDMI 輸出連接器。

節點是套件的界面，以及具有您在部署時間覆寫之預設值的非套件參數。程式碼和模型套件在檔案中定義指定輸入和輸出package.json的界面，可以是影片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套件的 界面。

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

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

此界面在套件組態檔案 中定義package.json。界面指定套件是商業邏輯，並且需要名為 的影片串流video_in和名為 threshold 的浮點數做為輸入。介面也指定程式碼需要名為 的影片串流緩衝，video_out才能將影片輸出至顯示器

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"
                    }
                ]
            }
        ]
```

```
    }
}
```

回到應用程式資訊清單，camera_node節點代表來自攝影機的視訊串流。它包含裝飾項目，在您部署應用程式時會顯示在主控台中，提示您選擇攝影機串流。

Example **graphs/aws-panorama-sample/graph.json** – 攝影機節點

```
            {
                "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使用的可信度閾值參數。其預設值為 60，可在部署期間覆寫。

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."
                }
            }
```

應用程式資訊清單的最後一個區段 edges會在節點之間進行連線。攝影機的視訊串流和閾值參數會連接到程式碼節點的輸入，而來自程式碼節點的視訊輸出則會連接到顯示器。

Example **graphs/aws-panorama-sample/graph.json** – 邊緣

```
        "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

您可以使用範例應用程式做為您自己的應用程式的起點。

每個套件的名稱在您的帳戶中必須是唯一的。如果您和帳戶中的另一個使用者都使用通用套件名稱，例如 code或 model，則部署時可能會收到錯誤的套件版本。將程式碼套件的名稱變更為代表您應用程式的名稱。

重新命名程式碼套件

1. 重新命名套件資料夾：packages/123456789012-*SAMPLE_CODE*-1.0/。
2. 在下列位置更新套件名稱。

   - 應用程式資訊清單 – graphs/aws-panorama-sample/graph.json
   - 套件組態 – packages/123456789012-SAMPLE_CODE-1.0/package.json
   - 組建指令碼 – 3-build-container.sh

更新應用程式的程式碼

1. 在 中修改應用程式碼packages/123456789012-SAMPLE_CODE-1.0/src/application.py。
2. 若要建置容器，請執行 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 會自動從 `assets` 資料夾刪除舊容器資產，並更新套件組態。

3. 若要上傳套件，請執行 `4-package-application.py`。
4. 開啟 AWS Panorama 主控台部署的應用程式頁面。
5. 選擇應用程式。
6. 選擇 Replace (取代)。
7. 完成部署應用程式的步驟。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

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，並使用 AWS Panorama 應用程式 CLI 將其匯入為資產。

下列範例使用 MXNet SSD ResNet50 模型，您可以從本指南的 GitHub 儲存庫下載：ssd_512_resnet50_v1_voc.tar.gz

變更範例應用程式的模型

1. 重新命名套件資料夾以符合您的模型。例如，到 packages/*123456789012*-*SSD_512_RESNET50_V1_VOC*-1.0/。
2. 在下列位置更新套件名稱。

   - 應用程式資訊清單 – graphs/aws-panorama-sample/graph.json
   - 套件組態 – packages/*123456789012*-*SSD_512_RESNET50_V1_VOC*-1.0/package.json
3. 在套件組態檔案中 (package.json)。將assets值變更為空白陣列。

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

4. 開啟套件描述項檔案 (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 ]
            }
        ]
    }
}
```

   形狀的值 表示模型作為輸入 (1) 拍攝的影像數量1,3,512,512、每個影像中的通道數量 (3-紅色、綠色和藍色），以及影像的維度 (512 x 512)。陣列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 Application CLI 會將模型和描述項檔案複製到具有唯一名稱的assets資料夾中，並更新套件組態。

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

```
    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)
```

根據您的模型，您可能還需要更新 `preprocess`方法。

預先處理映像

在應用程式將映像傳送至模型之前，它會調整其大小並正規化顏色資料，以準備進行推論。應用程式使用的模型需要具有三個顏色通道的 224 x 224 像素影像，以符合其第一層中的輸入數目。應用程式透過將其轉換為介於 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，它會結合顏色通道，並將其轉換為模型可以處理的 NumPy 陣列。

Example https：//application.py – 預先處理

```
 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)
```

此程序提供以 0 為中心的可預測範圍內的模型值。它符合訓練資料集中套用至影像的預先處理，這是標準方法，但可能因模型而異。

使用適用於 Python 的 SDK 上傳指標

範例應用程式使用適用於 Python 的 SDK 將指標上傳至 Amazon CloudWatch。

Example https：//application.py – 適用於 Python 的 SDK

```
    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.")
```

它從您在部署期間指派的執行期角色取得許可。角色在 `aws-panorama-sample.yml` CloudFormation 範本中定義。

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/
```

範例應用程式會使用 pip 安裝適用於 Python 的 SDK 和其他相依性。當您建置應用程式容器時，會`Dockerfile`執行命令，在基礎映像隨附的內容上安裝程式庫。

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
```

若要在應用程式程式碼中使用 AWS SDK，請先修改範本，以新增應用程式使用之所有 API 動作的許可。`1-create-role.sh` 每次進行變更時，請執行 來更新 CloudFormation 堆疊。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，最佳實務是透過在個別陳述式`Resource`中指定目標的名稱或模式，將此政策的範圍降至最低。如需每個服務支援的動作和資源的詳細資訊，請參閱服務授權參考中的動作、資源和條件索引鍵

後續步驟

如需使用 AWS Panorama 應用程式 CLI 從頭建置應用程式和建立套件的說明，請參閱 CLI 的 README。

- github.com/aws/aws-panorama-cli

如需更多範例程式碼和測試公用程式，您可以在部署之前用來驗證應用程式程式碼，請造訪 AWS Panorama 範例儲存庫。

- github.com/aws-samples/aws-panorama-samples

支援的電腦視覺模型和攝影機

AWS Panorama 支援使用 PyTorch、Apache MXNet 和 TensorFlow 建置的模型。當您部署應用程式時，AWS Panorama 會在 SageMaker AI Neo 中編譯模型。只要使用與 Amazon SageMaker SageMaker AI 或開發環境中建置模型。

為了處理影片並取得影像以傳送至模型，AWS Panorama 設備會使用 RTSP 通訊協定連線至 H.264 編碼的影片串流。AWS Panorama 會測試各種常用攝影機的相容性。

章節

支援的模型

當您為 AWS Panorama 建置應用程式時，您會提供機器學習模型，供應用程式用於電腦視覺。您可以使用模型架構、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模型。

> **Note**
>
> 如需已使用 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 AI Neo 編譯器來編譯您的電腦視覺模型。SageMaker AI Neo 是一種編譯器，可將模型最佳化，以便在目標平台上有效率地執行，而目標平台可以是 Amazon Elastic Compute Cloud (Amazon EC2) 中的執行個體，或像是 AWS Panorama Appliance 等邊緣裝置。

AWS Panorama 支援 SageMaker AI Neo 支援邊緣裝置的 PyTorch、Apache MXNet 和 TensorFlow 版本。當您建置自己的模型時，您可以使用 SageMaker AI Neo 版本備註中列出的架構版本。在 SageMaker AI 中，您可以使用內建影像分類演算法。

如需在 AWS Panorama 中使用模型的詳細資訊，請參閱 電腦視覺模型。

支援的攝影機

AWS Panorama 設備支援來自透過本機網路輸出 RTSP 之攝影機的 H.264 視訊串流。對於大於 200 萬像素的攝影機串流，設備會將影像縮減至 1920x1080 像素，或保留串流長寬比的同等大小。

下列攝影機模型已通過與 AWS Panorama 設備的相容性測試：

- 軸 – M3057-PLVE, M3058-PLVE, P1448-LE, P3225-LV Mk II
- LaView – LV-PB3040W
- Vivotek – IB9360-H
- Amcrest – IP2M-841B
- Anpviz – IPC-B850W-S-3X, IPC-D250W-S
- WGCC - 半球 PoE 4MP 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 設備具有下列硬體規格。如需其他相容的裝置，請參閱製造商的文件。

元件	規格
處理器和 GPU	Nvidia Jetson AGX Xavier 搭配 32GB RAM
乙太網路	2x 1000 Base-T (Gigabyte)
USB	1 個 USB 2.0 和 1 個 USB 3.0 Type-A 母頭
HDMI 輸出	2.0a
維度	7.75 英吋 x 9.6 英吋 x 1.6 英吋 (197 公釐 x 243 公釐 x 40 公釐)
Weight	3.7 磅 (1.7 公斤)
電源供應器	100V-240V 50-60Hz AC 65W
電源輸入	IEC 60320 C6 (3 接腳) 插座
灰塵和液體保護	IP-62
EMI/EMC 法規合規	FCC Part-15 (美國)
熱觸控限制	IEC-62368
操作溫度	-20°C 到 60°C
操作濕度	0% 到 95% RH
儲存溫度	-20°C 到 85°C
儲存濕度	無法控制低溫。90% RH，在高溫下
冷卻	強制空氣熱擷取 (風扇)
安裝選項	Rackmount 或獨立

元件	規格
電源線	6 英尺 (1.8 公尺)
電源控制	按鈕
Reset	瞬間切換
狀態和網路 LEDs	可程式設計的 3 顏色 RGB LED

Wi-Fi、藍牙和 SD 卡儲存存在於設備上，但無法使用。

AWS Panorama 設備包含兩個螺絲，用於安裝在伺服器機架上。您可以在 19 英吋機架上side-by-side掛兩個設備。

Service Quotas

AWS Panorama 會將配額套用至您在帳戶中建立的資源，以及您部署的應用程式。如果您在多個 AWS 區域中使用 AWS Panorama，配額會分別套用至每個區域。AWS Panorama 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。

- 裝置 – 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。
- 應用程式執行個體 – 每個裝置最多 10 個應用程式。每個應用程式最多可監控 8 個攝影機串流。每個裝置的部署限制為每天 200 個。

當您搭配 AWS Panorama 服務使用 AWS Panorama 應用程式 CLI AWS Command Line Interface或 AWS SDK 時，配額會套用至您進行的 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 操作子集會套用每秒 1 個請求的額外限制。

如需配額的完整清單，請造訪 Service Quotas 主控台，或參閱 中的 AWS Panorama 端點和配額 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 AWS Panorama 服務和資源的存取，例如設備和應用程式。對於您帳戶中使用 的使用者 AWS Panorama，您可以在可套用至 IAM 角色的許可政策中管理許可。若要管理應用程式的許可，您可以建立角色並將其指派給應用程式。

若要管理帳戶中使用者的許可，請使用 AWS Panorama 提供的受管政策，或自行撰寫。您需要其他 AWS 服務的許可，才能取得應用程式和設備日誌、檢視指標，以及將角色指派給應用程式。

AWS Panorama 設備也有 角色，授予其存取 AWS 服務和資源的許可。設備的 角色是 AWS Panorama 服務用來代表您存取其他 服務的其中一個服務角色。

應用程式角色是您為應用程式建立的個別服務角色，以授予它搭配 使用 AWS 服務的許可 適用於 Python (Boto) 的 AWS SDK。若要建立應用程式角色，您需要管理員的管理權限或 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，您可以指定應用程式 Amazon Resource Name (ARN) 的模式，要求使用者在其建立的應用程式名稱中包含其使用者名稱。如需每個動作支援的資源和條件，請參閱《服務授權參考》中的 的動作、資源和條件索引鍵 AWS Panorama。

如需詳細資訊，請參閱《IAM 使用者指南》中的什麼是 IAM？。

主題

- AWS Panorama 的身分型 IAM 政策
- AWS Panorama 服務角色和跨服務資源
- 將許可授予應用程式

AWS Panorama 的身分型 IAM 政策

若要授予您帳戶中的使用者對 AWS Panorama 的存取權，請在 AWS Identity and Access Management (IAM) 中使用身分型政策。將身分型政策套用至與使用者相關聯的 IAM 角色。您也可以授予另一個帳戶中的使用者在帳戶中擔任角色，並存取您的 AWS Panorama 資源。

AWS Panorama 提供受管政策，可授予對 AWS Panorama API 動作的存取權，以及在某些情況下，存取用於開發和管理 AWS Panorama 資源的其他 服務。AWS Panorama 會視需要更新受管政策，以確保您的使用者在發佈新功能時能夠存取新功能。

- AWSPanoramaFullAccess – 提供 AWS Panorama、Amazon S3 中的 AWS Panorama 存取點、 中的設備憑證 AWS Secrets Manager和 Amazon CloudWatch 中的設備日誌的完整存取權。包含為 AWS Panorama 建立服務連結角色的許可。檢視政策

此`AWSPanoramaFullAccess`政策可讓您標記 AWS Panorama 資源，但沒有 AWS Panorama 主控台使用的所有標籤相關許可。若要授予這些許可，請新增下列政策。

- ResourceGroupsandTagEditorFullAccess – 檢視政策

`AWSPanoramaFullAccess` 政策不包含從 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 – 檢視政策

受管政策會授予 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。使用完整存取政策做為政策的起點。

> **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 設備所使用的服務角色。服務角色提供服務許可，以管理資源或與其他服務互動。先建立此角色，再授予使用者存取權。

如需可用來限制 AWS Panorama 中使用者許可範圍的資源和條件詳細資訊，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。服務角色提供服務許可，以管理 資源或與其他服務互動。當您第一次登入 AWS Panorama 主控台時，您可以建立下列服務角色：

- AWSServiceRoleForAWSPanorama – 允許 AWS Panorama 管理 AWS IoT、AWS Secrets Manager 和 AWS Panorama 中的資源。

  受管政策：AWSPanoramaServiceLinkedRolePolicy
- AWSPanoramaApplianceServiceRole – 允許 AWS Panorama 設備將日誌上傳至 CloudWatch，並從 AWS Panorama 建立的 Amazon S3 存取點取得物件。

  受管政策：AWSPanoramaApplianceServiceRolePolicy

若要檢視連接到每個角色的許可，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 使用之命名模式的資源。例如， 僅`AWSServiceRoleForAWSPanorama`授予 服務存取其名稱`panorama`中具有 之 AWS IoT 資源的許可。

章節

保護設備角色

AWS Panorama 設備會使用 `AWSPanoramaApplianceServiceRole`角色來存取您帳戶中的資源。設備有權將日誌上傳至 CloudWatch Logs、讀取攝影機串流憑證 AWS Secrets Manager，以及存取 AWS Panorama 建立的 Amazon Simple 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許可。若要授予應用程式使用 AWS 服務的許可，請建立應用程式角色。

AWS Panorama 會將相同的服務角色與帳戶中的所有設備搭配使用，而不會跨帳戶使用角色。為了增加一層安全性，您可以修改設備角色的信任政策以明確強制執行此政策，這是當您使用角色授予服務存取帳戶中資源的許可時最佳實務。

更新設備角色信任政策

1. 在 IAM 主控台中開啟設備角色：AWSPanoramaApplianceServiceRole
2. 選擇編輯信任關係。
3. 更新政策內容，然後選擇更新信任政策。

下列信任政策包含一個條件，可確保當 AWS Panorama 擔任設備角色時，會為您帳戶中的設備執行此操作。`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信任政策 – 特定帳戶

JSON

```
{
  "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，並允許它只擔任具有特定裝置的角色，您可以依 ARN 指定裝置。`aws:SourceArn` 條件會將 AWS Panorama 指定的設備 ARN 與您包含在政策中的設備 ARN 進行比較。

Example信任政策 – 單一設備

JSON

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

如果您重設並重新佈建設備，則必須暫時移除來源 ARN 條件，然後使用新的裝置 ID 再次新增。

如需這些條件的詳細資訊，以及服務使用 角色存取帳戶中資源時的安全最佳實務，請參閱《IAM 使用者指南》中的混淆代理人問題。

使用其他 服務

AWS Panorama 會在下列服務中建立或存取資源：

- AWS IoT – AWS Panorama 設備的物件、政策、憑證和任務
- Amazon S3 – 預備應用程式模型、程式碼和組態的存取點。
- Secrets Manager – AWS Panorama 設備的短期登入資料。

如需有關每個服務的 Amazon Resource Name (ARN) 格式或許可範圍的資訊，請參閱此清單中連結至的 IAM 使用者指南中的主題。

將許可授予應用程式

您可以為您的應用程式建立角色，以授予它呼叫 AWS 服務的許可。根據預設，應用程式沒有任何許可。您可以在 IAM 中建立應用程式角色，並在部署期間將其指派給應用程式。若要僅授予應用程式所需的許可，請為應用程式建立具有特定 API 動作許可的角色。

範例應用程式包含建立應用程式角色的 CloudFormation 範本和指令碼。這是 AWS Panorama 可擔任的服務角色。此角色授予應用程式呼叫 CloudWatch 上傳指標的許可。

Example aws-panorama-sample.yml – 應用程式角色

```
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/
```

您可以透過指定 值的 API 動作或模式清單，延伸此指令碼以將許可授予其他 服務`Action`。

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

管理 AWS Panorama 設備

AWS Panorama 設備是執行您應用程式的硬體。您可以使用 AWS Panorama 主控台來註冊設備、更新其軟體，以及將應用程式部署到其中。 AWS Panorama 設備上的軟體會連線至攝影機串流、將影片影格傳送至您的應用程式，並在連接的顯示器上顯示影片輸出。

設定設備或其他相容裝置後，您可以註冊攝影機以搭配應用程式使用。您可以在 主控台中管理攝影機串流。 AWS Panorama 部署應用程式時，您可以選擇設備傳送給它的攝影機串流進行處理。

如需使用範例應用程式介紹 AWS Panorama 設備的教學課程，請參閱 入門 AWS Panorama。

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 設備和其他相容的裝置。

若要設定設備，請遵循入門教學中的指示。設定程序會在 AWS Panorama 中建立資源，以追蹤您的設備並協調更新和部署。

若要向 AWS Panorama API 註冊設備，請參閱 自動化裝置註冊。

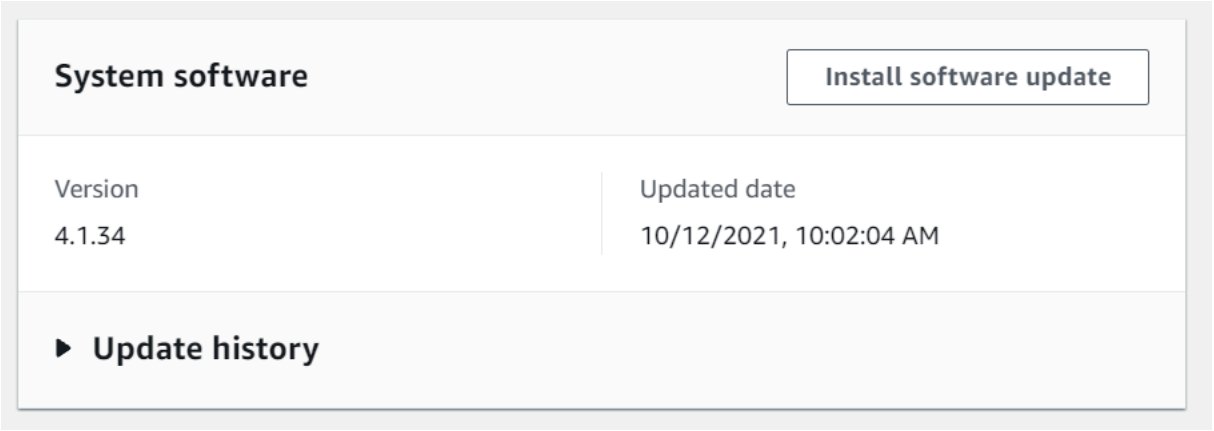
章節

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。更新可以是必要或選用。有可用的必要更新時，主控台會提示您套用。您可以在設備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，您可以使用 AWS Panorama 主控台取消註冊並刪除相關聯的 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 刪除 。
4. 輸入設備的名稱，然後選擇刪除。

當您從 AWS Panorama 服務刪除設備時，不會自動刪除設備上的資料。取消註冊的設備無法連線至 AWS 服務，而且在重設之前，無法再次註冊。

重新啟動設備

您可以遠端重新啟動設備。

重新啟動設備

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設備以重新啟動它。若要接收訊號，設備必須能夠連線到 AWS IoT。若要使用 AWS Panorama API 重新啟動設備，請參閱 重新啟動設備。

重設設備

若要在不同區域或不同帳戶中使用 設備，您必須將其重設，並使用新憑證重新佈建。重設裝置會套用最新的必要軟體版本，並刪除所有帳戶資料。

若要開始重設操作，設備必須插入電源並關閉。同時按住電源和重設按鈕 5 秒。當您放開按鈕時，狀態指示燈會閃爍橘色。等到狀態指示燈閃爍綠色，再佈建或中斷設備連線。

您也可以重設設備軟體，而無需從裝置刪除憑證。如需詳細資訊，請參閱電源和重設按鈕。

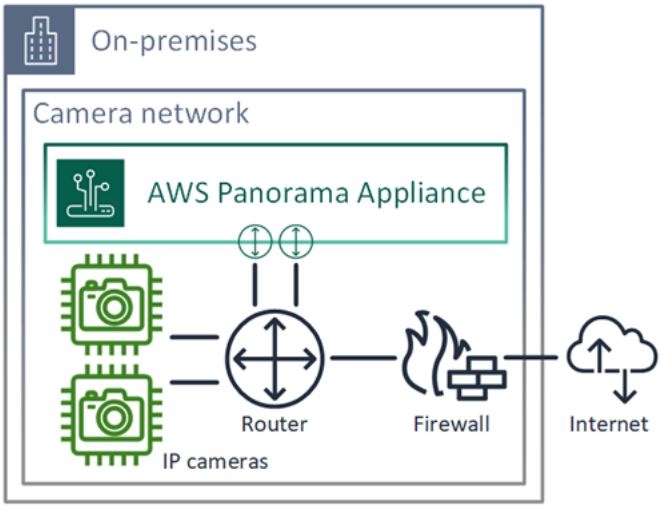
將 AWS Panorama 設備連接到您的網路

AWS Panorama 設備需要連線至 AWS 雲端和 IP 攝影機的內部部署網路。您可以將設備連接到同時授予存取權的單一防火牆，或將裝置的兩個網路介面分別連接到不同的子網路。無論哪種情況，您都必須保護設備的網路連線，以防止未經授權存取攝影機串流。

章節

單一網路組態

設備有兩個乙太網路連接埠。如果您透過單一路由器路由所有往返裝置的流量，您可以使用第二個連接埠進行備援，以防實體連線至第一個連接埠中斷。設定您的路由器，以允許設備僅連線到攝影機串流和網際網路，並封鎖攝影機串流，否則離開您的內部網路。

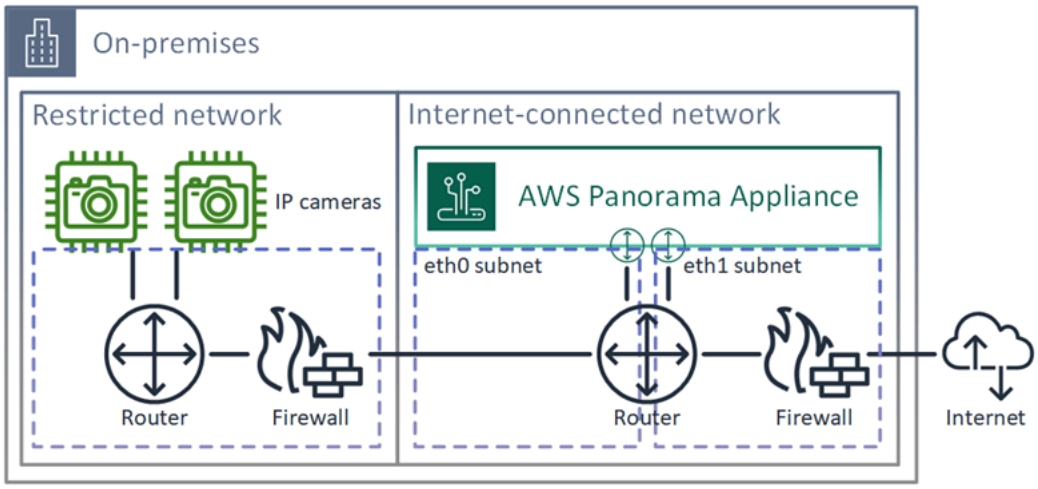


如需設備需要存取的連接埠和端點的詳細資訊，請參閱 設定服務存取和 設定本機網路存取。

雙網路組態

若要多一層安全性，您可以將設備放置在與攝影機網路分開的網際網路連線網路中。受限攝影機網路與設備網路之間的防火牆僅允許設備存取影片串流。如果您的攝影機網路之前已進行氣隙移植以求安全，則相較於將攝影機網路連線到也授予網際網路存取權的路由器，您可能更喜歡此方法。

下列範例顯示設備連接到每個連接埠上的不同子網路。路由器會將`eth0`介面放置在路由至攝影機網路的子網路上，以及路由至網際網路的子網路`eth1`上。



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

設定服務存取

在佈建期間，您可以設定設備來請求特定的 IP 地址。事先選擇 IP 地址，以簡化防火牆組態，並確保設備地址不會在長時間離線時變更。

設備使用 AWS 服務來協調軟體更新和部署。設定您的防火牆以允許設備連線到這些端點。

網際網路存取

- AWS IoT (HTTPS 和 MQTT、連接埠 443、8443 和 8883) – AWS IoT Core 和裝置管理端點。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 登入資料 (HTTPS、連接埠 443) – `credentials.iot.<region>.amazonaws.com`和子網域。
- Amazon Elastic Container Registry (HTTPS、連接埠 443) – `api.ecr.<region>.amazonaws.comdkr.ecr.<region>.amazonaws.com`和子網域。

- Amazon CloudWatch (HTTPS，連接埠 443) – `monitoring.<region>.amazonaws.com`。
- Amazon CloudWatch Logs (HTTPS，連接埠 443) – `logs.<region>.amazonaws.com`。
- Amazon Simple Storage Service (HTTPS、連接埠 443) – `s3.<region>.amazonaws.coms3-accesspoint.<region>.amazonaws.com`和子網域。

如果您的應用程式呼叫其他 AWS 服務，則設備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

設備需要在本機存取 RTSP 影片串流，但不需要透過網際網路存取。設定您的防火牆以允許設備在內部存取連接埠 554 上的 RTSP 串流，並不允許串流進出網際網路。

本機存取

- 即時串流通訊協定 (RTSP，連接埠 554) – 讀取攝影機串流。
- 網路時間通訊協定 (NTP，連接埠 123) – 保持設備的時鐘同步。如果您未在網路上執行 NTP 伺服器，設備也可以透過網際網路連線至公有 NTP 伺服器。

私有連線

如果您在具有 VPN 連線的私有 VPC 子網路中部署 AWS Panorama 設備，則不需要網際網路存取 AWS。您可以使用 Site-to-Site VPN 或 在內部部署路由器與 之間 Direct Connect 建立 VPN 連線 AWS。在私有 VPC 子網路中，您可以建立可讓設備連線至 Amazon Simple Storage Service AWS IoT 和其他 服務的端點。如需詳細資訊，請參閱將設備連接到私有子網路。

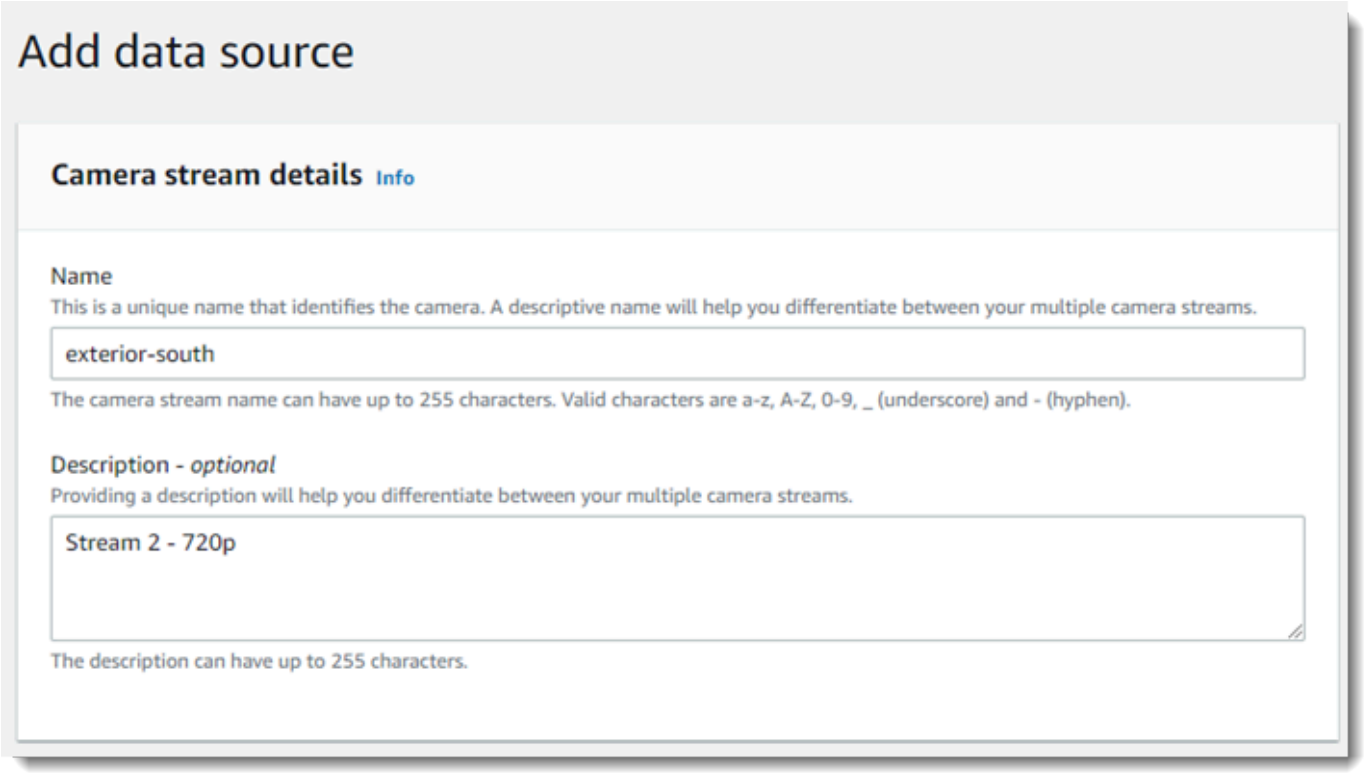
在 AWS Panorama 中管理攝影機串流

若要將影片串流註冊為應用程式的資料來源，請使用 AWS Panorama 主控台。應用程式可以同時處理多個串流，而多個設備可以連接到相同的串流。

> **Important**
>
> 應用程式可以連線到任何可從其連線的本機網路路由的攝影機串流。若要保護您的影片串流，請將您的網路設定為僅允許本機的 RTSP 流量。如需詳細資訊，請參閱AWS Panorama 的安全性。

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 – 攝影機、其位置或其他詳細資訊的簡短描述。

- RTSP URL – 指定攝影機 IP 地址和串流路徑的 URL。例如 `rtsp://192.168.0.77/live/mpeg4/`
- 登入資料 – 如果攝影機串流受密碼保護，請指定使用者名稱和密碼。

4. 選擇 Save (儲存)。

若要向 AWS Panorama API 註冊攝影機串流，請參閱 自動化裝置註冊。

如需與 AWS Panorama 設備相容的攝影機清單，請參閱 支援的電腦視覺模型和攝影機。

移除串流

您可以在 AWS Panorama 主控台中刪除攝影機串流。

移除攝影機串流

1. 開啟 AWS Panorama 主控台資料來源頁面。
2. 選擇攝影機串流。
3. 選擇刪除資料來源。

從服務移除攝影機串流不會停止執行應用程式，也不會從 Secrets Manager 刪除攝影機登入資料。若要刪除秘密，請使用 Secrets Manager 主控台。

管理 AWS Panorama 設備上的應用程式

應用程式是程式碼、模型和組態的組合。從 AWS Panorama 主控台的裝置頁面，您可以管理設備上的應用程式。

管理 AWS Panorama 設備上的應用程式

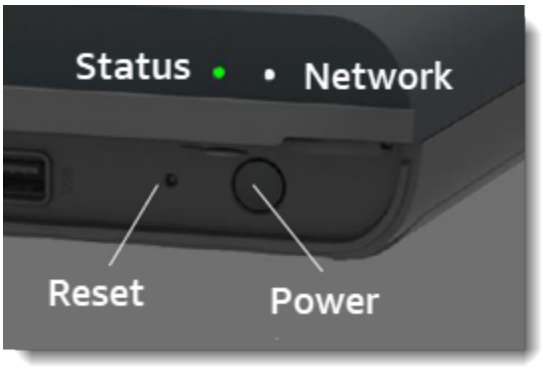
1. 開啟 AWS Panorama 主控台裝置頁面。
2. 選擇設備。

部署的應用程式頁面顯示已部署到設備的應用程式。

使用此頁面上的選項從設備中移除部署的應用程式，或將執行中的應用程式取代為新版本。您也可以複製應用程式（執行或刪除），以部署其新複本。

AWS Panorama Appliance 按鈕和燈光

AWS Panorama Appliance 在電源按鈕上方有兩個 LED 燈，指出裝置狀態和網路連線。



狀態指示燈

LEDs變更顏色並閃爍以指示狀態。慢速閃爍每三秒一次。快速閃爍是每秒一次。

狀態 LED 狀態

- 快速閃爍綠色 – 設備正在開機。
- 綠燈恒亮 – 設備正常運作。
- 緩慢閃爍藍燈 – 設備正在複製組態檔案並嘗試註冊 AWS IoT。
- 快速閃爍藍燈 – 設備正在將日誌映像複製到 USB 隨身碟。
- 快速閃爍紅色 – 設備在啟動期間遇到錯誤或過熱。
- 緩慢閃爍的橘色 – 設備正在還原最新的軟體版本。
- 快速閃爍的橘色 – 設備正在還原最低軟體版本。

網路光源

網路 LED 具有下列狀態：

網路 LED 狀態

- 綠燈恒亮 – 已連接乙太網路纜線。
- 閃爍綠色 – 設備正在透過網路進行通訊。
- 恆亮紅色 – 未連接乙太網路纜線。

電源和重設按鈕

電源和重設按鈕位於裝置正面的保護蓋下方。重設按鈕較小且凹陷。使用小型螺絲起子或迴紋針來按下它。

重設設備

1. 設備必須插入電源並關閉。若要關閉設備電源，請按住電源按鈕 1 秒，然後等待關機程序完成。關閉順序大約需要 10 秒。
2. 若要重設設備，請使用下列按鈕組合。短按是 1 秒。長按是 5 秒。對於需要多個按鈕的操作，同時按住兩個按鈕。

   - 完全重設 – 長按電源並重設。

     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 – 短按重設。

     將最新的軟體更新重新套用至設備。
   - 還原最低軟體版本 – 長按重設。

     將最新的必要軟體更新重新套用至設備。
3. 釋放兩個按鈕。設備會開啟電源，且狀態指示燈會閃爍橙色幾分鐘。
4. 當設備準備就緒時，狀態指示燈會閃爍綠色。

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

管理 AWS Panorama 應用程式

應用程式會在 AWS Panorama 設備上執行，以在影片串流上執行電腦視覺任務。您可以透過結合 Python 程式碼和機器學習模型來建置電腦視覺應用程式，並透過網際網路將其部署到 AWS Panorama 設備。應用程式可以將影片傳送至顯示器，或使用 AWS 開發套件將結果傳送至 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 Application CLI 將其匯入您的帳戶、建置容器、上傳和註冊資產，以及建立應用程式執行個體。本主題會詳細介紹每個步驟，並說明背景中發生的情況。

如果您尚未部署應用程式，請參閱 入門 AWS Panorama以取得逐步解說。

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 應用程式。

章節

- 安裝 AWS Panorama 應用程式 CLI
- 匯入應用程式
- 建置容器映像
- 匯入模型
- 上傳應用程式資產
- 使用 AWS Panorama 主控台部署應用程式
- 自動化應用程式部署

安裝 AWS Panorama 應用程式 CLI

若要安裝 AWS Panorama 應用程式 CLI 和 AWS CLI，請使用 pip。

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

若要使用 AWS Panorama 應用程式 CLI 建置應用程式映像，您需要 Docker。在 Linux 上， qemu和相關的系統程式庫也是必要的。如需安裝和設定 AWS Panorama 應用程式 CLI 的詳細資訊，請參閱專案 GitHub 儲存庫中的 README 檔案。

- github.com/aws/aws-panorama-cli

如需使用 WSL2 在 Windows 中設定建置環境的說明，請參閱 在 Windows 中設定開發環境。

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 Application CLI 匯入應用程式。

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

此命令會使用您的帳戶 ID 重新命名應用程式套件。套件名稱開頭為部署目標帳戶的帳戶 ID。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包含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。import-application 命令會重新命名這些項目，以使用 CLI 從工作區的 AWS 登入資料推論的帳戶 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 會在套件目錄名稱中，以您的帳戶 ID 取代，並在應用程式資訊清單 (graph.json) 中取代，而這些清單是參考。您可以使用 呼叫 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"
}
```

建置容器映像

您的應用程式碼封裝在 Docker 容器映像中，其中包含您在 Dockerfile 中安裝的應用程式碼和程式庫。使用 AWS Panorama Application CLI `build-container`命令來建置 Docker 映像並匯出檔案系統映像。

```
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
```

此命令會建立名為 的 Docker 映像，`code_asset`並將檔案系統匯出至 `assets` 資料夾中的`.tar.gz`封存。CLI 會從 Amazon Elastic Container Registry (Amazon ECR) 提取應用程式基礎映像，如應用程式的 Dockerfile 中所指定。

除了容器封存之外，CLI 還會為套件描述項 () 建立資產`descriptor.json`。這兩個檔案都會重新命名為唯一識別符，反映原始檔案的雜湊。AWS Panorama Application CLI 也會在記錄兩個資產名稱的套件組態中新增區塊。在部署過程中，設備會使用這些名稱。

Example package/123456789012-SAMPLE_CODE-1.0/package.json – 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"
                    },
```

build-container 命令中指定的程式碼資產名稱必須符合套件組態中 asset 欄位的值。在上述範例中，這兩個值都是 code_asset。

匯入模型

您的應用程式可能在其資產資料夾中有模型封存，或者您可以分別下載。如果您有新的模型、更新的模型或更新的模型描述項檔案，請使用 add-raw-model命令將其匯入。

```
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項檔案，則可以重複使用資產目錄中的現有模型。您可能需要更新描述項檔案，才能設定浮點精確度模式等功能。例如，以下指令碼示範如何使用範例應用程式執行此操作。

Example util-scripts/update-model-config.sh

```
#!/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
```

在您使用 CLI 重新匯入描述項檔案之前，不會套用模型套件目錄中的描述項檔案變更。CLI 會使用新的資產名稱來更新模型套件組態，類似於在重建容器時更新應用程式程式碼套件組態的方式。

上傳應用程式資產

若要上傳和註冊應用程式的資產，包括模型封存、容器檔案系統封存及其描述項檔案，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 Panorama 會為您管理存取點，並透過 DescribePackage API 提供相關資訊。CLI 會將每個套件的資產上傳至該套件提供的位置，並使用套件組態所述的設定向 AWS Panorama 服務註冊資產。

使用 AWS Panorama 主控台部署應用程式

您可以使用 AWS Panorama 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。

部署應用程式

1. 開啟 AWS Panorama 主控台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`graph.json`到文字編輯器中。選擇 Next (下一步)。
4. 輸入名稱和解壓縮。
5. 選擇繼續以部署。
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它。選擇 Next (下一步)。
8. 選擇選取裝置，然後選擇您的設備。選擇 Next (下一步)。
9. 在選取資料來源步驟中，選擇檢視輸入，然後將攝影機串流新增為資料來源。選擇 Next (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 Next (下一步)。
11. 選擇部署，然後選擇完成。
12. 在部署的應用程式清單中，選擇要監控其狀態的應用程式。

部署程序需要 15-20 分鐘。當應用程式啟動時，設備的輸出可以長時間空白。如果您遇到錯誤，請參閱 疑難排解。

自動化應用程式部署

您可以使用 CreateApplicationInstance API 自動化應用程式部署程序。API 會取得兩個組態檔案做為輸入。應用程式資訊清單會指定使用的套件及其關係。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，並設定其他應用程式特定的設定。

如需詳細資訊，以及本主題中每個步驟的範例指令碼，請參閱 自動化應用程式部署。

在 AWS Panorama 主控台中管理應用程式

使用 AWS Panorama 主控台來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。當您取代應用程式時，您可以更新其程式碼或模型。

更新應用程式

1. 開啟 AWS Panorama 主控台部署的應用程式頁面。
2. 選擇應用程式。
3. 選擇 Replace (取代)。
4. 請依照指示建立新的版本或應用程式。

還有一個複製選項，其作用類似於取代，但不會移除舊版本的應用程式。您可以使用此選項來測試應用程式的變更，而不停止執行中的版本，或重新部署您已刪除的版本。

刪除版本和應用程式

若要清除未使用的應用程式版本，請從您的設備中刪除它們。

如欲刪除應用程式

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

套件組態

當您使用 AWS Panorama Application CLI 命令 時`panorama-cli package-application`，CLI 會將您應用程式的資產上傳至 Amazon S3，並將其註冊至 AWS Panorama。資產包括二進位檔案（容器映像和模型）和描述項檔案，AWS Panorama 設備會在部署期間下載。若要註冊套件的資產，您需要提供獨立的套件組態檔案，以定義套件、其資產及其介面。

下列範例顯示具有一個輸入和一個輸出之程式碼節點的套件組態。視訊輸入可讓您從攝影機串流存取影像資料。輸出節點會將處理過的影像傳送到顯示器。

Example package/1234567890-SAMPLE_CODE-1.0/package.json

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

`assets` 本節指定 AWS Panorama Application CLI 上傳至 Amazon S3 的成品名稱。如果您從其他使用者匯入範例應用程式或應用程式，此區段可以是空的，或參考不在您帳戶中的資產。當您執行時`panorama-cli package-application`，AWS Panorama Application CLI 會以正確的值填入本節。

AWS Panorama 應用程式資訊清單

部署應用程式時，您會提供名為應用程式資訊清單的組態檔案。此檔案會將應用程式定義為具有節點和邊緣的圖形。應用程式資訊清單是應用程式原始碼的一部分，並存放在 graphs目錄中。

Example graphs/aws-panorama-sample/graph.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"
            }
        ]
    }
}
```

節點依邊緣連接，指定節點輸入和輸出之間的映射。一個節點的輸出會連接到另一個節點的輸入，形成圖形。

JSON 結構描述

應用程式資訊清單和覆寫文件的格式是在 JSON 結構描述中定義。您可以使用 JSON 結構描述，在部署之前驗證您的組態文件。JSON 結構描述可在本指南的 GitHub 儲存庫中找到。

- JSON 結構描述 – aws-panorama-developer-guide/resources

應用程式節點

節點是模型、程式碼、攝影機串流、輸出和參數。節點具有介面，可定義其輸入和輸出。可以在您帳戶中的套件、AWS Panorama 提供的套件或內建類型中定義介面。

在下列範例中，code_node並model_node參考範例應用程式隨附的範例程式碼和模型套件。camera_node使用 AWS Panorama 提供的套件，為您在部署期間指定的攝影機串流建立預留位置。

Example graph.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)

Edges 會將一個節點的輸出映射到另一個節點的輸入。在下列範例中，第一個邊緣會將攝影機串流節點的輸出映射至應用程式程式碼節點的輸入。名稱 video_in和 video_out是在節點套件的介面中定義。

Example graph.json – 邊緣

```
    "edges": [
        {
```

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

在應用程式程式碼中，您可以使用 inputs和 outputs 屬性從輸入串流取得影像，並將影像傳送至輸出串流。

Example application.py – 視訊輸入和輸出

```
    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

在應用程式資訊清單中，抽象節點是指 AWS Panorama 定義的套件，您可以在應用程式資訊清單中將其用作預留位置。AWS Panorama 提供兩種類型的抽象節點。

- 攝影機串流 – 選擇應用程式在部署期間使用的攝影機串流。

  套件名稱 – panorama::abstract_rtsp_media_source

  介面名稱 – rtsp_v1_interface
- HDMI 輸出 – 表示應用程式輸出影片。

  套件名稱 – panorama::hdmi_data_sink

  介面名稱 – hdmi0

下列範例顯示應用程式的基本套件、節點和邊緣集，可處理攝影機串流並將視訊輸出至顯示器。使用 AWS Panorama 中abstract_rtsp_media_source套件介面的攝影機節點可接受多個攝影機串流做為輸入。參考 的輸出節點hdmi_data_sink可讓應用程式碼存取從設備的 HDMI 連接埠輸出的視訊緩衝區。

Example graph.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，可在部署期間覆寫。參數可以具有預設值和裝飾器，指示應用程式的使用者如何設定它。

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 `false`。

下列範例顯示兩個參數：字串和數字，這些參數會以輸入形式傳送至程式碼節點。

Example graph.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式資訊清單中修改參數，或在部署時間使用覆寫提供新值。如需詳細資訊，請參閱使用覆寫進行部署時間組態。

使用覆寫進行部署時間組態

您可以在部署期間設定參數和抽象節點。如果您使用 AWS Panorama 主控台進行部署，您可以為每個參數指定值，然後選擇攝影機串流做為輸入。如果您使用 AWS Panorama API 部署應用程式，您可以使用覆寫文件來指定這些設定。

覆寫文件在結構上類似於應用程式資訊清單。對於具有基本類型的參數，您可以定義節點。對於攝影機串流，您可以定義節點和對應至已註冊攝影機串流的套件。然後，您為每個節點定義覆寫，以從其取代的應用程式資訊清單中指定節點。

Example overrides.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"
    }
}
```

在上述範例中，文件會定義一個字串參數和抽象攝影機節點的覆寫。nodeOverrides 告知 AWS Panorama 本文件中的哪些節點會覆寫應用程式資訊清單中的哪些節點。

建置 AWS Panorama 應用程式

應用程式會在 AWS Panorama 設備上執行，以在視訊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，並透過網際網路將其部署到 AWS Panorama 設備。應用程式可以將視訊傳送至顯示器，或使用 AWS 開發套件將結果傳送至 AWS 服務。

模型會分析影像以偵測人員、車輛和其他物件。根據在訓練期間看到的影像，模型會告訴您它認為什麼是事物，以及它對猜測的可信度。您可以使用自己的影像資料來訓練模型，或開始使用範例。

應用程式程式碼程序仍會處理來自攝影機串流的影像、將其傳送至模型，以及處理結果。模型可能會偵測多個物件，並傳回其形狀和位置。程式碼可以使用此資訊將文字或圖形新增至視訊，或將結果傳送至 AWS 服務以進行儲存或進一步處理。

若要從串流取得影像、與模型互動，以及輸出影片，應用程式程式碼會使用 AWS Panorama 應用程式開發套件。應用程式 SDK 是一個 Python 程式庫，支援使用 PyTorch、Apache MXNet 和 TensorFlow 產生的模型。

主題

電腦視覺模型

電腦視覺模型是經過訓練的軟體程式，可偵測影像中的物件。模型會學習透過訓練先分析這些物件的影像，以辨識一組物件。電腦視覺模型會將影像做為輸入，並輸出其偵測到之物件的相關資訊，例如物件類型及其位置。AWS Panorama 支援使用 PyTorch、Apache MXNet 和 TensorFlow 建置的電腦視覺模型。

> **Note**
>
> 如需已使用 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，其中可能包括偵測到的類別、位置資訊和其他資料的概率。下列範例示範如何從影片串流對影像執行推論，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。範例模型會傳回機率陣列，這是結果陣列中第一個且唯一的值。

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，並將其映射到初始化期間載入的資源檔案中的標籤。

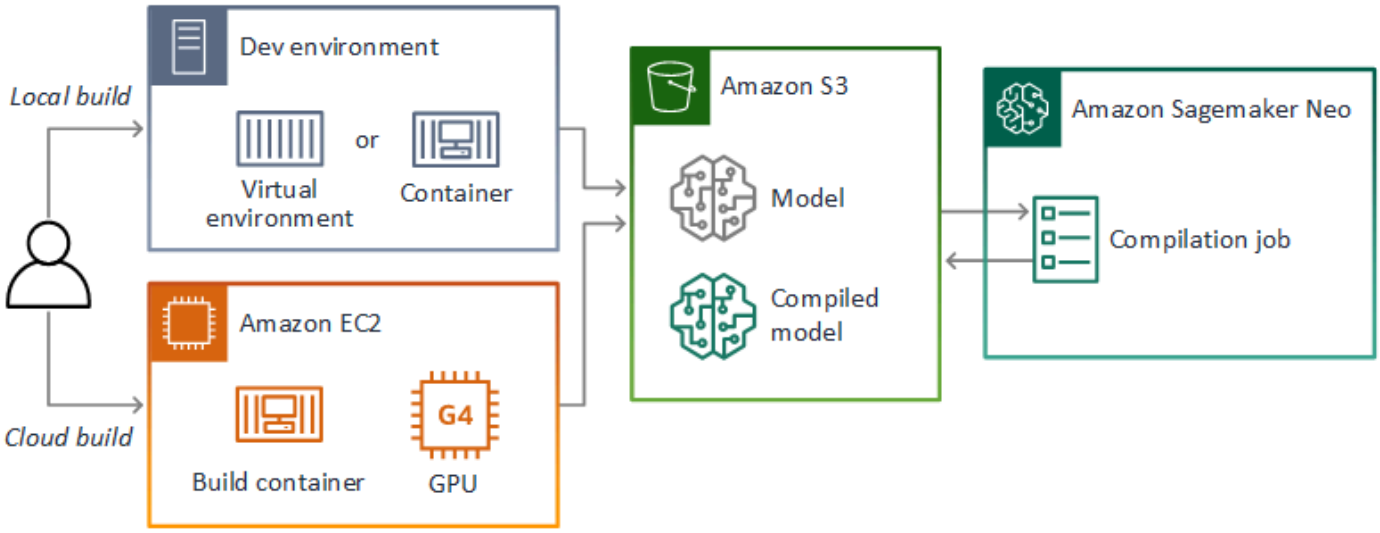
建置自訂模型

您可以在 AWS Panorama 應用程式中使用在 PyTorch、Apache MXNet 和 TensorFlow 中建置的模型。除了在 SageMaker AI 中建置和訓練模型之外，您也可以使用訓練模型，或使用支援的架構建置和訓練自己的模型，並在本機環境或 Amazon EC2 中匯出模型。

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

本指南的 儲存庫提供範例應用程式，以 TensorFlow SavedModel 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 容器中在本機執行。範例應用程式也包含範本和指令碼，用於在 Amazon EC2 執行個體上建置模型。

- 自訂模型範例應用程式



AWS Panorama 使用 SageMaker AI Neo 編譯模型，用於 AWS Panorama 設備。對於每個架構，請使用 SageMaker AI Neo 支援的格式，並將模型封裝在.tar.gz封存中。

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

封裝模型

模型套件包含描述項、套件組態和模型封存。與應用程式映像套件一樣，套件組態會通知 AWS Panorama 服務，模型和描述符存放在 Amazon S3 中。

Example packages/123456789012-SQUEEZENET_PYTORCH-1.0/descriptor.json

```
{
    "mlModelDescriptor": {
        "envelopeVersion": "2021-01-01",
        "framework": "PYTORCH",
        "frameworkVersion": "1.8",
        "precisionMode": "FP16",
        "inputs": [
            {
```

```
                "name": "data",
                "shape": [
                    1,
                    3,
                    224,
                    224
                ]
            }
        ]
    }
}
```

> **Note**
>
> 僅指定架構版本的主要和次要版本。如需支援的 PyTorch、Apache MXNet 和 TensorFlow 版本清單，請參閱支援的架構。

若要匯入模型，請使用 AWS Panorama Application CLI `import-raw-model`命令。如果您對模型或其描述項進行任何變更，則必須重新執行此命令來更新應用程式的資產。如需詳細資訊，請參閱變更電腦視覺模型。

如需描述項檔案的 JSON 結構描述，請參閱 assetDescriptor.schema.json。

訓練模型

當您訓練模型時，請使用目標環境的影像，或從與目標環境非常相似的測試環境。請考慮可能影響模型效能的下列因素：

- 照明 – 主體反射的光線量決定模型必須分析的細節。使用光線良好的主題影像訓練的模型可能無法在光線不足或背光的環境中正常運作。
- 解析度 – 模型的輸入大小通常以方形長寬比固定為 224 到 512 像素寬的解析度。將影片影格傳遞至模型之前，您可以縮減或裁切影片，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偏離影格中心。攝影機的位置也會決定可看見主體的哪些功能。例如，具有廣角鏡頭的頭頂攝影機會在主體位於影格中心時顯示其頂部，並在主體遠離中心時顯示其側面的傾斜檢視。

若要解決這些問題，您可以在將影像傳送到模型之前預先處理影像，並在反映真實環境中差異的更廣泛影像上訓練模型。如果模型需要在照明情況和各種攝影機中操作，您需要更多資料來進行訓練。除了收

集更多影像之外，您還可以透過建立現有影像的變體來取得更多訓練資料，這些變體是偏斜或有不同的光源。

建置應用程式映像

AWS Panorama Appliance 會以從您建置的映像匯出的容器檔案系統的形式執行應用程式。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。

若要建置應用程式映像，請使用 Docker 和 AWS Panorama 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 [package/123456789012-SAMPLE_CODE-1.0/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
```

使用下列 Dockerfile 指示。

- `FROM` – 載入應用程式基礎映像 (`public.ecr.aws/panorama/panorama-application`)。
- `WORKDIR` – 在映像上設定工作目錄。 `/panorama` 用於應用程式程式碼和相關檔案。此設定只會在建置期間持續進行，不會影響應用程式在執行時間 () 的工作目錄/。
- `COPY` – 將檔案從本機路徑複製到映像上的路徑。 會將目前目錄 （ 套件目錄 ） 中的檔案`COPY . .`複製到映像上的工作目錄。例如，應用程式程式碼會從 複製到 `packages/123456789012-SAMPLE_CODE-1.0/application.py` `/panorama/application.py`。
- `RUN` – 在建置期間對映像執行 Shell 命令。單一`RUN`操作可在命令`&&`之間使用 依序執行多個命令。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`。

您可以使用其他在建置時有用的指示`ARG`，例如 `ADD`和 。將執行時間資訊新增至容器的指示，例如 `ENV`，不適用於 AWS Panorama。AWS Panorama 不會從映像執行容器。它只會使用映像來匯出檔案系統，該系統會傳輸至設備。

指定相依性

`requirements.txt` 是一種 Python 需求檔案，可指定應用程式使用的程式庫。範例應用程式使用 Open CV 和 AWS SDK for Python (Boto3)。

Example package/123456789012-SAMPLE_CODE-1.0/requirements.txt

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

Dockerfile 中的 `pip install`命令會將這些程式庫安裝到 下的 Python `dist-packages`目錄`/usr/local/lib`，以便您的應用程式程式碼可以匯入這些程式庫。

本機儲存體

AWS Panorama 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建立和修改檔案。在儲存目錄中建立的檔案會在重新啟動時保留。其他暫存檔案位置會在開機時清除。

建置映像資產

當您使用 AWS Panorama Application CLI 建置應用程式套件的映像時，CLI `docker build`會在套件目錄中執行。這會建置包含應用程式程式碼的應用程式映像。然後，CLI 會建立容器、匯出其檔案系統、壓縮容器，並將其存放在`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 區塊是資產定義，CLI 會將其新增至套件組態 (`package.json`)，並使用 AWS Panorama 服務註冊。CLI 也會複製描述項檔案，指定應用程式指令碼的路徑 （應用程式的進入點）。

Example package/123456789012-SAMPLE_CODE-1.0/descriptor.json

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

在資產資料夾中，描述符和應用程式映像會針對其 SHA-256 檢查總和命名。儲存資產時，此名稱會用作資產的唯一識別符，即為 Amazon S3。

從應用程式碼呼叫 AWS 服務

您可以使用 從應用程式程式碼 適用於 Python (Boto) 的 AWS SDK 呼叫 AWS 服務。例如，如果您的模型偵測到不尋常的情況，您可以將指標發佈至 Amazon CloudWatch、使用 Amazon SNS 傳送通知、將映像儲存至 Amazon S3，或叫用 Lambda 函數進行進一步處理。大多數 AWS 服務都具有可搭配 AWS 開發套件使用的公有 API。

根據預設，設備沒有存取任何 AWS 服務的許可。若要授予許可，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

您可以使用適用於 Python 的 SDK (Boto3) 將訊息傳送到 中的 MQTT 主題 AWS IoT。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 AWS IoT 主控台中找到。

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

選擇指出裝置 ID 或您選擇的其他識別符的名稱。若要發佈訊息，應用程式需要呼叫 的許可`iot:Publish`。

監控 MQTT 佇列

1. 開啟AWS IoT 主控台測試頁面。
2. 針對訂閱主題，輸入主題的名稱。例如：`panorama/panorama_my-appliance_Thing_a01e373b`。
3. 請選擇 Subscribe to topic (訂閱主題)。

AWS Panorama 應用程式開發套件

AWS Panorama 應用程式 SDK 是用於開發 AWS Panorama 應用程式的 Python 程式庫。在應用程式程式碼中，您可以使用 AWS Panorama 應用程式開發套件來載入電腦視覺模型、執行推論，並將影片輸出至監視器。

> **Note**
>
> 若要確保您可存取 AWS Panorama 應用程式開發套件的最新功能，請升級設備軟體。

如需應用程式開發套件定義之類別及其方法的詳細資訊，請參閱應用程式開發套件參考。

章節

新增文字和方塊以輸出影片

使用 AWS Panorama SDK，您可以將影片串流輸出到顯示器。影片可以包含文字和方塊，顯示模型的輸出、應用程式的目前狀態或其他資料。

`video_in` 陣列中的每個物件都是來自連接到設備的攝影機串流的影像。此物件的類型為 `panoramasdk.media`。它有將文字和矩形方塊新增至映像的方法，然後您可以指派給`video_out`陣列。

在下列範例中，範例應用程式會為每個結果新增標籤。每個結果都位於相同的左側位置，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)
```

若要將方塊新增至輸出映像，請使用 `add_rect`。此方法需要 4 個介於 0 和 1 之間的值，表示方塊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)
```

執行多個執行緒

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程序。例如，您可以建立提供 HTTP 流量進行偵錯的執行緒，或監控推論結果並傳送資料的執行緒 AWS。

若要執行多個執行緒，您可以使用 Python 標準程式庫中的執行緒模組來為每個程序建立執行緒。下列範例顯示偵錯伺服器範例應用程式的主迴圈，這會建立應用程式物件，並使用它來執行三個執行緒。

Example package/123456789012-DEBUG_SERVER-1.0/application.py – 主迴圈

```
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。run_cv 迴圈處理來自攝影機串流的影像。如果它收到要停止的訊號，則會關閉偵錯工具程序，該程序會執行 HTTP 伺服器且無法自行關閉。每個執行緒都必須處理自己的錯誤。如果未攔截並記錄錯誤，執行緒會以無提示方式結束。

Example package/123456789012-DEBUG_SERVER-1.0/application.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")
```

執行緒會透過應用程式的`self`物件進行通訊。若要重新啟動應用程式處理迴圈，除錯器執行緒會呼叫`stop`方法。此方法會設定`terminate`屬性，以發出其他執行緒關閉的訊號。

Example package/123456789012-DEBUG_SERVER-1.0/application.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)
```

提供傳入流量

您可以搭配應用程式程式碼執行 HTTP 伺服器，在本機監控或偵錯應用程式。若要提供外部流量，請將 AWS Panorama 設備上的連接埠映射至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。在設備上開啟連接埠會有隱含的安全風險。使用此功能時，您必須採取其他步驟來保護您的設備免受外部流量影響，並保護授權用戶端與設備之間的通訊。
> 本指南中包含的範例程式碼僅供示範之用，不會實作身分驗證、授權或加密。

您可以在設備上開啟範圍介於 8000–9000 的連接埠。這些連接埠開啟時，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，並將設備上的連接埠映射到應用程式容器上的連接埠。設備軟體會將流量轉送至容器，並將回應傳回給請求者。請求會在您指定的設備連接埠上收到，而回應則會在隨機暫時性連接埠上傳出。

設定傳入連接埠

您可以在應用程式組態的三個位置指定連接埠映射。程式碼套件的 `package.json`，您可以指定程式碼節點在`network`區塊中接聽的連接埠。下列範例宣告節點接聽連接埠 80。

Example package/123456789012-DEBUG_SERVER-1.0/package.json

```
        "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，您會宣告路由規則，將設備上的連接埠映射到應用程式的程式碼容器上的連接埠。下列範例新增規則，將裝置上的連接埠 8080 映射至code_node容器上的連接埠 80。

Example graphs/my-app/graph.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，或將覆寫文件傳遞至 CreateApplicationInstance API。您必須在部署時間提供此組態，以確認您想要開啟設備上的連接埠。

Example graphs/my-app/override.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由另一個應用程式使用，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時，您可以開啟通訊端或執行伺服器來處理傳入的請求。此debug-server範例顯示搭配電腦視覺應用程式程式碼執行的 HTTP 伺服器的基本實作。

> **Important**
>
> 範例實作對於生產使用並不安全。為了避免讓您的設備容易遭受攻擊，您必須在程式碼和網路組態中實作適當的安全控制。

Example package/123456789012-DEBUG_SERVER-1.0/application.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.')
```

伺服器接受`/status`路徑上的 GET 請求，以擷取應用程式的一些資訊。它也接受 的 POST 請求`/restart`，以重新啟動應用程式。

為了示範此功能，範例應用程式會在單獨的執行緒上執行 HTTP 用戶端。用戶端會在啟動後立即透過本機網路呼叫`/status`路徑，並在幾分鐘後重新啟動應用程式。

Example package/123456789012-DEBUG_SERVER-1.0/application.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")
```

主迴圈會管理執行緒，並在結束時重新啟動應用程式。

Example package/123456789012-DEBUG_SERVER-1.0/application.py – 主迴圈

```
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，請參閱本指南的 GitHub 儲存庫中的指示。

使用 GPU

您可以在 AWS Panorama 設備上存取圖形處理器 (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，請在建置應用程式碼容器後，將 GPU 存取新增為套件組態的需求。

> **Important**
>
> 如果您啟用 GPU 存取，則無法在設備的任何應用程式中執行模型節點。為了安全起見，當設備執行使用 SageMaker AI Neo 編譯的模型時，GPU 存取會受到限制。使用 GPU 存取時，您必須在應用程式程式碼節點中執行模型，且裝置上的所有應用程式都必須共用 GPU 的存取權。

若要開啟應用程式的 GPU 存取，請在使用 AWS Panorama Application CLI 建置套件後更新套件組態。下列範例顯示將 GPU 存取權新增至應用程式程式碼節點的 `requirements` 區塊。

Example package.json 與需求區塊

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

使用 GPU 存取部署應用程式

1. 若要建置應用程式容器，請使用 `build-container`命令。

```
$ 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`命令。

```
$ 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-samples GitHub 儲存庫。

在 Windows 中設定開發環境

若要建置 AWS Panorama 應用程式，您可以使用 Docker、命令列工具和 Python。在 Windows 中，您可以使用 Docker Desktop 搭配 Windows Subsystem for Linux 和 Ubuntu 來設定開發環境。本教學課程會逐步引導您完成使用 AWS Panorama 工具和範例應用程式測試之開發環境的設定程序。

章節

先決條件

若要遵循本教學課程，您需要支援 Windows Subsystem for Linux 2 (WSL 2) 的 Windows 版本。

- Windows 10 1903 版及更高版本 （建置 18362 及更高版本） 或 Windows 11
- Windows 功能
  - 適用於 Linux 的 Windows 子系統
  - Hyper-V：
  - 虛擬機器平台

本教學課程使用下列軟體版本開發。

- Ubuntu 20.04
- Python 3.8.5
- Docker 20.10.8

安裝 WSL 2 和 Ubuntu

如果您有 Windows 10 2004 版和更新版本 （建置 19041 和更新版本），您可以使用下列 PowerShell 命令安裝 WSL 2 和 Ubuntu 20.04。

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

對於較舊的 Windows 版本，請遵循 WSL 2 文件中的指示：較舊版本的手動安裝步驟

安裝 Docker

若要安裝 Docker Desktop，請從 hub.docker.com：// 下載並執行安裝程式套件。如果您遇到問題，請遵循 Docker 網站的指示：Docker Desktop WSL 2 後端。

執行 Docker 桌面並遵循初次執行教學課程來建置範例容器。

> **Note**
>
> Docker Desktop 只會在預設分佈中啟用 Docker。如果您在執行本教學課程之前已安裝其他 Linux 發行版本，請在 資源、WSL 整合下的 Docker 桌面設定選單中，在新安裝的 Ubuntu 發行版本中啟用 Docker。

設定 Ubuntu

您現在可以在 Ubuntu 虛擬機器中執行 Docker 命令。若要開啟命令列終端機，請從開始功能表執行分佈。第一次執行它時，您可以設定使用者名稱和密碼，可用來執行管理員命令。

若要完成開發環境的組態，請更新虛擬機器的軟體並安裝工具。

設定虛擬機器

1. 更新 Ubuntu 隨附的軟體。

   ```
   $ 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. 開啟新的終端機，然後執行 `aws configure`來設定 AWS CLI。

```
$ aws configure
```

如果您沒有存取金鑰，您可以在 IAM 主控台中產生這些金鑰。

最後，下載並匯入範例應用程式。

取得範例應用程式

1. 下載並擷取範例應用程式。

```
$ 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. 執行包含的指令碼來測試編譯、建置應用程式容器，並將套件上傳至 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 Application CLI 會上傳套件，並使用 AWS Panorama 服務註冊這些套件。您現在可以使用 AWS Panorama 主控台部署範例應用程式。

後續步驟

若要探索和編輯專案檔案，您可以使用 File Explorer 或支援 WSL 的整合式開發環境 (IDE)。

若要存取虛擬機器的檔案系統，請開啟檔案總管，然後在導覽列\\wsl$中輸入 。此目錄包含虛擬機器檔案系統 (Ubuntu-20.04) 和 Docker 資料檔案系統的連結。在 下Ubuntu-20.04，您的使用者目錄位於 home*username*。

> **Note**
>
> 若要從 Ubuntu 內存取 Windows 安裝中的檔案，請導覽至 /mnt/c目錄。例如，您可以執行來列出下載目錄中的檔案ls /mnt/c/Users/*windows-username*/Downloads。

使用 Visual Studio Code，您可以在開發環境中編輯應用程式程式碼，並使用整合式終端機執行命令。若要安裝 Visual Studio Code，請造訪 https：//code.visualstudio.com。安裝後，新增遠端 WSL 延伸模組。

Windows 終端機是標準 Ubuntu 終端機的替代方案，而此終端機已執行命令。它支援多個索引標籤，並可以為您安裝的任何其他 Linux 類型執行 PowerShell、命令提示字元和終端機。它支援使用 Ctrl+C和 複製和貼上Ctrl+V、可點選URLs 和其他有用的改進。若要安裝 Windows 終端機，請造訪 https：//microsoft.com。

AWS Panorama API

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 開發套件，您可以開發管理資源和部署的指令碼或應用程式。本指南的 GitHub 儲存庫包含您可以使用的指令碼，做為您自己的程式碼的起點。

- aws-panorama-developer-guide/util-scripts

章節

自動化裝置註冊

若要佈建設備，請使用 ProvisionDevice API。回應包含具有裝置組態和臨時登入資料的 ZIP 檔案。解碼檔案，並將其儲存在具有字首 的封存中`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}"
```

組態封存中的登入資料會在 5 分鐘後過期。使用隨附的 USB 隨身碟將封存檔傳輸到您的設備。

若要註冊攝影機，請使用 CreateNodeFromTemplateJob API。此 API 會取得相機使用者名稱、密碼和 URL 的範本參數映射。您可以使用 Bash 字串操作，將此映射格式化為 JSON 文件。

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，您可以從檔案載入 JSON 組態。

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

使用 AWS Panorama API 管理設備

您可以使用 AWS Panorama API 自動化設備管理任務。

檢視裝置

若要取得具有裝置 IDs設備清單，請使用 ListDevices API。

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

若要取得設備的詳細資訊，請使用 DescribeDevice API。

```
$ 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 更新CurrentSoftware，您可以升級裝置。使用 CreateJobForDevices API over-the-air(OTA) 更新任務。

```
$ 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"
        }
    ]
}
```

在指令碼中，您可以使用 Bash 字串操作在任務組態檔案中填入映像版本欄位。

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 DescribeDeviceJob API 觀看更新進度。

```
$ 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
}
```

若要取得所有執行中任務的清單，請使用 ListDevicesJobs。

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

如需檢查和套用更新的範例指令碼，請參閱本指南的 GitHub 儲存庫中的 https：// check-updates.sh：// https：// https：// https：// https：// https：///www./www.

重新啟動設備

若要重新啟動設備，請使用 CreateJobForDevices API。

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

```
}
```

在指令碼中，您可以取得裝置清單，然後選擇一個以互動方式重新啟動。

Example reboot-device.sh – 用量

```
$ ./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"
        }
    ]
}
```

自動化應用程式部署

若要部署應用程式，您可以使用 AWS Panorama 應用程式 CLI 和 AWS Command Line Interface。建置應用程式容器之後，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 CreateApplicationInstance API 部署應用程式。

如需使用所示指令碼的更多內容和說明，請遵循範例應用程式 README 中的指示。

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`命令。此命令會建置 Docker 容器，並將其儲存為`assets`資料夾中的壓縮檔案系統。

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路徑引數，方法是輸入路徑的一部分，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`命令。此命令會將資產從 `assets` 資料夾上傳至 AWS Panorama 管理的 Amazon S3 存取點。

Example 4-package-app.sh

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

AWS Panorama Application CLI 會上傳每個套件中套件組態 (`package.json`) 參考的容器和描述項資產，並將套件註冊為 AWS Panorama 中的節點。然後，您可以參考應用程式資訊清單中的這些節點 (`graph.json`) 來部署應用程式。

部署應用程式

若要部署應用程式，請使用 CreateApplicationInstance API。此動作會採取下列參數等。

- `ManifestPayload` – 定義應用程式節點、套件、邊緣和參數的應用程式資訊清單 (`graph.json`)。
- `ManifestOverridesPayload` – 覆寫第一個 中參數的第二個資訊清單。應用程式資訊清單可視為應用程式來源中的靜態資源，其中覆寫資訊清單提供可自訂部署的部署時間設定。
- `DefaultRuntimeContextDevice` – 目標裝置。
- `RuntimeRoleArn` – 應用程式用來存取 AWS 服務和資源的 IAM 角色 ARN。
- `ApplicationInstanceIdToReplace` – 從裝置中移除的現有應用程式執行個體 ID。

資訊清單和覆寫承載是 JSON 文件，必須以巢狀於另一個文件內的字串值提供。若要這樣做，指令碼會從檔案載入資訊清單做為字串，並使用 jq 工具建構巢狀文件。

Example 5-deploy.sh – 撰寫資訊清單

```
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 ListDevices API 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 – 尋找裝置

```
    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 (1-create-role.sh：//) 建立。部署指令碼會從中取得此角色的 ARN AWS CloudFormation。如果應用程式已部署到裝置，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 和取代引數

```
# 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，指令碼會將所有片段放在一起，以建立應用程式執行個體並將應用程式部署到裝置。服務會以指令碼存放以供日後使用的執行個體 ID 來回應。

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

若要監控部署，請使用 ListApplicationInstances API。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，並使用它們來建構 CLI 命令。然後，它會在迴圈中呼叫 。

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
```

部署完成時，您可以呼叫 Amazon CloudWatch Logs API 來檢視日誌。檢視日誌指令碼使用 CloudWatch Logs `GetLogEvents` API。

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 API 監控和管理應用程式。

檢視應用程式

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

```
$ 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"
            }
        },
    ]
}
```

若要取得應用程式執行個體節點的詳細資訊，請使用 ListApplicationInstanceNodeInstances API。

```
$ 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 SignalApplicationInstanceNodeInstances API 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"
}
```

在指令碼中，您可以取得節點清單，然後選擇一個以互動方式暫停或繼續。

Example pause-camera.sh – 用量

```
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，您可以循環瀏覽比可同時處理更多數量的攝影機串流。若要這樣做，請將多個攝影機串流映射到覆寫資訊清單中的相同輸入節點。

在下列範例中，覆寫資訊清單會將兩個攝影機串流warehouse-floor和 映射entrance-north至相同的輸入節點 (camera_node)。當應用程式啟動且entrance-north節點等待訊號開啟時，warehouse-floor串流會處於作用中狀態。

Example override-multicam.json

```
    "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 自動化應用程式部署。

使用 VPC 端點

如果您在沒有網際網路存取的情況下使用 VPC，您可以建立 VPC 端點以搭配 AWS Panorama 使用。VPC 端點可讓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至 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
- 將設備連接到私有子網路
- 範本 AWS CloudFormation 範例

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 不需要 VPC 端點。只有在沒有網際網路存取的情況下使用 VPC 時，才需要建立 VPC 端點。當 AWS CLI 或 SDK 嘗試連線到 AWS Panorama 時，流量會透過 VPC 端點路由。

使用下列設定建立 AWS Panorama 的 VPC 端點：

- 服務名稱 – **com.amazonaws.*us-west-2*.panorama**
- 類型 – 介面

VPC 端點使用服務的 DNS 名稱從 AWS 開發套件用戶端取得流量，而不需要任何額外的組態。如需使用 VPC 端點的詳細資訊，請參閱《Amazon VPC 使用者指南》中的界面 VPC 端點。

將設備連接到私有子網路

AWS Panorama 設備可以透過使用 AWS Site-to-Site VPN 或 的私有 VPN AWS 連線連線至 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。設備會連線至私有子網路，並透過 VPC 端點存取 AWS 服務。

Site-to-Site和 Direct Connect 是將資料中心安全地連線至 Amazon VPC 的服務。使用Site-to-Site VPN，您可以使用商用網路裝置來連線。 Direct Connect 會使用 AWS 裝置來連線。

- Site-to-Site VPN – 什麼是 AWS Site-to-Site VPN？
- Direct Connect – 什麼是 AWS Direct Connect？

將本機網路連接到 VPC 中的私有子網路之後，請為下列服務建立 VPC 端點。

- Amazon Simple Storage Service – AWS PrivateLink 適用於 Amazon S3
- AWS IoT Core – AWS IoT Core 搭配界面 VPC 端點 （資料平面和登入資料提供者） 使用
- Amazon Elastic Container Registry – Amazon Elastic Container Registry 介面 VPC 端點
- Amazon CloudWatch – 搭配介面 VPC 端點使用 CloudWatch
- Amazon CloudWatch Logs – 搭配介面 VPC 端點使用 CloudWatch Logs

設備不需要連線至 AWS Panorama 服務。它透過 中的訊息管道與 AWS Panorama 通訊 AWS IoT。

除了 VPC 端點之外，Amazon S3 和 AWS IoT 還需要使用 Amazon Route 53 私有託管區域。私有託管區域會將來自子網域的流量路由至正確的 VPC 端點，包括 Amazon S3 存取點和 MQTT 主題的子網域。如需私有託管區域的資訊，請參閱《Amazon Route 53 開發人員指南》中的使用私有託管區域。

如需使用 VPC 端點和私有託管區域的範例 VPC 組態，請參閱 範本 AWS CloudFormation 範例。

範本 AWS CloudFormation 範例

本指南的 GitHub 儲存庫提供 AWS CloudFormation 範本，您可以用來建立資源以搭配 AWS Panorama 使用。範本會建立具有兩個私有子網路、公有子網路和 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。公有子網路中的資源可以與私有資源通訊，但私有資源無法從網際網路存取。

Example vpc-endpoint.yml – 私有子網路

```
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 範本示範如何為 AWS Panorama 建立 VPC 端點。您可以使用此端點，透過 AWS SDK 或 管理 AWS Panorama 資源 AWS CLI。

Example vpc-endpoint.yml – VPC 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:
          - "*"
```

PolicyDocument 是以資源為基礎的許可政策，定義可以使用端點進行的 API 呼叫。您可以修改政策來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 範本說明如何為 AWS Panorama 設備所使用的服務建立 VPC 端點和私有託管區域。

Example vpc-appliance.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 ] ] ]
```

範例範本示範使用範例 VPC 建立 Amazon VPC 和 Route 53 資源。您可以移除 VPC 資源，並將子網路、安全群組和 VPC IDs 的參考取代為資源IDs，以針對您的使用案例進行調整。

應用程式、指令碼和範本範例

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

章節

範例應用程式

範例應用程式示範使用 AWS Panorama 功能和常見的電腦視覺任務。這些範例應用程式包含可自動設定和部署的指令碼和範本。使用最少的組態，您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電腦視覺。使用 適用於 Python (Boto) 的 AWS SDK 將指標上傳至 CloudWatch、儀器預先處理和推論方法，以及設定記錄。
- debug-server – 在裝置上開啟傳入連接埠，並將流量轉送至應用程式碼容器。使用多執行緒同時執行應用程式碼、HTTP 伺服器和 HTTP 用戶端。
- 自訂模型 – 從程式碼匯出模型，並使用 SageMaker AI Neo 編譯，以測試與 AWS Panorama 設備的相容性。在 Python 開發、Docker 容器或 Amazon EC2 執行個體上於本機建置。匯出和編譯特定 TensorFlow 或 Python 版本的所有 Keras 內建應用程式模型。

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

公用程式指令碼

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

- provision-device.sh – 佈建裝置。
- check-updates.sh – 檢查並套用設備軟體更新。

- reboot-device.sh – 重新啟動裝置。
- register-camera.sh – 註冊攝影機。
- deregister-camera.sh – 刪除攝影機節點。
- view-logs.sh – 檢視應用程式執行個體的日誌。
- pause-camera.sh – 暫停或繼續攝影機串流。
- push.sh – 建置、上傳和部署應用程式。
- rename-package.sh – 重新命名節點套件。更新目錄名稱、組態檔案和應用程式資訊清單。
- samplify.sh – 將您的帳戶 ID 取代為範例帳戶 ID，並還原備份組態以移除本機組態。
- update-model-config.sh：// – 更新描述項檔案後，將模型重新新增至應用程式。
- cleanup-patches.sh：// – 取消註冊舊修補程式版本，並從 Amazon S3 刪除其資訊清單。

如需用量詳細資訊，請參閱 README。

CloudFormation 範本

使用 `cloudformation-templates`目錄中的 CloudFormation 範本為應用程式建立資源 AWS Panorama 。

- alarm-application.yml – 建立警示，監控應用程式是否有錯誤。如果應用程式執行個體引發錯誤或停止執行 5 分鐘，警示會傳送通知電子郵件。
- alarm-device.yml – 建立監控裝置連線能力的警示。如果裝置停止傳送指標 5 分鐘，警示會傳送通知電子郵件。
- application-role.yml – 建立應用程式角色。此角色包含將指標傳送至 CloudWatch 的許可。為您的應用程式使用的其他 API 操作將許可新增至政策陳述式。
- vpc-appliance.yml – 為 AWS Panorama 設備建立具有私有子網路服務存取權的 VPC。若要將設備連接至 VPC，請使用 AWS Direct Connect 或 AWS Site-to-Site VPN。
- vpc-endpoint.yml – 建立具有 AWS Panorama 服務私有子網路服務的 VPC。VPC 內的資源可以連線至 AWS Panorama 來監控和管理 AWS Panorama 資源，而無需連線至網際網路。

此目錄中的`create-stack.sh`指令碼會建立 CloudFormation 堆疊。它需要變數數量的引數。第一個引數是範本的名稱，其餘的引數是範本中參數的覆寫。

例如，以下命令會建立應用程式角色。

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

更多範例和工具

aws-panorama-samples 儲存庫有更多範例應用程式和有用的工具。

- 應用程式 – 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 – 驗證攝影機串流。
- PanoJupyter – 在 AWS Panorama 設備上執行 JupyterLab。
- Sideloading – 在不建置或部署應用程式容器的情況下更新應用程式程式碼。

AWS 社群也開發了 的工具和指引 AWS Panorama。在 GitHub 上查看下列開放原始碼專案。

- Cookiecutter-panorama – AWS Panorama 應用程式的 Cookiecutter 範本。
- 背包 – 用於存取執行期環境詳細資訊、分析和其他影片輸出選項的 Python 模組。

監控 AWS Panorama 資源和應用程式

您可以使用 AWS Panorama 主控台和 Amazon CloudWatch 監控 AWS Panorama 資源。 AWS Panorama 設備會透過網際網路連線至 AWS 雲端，以報告其狀態和已連線攝影機的狀態。開啟時，設備也會即時將日誌傳送至 CloudWatch Logs。

設備會從您第一次使用 AWS Panorama 主控台建立的服務角色取得使用、 AWS IoT CloudWatch Logs 和其他 AWS 服務的許可。如需詳細資訊，請參閱AWS Panorama 服務角色和跨服務資源。

如需疑難排解特定錯誤的說明，請參閱 疑難排解。

主題

在 AWS Panorama 主控台中監控

您可以使用 AWS Panorama 主控台來監控 AWS Panorama 設備與攝影機。主控台使用 AWS IoT 來監控設備的狀態。

在 AWS Panorama 主控台中監控您的設備

1. 開啟 AWS Panorama 主控台。
2. 開啟 AWS Panorama 主控台裝置頁面。
3. 選擇設備。
4. 若要查看應用程式執行個體的狀態，請從清單中選擇它。
5. 若要查看設備網路介面的狀態，請選擇設定。

設備的整體狀態會顯示在頁面頂端。如果狀態為線上，則設備會連線至 AWS 並傳送定期狀態更新。

檢視 AWS Panorama 日誌

AWS Panorama 會將應用程式和系統事件回報給 Amazon CloudWatch Logs。當您遇到問題時，您可以使用事件日誌來協助除錯 AWS Panorama 應用程式或對應用程式的組態進行故障診斷。

在 CloudWatch Logs 中檢視日誌

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

   - 裝置日誌 – `/aws/panorama/devices/device-id`
   - 應用程式日誌 – `/aws/panorama/devices/device-id/applications/instance-id`

當您在更新系統軟體後重新佈建設備時，您也可以檢視佈建 USB 磁碟機上的日誌。

章節

檢視裝置日誌

AWS Panorama Appliance 會為您部署的每個應用程式執行個體建立日誌群組和群組。裝置日誌包含應用程式狀態、軟體升級和系統組態的相關資訊。

裝置日誌 – **`/aws/panorama/devices/device-id`**

- `occ_log` – 控制器程序的輸出。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` – 協調over-the-air(OTA) 軟體升級的程序輸出。
- `syslog` – 裝置 syslog 程序的輸出，可擷取程序之間傳送的訊息。
- `kern_log` – 來自裝置 Linux 核心的事件。
- `logging_setup_logs` – 從設定 CloudWatch Logs 代理程式的程序輸出。

- `cloudwatch_agent_logs` – 從 CloudWatch Logs 代理程式輸出。
- `shadow_log` – 從AWS IoT 裝置影子輸出。

檢視應用程式日誌

應用程式執行個體的日誌群組包含每個節點的日誌串流，以節點命名。

應用程式日誌 – **/aws/panorama/devices/*device-id*/applications/*instance-id***

- 程式碼 – 從應用程式程式碼和 AWS Panorama 應用程式 SDK 輸出。從彙總應用程式日誌`/opt/aws/panorama/logs`。
- 模型 – 使用模型協調推論請求的程序輸出。
- 串流 – 從從攝影機串流解碼視訊的程序輸出。
- 顯示 – 轉譯 HDMI 連接埠視訊輸出的程序輸出。
- `mds` – 來自設備中繼資料伺服器的日誌。
- `console_output` – 從程式碼容器擷取標準輸出和錯誤串流。

如果您在 CloudWatch Logs 中看不到日誌，請確認您位於正確的 AWS 區域。如果是，則設備與 AWS 的連線或設備 AWS Identity and Access Management (IAM) 角色的許可可能會出現問題。

設定應用程式日誌

設定 Python 記錄器將日誌檔案寫入 `/opt/aws/panorama/logs`。設備會將日誌從此位置串流到 CloudWatch Logs。若要避免使用過多的磁碟空間，請使用檔案大小上限為 10 MiB，備份計數為 1。下列範例顯示建立記錄器的方法。

Example application.py – Logger 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
```

在全域範圍內初始化記錄器，並在整個應用程式程式碼中使用它。

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封存傳輸至設備的 USB 磁碟機。使用這些日誌來疑難排解具有最新軟體版本之設備上的佈建問題。

> **Important**
>
> 佈建日誌可供更新至軟體版本 4.3.23 或更新版本的設備使用。

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更新代理程式日誌。
- `/panorama/syslog.log` – Linux 系統日誌。
- `/panorama/kern.log` – Linux 核心日誌。

從裝置輸出日誌

如果您的裝置和應用程式日誌未出現在 CloudWatch Logs 中，您可以使用 USB 隨身碟從裝置取得加密的日誌映像。AWS Panorama 服務團隊可以代表您解密日誌並協助偵錯。

先決條件

若要遵循程序，您將需要下列硬體：

- USB 磁碟機 – 儲存至少 1 GB 的 FAT32-formatted USB 隨身碟，用於從 AWS Panorama 設備傳輸日誌檔案。

從裝置輸出日誌

1. 準備 USB 磁碟機，其中資料夾位於 `panorama` 資料夾`managed_logs`內。

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

2. 將 USB 隨身碟連接至裝置。
3. 關閉 AWS Panorama 設備。
4. 開啟 AWS Panorama 設備的電源。
5. 裝置會將日誌複製到裝置。狀態 LED 會在進行此作業時閃爍藍色。
6. 然後，您可以在 `managed_logs`目錄中找到具有 格式的日誌檔案 `panorama_device_log_`*`v1_dd_hh_mm`*`.img`

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

使用 Amazon CloudWatch 監控設備和應用程式

當設備上線時，AWS Panorama 會將指標傳送至 Amazon CloudWatch。您可以在 CloudWatch 主控台中使用這些指標建置圖形和儀表板，以監控設備活動，並設定警示，在裝置離線或應用程式發生錯誤時通知您。

若要在 CloudWatch 主控台中檢視指標

1. 開啟 AWS Panorama 主控台指標頁面 (`PanoramaDeviceMetrics` 命名空間）。
2. 選擇維度結構描述。
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 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 Always Free 方案。除了免費方案閾值之外，CloudWatch 還會收取指標、儀表板、警示、記錄和洞察的費用。如需詳細資訊，請參閱 CloudWatch 定價。

如需有關 CloudWatch 的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》。

章節

- 使用裝置指標
- 使用應用程式指標
- 設定警示

使用裝置指標

當設備上線時，它會將指標傳送至 Amazon CloudWatch。您可以使用這些指標來監控裝置活動，並在裝置離線時觸發警示。

- `DeviceActive` – 當裝置處於作用中狀態時定期傳送。

  維度 – `DeviceId`和 `DeviceName`。

使用 Average統計資料檢視DeviceActive指標。

使用應用程式指標

當應用程式遇到錯誤時，它會將指標傳送至 Amazon CloudWatch。如果應用程式停止執行，您可以使用這些指標來觸發警示。

- ApplicationErrors – 記錄的應用程式錯誤數目。

  維度 – ApplicationInstanceName和 ApplicationInstanceId。

使用 Sum 統計資料檢視應用程式指標。

設定警示

若要在指標超過閾值時取得通知，請建立警示。例如，您可以建立警示，在ApplicationErrors指標總和維持在 1 達 20 分鐘時傳送通知。

欲建立警示

1. 開啟 Amazon CloudWatch 主控台警示頁面。
2. 選擇 Create alarm (建立警示)。
3. 選擇選取指標並尋找裝置的指標，例如 ApplicationErrors 的指標applicationInstance-gk75xmplqbqtenlnmz4ehiu7xamy-application。
4. 請依照指示來設定警示的條件、動作和名稱。

如需詳細說明，請參閱《Amazon CloudWatch 使用者指南》中的建立 CloudWatch 警示。 Amazon CloudWatch

疑難排解

下列主題提供您在使用 AWS Panorama 主控台、設備或 SDK 時可能遇到的錯誤和問題的疑難排解建議。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回饋按鈕進行報告。

您可以在 Amazon CloudWatch Logs 主控台中找到設備的日誌。設備會在產生日誌時，從您的應用程式程式碼、設備軟體和 AWS IoT 程序上傳日誌。如需詳細資訊，請參閱檢視 AWS Panorama 日誌。

佈建中

問題：(macOS) 我的電腦無法透過 USB-C 轉接器識別隨附的 USB 隨身碟。

如果您將 USB 隨身碟插入已連接到電腦的 USB-C 轉接器，就可能發生這種情況。請嘗試中斷轉接器的連接，並將其重新連接到已連接的 USB 隨身碟。

問題：當我使用自己的 USB 隨身碟時佈建失敗。

問題：當我使用設備的 USB 2.0 連接埠時佈建失敗。

AWS Panorama 設備與 1 到 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，觀察到一些問題。為了取得一致的結果，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 Lenovo ThinkEdge® SE70，設備不包含 USB 隨身碟。使用至少具有 1 GB 儲存空間的 USB 3.0 隨身碟。

設備組態

問題：設備會在開機期間顯示空白畫面。

完成大約需要一分鐘的初始開機順序後，設備會在載入模型並啟動應用程式時，顯示一分鐘以上的空白畫面。此外，如果您在顯示器開啟後連接顯示器，則設備不會輸出視訊。

問題：當我按住電源按鈕將其關閉時，設備不會回應。

設備最多需要 10 秒才能安全地關閉。您只需按住電源按鈕 1 秒，即可啟動關機程序。如需按鈕操作的完整清單，請參閱AWS Panorama Appliance 按鈕和燈光。

問題：我需要產生新的組態封存，才能變更設定或取代遺失的憑證。

AWS Panorama 在您下載裝置憑證或網路組態之後， 不會儲存它，而且您無法重複使用組態封存。使用 AWS Panorama 主控台刪除設備，並使用新的組態封存建立新的設備。

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 。

當您部署具有輸出節點的多個應用程式時，最近啟動的應用程式會使用 HDMI 輸出。如果此應用程式停止執行，則另一個應用程式可以使用輸出。若要僅授予一個應用程式對輸出的存取權，請從另一個應用程式的應用程式資訊清單移除輸出節點和對應的邊緣，然後重新部署。

問題：應用程式輸出不會出現在日誌中

設定 Python 記錄器將日誌檔案寫入 `/opt/aws/panorama/logs`。這些會在程式碼容器節點的日誌串流中擷取。標準輸出和錯誤串流會擷取在名為 的個別日誌串流中`console-output`。如果您使用 `print`，請使用 `flush=True`選項來防止訊息卡在輸出緩衝區中。

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，您都會註冊一個修補程式版本，該版本代表其使用的每個套件的套件組態和資產檔案。使用清除修補程式指令碼來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：攝影機節點日誌

設備無法連線至應用程式的攝影機串流。發生這種情況時，當應用程式從 AWS Panorama 應用程式 SDK 等待影片影格時，影片輸出會在最後一個處理的影格上空白或凍結。設備軟體會嘗試連線至攝影

機串流，並在攝影機節點日誌中記錄逾時錯誤。確認您的攝影機串流 URL 正確，且 RTSP 流量可在網路中的攝影機和設備之間路由。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

找不到具有攝影機串流憑證的 AWS Secrets Manager 秘密。刪除攝影機串流並重新建立。

錯誤：Camera did not provide an H264 encoded stream

來源：攝影機節點日誌

攝影機串流具有 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。如需支援攝影機的詳細資訊，請參閱支援的攝影機。

AWS Panorama 的安全性

的雲端安全性 AWS 是最高優先順序。身為 AWS 客戶，您可以從資料中心和網路架構中受益，該架構專為滿足最安全敏感組織的需求而建置。

安全性是 AWS 和 之間的共同責任。共同責任模型 將此描述為雲端的安全和雲端內的安全：

- 雲端的安全性 – AWS 負責保護在 AWS Cloud 中執行 AWS 服務的基礎設施。 AWS 也為您提供可安全使用的服務。在 AWS 合規計畫中，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 合規計劃範圍內的服務。
- 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。

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

主題

- AWS Panorama 設備安全功能
- AWS Panorama 設備安全最佳實務
- AWS Panorama 中的資料保護
- AWS Panorama 的身分和存取管理
- AWS Panorama 的合規驗證
- AWS Panorama 中的基礎設施安全性
- AWS Panorama 中的執行期環境軟體

AWS Panorama 設備安全功能

為了保護您的應用程式、模型和硬體免受惡意程式碼和其他漏洞的侵害，AWS Panorama Appliance 實作了一組廣泛的安全功能。這些包括但不限於下列項目。

- 全磁碟加密 – 裝置實作 Linux 統一金鑰設定 (LUKS2) 全磁碟加密。所有系統軟體和應用程式資料都會使用您裝置專屬的金鑰進行加密。即使擁有裝置的實體存取權，攻擊者也無法檢查其儲存體的內容。
- 記憶體配置隨機化 – 為了防止以載入記憶體的可執行程式碼為目標的攻擊，AWS Panorama 設備使用地址空間配置隨機化 (ASLR)。ASLR 會在載入記憶體時隨機化作業系統程式碼的位置。這可防止使用嘗試覆寫或執行特定程式碼區段的漏洞，方法是預測其在執行階段存放的位置。
- 信任的執行環境 – 設備使用以 ARM TrustZone 為基礎的信任執行環境 (TEE)，並具有隔離的儲存、記憶體和處理資源。儲存在信任區域中的金鑰和其他敏感資料只能由信任的應用程式存取，而信任的應用程式會在 TEE 內的個別作業系統中執行。AWS Panorama Appliance 軟體會在不受信任的 Linux 環境中與應用程式程式碼一起執行。它只能透過向安全應用程式提出請求來存取密碼編譯操作。
- 安全佈建 – 當您佈建設備時，您傳輸至裝置的登入資料 （ 金鑰、憑證和其他密碼編譯材料 ） 僅在短時間內有效。設備會使用短期憑證來連線至 ， AWS IoT 並為其本身請求有效較長時間的憑證。AWS Panorama 服務會產生登入資料，並使用裝置上硬式編碼的金鑰加密登入資料。只有請求憑證的裝置可以解密憑證，並與 AWS Panorama 通訊。
- 安全開機 – 當裝置啟動時，每個軟體元件都會在執行前進行身分驗證。處理器中無法修改的軟體硬式編碼開機 ROM 會使用硬式編碼加密金鑰來解密開機載入器，以驗證信任的執行環境核心等。
- 已簽署核心 – 核心模組使用非對稱加密金鑰簽署。作業系統核心會使用公有金鑰解密簽章，並在將模組載入記憶體之前驗證其是否符合模組的簽章。
- dm-verity – 與核心模組的驗證方式類似，設備會使用 Linux Device Mapper `dm-verity`的功能，在掛載之前驗證設備軟體映像的完整性。如果修改設備軟體，則不會執行。
- 防止轉返 – 當您更新設備軟體時，設備會在 SoC （ 晶片上的系統 ） 上燒斷電子保險絲。每個軟體版本都預期要燒斷的保險絲數量增加，如果燒斷更多，則無法執行。

AWS Panorama 設備安全最佳實務

使用 AWS Panorama 設備時，請記住下列最佳實務。

- 實體保護設備 – 在封閉的伺服器機架或安全房間中安裝設備。將裝置的實體存取限制為授權人員。
- 保護設備的網路連線 – 將設備連接到限制存取內部和外部資源的路由器。設備需要連接到攝影機，其可以位於安全的內部網路上。它也需要連線到 AWS。僅使用第二個乙太網路連接埠進行實體備援，並將路由器設定為僅允許必要的流量。

  使用其中一個建議的網路組態來規劃您的網路配置。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 磁碟機 – 佈建設備之後，請移除 USB 磁碟機並進行格式化。在向 AWS Panorama 服務註冊後，設備不會使用 USB 隨身碟。格式化磁碟機以移除臨時登入資料、組態檔案和佈建日誌。
- 讓設備保持最新狀態 – 及時套用設備軟體更新。當您在 AWS Panorama 主控台中檢視設備時，主控台會通知您是否有可用的軟體更新。如需詳細資訊，請參閱管理 AWS Panorama 設備。

  使用 DescribeDevice API 操作，您可以比較 `LatestSoftware`和 `CurrentSoftware` 欄位來自動檢查更新。當最新的軟體版本與目前版本不同時，請使用主控台或使用 CreateJobForDevices 操作來套用更新。
- 如果您停止使用 設備，請將其重設 – 將設備移出安全資料中心之前，請將其完全重設。在設備關閉電源並插上電源的情況下，同時按下電源和重設按鈕 5 秒。這會從設備中刪除帳戶登入資料、應用程式和日誌。

  如需詳細資訊，請參閱AWS Panorama Appliance 按鈕和燈光。
- 限制對 AWS Panorama 和其他 AWS 服務的存取 – AWSPanoramaFullAccess 可讓您存取所有 AWS Panorama API 操作，並視需要存取其他服務。如果可能，政策會根據命名慣例限制對資源的存取。例如，它可讓您存取名稱開頭為 的 AWS Secrets Manager 秘密`panorama`。對於需要唯讀存取權或存取更特定資源集的使用者，請使用 受管政策做為最低權限政策的起點。

  如需詳細資訊，請參閱AWS Panorama 的身分型 IAM 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 的全域基礎設施 AWS 雲端。您負責維護在此基礎設施上託管內容的控制權。您也同時負責所使用 AWS 服務 的安全組態和管理任務。如需資料隱私權的詳細資訊，請參閱資料隱私權常見問答集。如需有關歐洲資料保護的相關資訊，請參閱 AWS 安全性部落格上的 AWS 共同的責任模型和 GDPR 部落格文章。

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

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

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

章節

傳輸中加密

AWS Panorama API 端點僅支援透過 HTTPS 的安全連線。當您使用 AWS 管理主控台、AWS 開發套件或 AWS Panorama API 管理 AWS Panorama 資源時，所有通訊都會使用 Transport Layer Security (TLS) 加密。AWS Panorama 設備與 AWS 之間的通訊也會使用 TLS 加密。AWS Panorama 設備與攝影機之間透過 RTSP 的通訊不會加密。

如需 API 端點的完整清單，請參閱《》中的 AWS 區域和端點AWS 一般參考。

AWS Panorama 設備

AWS Panorama 設備具有乙太網路、HDMI 影片和 USB 儲存體的實體連接埠。SD 卡插槽、Wi-Fi 和藍牙無法使用。USB 連接埠只會在佈建期間用來將組態封存傳輸至設備。

包含設備佈建憑證和網路組態的組態封存內容不會加密。AWS Panorama 不會存放這些檔案；只有在您註冊設備時才能擷取這些檔案。將組態封存檔傳輸至 設備後，請從您的電腦和 USB 儲存裝置將其刪除。

設備的整個檔案系統都會加密。此外，設備會套用數個系統層級保護，包括必要軟體更新的復原保護、已簽署的核心和開機載入器，以及軟體完整性驗證。

當您停止使用設備時，請執行完整重設以刪除應用程式資料並重設設備軟體。

應用程式

您可以控制部署到設備的程式碼。在部署安全性問題之前驗證所有應用程式程式碼，無論其來源為何。如果您在應用程式中使用第三方程式庫，請仔細考慮這些程式庫的授權和支援政策。

應用程式 CPU、記憶體和磁碟用量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操作產生負面影響。在合併或部署至生產環境之前，請分別測試應用程式。

應用程式資產（程式碼和模型）不會與您的帳戶、設備或建置環境中的存取隔離。AWS Panorama Application CLI 產生的容器映像和模型封存不會加密。針對生產工作負載使用個別帳戶，並只允許視需要進行存取。

其他服務

為了將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 Simple Storage Service 使用者指南》中的使用加密保護資料。

攝影機串流登入資料會靜態加密 AWS Secrets Manager。設備的 IAM 角色會授予它擷取秘密的許可，以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至 Amazon CloudWatch Logs。CloudWatch Logs 預設會加密此資料，並可設定為使用客戶受管金鑰。如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的使用 加密 CloudWatch Logs AWS KMS中的日誌資料。 Amazon CloudWatch

AWS Panorama 的身分和存取管理

AWS Identity and Access Management (IAM) 是一種 AWS 服務 ，可協助管理員安全地控制對 AWS 資源的存取。IAM 管理員可控制誰可以進行身分驗證 （ 登入 ） 和授權 （ 具有許可 ），以使用 AWS Panorama 資源。IAM 是您可以免費使用 AWS 服務 的 。

主題

目標對象

使用方式 AWS Identity and Access Management (IAM) 會根據您的角色而有所不同：

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

使用身分驗證

身分驗證是您 AWS 使用身分憑證登入 的方式。您必須以 AWS 帳戶根使用者、IAM 使用者或擔任 IAM 角色身分進行身分驗證。

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

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

AWS 帳戶 根使用者

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

IAM 使用者和群組

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

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

IAM 角色

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

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

使用政策管理存取權

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

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

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

身分型政策

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

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

資源型政策

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

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

存取控制清單 (ACL)

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

Amazon S3 AWS WAF和 Amazon VPC 是支援 ACLs的服務範例。如需進一步了解 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 支援其他政策類型，可設定更多常見政策類型授予的最大許可：

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

多種政策類型

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

AWS Panorama 如何與 IAM 搭配使用

在您使用 IAM 管理對 AWS Panorama 的存取之前，您應該了解哪些 IAM 功能可與 AWS Panorama 搭配使用。若要全面了解 AWS Panorama 和其他 AWS 服務如何與 IAM 搭配使用，請參閱《AWS IAM 使用者指南》中的與 IAM 搭配使用的 服務。

如需 AWS Panorama 使用許可、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 資源的許可。他們也無法使用 AWS 管理主控台 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 作業的所需許可。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

身分型政策會判斷您帳戶中的某個人員是否可以建立、存取或刪除 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 受管政策並邁向最低權限許可 – 若要開始將許可授予您的使用者和工作負載，請使用將許可授予許多常見使用案例的 AWS 受管政策。它們可在您的 中使用 AWS 帳戶。我們建議您定義特定於使用案例 AWS 的客戶受管政策，以進一步減少許可。如需更多資訊，請參閱《IAM 使用者指南》中的 AWS 受管政策或任務職能的AWS 受管政策。

- 套用最低權限許可 – 設定 IAM 政策的許可時，請僅授予執行任務所需的許可。為實現此目的，您可以定義在特定條件下可以對特定資源採取的動作，這也稱為最低權限許可。如需使用 IAM 套用許可的更多相關資訊，請參閱《IAM 使用者指南》中的 IAM 中的政策和許可。
- 使用 IAM 政策中的條件進一步限制存取權 – 您可以將條件新增至政策，以限制動作和資源的存取。例如，您可以撰寫政策條件，指定必須使用 SSL 傳送所有請求。如果透過特定 例如 使用服務動作 AWS 服務，您也可以使用條件來授予其存取權 CloudFormation。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM JSON 政策元素：條件。
- 使用 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作 – IAM Access Analyzer 驗證新政策和現有政策，確保這些政策遵從 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您撰寫安全且實用的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的使用 IAM Access Analyzer 驗證政策。
- 需要多重要素驗證 (MFA) – 如果您的案例需要 IAM 使用者或 中的根使用者 AWS 帳戶，請開啟 MFA 以提高安全性。如需在呼叫 API 操作時請求 MFA，請將 MFA 條件新增至您的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的透過 MFA 的安全 API 存取。

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

使用 AWS Panorama 主控台

若要存取 AWS Panorama 主控台，您必須擁有一組最低許可。這些許可必須允許您列出和檢視 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。

如需詳細資訊，請參閱AWS Panorama 的身分型 IAM 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或使用 或 AWS CLI AWS API 以程式設計方式完成此動作的許可。

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

```
                "iam: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 AWS Panorama 的 受管政策

AWS 受管政策是由 AWS AWS 受管政策建立和管理的獨立政策旨在為許多常用案例提供許可，以便您可以開始將許可指派給使用者、群組和角色。

請記住， AWS 受管政策可能不會授予特定使用案例的最低權限許可，因為這些許可可供所有 AWS 客戶使用。我們建議您定義使用案例專屬的客戶管理政策，以便進一步減少許可。

您無法變更 AWS 受管政策中定義的許可。如果 AWS 更新受 AWS 管政策中定義的許可，則更新會影響政策連接的所有主體身分（使用者、群組和角色）。 AWS 服務 當新的 啟動或新的 API 操作可用於現有服務時， AWS 最有可能更新 AWS 受管政策。

如需詳細資訊，請參閱《IAM 使用者指南》中的 AWS 受管政策。

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

- AWSPanoramaFullAccess – 提供 AWS Panorama、Amazon S3 中的 AWS Panorama 存取點、 中的設備憑證 AWS Secrets Manager和 Amazon CloudWatch 中的設備日誌的完整存取權。包含為 AWS Panorama 建立服務連結角色的許可。
- AWSPanoramaServiceLinkedRolePolicy – 允許 AWS Panorama 管理 AWS IoT、AWS Secrets Manager 和 AWS Panorama 中的資源。
- AWSPanoramaApplianceServiceRolePolicy – 允許 AWS Panorama 設備將日誌上傳至 CloudWatch，並從 AWS Panorama 建立的 Amazon S3 存取點取得物件。

AWS 受管政策的 AWS Panorama 更新

下表說明 AWS Panorama 受管政策的更新。

變更	描述	日期
AWSPanoramaApplianceServiceRolePolicy – 更新現有政策	將 StringLike 條件取代為 ArnLike，以寫入 ARNs。	2024-12-10
AWSPanoramaFullAccess – 更新現有政策	將 StringLike 條件取代為 ArnLike，以寫入 ARNs。	2024-12-10
AWSPanoramaFullAccess – 更新現有政策	新增使用者政策的許可，以允許使用者在 CloudWatch Logs 主控台中檢視日誌群組。	2022-01-13
AWSPanoramaFullAccess – 更新現有政策	已新增使用者政策的許可，以允許使用者管理 AWS Panorama 服務連結角色，以及存取其他 服務中的 AWS Panorama 資源，包括 IAM、Amazon S3、CloudWatch 和 Secrets Manager。	2021-10-20
AWSPanoramaApplianceServiceRolePolicy – 新政策	AWS Panorama 設備服務角色的新政策	2021-10-20

變更	描述	日期
AWSPanoramaService LinkedRolePolicy – 新政策	AWS Panorama 服務連結角色的新政策。	2021-10-20
AWS Panorama 開始追蹤變更	AWS Panorama 開始追蹤其 AWS 受管政策的變更。	2021-10-20

使用 的服務連結角色 AWS Panorama

AWS Panorama use AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是直接連結的唯一 IAM 角色類型 AWS Panorama。服務連結角色由 預先定義， AWS Panorama 並包含服務 AWS 代表您呼叫其他服務所需的所有許可。

服務連結角色可讓您更 AWS Panorama 輕鬆地設定，因為您不必手動新增必要的許可。 AWS Panorama 會定義其服務連結角色的許可，除非另有定義，否則 AWS Panorama 只能擔任其角色。定義的許可包括信任政策和許可政策，且該許可政策無法附加至其他 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。選擇具有連結的是，以檢視該服務的服務連結角色文件。

章節

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

的服務連結角色許可 AWS Panorama

AWS Panorama 使用名為 AWSServiceRoleForAWSPanorama 的服務連結角色 – 允許 AWS Panorama 管理 AWS IoT、AWS Secrets Manager 和 AWS Panorama 中的資源。

AWSServiceRoleForAWSPanorama 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 完成下列動作：

- 監控 AWS Panorama 資源
- 管理 AWS Panorama 設備的 AWS IoT 資源
- 存取 AWS Secrets Manager 秘密以取得攝影機登入資料

如需許可的完整清單，請在 IAM 主控台中檢視 AWSPanoramaServiceLinkedRolePolicy 政策。

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 建立服務連結角色 AWS Panorama

您不需要手動建立一個服務連結角色。當您在 AWS 管理主控台、 AWS CLI或 AWS API 中註冊設備時， 會為您 AWS Panorama 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。當您註冊設備時， 會再次為您 AWS Panorama 建立服務連結角色。

編輯 的服務連結角色 AWS Panorama

AWS Panorama 不允許您編輯 AWSServiceRoleForAWSPanorama 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」的編輯服務連結角色。

刪除 的服務連結角色 AWS Panorama

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。

若要刪除 AWSServiceRoleForAWSPanorama 所使用的 AWS Panorama 資源，請使用本指南下列各節中的程序。

- 刪除版本和應用程式

- 取消註冊設備

> **Note**
>
> 如果 AWS Panorama 服務在您嘗試刪除資源時使用角色，則刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

若要刪除 AWSServiceRoleForAWSPanorama 服務連結角色，請使用 IAM 主控台 AWS CLI、 或 AWS API。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 支援在提供服務的所有區域中使用服務連結角色。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問題，其中沒有執行動作許可的實體可以強制具有更多許可的實體執行該動作。在 中 AWS，跨服務模擬可能會導致混淆代理人問題。在某個服務 (呼叫服務) 呼叫另一個服務 (被呼叫服務) 時，可能會發生跨服務模擬。可以操縱呼叫服務來使用其許可，以其不應有存取許可的方式對其他客戶的資源採取動作。為了防止這種情況， AWS 提供工具，協助您保護所有 服務的資料，讓服務主體能夠存取您帳戶中的資源。

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，以限制將另一個 服務 AWS Panorama 提供給資源的許可。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 裝置的 ARN。

防範混淆代理人問題最有效的方法，是使用 `aws:SourceArn` 全域條件內容金鑰，以及資源的完整 ARN。如果不知道資源的完整 ARN，或者如果您指定了多個資源，請使用 `aws:SourceArn` 全域條件內容金鑰，同時使用萬用字元 (*) 表示 ARN 的未知部分。例如：`arn:aws:`*`servicename`*`::`*`123456789012`*`:*`。

如需保護 AWS Panorama 用於授予 AWS Panorama 設備許可之服務角色的安全說明，請參閱 保護設備角色。

故障診斷 AWS Panorama 身分和存取

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

主題

我無權在 AWS Panorama 中執行動作

如果 AWS 管理主控台 告知您無權執行 動作，則必須聯絡您的管理員尋求協助。您的管理員是為您提供使用者名稱和密碼的人員。

當 IAM `mateojackson` 使用者嘗試使用主控台檢視設備的詳細資訊，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。

我未獲得執行 iam:PassRole 的授權

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

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

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

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

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

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我想要允許帳戶 AWS 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。
- 若要了解如何 AWS 帳戶 在您擁有的 資源之間提供存取權，請參閱《IAM 使用者指南》中的在您擁有 AWS 帳戶 的另一個 中提供存取權給 IAM 使用者。
- 若要了解如何將資源的存取權提供給第三方 AWS 帳戶，請參閱《IAM 使用者指南》中的將存取權提供給第三方 AWS 帳戶 擁有。
- 如需了解如何透過聯合身分提供存取權，請參閱《IAM 使用者指南》中的將存取權提供給在外部進行身分驗證的使用者 (聯合身分)。
- 如需了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱《IAM 使用者指南》中的 IAM 中的跨帳戶資源存取。

AWS Panorama 的合規驗證

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

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

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

人員在場時的其他考量事項

以下是將 AWS Panorama 用於可能存在的人員案例時應考慮的一些最佳實務：

- 確保您了解並遵循使用案例的所有適用法律和法規。這可能包括與攝影機定位和視野相關的法律、放置和使用攝影機時的通知和看板要求，以及影片中可能存在的人員權利，包括其隱私權。
- 將攝影機對人員及其隱私權的影響納入考量。除了法律要求之外，請考慮在攝影機所在的區域發出通知是否適當，以及攝影機是否應置於清晰可見且無任何遮蔽處，因此人們並不意外他們可能位於攝影機上。
- 制定適當的政策和程序來操作攝影機，並檢閱從攝影機取得的資料。
- 針對從攝影機取得的資料，考慮適當的存取控制、用量限制和保留期間。

AWS Panorama 中的基礎設施安全性

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

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

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

在資料中心部署 AWS Panorama 設備

AWS Panorama 設備需要網際網路存取才能與 AWS 服務通訊。它也需要存取您的內部攝影機網路。請務必仔細考慮您的網路組態，並僅提供每個裝置所需的存取權。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。

您需負責下列事項：

- AWS Panorama 設備的實體和邏輯網路安全。
- 當您使用 AWS Panorama 設備時，請安全地操作網路連接的攝影機。
- 保持 AWS Panorama 設備與攝影機軟體的更新。
- 遵守您從生產環境收集的影片和影像內容相關的任何適用法律和法規，包括與隱私權相關的法律或法規。

AWS Panorama 設備使用未加密的 RTSP 攝影機串流。如需將 AWS Panorama 設備連線至網路的詳細資訊，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 中的執行期環境軟體

AWS Panorama 提供在 AWS Panorama 設備上以 Ubuntu Linux 為基礎的環境中執行應用程式程式碼的軟體。AWS Panorama 負責將設備映像中的軟體保持在最新狀態。AWS Panorama 會定期發行軟體更新，您可以使用 AWS Panorama 主控台加以套用。

您可以在應用程式的 中安裝程式庫，以便在應用程式程式碼中使用程式庫`Dockerfile`。若要確保建置之間的應用程式穩定性，請選擇每個程式庫的特定版本。定期更新您的相依性，以解決安全問題。

推出

下表顯示針對 AWS Panorama 服務、軟體和文件發佈功能和軟體更新的時間。為了確保您能夠存取所有功能，請將您的 AWS Panorama 設備更新為最新的軟體版本。如需版本的詳細資訊，請參閱連結主題。

變更	描述	日期
終止支援通知	終止支援通知：2026 年 5 月 31 日， AWS 將結束對 的支援 AWS Panorama。2026 年 5 月 31 日之後，您將無法再存取 AWS Panorama 主控台或 AWS Panorama 資源。如需詳細資訊，請參閱AWS Panorama 終止支援。	2025 年 5 月 20 日
更新 受管政策	AWS Identity and Access Management 的 受管政策 AWS Panorama 已更新。如需詳細資訊，請參閱 AWS 受管政策。	2024 年 12 月 10 日
設備軟體更新	7.0.13 版是一項主要版本更新，可變更設備管理軟體更新的方式。如果您限制從設備傳出的網路通訊，或將其連接到私有 VPC 子網路，則必須先允許存取其他端點和連接埠，才能套用更新。如需詳細資訊，請參閱變更日誌。	2023 年 12 月 28 日
設備軟體更新	6.2.1 版包含錯誤修正。如需詳細資訊，請參閱變更日誌。	2023 年 9 月 6 日

設備軟體更新	6.0.8 版包含錯誤修正和安全性改善。如需詳細資訊，請參閱變更日誌。	2023 年 7 月 6 日
設備軟體更新	5.1.7 版包含錯誤修正和錯誤處理改進。如需詳細資訊，請參閱變更日誌。	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 設備。若要授予使用者購買裝置的許可，請參閱 AWS Panorama 的身分型 IAM 政策。	2023 年 2 月 2 日
設備軟體更新	5.0.74 版包含錯誤修正和錯誤處理改進。如需詳細資訊，請參閱變更日誌。	2023 年 1 月 23 日
API 更新	新增`AllowMajorVersionUpdate` 選項至 `OTAJobConfig` ，讓設備的主要版本更新選擇加入。如需詳細資訊，請參閱 CreateJobForDevices。	2023 年 1 月 19 日
開發人員的新工具	您可以在 AWS Panorama 範例 GitHub 儲存庫中使用新工具「側邊載入」。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 README。	2022 年 11 月 16 日
應用程式基礎映像更新	1.2.0 版將逾時選項新增至 `video_in.get()` 、設定`AWS_REGION` 環境變數，並改善錯誤處理。如需詳細資訊，請參閱變更日誌。	2022 年 11 月 16 日

設備軟體更新	5.0.42 版包含錯誤修正和安全性更新。如需詳細資訊，請參閱變更日誌。	2022 年 11 月 16 日
設備軟體更新	5.0.7 版新增支援從遠端重新啟動設備，以及從遠端暫停攝影機串流。如需詳細資訊，請參閱變更日誌。	2022 年 10 月 13 日
設備軟體更新	4.3.93 版新增支援從離線裝置擷取日誌。如需詳細資訊，請參閱變更日誌。	2022 年 8 月 24 日
設備軟體更新	4.3.72 版包含錯誤修正和安全性更新。如需詳細資訊，請參閱變更日誌。	2022 年 6 月 23 日
AWS PrivateLink 支援	AWS Panorama 支援從私有子網路管理 AWS Panorama 資源的 VPC 端點。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	4.3.55 版改善了 `console_output` 日誌的儲存使用率。如需詳細資訊，請參閱變更日誌。	2022 年 5 月 5 日
Lenovo ThinkEdge® SE70	適用於 的新設備可從 Lenovo AWS Panorama 取得。Lenovo ThinkEdge® SE70 採用 Nvidia Jetson Xavier NX，支援與 AWS Panorama 設備相同的功能。如需詳細資訊，請參閱相容裝置。	2022 年 4 月 6 日

應用程式基礎映像更新	1.1.0 版可改善執行背景執行緒時的效能，並將旗標 (is_cached) 新增至媒體物件，指出映像是否新鮮。如需詳細資訊，請參閱 Gallery.ecr.aws。	2022 年 3 月 29 日
設備軟體更新	4.3.45 版新增對 GPU 存取和傳入連接埠的支援。如需詳細資訊，請參閱變更日誌。	2022 年 3 月 24 日
設備軟體更新	4.3.35 版可改善安全性和效能。如需詳細資訊，請參閱變更日誌。	2022 年 2 月 22 日
更新 受管政策	AWS Identity and Access Management 的 受管政策 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
佈建日誌	使用設備軟體 4.3.23 時，設備會在佈建期間將日誌寫入 USB 隨身碟。如需詳細資訊，請參閱 日誌。	2022 年 1 月 13 日
NTP 伺服器組態	您現在可以將 AWS Panorama 設備設定為使用特定 NTP 伺服器進行時鐘同步。使用其他聯網設定在設備設定期間設定 NTP 設定。如需詳細資訊，請參閱設定。	2022 年 1 月 13 日
其他區域	AWS Panorama 現可於亞太區域（新加坡）和亞太區域（雪梨）區域使用。	2022 年 1 月 13 日

設備軟體更新	4.3.4 版新增支援模型`precisionMode` 的設定，並更新記錄行為。如需詳細資訊，請參閱變更日誌。	2021 年 11 月 8 日
更新 受管政策	AWS Identity and Access Management 的 受管政策 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 AWS Panorama 設備，請造訪 AWS Panorama。	2021 年 10 月 20 日
預覽版	AWS Panorama 可透過美國東部（維吉尼亞北部）和美國西部（奧勒岡）區域的邀請取得。	2020 年 12 月 1 日