aws

Amazon Kinesis Video Streams WebRTC 開發人員指南

Kinesis Video Streams

Kinesis Video Streams: Amazon Kinesis Video Streams WebRTC 開發人員指南

Table of Contents

什麼是使用 WebRTC 的 Amazon Kinesis Video Streams？

WebRTC 是一種開放技術規格，可讓瀏覽器和行動應用程式之間透過簡單 API 進行即時通訊 (RTC)。利用對等技術在連線的對等節點之間進行即時資料交換，並提供人與人互動所需的低延遲媒體串流。WebRTC 規格除了有可靠和安全即時媒體和資料串流的通訊協定規格，還包含一組 IETF 通訊協定，包括用於建立點對點連線的 Interactive Connectivity Establishment、Traversal Using Relay around NAT (TURN) 和 Session Traversal Utilities for NAT (STUN)。

Amazon Kinesis Video Streams 以全受管功能提供符合標準的 WebRTC 實作。您可以使用 Amazon Kinesis Video Streams with WebRTC 安全地即時串流媒體，或在任何攝影機 IoT 裝置與符合 WebRTC 標準的行動或 Web 播放器之間，執行雙向音訊或視訊互動。因為是全受管功能，您不需要建置、執行或擴展任何與 WebRTC 相關的雲端基礎設施，例如訊號或媒體轉送伺服器，即可在應用程式和裝置之間安全地串流媒體。

使用 Kinesis Video Streams 搭配 WebRTC，您可以輕鬆地為即時peer-to-peer媒體串流建置應用程式，或為各種使用案例在攝影機 IoT 裝置、網頁瀏覽器和行動裝置之間建立即時音訊或視訊互動。這些應用程式可以幫助家長監控嬰兒的房間、讓屋主使用視訊門鈴檢查誰在門口、讓攝影功能掃地機器人的擁有者在手機上觀看即時攝影機串流，以從遠端控制機器人，諸如此類。

如果您是第一次使用 Kinesis Video Streams with WebRTC，我們建議您閱讀以下章節：

- 運作方式
- Amazon Kinesis Video Streams with WebRTC SDK in C for embedded device
- 適用於 Web 應用程式的 JavaScript 中的 Amazon Kinesis Video Streams with WebRTC SDK WebRTC
- 適用於 Android 的 Amazon Kinesis Video Streams WebRTC 開發套件
- 適用於 iOS 的 Amazon Kinesis Video Streams WebRTC 開發套件
- 控制平台 API
- 資料平面 REST API
- 資料平面 Websocket API

區域可用性

> **Note**
>
> 中尚不支援 Amazon Kinesis Video Streams with WebRTC AWS GovCloud (US) Region。

Amazon Kinesis Video Streams with WebRTC 可在下列區域使用：

區域名稱	AWS 區域碼
美國東部 (俄亥俄)	us-east-2
美國東部 (維吉尼亞北部)	us-east-1
美國西部 (奧勒岡)	us-west-2
Africa (Cape Town)	af-south-1
亞太地區 (香港)	ap-east-1
亞太區域 (孟買)	ap-south-1
亞太區域 (首爾)	ap-northeast-2
亞太區域 (新加坡)	ap-southeast-1
亞太區域 (雪梨)	ap-southeast-2
亞太地區 (馬來西亞)	ap-southeast-5
亞太區域 (東京)	ap-northeast-1
加拿大 (中部)	ca-central-1
歐洲 (法蘭克福)	eu-central-1
歐洲 (愛爾蘭)	eu-west-1
歐洲 (倫敦)	eu-west-2

區域名稱	AWS 區域碼
歐洲 (巴黎)	eu-west-3
歐洲 (西班牙)	eu-south-2
南美洲 (聖保羅)	sa-east-1
Middle East (Bahrain)	me-south-1

運作方式

Amazon Kinesis Video Streams with WebRTC 可在 Web 瀏覽器、行動裝置和其他支援 WebRTC 的應用程式之間進行即時視訊通訊。它提供安全且可擴展的基礎設施，用於建置影片串流應用程式、處理訊號、媒體串流等任務，以及與其他 AWS 服務的整合。本節說明 服務的基礎架構、元件和工作流程，協助解釋其設計原則和機制。

主題

重要概念

以下是 Amazon Kinesis Video Streams with WebRTC 特有的關鍵術語和概念。

訊號頻道

訊號頻道可讓應用程式交換訊號訊息，以探索、設定、控制和終止點對點連線。訊號訊息是兩個應用程式彼此交換以建立點對點連線的中繼資料。此中繼資料包括本機媒體資訊，例如媒體轉碼器和轉碼參數，以及兩個應用程式相互連線以進行即時串流可能使用的網路候選路徑。

串流應用程式可以維持與訊號頻道之間的持續連線，並等待其他應用程式來連接它們。或者，只在需要即時串流媒體時才連線到訊號頻道。訊號頻道可讓應用程式採用一個主節點連線到多個檢視器的概念，在一對少數的模型中相互連線。啟動連線的應用程式負起主節點的責任，使用 `ConnectAsMaster` API 並等待檢視器。然後，最多 10 個應用程式可以負起檢視器的責任，叫用

`ConnectAsViewer` API 來連線到該訊號頻道。連線到訊號頻道後，主節點應用程式和檢視器應用程式可以互相傳送訊號訊息，以建立點對點連線來即時媒體串流。

對等節點

設定為透過 Kinesis Video Streams with WebRTC 進行即時雙向串流的任何裝置或應用程式（例如，行動或 Web 應用程式、網路攝影機、家用安全攝影機、嬰兒監視器等）。

主

啟動連線並連線至訊號頻道的對等節點，能夠探索任何連接訊號頻道的檢視器並與之交換媒體。

> **Important**
>
> 目前，一個訊號頻道只能有一個主節點。

觀眾

連線至訊號頻道的對等節點，只能夠探索訊號頻道的主節點並與之交換媒體。檢視器無法透過給定的訊號頻道來探索其他檢視器或與之互動。一個訊號頻道最多可以有 10 個連線的檢視器。

技術概念

當您開始使用 Kinesis Video Streams with WebRTC 時，您也可以從了解 WebRTC 技術包含的幾個相互關聯的通訊協定和 APIs 中受益。

Session Traversal Utilities for NAT (STUN)

一種通訊協定，用來探索您的公有地址，並判斷路由器中會阻止直接連接對等節點的任何限制。

Traversal Using Relays around NAT (TURN)

一種伺服器，可對 TURN 伺服器建立連線，並透過該伺服器轉送資訊，以避開對稱 NAT 限制。

Session Description Protocol (SDP)

一種標準，描述連線的多媒體內容，例如解析度、格式，轉碼器、加密等，讓對等節點在資料開始傳輸時可以了解彼此。

SDP 提議

由代理程式傳送的 SDP 訊息，以產生工作階段描述來建立或修改工作階段。描述所需媒體通訊的各方面。

SDP 回答

由回答者傳送的 SDP 訊息，以回應來自提議者的提議。回答指出接受哪些方面。例如，是否接受提議中的所有音訊和視訊串流。

互動式連線建立 (ICE)

允許 Web 瀏覽器與對等節點連接的框架。

ICE 候選項

可供傳送端對等節點用來通訊的方法。

STUN、TURN 和 ICE 如何一起運作

假設有 A 和 B 兩個對等節點，兩者都使用 WebRTC 點對點雙向媒體串流 (例如，視頻聊天應用程式)。當 A 想打電話給 B 時會發生什麼情況？

為了連線至 B 的應用程式，A 的應用程式必須產生 SDP 提議。SDP 提議包含 A 的應用程式想建立的工作階段的相關資訊，包括使用什麼轉碼器、這是音訊還是視訊工作階段等。還包含 ICE 候選項清單，這是可供 B 的應用程式嘗試用來連線至 A 的 IP 和連接埠配對。

為了建置 ICE 候選項清單，A 的應用程式會向 STUN 伺服器提出一連串請求。伺服器傳回發出請求的公有 IP 地址和連接埠配對。A 的應用程式將每一對新增至 ICE 候選項清單，換句話說，收集 ICE 候選項。A 的應用程式一旦完成收集 ICE 候選項，就可以傳回 SDP。

接下來，A 的應用程式必須透過這些應用程式通訊的訊號頻道，將 SDP 傳送到 B 的應用程式。WebRTC 標準並未規定此交換的傳輸通訊協定。可以透過 HTTPS，安全 WebSocket 或任何其他通訊協定來執行。

現在，B 的應用程式必須產生 SDP 回答。B 的應用程式遵循 A 在上一步使用的相同步驟：收集 ICE 候選項等。然後，B 的應用程式需要將此 SDP 回答傳回至 A 的應用程式。

A 和 B 交換 SDPs之後，他們會執行一系列的連線檢查。在每個應用程式中，ICE 演算法採用在對方 SDP 中收到的清單中的候選 IP/連接埠配對，並將 STUN 請求傳送至此配對。如果另一個應用程式有所回應，則起源應用程序會認為檢查成功，並將該 IP/連接埠配對標示為有效的 ICE 候選項。

在所有 IP/連接埠配對上完成連線檢查後，應用程式會交涉並決定使用其中一個剩餘的有效配對。選取配對後，媒體就會開始在應用程式之間流動。

如果任一應用程式找不到通過連線檢查的 IP/連接埠配對，則會向 TURN 伺服器提出 STUN 請求，以取得媒體轉送地址。轉送位址是公有 IP 地址和連接埠，可轉送往返於應用程式收到的封包，以設定轉送地址。然後，此轉送地址會新增至候選項清單，並透過訊號頻道交換。

元件

Kinesis Video Streams with WebRTC 包含下列元件：

- 控制平台

  控制平面元件負責建立和維護 Kinesis Video Streams with WebRTC 訊號頻道。如需詳細資訊，請參閱 Amazon Kinesis Video Streams API 參考。

- 訊號

  訊號元件管理 WebRTC 訊號端點，可讓應用程式彼此安全地連線，以進行點對點即時媒體串流。訊號元件包含 Amazon Kinesis Video Signaling REST API 和一組 Websocket API。

- STUN

  此元件管理 STUN 端點，當應用程式位於 NAT 或防火牆後方時，這些端點可讓應用程式探索其公有 IP 地址。

- TURN

  此元件管理 TURN 端點，當應用程式無法點對點串流媒體時，這些端點可支援透過雲端轉送媒體。

- Kinesis Video Streams WebRTC 開發套件

  這些是您可以在裝置和應用程式用戶端下載、安裝和設定的軟體程式庫，可讓具有 WebRTC 功能的攝影機 IoT 裝置進行低延遲的點對點媒體串流。這些SDKs也可讓 Android、iOS 和 Web 應用程式用戶端將 Kinesis Video Streams 與 WebRTC 訊號、TURN 和 STUN 功能與任何 WebRTC 相容的行動或 Web 播放器整合。

  - Amazon Kinesis Video Streams with WebRTC SDK in C for embedded device
  - 適用於 Web 應用程式的 JavaScript 中的 Amazon Kinesis Video Streams with WebRTC SDK WebRTC
  - 適用於 Android 的 Amazon Kinesis Video Streams WebRTC 開發套件
  - 適用於 iOS 的 Amazon Kinesis Video Streams WebRTC 開發套件

系統要求

本節涵蓋搭配 WebRTC 使用 Amazon Kinesis Video Streams 的基本系統需求，包括網路需求和環境。它也包含偵錯連線的相關資訊。

網路要求

Kinesis Video Streams with WebRTC 訊號頻道服務端點的一般聯網需求如下：

- 託管於 的端點的 HTTPS 呼叫 `https://*.kinesisvideo.{region}.amazonaws.com`
- WebSocket 與端點的整合 `wss://*.kinesisvideo.{region}.amazonaws.com`
- STUN 伺服器位於 `stun:stun.kinesisvideo.{aws-region}.amazonaws.com:443`
- TURN 伺服器位於 `turn:_._.kinesisvideo.{aws-region}.amazonaws.com:443`和 `turns:_._.kinesisvideo.{aws-region}.amazonaws.com:443`

> **Note**
>
> STUN 和 TURN 伺服器目前不支援 IPv6 地址。

在對等之間用作 RTCPeerConnection 一部分的通訊協定可以是 TCP 或 UDP 型。

大多數應用程式會決定每個peer-to-peer的 IP 地址，以及做為 ICE 候選項目交換的連接埠和通訊協定，嘗試建立直接的點對點連線。這些候選項目用於嘗試使用這些候選項目彼此連線。他們會嘗試每個配對，直到可以建立連線為止。

網路環境

如果來自檢視器的訊息已傳送至主伺服器，且記錄`No valid ICE candidate`了 等日誌，則找不到有效的連線路由。如果防火牆防止直接連線，或無法連線網路，就可能會發生這種情況。

執行下列動作來疑難排解連線問題：

- 如果您不是TURN在主端使用 ，請務必啟用 TURN。

  TURN 根據預設， 會在 C 開發套件中啟用。在 JavaScript 開發套件中，選取 NAT 周遊`TURN only`中的 STUN/TURN或 。
- 對於受限的網路，例如企業網路，請嘗試其他可用的網路 （有線或無線）。

  如果您要連線至 VPN，請中斷與 VPN 的連線。

> **Note**
>
> 您可以忽略 Kinesis Video Streams TURN 伺服器傳回的 403 `Forbidden IP`錯誤。伺服器會拒絕包含 `localhost` IPs 的 ICE 候選配對，例如 `192.168.*`或 `10.0.0.*`。這可能會導致某些但不是全部的 ICE 候選配對失敗。

偵錯進行中連線

有幾個領域可能會導致已建立、持續的 WebRTC 連線發生問題，但聯網是最常見的。

您可以將 `export AWS_KVS_LOG_LEVEL=1`設定為環境變數，從 SDK 確認 VERBOSE 層級日誌。

> **Note**
>
> 如果在分配的逾時內找不到候選配對，ICE 代理程式會傳回錯誤狀態 `0x5a00000d`。

如需日誌層級的詳細資訊，請參閱 GitHub。

```
export AWS_KVS_LOG_LEVEL=1
./kvsWebrtcClientMasterGstSample TestChannel
```

您將看到類似以下的日誌。您可以從這些日誌確認互動式連線建立 (ICE) 候選 (IP 地址和連接埠） 和選取的候選配對。

```
2023-02-13 05:57:16 DEBUG   iceAgentReadyStateSetup(): Selected pair w1UdV9fE+_/
CuBel1pl, local candidate type: srflx. Round trip time 7 ms. Local candidate priority:
 1694498815, ice candidate pair priority: 7240977859836116990
2023-02-13 05:57:16 INFO    onConnectionStateChange(): New connection state 3
2023-02-13 05:57:16 DEBUG   rtcPeerConnectionGetMetrics(): ICE local candidate Stats
 requested at 16762678365731494
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate IP
 Address: XXX.XXX.XXX.XXX
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate
 type: srflx
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate
 port: 38563
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate
 priority: 1694498815
```

```
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate
 transport protocol: udp
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate
 relay protocol: N/A
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Local Candidate Ice
 server source: stun.kinesisvideo.ap-northeast-1.amazonaws.com
2023-02-13 05:57:16 DEBUG   rtcPeerConnectionGetMetrics(): ICE remote candidate Stats
 requested at 16762678365732111
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Remote Candidate IP
 Address: XXX.XXX.XXX.XXX
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Remote Candidate
 type: srflx
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Remote Candidate
 port: 45867
2023-02-13 05:57:16 VERBOSE signalingClientGetCurrentState(): Signaling Client Get
 Current State
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Remote Candidate
 priority: 1685921535
2023-02-13 05:57:16 DEBUG   logSelectedIceCandidatesInformation(): Remote Candidate
 transport protocol: udp
```

Amazon Kinesis Video Streams with WebRTC 服務配額

Kinesis Video Streams with WebRTC 具有下列服務配額：

> **Important**
>
> 下列服務配額是可增加的軟性 【s】，或無法增加的硬性 【h】。您會在下表中看到個別服務配額旁的 【s】 和 【h】。

> **Note**
>
> TPS 代表每秒的交易。

主題

- WebRTC 擷取服務配額
- 疑難排解

控制平面 API 服務配額

下列各節描述控制平面 API 的服務配額。

API	每個 的最大 API 請求率 AWS 帳戶	AWS 帳戶 每個 的頻道數目上限 AWS 區域	每個頻道的 API 請求上限
CreateSignalingChannel	50 TPS [s]	• 僅適用於美國東部（維吉尼亞北部）(us-east-1) 和美國西部（奧勒岡）(us-west-2) - 10，000 • 所有其他支援的區域 - 5，000	N/A
DeleteSignalingChannel	50 TPS [h]	N/A	5 TPS [h]
DescribeMediaStorageConfiguration	50 TPS [h]	N/A	5 TPS [h]
DescribeSignalingChannel	300 TPS [h]	N/A	5 TPS [h]
GetSignalingChannelEndpoint	300 TPS [h]	N/A	N/A
ListSignalingChannels	50 TPS [h]	N/A	N/A
ListTagsForResource	50 TPS [h]	N/A	5 TPS [h]
TagResource	50 TPS [h]	N/A	5 TPS [h]
UntagResource	50 TPS [h]	N/A	5 TPS [h]

API	每個 的最大 API 請求率 AWS 帳戶	AWS 帳戶 每個 的頻道數目上限 AWS 區域	每個頻道的 API 請求上限
UpdateMediaStorageConfiguration	10 TPS [h]	N/A	5 TPS [h]
UpdateSignalingChannel	50 TPS [h]	N/A	5 TPS [h]

訊號 API 服務配額

下一節說明 Kinesis Video Streams with WebRTC 中訊號元件的服務配額。如需詳細資訊，請參閱運作方式。

API	終止連線之前的 GO_AWAY 訊息寬限期上限	每個頻道的 API 請求率上限	每個頻道的並行連線數目上限	連線持續時間上限	閒置連線逾時期間上限
ConnectAsMaster	1 分鐘（小時）	3 TPS (h)	1 (h)	1 小時 (h)	10 分鐘（小時）
ConnectAsViewer	1 分鐘（小時）	3 TPS (h)	10（秒）	1 小時 (h)	10 分鐘（小時）

API	API 請求率上限	訊息承載大小上限
SendAlexaOffertoMaster	• 每個訊號通道 5 TPS (h) • AWS 區域 每個 AWS 帳戶 (100 TPS)	N/A
SendICECandidate	每個 WebSocket 連線 20 TPS (h)	10k (h)
SendSDPAnswer	每個 WebSocket 連線 5 TPS (h)	10k (h)

API	API 請求率上限	訊息承載大小上限
SendSDPOffer	每個 WebSocket 連線 5 TPS (h)	10k (h)

TURN 服務配額

下一節說明在 Kinesis Video Streams with WebRTC 中使用轉接圍繞 NAT (TURN) 元件的周遊服務配額。如需詳細資訊，請參閱運作方式。

API 或 參數	Value
GetIceServerConfig	• 每個訊號通道 5 TPS (h) • AWS 區域 每個 AWS 帳戶 (100 TPS)
位元速率	5Mbps (h)
登入資料生命週期	5 分鐘 （ 小時 ）
配置數量	每個訊號通道 50 個 (h)

WebRTC 擷取服務配額

下一節說明 Amazon Kinesis Video Streams WebRTC 中媒體錄製元件的服務配額。如需詳細資訊，請參閱搭配 WebRTC 使用 Amazon Kinesis Video Streams 來擷取和存放媒體。

JoinStorageSession

- API：
  - 每 AWS 帳戶 50 TPS (h)
  - 每個頻道 - 2 TPS (h)
- 串流工作階段配額：
  - 位元速率 - 1 Mbps （ 秒 ）
  - 工作階段持續時間 - 1 小時 (h)
  - 閒置逾時 - 3 分鐘 (h)

JoinStorageSessionAsViewer

- API：
  - 每 AWS 帳戶 50 TPS (h)
  - 每個頻道 - 2 TPS (h)
- 串流工作階段配額：
  - 工作階段中並行用戶端上限 - 3 個計數 (h)
  - 工作階段持續時間 - 1 小時 (h)
  - 閒置逾時 - 1 分鐘 (h)

疑難排解

您只能將一個主要和一或多個檢視器連接到單一訊號頻道。

無法將多個主控端連接到單一訊號頻道。

使用 WebRTC 存取 Kinesis Video Streams

您可以使用下列任何方式搭配 WebRTC 使用 Kinesis Video Streams：

AWS 管理主控台

入門 AWS 管理主控台

主控台是瀏覽器型界面，可用來存取和使用 AWS 服務，包括 Kinesis Video Streams with WebRTC。

AWS SDKs

AWS 提供軟體開發套件 (SDKs)，其中包含適用於各種程式設計語言和平台 （例如 Java、Python、Ruby、.NET、iOS、Android 等） 的程式庫和範本程式碼。SDKs提供使用 WebRTC 建立 Kinesis Video Streams 程式設計存取的便利方式。如需 AWS SDKs 的相關資訊，包括如何下載和安裝，請參閱適用於 Amazon Web Services 的工具。

Kinesis Video Streams with WebRTC HTTPS API

您可以使用 Kinesis Video Streams with WebRTC 並以 AWS 程式設計方式存取 Kinesis Video Streams with WebRTC APIs，這可讓您直接向服務發出 API 請求。如需詳細資訊，請參閱 Amazon Kinesis Video Streams API 參考。

開始使用

本節說明如何在 Amazon Kinesis Video Streams with WebRTC 中執行下列任務：

- 設定您的 AWS 帳戶 並建立管理員。
- 使用 WebRTC 訊號頻道建立 Kinesis Video Streams。
- 在 Ingenic T31 硬體上使用 WebRTC 設定 Kinesis Video Streams。

如果您是初次使用 Kinesis Video Streams with WebRTC，建議您運作方式先閱讀。

設定 AWS 帳戶

第一次將 Kinesis Video Streams 與 WebRTC 搭配使用之前，請先完成下列任務：

主題

- 註冊 AWS 帳戶
- 建立具有管理存取權的使用者
- 建立 AWS 帳戶金鑰

註冊 AWS 帳戶

如果您沒有 AWS 帳戶，請完成下列步驟來建立一個。

註冊 AWS 帳戶

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

   部分註冊程序需接收來電或簡訊，並在電話鍵盤輸入驗證碼。

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

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

建立具有管理存取權的使用者

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

保護您的 AWS 帳戶根使用者

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

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

2. 若要在您的根使用者帳戶上啟用多重要素驗證 (MFA)。

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

建立具有管理存取權的使用者

1. 啟用 IAM Identity Center。

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

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

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

以具有管理存取權的使用者身分登入

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

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

指派存取權給其他使用者

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

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的建立許可集。

2. 將使用者指派至群組，然後對該群組指派單一登入存取權。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的新增群組。

建立 AWS 帳戶金鑰

您需要 AWS 帳戶金鑰以程式設計方式存取 Kinesis Video Streams with WebRTC。

若要建立 AWS 帳戶金鑰，請執行下列動作：

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽列中選擇 Users (使用者)，然後選擇 Administrator (管理員) 使用者。
3. 開啟 Security credentials (安全性登入資料) 標籤，然後選擇 Create access key (建立存取金鑰)。
4. 記錄 Access key ID (存取金鑰 ID)。選擇在私密存取金鑰下顯示，然後記錄私密存取金鑰。

建立訊號頻道

Kinesis Video Streams with WebRTC 訊號頻道有助於交換在 WebRTC 用戶端之間建立和維護peer-to-peer連線所需的訊號訊息。它處理工作階段描述協定 (SDP) 提供的交涉和工作階段參數的答案，以及交換互動式連線建立 (ICE) 候選網路資訊。

若要建立訊號頻道，請呼叫 CreateSignalingChannel API。此頁面將示範如何使用 AWS 管理主控台 AWS CLI和其中一個 AWS SDKs 叫用該 API。

> **Important**
>
> 請記下頻道 ARN，您稍後會需要它。

AWS 管理主控台

請執行下列操作：

1. 開啟位於 https：//https://console.aws.amazon.com/kinesisvideo/home/#/signalingChannels 的 Kinesis Video Streams Signaling Channels 主控台。
2. 選擇 Create signaling channel (建立訊號頻道)。
3. 在建立新的訊號頻道頁面上，輸入訊號頻道的名稱。

將預設Time-to-live(Ttl) 值保留為 60 秒。

選擇 Create signaling channel (建立訊號頻道)。

4. 建立訊號頻道後，請檢閱頻道詳細資訊頁面上的詳細資訊。

AWS CLI

確認您已安裝 AWS CLI 並設定 。如需詳細資訊，請參閱「AWS Command Line Interface 使用者指南」。

如需安裝說明，請參閱AWS Command Line Interface 《 使用者指南》。安裝後，使用登入資料和區域設定 AWS CLI。

或者，開啟已安裝並設定 AWS CLI 的 AWS CloudShell 終端機。如需詳細資訊，請參閱 AWS CloudShell 使用者指南。

使用 執行下列 Create-Signaling-Channel 命令 AWS CLI：

```
aws kinesisvideo create-signaling-channel \
  --channel-name "YourChannelName" \
  --region "us-west-2"
```

回應如下所示：

```
{
    "ChannelARN": "arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123"
}
```

AWS SDK

此程式碼片段說明如何使用適用於 JavaScript 的 AWS SDK v2 建立 Kinesis Video Streams with WebRTC 訊號頻道。語法將與其他 AWS SDKs 不同，但一般流程將相同。在 GitHub 上檢視完整的程式碼範例。

建立 Kinesis Video Streams 用戶端。這是用來呼叫 `CreateSignalingChannel` API 的用戶端。

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
```

```
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

使用 用戶端呼叫 `CreateSignalingChannel` API。

```
const createSignalingChannelResponse = await kinesisVideoClient
    .createSignalingChannel({
        ChannelName: 'YourChannelName',
    })
    .promise();
```

列印回應。

```
console.log(createSignalingChannelResponse.ChannelARN);
```

具有此程式碼範例的即時網頁可在 GitHub 上使用。輸入您的區域、 AWS 憑證和訊號頻道的名稱。

選取建立頻道。

與 Ingenic T31 整合

> **Note**
>
> AWS 不以任何方式為此晶片組背書，也不保證整合可以正常運作。本指南是以 Amazon Kinesis Video Streams 團隊的測試為基礎，旨在協助客戶設定裝置。

請依照這些程序，在 Ingenic T31 硬體上使用 WebRTC 設定 Amazon Kinesis Video Streams。

下載程式碼

1. 設定建置目錄。

   在本教學課程中，請在 `Downloads` 資料夾中建立名為 `ingenic`的目錄。

   ```
   mkdir ~/Downloads/ingenic
   ```

```
cd ~/Downloads/ingenic
```

2. 將下列儲存庫複製到您的新目錄：https：//https://github.com/aws-samples/amazon-kinesis-video-streams-media-interface。

```
git clone https://github.com/aws-samples/amazon-kinesis-video-streams-media-
interface.git
```

設定建置環境

1. 取得包含工具鏈的預先建置 Docker 影像，並將其放在 ingenic 資料夾中。

> **Important**
>
> 工具鏈可能會因主機板和 CPU 類型而有所不同。請洽詢廠商以取得正確的工具鏈。

2. 載入 Ingenic-T31-app-build.tar.gz docker 映像。

```
docker load --input ./Ingenic-T31-app-build.tar.gz
```

預期的輸出結果：

```
580272b5675c: Loading layer [==================================================>]
 1.666GB/1.666GB
Loaded image ID:
 sha256:76d41ef9b2f53ad3f2a33f00ae110df3d1b491378a4005e19ea989ce97e99bc1
```

請記下 Image Id。在下一個步驟中，您將需要此值。

3. 執行下列命令，將主機電腦上的目前目錄 (~/Downloads/ingenic) 掛載到 Docker 容器內的/ingenicMappedFolder目錄。

此命令也會使用終端機工作階段，以互動式模式執行容器。系統會將您放入容器內的 Bash shell，可讓您直接執行命令。

```
docker run \
   -v `pwd`:/ingenicMappedFolder \
   -it 76d41ef9b2f53ad3f2a33f00ae110df3d1b491378a4005e19ea989ce97e99bc1 \
   /bin/bash
```

4. 檢閱/ingenicMappedFolder資料夾的內容，以確認綁定掛載磁碟區成功。

   提示：

   ```
   ls /ingenicMappedFolder
   ```

   回應：

   ```
   AWS-Solution-Remote-Diagnostic-Media-Interface Ingenic-T31-app-build.tar.gz
   ```

5. 確認 Docker 容器已設定環境變數：

   提示：

   ```
   env
   ```

   回應：

   ```
   CC=mips-linux-gnu-gcc
   CXX=mips-linux-gnu-g++
   PATH=/mips-gcc540-glibc222-64bit-r3.3.0/bin:/usr/local/sbin:/usr/local/bin:/usr/
   sbin:/usr/bin:/sbin:/bin
   ```

6. 確認 Docker 容器包含工具鏈和 Ingenic SDK。它應該位於 /ingenic-sdk。

   提示：

   ```
   ls /
   ```

   回應：

   ```
   bin   ingenic-sdk          libx32                              proc  sys
   boot  ingenicMappedFolder  media                               root  tmp
   dev   lib                  mips-gcc540-glibc222-64bit-r3.3.0  run   usr
   etc   lib32                mnt                                 sbin  var
   home  lib64                opt                                 srv
   ```

使用 WebRTC 應用程式建置 Amazon Kinesis Video Streams

1. 輸入下列內容：

```
cd /ingenicMappedFolder/AWS-Solution-Remote-Diagnostic-Media-Interface
cp -r /ingenic-sdk/* \
    /ingenicMappedFolder/amazon-kinesis-video-streams-media-interface/3rdparty/T31/
export LDFLAGS=-Wl,--dynamic-linker=/lib/ld.so.1
mkdir build
cmake -B ./build -DBOARD=T31 -DCMAKE_BUILD_TYPE=Release -DBUILD_WEBRTC_SAMPLES=ON \
    -DBUILD_KVS_SAMPLES=ON -DBUILD_SAVE_FRAME_SAMPLES=ON
cmake --build ./build --config Release
```

`kvswebrtcmaster-static` 應用程式位於 `/ingenicMappedFolder/AWS-Solution-Remote-Diagnostic-Media-Interface/build/samples/webrtc/`。

2. 執行下列命令以結束 Docker 容器：

```
exit
```

3. 在您的主機電腦上，驗證`kvswebrtcmaster-static`應用程式是否存在。

```
ls ~/Downloads/ingenic/AWS-Solution-Remote-Diagnostic-Media-Interface/build/
samples/webrtc/kvswebrtcmaster-static
```

將應用程式上傳至裝置

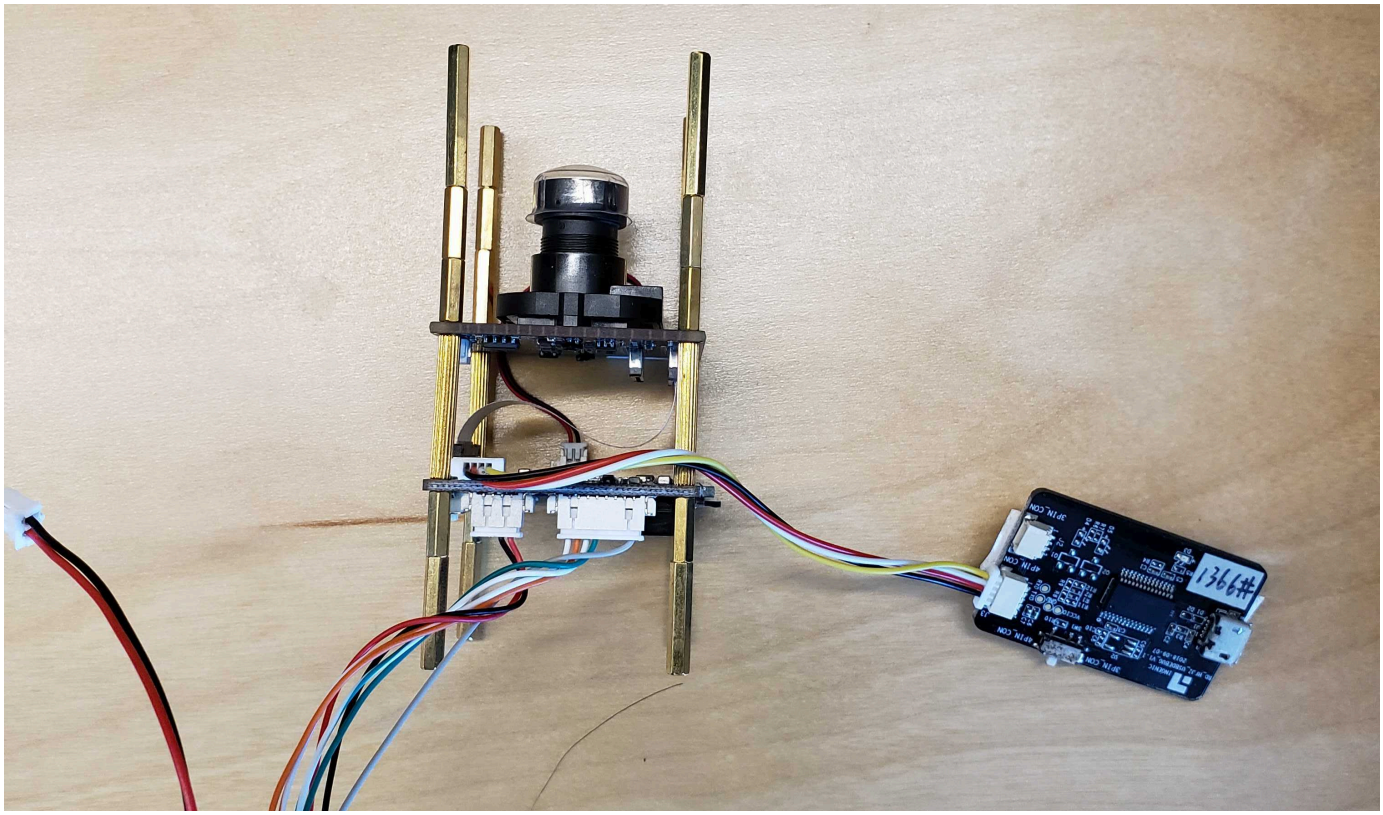
1. 在主機機器上，將靜態二進位檔複製到微型 SD 卡。

```
cp ~/Downloads/ingenic/AWS-Solution-Remote-Diagnostic-Media-Interface/build/
samples/webrtc/kvswebrtcmaster-static /Volumes/IngenicSDCard
```

2. 下載此 .pem 檔案，並將其放在微型 SD 卡中做為 `cert.pem`。
3. 將微型 SD 卡放入微型 SD 卡插槽。

連線至裝置

1. 將序列連接埠工具連接至電路板，如下所示。



2. 將乙太網路和電源線插入裝置。紅色 LED 應該會開啟。
3. 將主機機器連接到序列連接埠工具上的 micro-usb 插槽。
4. 檢查連線的裝置：

```
ls /dev/tty.*
```

> **Note**
>
> 如果您有多個 TTY 裝置，請判斷哪個 TTY 裝置是 Ingenic 電路板。中斷裝置與主機的連線，然後`ls`再次執行。比較命令輸出並找出差異。

5. 使用 `screen`或其他序列連接埠工具 （例如，Tera Term) 與其連線。將傳輸速率設定為 115200。

```
screen /dev/tty.usbserial-XXXXXXX 115200
```

6. 根據電路板的狀態決定適當的動作：

   - 如果 Shell 工作階段以 結尾`#`，則開機會中斷。使用 `boot`命令來啟動 Linux 作業系統，然後繼續此步驟的其餘部分。

- 如果 Shell 工作階段要求您登入，請輸入您的密碼。
- 如果畫面空白，請按 Enter。
- 如果畫面顯示錯誤 `Cannot exec '/dev/tty.usbserial-XXXXXXX': No such file or directory`，請仔細檢查主機板和主機機器之間的所有實體連線。

將微型 SD 卡掛載在電路板上

1. 建立目錄：

   > **Note**
   >
   > 在此範例中，我們使用 `sdcard`。

   ```
   mkdir /tmp/sdcard
   ```

2. 輸入下列內容，將 micro SD 卡掛載至 資料夾：

   ```
   mount /dev/mmcblk0p1 /tmp/sdcard
   ```

3. 檢閱資料夾的內容：

   ```
   ls /tmp/sdcard
   ```

執行應用程式

1. 導覽至掛載目錄：

   ```
   cd /tmp/sdcard
   ```

2. 從電路板匯出您的登入資料和其他資訊：

   ```
   export AWS_ACCESS_KEY_ID=ID
   export AWS_SECRET_ACCESS_KEY=key
   export AWS_DEFAULT_REGION=us-west-2
   export AWS_KVS_CACERT_PATH=`pwd`/cert.pem
   ```

3. 執行應用程式：

```
./kvswebrtcmaster-static channel-name
```

檢視媒體

若要檢視媒體，請以檢視器身分連線至訊號頻道。如需範例，請參閱下列各節：

- JavaScript
- AWS 管理主控台
- Android
- iOS

疑難排解

本節包含我們遇到的常見問題和問題。

當我將主機板連接至主機機器並執行 時`ls /dev/tty.*`，裝置不會出現。

確認連線和所有線路都已受保護，且裝置已開啟電源。檢查 USB-to-TTY 裝置上的任何 LED 指示燈是否亮起。如果您仍然有問題，請聯絡廠商以進一步診斷連接到電路板的問題。

應用程式無法連線至訊號

如果 收到下列其中一個錯誤：

```
SSL error: certificate is not yet valid
```

或

```
2024-09-19 08:56:34.920 WARN    lwsHttpCallbackRoutine(): Received client http read
 response:  { "message": "Signature expired: 20240919T085634Z is now earlier than
 20240919T155135Z (20240919T155635Z - 5 min.)" }
```

這表示您的 Ingenic 電路板上的時間不正確。執行 date 命令來驗證這一點。根據廠商的指示設定時間，然後再次執行應用程式。

即使檔案存在，應用程式仍無法啟動

如果錯誤看起來像：

```
-sh ./kvswebrtcmaster-static: not found
```

這表示作業系統找不到應用程式 ELF 標頭中列出的相依性。例如，libc在裝置上。

通常，這表示所使用的工具鏈與電路板不相容，或連結器旗標 (LDFLAGS) 未正確指向所需的程式庫路徑，導致執行時間無法解決的相依性。

請確定正確的工具鏈 (uclibc 與 glibc及其版本） 符合主機板上的軟體，並根據 LDFLAGS 設定the section called “使用 WebRTC 應用程式建置”。

如何卸載微型 SD 卡？

若要卸載微型 SD 卡，請使用 命令：

```
umount /tmp/sdcard
```

串流即時媒體 (SDKs)

在 Amazon Kinesis Video Streams WebRTC 中，對等是設定為透過訊號頻道進行即時雙向串流的裝置。Amazon Kinesis Video Streams with WebRTC SDKs easy-to-use的軟體程式庫，您可以下載並安裝在您想要設定為指定訊號頻道之對等的裝置和應用程式用戶端上。

Amazon Kinesis Video Streams with WebRTC 包含下列SDKs：

- Amazon Kinesis Video Streams with WebRTC SDK in C for embedded device
- 適用於 Web 應用程式的 JavaScript 中的 Amazon Kinesis Video Streams with WebRTC SDK WebRTC
- 適用於 Android 的 Amazon Kinesis Video Streams WebRTC 開發套件
- 適用於 iOS 的 Amazon Kinesis Video Streams WebRTC 開發套件

每個 SDK 都包含對應的範例和逐步指示，可協助您建置和執行這些應用程式。您可以使用這些範例進行低延遲、即時、雙向的音訊和視訊串流，以及在 Web/Android/iOS 應用程式或嵌入式裝置的任何組合之間進行資料交換。換句話說，您可以將嵌入式攝影機裝置的即時音訊和視訊串流至 Android 或 Web 應用程式，或在兩個 Android 應用程式之間串流。

Amazon Kinesis Video Streams with WebRTC SDK in C for embedded device

以下step-by-step說明說明如何下載、建置和執行適用於嵌入式裝置的 C 中的 Kinesis Video Streams with WebRTC SDK 及其對應的範例。

支援下列轉碼器：

- 音訊：
  - G.711 A-Law
  - G.711 U-Law
  - Opus
- 影片：
  - H.264
  - H.265

- VP8

下載軟體開發套件

若要下載適用於嵌入式裝置的 C 中的 Kinesis Video Streams with WebRTC SDK，請執行下列命令：

```
$ git clone https://github.com/awslabs/amazon-kinesis-video-streams-webrtc-sdk-c.git
```

建置 SDK

> **Important**
>
> 在 macOS 上完成這些步驟之前，視您擁有的 macOS 版本而定，您必須執行 `xcode-select --install`以使用命令列工具和標頭下載套件。然後開啟 `/Library/Developer/CommandLineTools/Packages/macOS_SDK_headers_for_macOS_10.14.pkg` 並遵循安裝程式來安裝命令列工具和標頭。您只需要在叫用 之前執行一次。 `cmake`如果您已安裝命令列工具和標頭，則不需要再次執行此命令。

請完成下列步驟：

1. 安裝 CMake：
   - 在 macOS 上，執行 `brew install cmake pkg-config srtp`
   - 在 Ubuntu 上，執行 `sudo apt-get install pkg-config cmake libcap2 libcap-dev`
2. 取得您要用於此示範之 AWS 帳戶 的存取金鑰和私密金鑰。
3. 執行以下命令，在您下載的 WebRTC C 開發套件中建立 `build` 目錄，然後從中執行 `cmake`：

   ```
   $ mkdir -p amazon-kinesis-video-streams-webrtc-sdk-c/build; cd amazon-kinesis-video-streams-webrtc-sdk-c/build; cmake ..
   ```
4. 現在您已位於您剛使用上述步驟建立的`build`目錄中，請執行 `make` 來建置 WebRTC C 開發套件及其提供的範例。

> **Note**
>
> 如果系統尚未gstreamer安裝 ，則kvsWebrtcClientMasterGstSample不會建置 。若要確保其已建置 （ 在 macOS 上 ），您必須執行： `brew install gstreamer gst-plugins-base gst-plugins-good`

執行 SDK 範例

完成上述程序之後，最後在 build 目錄中會出現下列範例應用程式：

- kvsWebrtcClientMaster - 此應用程式透過訊號頻道傳送範例 H264/Opus 影格 (路徑：/samples/h264SampleFrames 和 /samples/opusSampleFrames)。也會接受傳入的音訊 (如果在瀏覽器中啟用)。在瀏覽器中檢查時，將會在終端機列印收到的音訊封包的中繼資料。
- kvsWebrtcClientViewer - 此應用程式接受並印列範例 H264/Opus 影格。
- kvsWebrtcClientMasterGstSample - 此應用程式從 GStreamer 管道傳送範例 H264/Opus 影格。

若要執行任何這些範例，請完成下列步驟：

1. 使用 AWS 帳戶 登入資料設定您的環境：

```
export AWS_ACCESS_KEY_ID=YourAccessKey
export AWS_SECRET_ACCESS_KEY=YourSecretKey
export AWS_DEFAULT_REGION=YourAWSRegion
```

如果您使用的是暫時 AWS 登入資料，也請匯出工作階段字符：

```
export AWS_SESSION_TOKEN=YourSessionToken
```

如果您有自訂 CA 憑證路徑要設定，您可以使用下列方式設定：

```
export AWS_KVS_CACERT_PATH=../certs/cert.pem
```

> **Note**
>
> 根據預設，SSL CA 憑證設定為 ../certs/cert.pem，指向 GitHub 中此儲存庫中的 檔案。

2. 將您要給訊號頻道的名稱傳給其中一個範例應用程式，以執行應用程式。應用程式會使用您提供的名稱建立訊號頻道。例如，若要建立名為 `myChannel` 的訊號頻道，並開始透過此頻道傳送範例 H264/Opus 影格，請執行下列命令：

   ```
   ./kvsWebrtcClientMaster myChannel
   ```

   當命令列應用程式列印 `Connection established` 時，您可以繼續進行下一個步驟。

3. 現在，訊號頻道已建立，且連線的主節點正在將媒體串流至訊號頻道，您可以檢視此串流。例如，您可以在 Web 應用程式中檢視此即時串流。若要這樣做，請使用 中的步驟開啟 WebRTC SDK 測試頁面，使用範例應用程式並使用您為上述主伺服器指定的相同 AWS 登入資料和相同訊號通道來設定下列值：

   - 存取金鑰 ID
   - 私密存取金鑰
   - 訊號頻道名稱
   - 用戶端 ID (選擇性)

   選擇 Start viewer (啟動檢視器)，開始範例 H264/Opus 影格的即時視訊串流。

影片教學課程

此影片示範如何連接相機並開始使用 Amazon Kinesis Video Streams for WebRTC。

適用於 Web 應用程式的 JavaScript 中的 Amazon Kinesis Video Streams with WebRTC SDK WebRTC

您可以在適用於 Web 應用程式的 JavaScript 中找到 Kinesis Video Streams with WebRTC SDK，以及其在 GitHub 中的對應範例。 JavaScript

主題

- 安裝軟體開發套件

- WebRTC JavaScript SDK 文件
- 使用範例應用程式
- 編輯範例應用程式

安裝軟體開發套件

在 JavaScript 中使用 WebRTC 開發套件安裝 Kinesis Video Streams 是否及如何安裝取決於程式碼是否在Node.js模組或瀏覽器指令碼中執行。

NodeJS module

在適用於 Node.js 的 JavaScript 中使用 WebRTC 開發套件安裝 Kinesis Video Streams 的首選方法是使用 Node.js 套件管理員 npm。

套件託管於 https：//https://www.npmjs.com/package/amazon-kinesis-video-streams-webrtc。

若要在Node.js專案中安裝此開發套件，請使用終端機導覽至與您專案的 相同的目錄package.json：

輸入下列內容：

```
npm install amazon-kinesis-video-streams-webrtc
```

您可以匯入 SDK 類別，例如典型的 Node.js 模組：

```
// JavaScript
const SignalingClient = require('amazon-kinesis-video-streams-
webrtc').SignalingClient;
// TypeScript
import { SignalingClient } from 'amazon-kinesis-video-streams-webrtc';
```

Browser

您無需另行安裝，也能在瀏覽器指令碼中使用軟體開發套件。您可以使用 HTML 頁面中的 AWS 指令碼，直接從 載入託管 SDK 套件。

若要在瀏覽器中使用 SDK，請將下列指令碼元素新增至 HTML 頁面：

```
<script src="https://unpkg.com/amazon-kinesis-video-streams-webrtc/dist/kvs-
webrtc.min.js"></script>
```

當您將軟體開發套件載入至頁面後，便可從全域變數 KVSWebRTC (或 window.KVSWebRTC) 取得該軟體開發套件。

例如 window.KVSWebRTC.SignalingClient。

WebRTC JavaScript SDK 文件

SDK 方法的文件位於 GitHub 讀我檔案下。

在用量區段中，還有其他資訊可整合此 SDK 與適用於 JavaScript 的 AWS SDK，以建置 Web 型檢視器應用程式。

如需完整應用程式的範例，包括主要和檢視器角色，請參閱 examples目錄。

使用範例應用程式

Kinesis Video Streams with WebRTC 也會託管範例應用程式，您可以使用該應用程式建立新的訊號頻道，或連接到現有的頻道，並將其用作主頻道或檢視器。

Kinesis Video Streams with WebRTC 範例應用程式位於 GitHub。

範例應用程式的程式碼位於 examples目錄中。

主題

- 從範例應用程式將peer-to-peer串流至 AWS 管理主控台
- 從範例應用程式將peer-to-peer串流至範例應用程式
- 使用 WebRTC 擷取從範例頁面串流peer-to-peer範例頁面

從範例應用程式將peer-to-peer串流至 AWS 管理主控台

1. 開啟 Kinesis Video Streams with WebRTC 範例應用程式，並完成下列操作：

   - AWS 區域。 例如，us-west-2。
   - IAM 使用者或角色的 AWS 存取金鑰和私密金鑰。如果您使用長期 AWS 登入資料，請將工作階段字符保留空白。
   - 您要連線的訊號頻道的名稱。

     如果您想要連接到新的訊號頻道，請選擇建立頻道，以使用方塊中提供的值來建立訊號頻道。

> **Note**
>
> 目前帳戶和區域的訊號頻道名稱必須是唯一的。您可以使用字母、數字、底線 (_) 和連字號 (-)，但不能使用空格。

- 是否要傳送音訊、視訊或兩者。
- WebRTC 擷取和儲存。展開節點，然後選擇下列其中一項：
  - 選取自動判斷擷取模式。
  - 確定未選取自動判斷擷取模式，並將手動覆寫設定為 OFF。

> **Note**
>
> 自動判斷擷取模式可讓應用程式呼叫 DescribeMediaStorageConfiguration API，以決定要在哪個模式中執行 (Peer-to-peer或 WebRTC 擷取）。此額外的 API 呼叫會將少量新增至啟動時間。
> 如果您事先知道此訊號頻道執行的模式，請使用手動覆寫略過此 API 呼叫。

- 產生 ICE 候選項目。保持STUN/TURN選取並保持`Trickle ICE`啟用狀態。

2. 選擇 Start Master 以連接至訊號頻道。

   如有需要，允許存取您的攝影機和/或麥克風。

3. 在 中開啟 Kinesis Video Streams 主控台 AWS 管理主控台。

   確定已選取正確的區域。

4. 在左側導覽中，選取訊號頻道。

   選取上述訊號頻道的名稱。如有需要，請使用搜尋列。

5. 展開媒體播放檢視器區段。
6. 選擇影片播放器上的播放按鈕。這會以 的形式加入 WebRTC 工作階段`viewer`。

   在示範頁面上傳送的媒體應該會顯示在 中 AWS 管理主控台。

從範例應用程式將peer-to-peer串流至範例應用程式

1. 開啟 Kinesis Video Streams with WebRTC 範例應用程式，並完成下列資訊：

- AWS 區域。 例如，us-west-2。
- IAM 使用者或角色的 AWS 存取金鑰和私密金鑰。如果您使用長期 AWS 登入資料，請將工作階段字符保留空白。
- 您要連線的訊號頻道的名稱。

  如果您想要連接到新的訊號頻道，請選擇建立頻道，以使用方塊中提供的值來建立訊號頻道。

  > **Note**
  >
  > 目前帳戶和區域的訊號頻道名稱必須是唯一的。您可以使用字母、數字、底線 (_) 和連字號 (-)，但不能使用空格。

- 是否要傳送音訊、視訊或兩者。
- WebRTC 擷取和儲存。展開節點，然後選擇下列其中一項：
  - 選取自動判斷擷取模式。
  - 確定未選取自動判斷擷取模式，並將手動覆寫設定為 OFF。

  > **Note**
  >
  > 自動判斷擷取模式可讓應用程式呼叫 DescribeMediaStorageConfiguration API，以決定要在哪個模式中執行 (Peer-to-peer或 WebRTC 擷取 ）。此額外的 API 呼叫會將少量新增至啟動時間。
  > 如果您事先知道此訊號頻道執行的模式，請使用手動覆寫略過此 API 呼叫。

- 產生 ICE 候選項目。保持STUN/TURN選取並保持Trickle ICE啟用狀態。

2. 選擇 Start Master 以做為master角色連線至訊號頻道。

   如有需要，允許存取您的攝影機和/或麥克風。

3. 開啟另一個瀏覽器索引標籤，並開啟 Kinesis Video Streams with WebRTC 範例應用程式。應載入先前執行的所有資訊。
4. 向下捲動並選擇啟動檢視器，以做為viewer角色連線至訊號頻道。

   您應該會看到在 master和 之間交換的媒體viewer。

使用 WebRTC 擷取從範例頁面串流peer-to-peer範例頁面

1. 遵循 the section called “從瀏覽器擷取媒體” 來連接主要參與者，並確認其已連線至儲存工作階段。
2. 遵循 the section called “將檢視器新增至擷取工作階段” 新增瀏覽者參與者。

   檢視器參與者將連線至儲存工作階段並從中接收媒體。他們可以將選用的音訊傳回儲存工作階段。

   儲存工作階段會處理混合從主要和檢視器參與者收到的媒體，並將其傳送至適當的目的地。
3. 您可以透過 Kinesis Video Streams 播放來檢視和使用擷取的媒體。

編輯範例應用程式

若要編輯 SDK 和範例應用程式以進行開發，請遵循下列指示。

必要條件

NodeJS 16+ 版

> **Note**
>
> 我們建議您從 https://nodejs.org/en/download：// 下載最新的長期支援 (LTS) 版本。

編輯範例應用程式

1. 在 JavaScript 中下載 Kinesis Video Streams with WebRTC SDK。

   在終端機中輸入下列項目：

   ```
   git clone https://github.com/awslabs/amazon-kinesis-video-streams-webrtc-sdk-js.git
   ```
2. 使用 package.json 檔案導覽至 目錄。檔案位於儲存庫的根目錄中。

   在終端機中輸入下列項目：

   ```
   cd amazon-kinesis-video-streams-webrtc-sdk-js
   ```
3. 安裝依存項目。

   在終端機中輸入下列 npm CLI 命令：

```
npm install
```

4. 啟動 Web 伺服器以開始提供網頁。

   在終端機中輸入下列 npm CLI 命令：

```
npm run develop
```

5. 在您的瀏覽器中，請造訪 http：//localhost：3001/。

   您可以編輯 `examples`目錄中的檔案來編輯網頁。

適用於 Android 的 Amazon Kinesis Video Streams WebRTC 開發套件

下列逐步指示說明如何下載、建置和執行適用於 Android 的 Kinesis Video Streams with WebRTC 開發套件及其對應的範例。

> **Note**
>
> Amazon Kinesis Video Streams 不支援 Android 上的 IPv6 地址。請參閱有關在 Android 裝置上停用 IPv6 的詳細資訊。

下載軟體開發套件

若要在 Android 中下載 WebRTC 開發套件，請執行下列命令：

```
$ git clone https://github.com/awslabs/amazon-kinesis-video-streams-webrtc-sdk-android.git
```

建置 SDK

若要在 Android 中建置 WebRTC 開發套件，請完成以下步驟：

1. `amazon-kinesis-video-streams-webrtc-sdk-android/build.gradle` 使用 Open as Project 開啟，將 Android WebRTC SDK 匯入 Android Studio 整合開發環境 (IDE)。

2. 如果是第一次開啟專案，專案會自動同步。如果不是，請啟動同步。當您看到組建錯誤時，請選擇 Install missing SDK package(s) (安裝缺少的 SDK 套件)，然後選擇 Accept (接受) 並完成安裝，以安裝任何必要的 SDK 套件。
3. 進行 Amazon Cognito (使用者集區和身分集區) 設定。如需詳細步驟，請參閱為 SDK 設定 Amazon Cognito。這將產生建置 Android WebRTC 開發套件所需的身分驗證和授權設定。
4. 在您的 Android IDE 中，開啟 `awsconfiguration.json` (從 `src/main/res/raw/`)。檔案看起來如下：

```
{
  "Version": "1.0",
  "CredentialsProvider": {
    "CognitoIdentity": {
      "Default": {
        "PoolId": "REPLACE_ME",
        "Region": "REPLACE_ME"
      }
    }
  },
  "IdentityManager": {
    "Default": {}
  },
  "CognitoUserPool": {
    "Default": {
      "AppClientSecret": "REPLACE_ME",
      "AppClientId": "REPLACE_ME",
      "PoolId": "REPLACE_ME",
      "Region": "REPLACE_ME"
    }
  }
}
```

   以執行 為 SDK 設定 Amazon Cognito中的步驟所產生的值，更新 `awsconfiguration.json`。
5. 確保您的 Android 裝置已連線至執行 Android IDE 的電腦。在 Android IDE 中，選取連接的裝置，然後建置並執行 WebRTC Android 開發套件。

   此步驟會在您的 Android 裝置上安裝名為 `AWSKinesisVideoWebRTCDemoApp` 的應用程式。您可以使用此應用程式，驗證行動、Web 和 IoT 裝置用戶端之間的即時 WebRTC 音訊/視訊串流。

執行範例應用程式

請完成下列步驟：

1. 在您的 Android 裝置上，開啟 AWSKinesisVideoWebRTCDemoApp，並使用新的 (先建立) 或現有的 Amazon Cognito 帳戶登入。
2. 在 AWSKinesisVideoWebRTCDemoApp 中，瀏覽至 Channel Configuration (頻道組態) 頁面，然後建立新的訊號頻道，或選擇現有的訊號頻道。

   > **Note**
   >
   > 目前，使用此開發套件中的範例應用程式時，您在 AWSKinesisVideoWebRTCDemoApp 中只能執行一個訊號頻道。

3. 選擇性：如果您要以檢視器身分連線到此頻道，請選擇唯一的 Client Id (用戶端 ID)。只有在多個檢視器連線至頻道時，才需要用戶端 ID。這有助於頻道的主節點識別各自的檢視器。
4. 選擇 AWS 區域 ，以及您要傳送音訊或視訊資料，還是兩者。
5. 若要驗證點對點串流，請執行下列任何一項：

   > **Note**
   >
   > 請務必在此示範中使用的所有用戶端上，指定相同的訊號頻道名稱、 AWS 區域、檢視器 ID 和 AWS 帳戶 ID。

   - 兩個 Android 裝置之間的點對點串流：主節點和檢視器
     - 在兩個 Android 裝置上，使用上述程序下載、建置和執行 Android WebRTC 開發套件。
     - 在一個 Android 裝置上，以主節點模式開啟 AWSKinesisVideoWebRTCDemoApp，選擇 START MASTER (啟動主節點)，啟動新的工作階段 (訊號頻道)。

       > **Note**
       >
       > 目前，任何給定的訊號頻道只能有一個主節點。

     - 在您的第二個 Android 裝置上，以檢視器模式開啟 AWSKinesisVideoWebRTCDemoApp，以連線至上述步驟中啟動的訊號頻道 (工作階段)，選擇 START VIEWER (啟動檢視器)。

確認檢視器可以看到主節點的音訊/視訊資料。

- 嵌入式開發套件主節點和 Android 裝置檢視器之間的點對點串流
  - 在攝影機裝置上，以主節點模式下載、建置和執行 Amazon Kinesis Video Streams with WebRTC SDK in C for embedded device。
  - 在 Android 裝置上，使用上述程序下載、建置和執行 Android WebRTC 開發套件。在此 Android 裝置上，以檢視器模式開啟 AWSKinesisVideoWebRTCDemoApp，並確認檢視器可以看到嵌入式 SDK 主節點的音訊/視訊資料。
- Android 裝置 (主節點) 和 Web 瀏覽器 (檢視器) 之間的點對點串流
  - 在 Android 裝置上，使用上述程序下載、建置和執行 Android WebRTC 開發套件。在此 Android 裝置上，以主節點模式開啟 AWSKinesisVideoWebRTCDemoApp，選擇 START MASTER (啟動主節點)，啟動新的工作階段 (訊號頻道)。
  - 以檢視器身分下載、建置和執行 適用於 Web 應用程式的 JavaScript 中的 Amazon Kinesis Video Streams with WebRTC SDK WebRTC，並確認檢視器可以看到 Android 主節點的音訊/視訊。

為 SDK 設定 Amazon Cognito

先決條件

- 建議使用 Android Studio 檢查、編輯和執行應用程式的程式碼。建議使用最新的穩定版本。
- 在範本程式碼中，您會提供 Amazon Cognito 登入資料。

請依照這些程序來設定 Amazon Cognito 使用者集區和身分集區。

設定使用者集區

設定使用者集區

1. 登入 Amazon Cognito 主控台並確認區域正確無誤。
2. 在左側導覽中，選擇使用者集區。
3. 在使用者集區區段中，選擇建立使用者集區。
4. 完成下列各節：
   a. 步驟 1：設定登入體驗 - 在 Cognito 使用者集區登入選項區段中，選取適當的選項。

選取下一步。

b. 步驟 2：設定安全需求 - 選取適當的選項。

選取下一步。

c. 步驟 3：設定註冊體驗 - 選取適當的選項。

選取下一步。

d. 步驟 4：設定訊息傳遞 - 選取適當的選項。

在 IAM 角色選取欄位中，選取現有角色或建立新角色。

選取下一步。

e. 步驟 5：整合您的應用程式 - 選取適當的選項。

在初始應用程式用戶端欄位中，選擇機密用戶端。

選取下一步。

f. 步驟 6：檢閱並建立 - 檢閱先前區段的選擇，然後選擇建立使用者集區。

5. 在使用者集區頁面上，選取您剛建立的集區。

   複製使用者集區 ID，並記下此 ID 以供稍後使用。在 awsconfiguration.json檔案中，這是 CognitoUserPool.Default.PoolId。

6. 選取應用程式整合索引標籤，然後移至頁面底部。

7. 在應用程式用戶端清單區段中，選擇您剛建立的應用程式用戶端名稱。

   複製用戶端 ID 並記下此 ID 以供稍後使用。在 awsconfiguration.json 檔案中，這是 CognitoUserPool.Default.AppClientId。

8. 顯示用戶端秘密，並記下此秘密以供稍後使用。在 awsconfiguration.json 檔案中，這是 CognitoUserPool.Default.AppClientSecret。

設定身分集區

設定身分集區

1. 登入 Amazon Cognito 主控台並確認區域正確無誤。
2. 在左側導覽中，選擇身分集區。

3. 選擇 建立身分池。
4. 設定身分集區。

   a. 步驟 1：設定身分集區信任 - 完成下列區段：

      - 使用者存取 - 選取已驗證的存取
      - 已驗證的身分來源 - 選取 Amazon Cognito 使用者集區

      選取下一步。

   b. 步驟 2：設定許可 - 在已驗證角色區段中，完成下列欄位：

      - IAM 角色 - 選取建立新的 IAM 角色
      - IAM 角色名稱 - 輸入名稱並將其記下以供後續步驟使用。

      選取下一步。

   c. 步驟 3：連接身分提供者 - 在使用者集區詳細資訊區段中，完成下列欄位：

      - 使用者集區 ID - 選取您先前建立的使用者集區。
      - 應用程式用戶端 ID - 選取您先前建立的應用程式用戶端 ID。

      選取下一步。

   d. 步驟 4：設定屬性 - 在身分集區名稱欄位中輸入名稱。

      選取下一步。

   e. 步驟 5：檢閱並建立 - 檢閱每個區段中的選擇，然後選取建立身分集區。

5. 在身分集區頁面上，選取新的身分集區。

   複製身分集區 ID 並記下此 ID 以供稍後使用。在 `awsconfiguration.json`檔案中，這是 `CredentialsProvider.CognitoIdentity.Default.PoolId`。

6. 更新 IAM 角色的許可。

   a. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。

   b. 在左側導覽中，選擇角色。

   c. 尋找並選取您在上面建立的角色。

> **Note**
>
> 如有需要，請使用搜尋列。

d. 選取連接的許可政策。

選擇 Edit (編輯)。

e. 選取 JSON 索引標籤，並以下列內容取代政策：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cognito-identity:*",
                "kinesisvideo:*"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

選取下一步。

f. 如果尚未選取，請選取此新版本設定為預設值旁的方塊。

選取儲存變更。

適用於 iOS 的 Amazon Kinesis Video Streams WebRTC 開發套件

下列逐步指示說明如何在 iOS 中下載、建置和執行 Kinesis Video Streams WebRTC 開發套件及其對應的範例。

下載軟體開發套件

若要在 iOS 中下載 WebRTC 開發套件，請執行下列命令：

```
$ git clone https://github.com/awslabs/amazon-kinesis-video-streams-webrtc-sdk-ios.git
```

建置 SDK

請完成下列步驟：

1. 在 iOS 電腦上，開啟 KinesisVideoWebRTCDemoApp.xcworkspace (路徑：amazon-kinesis-video-streams-webrtc-sdk-ios/Swift/AWSKinesisVideoWebRTCDemoApp.xcworkspace)，將 iOS WebRTC 開發套件匯入 XCode 整合開發環境 (IDE)。
2. 如果是第一次開啟專案，專案會自動建置。如果不是，請啟動建置。

   您可能會看到下列錯誤：

   ```
   error: The sandbox is not in sync with the Podfile.lock. Run 'pod install' or
    update your CocoaPods installation.
   ```

   如果您看到此錯誤，請執行下列動作：

   a. 從目前的工作目錄切換至 amazon-kinesis-video-streams-webrtc-sdk-ios/Swift，並在命令列中執行以下命令：

      ```
      pod cache clean --all
      pod install
      ```

   b. 從目前的工作目錄切換至 amazon-kinesis-video-streams-webrtc-sdk-ios，並在命令列中執行以下命令：

      ```
      $ git checkout Swift/Pods/AWSCore/AWSCore/Service/AWSService.m
      ```

   c. Build。
3. 進行 Amazon Cognito (使用者集區和身分集區) 設定。如需詳細步驟，請參閱為 SDK 設定 Amazon Cognito。這將產生建置 iOS WebRTC 開發套件所需的身分驗證和授權設定。
4. 在 IDE 中，開啟 awsconfiguration.json 檔案 (從 /Swift/KVSiOSApp)。檔案看起來如下：

   ```
   {
       "Version": "1.0",
       "CredentialsProvider": {
           "CognitoIdentity": {
   ```

```
            "Default": {
                "PoolId": "REPLACEME",
                "Region": "REPLACEME"
            }
        }
    },
    "IdentityManager": {
        "Default": {}
    },
    "CognitoUserPool": {
        "Default": {
            "AppClientSecret": "REPLACEME",
            "AppClientId": "REPLACEME",
            "PoolId": "REPLACEME",
            "Region": "REPLACEME"
        }
    }
}
```

以執行 為 SDK 設定 Amazon Cognito中的步驟所產生的值，更新 awsconfiguration.json。

5. 在 IDE 中，開啟 Constants.swift 檔案 (從 /Swift/KVSiOSApp)。檔案看起來如下：

```
import Foundation
import AWSCognitoIdentityProvider

let CognitoIdentityUserPoolRegion = AWSRegionType.USWest2
let CognitoIdentityUserPoolId = "REPLACEME"
let CognitoIdentityUserPoolAppClientId = "REPLACEME"
let CognitoIdentityUserPoolAppClientSecret = "REPLACEME"

let AWSCognitoUserPoolsSignInProviderKey = "UserPool"
let CognitoIdentityPoolID = "REPLACEME"

let AWSKinesisVideoEndpoint = "https://kinesisvideo.us-west-2.amazonaws.com"
let AWSKinesisVideoKey = "kinesisvideo"

let VideoProtocols =  ["WSS", "HTTPS"]

let ConnectAsMaster = "connect-as-master"
let ConnectAsViewer = "connect-as-viewer"

let MasterRole = "MASTER"
let ViewerRole = "VIEWER"
```

```
let ClientID = "ConsumerViewer"
```

以執行 為 SDK 設定 Amazon Cognito中的步驟所產生的值，更新 `Constants.swift`。

6. 確保您的 iOS 裝置已連線至執行 XCode 的 Mac 電腦。在 XCode 中，選取連接的裝置，然後建置並執行 WebRTC iOS 開發套件。

   此步驟會在您的 iOS 裝置上安裝名為 `AWSKinesisVideoWebRTCDemoApp` 的應用程式。您可以使用此應用程式，驗證行動、Web 和 IoT 裝置用戶端之間的即時 WebRTC 音訊/視訊串流。

執行範例應用程式

請完成下列步驟：

1. 在您的 iOS 裝置上，開啟 AWSKinesisVideoWebRTCDemoApp，並使用新的 (先建立) 或現有的 Amazon Cognito 帳戶登入。
2. 在 AWSKinesisVideoWebRTCDemoApp 中，瀏覽至 Channel Configuration (頻道組態) 頁面，然後建立新的訊號頻道，或選擇現有的訊號頻道。

   > **Note**
   >
   > 目前，使用此開發套件中的範例應用程式時，您在 AWSKinesisVideoWebRTCDemoApp 中只能執行一個訊號頻道。

3. (選擇性) 如果您要以檢視器身分連線到此頻道，請選擇唯一的 Client Id (用戶端 ID)。只有在多個檢視器連線至頻道時，才需要用戶端 ID。這有助於頻道的主節點識別各自的檢視器。
4. 選擇 AWS 區域 ，以及您要傳送音訊或視訊資料，還是兩者。
5. 若要驗證點對點串流，請執行下列任何一項：

   > **Note**
   >
   > 請務必在此示範中使用的所有用戶端上，指定相同的訊號頻道名稱、 AWS 區域、檢視器 ID 和 AWS 帳戶 ID。

   - 兩個 iOS 裝置之間的點對點串流：主節點和檢視器

- 在兩個 iOS 裝置上，使用上述程序下載、建置和執行 iOS WebRTC 開發套件。
- 在一個 iOS 裝置上，以主節點模式開啟 AWSKinesisVideoWebRTCDemoApp，選擇 START MASTER (啟動主節點)，啟動新的工作階段 (訊號頻道)。

  > **Note**
  >
  > 目前，任何給定的訊號頻道只能有一個主節點。

- 在您的第二個 iOS 裝置上，以檢視器模式開啟 AWSKinesisVideoWebRTCDemoApp，以連線至上述步驟中啟動的訊號頻道 (工作階段)，選擇 START VIEWER (啟動檢視器)。

  確認檢視器可以看到主節點的音訊/視訊資料。

- 嵌入式開發套件主節點和 iOS 裝置檢視器之間的點對點串流
  - 在攝影機裝置上，以主節點模式下載、建置和執行 Amazon Kinesis Video Streams with WebRTC SDK in C for embedded device。
  - 在 iOS 裝置上，使用上述程序下載、建置和執行 iOS WebRTC 開發套件。在此 iOS 裝置上，以檢視器模式開啟 AWSKinesisVideoWebRTCDemoApp，並確認 iOS 檢視器可以看到嵌入式 SDK 主節點的音訊/視訊資料。
- iOS 裝置 (主節點) 和 Web 瀏覽器 (檢視器) 之間的點對點串流
  - 在 iOS 裝置上，使用上述程序下載、建置和執行 iOS WebRTC 開發套件。在此 iOS 裝置上，以主節點模式開啟 AWSKinesisVideoWebRTCDemoApp，選擇 START MASTER (啟動主節點)，啟動新的工作階段 (訊號頻道)。
  - 以檢視器身分下載、建置和執行 適用於 Web 應用程式的 JavaScript 中的 Amazon Kinesis Video Streams with WebRTC SDK WebRTC，並確認 JavaScript 檢視器可以看到 Android 主節點的音訊/視訊。

為 SDK 設定 Amazon Cognito

先決條件

- 我們建議您使用 XCode 來檢查、編輯和執行應用程式碼。我們建議使用最新版本。
- 在範本程式碼中，您會提供 Amazon Cognito 登入資料。

請依照這些程序來設定 Amazon Cognito 使用者集區和身分集區。

設定使用者集區

設定使用者集區

1. 登入 Amazon Cognito 主控台並確認區域正確無誤。
2. 在左側導覽中，選擇使用者集區。
3. 在使用者集區區段中，選擇建立使用者集區。
4. 完成下列各節：

   a. 步驟 1：設定登入體驗 - 在 Cognito 使用者集區登入選項區段中，選取適當的選項。

      選取下一步。

   b. 步驟 2：設定安全需求 - 選取適當的選項。

      選取下一步。

   c. 步驟 3：設定註冊體驗 - 選取適當的選項。

      選取下一步。

   d. 步驟 4：設定訊息傳遞 - 選取適當的選項。

      在 IAM 角色選取欄位中，選取現有角色或建立新角色。

      選取下一步。

   e. 步驟 5：整合您的應用程式 - 選取適當的選項。

      在初始應用程式用戶端欄位中，選擇機密用戶端。

      選取下一步。

   f. 步驟 6：檢閱並建立 - 檢閱先前區段的選擇，然後選擇建立使用者集區。
5. 在使用者集區頁面上，選取您剛建立的集區。

   複製使用者集區 ID，並記下此 ID 以供稍後使用。在 `awsconfiguration.json` 檔案中，這是 `CognitoUserPool.Default.PoolId`。
6. 選取應用程式整合索引標籤，然後移至頁面底部。
7. 在應用程式用戶端清單區段中，選擇您剛建立的應用程式用戶端名稱。

   複製用戶端 ID 並記下此 ID 以供稍後使用。在 `awsconfiguration.json` 檔案中，這是 `CognitoUserPool.Default.AppClientId`。

8. 顯示用戶端秘密，並記下此秘密以供稍後使用。在 awsconfiguration.json檔案中，這是 CognitoUserPool.Default.AppClientSecret。

設定身分集區

設定身分集區

1. 登入 Amazon Cognito 主控台並確認區域正確無誤。
2. 在左側導覽中，選擇身分集區。
3. 選擇 建立身分池。
4. 設定身分集區。
   a. 步驟 1：設定身分集區信任 - 完成下列區段：
      - 使用者存取 - 選取已驗證的存取
      - 已驗證的身分來源 - 選取 Amazon Cognito 使用者集區

      選取下一步。
   b. 步驟 2：設定許可 - 在已驗證角色區段中，完成下列欄位：
      - IAM 角色 - 選取建立新的 IAM 角色
      - IAM 角色名稱 - 輸入名稱並將其記下以供後續步驟使用。

      選取下一步。
   c. 步驟 3：連接身分提供者 - 在使用者集區詳細資訊區段中，完成下列欄位：
      - 使用者集區 ID - 選取您先前建立的使用者集區。
      - 應用程式用戶端 ID - 選取您先前建立的應用程式用戶端 ID。

      選取下一步。
   d. 步驟 4：設定屬性 - 在身分集區名稱欄位中輸入名稱。

      選取下一步。
   e. 步驟 5：檢閱並建立 - 檢閱每個區段中的選擇，然後選取建立身分集區。
5. 在身分集區頁面上，選取新的身分集區。

複製身分集區 ID，並記下此 ID 以供稍後使用。在 awsconfiguration.json檔案中，這是 CredentialsProvider.CognitoIdentity.Default.PoolId。

6. 更新 IAM 角色的許可。

   a. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/iam/ 的 IAM 主控台。

   b. 在左側導覽中，選擇角色。

   c. 尋找並選取您在上面建立的角色。

   > **Note**
   >
   > 如有需要，請使用搜尋列。

   d. 選取連接的許可政策。

      選擇 Edit (編輯)。

   e. 選取 JSON 索引標籤，並以下列內容取代政策：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cognito-identity:*",
                "kinesisvideo:*"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

      選取下一步。

   f. 如果尚未選取，請選取此新版本設定為預設值旁的方塊。

      選取儲存變更。

C SDK 的用戶端指標

使用 Amazon Kinesis Video Streams with WebRTC 建置的應用程式由各種移動組件組成，包括聯網、訊號、候選者交換、對等連線和資料交換。C 中的 Kinesis Video Streams with WebRTC 支援各種用戶端指標，可讓您監控和追蹤應用程式中這些元件的效能和用量。支援的指標分為兩個主要類別：專門為 Kinesis Video Streams 的訊號和聯網實作定義的自訂指標，以及衍生自 W3C 標準的媒體和資料相關通訊協定特定指標。請注意，目前 C 中的 Kinesis Video Streams with WebRTC 僅支援一部分的 W3C 標準指標。 WebRTC

主題

訊號指標

訊號指標可用來了解訊號用戶端在應用程式執行時的行為。您可以使用 `STATUS signalingClientGetMetrics (SIGNALING_CLIENT_HANDLE, PSignalingClientMetrics)` API 來取得這些訊號指標。以下是使用模式範例：

```
SIGNALING_CLIENT_HANDLE signalingClientHandle;
SignalingClientMetrics signalingClientMetrics;
STATUS retStatus = signalingClientGetMetrics(signalingClientHandle,
 &signalingClientMetrics);
printf("Signaling client connection duration: %" PRIu64 " ms",
       (signalingClientMetrics.signalingClientStats.connectionDuration /
 HUNDREDS_OF_NANOS_IN_A_MILLISECOND));
```

您可以在 Stats.h `signalingClientStats`中找到 的定義。

目前支援下列訊號指標：

指標	Description	
cpApiCallLatency	計算控制平面 API 呼叫的延遲。使用指數移動平均 (EMA) 完成計算。相關聯的呼叫包括： describeChannel、cr eateChannel、getCha	

指標	Description	
	nnelEndpoint 和 deleteCha nnel。	
dpApiCallLatency	計算資料平面 API 呼叫的延遲。使用指數移動平均 (EMA) 完成計算。相關聯的呼叫包括：getIceConfig。	
signalingClientUptime	這表示用戶端物件存在的時間。每次叫用此指標時，都會發出最新的執行時間值。	
connectionDuration	如果建立連線，這會發出連線存活的持續時間。否則，會發出 0 的值。這與發出用戶端正常運作時間不同，因為連線會不斷出現，但 signaling ClientUptime 表示用戶端物件本身。	
numberOfMessagesSent	此值會在對等傳送優惠、回答或 ICE 候選項目時更新。	
numberOfMessagesReceived	與 numberOfMessagesSent 不同，此指標會針對任何類型的訊號訊息進行更新。訊號訊息的類型可在 中使用`SIGNALING_MESSAGE_TYPE` 。	

指標	Description
iceRefreshCount	這會在叫用 getIceConfig 時遞增。調用此 的速率是以 TTL 為基礎，做為所收到 ICE 組態的一部分。每次收到一組新的 ICE 組態時，都會將計時器設定為下次重新整理，因為組態中的登入資料有效性減去一些寬限期。
numberOfErrors	計數器用於追蹤訊號用戶端內產生的錯誤數目。取得 ICE 組態、取得訊號狀態、追蹤訊號指標、傳送訊號訊息，以及將訊號用戶端連線至 Web 通訊端以追蹤傳送/接收訊息時所產生的錯誤。
numberOfRuntimeErrors	指標包含訊號用戶端核心執行時所發生的錯誤。這裡會追蹤重新連線失敗、訊息接收失敗和 ICE 組態重新整理錯誤等案例。
numberOfReconnects	指標會在每次重新連線時遞增。這是了解 設定中網路連線穩定性的實用指標。

C SDK 支援的 W3C 標準指標

使用 W3CWebRTC 標準指標的子集。這些分為下列類別：

- 網路：
  - Ice Candidate：這些指標提供有關所選本機和遠端候選項目的資訊，以便在對等之間進行資料交換。這包括候選伺服器的伺服器來源、IP 地址、為通訊選取的候選類型，以及候選優先順序。這些指標可作為快照報告。

  - Ice 伺服器：這些指標用於收集支援的不同 ICE 伺服器的操作資訊。這在嘗試了解主要用於通訊和連線檢查的伺服器時非常有用。在某些情況下，如果收集候選項目失敗，檢查這些指標也很有用。
  - Ice Candidate Pair：這些指標用於了解在對等和時間相關測量之間交換的位元組/封包數量。
- 媒體和資料：
  - 遠端傳入 RTP：這些指標代表寄件者傳送之資料串流的端點觀點。
  - 傳出 RTP：這些指標提供傳出 RTP 串流的相關資訊。在分析冗長的串流或串流停止時，它們也非常有用。
  - 傳入 RTP：這些指標提供傳入媒體的相關資訊。
  - 資料管道指標：這些指標可協助您分析透過資料管道傳送和接收的訊息和位元組數。您可以使用頻道 ID 提取指標。

您可以使用 STATUS rtcPeerConnectionGetMetrics (PRtcPeerConnection, PRtcRtpTransceiver, PRtcStats) API 來收集與 ICE、RTP 和資料通道相關的指標。以下是使用範例：

```
RtcStats rtcStats;
rtcStats.requestedTypeOfStats = RTC_STATS_TYPE_LOCAL_CANDIDATE;
STATUS retStatus = rtcPeerConnectionGetMetrics (pRtcPeerConnection, NULL, &rtcStats);
printf("Local Candidate address: %s\n",
 rtcStats.rtcStatsObject.localIceCandidateStats.address);
```

以下是另一個顯示使用模式以取得收發器相關統計資料的範例：

```
RtcStats rtcStats;
PRtcRtpTransceiver pVideoRtcRtpTransceiver;
rtcStats.requestedTypeOfStats = RTC_STATS_TYPE_OUTBOUND_RTP;
STATUS retStatus = rtcPeerConnectionGetMetrics (pRtcPeerConnection,
 pVideoRtcRtpTransceiver, &rtcStats);
printf("Number of packets discarded on send: %s\n",
 rtcStats.rtcStatsObject.outboundRtpStreamStats.packetsDiscardedOnSend);
```

在上述範例中，如果 rtcPeerConnectionGetMetrics() 的第二個引數是 NULL，則會傳回清單中第一個收發器的資料。

rtcStatsObject 的定義可在 Stats.h 中找到。RtcStats 的定義可在 Include.h 中找到。

您可以在 WebRTC C SDK 儲存庫的範例目錄中，以及 Kinesis Video Streams 示範儲存庫中找到 APIs 的範例用量和不同的指標。

使用 W3CC 標準指標。 WebRTC

主題

- 聯網
- 媒體
- 資料管道

聯網

ICE 伺服器指標：

指標	Description
URL	要追蹤之 STUN/TURN 伺服器的 URL
連接埠	用戶端使用的連接埠號碼
通訊協定	從 ICE 伺服器 URI 擷取的傳輸通訊協定。如果值為 UDP，ICE 會透過 UDP 嘗試 TURN，否則 ICE 會透過 TCP/TLS 嘗試 TURN。如果 URI 不包含傳輸，ICE 會透過 UDP 和 TCP/TLS 嘗試 TURN。如果是 STUN 伺服器，此欄位為空白。
傳送的請求總數	值會針對每個 srflx 候選請求更新，同時從輪換候選項目傳送繫結請求。
收到的回應總數	每次收到 STUN 繫結回應時，都會更新此值。
往返時間總計	每次收到請求的同等回應時，此值都會更新。系統會使用檢

指標	Description	
	查總和做為金鑰，在雜湊映射中追蹤請求封包。	

ICE 候選統計資料：僅包含所選候選項目 （本機和遠端） 的相關資訊。

指標	Description	
address	這表示本機和遠端候選項目的 IP 地址。	
port	候選的連接埠號碼	
protocol	用來取得候選項目的通訊協定。有效值為 UDP/TCP。	
candidateType	選取的候選類型 - 主機、srflx 或轉送。	
priority	所選本機和遠端候選項目的優先順序。	
url	所選本機候選項目的來源。這會顯示所選候選項目是否從 STUN 伺服器或 TURN 伺服器接收。	
relayProtocol	如果使用 TURN 伺服器來取得選取的本機候選項目，此欄位會指出使用哪個通訊協定來取得它。有效值為 TCP/UDP。	

ICE 候選配對統計資料：僅包含所選候選配對的相關資訊。

指標	Description	
localCandidateId	配對中所選本機候選項目的 ID。	
remoteCandidateId	配對中所選遠端候選項目的 ID。	
state	正在檢查的候選配對狀態。	
已指定	設定為 TRUE，因為正在為選取的候選配對提取統計資料。	
packetsSent	傳送的封包數量。這是在 . 呼叫中計算的`writeFrame` 。此資訊也可以從傳出的 RTP 統計資料中擷取，但由於 Ice 候選配對包含 lastPacketSent 時間戳記，因此計算兩個時間點之間傳送的封包數量可能很有用。	
packetsReceived	每次呼叫 incomingDataHandler 時都會更新。	
bytesSent	這是在 `writeFrame()` 呼叫的 `iceAgentSendPacket()` 中計算的。這在計算位元速率時非常有用。目前，這也包含 標頭和填補，因為 ICE 層對 RTP 封包格式來說是絕對的。	
bytesReceived	每次呼叫 incomingDataHandler 時都會更新。目前，這也包含 標頭和填補，因為 ICE 層對 RTP 封包格式來說是絕對的。	

指標	Description	
lastPacketSentTimestamp	每次傳送封包時都會更新。這可以與 packetsSent 和應用程式中記錄的開始時間搭配使用，以達到目前的封包傳輸速率。	
lastPacketReceived Timestamp	這會在 中接收資料時更新`incomingDataHandler()` 。這可與 packetsReceived 搭配使用，以推斷目前的封包接收速率。開始時間必須記錄在回`transceiverOnFrame()` 呼中的應用程式層。	
firstRequestTimestamp	在 中成功傳送第一個 STUN 繫結請求時記錄`iceAgentSendStunPacket()` 。這可以與 lastRequestTimestamp 和 requestsSent 搭配使用，以尋找 STUN 繫結請求之間的平均時間。	
lastRequestTimestamp	每次成功在 中傳送 STUN 繫結請求時都會記錄`iceAgentSendStunPacket()` 。	
lastResponseTimestamp	每次收到 STUN 繫結回應時都會記錄。	
totalRoundTripTime	在收到請求的繫結回應時更新。請求和回應會根據檢查總和映射到雜湊表中。	

指標	Description
currentRoundTripTime	在收到候選配對請求的繫結回應時，最新的往返時間會更新。
requestsReceived	在每個收到的 STUN 繫結請求上更新的計數器。
requestsSent	在 中傳送的每個 STUN 繫結請求上更新的計數器`iceAgentSendStunPacket()` 。
responsesSent	在傳送的每個 STUN 繫結回應上更新的計數器，以回應中的繫結請求`handleStunPacket()` 。
responsesReceived	在 中收到的每個 STUN 繫結回應上更新的計數器`handleStunPacket()` 。
packetsDiscardedOnSend	當封包傳送失敗時更新。換句話說，這會在`iceUtilsSendData()` 失敗時更新。這有助於判斷在特定持續時間內捨棄的封包百分比。
bytesDiscardedOnSend	當封包傳送失敗時更新。換句話說，這會在`iceUtilsSendData()` 失敗時更新。這在判斷在特定持續時間內捨棄的封包百分比時非常有用。請注意，計數器也包含封包的標頭。

媒體

傳出 RTP 統計資料

指標	Description	
voiceActivityFlag	這目前是 Include.h 中`RtcEncoderStats` 定義的一部分。如果最後一個音訊封包包含語音，則旗標設定為 TRUE。目前未在範例中設定旗標。	
packetsSent	這表示針對所選 SSRC 送出的 RTP 封包總數。這是 https://www.w3.org/TR/webrtc-stats/#sentrtpstats-dict* 的一部分，包含在傳出統計資料中。每次呼叫 writeFrame() 時都會遞增。	
bytesSent	已傳送的位元組總數，不包括 RTP 標頭和填補。這會在每次 writeFrame 呼叫時更新。	
encoderImplementation	這是由應用程式層更新為 RtcEncoderStats 物件的一部分。	
packetsDiscardedOnSend	如果 ICE 代理程式因任何原因無法傳送加密的 RTP 封包，此欄位會更新`iceAgentSendPacket` 。	
bytesDiscardedOnSend	如果 ICE 代理程式因任何原因無法傳送加密的 RTP 封包，此欄位也會更新`iceAgentSendPacket` 。	

指標	Description	
framesSent	只有在媒體串流堆疊類型為 MEDIA_STREAM_TRACK _KIND_VIDEO 時，才會遞增。	
hugeFramesSent	此計數器會針對影格的平均大小 2.5 倍的影格進行更新。透過計算 fps （根據上次已知的影格計數時間和以時間間隔編碼的影格數量） 並使用應用程式設定的 RtcEncoderStats 中的 targetBitrate 來取得影格的大小。	
framesEncoded	只有在成功編碼影格後，才會針對影片軌跡更新此計數器。它會在每次 writeFrame 呼叫時更新。	
keyFramesEncoded	只有在金鑰影格成功編碼後，才會針對視訊軌更新此計數器。它會在每次 writeFrame 呼叫時更新。	
framesDiscardedOnSend	當影格傳送因`iceAgentS endPacket` 呼叫失敗而失敗時，會更新此項目。框架由一組封包組成，目前，如果傳送時因錯誤而捨棄任何封包，framesDiscardedOnSend 會失敗。	

指標	Description	
frameWidth	這理想地代表最後一個編碼影格的影格寬度。目前，應用程式將此值設定為 RtcEncoderStats* * 的一部分，而且意義不大。	
frameHeight	這理想地代表最後一個編碼影格的影格高度。目前，應用程式將此值設定為 RtcEncoderStats 的一部分，意義不大。	
frameBitDepth	這表示最後一個編碼影格的每個像素寬度的位元深度。目前，這是由應用程式設定為 RtcEncoderStats 的一部分，並翻譯為傳出統計資料。	
nackCount	每次在 RTP 封包上收到 NACK 並再次嘗試傳送封包時，都會更新此值。堆疊支援在接收 NACK 時重新傳輸封包。	
firCount	收到 FIR 封包 (onRtcpPacket->onRtcpFIRPacket) 時會更新此值。它會指出串流落後且必須略過影格才能趕上進度的頻率。FIR 封包目前未解碼以擷取欄位，因此即使已設定計數，也不會採取任何動作。	
pliCount	收到 PLI 封包 (onRtcpPacket->onRtcpPLIPacket) 時會更新此值。這表示一或多個影格已遺失部分編碼的影片資料量。	

指標	Description	
sliCount	收到 SLI 封包 (onRtcpPacket->onRtcpSLIPacket) 時會更新此值。它指出封包遺失影響單一影格的頻率。	
qualityLimitationResolutionChanges	目前，堆疊支援此指標，但不會監控每個編碼影格的影格寬度和高度。	
lastPacketSentTimestamp	傳送最後一個封包的時間戳記。它會在每次 writeFrame 呼叫時更新。	
headerBytesSent	為此 SSRC 傳送的 RTP 標頭和填補位元組總數，不包括實際 RTP 承載。	
bytesDiscardedOnSend	當因 iceAgentSendPacket 呼叫失敗而導致框架傳送失敗時，就會更新此項目。框架由一組封包組成，而封包則由位元組組成，目前，如果傳送時因錯誤而捨棄任何封包，則 bytesDiscardedOnSend 會失敗。	
retransmittedPacketsSent	接收 PLI/SLI/NACK 時重新傳輸的封包數量。目前，堆疊只會計算 NACK 的重新傳送封包，因為不支援 PLI 和 SLI 型重新傳輸。	

指標	Description	
retransmittedBytesSent	接收 PLI/SLI/NACK 時重新傳輸的位元組數。目前，堆疊只會計算 NACK 的重新傳送位元組，因為不支援 PLI 和 SLI 型重新傳輸。	
targetBitrate	這是在應用程式層級中設定。	
totalEncodedBytesTarget	每次對影格進行編碼時，目標影格大小都會增加，以位元組為單位。這是使用框架結構中的大小參數進行更新。	
framesPerSecond	這是根據上次已知編碼影格所記錄的時間，以及在一秒內傳送的影格數來計算。	
totalEncodeTime	這在應用程式中設定為任意值，並在內部轉譯為傳出統計資料。	
totalPacketSendDelay	目前將此設為 0，因為 iceAgentSendPacket 會立即傳送封包。	

遠端傳入 RTP 統計資料：

指標	Description	
roundTripTime	接收 RTCP 封包類型 201（接收者報告）時，會從 RTCP 接收者報告中擷取此值。報告包含上次傳送者報告和自上次傳送者報告後延遲等詳細資訊，以計算往返時間。寄件者報告	

指標	Description	
	大約每 200 毫秒產生一次，其中包含傳送的封包數和從傳出統計資料擷取的位元組數等資訊。	
totalRoundTripTime	計算出的往返時間總和	
fractionLost	代表自傳送上一個寄件者/接收者 reporfractionLost 以來，SSRC 遺失的 RTP 封包部分。	
reportsReceived	每次收到接收者報告類型封包時更新。	
roundTripTimeMeasurements	指出針對包含有效往返時間的 SSRC 收到的報告總數。不過，目前此值會遞增，因此其意義與 reportsReceived 相同。	

傳入 RTP 統計資料：

指標	Description	
packetsReceived	收到特定 SSRC 的封包時，計數器會更新。	
抖動	此指標表示特定 SSRC 以秒為單位測量的封包抖動。	
jitterBufferDelay	此指標表示抖動緩衝區中每個封包所花費的時間總和。	
jitterBufferEmittedCount	從抖動緩衝區流出的音訊範例或影片影格總數。	

指標	Description	
packetsDiscarded	當抖動緩衝區已滿且封包無法推送至其中時，計數器會更新。這可用於計算在固定持續時間內捨棄的封包百分比。	
framesDropped	此值會在叫用 `onFrameDroppedFunc()` 時更新。	
lastPacketReceivedTimestamp	代表此 SSRC 收到最後一個封包的時間戳記。	
headerBytesReceived	計數器會在接收 RTP 封包時更新。	
bytesReceived	收到的位元組數。這不包含標頭位元組。此指標可用來計算傳入位元速率。	
packetsFailedDecryption	當 SRTP 封包的解密失敗時，這會遞增。	

資料管道

資料管道指標：

指標	Description	
label	Label 是要檢查的資料通道名稱。	
protocol	由於我們的堆疊使用 SCTP，因此通訊協定會設定為常數 SCTP。	
dataChannelIdentifier	用於唯一識別資料通道的偶數或奇數識別符。如果 SDK 是提	

指標	Description	
	供者，即使 SDK 是接聽者，此值也會更新為奇數值。	
state	查詢統計資料時資料通道的狀態。目前支援的兩個狀態為 RTC_DATA_CHANNEL_STATE_CONNECTING（建立頻道時）和 RTC_DATA_CHANNEL_STATE_OPEN（在 onOpen() 事件中設定）。	
messagesSent	當 SDK 透過資料通道傳送訊息時，計數器會更新。	
bytesSent	計數器會以送出的訊息中的位元組進行更新。這可用來了解有多少位元組不會因為失敗而傳送，也就是了解傳送的位元組百分比。	
messagesReceived	指標會在回`onMessage()`呼中遞增。	
bytesReceived	指標會在回`onMessage()`呼中產生。	

搭配 WebRTC 使用 Amazon Kinesis Video Streams 來擷取和存放媒體

Amazon Kinesis Video Streams 提供透過 WebRTC 即時將視訊和音訊串流至雲端的功能，以進行儲存、播放和分析處理。客戶可以使用增強型 WebRTC SDK 和雲端 APIs 來啟用即時串流，以及將媒體擷取至雲端。

若要開始使用，您可以在具有視訊感應器的任何安全攝影機或裝置上安裝 Amazon Kinesis Video Streams with WebRTC SDK，並使用我們的 APIs 來啟用延遲低於 1 秒的媒體串流，以及在雲端中擷取和儲存。 AWS IoT 擷取後，您可以透過我們easy-to-use存取資料。 APIs Amazon Kinesis Video Streams 可讓您播放影片以進行即時和隨需檢視，並透過與 Amazon Rekognition Video 和 SageMaker AI 整合，快速建置利用電腦視覺和影片分析的應用程式。

主題

API 操作

使用下列 API 操作來設定 Amazon Kinesis Video Streams WebRTC 擷取：

- DescribeMappedResourceConfiguration
- DescribeMediaStorageConfiguration
- JoinStorageSession
- JoinStorageSessionAsViewer

- UpdateMediaStorageConfiguration

什麼是 Amazon Kinesis Video Streams with WebRTC 擷取和儲存？

Amazon Kinesis Video Streams 提供透過 WebRTC 即時將視訊和音訊串流至雲端的功能，以進行儲存、播放和分析處理。本主題將提供step-by-step說明，以設定和使用我們的 WebRTC 開發套件和雲端 APIs，以啟用即時串流和媒體擷取至雲端。這些指示包括使用 AWS Command Line Interface 和 Kinesis Video Streams 主控台的指引。

首次將 Amazon Kinesis Video Streams 與 WebRTC 搭配使用之前，請參閱 the section called “設定 AWS 帳戶”。

了解 WebRTC 擷取和儲存

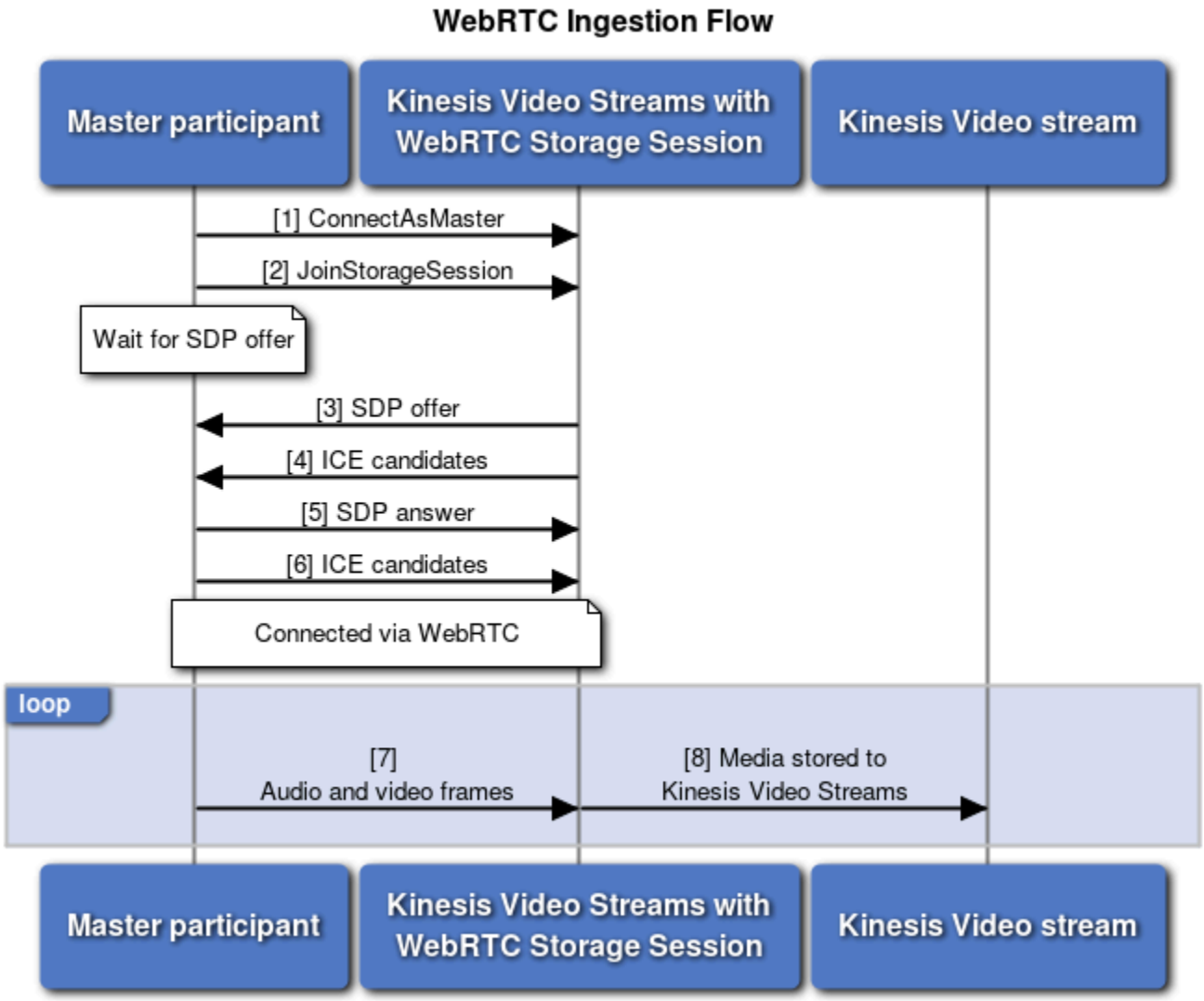
下列各節說明 Kinesis Video Streams with WebRTC 中可用的不同擷取和儲存選項。

主題

- 僅限主要參與者
- 主要參與者和瀏覽者參與者一起

僅限主要參與者

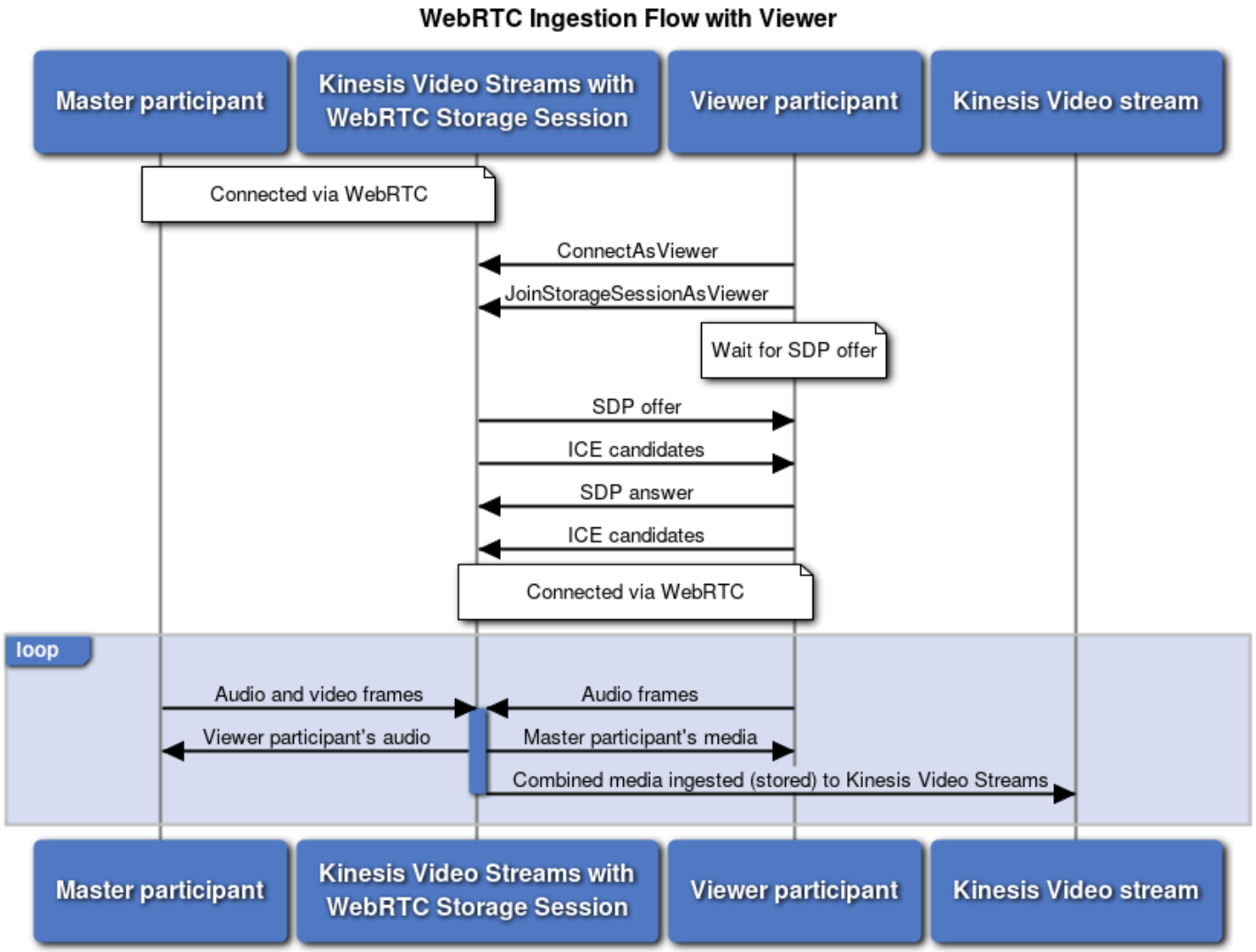
主要參與者會先透過 使用 WebRTC 訊號連線至 Kinesis Video Streamsthe section called “ConnectAsMaster”。接著，他們呼叫 JoinStorageSession API，讓儲存工作階段啟動 WebRTC 連線。建立 WebRTC 連線後，會將媒體擷取至設定的 Kinesis 影片串流。



主要參與者和瀏覽者參與者一起

觀眾參與者首先透過 使用 WebRTC 訊號連接到 Kinesis Video Streamsthe section called "ConnectAsViewer"。接著，他們呼叫 JoinStorageSessionAsViewer API，讓儲存工作階段啟動 WebRTC 連線。建立 WebRTC 連線後，只要主要參與者存在，來自主要參與者和所有檢視器參與者的合併媒體就會擷取至設定的 Kinesis 影片串流。

儲存工作階段會合併所有檢視器參與者的音訊，並將其轉送給主要參與者。檢視器參與者會從主要參與者接收合併媒體，並從儲存工作階段接收任何其他檢視器參與者的音訊。



使用儲存工作階段建立 WebRTC 連線

由於儲存工作階段位於 Amazon 網路內，因此儲存工作階段只會將 `relay`(TURN) 候選項目傳送給參與者。如果參與者的網路允許，則 `srflx`(STUN) 候選項目可用來連線至儲存工作階段。換言之，從參與者的角度來看，本機指定的 ICE 候選者可以是 `srflx`或 `relay`，而遠端 ICE 候選者一律是 `relay`。

若要最佳化連線時間，請勿將host候選項目傳送至儲存工作階段。儲存工作階段也Trickle ICE需要使用。

請參閱 the section called “對儲存工作階段連線進行故障診斷”以疑難排解儲存工作階段的連線問題。

建立訊號頻道

Kinesis Video Streams with WebRTC 訊號頻道有助於交換在 WebRTC 用戶端之間建立和維護peer-to-peer連線所需的訊號訊息。它處理工作階段描述協定 (SDP) 提供和回應工作階段參數的交涉，以及交換互動式連線建立 (ICE) 候選項目以取得網路資訊。

若要建立訊號頻道，請呼叫 CreateSignalingChannel API。此頁面將示範如何使用 AWS 管理主控台 AWS CLI和其中一個 AWS SDKs 叫用該 API。

> **Important**
>
> 請記下頻道 ARN，您稍後會需要它。

AWS 管理主控台

請執行下列操作：

1. 在 https：//https://console.aws.amazon.com/kinesisvideo/home/#/signalingChannels 開啟 Kinesis Video Streams Signaling Channels 主控台。
2. 選擇 Create signaling channel (建立訊號頻道)。
3. 在建立新的訊號頻道頁面上，輸入訊號頻道的名稱。

   將預設Time-to-live(Ttl) 值保留為 60 秒。

   選擇 Create signaling channel (建立訊號頻道)。
4. 建立訊號頻道後，請檢閱頻道詳細資訊頁面上的詳細資訊。

AWS CLI

確認您已安裝 AWS CLI 並設定 。如需詳細資訊，請參閱「AWS Command Line Interface 使用者指南」。

如需安裝說明，請參閱AWS Command Line Interface 《 使用者指南》。安裝後，使用登入資料和區域設定 AWS CLI。

或者，開啟已安裝並設定 AWS CLI 的 AWS CloudShell 終端機。如需詳細資訊，請參閱 AWS CloudShell 使用者指南。

使用 執行下列 Create-Signaling-Channel 命令 AWS CLI：

```
aws kinesisvideo create-signaling-channel \
  --channel-name "YourChannelName" \
  --region "us-west-2"
```

回應如下所示：

```
{
    "ChannelARN": "arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123"
}
```

AWS SDK

此程式碼片段說明如何使用適用於 JavaScript 的 AWS SDK v2 建立 Kinesis Video Streams with WebRTC 訊號頻道。語法將與其他 AWS SDKs不同，但一般流程將相同。在 GitHub 上檢視完整的程式碼範例。

建立 Kinesis Video Streams 用戶端。這是用來呼叫 `CreateSignalingChannel` API 的用戶端。

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

使用 用戶端呼叫 `CreateSignalingChannel` API。

```
const createSignalingChannelResponse = await kinesisVideoClient
    .createSignalingChannel({
        ChannelName: 'YourChannelName',
    })
    .promise();
```

列印回應。

```
console.log(createSignalingChannelResponse.ChannelARN);
```

具有此程式碼範例的即時網頁可在 GitHub 上使用。輸入您的區域、 AWS 憑證和訊號頻道的名稱。

選取建立頻道。

建立影片串流

請依照這些程序建立將擷取媒體的串流。如果您已建立目的地串流，請略過此步驟。

> **Important**
>
> WebRTC 擷取需要資料保留大於 0 的 Kinesis 影片串流。最短為 1 小時。

若要建立串流，請使用 AWS 管理主控台 AWS CLI、 或其中一個 AWS SDK 呼叫 CreateStream API。

> **Important**
>
> 請記下串流 ARN，稍後會需要它。

AWS 管理主控台

請執行下列操作：

1. 開啟位於 https：//https://console.aws.amazon.com/kinesisvideo/home/ 的 Kinesis Video Streams 主控台。
2. 在 Video streams (影片串流) 頁面上，選擇 Create video stream (建立影片串流).
3. 在建立新的影片串流頁面上，輸入 *YourStreamName* 做為串流名稱。保持選取預設組態按鈕。

   這會建立資料保留大於 0 的串流。

   選擇 Create video stream (建立影片串流。
4. Kinesis Video Streams 建立串流後，請檢閱 *YourStreamName* 頁面上的詳細資訊。

AWS CLI

確認您已安裝 AWS CLI 並設定 。如需詳細資訊，請參閱「AWS Command Line Interface 使用者指南」。

如需安裝說明，請參閱AWS Command Line Interface 《 使用者指南》。安裝後，使用登入資料和區域設定 AWS CLI。

或者，開啟已安裝並設定 AWS CLI 的 AWS CloudShell 終端機。如需詳細資訊，請參閱 AWS CloudShell 使用者指南。

使用 執行下列Create-Stream命令 AWS CLI：

```
aws kinesisvideo create-stream \
  --stream-name "YourStreamName" \
  --data-retention-in-hours 24 \
  --region "us-west-2"
```

回應如下所示：

```
{
    "StreamARN": "arn:aws:kinesisvideo:us-
west-2:123456789012:stream/YourStreamName/1234567890123"
}
```

AWS SDK

此程式碼片段說明如何使用 AWS 適用於 JavaScript 的 SDK v2 建立 Kinesis 影片串流。語法將與其他 AWS SDKs不同，但一般流程將相同。在 GitHub 上檢視完整的程式碼範例。

建立 Kinesis Video Streams 用戶端。這是用來呼叫 CreateStream API 的用戶端。

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

使用 用戶端呼叫 CreateStream API。

```
const createStreamResponse = await kinesisVideoClient
    .createStream({
        StreamName: 'YourStreamName',
        DataRetentionInHours: 48,
    })
    .promise();
```

列印回應。

```
console.log(createStreamResponse.StreamARN);
```

具有此程式碼範例的即時網頁可在 GitHub 上使用。輸入您的區域、 AWS 憑證和訊號頻道的名稱。

展開 WebRTC 擷取和儲存節點，輸入串流的名稱，然後選擇建立串流。彈出式視窗會詢問您要保留串流資料的時數。輸入大於 0 的值，然後選擇建立串流。

授予許可

您必須將串流許可授予 IAM 角色，才能透過 WebRTC 在 Amazon Kinesis Video Streams 中擷取串流。

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

主要和檢視器角色也必須具有 `DescribeStream`、 `GetDataEndpoint`和 `PutMedia`許可，才能將媒體擷取至 Kinesis Video Streams。

請參閱以下適用於主要參與者的範例 IAM 政策：

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kinesisvideo:DescribeSignalingChannel",
        "kinesisvideo:DescribeMediaStorageConfiguration",
        "kinesisvideo:GetSignalingChannelEndpoint",
        "kinesisvideo:GetIceServerConfig",
        "kinesisvideo:ConnectAsMaster",
        "kinesisvideo:JoinStorageSession"
      ],
```

```
      "Resource": "arn:aws:kinesisvideo:us-west-2:123456789012:channel/
SignalingChannelName/1234567890123"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kinesisvideo:GetDataEndpoint",
        "kinesisvideo:DescribeStream",
        "kinesisvideo:PutMedia"
      ],
      "Resource": "arn:aws:kinesisvideo:us-west-2:123456789012:stream/
VideoStreamName/1234567890123"
    }
  ]
}
```

設定目的地

建立 Kinesis Video Streams 資源後，您需要告知訊號頻道要將其儲存到哪個串流。

如果您想要刪除訊號頻道或串流，您必須先將它們取消連結。請參閱 the section called “取消連結訊號頻道和串流”。

連結訊號頻道和串流

使用 UpdateMediaStorageConfiguration API，並為您要連結的 Kinesis Video Streams 資源輸入 ARNs。

> **Important**
>
> 一旦`StorageStatus`啟用，就不會再發生對peer-to-peer（主要檢視者）連線。對等直接連線至儲存工作階段。您必須呼叫 `JoinStorageSession` API 來觸發 SDP 優惠傳送，並在對等和儲存工作階段之間建立連線。

AWS 管理主控台

> **Note**
>
> Kinesis Video Streams 目前不支援此操作 AWS 管理主控台。

開啟 AWS CLI 已安裝並設定 的 AWS CloudShell 終端機。如需詳細資訊，請參閱 AWS CloudShell 使用者指南。

遵循 AWS CLI 索引標籤中的指示。

AWS CLI

確認您已安裝 AWS CLI 並設定 。如需詳細資訊，請參閱AWS Command Line Interface文件。

如需安裝說明，請參閱AWS Command Line Interface 《 使用者指南》。安裝後，使用登入資料和區域設定 AWS CLI。

或者，開啟已安裝並設定 AWS CLI 的 AWS CloudShell 終端機。如需詳細資訊，請參閱 AWS CloudShell 使用者指南。

在 中執行 `Update-Media-Storage-Configuration`命令 AWS CLI：

```
aws kinesisvideo update-media-storage-configuration \
  --channel-arn arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123 \
  --media-storage-configuration \
    StreamARN="arn:aws:kinesisvideo:us-
west-2:123456789012:stream/YourStreamName/1234567890123",Status="ENABLED" \
  --region "us-west-2"
```

AWS SDK

此程式碼片段說明如何使用適用於 JavaScript 的 AWS SDK v2 設定訊號頻道，將媒體擷取至指定的 Kinesis 影片串流。語法將與其他 AWS SDKs不同，但一般流程將相同。在 GitHub 上檢視完整的程式碼範例。

建立 Kinesis Video Streams 用戶端。這是用來呼叫 UpdateMediaStorageConfiguration API 的用戶端。

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

使用 用戶端呼叫 `UpdateMediaStorageConfiguration` API。

```
await kinesisVideoClient
    .updateMediaStorageConfiguration({
        ChannelARN: 'YourChannelARN',
        MediaStorageConfiguration: {
            Status: 'ENABLED',
            StreamARN: 'YourStreamARN',
        },
    })
    .promise();
```

具有此程式碼範例的即時網頁可在 GitHub 上使用。輸入您的區域、 AWS 憑證和訊號頻道的名稱。

展開 WebRTC 擷取和儲存節點，輸入串流的名稱，然後選擇更新媒體儲存組態。頻道會設定為將媒體擷取至指定的串流。

取消連結訊號頻道和串流

> **Important**
>
> 在訊號頻道或串流彼此取消連結之前，您無法將其刪除。

如果您不希望訊號頻道的媒體擷取到串流，請使用 UpdateMediaStorageConfiguration API 來取消連結 Kinesis Video Streams 資源。通道取消連結後，可以繼續直接peer-to-peer連線。

AWS 管理主控台

> **Note**
>
> Kinesis Video Streams 目前不支援此操作 AWS 管理主控台。

開啟 AWS CLI 已安裝並設定 的 AWS CloudShell 終端機。如需詳細資訊，請參閱 AWS CloudShell 使用者指南。

遵循 AWS CLI 索引標籤中的指示。

AWS CLI

確認您已安裝 AWS CLI 並設定 。如需詳細資訊，請參閱AWS Command Line Interface文件。

如需安裝說明，請參閱AWS Command Line Interface 《 使用者指南》。安裝後，使用登入資料和區域設定 AWS CLI。

或者，開啟 AWS CLI 已安裝並設定 的 AWS CloudShell 終端機。如需詳細資訊，請參閱 AWS CloudShell 使用者指南。

在 中執行 Update-Media-Storage-Configuration命令 AWS CLI：

```
aws kinesisvideo update-media-storage-configuration \
  --channel-arn arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123 \
  --media-storage-configuration \
    StreamARN="null",Status="DISABLED" \
  --region "us-west-2"
```

AWS SDK

此程式碼片段說明如何使用適用於 JavaScript 的 AWS SDK v2 設定訊號頻道，將媒體擷取至指定的 Kinesis 影片串流。語法將與其他 AWS SDKs不同，但一般流程將相同。在 GitHub 上檢視完整的程式碼範例。

建立 Kinesis Video Streams 用戶端。這是用來呼叫 UpdateMediaStorageConfiguration API 的用戶端。

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

使用 用戶端呼叫 UpdateMediaStorageConfiguration API。

```
await kinesisVideoClient
    .updateMediaStorageConfiguration({
        ChannelARN: 'YourChannelARN',
        MediaStorageConfiguration: {
            Status: 'DISABLED',
            StreamARN: 'null',
        },
    })
```

```
    .promise();
```

具有此程式碼範例的即時網頁可在 GitHub 上使用。輸入您的區域、 AWS 憑證和訊號頻道的名稱。

展開 WebRTC 擷取和儲存節點，確認串流名稱欄位為空白，然後選擇更新媒體儲存組態。頻道將不再設定為將媒體擷取至指定的串流。

擷取媒體

有下列限制：

- 工作階段持續時間：一小時，最長
- 訊號頻道：每個已啟用儲存組態的帳戶最多 100 個

主題

- 從瀏覽器擷取媒體
- 從 WebRTC C SDK 擷取媒體
- 將檢視器新增至擷取工作階段

從瀏覽器擷取媒體

> **Important**
>
> Chrome 目前是唯一支援的瀏覽器。

1. 在 JavaScript 範例頁面中開啟 Amazon Kinesis Video Streams with WebRTC SDK。
2. 請填妥下列資訊：
   - KVS 端點 - 在區域欄位中，選取您的區域。

     例如 `us-west-2`。
   - AWS 憑證

     完成下列欄位：
     - Access Key ID (存取金鑰 ID)

    - Secret Access Key (私密存取金鑰)
    - 工作階段字符 - 範例應用程式支援臨時和長期登入資料。如果您使用長期 IAM 登入資料，請將此欄位保留空白。如需詳細資訊，請參閱 IAM 中的臨時安全登入資料。
  - 訊號頻道 - 在頻道名稱欄位中，輸入您先前設定的訊號頻道名稱。如需詳細資訊，請參閱the section called “設定目的地”。
  - 音軌 - 選取傳送視訊和傳送音訊。
  - WebRTC 擷取和儲存 - 展開節點，然後選取自動判斷擷取模式。此選項可讓範例應用程式呼叫 DescribeMediaStorageConfiguration API，以判斷要在哪個模式下執行。
3. 選取啟動主伺服器。

   如果訊號頻道設定為使用 DescribeMediaStorageConfiguration API 擷取，則範例應用程式會在連線至訊號頻道以啟動 WebRTC 擷取工作流程後立即自動叫用 JoinStorageSession API。

從 WebRTC C SDK 擷取媒體

依照the section called “適用於嵌入式裝置的 C 開發套件”程序建置範例應用程式。

1. 使用 AWS 帳戶 登入資料設定您的環境：

   ```
   export AWS_ACCESS_KEY_ID=YourAccessKey
   export AWS_SECRET_ACCESS_KEY=YourSecretKey
   export AWS_DEFAULT_REGION=YourAWSRegion
   ```

   如果您使用的是暫時 AWS 登入資料，也請匯出工作階段字符：

   ```
   export AWS_SESSION_TOKEN=YourSessionToken
   ```

2. 執行範例：

   主要範例

   導覽至 build 資料夾，並使用「1」做為第二個引數。類型：

   ```
   ./samples/kvsWebrtcClientMaster channel-name 1
   ```

   GStreamer 主要範例

   導覽至 build 資料夾，並使用「audio-video-storage」做為第二個引數。類型：

```
./samples/kvsWebrtcClientMasterGstSample channel-name audio-video-storage testsrc
```

這會啟動 WebRTC 擷取。

> **Note**
>
> 您提供的訊號頻道必須設定為儲存。使用 DescribeMediaStorageConfiguration API 進行確認。

將檢視器新增至擷取工作階段

一旦訊號頻道進入 WebRTC 擷取模式，檢視器參與者就不會再直接連線至主要參與者。檢視器參與者會直接連線至儲存工作階段。檢視器參與者會收到主要參與者傳送的媒體，檢視器參與者可以將選用的音訊傳回給主要參與者。只要主要參與者連線至儲存工作階段，檢視器傳回的任何音訊都會傳送至連線至儲存工作階段並擷取至 Kinesis Video Stream 的所有其他對等。

有下列限制：

- 檢視器數量上限：3
- 檢視器參與者可以在沒有主要參與者的情況下連線到儲存工作階段的時間上限：3 分鐘

> **Important**
>
> 如果檢視器與儲存工作階段中斷連線（關閉對等連線），其配額（檢視器限制）會維持 1 分鐘。在此 1 分鐘期間，檢視器可以使用相同的用戶端 ID 叫用此 API，以重新加入工作階段，而不需要額外的檢視器配額。1 分鐘後，會釋出檢視器配額，供其他檢視器加入。

瀏覽器

> **Important**
>
> Chrome 是唯一支援的瀏覽器。

1. 在 JavaScript 範例頁面的 Amazon Kinesis Video Streams with WebRTC SDK 中開啟另一個索引標籤。上一頁的所有資訊都會自動填入。如果沒有，請完成下列資訊：

   - KVS 端點 - 在區域欄位中，選取您的區域。

     例如 `us-west-2`。

   - AWS 憑證

     完成下列欄位：

     - Access Key ID (存取金鑰 ID)
     - Secret Access Key (私密存取金鑰)
     - 工作階段字符 - 範例應用程式支援臨時和長期憑證。如果您使用長期 IAM 登入資料，請將此欄位保留空白。如需詳細資訊，請參閱 IAM 中的臨時安全登入資料。

   - 訊號頻道 - 在頻道名稱欄位中，輸入您先前設定的訊號頻道名稱。如需詳細資訊，請參閱the section called “設定目的地”。
   - 音軌 - 選取傳送音訊。請注意，如果已勾選傳送視訊，則選擇啟動檢視器時會自動取消勾選。
   - WebRTC 擷取和儲存 - 展開節點，然後選取自動判斷擷取模式。此選項可讓範例應用程式呼叫 DescribeMediaStorageConfiguration API，以判斷要在哪個模式下執行。

2. 選取啟動檢視器。

   應用程式會在連線至訊號頻道後立即自動呼叫 JoinStorageSessionAsViewer API，以觸發從工作階段傳送給檢視器的 SDP 優惠。

> **Note**
>
> 透過peer-to-peer WebRTC，檢視器參與者是控制對等，而主參與者是控制對等。在 WebRTC 擷取模式中，儲存工作階段現在是控制對等。連線至訊號並叫用 JoinStorageSessionAsViewer 之後，檢視器將需要回應 SDP 優惠，並透過 WebRTC 建立與儲存工作階段的連線。

> **Note**
>
> 儲存工作階段只會傳送TURN候選項目。從參與者的觀點來指定 ICE 候選配對時，遠端候選一律為類型 `relay`。

播放擷取的媒體

您可以在 主控台中檢視媒體資料，或使用 Hypertext Live Streaming (HLS) 建立從串流讀取媒體資料的應用程式，以取用媒體資料。

在 主控台中檢視媒體

在另一個瀏覽器索引標籤中，開啟 AWS 管理主控台。在 Kinesis Video Streams Dashboard 中，選取影片串流。

在串流清單中選取串流的名稱。如有必要，請使用搜尋列。

展開媒體播放區段。如果影片仍在上傳，則會顯示該影片。如果上傳已完成，請選取左雙箭頭。

使用 HLS 使用媒體資料

您可以建立使用 HLS 從 Kinesis 影片串流取用資料的用戶端應用程式。如需有關建立使用 HLS 取用媒體資料的應用程式的資訊，請參閱 Kinesis Video Streams 播放。

使用範例媒體檢視器應用程式檢視媒體

Amazon Kinesis Video Streams 建立了媒體檢視器的範例網頁實作，能夠在 HLS 或 DASH 中播放 Kinesis Video Streams。您可以在 GitHub 上檢視原始程式碼，並嘗試使用託管網頁。

1. 開啟 Amazon Kinesis Video Streams Media Viewer。
2. 完成下列欄位：
   - 區域 - 選取 us-west-2。
   - AWS 存取金鑰
   - AWS 私密金鑰
   - 串流名稱
   - 播放模式 - 選取即時。
3. 選取開始播放。

連線至儲存工作階段

請依照這些程序建立儲存工作階段並啟動 WebRTC 連線程序。主要參與者應呼叫 JoinStorageSession。觀眾參與者應呼叫 JoinStorageSessionAsViewer。

這可讓儲存工作階段透過向透過 連線的主要參與者，或透過 連線的指定檢視器參與者發出訊號the section called “ConnectAsMaster”，來傳送 SDP 優惠和 ICE 候選者the section called “ConnectAsViewer”。

1. 取得訊號頻道的 ARN，因為它是下一個步驟所需的輸入。如果您已經知道 ARN，請繼續下一個步驟。

   AWS 管理主控台

   1. 開啟 Kinesis Video Streams Signaling Channels 主控台。
   2. 選擇訊號頻道的名稱。
   3. 在訊號頻道資訊索引標籤下，找到訊號頻道的 ARN。

   AWS CLI

   確認您已安裝 AWS CLI 並設定 。如需詳細資訊，請參閱「AWS Command Line Interface 使用者指南」。

   如需安裝說明，請參閱AWS Command Line Interface 《 使用者指南》。安裝後，使用登入資料和區域設定 AWS CLI。

   或者，開啟已安裝並設定 AWS CLI 的 AWS CloudShell 終端機。如需詳細資訊，請參閱 AWS CloudShell 使用者指南。

   使用 執行下列 Describe-Signaling-Channel 命令 AWS CLI：

   ```
   aws kinesisvideo describe-signaling-channel \
     --channel-name "YourChannelName" \
   ```

   回應如下所示：

   ```
   {
       "ChannelInfo": {
           "ChannelName": "YourChannelName",
           "ChannelARN": "arn:aws:kinesisvideo:us-
   west-2:123456789012:channel/YourChannelName/1234567890123",
           "ChannelType": "SINGLE_MASTER",
           "ChannelStatus": "ACTIVE",
           "CreationTime": "2024-07-07T23:28:24.941000-07:00",
           "SingleMasterConfiguration": {
   ```

```
        "MessageTtlSeconds": 60
    },
    "Version": "Ws0fZvFGXzEpuZ2CE1s9"
    }
}
```

您可以在 ChannelInfo 物件中找到訊號頻道 ARN。

AWS SDK

此程式碼片段說明如何使用適用於 JavaScript 的 AWS SDK v2 建立 Kinesis Video Streams with WebRTC 訊號頻道。語法將與其他 AWS SDKs 不同，但一般流程將相同。

您可以檢視 JoinStorageSession 或 JoinStorageSessionAsViewer 的完整程式碼範例。

建立 Kinesis Video Streams 用戶端。這用於呼叫 DescribeSignalingChannel API。

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

使用 用戶端呼叫 DescribeSignalingChannel API。

```
const describeSignalingChannelResponse = await kinesisVideoClient
    .describeSignalingChannel({
        ChannelName: 'YourChannelName',
    })
    .promise();
```

儲存回應。

```
const channelARN =
    describeSignalingChannelResponse.ChannelInfo.ChannelARN;
```

2. 取得 WEBRTC 端點。對於特定訊號頻道的 JoinStorageSession 或 JoinStorageSessionAsViewer 請求，必須向指定的端點提出。

AWS 管理主控台

> **Note**
>
> Kinesis Video Streams 目前不支援此操作 AWS 管理主控台。

開啟 AWS CLI 已安裝並設定 的 AWS CloudShell 終端機。如需詳細資訊，請參閱 AWS CloudShell 使用者指南。

遵循 AWS CLI 索引標籤中的指示。

AWS CLI

確認您已安裝 AWS CLI 並設定 。如需詳細資訊，請參閱AWS Command Line Interface文件。

如需安裝說明，請參閱AWS Command Line Interface 《 使用者指南》 。安裝後，使用登入資料和區域設定 AWS CLI。

或者，開啟已安裝並設定 AWS CLI 的 AWS CloudShell 終端機。如需詳細資訊，請參閱 AWS CloudShell 使用者指南。

在 中執行 `Get-Signaling-Channel-Endpoint`命令 AWS CLI：

```
aws kinesisvideo get-signaling-channel-endpoint \
  --channel-arn "arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123" \
  --single-master-channel-endpoint-configuration
 "Protocols=['WEBRTC'],Role=MASTER" \
  --region "us-west-2"
```

回應看起來類似以下的內容。

```
{
    "ResourceEndpointList": [
        {
            "Protocol": "WEBRTC",
            "ResourceEndpoint": "https://w-abcd1234.kinesisvideo.aws-
region.amazonaws.com"
        }
    ]
```

```
}
```

AWS SDK

此程式碼片段說明如何使用適用於 JavaScript 的 AWS SDK v2 呼叫 Kinesis Video Streams with WebRTC 訊號頻道的 `GetSignalingChannelEndpoint` API。語法將與其他 AWS SDKs不同，但一般流程將相同。檢視 JoinStorageSession 或 JoinStorageSessionAsViewer 的完整程式碼範例。

建立 Kinesis Video Streams 用戶端。這是用來呼叫 DescribeSignalingChannel API 的用戶端。

如果您先前已建立 Kinesis Video Streams 用戶端來呼叫 `DescribeSignalingChannel`，則可以重複使用相同的用戶端。

```
const clientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2'
};
const kinesisVideoClient = new AWS.KinesisVideo(clientConfig);
```

使用 用戶端呼叫 `GetSignalingChannelEndpoint` API。

```
const getSignalingChannelEndpointResponse = await kinesisVideoClient
    .getSignalingChannelEndpoint({
        ChannelARN: channelARN,
        SingleMasterChannelEndpointConfiguration: {
            Protocols: ['WEBRTC'],
            Role: 'MASTER',
        },
    })
    .promise();
```

儲存回應：

```
const webrtcEndpoint =
 getSignalingChannelEndpointResponse.ResourceEndpointList[0].ResourceEndpoint;
```

3. 使用頻道 ARN 和 WEBRTC 端點進行 API 呼叫。如果參與者透過 `ConnectAsMaster`或 WebSocket API 使用 WebRTC Signaling 正確連接到 Kinesis Video Streams，則 SDP 優惠和 ICE 候選訊息串流將沿著 WebSocket 從儲存工作階段傳送給參與者。 `ConnectAsViewer` WebSocket APIs

   AWS 管理主控台

   > **Note**
   >
   > Kinesis Video Streams 目前不支援此操作 AWS 管理主控台。

   開啟 AWS CLI 已安裝並設定 的 AWS CloudShell 終端機。如需詳細資訊，請參閱 AWS CloudShell 使用者指南。

   遵循 AWS CLI 索引標籤中的指示。

   AWS CLI

   確認您已安裝 AWS CLI 並設定 。如需詳細資訊，請參閱AWS Command Line Interface文件。

   如需安裝說明，請參閱AWS Command Line Interface 《 使用者指南》 。安裝後，使用登入資料和區域設定 AWS CLI。

   或者，開啟已安裝並設定 AWS CLI 的 AWS CloudShell 終端機。如需詳細資訊，請參閱 AWS CloudShell 使用者指南。

   AWS CLI 使用先前步驟中的頻道 ARN 和 WEBRTC 端點，在 中為主要參與者執行 `Join-Storage-Session`命令：

   ```
   aws kinesis-video-webrtc-storage join-storage-session \
     --endpoint-url https://w-abcd1234.kinesisvideo.us-west-2.amazonaws.com \
     --channel-arn arn:aws:kinesisvideo:us-
   west-2:123456789012:channel/YourChannelName/1234567890123 \
     --region "us-west-2"
   ```

   成功執行後，會傳回空的回應，而且不會列印任何內容。

   如果是瀏覽者參與者，請使用 之前的頻道 ARN 和 WEBRTC 端點，在 AWS CLI 中執行下列`Join-Storage-Session-As-Viewer`命令：

```
aws kinesis-video-webrtc-storage join-storage-session-as-viewer \
  --endpoint-url https://w-abcd1234.kinesisvideo.us-west-2.amazonaws.com \
  --channel-arn arn:aws:kinesisvideo:us-
west-2:123456789012:channel/YourChannelName/1234567890123 \
  --client-id "ExampleViewerClientID"
  --region "us-west-2"
```

成功執行後，會傳回空的回應，而且不會列印任何內容。

AWS SDK

此程式碼片段說明如何使用適用於 JavaScript 的 AWS SDK v2 呼叫 Kinesis Video Streams with WebRTC 訊號頻道的 `JoinStorageSession`或 `JoinStorageSessionAsViewer` API。語法將與其他 AWS SDKs 不同，但一般流程將相同。檢視 JoinStorageSession 或 JoinStorageSessionAsViewer 的完整程式碼範例。

建立 Kinesis Video Streams WebRTC 儲存用戶端。這是用來呼叫 `JoinStorageSession`或的用戶端，`JoinStorageSessionAsViewer`與先前步驟中建立的 Kinesis Video Streams 用戶端不同。

```
const webrtcStorageClientConfig = {
    accessKeyId: 'YourAccessKey',
    secretAccessKey: 'YourSecretKey',
    region: 'us-west-2',
    endpoint: webrtcEndpoint
};
const kinesisVideoWebRTCStorageClient = new
 AWS.KinesisVideoWebRTCStorage(clientConfig);
```

使用 用戶端呼叫主要參與者的 `JoinStorageSession` API。

```
await kinesisVideoWebRTCStorageClient
    .joinStorageSession({
        channelArn: channelARN,
    })
    .promise();
```

如果是瀏覽者參與者，請使用 用戶端呼叫 `JoinStorageSessionAsViewer` API。

```
await kinesisVideoWebRTCStorageClient
```

```
    .joinStorageSessionAsViewer({
        channelArn: channelARN,
        clientId: "ExampleViewerClientID",
    })
    .promise();
```

具有此程式碼範例的即時網頁可在 GitHub 上取得。輸入您的區域、 AWS 憑證和訊號頻道的名稱。展開 WebRTC 擷取和儲存節點，然後取消核取自動判斷擷取模式。

切換手動覆寫至 ON，然後選取顯示按鈕以手動呼叫 JoinStorageSession API 和/或顯示按鈕以手動呼叫 JoinStorageSessionAsViewer API。

當您選取 Start Master 或 Start Viewer 時，在應用程式透過 `ConnectAsMaster`或 連線至訊號後`ConnectAsViewer`，會顯示一個按鈕，讓儲存工作階段啟動與對等的 WebRTC 連線，而不是在連線至訊號後立即呼叫 API 的應用程式。

對與儲存工作階段連線的問題進行故障診斷

本節提供有關設定和設定錄製影片串流之儲存體相關問題的疑難排解指引。

主題

- 控制和控制對等
- 檢閱支援的轉碼器
- 如果頻道未映射至串流， 也會擲回 400 InvalidArgumentException

控制和控制對等

在 WebRTC 中，控制對等會透過傳送 SDP 優惠來啟動與控制對等的連線。對於peer-to-peer工作階段，檢視器參與者會透過訊號傳送優惠給主要參與者，以啟動連線。連線至 WebRTC 擷取的儲存工作階段時，儲存工作階段是控制對等。對於主要參與者，他們仍然是受控參與者。不過，瀏覽者參與者會從控制切換到控制。

呼叫 JoinStorageSession 或 JoinStorageSessionAsViewer 時，所有參與者都必須以 SDP 答案回應，並使用儲存工作階段交換 ICE 候選者。

如需序列圖，請參閱了解 WebRTC 擷取和儲存。

> **Important**
>
> 從儲存工作階段收到的訊號訊息在 JSON 中沒有 senderClientId 欄位，這與peer-to-peer主控端不同，該主控端一律會接收具有 SDP 優惠和 ICE 候選項目的 senderClientId 欄位。

檢閱支援的轉碼器

傳送 SDP 答案並將 ICE 候選項目與儲存工作階段交換時，我們建議在訊息`correlationId`中包含 。在訊息`correlationId`中包含 可讓儲存工作階段傳回`statusResponse`訊息。這些訊息將包含輸入訊息`correlationId`的 ，可讓您追蹤 `statusResponse`所屬的訊息。這可讓您立即收到拒絕 SDP 答案原因的意見回饋。

如需 `correlationId` 和 `statusResponse` 的更多相關資訊，請參閱the section called “非同步訊息接收”。

儲存工作階段可能會拒絕 SDP 答案的一個常見原因是儲存工作階段無法接受答案中指定的轉碼器。範例`statusResponse`可能如下所示：

```
{
  "correlationId": "1700186220273",
  "errorType": "InvalidArgumentException",
  "statusCode": "400",
  "success": false
}
```

當您檢閱 SDP 答案內容時，請檢閱開頭為 的行，`a=rtpmap`並確認轉碼器符合儲存工作階段支援的轉碼器。以下是包含 opus 音訊和 VP8 影片的範例 SDP 答案程式碼片段。

```
...
a=rtpmap:111 opus/48000/2
...
a=rtpmap:120 VP8/90000
...
```

如需支援的轉碼器清單，請參閱 JoinStorageSession。

如果頻道未映射至串流，也會擲回 400 InvalidArgumentException

檢閱收發器方向

在 SDP 答案中，檢閱視訊和音訊收發器的方向性。SDP 中的行如下所示：

```
a=sendrecv
a=recvonly
```

對於主要參與者，有下列要求：

- H.264 影片：僅傳送
- Opus 音訊：sendonly 或 sendrecv

對於瀏覽者參與者，有下列要求：

- H.264 影片：recvonly
- Opus 音訊：recvonly 或 sendrecv

如果不符合服務需求，則在傳送答案時提供 correlationId 時，將會傳回具有 400 IllegalArgumentException 的 statusResponse。

使用 KVS 提供的 WebRTC 擷取範例應用程式時：

- 主要參與者：確保已啟用「傳送視訊」和「傳送音訊」設定
- 觀眾參與者：確保已停用「傳送影片」設定

檢閱 ICE 候選轉換

從 KVS Signaling SDK 接收 ICE 候選項目時，應用程式可能需要將字串轉換為 ICE 候選物件。

從儲存工作階段收到的 ICE 候選項目不包含 SDPMID 屬性，也不隨附 senderClientId。

檢閱您的應用程式邏輯，以將從遠端接收的 ICE 候選項目新增至應用程式的 RTCPeerConnection 物件。

檢閱 ICE 候選佇列邏輯

從儲存工作階段收到的所有 ICE 候選項目都需要透過 addIceCandidate API 新增至 RTCPeerConnection。

即使儲存工作階段在 ICE 候選者之前傳送 SDP 優惠，由於訊號訊息傳遞的非同步性質，應用程式可能會在收到優惠之前收到 ICE 候選者。

在這種情況下，您將需要實作暫時緩衝區來保留 ICE 候選項目。

KVS 範例應用程式中實作的邏輯如下：

1. 這些範例會維護 senderClientId 至其 RTCPeerConnection 的映射。
2. 這些範例會將 senderClientId 的另一個映射維護到待定的 Ice Candidates 清單。
3. 收到 ICE 候選項目時，請檢查 PeerConnection 映射。如果 PeerConnection 映射還沒有連線，請將其新增至待定清單。否則，請將 ICE 候選項目新增至 PeerConnection。
4. 收到優惠時，它會建立 RTCPeerConnection 並將其新增至 PeerConnection 映射。然後，從待定佇列將所有 ICE 候選項目新增至此 PeerConnection。

搭配 Amazon Kinesis Video WebRTC 使用 IPv6/雙堆疊端點

您可以設定 Amazon Kinesis Video WebRTC 為控制平面和資料平面操作使用 IPv6。這可讓您的應用程式透過雙堆疊端點，使用 IPv6 地址與 Kinesis Video WebRTC 服務通訊。

> **Note**
>
> IPv6 支援需要特定的 SDK 版本和組態設定。確保您的 Kinesis Video WebRTC SDK 和 Amazon Web Services SDK 版本支援 IPv6 雙堆疊端點。雙堆疊端點同時支援 IPv4 和 IPv6 流量，適用於某些區域中的某些 服務。

Amazon Kinesis Video WebRTC 透過雙堆疊端點支援主要和檢視器應用程式的 IPv6。您可以設定應用程式使用 IPv6/雙堆疊端點進行控制平面 API 呼叫和資料平面操作。

設定適用於 IPv6-Dual-Stack 端點的 Amazon Web Services SDK

如果您使用 Amazon Web Services SDK 在生產設定中呼叫 Kinesis Video WebRTC 控制平面 APIs，您可以透過設定雙堆疊端點來啟用 IPv6。Amazon Web Services SDK 提供多種標準化方法來啟用雙堆疊端點。

> **Important**
>
> 啟用雙堆疊端點時，軟體開發套件會嘗試使用雙堆疊端點來提出網路請求。如果服務或區域不存在雙堆疊端點，則請求會失敗。

使用環境變數

設定下列環境變數以啟用 IPv6 雙堆疊端點：

```
export AWS_USE_DUALSTACK_ENDPOINT=true
```

使用 Amazon Web Services 組態檔案

將下列設定新增至您的 Amazon Web Services 組態檔案 (~/.aws/config)：

```
[default]
use_dualstack_endpoint = true
```

使用 JVM 系統屬性 （僅限 Java 和 Kotlin SDKs)

對於 Java 和 Kotlin 應用程式，設定下列 JVM 系統屬性：

```
-Daws.useDualstackEndpoint=true
```

或以程式設計方式在您的 Java 程式碼中：

```
System.setProperty("aws.useDualstackEndpoint", "true");
```

開發套件支援

下列 Amazon Web Services SDKs支援雙堆疊端點組態：

SDK	支援	組態方法
AWS CLI v2	是	環境變數、組態檔案
適用於 C++ 的 SDK	是	環境變數、組態檔案
適用於 Go V2 的 SDK (1.x)	是	環境變數、組態檔案
適用於 Go 的 SDK 1.x (V1)	是	環境變數、組態檔案
適用於 Java 2.x 的 SDK	是	環境變數、組態檔案、JVM 屬性
適用於 Java 1.x 的 SDK	否	不支援
適用於 JavaScript 3.x 的 SDK	是	環境變數、組態檔案
適用於 JavaScript 2.x 的 SDK	是	環境變數、組態檔案
SDK for Kotlin	是	環境變數、組態檔案、JVM 屬性

SDK	支援	組態方法
適用於 .NET 4.x 的 SDK	是	環境變數、組態檔案
適用於 .NET 3.x 的 SDK	是	環境變數、組態檔案
適用於 PHP 的 SDK 3.x	是	環境變數、組態檔案
適用於 Python 的 SDK (Boto3)	是	環境變數、組態檔案
適用於 Ruby 的 SDK 3.x	是	環境變數、組態檔案
適用於 Rust 的 SDK	是	環境變數、組態檔案
適用於 Swift 的 SDK	是	環境變數、組態檔案
Tools for PowerShell V5	是	環境變數、組態檔案
Tools for PowerShell V4	是	環境變數、組態檔案

設定雙堆疊端點之後，Amazon Web Services SDK 會在呼叫 Kinesis Video WebRTC 控制平面 APIs 時自動使用 IPv6 端點。

設定適用於 IPv6/雙堆疊端點的 Kinesis Video WebRTC 開發套件

Kinesis Video WebRTC SDK 為控制平面和資料平面操作提供雙堆疊組態選項。這些設定適用於 Amazon Web Services SDK 雙堆疊端點組態。

設定 WebRTC C 開發套件

若要使用雙堆疊 AWS KVS 端點並嘗試收集 IPv6 ICE 候選項目，請設定下列環境變數：

```
export KVS_DUALSTACK_ENDPOINTS=ON
```

在雙堆疊模式中，ICE 收集會嘗試包含 IPv6 候選項目，但相容性最終取決於本機網路組態和接收對等的功能。

若要停用雙堆疊模式，請取消設定環境變數：

```
unset KVS_DUALSTACK_ENDPOINTS
```

資料平面端點解析

對於資料平面操作，Kinesis Video WebRTC SDK 使用 GetSignalingChannelEndpoint API 擷取適當的 IPv6/雙堆疊資料平面端點。設定 IPv6/雙堆疊時，軟體開發套件會自動請求 IPv6/雙堆疊端點。

> **Important**
>
> GetSignalingChannelEndpoint API 已更新以支援 IPv6 端點。請確定您使用的是支援此功能的相容 SDK 版本。

設定 IPv6/Dual-Stack AWS CLI 的

如果您使用 AWS CLI 進行 Kinesis Video WebRTC 操作（通常用於proof-of-concept工作），您可以透過設定雙堆疊端點來啟用 IPv6。

使用環境變數

```
export AWS_USE_DUALSTACK_ENDPOINT=true
```

使用 Amazon Web Services 組態檔案

將下列項目新增至您的 AWS CLI 組態檔案 (~/.aws/config)：

```
[default]
use_dualstack_endpoint = true
```

設定雙堆疊端點之後， AWS CLI 會將 IPv6 雙堆疊端點用於所有 Amazon Web Services 呼叫，包括 Kinesis Video WebRTC 操作。

考量事項

IoT 登入資料供應商

如果您使用 IoT 登入資料進行身分驗證：

- IoT 登入資料端點支援 IPv6
- 使用上述標準 Amazon Web Services SDK 組態方法設定雙堆疊端點
- IoT 登入資料流程與 Kinesis Video WebRTC 特定的 IPv6 組態不同

網路需求

- 確保您的網路基礎設施支援 IPv6 連線
- 確認您的安全群組和網路 ACLs IPv6 流量
- 從部署環境測試 Amazon Web Services IPv6 端點的連線
- 某些 區域中的某些服務可使用雙堆疊端點 - 驗證目標區域的可用性

開發套件相容性

- 確保您使用的是支援的 Amazon Web Services SDK 版本 （請參閱相容性資料表）
- 適用於 Java 的 Amazon Web Services 開發套件 1.x 不支援雙堆疊端點組態
- 對於適用於 Go 1.x (V1) 的 SDK，您必須啟用從組態檔案載入，才能使用共用的組態檔案設定

測試和驗證

將IPv6-enabled Kinesis Video WebRTC 應用程式部署到生產環境之前：

- 測試控制平面操作 （頻道建立、刪除、列出）
- 驗證資料平面操作 (STUN、TURN 和 WebRTC 訊號）
- 驗證成功建立peer-to-peer串流工作階段
- 驗證網路環境中的效能和連線
- 執行 Canary 測試以確保一致的 IPv6 功能
- 雙堆疊端點無法使用時的測試容錯移轉行為

受 IPv6 升級影響的客戶

當您為 Amazon Kinesis Video WebRTC 啟用 IPv6 時，可能需要更新現有組態和政策以確保持續功能。本節概述轉換到IPv6-enabled的端點時需要注意的關鍵區域。

IAM 政策和 IP 地址篩選

如果您在 IAM 使用者政策、角色政策或資源型政策中使用來源 IP 地址篩選，則需要更新這些政策以包含 IPv6 地址範圍。

> **Important**
>
> 在 IpAddress 或 NotIpAddress 條件中使用 IPv4 CIDR 區塊的現有 IAM 政策不會自動使用 IPv6 地址。您必須明確新增 IPv6 範圍以維護存取控制。

IPv6 的 IAM 政策更新範例：

```
{
  "Version": "2012-10-17"       ,
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "kinesisvideo:*",
      "Resource": "*",
      "Condition": {
        "IpAddress": {
          "aws:SourceIp": [
            "192.0.2.0/24",
            "203.0.113.0/24",
            "2001:db8::/32"
          ]
        }
      }
    }
  ]
}
```

IAM 政策更新的重要考量事項：

- 新增 IPv6 CIDR 區塊與現有的 IPv4 範圍
- 使用 IPv4 和 IPv6 地址的 aws：SourceIp 條件索引鍵
- 在非生產環境中測試政策再部署
- 考慮使用 aws：RequestedRegion 作為增強安全性的額外條件

網路安全群組和存取控制清單

如果您在 Amazon EC2 執行個體或其他 Amazon Web Services 服務上執行 Kinesis Video WebRTC 應用程式，則需要更新您的安全群組和網路 ACLs以允許 IPv6 流量。

- 安全群組 – 新增 IPv6 CIDR 區塊的傳入和傳出規則 （所有 IPv6 流量或特定 IPv6 範圍：：/0)
- 網路 ACLs – 更新子網路層級網路 ACLs，以允許必要連接埠上的 IPv6 流量
- 路由表 – 確保您的 VPC 路由表包含 IPv6 流量到達網際網路閘道或 NAT 閘道的路由

日誌記錄和監控

IPv6 地址的格式與 IPv4 地址不同，這可能會影響您的記錄、監控和分析系統。

AWS CloudTrail 日誌

AWS CloudTrail 當透過 IPv6 提出請求時， 日誌會在 sourceIPAddress 欄位中包含 IPv6 地址。更新您的日誌剖析工具和指令碼，以處理 IPv6 地址格式。

AWS CloudTrail 日誌中 IPv6 地址的範例：

```
{
  "sourceIPAddress": "2001:db8::1",
  "eventName": "CreateSignalingChannel",
  "eventSource": "kinesisvideo.amazonaws.com"
}
```

應用程式記錄

如果您的應用程式記錄用戶端 IP 地址或執行 IP 型分析，請確定您的記錄基礎設施可以處理 IPv6 地址：

- 更新日誌剖析規則運算式以符合 IPv6 格式
- 如果您使用固定長度欄位存放 IP 地址，請修改資料庫結構描述
- 更新分析查詢和儀表板以使用 IPv6 地址
- 考慮使用 IP 地址標準化程式庫進行一致處理

監控和提醒

更新您的監控和提醒系統，以考量 IPv6 流量：

- 依 IP 地址篩選的 Amazon CloudWatch 指標和警示
- 追蹤 IP 模式的自訂指標
- 分析流量模式的安全性監控工具
- 將 IP 地址映射至 位置的地理位置服務

第三方整合

檢閱和更新與 Kinesis Video WebRTC 應用程式整合的第三方服務和工具：

- 內容交付網路 CDNs) – 如果您使用 CDNs 組態支援 IPv6
- 負載平衡器 – 設定 Application Load Balancer 或 Network Load Balancer 來處理 IPv6 流量
- DNS 服務 – 更新 DNS 記錄以包含 IPv6 地址的 AAAA 記錄
- 防火牆和安全設備 – 設定網路安全設備以允許 IPv6 流量
- 監控工具 – 驗證第三方監控和分析工具是否支援 IPv6 地址格式

應用程式程式碼更新

檢閱您的應用程式程式碼，了解可能需要更新的 IPv4-specific假設：

- IP 地址驗證 – 更新輸入驗證以接受 IPv6 地址格式
- 資料庫結構描述 – 確保 IP 地址欄位可以存放 IPv6 地址 （通常需要較大的欄位大小）
- 組態檔案 – 更新任何硬式編碼 IPv4 地址或 CIDR 區塊
- 用戶端程式庫 – 驗證 HTTP 用戶端和網路程式庫支援 IPv6
- 錯誤處理 – 更新錯誤處理以考量 IPv6-specific網路錯誤

測試和驗證

在生產環境中啟用 IPv6 之前，請徹底測試您的應用程式和基礎設施：

- 連線測試 – 驗證所有元件都可以透過 IPv6 進行通訊
- 效能測試 – 比較 IPv6 和 IPv4 效能以識別任何問題
- 安全測試 – 驗證安全控制是否與 IPv6 流量正常運作
- 容錯移轉測試 – 無法使用 IPv6 連線時的測試行為

- 日誌分析 – 驗證日誌記錄和監控系統是否正確處理 IPv6 地址
- 整合測試 – 測試所有啟用 IPv6 的第三方整合

遷移策略

考慮實作 IPv6 採用的分階段方法：

1. 評估階段 – 清查所有系統並識別 IPv6 準備程度
2. 準備階段 – 更新政策、安全群組和應用程式碼
3. 測試階段 – 在開發和預備環境中啟用 IPv6
4. 試驗階段 – 為生產流量的子集啟用 IPv6
5. 完整部署 – 逐漸增加 IPv6 流量，直到完全部署為止
6. 監控階段 – 持續監控問題並最佳化效能

疑難排解

常見問題

- 連線失敗 – 驗證 IPv6 網路連線和 DNS 解析
- SDK 錯誤 – 確保您使用的是支援雙堆疊端點的相容 SDK 版本
- 身分驗證問題 – 確認 IAM 政策和登入資料可與 IPv6 端點搭配使用
- 端點無法使用 – 如果服務或區域不存在雙堆疊端點，則請求會失敗

驗證步驟

- 檢查您的組態檔案中是否已設定 AWS_USE_DUALSTACK_ENDPOINT=true 或使用 _dualstack_endpoint = true
- 確認 Kinesis Video WebRTC SDK IPv6 組態旗標已正確設定
- 測試 Amazon Web Services IPv6 端點的網路連線能力
- 檢閱應用程式日誌是否有 IPv6-specific錯誤訊息
- 確認您的區域支援 Kinesis Video WebRTC 的雙堆疊端點

組態驗證

您可以檢查以下項目來驗證雙堆疊端點組態：

- 環境變數：回應 $AWS_USE_DUALSTACK_ENDPOINT
- Amazon Web Services 組態檔案：cat ~/.aws/config | grep use_dualstack_endpoint
- JVM 屬性 (Java)：檢查應用程式日誌中的系統屬性

如需其他支援和故障診斷，請參閱 AWS 文件或聯絡人 AWS。

多檢視程式

Amazon Kinesis Video Streams Multiviewer 是一種雲端型 WebRTC 解決方案，可讓多個檢視器從單一攝影機裝置同時加入即時影片串流工作階段。此功能透過處理雲端中的影片串流來解決邊緣裝置的頻寬和運算限制，而不是要求攝影機將個別串流傳送給每個檢視器。

主題

概觀

傳統的peer-to-peer WebRTC 連線需要攝影機裝置分別將影片傳送至每個檢視器，如此可以快速讓頻寬或運算容量有限的裝置過載。Multiviewer 會使用雲端型「混音器」來解決此問題：

- 從攝影機裝置接收單一視訊串流
- 處理串流並將其轉送給多個檢視器
- 處理多參與者對話的音訊混合
- 保留邊緣裝置運算和頻寬容量

Multiviewer 對於使用案例特別有用，包括：

- 智慧家庭安全：多個家庭成員可以同時檢視攝影機摘要，而不會降低效能
- 企業安全：安全團隊可以同時監控摘要
- 汽車監控：機群管理員和控制器可以同時檢視車輛攝影機
- 機器人和無人機：多個操作員可以監控自動化系統
- 教育/執行：多個執行者可以觀察測試工作階段
- 遠距醫療：醫療保健團隊可以參與遠端諮詢

需求和資源

若要使用 Multiviewer，您需要下列資源：

- Kinesis Video Streams Streams：擷取和儲存影片和音訊內容的目的地
- WebRTC 訊號頻道：使用 KVS WebRTC SDK 啟用與裝置的連線
- 媒體儲存組態：使用 UpdateMediaStorageConfiguration API 連結串流和頻道

> **Important**
>
> 多檢視程式目前只有在與 WebRTC 擷取結合時才可用。主要或檢視器都可以啟動 WebRTC 擷取工作階段，而視訊和音訊軌會同時存放在 Kinesis Video Streams Stream 中，同時分發給多個檢視器。

裝置需求：

- 攝影機裝置必須支援 KVS WebRTC SDK
- 檢視器應用程式必須使用 KVS WebRTC SDK
- 所有參與者都連接到相同的訊號頻道

追蹤要求：

- 主要參與者：WebRTC 擷取需要音訊和視訊音軌
- 觀眾參與者：可以傳送選用的音軌或完全不傳送音軌。觀眾無法傳送影片音軌

設定多檢視程式

請依照下列步驟為您的應用程式設定 Multiviewer：

1. 建立必要資源

   使用主控台、CLI 或 SDKs 建立 Kinesis Video Streams Streams 和 WebRTC 訊號頻道。如需詳細說明the section called “建立訊號頻道”，請參閱 the section called “建立影片串流”和 。

2. 連結資源

使用 UpdateMediaStorageConfiguration API 來連結您的串流和頻道。此組態會啟用 Multiviewer 功能。如需實作詳細資訊the section called “設定目的地”，請參閱 。

3. 設定攝影機應用程式

   使用 KVS WebRTC SDK 實作攝影機應用程式，以呼叫 JoinStorageSession API。這會啟動其他檢視器可以加入的擷取工作階段。

4. 設定檢視器應用程式

   使用 KVS WebRTC 開發套件實作檢視器應用程式，以呼叫 JoinStorageSessionAsViewer API。多個檢視器可以同時加入相同的工作階段。

與擷取整合

Multiviewer 建置在 WebRTC 擷取功能之上，該功能於 2023 年推出。此整合提供數種優點：

- 自動錄製：所有多檢視程式工作階段都會自動錄製到 Kinesis Video Streams Streams，以供日後播放和分析
- 雲端處理：影片處理會在雲端進行，減少邊緣裝置上的運算負載
- 可擴展架構：以雲端為基礎的方法可以處理多個並行多檢視器工作階段
- 一致的體驗：檢視器無論其網路條件為何，都會收到相同的高品質串流

具有擷取的一般多檢視程式工作階段工作流程為：

1. 攝影機裝置呼叫 JoinStorageSession 以開始將視訊擷取至雲端
2. 影片串流會在設定的 Kinesis Video Streams 串流中處理和存放
3. 多個檢視器裝置呼叫 JoinStorageSessionAsViewer 以加入工作階段
4. 雲端混音器會將視訊串流分發給所有連線的檢視器
5. 所有參與者的音訊都已適當混合和分配

API 操作

Multiviewer 使用與 WebRTC 擷取相同的 API 操作。金鑰 APIs包括：

- UpdateMediaStorageConfiguration - 將訊號頻道連結至串流以進行擷取

- JoinStorageSession - 從攝影機裝置啟動擷取工作階段
- JoinStorageSessionAsViewer - 允許檢視器加入作用中的擷取工作階段
- DescribeMediaStorageConfiguration - 擷取目前的媒體儲存組態

如需詳細的 API 使用範例，請參閱 the section called “使用儲存工作階段建立 WebRTC 連線”。

最佳實務

實作 Multiviewer 時，請遵循下列最佳實務：

- 邊緣裝置最佳化：當您需要超過 2 到 3 個並行檢視器時，請特別使用 Multiviewer，因為這是邊緣裝置限制通常會明顯顯現的地方
- 監控工作階段運作狀態：實作監控以追蹤工作階段品質和檢視器連線
- 處理連線失敗：為攝影機和檢視器連線實作重試邏輯
- 音訊管理：考慮預設將檢視器靜音，以防止大型工作階段中的音訊回饋
- 資源清除：確保檢視器離開工作階段時 WebRTC 連線的正確清除

安全

的雲端安全性 AWS 是最高優先順序。身為 AWS 客戶，您將受益於資料中心和網路架構，該架構旨在滿足最安全敏感組織的需求。

安全性是 AWS 與您之間的共同責任。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。第三方稽核人員定期檢測及驗證安全的效率也是我們 AWS 合規計劃的一部分。如要了解適用於 Kinesis Video Streams 的合規計劃，請參閱 合規計劃的AWS 服務範圍。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。您也必須對資料敏感度、組織要求，以及適用法律和法規等其他因素負責。

本文件可協助您了解如何在搭配 WebRTC 使用 Amazon Kinesis Video Streams 時套用共同責任模型。下列主題說明如何使用 WebRTC 設定 Amazon Kinesis Video Streams，以符合您的安全與合規目標。您也將了解如何使用其他 AWS 服務，協助您透過 WebRTC 資源監控和保護 Amazon Kinesis Video Streams。

主題

使用 控制對 Amazon Kinesis Video Streams with WebRTC 資源的存取 AWS Identity and Access Management

透過搭配 Amazon Kinesis Video Streams with WebRTC 使用 AWS Identity and Access Management (IAM)，您可以控制組織中的使用者是否可以搭配 WebRTC API 操作使用特定 Kinesis Video Streams 來執行任務，以及是否可以使用特定 AWS 資源。

如需 IAM 的詳細資訊，請參閱下列各項：

- AWS Identity and Access Management (IAM)
- IAM 入門
- IAM 使用者指南

目錄

政策語法

IAM 政策為包含一或多個陳述式的 JSON 文件。每個陳述式的結構如下所示：

```
{
  "Statement":[{
    "Effect":"effect",
    "Action":"action",
    "Resource":"arn",
    "Condition":{
      "condition":{
        "key":"value"
        }
      }
    }
  ]
}
```

陳述式由各種元素組成：

- Effect (效果)：效果 可以是 `Allow` 或 `Deny`。根據預設，IAM 使用者沒有使用資源和 API 動作的許可，因此所有請求均會遭到拒絕。明確允許覆寫預設值。明確拒絕覆寫任何允許。
- Action (動作)：動作 是您授予或拒絕許可的特定 API 動作。

- Resource (資源)：受動作影響的資源。若要在陳述式中指定資源，您必須使用其 Amazon Resource Name (ARN)。
- Condition (條件)：條件為選擇性。您可以使用它們來控制何時政策開始生效。

當您建立和管理 IAM 政策時，可能想要使用 IAM 政策產生器和 IAM 政策模擬器。

API 動作

在 IAM 政策陳述式中，您可以從任何支援 IAM 的服務指定任何 API 動作。對於 Kinesis Video Streams with WebRTC，請使用下列字首搭配 API 動作的名稱：`kinesisvideo:`。例如：`kinesisvideo:CreateSignalingChannel`、`kinesisvideo:ListSignalingChannels` 和 `kinesisvideo:DescribeSignalingChannel`。

若要在單一陳述式中指定多個動作，請用逗號分隔，如下所示：

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

您也可以使用萬用字元指定多個動作。例如，您可以指定名稱開頭有「Get」文字的所有動作，如下所示：

```
"Action": "kinesisvideo:Get*"
```

若要指定所有 Kinesis Video Streams 操作，請使用星號 (*) 萬用字元，如下所示：

```
"Action": "kinesisvideo:*"
```

如需 Kinesis Video Streams API 動作的完整清單，請參閱 Kinesis Video Streams API 參考。

Amazon Resource Name (ARN)Amazon Resource Name (ARN)Amazon Resource Name (ARN)Amazon Resource Name (ARN)Amazon Resource Name (ARN)Amazon Resource Name (ARN)Amazon Resource Name (ARN)Amazon Resource Name (ARN)

每個 IAM 政策陳述式都會套用到您使用其 ARN 指定的資源。

針對 Kinesis Video Streams 使用以下 ARN 資源格式：

```
arn:aws:kinesisvideo:region:account-id:channel/channel-name/code
```

例如：

```
"Resource": arn:aws:kinesisvideo::*:111122223333:channel/my-channel/0123456789012
```

您可以使用 DescribeSignalingChannel 取得頻道的 ARN。

授予其他 IAM 帳戶對 Kinesis 影片串流的存取權

您可能需要將許可授予其他 IAM 帳戶，才能對具有 WebRTC 訊號頻道的 Kinesis Video Streams 執行操作。服務角色是服務擔任的 IAM 角色，可代您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的建立角色以委派許可給 AWS 服務。

政策範例

下列範例政策示範如何使用 WebRTC 頻道控制使用者對 Kinesis Video Streams 的存取。

Example 1：允許使用者從任何訊號頻道取得資料

此政策可讓使用者或群組在任何訊號頻道上執行 `DescribeSignalingChannel`、`GetSignalingChannelEndpoint`、`ListSignalingChannels` 和 `ListTagsForResource` 操作。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kinesisvideo:Describe*",
                "kinesisvideo:Get*",
                "kinesisvideo:List*"
            ],
            "Resource": "*"
        }
    ]
}
```

Example 2：允許使用者建立訊號頻道

此政策可讓使用者或群組執行 `CreateSignalingChannel` 操作。

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kinesisvideo:CreateSignalingChannel"
            ],
            "Resource": "*"
        }
    ]
}
```

Example 3：允許使用者完整存取所有 Kinesis Video Streams 和 Kinesis Video Streams with WebRTC 資源

此政策允許使用者或群組在任何資源上執行任何 Kinesis Video Streams 操作。此政策適用於管理員。

JSON

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

Example 4：允許使用者從特定訊號頻道取得資料

此政策允許使用者或群組從特定訊號頻道取得資料。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "kinesisvideo:DescribeSignalingChannel",
            "Resource": "arn:aws:kinesisvideo:us-west-2:123456789012:channel/
channel_name/0123456789012"
        }
    ]
}
```

Amazon Kinesis Video Streams with WebRTC 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。

Amazon Kinesis Video Streams with WebRTC 中的彈性

AWS 全球基礎設施是以 AWS 區域 和 可用區域為基礎建置。 AWS 區域 提供多個實體分隔和隔離的可用區域，這些可用區域與低延遲、高輸送量和高備援聯網連接。您可以使用可用區域來設計和操作應用程式和資料庫，這些應用程式和資料庫會在可用區域之間自動容錯移轉，而不會中斷。可用區域的可用性、容錯能力和擴展能力，均較單一或多個資料中心的傳統基礎設施還高。

如需 AWS 區域 和可用區域的詳細資訊，請參閱 AWS 全球基礎設施。

Kinesis Video Streams with WebRTC 中的基礎設施安全性

作為受管服務，Kinesis Video Streams （包括其 WebRTC 功能） 受到 Amazon Web Services：安全程序概觀白皮書中所述 AWS 的全球網路安全程序的保護。

您可以使用 AWS 發佈的 API 呼叫，透過網路存取 Kinesis Video Streams。用戶端必須支援 Transport Layer Security (TLS) 1.2 或更新版本。建議使用 TLS 1.3 或更新版本。用戶端也必須支援具備完美轉送私密 (PFS) 的密碼套件，例如臨時 Diffie-Hellman (DHE) 或橢圓曲線臨時 Diffie-Hellman (ECDHE)。現代系統 (如 Java 7 和更新版本) 大多會支援這些模式。

此外，請求必須使用存取金鑰 ID 和與 IAM 主體相關聯的私密存取金鑰來簽署。或者，您可以使用 AWS Security Token Service (AWS STS) 以產生暫時安全憑證以簽署請求。

Amazon Kinesis Video Streams with WebRTC 的安全最佳實務

Amazon Kinesis Video Streams （包括其 WebRTC 功能） 提供許多安全功能，供您在開發和實作自己的安全政策時考慮。以下最佳實務為一般準則，並不代表完整的安全解決方案。這些最佳實務可能不適用或無法滿足您的環境需求，因此請將其視為實用建議就好，而不要當作是指示。

如需您遠端裝置的安全最佳實務，請參閱裝置代理程式的安全最佳實務。

實作最低權限存取

授予許可時，您可以決定誰要取得哪些 Kinesis Video Streams 資源的許可。您還需針對這些資源啟用允許執行的動作，因此，您只應授與執行任務所需的許可。對降低錯誤或惡意意圖所引起的安全風險和影響而言，實作最低權限存取是相當重要的一環。

例如，將資料傳送至 Kinesis Video Streams 的生產者只需要 `PutMedia`、`GetStreamingEndpoint`和 `DescribeStream`。請勿授予生產者應用程式所有動作 (*) 或其他動作 (例如 `GetMedia`) 的許可。

如需詳細資訊，請參閱套用最低權限許可。

使用 IAM 角色

製作者和用戶端應用程式必須具有有效的登入資料，才能存取 Kinesis 影片串流。您不應將 AWS 登入資料直接存放在用戶端應用程式或 Amazon S3 儲存貯體中。這些是不會自動輪換的長期憑證，如果遭到盜用，可能會對業務造成嚴重的影響。

反之，您應該使用 IAM 角色來管理生產者和用戶端應用程式的臨時登入資料，以存取 Kinesis 影片串流。使用角色時，您不必使用長期憑證來存取其他資源。

如需詳細資訊，請參閱《IAM 使用者指南》中的下列主題：

- IAM 角色

- 角色的常見案例：使用者、應用程式和服務

使用 CloudTrail 監控 API 呼叫

Kinesis Video Streams with WebRTC 已與 整合 AWS CloudTrail，此服務可提供使用者、角色或 Kinesis Video Streams with WebRTC 中 AWS 服務所採取之動作的記錄。

您可以使用 CloudTrail 所收集的資訊，判斷使用 WebRTC 對 Kinesis Video Streams 提出的請求、提出請求的 IP 地址、提出請求的人員、提出請求的時間，以及其他詳細資訊。

如需詳細資訊，請參閱the section called “使用 CloudTrail 記錄 API 呼叫”。

WebRTC 加密

端對端加密是 Amazon Kinesis Video Streams with WebRTC 的強制性功能，Kinesis Video Streams 會在所有元件上強制執行此功能，包括訊號和媒體或資料串流。無論通訊是peer-to-peer還是透過 Kinesis Video Streams TURN 端點轉送，所有 WebRTC 通訊都會透過標準化加密通訊協定安全地加密。

使用安全 Websocket (WSS) 交換訊號訊息、使用 Datagram Transport Layer Security (DTLS) 加密資料串流，以及使用安全即時傳輸通訊協定 (SRTP) 加密媒體串流。

監控指標和 API 呼叫

監控是使用 WebRTC 和您的 AWS 解決方案維護 Amazon Kinesis Video Streams 可靠性、可用性和效能的重要部分。您應該從 AWS 解決方案的所有部分收集監控資料，以便在發生多點失敗時更輕鬆地偵錯。不過，在開始使用 WebRTC 監控 Kinesis Video Streams 之前，您應該建立監控計畫，其中包含下列問題的答案：

- 監控目標是什麼？
- 要監控哪些資源？
- 監控這些資源的頻率為何？
- 要使用哪些監控工具？
- 誰將執行監控任務？
- 發生問題時應該通知誰？

在您定義監控目標並建立監控計劃之後，下一步是為環境中的 Kinesis Video Streams with WebRTC 正常效能建立基準。您應該在不同的時間和不同的負載條件下測量 Kinesis Video Streams with WebRTC 效能。當您監控 Kinesis Video Streams with WebRTC 時，您已收集的監控資料應該保存一份歷史記錄。您可以比較目前的 Kinesis Video Streams with WebRTC 效能資料與歷史資料，協助您辨別正常效能模式和效能異常狀況，並設法解決可能發生的問題。

主題

使用 WebRTC 監控 Kinesis 影片串流

您可以使用 Amazon CloudWatch 監控 Amazon Kinesis Video Streams with WebRTC，該功能會將 Amazon Kinesis Video Streams with WebRTC 的原始資料收集並處理為可讀且幾近即時的指標。Amazon CloudWatch 這些統計資料會記錄 15 個月的時間，以便您存取歷史資訊，並更清楚 Web 應用程式或服務的執行效能。

Amazon Kinesis Video Streams 提供下列指標：

主題

訊號指標

本節提供如何使用 CloudWatch Logs 監控和疑難排解訊號相關問題的相關資訊。

指標名稱	Description	單位	維度
失敗	如果維度中提及的操作傳回 200 狀態碼回應，則會發出 '0'。否則為 '1'。	計數	Operation、SignalingChannelName
Latency (延遲)	測量服務接收請求並傳回回應所需的時間。	毫秒	Operation、SignalingChannelName
MessagesTransferred.Count	頻道傳送和接收的訊息總數。	計數	SignalingChannelName

`Operation` 維度適用於下列 APIs：

- ConnectAsMaster
- ConnectAsViewer
- SendSdpOffer
- SendSdpAnswer
- SendCandidate
- SendAlexaOfferToMaster
- GetIceServerConfig
- 中斷連線

TURN 指標

本節提供如何使用 CloudWatch Logs 監控和疑難排解 TURN 相關問題的相關資訊。

指標名稱	Description	單位	維度
TURNConnectedMinutes	對於一分鐘內用來串流資料的每個 TURN 配置，發出 '1'。	計數	SignalingChannelName

WebRTC 擷取指標

本節提供如何使用 CloudWatch Logs 監控和疑難排解 WebRTC 擷取相關問題的相關資訊。

指標名稱	Description	單位	維度
失敗	如果維度中提及的操作傳回 200 狀態碼回應，則會發出 '0'。否則為 '1'。	計數	Operation、SignalingChannelName
Latency (延遲)	測量服務接收請求並傳回回應所需的時間。	毫秒	Operation、SignalingChannelName
TotalBitrate	傳送的總位元速率。如果指定角色，則表示主參與者傳送的總位元速率，或 VIEWER 參與者傳送的彙總總位元速率。	bps	操作、SignalingChannelName、角色
TotalPacketCount	傳送的總封包計數。如果指定角色，則表示主參與者傳送的總位元速率，或 VIEWER 參與者傳送的彙總總位元速率。	計數	操作、SignalingChannelName、角色
WebRTCRecordingMinutes	頻道發生的 WebRTCRecordingMinute 數量。	計數	操作、SignalingChannelName、角色

指標名稱	Description	單位	維度
WebRTCViewerMinutes	頻道發生的 WebRTCViewerMinute 數量。	計數	操作、SignalingChannelName、角色

`Operation` 維度適用於下列 APIs：

- JoinStorageSession
- JoinStorageSessionAsViewer

`Role` 維度：

- 主要
- 檢視程式

使用 記錄 API 呼叫 AWS CloudTrail

Amazon Kinesis Video Streams with WebRTC 已與 服務整合 AWS CloudTrail，此服務提供使用者、角色或 AWS 服務在 Amazon Kinesis Video Streams with WebRTC 中採取的動作記錄。CloudTrail 會將 Amazon Kinesis Video Streams with WebRTC 的所有 API 呼叫擷取為事件。擷取的呼叫包括來自 Amazon Kinesis Video Streams 主控台的呼叫，以及程式碼對 Amazon Kinesis Video Streams with WebRTC API 操作的呼叫。如果您建立線索，則可以將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括使用 WebRTC 的 Amazon Kinesis Video Streams 事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台中的事件歷史記錄檢視最新事件。您可以使用 CloudTrail 收集的資訊，判斷透過 WebRTC 向 Amazon Kinesis Video Streams 提出的請求、提出請求的 IP 地址、提出請求的人員、提出請求的時間，以及其他詳細資訊。

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

Amazon Kinesis Video Streams with WebRTC and CloudTrail

當您建立 AWS 帳戶時，會在您的帳戶上啟用 CloudTrail。當 Amazon Kinesis Video Streams with WebRTC 中發生支援的事件活動時，該活動會與事件歷史記錄中的其他 AWS 服務事件一起記錄在 CloudTrail 事件中。您可以在 AWS 帳戶中檢視、搜尋和下載最近的事件。如需詳細資訊，請參閱《使

用 CloudTrail 事件歷史記錄檢視事件》https://docs.aws.amazon.com/awscloudtrail/latest/userguide/view-cloudtrail-events.html。

若要持續記錄您 AWS 帳戶中的事件，包括 Amazon Kinesis Video Streams with WebRTC 的事件，請建立追蹤。線索能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。依預設，當您在主控台中建立追蹤時，該追蹤會套用至所有的 AWS 區域。線索會記錄 AWS 分割區中所有區域的事件，並將日誌檔案傳送到您指定的 Amazon S3 儲存貯體。此外，您可以設定其他 AWS 服務，以進一步分析和處理 CloudTrail 日誌中所收集的事件資料。如需詳細資訊，請參閱下列內容：

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

Amazon Kinesis Video Streams with WebRTC 支援將下列動作記錄為 CloudTrail 日誌檔案中的事件：

- CreateSignalingChannel
- DeleteSignalingChannel
- DescribeSignalingChannel
- GetSignalingChannelEndpoint
- ListSignalingChannels
- ListTagsForResource
- TagResource
- UntagResource
- UpdateSignalingChannel

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

- 是否使用根或 AWS Identity and Access Management (IAM) 使用者登入資料提出請求。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 請求是否由其他 AWS 服務提出。

如需詳細資訊，請參閱 CloudTrail userIdentity 元素。

範例：日誌檔案項目

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

下列範例顯示示範 CreateSignalingChannel 動作的 CloudTrail 日誌項目。

```
{
   "eventVersion":"1.05",
   "userIdentity":{
      "type":"IAMUser",
      "principalId":"EX_PRINCIPAL_ID",
      "arn":"arn:aws:iam::123456789012:user/Alice",
      "accountId":"123456789012",
      "accessKeyId":"EXAMPLE_KEY_ID",
      "userName":"Alice"
   },
   "eventTime":"2019-11-19T22:49:04Z",
   "eventSource":"kinesisvideo.amazonaws.com",
   "eventName":"CreateSignalingChannel",
   "awsRegion":"us-west-2",
   "sourceIPAddress":"127.0.0.1",
   "userAgent":"aws-sdk-java/unknown-version Linux/x.xx",
   "requestParameters":{
      "channelName":"YourChannelName"
   },
   "responseElements":{
      "channelARN":"arn:aws:kinesisvideo:us-west-2:123456789012:channel/
YourChannelName/1574203743620"
   },
   "requestID":"df3c99c4-1d97-49da-8569-7de6c92b4856",
   "eventID":"bb74bac2-964c-49b0-903a-3501c6bde632"
}
```

API 參考

主題

WebSocket APIs

以下是 Amazon Kinesis Video Streams with WebRTC WebSocket APIs：

主題

ConnectAsMaster

以主節點身分連線至端點指定的訊號頻道。任何 WebSocket-complaint 程式庫都可用來連線到從 `GetSignalingChannelEndpoint` API 呼叫取得的安全 Websocket (WSS) 端點。必須提供訊號頻道的 Amazon Resource Name (ARN) 作為查詢字串參數。有個別端點可供以主節點和檢視器身分來連線。如果多個用戶端以主節點身分連接到特定頻道，則最新請求優先。新的連線中繼資料會覆寫現有的連線中繼資料。

請求

```
"X-Amz-ChannelARN": "string"
```

- X-Amz-ChannelARN - 訊號頻道的 ARN。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 1024。
  - 模式：`arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`
  - 必要：是

回應

200 OK HTTP 狀態碼和空白內文。

錯誤

- InvalidArgumentException

  指定的參數超過其限制、不支援或無法使用。如需詳細資訊，請參閱傳回的訊息。

  HTTP 狀態碼：400
- AccessDeniedException

  發起人未獲授權存取指定的通道或符記已過期。

  HTTP 狀態碼：403
- ResourceNotFoundException

  頻道不存在。

  HTTP 狀態碼：404
- ClientLimitExceededException

  以太高的速率叫用 API 時。如需詳細資訊，請參閱 Amazon Kinesis Video Streams with WebRTC 服務配額 和錯誤重試和指數退避。 AWS

  HTTP 狀態碼：400

限制/節流

如果以太高的速率叫用 API，則會在帳戶層級節流此 API。節流時會以 `ClientLimitExceededException` 傳回錯誤。

等冪

如果指定的 clientId 和頻道已存在連線，則會以新資訊更新連線中繼資料。

重試行為

這視為新的 API 呼叫。

並行呼叫

允許並行呼叫，每次呼叫都會更新連線中繼資料。

ConnectAsViewer

以檢視器身分連線至端點所指定的訊號頻道。任何符合 WebSocket 的程式庫都可以用來連線到從 `GetSignalingChannelEndpoint` API 呼叫取得的安全 Websocket (WSS) 端點。必須提供訊號頻道的 Amazon Resource Name (ARN) 和用戶端 ID 作為查詢字串參數。有個別端點可供以主節點和檢視器身分來連線。如果現有的連線與請求中指定的連線`ClientId`相同，則以新連線為優先。新的資訊會覆寫連線中繼資料。

請求

```
"X-Amz-ChannelARN": "string",
"X-Amz-ClientId": "string"
```

- X-Amz-ChannelARN - 訊號頻道的 ARN。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 1024。
  - 模式：`arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`
  - 必要：是
- X-Amz-ClientId - 用戶端的唯一識別符。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：`^(?!(?i)AWS_.*)[a-zA-Z0-9_.-]`

    > **Note**
    >
    > `X-Amz-ClientId` 無法以 開頭`AWS_`。

  - 必要：是

回應

200 OK HTTP 狀態碼和空白內文。

錯誤

- InvalidArgumentException

  指定的參數超過其限制、不支援或無法使用。如需詳細資訊，請參閱傳回的訊息。

  HTTP 狀態碼：400
- AccessDeniedException

  發起人未獲授權存取指定的通道或符記已過期。

  HTTP 狀態碼：403
- ResourceNotFoundException

  頻道不存在。

  HTTP 狀態碼：404
- ClientLimitExceededException

  以太高的速率叫用 API 時，或連線至頻道的檢視器數目超出支援的上限時。如需詳細資訊，請參閱 Amazon Kinesis Video Streams with WebRTC 服務配額 和錯誤重試和指數退避。 AWS

  HTTP 狀態碼：400

限制/節流

如果以太高的速率叫用 API，或連線至頻道的檢視器數目超出支援的上限時，則會在帳戶層級節流此 API。節流時會以 `ClientLimitExceededException` 傳回錯誤。

等冪

如果指定的 `ClientId`和 頻道已存在連線，則會使用新資訊更新連線中繼資料。

重試行為

這視為新的 API 呼叫。

並行呼叫

允許並行呼叫，每次呼叫都會更新連線中繼資料。

非同步訊息接收

所有回應訊息都視為事件非同步傳遞至收件者 (例如，SDP 提議或 SDP 回答傳遞)。以下是事件訊息結構。

事件

```
{
    "senderClientId": "string",
    "messageType": "string",
    "messagePayload": "string",
    "statusResponse": {
        "correlationId": "string",
        "errorType": "string",
        "statusCode": "string",
        "description": "string"
    }
}
```

- senderClientId - 傳送者用戶端的唯一識別符。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必要：否
- messageType - 事件的類型。
  - 類型：ENUM
  - 有效類型：SDP_OFFER、SDP_ANSWER、ICE_CANDIDATE、GO_AWAY、RECONNECT_ICE_SERVER、STATUS_RESPONSE
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必要：是
- messagePayload - base64 編碼的訊息內容。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 10K。
  - 必要：否

- correlationId - 狀態所指的訊息的唯一識別符。這是用戶端訊息 (例如，SDP 提議、SDP 回答或 ICE 候選項) 中提供的相同 correlationId。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必要：是
- errorType - 錯誤的唯一識別名稱。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必要：否
- statusCode - 對應於回應性質的 HTTP 狀態碼。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必要：否
- description - 解釋狀態的字串描述。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 1K。
  - 必要：否

SendSdpOffer

將提議傳送給目標收件者。先決條件是用戶端必須已連線到從 `GetSignalingChannelEndpoint` API 取得的 WebSocket 端點。

如果傳送者類型是檢視器，則將提議傳送給主節點。此外，不需要指定 `RecipientClientId`，系統會忽略 `RecipientClientId` 的任何指定值。如果傳送者類型是主節點，則提議會傳送給 `RecipientClientId` 指定的目標檢視器。在這種情況下，`RecipientClientId` 是必要輸入。

允許主節點用戶端應用程式將提議傳送給任何檢視器，但只允許檢視器用戶端應用程式將提議傳送給主節點用戶端應用程式。如果檢視器用戶端應用程式嘗試將提議傳送給另一個檢視器用戶端應用程式，則「不會」執行該請求。如果同一個用戶端有尚未傳遞的未完成提議，則會以新的提議覆寫此提議。

請求

```
{
    "action": "SDP_OFFER",
    "recipientClientId": "string",
    "messagePayload": "string",
    "correlationId": "string"
}
```

- action - 傳送的訊息類型。
  - 類型：ENUM
  - 有效值：SDP_OFFER、SDP_ANSWER、ICE_CANDIDATE
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：[a-zA-Z0-9_.-]+
  - 必要：是
- recipientClientId - 收件者的唯一識別符。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：[a-zA-Z0-9_.-]+
  - 必要：是
- messagePayload - base-64 編碼的訊息內容。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 10K。
  - 必要：是
- correlationId - 訊息的唯一識別符。這是選擇性的參數。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：[a-zA-Z0-9_.-]+
  - 必要：否

回應

如果訊號後端成功接收訊息，則不會傳回任何回應。如果服務遇到錯誤，且請求中指定 `correlationId`，則會以 `STATUS_RESPONSE` 訊息形式傳回錯誤詳細資訊。如需詳細資訊，請參閱the section called “非同步訊息接收”。

錯誤

- InvalidArgumentException

  指定的參數超過其限制、不支援或無法使用。如需詳細資訊，請參閱傳回的訊息。

  HTTP 狀態碼：400
- ClientLimitExceededException

  以太高的速率叫用 API 時。如需詳細資訊，請參閱 Amazon Kinesis Video Streams with WebRTC 服務配額 和錯誤重試和指數退避。 AWS

  HTTP 狀態碼：400

限制/節流

如果以太高的速率叫用 API，則會在帳戶層級節流此 API。節流時會以 `ClientLimitExceededException` 傳回錯誤。

等冪

此 API 不是等冪。

重試行為

這視為新的 API 呼叫。

並行呼叫

允許並行呼叫。每次呼叫會傳送一次提議。

SendSdpAnswer

將回答傳送給目標收件者。先決條件是用戶端必須已連線到從 `GetSignalingChannelEndpoint` API 取得的 WebSocket 端點。

如果傳送者類型是檢視器，則將回答傳送給主節點。此外，不需要指定 RecipientClientId，系統會忽略 RecipientClientId 的任何指定值。如果傳送者類型是主節點，則回答會傳送給 RecipientClientId 指定的目標檢視器。在這種情況下，RecipientClientId 是必要輸入。

允許主節點用戶端應用程式將回答傳送給任何檢視器，但只允許檢視器用戶端應用程式將回答傳送給主節點用戶端應用程式。如果檢視器用戶端應用程式嘗試將回答傳送給另一個檢視器用戶端應用程式，則「不會」執行該請求。如果同一個用戶端有尚未傳遞的未完成回答，則會以新的回答覆寫此回答。

請求

```
{
    "action": "SDP_ANSWER",
    "recipientClientId": "string",
    "messagePayload": "string",
    "correlationId": "string"
}
```

- action - 傳送的訊息類型。
  - 類型：ENUM
  - 有效值：SDP_OFFER、SDP_ANSWER、ICE_CANDIDATE
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：[a-zA-Z0-9_.-]+
  - 必要：是
- recipientClientId - 收件者的唯一識別符。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：[a-zA-Z0-9_.-]+
  - 必要：是
- messagePayload - base-64 編碼的訊息內容。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 10K。
  - 必要：是
- correlationId - 訊息的唯一識別符。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 256。

- 模式：[a-zA-Z0-9_.-]+
- 必要：否

回應

如果訊號後端成功接收訊息，則不會傳回任何回應。如果服務遇到錯誤，且請求中指定 correlationId，則會以 STATUS_RESPONSE 訊息形式傳回錯誤詳細資訊。如需詳細資訊，請參閱非同步訊息接收。

錯誤

- InvalidArgumentException

  指定的參數超過其限制、不支援或無法使用。如需詳細資訊，請參閱傳回的訊息。

  HTTP 狀態碼：400
- ClientLimitExceededException

  以太高的速率叫用 API 時傳回。如需詳細資訊，請參閱 Amazon Kinesis Video Streams with WebRTC 服務配額 和錯誤重試和指數退避。 AWS

  HTTP 狀態碼：400

限制/節流

如果以太高的速率叫用 API，則會在帳戶層級節流此 API。節流時會以 ClientLimitExceededException 傳回錯誤。

等冪

此 API 不是等冪。

重試行為

這視為新的 API 呼叫。

並行呼叫

允許並行呼叫。每次呼叫會傳送一次提議。

SendIceCandidate

將 ICE 候選項傳送給目標收件者。先決條件是用戶端必須已連線到從 `GetSignalingChannelEndpoint` API 取得的 WebSocket 端點。

如果傳送者類型是檢視器，則會將 ICE 候選項傳送給主節點。此外，不需要指定 `RecipientClientId`，系統會忽略 `RecipientClientId` 的任何指定值。如果寄件者類型是主節點，ICE 候選項目會傳送至 指定的目標`RecipientClientId`。在此情況下， `RecipientClientId` 是必要的輸入。

允許主節點用戶端應用程式將 ICE 候選項傳送給任何檢視器，但只允許檢視器用戶端應用程式將 ICE 候選項傳送給主節點用戶端應用程式。如果檢視器用戶端應用程式嘗試將 ICE 候選項傳送給另一個檢視器用戶端應用程式，則「不會」執行該請求。

請求

```
{
    "action": "ICE_CANDIDATE",
    "recipientClientId": "string",
    "messagePayload": "string",
    "correlationId": "string"
}
```

- action - 傳送的訊息類型。
  - 類型：ENUM
  - 有效值：`SDP_OFFER`、`SDP_ANSWER`、`ICE_CANDIDATE`
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必要：是
- recipientClientId - 收件者的唯一識別符。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必要：否
- messagePayload - base64 編碼的訊息內容。
  - 類型：字串

  - 長度限制：長度下限為 1。長度上限為 10K。
  - 必要：是
- correlationId - 訊息的唯一識別符。
  - 類型：字串
  - 長度限制：長度下限為 1。長度上限為 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必要：否

回應

如果訊號後端成功接收訊息，則不會傳回任何回應。如果服務遇到錯誤，且請求中指定 `correlationId`，則會以 `STATUS_RESPONSE` 訊息形式傳回錯誤詳細資訊。如需詳細資訊，請參閱非同步訊息接收。

錯誤

- InvalidArgumentException

  指定的參數超過其限制、不支援或無法使用。如需詳細資訊，請參閱傳回的訊息。

  HTTP 狀態碼：400
- ClientLimitExceededException

  以太高的速率叫用 API 時。如需詳細資訊，請參閱 Amazon Kinesis Video Streams with WebRTC 服務配額 和錯誤重試和指數退避。 AWS

  HTTP 狀態碼：400

限制/節流

如果以太高的速率叫用 API，則會在帳戶層級節流此 API。節流時會以 `ClientLimitExceededException` 傳回錯誤。

等冪

此 API 不是等冪。

重試行為

這視為新的 API 呼叫。

並行呼叫

允許並行呼叫。每次呼叫會傳送一次提議。

中斷連線

用戶端可以隨時關閉連線。與 WebSocket 相容的程式庫支援關閉功能。當連線關閉時，服務會將用戶端標示為在特定訊號頻道中離線，而且不會嘗試傳遞任何訊息。在閒置連線逾時的情況下也採取相同的行為。

此服務也會傳送中斷連線指示給用戶端，例如在部署或伺服器維護期間。以下是定義的指示訊息：

- GO_AWAY：此訊息用來初始化連線關閉。可讓用戶端正常處理先前的消息、中斷連線，並在必要時重新連線至訊號頻道。
- RECONNECT_ICE_SERVER：此訊息用於初始化轉送連線關閉，還可讓用戶端正常中斷連線、取得新的 ICE 伺服器組態，並在必要時重新連線至轉送伺服器。

使用 WebRTC 對 Amazon Kinesis Video Streams 進行故障診斷

使用下列資訊來疑難排解使用 Amazon Kinesis Video Streams with WebRTC 時可能遇到的常見問題。

建立peer-to-peer工作階段的問題

WebRTC 可協助減輕因下列原因發生的問題：

- 網路位址轉譯 (NAT)
- 防火牆
- 對等之間的代理

WebRTC 提供架構，只要對等連線，即可協助交涉和維護連線。它還提供一種機制，用於透過TURN伺服器轉送媒體，以防無法交涉peer-to-peer連線。

鑑於建立連線所需的所有元件，請務必了解一些可用於協助疑難排解建立工作階段相關問題的工具。

主題

- 工作階段描述協定 (SDP) 優惠和答案
- 評估 ICE 候選項目產生
- 決定使用哪些候選項目來建立連線
- ICE 相關逾時

工作階段描述協定 (SDP) 優惠和答案

工作階段描述通訊協定 (SDP) 提供並回答會初始化對等之間的 RTC 工作階段。

若要進一步了解 SDP 通訊協定，請參閱 規格。

- 優惠是由「檢視者」所產生，他們想要連接到在 Kinesis Video Streams with WebRTC 中以「主要」身分連線至訊號頻道的對等。
- 答案由優惠的接收者產生。

優惠和答案都是在用戶端產生，雖然它們可能包含到目前為止已收集的 ICE 候選項目。

適用於 C 的 Kinesis Video Streams WebRTC 開發套件包含一個簡單的環境變數，您可以將其設定為記錄 SDP。這有助於了解收到的優惠和產生的答案。

若要stdout從 SDK 將 SDPs記錄到 ，請設定下列環境變數：export DEBUG_LOG_SDP=TRUE。您也可以使用 sdpOffer事件，在 JavaScript 型用戶端中記錄 SDP 優惠和答案。若要查看此示範，請參閱 GitHub。

如需其他資訊，請參閱 the section called “使用 WebRTC 監控 Kinesis 影片串流”。

如果未傳回 SDP 答案，則對等可能無法接受 SDP 優惠，因為優惠不包含任何相容的媒體轉碼器。您可能會看到類似以下的日誌：

```
I/webrtc_video_engine.cc: (line 808): SetSendParameters: {codecs:
 [VideoCodec[126:H264]], conference_mode: no, extensions: [], extmap-allow-mixed:
 false, max_bandwidth_bps: -1, mid: video1}
E/webrtc_video_engine.cc: (line 745): No video codecs supported.
E/peer_connection.cc: (line 6009): Failed to set remote video description send
 parameters for m-section with mid='video1'. (INVALID_PARAMETER)
E/peer_connection.cc: (line 3097): Failed to set remote offer sdp: Failed to set remote
 video description send parameters for m-section with mid='video1'.
E/KinesisVideoSdpObserver: onSetFailure(): Error=Failed to set remote offer sdp: Failed
 to set remote video description send parameters for m-section with mid='video1'.
D/KVSWebRtcActivity: Received SDP offer for client ID: null. Creating answer
E/peer_connection.cc: (line 2373): CreateAnswer: Session error code: ERROR_CONTENT.
 Session error description: Failed to set remote video description send parameters for
 m-section with mid='video1'..
E/KinesisVideoSdpObserver: onCreateFailure(): Error=Session error code: ERROR_CONTENT.
 Session error description: Failed to set remote video description send parameters for
 m-section with mid='video1'..
```

當您檢閱 SDP 優惠內容時，請尋找開頭為 的行a=rtpmap，以查看正在請求哪些媒體轉碼器。

```
...
a=rtpmap:126 H264/90000
...
a=rtpmap:111 opus/48000/2
...
```

如果您使用 Safari 做為檢視器連線至主要傳送 H.265 媒體，且您遇到下列情況：

- `InvalidAccessError: Failed to set remote answer sdp: Called with SDP without DTLS fingerprint.`
- `InvalidAccessError: Failed to set remote answer sdp: rtcp-mux must be enabled when BUNDLE is enabled.`

確認問題是由瀏覽器產生的 SDP 優惠所造成。在 SDP 優惠中，搜尋從 開始的行`a=rtpmap`，並檢查 H.265 行是否存在。它應該如下所示：

```
a=rtpmap:104 H265/90000
```

如果不存在，請在 Safari 設定中啟用適用於 WebRTC 的 H.265 轉碼器。

在 Safari 頂端導覽中，執行下列動作：

- 選取 Safari > 設定... > 進階。選取顯示 Web 開發人員的功能方塊。
- 選取特徵旗標。選取 WebRTC H265 轉碼器方塊。

重新啟動您的瀏覽器，讓變更生效。

評估 ICE 候選項目產生

ICE 候選項目是由對STUN伺服器進行呼叫的每個用戶端產生。對於使用 WebRTC 的 Kinesis Video Streams，STUN伺服器為 `stun:stun.kinesisvideo.{aws-region}.amazonaws.com:443`。

除了呼叫STUN伺服器以取得候選項目之外，用戶端通常也會呼叫TURN伺服器。他們會進行此呼叫，以便在無法建立直接peer-to-peer連線時使用轉送伺服器做為備用連線。

您可以使用下列工具來產生 ICE 候選項目：

- Trickle ICE WebRTC 範例，使用 Trickle ICE 來收集候選者
- IceTest.Info

使用這兩種工具。您可以輸入 STUN和 TURN 伺服器資訊來收集候選項目。

若要使用 WebRTC 取得 Kinesis Video Streams 的TURN伺服器資訊和必要的登入資料，您可以呼叫 GetIceServerConfig API 操作。

下列 AWS CLI 呼叫示範如何取得此資訊，以便在這兩個工具中使用。

```
export CHANNEL_ARN="YOUR_CHANNEL_ARN"

aws kinesisvideo get-signaling-channel-endpoint \
    --channel-arn $CHANNEL_ARN \
    --single-master-channel-endpoint-configuration Protocols=WSS,HTTPS,Role=MASTER
```

get-signaling-channel-endpoint 命令的輸出會傳回如下所示的回應：

```
{
  "ResourceEndpointList": [
    {
      "Protocol": "HTTPS",
      "ResourceEndpoint": "https://your-endpoint.kinesisvideo.us-east-1.amazonaws.com"
    },
    {
      "Protocol": "WSS",
      "ResourceEndpoint": "wss://your-endpoint.kinesisvideo.us-east-1.amazonaws.com"
    }
  ]
}
```

使用 HTTPS ResourceEndpoint值來取得TURN伺服器清單，如下所示：

```
export ENDPOINT_URL="https://your-endpoint.kinesisvideo.us-east-1.amazonaws.com"

aws kinesis-video-signaling get-ice-server-config \
    --channel-arn $CHANNEL_ARN \
    --service TURN \
    --client-id my-amazing-client \
    --endpoint-url $ENDPOINT_URL
```

回應包含TURN伺服器詳細資訊，包括 TCP 和 UDP 的端點，以及存取它們所需的登入資料。

> **Note**
>
> 回應中的 TTL 值會決定這些登入資料有效的持續時間，以秒為單位。在 Trickle ICE WebRTC 範例或 IceTest.Info 中使用這些值，使用 Kinesis Video Streams 受管服務端點產生 ICE 候選項目。

決定使用哪些候選項目來建立連線

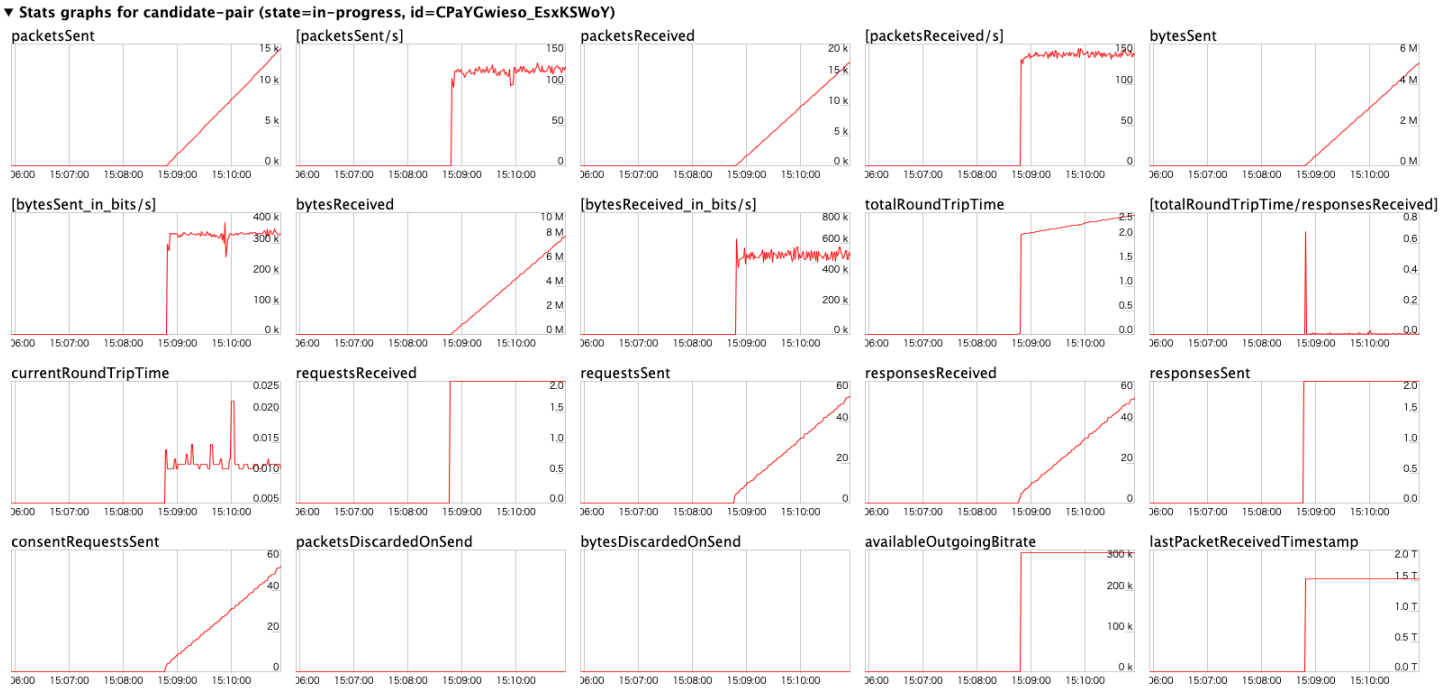
了解哪些候選者用於成功建立工作階段會很有幫助。如果您有以瀏覽器為基礎的用戶端執行已建立的工作階段，您可以使用內建的 webrtc-internals 公用程式，在 Google Chrome 中判斷此資訊。

在一個瀏覽器索引標籤中開啟 WebRTC 工作階段。

在另一個索引標籤中，開啟 `chrome://webrtc-internals/`。您可以在此索引標籤中檢視有關進行中工作階段的所有資訊。

您將看到已建立連線的相關資訊。例如：

您也可以確認已建立連線的指標，如下所示。



ICE 相關逾時

在 KvsRtcConfiguration 中為 ICE 設定預設逾時值。對於大多數使用者而言，預設值應該足夠，但您可能需要進行調整，以提高透過不良網路建立連線的機會。您可以在應用程式中設定這些預設值。

檢閱日誌的預設設定：

```
2024-01-08 19:43:44.433 INFO    iceAgentValidateKvsRtcConfig():
 iceLocalCandidateGatheringTimeout: 10000 ms
 iceConnectionCheckTimeout: 12000 ms
 iceCandidateNominationTimeout: 12000 ms
 iceConnectionCheckPollingInterval: 50 ms
```

如果您的網路品質不佳，並且想要提高連線機會，請嘗試調整下列值：

- `iceLocalCandidateGatheringTimeout` - 提高此逾時限制，以收集其他潛在候選者來嘗試連線。目標是嘗試所有可能的候選配對，因此如果您的網路狀況不佳，請提高此限制，以便有更多時間收集。

  例如，如果主機候選項目無法運作，且需要嘗試伺服器反射 (srflx) 或轉送候選項目，您可能需要增加此逾時。由於網路不佳，系統會緩慢地收集候選項目，而應用程式不想在此步驟上花費超過 20 秒。增加逾時可提供更多時間來收集潛在候選者以嘗試連線。

  > **Note**
  >
  > 我們建議此值小於 `iceCandidateNominationTimeout`，因為 推薦步驟需要時間來處理新的候選項目。

- `iceConnectionCheckTimeout` - 在不穩定或緩慢的網路中增加此逾時，其中封包交換和繫結請求/回應需要時間。增加此逾時允許至少一個候選配對嘗試由其他對等進行提名。
- `iceCandidateNominationTimeout` - 增加此逾時，以確保嘗試與本機轉送候選項目的候選配對。

  例如，如果收集第一個本機轉送候選項目大約需要 15 秒，請將逾時設定為超過 15 秒的值，以確保嘗試與本機轉送候選項目配對以取得成功。如果值設定為小於 15 秒，則 SDK 會在嘗試潛在候選配對時遺失，導致連線建立失敗。

  > **Note**
  >
  > 我們建議此值大於 `iceLocalCandidateGatheringTimeout`，以便產生效果。

- `iceConnectionCheckPollingInterval` - 此值預設為每個規格 50 毫秒。變更此值會變更連線檢查的頻率，基本上會變更 ICE 狀態機器轉換的頻率。

在具有良好系統資源的可靠、高效能網路設定中，您可以降低 值，以協助更快速建立連線。增加 值有助於降低網路負載，但建立連線的速度可能會變慢。

> **Important**
>
> 我們不建議變更此預設值。

Amazon Kinesis Video Streams with WebRTC 開發人員指南的文件歷史記錄

變更	描述	日期
修訂內容組織	已修訂 Amazon Kinesis Video Streams with WebRTC 開發人員指南中的配置和組織。	2024 年 9 月 18 日
初次出版	Amazon Kinesis Video Streams with WebRTC 開發人員指南初版。進一步了解	2019 年 11 月 4 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。