



Amazon Kinesis Video Streams WebRTC 开发人员指南

# Kinesis Video Streams



# Kinesis Video Streams: Amazon Kinesis Video Streams WebRTC 开发人员指南

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商标和商业外观不得用于任何非 Amazon 的商品或服务，也不得以任何可能引起客户混淆、贬低或诋毁 Amazon 的方式使用。所有非 Amazon 拥有的其他商标均为各自所有者的财产，这些所有者可能附属于 Amazon、与 Amazon 有关联或由 Amazon 赞助，也可能不是如此。

# Table of Contents

带有 WebRTC 的 Amazon Kinesis Video Streams 是什么？ .....	1
区域可用性 .....	2
工作方式 .....	3
重要概念 .....	3
技术概念 .....	4
STUN、TURN 和 ICE 是如何协同工作的 .....	5
组件 .....	6
系统要求 .....	7
联网要求 .....	7
网络环境 .....	7
调试正在进行的连接 .....	8
限额 .....	9
控制平面 API 服务配额 .....	10
信令 API 服务配额 .....	11
TURN 服务配额 .....	12
WebRTC 摄取服务配额 .....	12
故障排除 .....	13
使用 WebRTC 访问 Kinesis Video Streams .....	13
入门 .....	14
设置一个 AWS 账户 .....	14
注册获取 AWS 账户 .....	14
创建具有管理访问权限的用户 .....	15
创建 AWS 账户密钥 .....	16
创建信令通道 .....	16
快速入门：Ingenic T31 .....	18
下载代码 .....	18
设置构建环境 .....	19
使用 WebRTC 应用程序进行构建 .....	21
将应用程序上传到设备 .....	21
Connect 连接到设备 .....	21
将 micro SD 卡安装在主板上 .....	23
运行应用程序 .....	23
查看媒体 .....	24
故障排除 .....	24

直播媒体 (SDKs) .....	26
适用于嵌入式设备的 C SDK .....	26
下载 SDK .....	27
构建 SDK .....	27
运行 SDK 示例 .....	28
视频教程 .....	29
JavaScript 适用于 Web 应用程序的 SDK .....	29
安装 SDK .....	30
WebRTC 软件开发工具 JavaScript 包文档 .....	31
使用 WebRTC 示例应用程序 .....	31
编辑示例应用程序 .....	34
Android SDK .....	35
下载 SDK .....	35
构建 SDK .....	35
运行示例应用程序 .....	37
为软件开发工具包配置 Amazon Cognito .....	38
iOS SDK .....	41
下载 SDK .....	41
构建 SDK .....	42
运行示例应用程序 .....	44
为软件开发工具包配置 Amazon Cognito .....	45
C SDK 的客户端指标 .....	49
信令指标 .....	49
C SDK 支持 W3C 标准指标 .....	51
摄取和存储媒体 .....	66
支持多画面的 WebRTC 收录 (预览版) .....	66
API 操作 .....	67
什么是 WebRTC 的摄取和存储? .....	67
了解 WebRTC 的摄取和存储 .....	67
与存储会话建立 WebRTC 连接 .....	69
创建信令通道 .....	70
创建视频流 .....	72
授予权限 .....	74
配置目的地 .....	75
链接信令频道和直播 .....	75
取消信令频道和直播的关联 .....	77

提取媒体 .....	79
从浏览器提取媒体 .....	79
从 WebRTC C SDK 中提取媒体 .....	80
将观众添加到摄取会话中 .....	81
播放摄取的媒体 .....	83
在控制台中查看媒体 .....	83
使用 HLS 使用媒体数据 .....	83
使用示例媒体查看器应用程序查看媒体 .....	83
Connect 到存储会话 .....	84
对存储会话连接进行故障排除 .....	90
控制和控制同行 .....	90
查看支持的编解码器 .....	91
安全性 .....	92
使用 IAM 控制对资源的访问权限 .....	92
策略语法 .....	93
API 操作 .....	94
亚马逊资源名称 (ARNs) .....	94
向其他 IAM 账户授予访问 Kinesis 视频流的权限 .....	95
策略示例 .....	95
合规性验证 .....	97
恢复能力 .....	98
基础结构安全性 .....	98
安全最佳实践 .....	98
实施最低权限访问 .....	98
使用 IAM 角色 .....	99
CloudTrail 用于监控 API 调用 .....	99
WebRTC 加密 .....	99
监控指标和 API 调用 .....	100
使用 WebRTC 监控 Kinesis Video Streams .....	100
信令指标 .....	101
TURN 指标 .....	102
WebRTC 摄取指标 .....	102
使用记录 API 调用 CloudTrail .....	102
亚马逊 Kinesis Video 通过 WebRTC 直播 Kinesis Video Streams 和 CloudTrail .....	103
示例：日志文件条目 .....	104
API 参考 .....	106

WebSocket 终端节点 APIs .....	106
ConnectAsMaster .....	106
ConnectAsViewer .....	108
异步消息接收 .....	110
事件 .....	110
SendSdpOffer .....	111
SendSdpAnswer .....	113
SendIceCandidate .....	116
Disconnect .....	118
故障排除 .....	13
建立 peer-to-peer会话的问题 .....	119
会话描述协议 (SDP) 提议和应答 .....	119
评估 ICE 候选项的生成 .....	121
确定使用了哪些候选项来建立连接 .....	122
与冰有关的超时 .....	123
文档历史记录 .....	126
.....	CXXVII

# 带有 WebRTC 的 Amazon Kinesis Video Streams 是什么？

WebRTC 是一项开放技术规范，用于通过简单的方式实现浏览器和移动应用程序之间的实时通信（RTC）。APIs 使用对等技术在对等的点之间进行实时数据交换，并提供 human-to-human 交互所需的低延迟媒体流。WebRTC 规范包括一组 IETF 协议，[包括建立交互式连接、使用 NAT 周围的中继遍历（TURN）和用于建立连接的 NAT 会话遍历实用程序（STUN）](#)，以及用于 peer-to-peer 建立连接的用于可靠和安全的实时媒体和数据流的协议规范。

[Amazon Kinesis Video Streams](#) 提供符合标准的 WebRTC 实现作为完全托管的功能。您可以使用 Amazon Kinesis Video Streams with WebRTC 安全地进行媒体的实时流式传输，或在任何摄像头 IoT 设备与符合 WebRTC 的移动或 Web 播放器之间执行双向音频或视频交互。借助这项全面托管的功能，您不必构建、运营或扩展任何与 WebRTC 相关的云基础设施（例如信令或媒体中继服务器）便能安全地在应用程序和设备间流式传输媒体。

使用带有 WebRTC 的 Kinesis Video Streams，您可以轻松构建应用程序，用于 peer-to-peer 直播媒体流，或者针对各种用例在摄像机物联网设备、网络浏览器和移动设备之间进行实时音频或视频交互。这样的应用程序可以帮助父母关注婴儿的房间，使房主可以使用视频门铃来检查谁在门口，使支持摄像头的机器人吸尘器的所有者可以通过查看手机上的实时摄像头流来远程控制机器人，等等。

如果您是 Kinesis Video Streams with WebRTC 的新用户，建议您阅读以下内容：

- [工作方式](#)
- [亚马逊 Kinesis Video Streams 带有 WebRTC 开发工具包的 C 语言版本，适用于嵌入式设备](#)
- [带有 WebRTC 开发工具包的 Amazon Kinesis Video Streams 适用于网络应用程序 JavaScript](#)
- [适用于安卓的亚马逊 Kinesis Video Streams WebRTC SDK](#)
- [适用于 iOS 的亚马逊 Kinesis Video Streams WebRTC SDK](#)
- [控制层面 APIs](#)
- [数据平面 REST APIs](#)
- [数据平面网络套接字 APIs](#)

## 区域可用性

### Note

目前尚不支持带有 WebRTC 的 Amazon Kinesis Video Streams。AWS GovCloud (US) Region

Amazon Kinesis Video Streams with WebRTC 功能已在下列区域开放：

区域名称	AWS 地区代码
美国东部 ( 俄亥俄州 )	us-east-2
美国东部 ( 弗吉尼亚州北部 )	us-east-1
美国西部 ( 俄勒冈州 )	us-west-2
非洲 ( 开普敦 )	af-south-1
亚太地区 ( 香港 )	ap-east-1
亚太地区 ( 孟买 )	ap-south-1
亚太地区 ( 首尔 )	ap-northeast-2
亚太地区 ( 新加坡 )	ap-southeast-1
亚太地区 ( 悉尼 )	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
加拿大 ( 中部 )	ca-central-1
欧洲地区 ( 法兰克福 )	eu-central-1
欧洲地区 ( 爱尔兰 )	eu-west-1
欧洲地区 ( 伦敦 )	eu-west-2

区域名称	AWS 地区代码
欧洲地区 ( 巴黎 )	eu-west-3
南美洲 ( 圣保罗 )	sa-east-1

## 工作方式

带有 WebRTC 的 Amazon Kinesis Video Streams 支持网络浏览器、移动设备和其他支持 WebRTC 的应用程序之间的实时视频通信。它为构建视频流应用程序、处理信号、媒体流以及与其他 AWS 服务集成等任务提供了安全且可扩展的基础架构。本节介绍服务的底层架构、组件和 workflow，帮助解释其设计原则和机制。

### 主题

- [重要概念](#)
- [技术概念](#)
- [STUN、TURN 和 ICE 是如何协同工作的](#)
- [组件](#)

## 重要概念

以下是特定于 Amazon Kinesis Video Streams with WebRTC 的关键术语和概念。

### 信令通道

一种资源，使应用程序能够通过交换信令消息来发现、设置、控制和终止 peer-to-peer 连接。信令消息是两个应用程序相互交换以建立 peer-to-peer 连接的元数据。此元数据包括媒体编解码器和编解码器参数等本地媒体信息，及两个应用程序互相连接以进行实时流式传输而可能使用的网络候选路径。

流式传输应用程序可以与信令通道保持持续连接，并等待其他应用程序连接到它们。或者，只有当它们需要实时流媒体时，才能连接到信令通道。信号通道使用一个主站连接到多个查看器的概念，使应用程序能够在 one-to-few 模型中相互连接。发起连接的应用程序使用 ConnectAsMaster API 承担主设备的责任，并等待查看器连接。然后，多达 10 个应用程序可以通过调用 ConnectAsViewer API 来承担查看器责任，以连接到该信令通道。在它们连接到信令通道后，主应用程序和查看器应用程序可以互相发送信令消息，以建立直播媒体 peer-to-peer 流的连接。

## 对等

任何配置为通过带有 WebRTC 的 Kinesis Video Streams 进行实时、双向直播的设备或应用程序（例如，移动或网络应用程序、网络摄像头、家庭安全摄像头、婴儿监视器等）。

## 主实例

发起连接并连接到信令通道的对等方，能够发现媒体并与任何信令通道的已连接查看器交换媒体。

### Important

目前，一个信令通道只能有一个主设备。

## 查看者

连接到信令通道的对等方只能通过信令通道的主设备发现和交换媒体。查看器无法通过给定的信令通道发现其他查看器或与其交互。一个信令通道最多可以有 10 个连接的查看器。

## 技术概念

当你开始使用带有 WebRTC 的 Kinesis Video Streams 时，你还可以从学习几种相互关联的协议中受益，WebRTC 技术就是这些协议 APIs 的组成部分。

### NAT 的会话遍历实用工具 (STUN)

一种协议，用于发现您的公有地址并确定路由器中阻止与对等方直接连接的任何限制。

### 使用中继绕过 NAT 的遍历 (TURN)

通过打开与 TURN 服务器的连接并通过该服务器中继所有信息来绕过对称 NAT 限制的服务器。

### 会话描述协议 (SDP)

一种描述连接的多媒体内容的标准，例如分辨率、格式、编解码器、加密等，以便一旦传输数据，两个对等方就可以相互理解。

### SDP 提议

由代理发送的 SDP 消息，代理生成会话描述以创建或修改会话。它描述所需媒体通信的各个方面。

## SDP 应答

应答者响应从提议人收到的提议而发送的 SDP 消息。应答指出了已接受的各个方面。例如，提议中的所有音频和视频流是否都被接受。

## 交互式连接建立 (ICE)

允许您的 Web 浏览器与对等方连接的框架。

## ICE 候选项

发送对等方能够用于通信的方法。

## STUN、TURN 和 ICE 是如何协同工作的

让我们来看看两个对等方 A 和 B 的情况，它们都使用 WebRTC 对等双向媒体流式传输（例如，视频聊天应用程序）。当 A 想打电话给 B 时会发生什么？

要连接到 B 的应用程序，A 的应用程序必须生成 SDP 提议。SDP 提议包含有关 A 的应用程序要建立的会话的信息，包括要使用的编解码器、这是音频会话还是视频会话等。它还包含 ICE 候选项列表，这些候选项是 B 的应用程序可以尝试用来连接到 A 的 IP 和端口对。

要构建 ICE 候选项列表，A 的应用程序向 STUN 服务器发出一系列请求。服务器返回发起请求的公有 IP 地址和端口对。A 的应用程序将每个对添加到 ICE 候选项列表中，换句话说，它收集 ICE 候选项。一旦 A 的应用程序完成收集 ICE 候选项，它可以返回 SDP。

接下来，A 的应用程序必须通过这些应用程序用于通信的信令通道将 SDP 传递给 B 的应用程序。WebRTC 标准中未指定用于此交换的传输协议。它可以通过 HTTPS WebSocket、安全协议或任何其他通信协议执行。

现在，B 的应用程序必须生成 SDP 应答。B 的应用程序遵循 A 在上一步中使用的相同步骤：收集 ICE 候选项等。然后，B 的应用程序需要将此 SDP 应答返回给 A 的应用程序。

在 A 和 B 交换完毕后 SDPs，他们会执行一系列连通性检查。每个应用程序中的 ICE 算法都将候选 IP/port pair from the list it received in the other party's SDP, and sends it a STUN request. If a response comes back from the other application, the originating application considers the check successful and marks that IP/port 对作为有效的 ICE 候选对象。

在所有 IP/端口对上完成连接性检查后，应用程序协商并决定使用剩余的有效对之一。当选择一个对时，媒体开始在应用程序之间流动。

如果其中一个应用程序找不到通过了连接性检查的 IP/端口对，它们将向 TURN 服务器发出 STUN 请求以获取媒体中继地址。中继地址是一种公有 IP 地址和端口，用于将接收的数据包转发到应用程序或从应用程序中转发收到的数据包，以设置中继地址。然后，将该中继地址添加到候选项列表中，并通过信令通道进行交换。

## 组件

Kinesis Video Streams with WebRTC 包括以下组件：

- 控制层面

控制面板组件负责创建和维护 Kinesis Video Streams with WebRTC 信令通道。有关更多信息，请参阅 [Amazon Kinesis Video Streams API 参考](#)。

- 正在发送信号

信令组件管理 WebRTC 信令端点，这些端点允许应用程序安全地相互连接以进行实时媒体流。peer-to-peer [信令组件包括 Amazon Kinesis 视频信令 APIs REST 和一组 Websocket。 APIs](#)

- STUN

此组件管理 STUN 终端节点，这些终端节点使应用程序能够在其位于 NAT 或防火墙后面时发现其公有 IP 地址。

- TURN

此组件管理 TURN 端点，这些端点可在应用程序无法流式传输媒体时通过云端进行媒体中继 peer-to-peer。

- Kinesis Video Streams Streams WebRTC SDKs

您可以在设备和应用程序客户端上下载、安装和配置这些软件库，使具有 WebRTC 功能的相机物联网设备能够进行低 peer-to-peer 延迟的媒体流。它们 SDKs 还使安卓、iOS 和网络应用程序客户端能够将 Kinesis Video Streams 与 WebRTC 信令、TURN 和 STUN 功能与任何符合 WebRTC 标准的移动或网络播放器集成。

- [亚马逊 Kinesis Video Streams 带有 WebRTC 开发工具包的 C 语言版本，适用于嵌入式设备](#)
- [带有 WebRTC 开发工具包的 Amazon Kinesis Video Streams 适用于网络应用程序 JavaScript](#)
- [适用于安卓的亚马逊 Kinesis Video Streams WebRTC SDK](#)
- [适用于 iOS 的亚马逊 Kinesis Video Streams WebRTC SDK](#)

## 系统要求

本节介绍将 Amazon Kinesis Video Streams 与 WebRTC 配合使用的基本系统要求，包括网络要求和环境。它还包括有关调试连接的信息。

### 联网要求

Kinesis Video Streams with WebRTC 的信令服务端点的一般网络要求是：

- HTTPS 调用托管在 `https://*.kinesisvideo.{region}.amazonaws.com` 的端点
- WebSocket 与端点集成 `wss://*.kinesisvideo.{region}.amazonaws.com`
- 位于 `stun:stun.kinesisvideo.{aws-region}.amazonaws.com:443` 的 STUN 服务器
- 位于 `turn:._.kinesisvideo.{aws-region}.amazonaws.com:443` 和 `turns:._.kinesisvideo.{aws-region}.amazonaws.com:443` 的 TURN 服务器

#### Note

IPv6 STUN和TURN服务器目前不支持地址。

作为 RTCPeer连接的一部分，对等体之间使用的协议可以基于 TCP 或 UDP。

大多数应用程序都试图通过确定每个对等体的 IP 地址以及要作为 ICE 候选交换的端口和协议来建立直接 peer-to-peer 连接。这些候选项用于尝试使用这些候选项相互连接。他们将尝试每一对，直到可以建立连接。

### 网络环境

如果已将来自查看器的消息发送到主设备并记录了诸如 No valid ICE candidate 之类的日志，则未找到有效的连接路由。如果有防火墙阻止直接连接，或网络无法访问，则可能会发生这种情况。

要解决连接问题，请执行以下操作：

- 如果您不在主设备端使用 TURN，请务必启用 TURN。

TURN 在 C SDK 中默认处于启用状态。在 JavaScript SDK 中，TURN only 在 NAT 遍历中选择 STUN/TURN 或。

- 对于受限的网络，例如企业网络，请尝试其他可用的网络（有线或无线网络）。

如果您正在连接 VPN，请断开连接。

### Note

您可以忽略 Kinesis Video Streams TURN 服务器返回的 403 Forbidden IP 错误。服务器拒绝包含 localhost IPs 192.168.\* 或之类的 ICE 候选对 10.0.0.\*。这可能会导致一些 (但不是全部) ICE 候选对失败。

## 调试正在进行的连接

有几个方面可能会导致已建立的、持续 WebRTC 连接出现问题，但最常见的是联网。

您可以通过设置 `export AWS_KVS_LOG_LEVEL=1` 为环境变量来确认来自 SDK 的 VERBOSE 级别日志。

### Note

如果在分配的超时时间内未找到候选对，ICE 代理将返回错误状态 `0x5a00000d`。

有关日志级别的更多信息，请参阅 [GitHub](#)。

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

您将看到类似于以下内容的日志。从这些日志中，您可以确认交互式连接建立 (ICE) 候选对象 (IP 地址和端口) 和所选候选对。

```
2023-02-13 05:57:16 DEBUG    iceAgentReadyStateSetup(): Selected pair w1UdV9fE+_/
CuBel1p1, 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
```

## 带有 WebRTC 服务配额的亚马逊 Kinesis Video Streams

Kinesis Video Streams with WebRTC 具有以下服务限额：

### Important

以下服务配额要么是软配额 [s] (可以增加)，要么是硬配额 [h] (无法增加)。在下表中，您将在单个服务配额旁边看到 [s] 和 [h]。

### Note

TPS 代表每秒的交易次数。

## 主题

- [控制平面 API 服务配额](#)
- [信令 API 服务配额](#)
- [TURN 服务配额](#)
- [WebRTC 摄取服务配额](#)
- [故障排除](#)

## 控制平面 API 服务配额

下一节介绍控制平面的服务配额 APIs。

API	每人的最大 API 请求速率 AWS 账户	每人的最大频道 AWS 账户数 AWS 区域	每个频道的最大 API 请求量
CreateSignalingChannel	50 TPS [s]	<ul style="list-style-type: none"> <li>• 仅限美国东部 ( 弗吉尼亚北部 ) ( us-east-1 ) 和美国西部 ( 俄勒冈 ) ( us-west-2 ) —— 10,000</li> <li>• 所有其他支持的区域-5,000</li> </ul>	不适用
DeleteSignalingChannel	50 TPS [h]	不适用	5 TPS [h]
DescribeMediaStorageConfiguration	50 TPS [h]	不适用	5 TPS [h]
DescribeSignalingChannel	300 TPS [h]	不适用	5 TPS [h]
GetSignalingChannelEndpoint	300 TPS [h]	不适用	不适用
ListSignalingChannels	50 TPS [h]	不适用	不适用
ListTagsForResource	50 TPS [h]	不适用	5 TPS [h]

API	每人的最大 API 请求速率 AWS 账户	每人的最大频道 AWS 账户数 AWS 区域	每个频道的最大 API 请求量
TagResource	50 TPS [h]	不适用	5 TPS [h]
UntagResource	50 TPS [h]	不适用	5 TPS [h]
UpdateMediaStorage Configuration	10 TPS [h]	不适用	5 TPS [h]
UpdateSignalingChannel	50 TPS [h]	不适用	5 TPS [h]

## 信令 API 服务配额

以下部分介绍 Kinesis Video Streams with WebRTC 中的信令组件的服务限额。有关更多信息，请参阅 [工作方式](#)。

API	终止连接之前的最大 GO_AWAY 消息宽限期	每个通道的最大 API 请求速率	每个信道的最大并发连接数	最大连接时长	最大空闲连接超时时间
ConnectAs Master	1 分钟 (h)	3 TPS (h)	1 (h)	1 小时 (h)	10 分钟 (h)
ConnectAs Viewer	1 分钟 (h)	3 TPS (h)	10 (s)	1 小时 (h)	10 分钟 (h)

API	最大 API 请求速率	最大消息负载大小
SendAlexaOfferToMaster	<ul style="list-style-type: none"> <li>每个信令通道 5 TPS (h)</li> <li>AWS 区域 每 AWS 账户 (s) 100 TPS</li> </ul>	不适用
发送 ICECandidate	每个 WebSocket 连接 20 TPS (h)	10k (h)

API	最大 API 请求速率	最大消息负载大小
发送 SDPAnswer	每个 WebSocket 连接 5 TPS (h)	10k (h)
发送 SDPOffer	每个 WebSocket 连接 5 TPS (h)	10k (h)

## TURN 服务配额

以下部分介绍了 Kinesis Video Streams with WebRTC 中 Traversal Using Relays around NAT (TURN) 组件的服务限额。有关更多信息，请参阅 [工作方式](#)。

API 或参数	值
GetIceServerConfig	<ul style="list-style-type: none"> <li>每个信令通道 5 TPS (h)</li> <li>AWS 区域 每 AWS 账户 (s) 100 TPS</li> </ul>
比特率	5Mbps (h)
凭证生命周期	5 分钟 (h)
分配次数	每个信令通道 50 (h)

## WebRTC 摄取服务配额

以下部分介绍了 Amazon Kinesis Video Streams WebRTC 中媒体录制组件的服务限额。有关更多信息，请参阅 [使用带有 WebRTC 的 Amazon Kinesis Video Streams 来摄取和存储媒体](#)。

### JoinStorageSession

- API :
  - 每 AWS 账户 -50 TPS (h)
  - 每个频道-2 TPS (h)
- 流会话配额 :
  - 比特率-1 Mbps

- 会话时长-1 小时 (h)
- 空闲超时-3 分钟 (h)

## 故障排除

您只能将一个主设备和一个或多个查看器连接到单个信令通道。

不可能将多个主控制器连接到单个信令信道。

## 使用 WebRTC 访问 Kinesis Video Streams

您可以通过以下任何方式使用 Kinesis Video Streams with WebRTC :

### AWS Management Console

#### [入门 AWS Management Console](#)

该控制台是一个基于浏览器的界面，用于访问和使用 AWS 服务，包括带有 WebRTC 的 Kinesis Video Streams。

### AWS SDKs

AWS 提供软件开发套件 (SDKs)，其中包括适用于各种编程语言和平台（例如 Java、Python、Ruby、.NET、iOS、Android 等）的库和示例代码。这些 SDKs 提供了一种通过 WebRTC 创建对 Kinesis Video Streams 的编程访问权限的便捷方式。有关信息 AWS SDKs，包括如何下载和安装它们，请参阅[适用于 Amazon Web Services 的工具](#)。

### Kinesis Video Streams with WebRTC HTTPS API

您可以通过 WebRTC 访问 Kinesis Video Streams，也可以使用带有 WebRTC 的 Kinesis Video Streams 以编程方式访问 Kinesis Video Streams，它允许你直接向服务发出 APIs API 请求。有关更多信息，请参阅[亚马逊 Kinesis Video Streams Video Streams API 参考](#)。

# 入门

本节介绍如何在 Amazon Kinesis Video Streams with WebRTC 中执行下面的任务：

- [设置您的 AWS 账户 并创建管理员。](#)
- [使用 WebRTC 信令通道创建 Kinesis Video Streams。](#)
- [在 Ingenic T31 硬件上使用 WebRTC 设置 Kinesis Video Streams。](#)

如果您是首次使用 Kinesis Video Streams with WebRTC，建议先阅读 [工作方式](#)。

## 设置一个 AWS 账户

首次使用 Kinesis Video Streams with WebRTC 前，请完成以下任务：

主题

- [注册获取 AWS 账户](#)
- [创建具有管理访问权限的用户](#)
- [创建 AWS 账户密钥](#)

## 注册获取 AWS 账户

如果您没有 AWS 账户，请完成以下步骤来创建一个。

要注册 AWS 账户

1. 打开<https://portal.aws.amazon.com/billing/注册>。
2. 按照屏幕上的说明操作。

在注册时，将接到电话或收到短信，要求使用电话键盘输入一个验证码。

当您注册时 AWS 账户，就会创建AWS 账户根用户一个。根用户有权访问该账户中的所有 AWS 服务和资源。作为最佳安全实践，请为用户分配管理访问权限，并且只使用根用户来执行[需要根用户访问权限的任务](#)。

AWS 注册过程完成后会向您发送一封确认电子邮件。您可以随时前往 <https://aws.amazon.com/> 并选择“我的账户”，查看您当前的账户活动并管理您的账户。

## 创建具有管理访问权限的用户

注册后，请保护您的安全 AWS 账户 AWS 账户根用户 AWS IAM Identity Center，启用并创建管理用户，这样您就可以不会使用 root 用户执行日常任务。

### 保护你的 AWS 账户根用户

1. 选择 Root 用户并输入您的 AWS 账户 电子邮件地址，以账户所有者的身份登录。[AWS Management Console](#)在下一页上，输入您的密码。

要获取使用根用户登录方面的帮助，请参阅《AWS 登录 用户指南》中的 [Signing in as the root user](#)。

2. 为您的根用户启用多重身份验证 ( MFA )。

有关说明，请参阅 [IAM 用户指南中的为 AWS 账户 根用户启用虚拟 MFA 设备 \( 控制台 \)](#)。

### 创建具有管理访问权限的用户

1. 启用 IAM Identity Center。

有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 [Enabling AWS IAM Identity Center](#)。

2. 在 IAM Identity Center 中，为用户授予管理访问权限。

有关使用 IAM Identity Center 目录 作为身份源的教程，请参阅《[用户指南](#)》[IAM Identity Center 目录中的使用默认设置配置AWS IAM Identity Center 用户访问权限](#)。

### 以具有管理访问权限的用户身份登录

- 要使用您的 IAM Identity Center 用户身份登录，请使用您在创建 IAM Identity Center 用户时发送到您的电子邮件地址的登录网址。

有关使用 IAM Identity Center 用户[登录的帮助](#)，请参阅[AWS 登录 用户指南中的登录 AWS 访问门户](#)。

### 将访问权限分配给其他用户

1. 在 IAM Identity Center 中，创建一个权限集，该权限集遵循应用最低权限的最佳做法。

有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 [Create a permission set](#)。

2. 将用户分配到一个组，然后为该组分配单点登录访问权限。

有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 [Add groups](#)。

## 创建 AWS 账户密钥

你需要一个 AWS 账户密钥才能通过 WebRTC 以编程方式访问 Kinesis Video Streams。

要创建 AWS 账户密钥，请执行以下操作：

1. 登录 AWS Management Console 并打开 IAM 控制台，网址为 <https://console.aws.amazon.com/iam/>。
2. 在导航栏中选择 Users (用户)，然后选择 Administrator (管理员) 用户。
3. 选择安全凭证选项卡，然后选择创建访问密钥。
4. 记录访问密钥 ID。在秘密访问密钥列下选择显示，然后查看秘密访问密钥。

## 创建信令通道

带有 WebRTC 信令通道的 Kinesis Video Streams 可促进在 WebRTC 客户端之间建立和维护连接所需的信令消息交换。peer-to-peer 它处理会话描述协议 (SDP) 报价和会话参数答案的协商，以及交互式连接建立 (ICE) 候选网络信息的交换。

要创建信令通道，请调用 [CreateSignalingChannel](#) API。本页将向您展示如何使用 AWS Management Console AWS CLI、和其中一个来调用该 API AWS SDKs。

### Important

记下频道 ARN，稍后你会用到的。

### AWS Management Console

执行以下操作：

1. [在 HOME/#/SignalingChannels 上打开 Kinesis Video Streams 信令频道控制台 https://console.aws.amazon.com/kinesisvideo/。](https://console.aws.amazon.com/kinesisvideo/)

2. 选择 Create signaling channel (创建信令通道)。
3. 在创建新的信令通道页面上，输入信令通道的名称。

将默认 Time-to-live (Ttl) 值保留为 60 秒。

选择 Create signaling channel (创建信令通道)。

4. 一旦创建信令通道，在通道的详细信息页面上查看其详细信息。

## AWS CLI

确认您已 AWS CLI 安装并配置了。有关更多信息，请参阅 [AWS Command Line Interface 《用户指南》](#)。

有关安装说明，请参阅 [《AWS Command Line Interface 用户指南》](#)。安装完成后，AWS CLI 使用凭据和区域进行 [配置](#)。

或者，打开已 AWS CLI 安装和配置的 AWS CloudShell 终端。有关更多信息，请参阅 [AWS CloudShell 用户指南](#)。

使用以下命令运行以下 [Create-Signaling-Channel](#) 命令：AWS CLI

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

响应看起来与以下内容类似：

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

## AWS SDK

此代码片段向您展示了如何使用适用于 v2 的 SDK 创建带有 WebRTC 信令通道的 Kinesis Video Streams。AWS JavaScript 语法将与其他语法不同 AWS SDKs，但一般流程将相同。在上查看完整的代码示例 [GitHub](#)。

创建 Kinesis Video Streams 客户端。这是用于调用 CreateSignalingChannel API 的客户端。

```
const clientConfig = {
```

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

使用客户端调用 CreateSignalingChannel API。

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

打印响应。

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

包含此代码示例的实时网页可在上使用 [GitHub](#)。输入您的区域、AWS 凭证和信令频道的名称。

选择“创建频道”。

## 与 Ingenic T31 集成

### Note

AWS 不以任何方式认可该芯片组，也不保证集成会正常运行。本指南基于亚马逊 Kinesis Video Streams Video Streams 团队的测试，旨在帮助客户设置设备。

按照以下步骤在 Ingenic T31 硬件上使用 WebRTC 设置亚马逊 Kinesis Video Streams。

## 下载代码

### 1. 设置生成目录。

在本教程中，在 Downloads 文件夹 ingenic 中创建一个名为的目录。

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

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

2. 将以下存储库克隆到您的新目录中：<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
```

## 设置构建环境

1. 获取包含工具链的预构建的 docker 镜像并将其放入文件夹。ingenic

### Important

工具链可能因主板和 CPU 类型而异。请向供应商查询正确的工具链。

2. 加载 d Ingenic-T31-app-build.tar.gz ocker 镜像。

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

预期输出：

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

记下Image Id. 下一步你需要这个。

3. 运行以下命令将主机上的当前目录 (~Downloads/ingenic) 挂载到 Docker 容器内的/ingenicMappedFolder目录中。

此命令还通过终端会话在交互模式下运行容器。你将被放到容器内的 Bash shell 中，它允许你直接执行命令。

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

#### 4. 查看该/`ingenicMappedFolder`文件夹的内容以确认绑定挂载卷是否成功。

提示：

```
ls /ingenicMappedFolder
```

响应：

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

#### 5. 验证 docker 容器是否设置了环境变量：

提示：

```
env
```

响应：

```
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. 验证 docker 容器是否包含工具链和 Ingenic SDK。它应该位于/`ingenic-sdk`。

提示：

```
ls /
```

响应：

```
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 应用程序构建 Amazon Kinesis Video Streams

1. 键入以下内容：

```
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
```

该kvswebrtcmaster-static应用程序位于/ingenicMappedFolder/AWS-Solution-Remote-Diagnostic-Media-Interface/build/samples/webrtc/。

2. 运行以下命令退出 docker 容器：

```
exit
```

3. 在您的主机上，验证kvswebrtcmaster-static应用程序是否存在。

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

## 将应用程序上传到设备

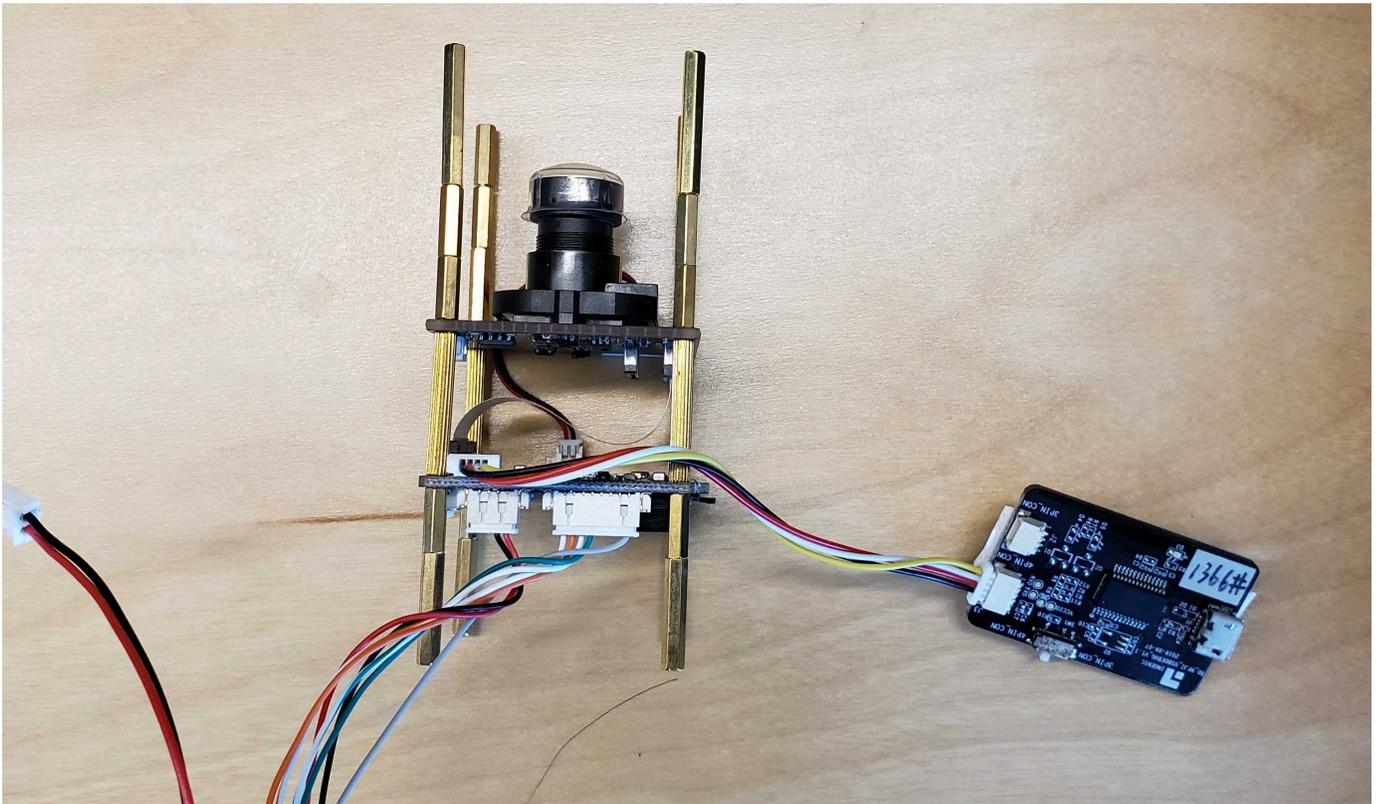
1. 在您的主机上，将静态二进制文件复制到 micro SD 卡。

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

2. 下载这个 [.pem 文件](#) 并将其放入 micro SD 卡中，如。cert.pem
3. 将 micro SD 卡放入 micro SD 卡插槽中。

## Connect 连接到设备

1. 将串行端口工具连接到主板，如下所示。



2. 将以太网和电源线插入设备。红色 LED 应亮起。
3. 将您的主机连接到串行端口工具上的 micro-USB 插槽。
4. 检查连接的设备：

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

#### **Note**

如果您有多个 TTY 设备，请确定哪个 TTY 设备是 Ingenic 主板。断开设备与主机的连接，ls 然后重新运行。比较命令输出并找出差异。

5. 使用 screen 或其他串行端口工具（例如 Tera Term）连接到它。将波特率设置为 115200。

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

6. 根据主板的状况确定相应的操作：
  - 如果 shell 会话以结尾 #，则启动已中断。使用 boot 命令启动 Linux 操作系统，然后继续执行此步骤的其余部分。

- 如果 shell 会话要求您登录，请键入您的密码。
- 如果屏幕为空白，请按 Enter。
- 如果屏幕显示错误 `Cannot exec '/dev/tty.usbserial-XXXXXXX': No such file or directory`，请仔细检查主板和主机之间的所有物理连接。

## 将 micro SD 卡安装在主板上

1. 创建目录：

### Note

在这个例子中，我们使用的是 `sdcard`。

```
mkdir /tmp/sdcard
```

2. 键入以下内容将 micro SD 卡安装到文件夹：

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

3. 查看该文件夹的内容：

```
ls /tmp/sdcard
```

## 运行应用程序

1. 导航到已安装的目录：

```
cd /tmp/sdcard
```

2. 从董事会导出您的凭证和其他信息：

```
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. 运行应用程序：

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

## 查看媒体

要查看媒体，请以查看者身份连接到信令频道。有关示例，请参阅以下部分：

- [JavaScript](#)
- [AWS Management Console](#)
- [Android](#)
- [iOS](#)

## 故障排除

本节包含我们遇到的常见问题和问题。

当我将主板连接到我的主机并运行时 `ls /dev/tty.*`，设备不会出现。

验证连接和所有电线是否安全，以及设备是否已开机。检查 USB-to-TTY 设备上的所有 LED 指示灯是否亮起。如果您仍然遇到问题，请联系供应商以进一步诊断与主板连接的问题。

### 应用程序无法连接到信令

如果收到以下错误之一：

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

或者

```
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.)" }
```

这意味着 Ingenic 主板上的时间不正确。通过运行 `date` 命令进行验证。根据供应商的说明设置时间，然后再次运行应用程序。

## 即使文件在那里，应用程序也无法启动

如果错误如下所示：

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

这意味着操作系统无法找到应用程序的 ELF 标头中列出的依赖项。例如，libc 在设备上。

通常，这表示所使用的工具链与开发板不兼容，或者链接器标志 (LDFLAGS) 未正确指向所需的库路径，从而导致运行时依赖关系未解决。

确保正确的工具链 (uclibc 与其版本对比) glibc 与主板上的软件相匹配，并 LDFLAGS 根据其进行 [the section called “使用 WebRTC 应用程序进行构建”](#) 设置。

## 如何卸载 micro SD 卡？

要卸载 micro SD 卡，请使用以下命令：

```
umount /tmp/sdcard
```

## 直播媒体 (SDKs)

在 Amazon Kinesis Video Streams WebRTC 中，对等设备是配置为通过信令通道进行实时、双向直播的设备。带有 WebRTC 的 Amazon Kinesis Video Streams easy-to-use SDKs 是软件库，您可以将其下载并安装在要通过给定信令通道配置为对等设备的设备和应用程序客户端上。

带有 WebRTC 的 Amazon Kinesis Video Streams 包括以下内容：SDKs

- [亚马逊 Kinesis Video Streams 带有 WebRTC 开发工具包的 C 语言版本，适用于嵌入式设备](#)
- [带有 WebRTC 开发工具包的 Amazon Kinesis Video Streams 适用于网络应用程序 JavaScript](#)
- [适用于安卓的亚马逊 Kinesis Video Streams WebRTC SDK](#)
- [适用于 iOS 的亚马逊 Kinesis Video Streams WebRTC SDK](#)

每个 SDK 都包含相应的示例和 step-by-step 说明，可帮助您构建和运行这些应用程序。您可以使用这些样本在 Web/Android/iOS 应用程序或嵌入式设备的任何组合之间进行低延迟、直播、双向音频和视频流以及数据交换。换句话说，您可以将实时音频和视频从嵌入式摄像头设备流式传输到 Android 或 Web 应用程序，或在两个 Android 应用程序之间进行流式传输。

## 亚马逊 Kinesis Video Streams 带有 WebRTC 开发工具包的 C 语言版本，适用于嵌入式设备

以下 step-by-step 说明描述了如何为嵌入式设备下载、构建和运行带有 WebRTC SDK 的 C 语言开发工具包的 Kinesis Video Streams 及其相应示例。

支持以下编解码器：

- 音频：
  - G.711 A-Law
  - G.711 U-Law
  - Opus
- 视频：
  - H.264
  - H.265
  - VP8

## 下载 SDK

要下载 C 中适用于嵌入式设备的 Kinesis Video Streams with WebRTC 开发工具包，请运行以下命令：

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

## 构建 SDK

### Important

在 macOS 上完成这些步骤之前，根据您的 macOS 版本，您必须运行 `xcode-select --install` 下载带有命令行工具和标头的软件包。然后打开 `/Library/Developer/CommandLineTools/Packages/macOS_SDK_headers_for_macOS_10.14.pkg` 并按照安装程序安装命令行工具和标头。这些操作只需要在调用 `cmake` 前执行一次。如果您已经安装了命令行工具和标头，则无需再次运行此命令。

完成以下步骤：

#### 1. 安装 `cmake`：

- 在 macOS 上运行 `brew install cmake pkg-config srt`
- 在 Ubuntu 上运行 `sudo apt-get install pkg-config cmake libcap2 libcap-dev`

#### 2. 获取要用于此演示的 AWS 账户 的访问密钥和密钥。

#### 3. 运行以下命令在您下载的 WebRTC C 开发工具包中创建一个 `build` 目录，并从中执行 `cmake`：

```
$ mkdir -p amazon-kinesis-video-streams-webrtc-sdk-c/build; cd amazon-kinesis-video-streams-webrtc-sdk-c/build; cmake ..
```

#### 4. 现在，导航到刚使用上述步骤创建的 `build` 目录，然后运行 `make` 以构建 WebRTC C 开发工具包及其提供的示例。

**Note**

如果系统尚未安装 `gststreamer`，则无法构建 `kvsWebrtcClientMasterGstSample`。要确保它是（在 macOS 上）构建的，您必须运行：`brew install gststreamer gst-plugins-base gst-plugins-good`

## 运行 SDK 示例

完成上述过程后，您最终会在您的 `build` 目录中使用以下示例应用程序：

- `kvsWebrtcClientMaster`-此应用程序通过信令通道发送样本 H264/Opus 帧（路径：`/samples/h264SampleFrames` and `/samples/opusSampleFrames`）。它也接受传入的音频（如果在浏览器中启用）。在浏览器中勾选时，它会打印终端中收到的音频数据包的元数据。
- `kvsWebrtcClientViewer` - 此应用程序接受示例 H264/Opus 帧并打印出来。
- `kvsWebrtcClientMasterGstSample`-此应用程序从管道发送样本 H264/Opus 帧。GStreamer

要运行任何这些示例，请完成以下步骤：

1. 使用您的 AWS 账户 凭据设置您的环境：

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

如果您使用的是临时 AWS 证书，请同时导出您的会话令牌：

```
export AWS_SESSION_TOKEN=YourSessionToken
```

如果您要设置自定义 CA 证书路径，则可以使用以下方法进行设置：

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

**Note**

默认情况下，SSL CA 证书设置为... /certs/cert.pem，它指向中此存储库中的文件。[GitHub](#)

2. 通过将您要向信令通道提供的名称传递给示例应用程序来运行任一应用程序。应用程序使用您提供的名称创建信令通道。例如，要创建一个名为 myChannel 的信令通道并开始通过该通道发送示例 H264/Opus 帧，请运行以下命令：

```
./kvsWebrtcClientMaster myChannel
```

当命令行应用程序打印 Connection established 时，您可以继续下一步。

3. 现在您的信令通道已创建，并且连接的主设备正在将媒体流式传输到它，您可以查看此流。例如，您可以在 Web 应用程序中查看此实时流。为此，请按照[使用示例应用程序](#)中的步骤打开 WebRTC SDK 测试页面，并使用您在上面为主服务器指定的 AWS 相同凭据和相同的信令通道设置以下值：

- 访问密钥 ID
- 秘密访问密钥
- 信令通道名称
- 客户端 ID ( 可选 )

选择 Start viewer (启动查看器) 以启动示例 H264/Opus 帧的实时视频流式传输。

## 视频教程

本视频演示了如何连接摄像头并开始使用适用于 WebRTC 的 Amazon Kinesis Video Streams。

## 带有 WebRTC 开发工具包的 Amazon Kinesis Video Streams 适用于网络应用程序 JavaScript

你可以在网络应用程序中找到带有 WebRTC SDK JavaScript 的 Kinesis Video Streams 及其相应的示例。[GitHub](#)

### 主题

- [安装 SDK](#)
- [WebRTC 软件开发工具 JavaScript 包文档](#)
- [使用示例应用程序](#)
- [编辑示例应用程序](#)

## 安装 SDK

是否以及如何安装带有 WebRTC SDK 的 Kinesis Video Streams 取决于代码是在模块 JavaScript 中执行还是浏览器脚本执行 Node.js。

### NodeJS module

在 Node.js 中安装带有 WebRTC SDK JavaScript 的 Kinesis Video Streams 的首选方法是[使用 Node.js 包管理器 npm](#)。

该软件包托管在 <https://www.npmjs.com/package/amazon-kinesis-video-streams-webrtc>。

要在 Node.js 项目中安装此 SDK，请使用终端导航到与项目相同的目录 package.json：

键入以下内容：

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

你可以像典型的 Node.js 模块一样导入 SDK 类：

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

### Browser

您无需安装开发工具包以在浏览器脚本中使用它。您可以在 HTML 页面中 AWS 使用脚本直接加载托管 SDK 包。

要在浏览器中使用 SDK，请在您的 HTML 页面中添加以下脚本元素：

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

页面中加载开发工具包之后，可从全局变量 `KVSWebRTC` ( 或 `window.KVSWebRTC` ) 使用开发工具包。

例如，`window.KVSWebRTC.SignalingClient`。

## WebRTC 软件开发工具 JavaScript 包文档

SDK 方法的文档位于 GitHub 自述文件的“[文档](#)”下。

在“[用法](#)”部分中，提供了有关将此 SDK 与用于构建基于 Web 的 JavaScript 查看器应用程序的 AWS SDK 集成的更多信息。

有关完整应用程序 ( 包括主角色和查看者角色 ) 的示例，请参阅 `examples` 目录。

### 使用示例应用程序

带有 WebRTC 的 Kinesis Video Streams 还托管了一个示例应用程序，您可以使用该应用程序创建新的信令信道或连接到现有频道并将其用作主频道或查看器。

带有 WebRTC 的 Kinesis Video Streams 示例应用程序位于。[GitHub](#)

示例应用程序的代码位于 `examples` 目录中。

#### 主题

- [peer-to-peer从示例应用程序流式传输到 AWS Management Console](#)
- [peer-to-peer从示例应用程序流式传输到示例应用程序](#)
- [peer-to-peer使用 WebRTC Ingestion 从示例页面直播到示例页面](#)

### peer-to-peer从示例应用程序流式传输到 AWS Management Console

1. 打开 [Kinesis Video Streams with WebRTC 示例应用程序](#) 并完成以下操作：

- AWS 区域。例如，`us-west-2`。
- 您的 IAM 用户或角色的 AWS 访问密钥和密钥。如果您使用的是长期 AWS 证书，请将会话令牌留空。
- 要连接的信令通道的名称。

如果要连接到新的信令信道，请选择 `create Channel` 以使用框中提供的值创建一个信令信道。

**Note**

对于当前账户和地区，您的信令频道名称必须是唯一的。您可以使用字母、数字、下划线 ( \_ ) 和连字符 ( - )，但不能使用空格。

- 是发送音频、视频还是两者。
- WebRTC 摄取和存储。展开节点并选择以下选项之一：
  - 选择“自动确定摄取模式”。
  - 确保未选择“自动确定摄取模式”，并将手动覆盖设置为“关闭”。

**Note**

自动确定摄取模式让应用程序调用 [DescribeMediaStorageConfiguration](#) API 来确定要在哪个模式下运行（或 Peer-to-peer WebRTC 摄取）。这个额外的 API 调用会增加少量启动时间。

如果您提前知道此信令通道在何种模式下运行，请使用手动替代来跳过此 API 调用。

- 移民局候选人一代。保留 STUN/TURN 选中状态并保持 Trickle ICE 启用状态。

## 2. 选择 Start Master 以连接到信令信道。

如果需要，允许访问您的摄像头和/或麦克风。

## 3. 在中打开 [Kinesis Video Streams](#) 控制台。AWS Management Console

确保选择了正确的区域。

## 4. 在左侧导航栏中，选择 [信令频道](#)。

在上面选择信令通道的名称。如果需要，请使用搜索栏。

## 5. 展开“媒体播放查看器”部分。

## 6. 选择视频播放器上的播放按钮。这作为一个人加入了 WebRTC 会话。viewer

在演示页面上发送的媒体应显示在 AWS Management Console。

## peer-to-peer 从示例应用程序流式传输到示例应用程序

### 1. 打开 [Kinesis Video Streams with WebRTC 示例](#) 应用程序并填写以下信息：

- AWS 区域。例如，us-west-2。
- 您的 IAM 用户或角色的 AWS 访问密钥和密钥。如果您使用的是长期 AWS 证书，请将会话令牌留空。
- 要连接的信令通道的名称。

如果要连接到新的信令信道，请选择 `Create Channel` 以使用框中提供的值创建一个信令信道。

#### Note

对于当前账户和地区，您的信令频道名称必须是唯一的。您可以使用字母、数字、下划线 (`_`) 和连字符 (`-`)，但不能使用空格。

- 是发送音频、视频还是两者。
- WebRTC 摄取和存储。展开节点并选择以下选项之一：
  - 选择“自动确定摄取模式”。
  - 确保未选择“自动确定摄取模式”，并将手动覆盖设置为“关闭”。

#### Note

自动确定摄取模式让应用程序调用 [DescribeMediaStorageConfiguration](#) API 来确定要在哪个模式下运行（或 Peer-to-peer WebRTC 摄取）。这个额外的 API 调用会增加少量启动时间。

如果您提前知道此信令通道在何种模式下运行，请使用手动替代来跳过此 API 调用。

- 移民局候选人一代。保留 STUN/TURN 选中状态并保持 Trickle ICE 启用状态。
2. 选择 `Start Master` 以 `master` 角色身份连接到信令通道。

如果需要，允许访问您的摄像头和/或麦克风。

3. 打开另一个浏览器选项卡，然后打开 [Kinesis Video Streams with WebRTC 示例应用程序](#)。上一次运行的所有信息都应加载完毕。
4. 向下滚动并选择“启动查看器”，以 `viewer` 角色身份连接到信令频道。

您应该会看到 `master` 和 `viewer` 之间正在交换媒体。

## peer-to-peer使用 WebRTC Ingestion 从示例页面直播到示例页面

1. [the section called “从浏览器提取媒体”](#)按照操作连接主参与者并确保其已连接到存储会话。
2. 按照操作[the section called “将观众添加到摄取会话中”](#)添加观看者参与者。

查看者参与者将连接存储会话并从中接收媒体。他们可以将可选音频发送回存储会话。

存储会话负责混合从主参与者和查看者参与者那里收到的媒体，然后将其发送到相应的目的地。

3. 你可以通过 [Kinesis Video Streams](#) 播放来查看和消费摄取的媒体。

## 编辑示例应用程序

要出于开发目的编辑 SDK 和示例应用程序，请按照以下说明进行操作。

先决条件

NodeJS 版本 16+

### Note

我们建议从 <https://nodejs.org/en/download> 下载最新的长期支持 (LTS) 版本。

## 编辑示例应用程序

1. 下载带有 WebRTC SDK 的 Kinesis Video Streams。JavaScript

在终端中键入以下内容：

```
git clone https://github.com/aws-labs/amazon-kinesis-video-streams-webrtc-sdk-js.git
```

2. 导航到包含 package.json 文件的目录。该文件位于存储库的根目录中。

在终端中键入以下内容：

```
cd amazon-kinesis-video-streams-webrtc-sdk-js
```

3. 安装依赖项。

在终端中键入以下 [npm CLI](#) 命令：

```
npm install
```

4. 启动 Web 服务器以开始提供网页。

在终端中键入以下 [npm CLI](#) 命令：

```
npm run develop
```

5. 在浏览器中，访问 <http://localhost:3001/>。

您可以通过编辑 examples 目录中的文件来编辑网页。

## 适用于安卓的亚马逊 Kinesis Video Streams WebRTC SDK

以下 step-by-step 说明描述了如何使用安卓版 WebRTC SDK 下载、构建和运行 Kinesis Video Streams 及其相应示例。

### Note

亚马逊 Kinesis Video Streams 不 IPv6 支持安卓设备上的地址。查看有关在您的 [Android 设备 IPv6 上禁用](#) 的更多信息。

## 下载 SDK

要在 Android 中下载 WebRTC 开发工具包，请运行以下命令：

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

## 构建 SDK

要在 Android 中构建 WebRTC 开发工具包，请完成以下步骤：

1. 选择以项目形式打开打开 `amazon-kinesis-video-streams-webrtc-sdk-android/build.gradle`，将安卓 WebRTC SDK 导入 Android Studio 集成式开发环境 (IDE)。
2. 如果您第一次打开项目，它会自动同步。如果不是，则发起同步。当您看到编译错误时，请选择安装缺失 SDKs 的 SDK 软件包，然后选择“接受”并完成安装，从而选择安装所需的任何内容。

3. 配置 Amazon Cognito ( 用户池和身份池 ) 设置。有关详细步骤，请参阅[为软件开发工具包配置 Amazon Cognito](#)。这会生成构建 Android WebRTC 开发工具包所需的身份验证和授权设置。
4. 在 Android IDE 中，打开 `awsconfiguration.json` ( 从 `src/main/res/raw/` )。此文件如下所示：

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

使用通过运行[为软件开发工具包配置 Amazon Cognito](#)中的步骤生成的值更新 `awsconfiguration.json`。

5. 确保您的 Android 设备已连接到运行 Android IDE 的计算机。在 Android IDE 中，选择连接的设备，然后构建并运行 WebRTC Android 开发工具包。

此步骤将在您的 Android 设备上安装一个名为 `AWSKinesisVideoWebRTCDemoApp` 的应用程序。使用此应用程序，您可以验证移动、网络 and IoT 设备客户端之间的实时 WebRTC 音频/视频流式传输。

## 运行示例应用程序

完成以下步骤：

1. 在您的 Android 设备上，打开AWSKinesisVideoWebRTCDemo应用程序并使用新的（先创建）或现有的 Amazon Cognito 账户登录。
2. 在AWSKinesisVideoWebRTCDemo应用程序中，导航到信道配置页面，然后创建新的信令信道或选择现有信令信道。

### Note

目前，使用此 SDK 中的示例应用程序，您只能在 AWSKinesisVideoWebRTCDemoApp 中运行一个信令通道。

3. 可选：如果您想以查看器身份连接到此通道，请选择唯一的 Client Id (客户端 ID)。仅当多个查看器连接到一个通道时，才需要客户端 ID。这有助于通道的主设备识别各自的查看器。
4. 选择 AWS 区域 以及是否要发送音频或视频数据，或者两者兼而有之。
5. 要验证 peer-to-peer直播，请执行以下任一操作：

### Note

请务必在本演示中使用的所有客户端上指定相同的信令频道名称、AWS 区域、观众 ID 和 AWS 账户 ID。

- Peer-to-peer 在两台 Android 设备之间进行直播：主设备和查看器
  - 使用上述步骤，在两个 Android 设备上下载、构建和运行 Android WebRTC 开发工具包。
  - 在一台 Android 设备上以主模式打开AWSKinesisVideoWebRTCDemo应用程序（选择 S TART MASTER），开始新的会话（信令信道）。

### Note

目前，任何给定信令通道只能有一个主设备。

- 在查看器模式下在第二台 Android 设备上打开AWSKinesisVideoWebRTCDemo应用程序，以连接到在上述步骤中启动的信令频道（会话）（选择 S TART VIEWER）。

验证查看器是否可以查看主设备的音频/视频数据。

- Peer-to-peer 在嵌入式 SDK 网站和安卓设备查看器之间进行流式传输
- 在摄像头设备上，下载、构建 [亚马逊 Kinesis Video Streams 带有 WebRTC 开发工具包的 C 语言版本，适用于嵌入式设备](#) 并以主设备模式运行它。
- 使用上述步骤，在 Android 设备上下载、构建和运行 Android WebRTC 开发工具包。在查看器模式下在此 Android 设备上打开AWSKinesisVideoWebRTCDemo应用程序，确认查看者可以看到嵌入式 SDK 主服务器的音频/视频数据。
- Peer-to-peer 在作为主设备的 Android 设备和作为查看器的网络浏览器之间进行流式传输
- 使用上述步骤，在 Android 设备上下载、构建和运行 Android WebRTC 开发工具包。在主模式下打开此 Android 设备上的AWSKinesisVideoWebRTCDemo应用程序（选择 S TART MASTER），开始新的会话（信令信道）。
- 下载、构建 [带有 WebRTC 开发工具包的 Amazon Kinesis Video Streams 适用于网络应用程序 JavaScript](#) 并以查看器身份运行它，并验证查看器是否可以查看 Android 主设备的音频/视频。

## 为软件开发工具包配置 Amazon Cognito

### 先决条件

- 建议使用 [Android Studio](#) 检查、编辑和运行应用程序代码。我们建议使用最新的稳定版本。
- 在示例代码中，您需要提供亚马逊 Cognito 凭证。

按照以下步骤设置 Amazon Cognito 用户池和身份池。

### 设置用户池

#### 设置用户池

1. 登录 [Amazon Cognito 控制台](#) 并验证区域是否正确。
2. 在左侧导航栏中，选择“用户池”。
3. 在用户池部分，选择创建用户池。
4. 完成以下各节：
  - a. 第 1 步：配置登录体验-在 Cognito 用户池登录选项部分，选择相应的选项。

选择下一步。

- b. 步骤 2：配置安全要求-选择相应的选项。

选择下一步。

- c. 第 3 步：配置注册体验-选择相应的选项。

选择下一步。

- d. 步骤 4：配置消息传送-选择相应的选项。

在 IAM 角色选择字段中，选择现有角色或创建新角色。

选择下一步。

- e. 第 5 步：集成您的应用程序-选择相应的选项。

在“初始应用程序客户端”字段中，选择“机密客户端”。

选择下一步。

- f. 步骤 6：查看并创建-查看您在前面部分中的选择，然后选择创建用户池。

5. 在用户池页面上，选择您刚刚创建的池。

复制用户池 ID 并记下来以备后用。在awsconfiguration.json文件中，这是CognitoUserPool.Default.PoolId。

6. 选择“应用程序集成”选项卡，然后转到页面底部。
7. 在应用程序客户端列表部分，选择您刚刚创建的应用程序客户端名称。

复制客户端 ID 并记下来以备后用。在awsconfiguration.json文件中，这是CognitoUserPool.Default.AppClientId。

8. 出示客户机密并记下来以备后用。在awsconfiguration.json文件中，这是CognitoUserPool.Default.AppClientSecret。

## 设置身份池

### 设置身份池

1. 登录 [Amazon Cognito 控制台](#) 并验证区域是否正确。
2. 在左侧导航栏中，选择身份池。

3. 选择创建身份池。

4. 配置身份池。

a. 步骤 1：配置身份池信任-完成以下部分：

- 用户访问权限-选择经过身份验证的访问权限
- 经过身份验证的身份源-选择 Amazon Cognito 用户池

选择下一步。

b. 步骤 2：配置权限-在“经过身份验证的角色”部分，填写以下字段：

- IAM 角色-选择创建新的 IAM 角色
- IAM 角色名称-输入名称并记下来供后续步骤使用。

选择下一步。

c. 步骤 3：Connect 身份提供商-在“用户池详情”部分填写以下字段：

- 用户池 ID-选择您之前创建的用户池。
- 应用程序客户端 ID-选择您之前创建的应用程序客户端 ID。

选择下一步。

d. 步骤 4：配置属性-在身份池名称字段中键入名称。

选择下一步。

e. 第 5 步：查看并创建-查看您在每个部分中的选择，然后选择创建身份池。

5. 在身份池页面上，选择您的新身份池。

复制身份池 ID 并记下来以备后用。在 `awsconfiguration.json` 文件中，这是 `CredentialsProvider.CognitoIdentity.Default.PoolId`。

6. 更新 IAM 角色的权限。

a. 登录 AWS Management Console 并打开 IAM 控制台，网址为 <https://console.aws.amazon.com/iam/>。

b. 在左侧导航栏中，选择“角色”。

c. 找到并选择您在上面创建的角色。

**Note**

如果需要，请使用搜索栏。

- d. 选择附加的权限策略。

选择编辑。

- e. 选择 JSON 选项卡，然后将策略替换为以下内容：

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

选择下一步。

- f. 如果尚未选中“将此新版本设为默认版本”旁边的复选框。

选择保存更改。

## 适用于 iOS 的亚马逊 Kinesis Video Streams WebRTC SDK

以下 step-by-step 说明描述了如何在 iOS 中下载、构建和运行 Kinesis Video Streams WebRTC SDK 及其相应示例。

### 下载 SDK

要在 iOS 中下载 WebRTC 开发工具包，请运行以下命令：

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

## 构建 SDK

完成以下步骤：

1. 通过KinesisVideoWebRTCDemoApp.xcworkspace打开（路径 amazon-kinesis-video-streams : webrtc-sdk-ios-AWSKinesis VideoWeb RTCDemo /Swift/ App.xcWorkspace），将 XCode iOS WebRTC SDK 导入 iOS 电脑上的集成开发环境 (IDE)。
2. 如果您第一次打开该项目，它会自动构建。如果没有，则开始构建。

您可能会看到以下错误：

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

如果您看到此错误，请执行以下操作：

- a. 将当前工作目录更改为 amazon-kinesis-video-streams-webrtc-sdk-ios/Swift 并在命令行中运行以下命令：

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

- b. 将当前工作目录更改为 amazon-kinesis-video-streams-webrtc-sdk-ios 并在命令行中运行以下命令：

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

- c. Build。

3. 配置 Amazon Cognito（用户池和身份池）设置。有关详细步骤，请参阅[为软件开发工具包配置 Amazon Cognito](#)。这会生成构建 iOS WebRTC 开发工具包所需的身份验证和授权设置。
4. 在 IDE 中，打开 awsconfiguration.json 文件（来自 /Swift/KVSiOSApp）。此文件如下所示：

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

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

使用通过运行[为软件开发工具包配置 Amazon Cognito](#)中的步骤生成的值更新 `awsconfiguration.json`。

5. 在 IDE 中，打开 `Constants.swift` 文件（来自 `/Swift/KVSiOSApp`）。此文件如下所示：

```
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"
```

使用通过运行[为软件开发工具包配置 Amazon Cognito](#)中的步骤生成的值更新 Constants.swift。

6. 确保你的 iOS 设备已连接到你正在运行的 Mac 电脑 XCode。在中 XCode，选择连接的设备，然后构建并运行 WebRTC iOS 开发工具包。

此步骤将在 iOS 设备上安装名为 AWSKinesisVideoWebRTCDemoApp 的应用程序。使用此应用程序，您可以验证移动、网络 and IoT 设备客户端之间的实时 WebRTC 音频/视频流式传输。

## 运行示例应用程序

完成以下步骤：

1. 在你的 iOS 设备上，打开AWSKinesisVideoWebRTCDemo应用程序，然后使用新的（先创建）或现有的 Amazon Cognito 账户登录。
2. 在AWSKinesisVideoWebRTCDemo应用程序中，导航到信道配置页面，然后创建新的信令信道或选择现有信令信道。

### Note

目前，使用此 SDK 中的示例应用程序，您只能在 AWSKinesisVideoWebRTCDemoApp 中运行一个信令通道。

3. （可选）如果您想以查看器身份连接到此通道，请选择唯一的 Client Id (客户端 ID)。仅当多个查看器连接到一个通道时，才需要客户端 ID。这有助于通道的主设备识别各自的查看器。
4. 选择 AWS 区域 以及是否要发送音频或视频数据，或者两者兼而有之。
5. 要验证 peer-to-peer直播，请执行以下任一操作：

### Note

请务必在本演示中使用的所有客户端上指定相同的信令频道名称、AWS 区域、观众 ID 和 AWS 账户 ID。

- Peer-to-peer 在两台 iOS 设备之间进行流式传输：主设备和查看器
- 使用上述步骤，在两个 iOS 设备上下载、构建和运行 iOS WebRTC 开发工具包。
- 在一台 iOS 设备上以主模式打开AWSKinesisVideoWebRTCDemo应用程序（选择启动主模式），开始新的会话（信令信道）。

 Note

目前，任何给定信令通道只能有一个主设备。

- 在查看器模式下在第二台 iOS 设备上打开AWSKinesisVideoWebRTCDemo应用程序，连接上面步骤中启动的信令频道（会话）（选择启动查看器）。

验证查看器是否可以看见主设备的音频/视频数据。

- Peer-to-peer 在嵌入式 SDK 主设备和 iOS 设备查看器之间进行流式传输
- 在摄像头设备上，下载、构建 [亚马逊 Kinesis Video Streams 带有 WebRTC 开发工具包的 C 语言版本](#)，适用于嵌入式设备 并以主设备模式运行它。
- 使用上述步骤，在一个 iOS 设备上下载、构建和运行 iOS WebRTC 开发工具包。在查看器模式下打开此 iOS 设备上的AWSKinesisVideoWebRTCDemo应用程序，并确认 iOS 查看者可以看到嵌入式 SDK 主服务器的音频/视频数据。
- Peer-to-peer 在作为主设备的 iOS 设备和作为查看器的网络浏览器之间进行流式传输
- 使用上述步骤，在一个 iOS 设备上下载、构建和运行 iOS WebRTC 开发工具包。在主模式下打开此 iOS 设备上的AWSKinesisVideoWebRTCDemo应用程序（选择启动主模式），开始新的会话（信令信道）。
- 下载、构建并以查看者[带有 WebRTC 开发工具包的 Amazon Kinesis Video Streams 适用于网络应用程序 JavaScript](#) 身份运行，并验证查看者是否可以看见 Android 主服务器的音频/视频。JavaScript

## 为软件开发工具包配置 Amazon Cognito

### 先决条件

- 我们建议 XCode 检查、编辑和运行应用程序代码。我们推荐最新版本。
- 在示例代码中，您需要提供亚马逊 Cognito 凭证。

按照以下步骤设置 Amazon Cognito 用户池和身份池。

## 设置用户池

### 设置用户池

1. 登录 [Amazon Cognito 控制台](#) 并验证区域是否正确。
2. 在左侧导航栏中，选择“用户池”。
3. 在“用户池”部分，选择“创建用户池”。
4. 完成以下各节：
  - a. 第 1 步：配置登录体验-在 Cognito 用户池登录选项部分，选择相应的选项。  
选择下一步。
  - b. 步骤 2：配置安全要求-选择相应的选项。  
选择下一步。
  - c. 第 3 步：配置注册体验-选择相应的选项。  
选择下一步。
  - d. 步骤 4：配置消息传送-选择相应的选项。  
在 IAM 角色选择字段中，选择现有角色或创建新角色。  
选择下一步。
  - e. 第 5 步：集成您的应用程序-选择相应的选项。  
在“初始应用程序客户端”字段中，选择“机密客户端”。  
选择下一步。
  - f. 步骤 6：查看并创建-查看您在前面部分中的选择，然后选择创建用户池。
5. 在“用户池”页面上，选择您刚刚创建的池。  
  
复制用户池 ID 并记下来以备后用。在 `awsconfiguration.json` 文件中，这是 `CognitoUserPool.Default.PoolId`。
6. 选择“应用程序集成”选项卡，然后转到页面底部。
7. 在应用程序客户端列表部分，选择您刚刚创建的应用程序客户端名称。

复制客户端 ID 并记下来以备后用。在awsconfiguration.json文件中，这是CognitoUserPool.Default.AppClientId。

8. 出示客户机密并记下来以备后用。在awsconfiguration.json文件中，这是CognitoUserPool.Default.AppClientSecret。

## 设置身份池

### 设置身份池

1. 登录 [Amazon Cognito 控制台](#) 并验证区域是否正确。
2. 在左侧导航栏中，选择“身份池”。
3. 选择创建身份池。
4. 配置身份池。
  - a. 步骤 1：配置身份池信任-完成以下部分：
    - 用户访问权限-选择经过身份验证的访问权限
    - 经过身份验证的身份源-选择 Amazon Cognito 用户池选择下一步。
  - b. 步骤 2：配置权限-在“经过身份验证的角色”部分，填写以下字段：
    - IAM 角色-选择创建新的 IAM 角色
    - IAM 角色名称-输入名称并记下来供后续步骤使用。选择下一步。
  - c. 步骤 3：Connect 身份提供商-在“用户池详细信息”部分中，填写以下字段：
    - 用户池 ID-选择您之前创建的用户池。
    - 应用程序客户端 ID-选择您之前创建的应用程序客户端 ID。选择下一步。
  - d. 步骤 4：配置属性-在身份池名称字段中键入名称。  
选择下一步。

- e. 第 5 步：查看并创建-查看您在每个部分中的选择，然后选择创建身份池。
5. 在身份池页面上，选择您的新身份池。

复制身份池 ID 并记下来以备后用。在 `awsconfiguration.json` 文件中，这是 `CredentialsProvider.CognitoIdentity.Default.PoolId`。

6. 更新 IAM 角色的权限。
  - a. 登录 AWS Management Console 并打开 IAM 控制台，网址为 <https://console.aws.amazon.com/iam/>。
  - b. 在左侧的导航栏中，选择“角色”。
  - c. 找到并选择您在上面创建的角色。

 Note

如果需要，请使用搜索栏。

- d. 选择附加的权限策略。  
  
选择编辑。
- e. 选择 JSON 选项卡，然后将策略替换为以下内容：

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

选择下一步。

- f. 如果尚未选中“将此新版本设为默认版本”旁边的复选框。

选择保存更改。

## C SDK 的客户端指标

使用 Amazon Kinesis Video Streams with WebRTC 构建的应用程序由各种移动部件组成，包括网络、信号、候选交换、对等连接和数据交换。C 语言版 Kinesis Video Streams with WebRTC 支持各种客户端指标，使您能够监控和跟踪这些组件在应用程序中的性能和使用情况。支持的指标分为两大类：专为 Kinesis Video Streams 实现信令和网络而定义的自定义指标，以及源自 [W3C](#) 标准的媒体和数据相关协议特定指标。请注意，C 语言版 Kinesis Video Streams with WebRTC 目前仅支持 W3C 标准指标的子集。

### 主题

- [信令指标](#)
- [C SDK 支持 W3C 标准指标](#)

## 信令指标

信令指标可用于了解信令客户端在应用程序运行时的行为。您可以使用 `STATUS signalingClientGetMetrics (SIGNALING_CLIENT_HANDLE, PSignalingClientMetrics)` API 获取这些信令指标。下面是一个示例使用模式：

```
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));
```

`signalingClientStats` 的定义可以在 [Stats.h](#) 中找到。

目前支持以下信令指标：

指标	描述
cpApiCall延迟	计算控制面板 API 调用的延迟。使用指数移动平均线 (EM

指标	描述
	A) 进行计算。相关的电话包括：describeChannel、CreateChannel 和 DeleteChannel。getChannelEndpoint
dpApiCall延迟	计算数据面板 API 调用的延迟。使用指数移动平均线 (EMA) 进行计算。相关的呼叫包括: getIcConfig.
signalingClientUptime	这表示客户端对象存在的时间。每次调用此指标时，都会发出最新的正常运行时间值。
connectionDuration	如果连接已建立，则会发出连接处于活动状态的持续时间。否则，会发出值为 0。这与发出客户端正常运行时间信号不同，因为连接来来去去 signalingClientUptime，但表示客户端对象本身。
numberOfMessages已发送	当对等设备发送提议、应答或 ICE 候选项时，此值会更新。
numberOfMessages已收到	与“numberOfMessages已发送”不同，此指标会针对任何类型的信令消息进行更新。信令消息的类型可在 SIGNALING_MESSAGE_TYPE 中找到。

指标	描述
iceRefreshCount	调用时会 getIceConfig 递增。它的调用速率基于收到的 ICE 配置中的 TTL。每次收到一组新的 ICE 配置时，都会将计时器设置为下次刷新，前提是配置中凭证的有效期限减去一段宽限期。
numberOfErrors	该计数器用于跟踪信令客户端中生成的错误数量。跟踪在获取 ICE 配置、获取信令状态、跟踪信令指标、发送信令消息以及将信令客户端连接到 Web 套接字以发送/接收消息时产生的错误。
numberOfRuntime错误	该指标包括信令客户端核心运行时发生的错误。此处跟踪重新连接失败、消息接收失败和 ICE 配置刷新错误等情况。
numberOfReconnects	该指标在每次重新连接时都会递增。该指标有助于了解设置中网络连接的稳定性。

## C SDK 支持 W3C 标准指标

使用 WebRTC C SDK 构建的应用程序目前支持 [W3C](#) 标准指标的子集。这些指标为以下类别：

- 联网：
  - [ICE 候选项](#)：这些指标提供有关选定的本地和远程候选项的信息，以便在对等设备之间进行数据交换。这包括候选项的服务器来源、IP 地址、为通信选择的候选类型和候选优先级。这些指标可用作快照报告。
  - [ICE 服务器](#)：这些指标用于收集有关支持的不同 ICE 服务器的操作信息。这有助于尝试了解主要用于通信和连接检查的服务器。在某些情况下，如果候选项收集失败，则将有助于检查这些指标。

- [ICE 候选对](#)：这些指标用于了解对等设备之间交换的字节/数据包的数量以及与时间相关的测量值。
- 媒体和数据：
  - [远程入站 RTP](#)：这些指标代表发送方发送的数据流的端点视角。
  - [出站 RTP](#)：这些指标提供有关传出 RTP 流的信息。在分析不连贯的流或流停止时，它们也非常有用。
  - [入站 RTP](#)：这些指标提供有关传入媒体的信息。
  - [数据通道指标](#)：这些指标可以帮助您分析通过数据通道发送和接收的消息和字节数。可以使用通道 ID 提取指标。

您可以使用 STATUS `rtcPeerConnectionGetMetrics` (`PRtcPeerConnection`, `PRtcRtpTransceiver`, `PRtcStats`) API 收集与 ICE、RTP 和数据通道相关的指标。下面是一个用法示例：

```
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);
```

以下是另一个示例，展示了获取收发器相关统计数据的使用模式：

```
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);
```

在上面的示例中，如果 `rtcPeerConnection GetMetrics ()` 的第二个参数为 `NULL`，则返回列表中第一个收发器的数据。

[的定义 `rtcStatsObject` 可以在 `Stats.h` 中找到](#)，[的定义 `RtcStats` 可以在 `Include.H` 中找到](#)。

可以在 WebRTC C SDK 存储库的[示例目录 APIs](#) 和 [Kinesis Video Stream 演示存储库中找到不同指标的示例用法](#)。

使用 WebRTC C SDK 构建的应用程序目前支持以下 [W3C](#) 标准指标。

## 主题

- [网络连接](#)
- [媒体](#)
- [数据频道](#)

## 网络连接

ICE 服务器指标：

指标	描述	
URL	正在追踪的 STUN/TURN 服务器的网址	
端口	客户端使用的端口号	
协议	从 ICE 服务器 URI 中提取的传输协议。如果值为 UDP，ICE 会尝试翻转 UDP，否则 ICE 会尝试翻转。TCP/TLS. If the URI does not contain transport , ICE tries TURN over UDP and TCP/TLS如果是 STUN 服务器，此字段为空。	
发送的请求总数	每个 srflx 候选请求以及从 TURN 候选项发送绑定请求时，该值都会更新。	
收到的回复总数	每次收到 STUN 绑定响应时，该值都会更新。	
往返总时间	每次收到请求的等效响应时，该值都会更新。在以校验和为密钥的哈希映射中跟踪请求数据包。	

ICE 候选项统计信息：仅包括有关所选候选项（本地和远程）的信息。

指标	描述
address	表示本地和远程候选的 IP 地址。
port	候选端口号
protocol	用于获取候选项的协议。有效值为 UDP/TCP。
candidateType	所选候选项的类型 - 主机、srflx 或中继。
priority	所选本地和远程候选项的优先级。
url	所选本地候选项的来源。这表明所选候选项是从 STUN 服务器还是 TURN 服务器接收。
relayProtocol	如果使用 TURN 服务器来获取选定的本地候选项，则此字段表示使用了哪种协议来获取它。有效值为 TCP/UDP。

ICE 候选对统计信息：仅包含有关所选候选对的信息。

指标	描述
localCandidateId	配对中所选本地候选项的 ID。
remoteCandidateId	配对中所选远程候选项的 ID。
state	正在检查的候选对状态。
nominated	设置为 TRUE，因为正在提取选定候选对的统计信息。

指标	描述
packetsSent	发送的数据包数。这是在 writeFrame 调用中的 . 调用中计算得出的。这些信息也可以从传出的 RTP 统计数据中提取，但是由于 Ice 候选对包含 lastPacketSent 时间戳，因此计算两个时间点之间发送的数据包数量可能会很有用。
packetsReceived	每次调用时 incomingDataHandler 都会更新。
bytesSent	这是在 writeFrame() 调用中的 iceAgentSendPacket() 中计算得出的。这在计算比特率时很有用。当前，这还包括标头和填充，因为 ICE 层忽略了 RTP 数据包格式。
bytesReceived	每次调用时 incomingDataHandler 都会更新。当前，这还包括标头和填充，因为 ICE 层忽略了 RTP 数据包格式。
lastPacketSentTimestamp	每次发送数据包时都会更新此信息。它可以与 packetsSent 和记录的开始时间结合使用，应用于当前数据包传输速率。

指标	描述
lastPacketReceived Timestamp	在 <code>incomingDataHandler()</code> 中接收数据时更新此信息。可以与 <code>packetsReceived</code> 结合使用来推断当前的数据包接收速率。开始时间必须记录在应用程序层的 <code>transceiverOnFrame()</code> 回调中。
firstRequestTimestamp	当第一个 STUN 绑定请求在 <code>iceAgentSendStunPacket()</code> 中成功发出时记录此信息。它可以与 <code>lastRequestTimestamp</code> 和 <code>requestsSent</code> 一起使用，以查找 STUN 绑定请求之间的平均时间。
lastRequestTimestamp	每次 STUN 绑定请求在 <code>iceAgentSendStunPacket()</code> 中成功发出时都会记录此信息。
lastResponseTimestamp	每次收到 STUN 绑定响应时都会记录此信息。
totalRoundTrip时间	在收到请求的绑定响应时更新。请求和响应根据校验和映射到哈希表中。
currentRoundTrip时间	当收到有关候选对请求的绑定响应时，最新的往返时间会更新。
requestsReceived	在收到每个 STUN 绑定请求时，该计数器都会更新。

指标	描述
requestsSent	在 iceAgentSendStunPacket() 中发出每个 STUN 绑定请求时，该计数器都会更新。
responsesSent	在 handleStunPacket() 中为响应绑定请求而发出每个 STUN 绑定响应时，该计数器都会更新。
responsesReceived	在 handleStunPacket() 中收到每个 STUN 绑定响应时，该计数器都会更新。
packetsDiscardedOn发送	数据包发送失败时更新。换句话说，它会在 iceUtilsSendData() 失败时更新。这有助于确定在特定持续时间内丢弃的数据包的百分比。
bytesDiscardedOn发送	数据包发送失败时更新。换句话说，它会在 iceUtilsSendData() 失败时更新。这在确定特定持续时间内丢弃的数据包的百分比时很有用。请注意，计数器还包括数据包的标头。

## 媒体

### 出站 RTP 统计信息

指标	描述
voiceActivityFlag	这目前是 Include.h 中定义的 RtcEncoderStats 的组成部分。如果最后一个音频数据包包含语音，则该标志设置为 TRUE。样本中目前未设置该标志。
packetsSent	这表示为选定的 SSRC 发送的 RTP 数据包总数。这是 <a href="https://www.w3.org/TR/webrtc-stats/#sentrtptimestats-dict">https://www.w3.org/TR/webrtc-stats/#sentrtptimestats-dict</a> 并包含在出站统计数据中。每次调用 writeFrame() 时，该值都会递增。
bytesSent	发送的总字节数，不包括 RTP 标头和填充。每次 writeFrame 调用时，该值都会更新。
encoderImplementation	它由应用程序层作为 RtcEncoderStats 对象的一部分进行更新。
packetsDiscardedOn发送	如果 ICE 代理在 iceAgentSendPacket 调用中由于任何原因未能发送加密的 RTP 数据包，则会更新此字段。
bytesDiscardedOn发送	如果 ICE 代理在 iceAgentSendPacket 调用中由于任何原因未能发送加密的 RTP 数据包，也会更新此字段。
framesSent	只有当媒体流堆栈类型为 MEDIA_STREAM_TRACK

指标	描述
	<p>_KIND_VIDEO 时，该值才会递增。</p>
hugeFramesSent	<p>当帧大小是帧平均大小的2.5倍时，此计数器会更新。帧的大小是通过计算 fps（基于上次已知的帧计数时间和按时间间隔编码的帧数）并使用应用程序 RtcEncoderStats 设置的 targetBitRate 来获得的。</p>
framesEncoded	<p>仅在成功对帧进行编码后，才会针对视频轨道更新此计数器。它会在每次 writeFrame 调用时更新。</p>
keyFramesEncoded	<p>仅在成功编码关键帧后，才会针对视频轨道更新此计数器。它会在每次 writeFrame 调用时更新。</p>
framesDiscardedOn发送	<p>当由于 iceAgentS endPacket 调用失败而导致帧发送失败时，会更新此信息。一个帧由一组数据包组成，当前，如果在 framesDiscardedOn发送时由于错误而丢弃了任何数据包，则发送将失败。</p>
frameWidth	<p>理想情况下，这表示最后一个编码帧的帧宽。目前，应用程序将其设置为 RtcEncoderStats *的一部分的值，意义不大。</p>

指标	描述
frameHeight	理想情况下，这表示最后一个编码帧的帧高。目前，该值由应用程序设置为值的一部分 RtcEncoderStats，意义不大。
frameBitDepth	这表示最后一个编码帧的每像素宽度的位深度。目前，这是由应用程序设置为出站统计信息的一部分，RtcEncoderStats 并转换为出站统计信息。
nackCount	每次收到 RTP 数据包上的 NACK 并重新尝试发送该数据包时，都会更新此值。堆栈支持在收到 NACK 时重新传输数据包。
firCount	该值将在收到 FIR 数据包时更新 (onRtcpPacket->onFIRPacket rtCP)。它表示流落后且必须跳帧才能赶上的频率。目前无法解码 FIR 数据包以提取字段，因此，即使设置了计数，也不会采取任何操作。
pliCount	该值将在收到 PLI 数据包时更新 (onRtcpPacket-> PLIPacket onrtCP)。它表示在一帧或多帧中丢失了一定数量的编码视频数据。

指标	描述
sliCount	该值将在收到 SLI 数据包时更新 (onRtcpPacket-> SLIPacket onrtCP)。它表示丢包影响单个帧的频率。
qualityLimitationResolution更改	目前，堆栈支持此指标，但是，并非每个编码帧的帧宽和帧高都受到监控。
lastPacketSentTimestamp	发送最后一个数据包的时间戳。它会在每次 writeFrame 调用时更新。
headerBytesSent	为此 SSRC 发送的 RTP 标头和填充字节总数，不包括实际的 RTP 有效负载。
bytesDiscardedOn发送	当由于 iceAgentSend数据包调用失败而导致帧发送失败时，会更新此信息。帧由一组数据包组成，这些数据包又由字节组成，当前，如果在 bytesDiscardedOn发送时由于错误而丢弃了任何数据包，则发送将失败。
retransmittedPacketsSent	收到时重新传输的数据包的 PLI/SLI/NACK 数量。目前，由于不支持基于 PLI 和 SLI 的重新传输，堆栈仅计算 NACK 重新发送的数据包。

指标	描述
retransmittedBytesSent	接收时重新传输的字节数。PLI/SLI/NACK目前，由于不支持基于 PLI 和 SLI 的重新传输，堆栈仅计算 NACK 重新发送的字节数。
targetBitrate	这是在应用程序级别设置的。
totalEncodedBytes目标	每次对帧进行编码时，目标帧大小（以字节为单位）都会增加该值。这是使用框架结构中的大小参数进行更新的。
framesPerSecond	这是根据最后一个已知编码帧的记录时间和一秒钟内发送的帧数计算得出的。
totalEncodeTime	在应用程序中将其设置为任意值，并在内部转换为出站统计信息。
totalPacketSend延迟	由于 Packet 会立即发送 iceAgentSend数据包，因此当前将其设置为 0。

#### 远程入站 RTP 统计信息：

指标	描述
roundTripTime	该值是从 RTCP 接收方收到类型为 201 的 RTCP 报文(接收方报告)的报告中提取的。该报告包含诸如上次发送方报告和自上次发送方报告以来的延迟等详细信息，以计算往返时

指标	描述
	间。大约每 200 毫秒生成一次发送方报告，其中包括从出站统计信息中提取的信息，例如发送的数据包数量和发送的字节数。
totalRoundTrip时间	计算出的往返时间总和
fractionLost	表示自上一次发送方/接收方 reportfractionLost 发送以来 SSRC 丢失的 RTP 数据包的比例。
reportsReceived	每次收到接收方报告类型数据包时都会更新。
roundTripTime测量	表示已收到的包含有效往返时间的 SSRC 报告总数。但是，目前这个值仍是递增的，所以它的含义与 reportsReceived 相同。

#### 入站 RTP 统计信息：

指标	描述
packetsReceived	当收到特定 SSRC 的数据包时，计数器会更新。
jitter	此指标表示特定 SSRC 的数据包抖动（以秒为单位进行衡量）。
jitterBufferDelay	该指标表示抖动缓冲区中每个数据包所花费的时间总和。

指标	描述
jitterBufferEmitted计数	来自抖动缓冲区的音频样本或视频帧的总数。
packetsDiscarded	当抖动缓冲区已满且无法将数据包推入其中时，计数器会更新。这可以用来计算在固定时间内丢弃的数据包的百分比。
framesDropped	当调用 onFrameDroppedFunc() 时会更新此值。
lastPacketReceived Timestamp	表示收到此 SSRC 的最后一个数据包的时间戳。
headerBytesReceived	收到 RTP 数据包后，计数器即会更新。
bytesReceived	接收的字节数。其中不包括标头字节。此指标可用于计算传入比特率。
packetsFailedDecryption	当 SRTP 数据包的解密失败时，该值会递增。

## 数据频道

数据渠道指标：

指标	描述
label	标签是正在检查的数据通道的名称。
protocol	由于我们的堆栈使用 SCTP，因此协议设置为恒定 SCTP。

指标	描述	
<code>dataChannelIdentifier</code>	用于唯一标识数据通道的偶数或奇数标识符。如果 SDK 是提议者，则更新为奇数值；如果 SDK 是应答者，则更新为偶数值。	
<code>state</code>	查询统计信息时数据通道的状态。目前，支持的两种状态是 <code>RTC_DATA_CHANNEL_STATE_CONNECTING</code> （通道创建时）和 <code>RTC_DATA_CHANNEL_STATE_OPEN</code> （在 <code>onOpen()</code> 事件中设置）。	
<code>messagesSent</code>	当 SDK 通过数据通道发送消息时，计数器会更新。	
<code>bytesSent</code>	该计数器用发送出去的消息中的字节更新。这可以用来了解有多少字节由于失败而没有发送，也就是说，可以用来了解发送的字节百分比。	
<code>messagesReceived</code>	该指标在 <code>onMessage()</code> 回调中递增。	
<code>bytesReceived</code>	该指标在 <code>onMessage()</code> 回调中生成。	

# 使用带有 WebRTC 的 Amazon Kinesis Video Streams 来摄取和存储媒体

Amazon Kinesis Video Streams 支持通过 WebRTC 将视频和音频实时传输到云端进行存储、播放和分析处理。客户可以使用我们增强的 WebRTC SDK 和 APIs 云来实现实时流媒体以及向云端提取媒体。

首先，您可以在任何安全摄像头 AWS IoT 或带有视频传感器的设备上安装带有 [WebRTC SDK 的 Amazon Kinesis Video Streams](#)，然后使用 [APIs](#) 我们的来启用延迟低于 1 秒的媒体流以及云端的摄取和存储。摄取后，您可以通过我们的 easy-to-use APIs 网站访问您的数据。Amazon Kinesis Video Streams 使您能够播放用于直播和点播观看的视频，并通过与亚马逊 Rekognition Video 和 AI 集成，快速构建利用计算机视觉和视频 SageMaker 分析的应用程序。

## 主题

- [支持多画面的 WebRTC 收录 \(预览版\)](#)
- [API 操作](#)
- [什么是带有 WebRTC 摄取和存储功能的 Amazon Kinesis Video Streams ?](#)
- [创建信令通道](#)
- [创建视频流](#)
- [授予权限](#)
- [配置目的地](#)
- [提取媒体](#)
- [播放摄取的媒体](#)
- [Connect 到存储会话](#)
- [解决与存储会话连接的问题](#)

## 支持多画面的 WebRTC 收录 (预览版)

### Note

根据服务条款的定义，支持多画面浏览器 (预览版) 的 WebRTC 收录以预览版形式提供 AWS，可能会发生变化。它目前仅在 us-east-1 (IAD) 中可用。

要参与预览，<kvs-webrtc-multi-view#####-preview@amazon.com>。

Amazon Kinesis Video Streams 现在允许多个观众通过 WebRTC 加入实时视频直播会话。每位观众都可以观看流媒体视频，也可以与所有参与者接收和共享音频，同时将媒体上传到云端。

## API 操作

使用以下 API 操作配置 Amazon Kinesis Video Streams WebRTC 提取：

- [DescribeMappedResourceConfiguration](#)
- [DescribeMediaStorageConfiguration](#)
- [JoinStorageSession](#)
- [JoinStorageSessionAsViewer](#)
- [UpdateMediaStorageConfiguration](#)

## 什么是带有 WebRTC 摄取和存储功能的 Amazon Kinesis Video Streams ？

Amazon Kinesis Video Streams 支持通过 WebRTC 将视频和音频实时传输到云端进行存储、播放和分析处理。本主题将提供设置和使用我们的 WebRTC SDK 和 APIs 云端的 step-by-step 说明，以实现云端的实时流媒体和媒体摄取。这些说明包括使用 AWS Command Line Interface 和 Kinesis Video Streams 控制台的指南。

首次使用 Amazon Kinesis Video Streams with WebRTC 前，请参阅 [the section called “设置一个 AWS 账户”](#)。

### 了解 WebRTC 的摄取和存储

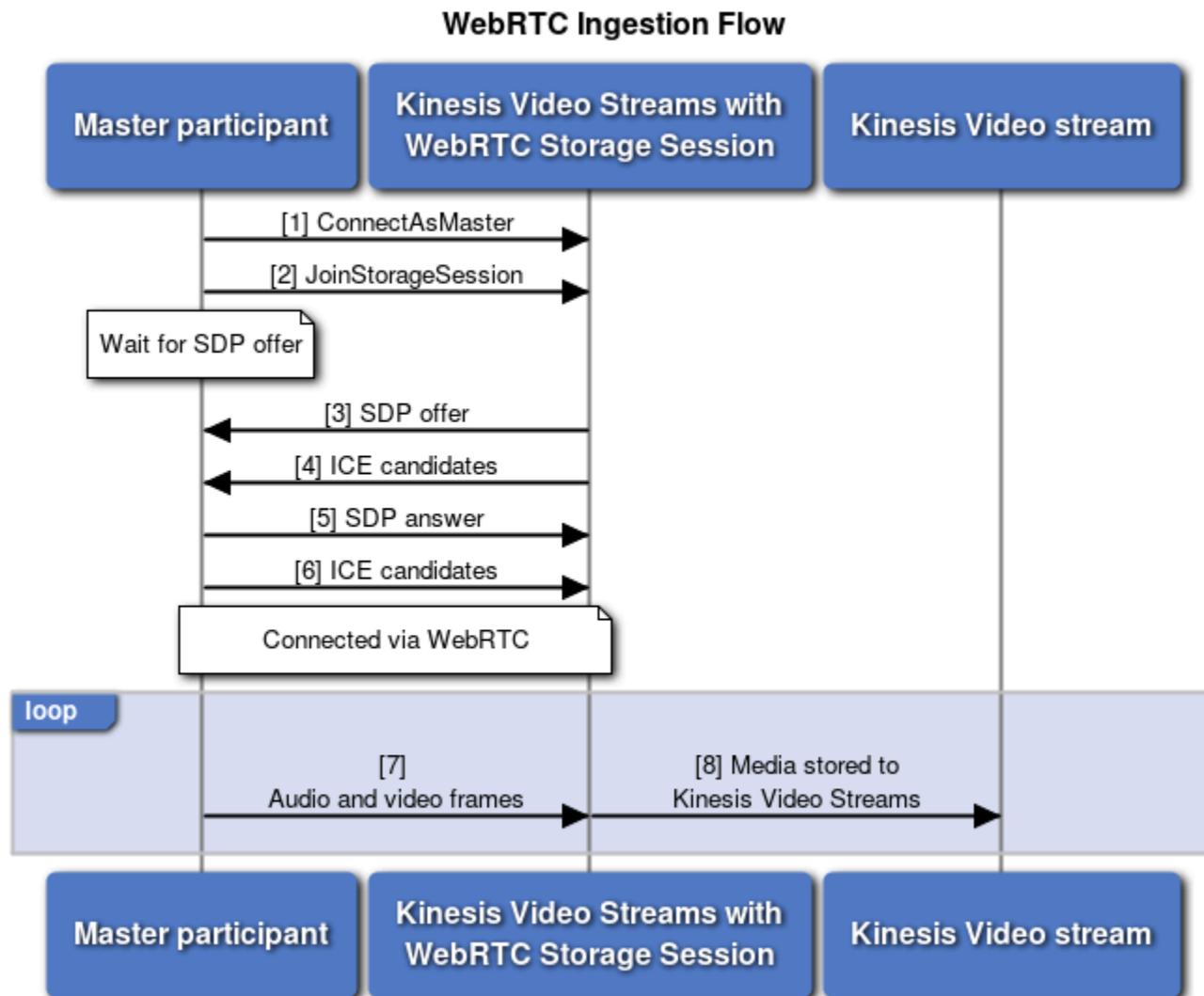
以下各节说明了带有 WebRTC 的 Kinesis Video Streams 中可用的不同摄取和存储选项。

#### 主题

- [仅限主参与者](#)
- [主人和观众参与者在一起](#)

## 仅限主参与者

主学员首先通过 WebRTC 信令连接到 Kinesis Video Streams。 [the section called “ConnectAsMaster”](#) 接下来，他们调用 [JoinStorageSession](#) API 让存储会话启动 WebRTC 连接。建立 WebRTC 连接后，媒体将被提取到配置的 Kinesis 视频流中。



## 主人和观众参与者在一起

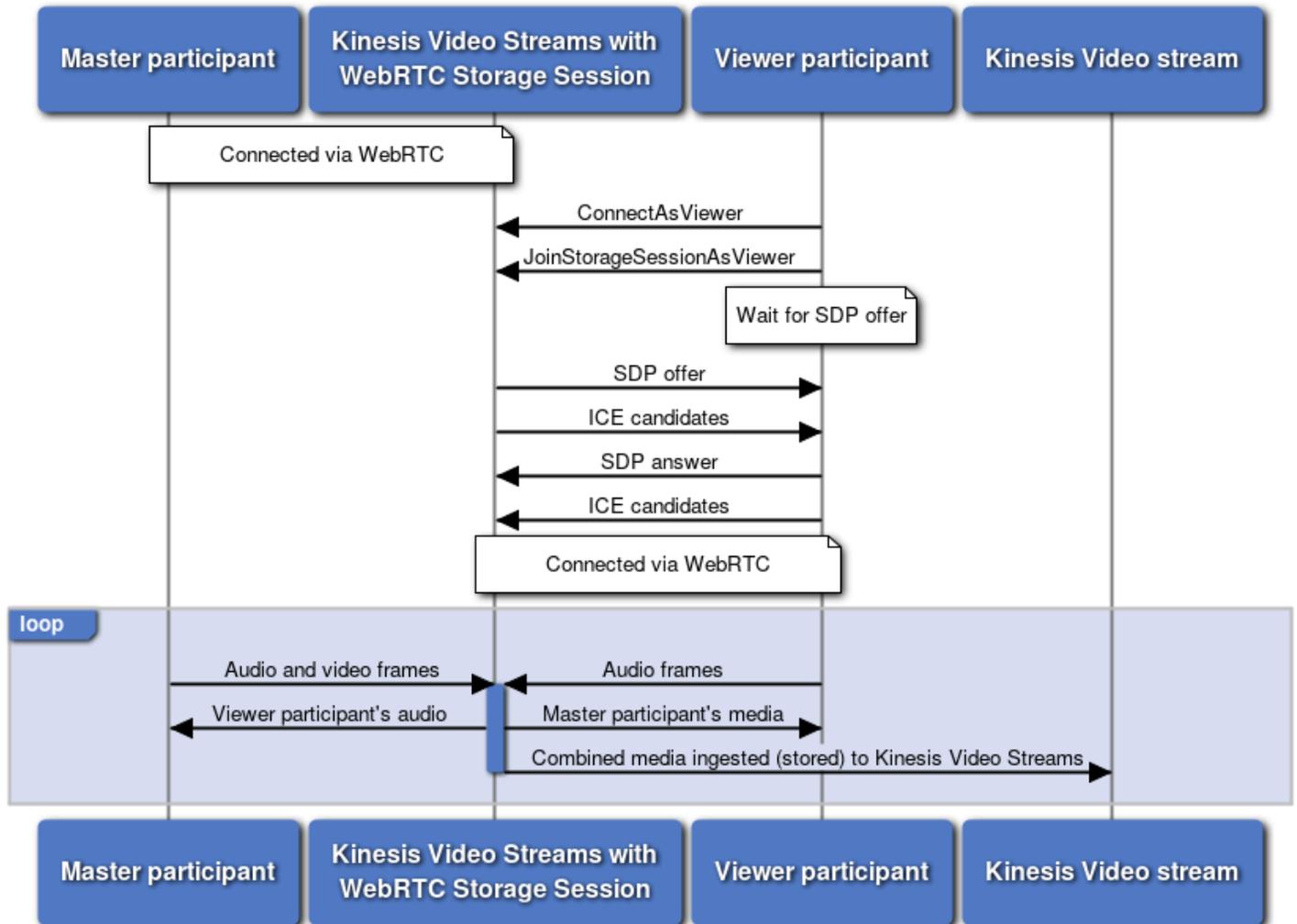
### Note

根据服务条款的定义，支持多画面浏览器（预览版）的 WebRTC 收录以预览版形式提供 AWS，可能会发生变化。它目前仅在 us-east-1 (IAD) 中可用。  
要参与预览，<kvs-webrtc-multi-view#####-preview@amazon.com>。

观众参与者首先通过 WebRTC 信令连接到 Kinesis Video Streams。 [the section called “ConnectAsViewer”](#) 接下来，他们调用 [JoinStorageSessionAsViewer](#) API 让存储会话启动 WebRTC 连接。建立 WebRTC 连接后，只要主参与者在场，来自主参与者和所有观众参与者的组合媒体就会被收录到配置的 Kinesis 视频流中。

存储会话合并所有观看者参与者的音频并将其转发给主参与者。查看者参与者接收来自主参与者的组合媒体以及存储会话中来自任何其他查看者参与者的音频。

WebRTC Ingestion Flow with Viewer



## 与存储会话建立 WebRTC 连接

由于存储会话位于 Amazon 网络内，因此存储会话只会向参与者发送 relay (TURN) 候选会话。如果参与者的网络允许，则可以使用 srflx (STUN) 候选项连接到存储会话。换句话说，从参与者的角度来看，当地提名的移民局候选人可以是 srflx 或 relay，而远程移民局候选人始终是 relay。

要优化连接时间，请勿将 host 候选人发送到存储会话。还需要 Trickle ICE 使用存储会话。

[the section called “对存储会话连接进行故障排除”](#)要解决与存储会话的连接问题，请参阅。

## 创建信令通道

带有 WebRTC 信令通道的 Kinesis Video Streams 可促进在 WebRTC 客户端之间建立和维护连接所需的信令消息交换。peer-to-peer 它处理会话描述协议 (SDP) 报价和会话参数答案的协商，以及交互式连接建立 (ICE) 候选网络信息的交换。

要创建信令通道，请调用 [CreateSignalingChannel](#) API。本页将向您展示如何使用 AWS Management Console AWS CLI、和其中一个来调用该 API AWS SDKs。

### Important

记下频道 ARN，稍后你会用到的。

### AWS Management Console

执行以下操作：

1. [在 HOME/#/SignalingChannels 上打开 Kinesis Video Streams 信令频道控制台](https://console.aws.amazon.com/kinesisvideo/) <https://console.aws.amazon.com/kinesisvideo/>。
2. 选择 Create signaling channel (创建信令通道)。
3. 在创建新的信令通道页面上，输入信令通道的名称。

将默认 Time-to-live (Ttl) 值保留为 60 秒。

选择 Create signaling channel (创建信令通道)。

4. 一旦创建信令通道，在通道的详细信息页面上查看其详细信息。

### AWS CLI

确认您已 AWS CLI 安装并配置了。有关更多信息，请参阅 [AWS Command Line Interface 《用户指南》](#)。

有关安装说明，请参阅 [《AWS Command Line Interface 用户指南》](#)。安装完成后，AWS CLI 使用凭据和区域进行 [配置](#)。

或者，打开已 AWS CLI 安装和配置的 AWS CloudShell 终端。有关更多信息，请参阅 [AWS CloudShell 用户指南](#)。

使用以下命令运行以下 [Create-Signaling-Channel](#) 命令：AWS CLI

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

响应看起来与以下内容类似：

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

## AWS SDK

此代码片段向您展示了如何使用适用于 v2 的 SDK 创建带有 WebRTC 信令通道的 Kinesis Video Streams。AWS JavaScript 语法将与其他语法不同 AWS SDKs，但一般流程将相同。在上查看完整的代码示例[GitHub](#)。

创建 Kinesis Video Streams 客户端。这是用于调用 CreateSignalingChannel API 的客户端。

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

使用客户端调用 CreateSignalingChannel API。

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

打印响应。

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

包含此代码示例的实时网页可在上使用[GitHub](#)。输入您的区域、AWS 凭证和信令频道的名称。

选择“创建频道”。

## 创建视频流

按照以下步骤创建要接收媒体的流。如果您已经创建了目标流，请跳过此步骤。

### Important

WebRTC Ingestion 需要数据保留率大于 0 的 Kinesis 视频流。最短为 1 小时。

要创建直播，请使用 AWS Management Console AWS CLI、或其中一个 AWS SDK 调用 [CreateStream](#) API。

### Important

记下直播 ARN，稍后你会用到的。

## AWS Management Console

执行以下操作：

1. 在家中打开 Kinesis Video Streams 控制台 <https://console.aws.amazon.com/kinesisvideo/台/>。
2. 在 Video streams (视频流) 页面上，选择 Create video stream (创建视频流)。
3. 在“创建新的视频流”页面上 *YourStreamName*，输入直播名称。保持“默认配置”按钮处于选中状态。

这将创建一个数据保留率大于 0 的流。

选择 Create video stream (创建视频流)。

4. Kinesis Video Streams 创建直播后，请查看页面 *YourStreamName* 上的详细信息。

## AWS CLI

确认您已 AWS CLI 安装并配置了。有关更多信息，请参阅 [AWS Command Line Interface 《用户指南》](#)。

有关安装说明，请参阅《[AWS Command Line Interface 用户指南](#)》。安装完成后，AWS CLI使用凭据和区域进行[配置](#)。

或者，打开已 AWS CLI 安装和配置的 AWS CloudShell 终端。有关更多信息，请参阅 [AWS CloudShell 用户指南](#)。

使用以下Create-Stream命令运行以下命令 AWS CLI：

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

响应看起来与以下内容类似：

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

## AWS SDK

此代码片段向您展示了如何使用 AWS 适用 JavaScript 于 v2 的软件开发工具包创建 Kinesis 视频流。语法将与其他语法不同 AWS SDKs，但一般流程将相同。在上查看完整的代码示例[GitHub](#)。

创建 Kinesis Video Streams 客户端。这是用于调用 [CreateStream](#)API 的客户端。

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

使用客户端调用 CreateStream API。

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

打印响应。

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

包含此代码示例的实时网页可在上使用 [GitHub](#)。输入您的区域、AWS 凭证和信令频道的名称。

展开 WebRTC 摄取和存储节点，键入直播名称，然后选择创建直播。弹出窗口会询问您想要保留直播数据的小时数。输入一个大于 0 的值，然后选择创建流。

## 授予权限

你必须向你的 IAM 角色授予直播权限才能通过 WebRTC 在 Amazon Kinesis Video Streams 中推送直播。

### Note

服务角色是由一项服务担任、代表您执行操作的 [IAM 角色](#)。IAM 管理员可以在 IAM 中创建、修改和删除服务角色。有关更多信息，请参阅《IAM 用户指南》中的 [创建向 AWS 服务委派权限的角色](#)。

主角色和查看者角色还必须具备 DescribeStreamGetDataEndpoint、以及向 Kinesis Video Streams 收录媒体的 PutMedia 权限。

请参阅下面针对主参与者的 IAM 政策示例：

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

## 配置目的地

创建 Kinesis Video Streams 资源后，你需要告诉信令频道将其保存到哪个直播中。

如果要删除信令频道或视频流，必须先将其取消关联。请参阅[the section called “取消信令频道和直播的关联”](#)。

## 链接信令频道和直播

使用 [UpdateMediaStorageConfiguration](#) API 并输入要关联 ARNs 的 Kinesis Video Streams 资源。

### Important

启用后 StorageStatus，将不再存在直接 peer-to-peer（主查看器）连接。对等方直接连接到存储会话。您必须调用 JoinStorageSession API 才能触发 SDP 报价发送并在对等方和存储会话之间建立连接。

## AWS Management Console

### Note

Kinesis Vid AWS Management Console eo Streams 目前不支持此操作。

打开已 AWS CLI 安装和配置的 AWS CloudShell 终端。有关更多信息，请参阅 [AWS CloudShell 用户指南](#)。

按照 AWS CLI 选项卡中的说明进行操作。

## AWS CLI

确认您已 AWS CLI 安装并配置了。有关更多信息，请参阅 [AWS Command Line Interface](#) 文档。

有关安装说明，请参阅《[AWS Command Line Interface 用户指南](#)》。安装完成后，AWS CLI 使用凭据和区域进行 [配置](#)。

或者，打开已 AWS CLI 安装和配置的 AWS CloudShell 终端。有关更多信息，请参阅 [AWS CloudShell 用户指南](#)。

在 Update-Media-Storage-Configuration 以下位置运行命令 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

此代码片段向您展示如何使用适用于 v2 的 SDK 配置信令通道以将媒体推送到指定的 Kinesis 视频流。AWS JavaScript 语法将与其他语法不同 AWS SDKs，但一般流程将相同。在上查看完整的代码示例 [GitHub](#)。

创建 Kinesis Video Streams 客户端。这是用于调用 [UpdateMediaStorageConfiguration](#) API 的客户端。

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

使用客户端调用 UpdateMediaStorageConfiguration API。

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

包含此代码示例的实时网页可在上使用[GitHub](#)。输入您的区域、AWS 凭证和信令频道的名称。

展开 WebRTC 摄取和存储节点，键入直播名称，然后选择更新媒体存储配置。该频道将被配置为将媒体推送到指定流。

## 取消信令频道和直播的关联

### Important

只有取消信令频道或直播之间的关联，才能将其删除。

如果您不希望将信令频道的媒体提取到直播中，请使用 [UpdateMediaStorageConfiguration](#) API 取消关联 Kinesis Video Streams 资源。取消频道链接后，可以恢复直接 peer-to-peer 连接。

### AWS Management Console

### Note

Kinesis Video Streams AWS Management Console 目前不支持此操作。

打开已 AWS CLI 安装和配置的 AWS CloudShell 终端。有关更多信息，请参阅 [AWS CloudShell 用户指南](#)。

按照 AWS CLI 选项卡中的说明进行操作。

### AWS CLI

确认您已 AWS CLI 安装并配置了。有关更多信息，请参阅 [AWS Command Line Interface](#) 文档。

有关安装说明，请参阅《[AWS Command Line Interface 用户指南](#)》。安装完成后，AWS CLI使用凭据和区域进行[配置](#)。

或者，打开已 AWS CLI 安装和配置的 AWS CloudShell 终端。有关更多信息，请参阅 [AWS CloudShell 用户指南](#)。

在Update-Media-Storage-Configuration以下位置运行命令 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

此代码片段向您展示如何使用适用于 v2 的 SDK 配置信令通道以将媒体推送到指定的 Kinesis 视频流。AWS JavaScript语法将与其他语法不同 AWS SDKs，但一般流程将相同。在上查看完整的代码示例[GitHub](#)。

创建 Kinesis Video Streams 客户端。这是用于调用 [UpdateMediaStorageConfiguration](#)API 的客户端。

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

使用客户端调用 UpdateMediaStorageConfiguration API。

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

```
.promise();
```

包含此代码示例的实时网页可在上使用[GitHub](#)。输入您的区域、AWS 凭证和信令频道的名称。

展开 WebRTC 摄取和存储节点，验证“流名称”字段是否为空，然后选择“更新媒体存储配置”。该频道将不再配置为向指定直播推送媒体。

## 提取媒体

以下限制已到位：

- 会话时长：一小时，最大值
- 信令通道：启用存储配置后，每个账户最多 100 个

主题

- [从浏览器提取媒体](#)
- [从 WebRTC C SDK 中提取媒体](#)
- [将观众添加到摄取会话中](#)

## 从浏览器提取媒体

### Important

Chrome 是目前唯一支持的浏览器。

1. [在示例页面中打开带有 WebRTC 软件开发工具包的 Amazon Kinesis Video Streams。JavaScript](#)
2. 完成以下信息：

- KVS 终端节点-在“区域”字段中，选择您的区域。

例如 us-west-2。

- AWS 凭据

填写以下字段：

- 访问密钥 ID

- 秘密访问密钥
  - 会话令牌-示例应用程序支持临时和长期证书。如果您使用的是长期 IAM 凭证，请将此字段留空。有关更多信息，请参阅 [IAM 中的临时安全证书](#)。
  - 信令信道-在“信道名称”字段中，键入您之前配置的信令信道的名称。有关更多信息，请参阅 [the section called “配置目的地”](#)。
  - 曲目-选择“发送视频”和“发送音频”。
  - WebRTC 摄取和存储-展开节点并选择“自动确定摄取模式”。此选项使示例应用程序调用 [DescribeMediaStorageConfiguration](#) API 以确定在何种模式下运行。
3. 选择启动主设备。

如果使用 [DescribeMediaStorageConfiguration](#) API 将信令通道配置为摄取，则示例应用程序将在连接到信令通道后立即自动调用 [JoinStorageSession](#) API 以启动 WebRTC 摄取工作流程。

## 从 WebRTC C SDK 中提取媒体

按照 [the section called “适用于嵌入式设备的 C SDK”](#) 过程生成示例应用程序。

1. 使用您的 AWS 账户 凭据设置您的环境：

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

如果您使用的是临时 AWS 证书，请同时导出您的会话令牌：

```
export AWS_SESSION_TOKEN=YourSessionToken
```

2. 运行示例：

主示例

导航到该build文件夹，并使用“1”作为第二个参数。类型：

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

GStreamer 主样本

导航到该build文件夹，然后使用 audio-video-storage “” 作为第二个参数。类型：

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

这将启动 WebRTC 提取。

#### Note

您提供的信令通道必须配置为用于存储。使用 [DescribeMediaStorageConfigurationAPI](#) 进行确认。

## 将观众添加到摄取会话中

#### Note

根据服务条款的定义，支持多画面浏览器（预览版）的 WebRTC 收录以预览版形式提供 AWS，可能会发生变化。它目前仅在 us-east-1 (IAD) 中可用。  
要参与预览，<kvs-webrtc-multi-view#####-preview@amazon.com>。

信号通道进入 WebRTC 摄取模式后，观看者参与者将不再直接连接到主参与者。查看者参与者直接连接到存储会话。查看者参与者接收主参与者发送的媒体，观看者参与者可以向主参与者发送可选音频。只要主参与者已连接到存储会话，观众发回的任何音频都将发送给连接到存储会话并收录到 Kinesis Video Stream 的所有其他对等方。

以下限制已到位：

- 最大观看人数：3
- 在没有 @@ 主参与者在场的情况下，查看者参与者可以连接到存储会话的最长时间：3 分钟

#### Important

如果查看者断开与存储会话的连接（关闭对等连接），则其配额（查看者限制）将在 1 分钟内保持消耗。在这个 1 分钟内，查看者可以使用相同的客户端 ID 调用此 API 以重新加入会话，而无需消耗额外的观看者配额。1 分钟后，观众配额已释放，可供其他观众加入。

## 浏览器

### Important

Chrome 是唯一支持的浏览器。

1. [在示例页面中打开带有 WebRTC SDK 的亚马逊 Kinesis Video Streams 中的另一个选项卡。](#) [JavaScript](#) 上一页的所有信息都将自动填充。如果不是，请填写以下信息：

- KVS 终端节点-在“区域”字段中，选择您的区域。

例如 us-west-2。

- AWS 凭据

填写以下字段：

- 访问密钥 ID
- 秘密访问密钥
- 会话令牌-示例应用程序支持临时和长期证书。如果您使用的是长期 IAM 凭证，请将此字段留空。有关更多信息，请参阅 [IAM 中的临时安全证书](#)。
- 信令信道-在“信道名称”字段中，键入您之前配置的信令信道的名称。有关更多信息，请参阅 [the section called “配置目的地”](#)。
- 曲目-选择“发送音频”。请注意，如果选中“发送视频”，则在选择“启动查看器”时将自动取消选中。
- WebRTC 摄取和存储-展开节点并选择“自动确定摄取模式”。此选项使示例应用程序调用 [DescribeMediaStorageConfigurationAPI](#) 以确定在何种模式下运行。

2. 选择“启动查看器”。

应用程序在连接到信令通道后立即自动调用 [JoinStorageSessionAsViewerAPI](#)，以触发会话向查看者发送的 SDP 报价。

### Note

在 peer-to-peer WebRTC 中，查看者参与者是控制对等方，主参与者是受控对等方。在 WebRTC 摄取模式下，存储会话现在是控制对等体。连接到信令并调用后

[JoinStorageSessionAsViewer](#)，查看者需要响应SDP报价，并通过WebRTC建立与存储会话的连接。

### Note

存储会话只会发送TURN候选人。从参与者的角度提名一对ICE候选人时，远程候选人将始终是类型relay的。

## 播放摄取的媒体

您可以通过在控制台中查看媒体数据来使用媒体数据，也可以创建使用超文本直播 (HLS) 从流中读取媒体数据的应用程序。

### 在控制台中查看媒体

在另一个浏览器选项卡中，打开 AWS Management Console。在 Kinesis Video Streams 控制面板中，[选择视频流](#)。

在直播列表中选择您的直播名称。如有必要，请使用搜索栏。

展开“媒体播放”部分。如果视频仍在上传，则会显示该视频。如果上传已完成，请选择左双箭头。

### 使用 HLS 使用媒体数据

您可以使用 HLS 创建使用来自 Kinesis 视频流的数据的客户端应用程序。有关创建使用 HLS 消耗媒体数据的应用程序的信息，请参阅 [Kinesis Video Streams 播放](#)。

### 使用示例媒体查看器应用程序查看媒体

亚马逊 Kinesis Video Streams 创建了一个能够在 HLS 或 DASH 中播放 Kinesis Video Streams 的媒体查看器的示例网页实现。您可以查看源代码[GitHub](#)并尝试使用[托管网页](#)。

1. 打开 [Amazon Kinesis Video Streams 媒体查看器](#)。
2. 填写以下字段：
  - 区域-选择 us-west-2。
  - AWS 访问密钥

- AWS 密钥
- 流名称
- 回放模式-选择“直播”。

3. 选择开始播放。

## Connect 到存储会话

按照以下步骤创建存储会话并启动 WebRTC 连接过程。主参与者应致电[JoinStorageSession](#)。观众参与者应致电[JoinStorageSessionAsViewer](#)。

这将使存储会话通过信号向通过连接的主参与者或通过连接的指定查看者参与者发送信号[the section called “ConnectAsMaster”](#)，发送SDP报价和ICE候选人。[the section called “ConnectAsViewer”](#)

1. 获取信令信道的 ARN，因为它是下一步的必填输入。如果您已经知道 ARN，请继续下一步。

### AWS Management Console

1. 打开 [Kinesis Video Streams 信令频道控制台](#)。
2. 选择您的信令频道的名称。
3. 在信令频道信息选项卡下，找到您的信令频道的 ARN。

### AWS CLI

确认您已 AWS CLI 安装并配置了。有关更多信息，请参阅 [AWS Command Line Interface 《用户指南》](#)。

有关安装说明，请参阅 [《AWS Command Line Interface 用户指南》](#)。安装完成后，AWS CLI 使用凭据和区域进行[配置](#)。

或者，打开已 AWS CLI 安装和配置的 AWS CloudShell 终端。有关更多信息，请参阅 [AWS CloudShell 用户指南](#)。

使用以下命令运行以下 [Describe-Signaling-Channel](#) 命令：AWS CLI

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

响应看起来与以下内容类似：

```
{
  "ChannelInfo": {
    "ChannelName": "YourChannelName",
    "ChannelARN": "arn:aws:kinesisvideo:us-  
west-2:123456789012:channel/YourChannelName/123456789012",
    "ChannelType": "SINGLE_MASTER",
    "ChannelStatus": "ACTIVE",
    "CreationTime": "2024-07-07T23:28:24.941000-07:00",
    "SingleMasterConfiguration": {
      "MessageTtlSeconds": 60
    }
  },
  "Version": "Ws0fZvFGXzEpuZ2CE1s9"
}
```

您将在对象中找到信令通道 ARN。ChannelInfo

## AWS SDK

此代码片段向您展示了如何使用适用于 v2 的 SDK 创建带有 WebRTC 信令通道的 Kinesis Video Streams。AWS JavaScript 语法将与其他语法不同 AWS SDKs，但一般流程将相同。

您可以查看[JoinStorageSession](#)或的完整代码示例[JoinStorageSessionAsViewer](#)。

创建 Kinesis Video Streams 客户端。这用于调用 [DescribeSignalingChannel API](#)。

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

使用客户端调用 DescribeSignalingChannel API。

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

保存响应。

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

2. 获取 WebRTC 终端节点。必须向其指定端点发出[JoinStorageSessionAsViewer](#)对特定信令通道的请求[JoinStorageSession](#)或针对该信令通道的请求。

## AWS Management Console

### Note

Kinesis Vid AWS Management Console 目前不支持此操作。

打开已 AWS CLI 安装和配置的 AWS CloudShell 终端。有关更多信息，请参阅 [AWS CloudShell 用户指南](#)。

按照 AWS CLI 选项卡中的说明进行操作。

## AWS CLI

确认您已 AWS CLI 安装并配置了。有关更多信息，请参阅[AWS Command Line Interface](#) 文档。

有关安装说明，请参阅《[AWS Command Line Interface 用户指南](#)》。安装完成后，AWS CLI 使用凭据和区域进行[配置](#)。

或者，打开已 AWS CLI 安装和配置的 AWS CloudShell 终端。有关更多信息，请参阅 [AWS CloudShell 用户指南](#)。

在Get-Signaling-Channel-Endpoint以下位置运行命令 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"
```

该响应应该类似于下列内容：

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

## AWS SDK

此代码片段向您展示了如何使用适用于 v2 的 SDK 调用带有 WebRTC 信令通道的 Kinesis Video Streams 的 `GetSignalingChannelEndpoint` API。AWS JavaScript 语法将与其他语法不同 AWS SDKs，但一般流程将相同。查看[JoinStorageSession](#)或的完整代码示例[JoinStorageSessionAsViewer](#)。

创建 Kinesis Video Streams 客户端。这是用于调用 [DescribeSignalingChannel API](#) 的客户端。

如果您之前创建了 Kinesis Video Streams 客户端进行 `DescribeSignalingChannel` 调用，则可以重复使用同一个客户端。

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

使用客户端调用 `GetSignalingChannelEndpoint` API。

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

```
    },
  })
  .promise();
```

保存响应：

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

3. 使用频道 ARN 和 WEBRTC 端点进行 API 调用。如果参与者通过 ConnectAsMasterConnectAsViewer WebSocket APIs 或使用 WebRTC 信号正确连接到 Kinesis Video Streams，则存储会话中将向参与者发送 SDP 报价和 ICE 候选消息流。WebSocket

AWS Management Console

#### Note

Kinesis Video Streams 目前不支持此操作。

打开已 AWS CLI 安装和配置的 AWS CloudShell 终端。有关更多信息，请参阅 [AWS CloudShell 用户指南](#)。

按照 AWS CLI 选项卡中的说明进行操作。

AWS CLI

确认您已 AWS CLI 安装并配置了。有关更多信息，请参阅 [AWS Command Line Interface](#) 文档。

有关安装说明，请参阅《[AWS Command Line Interface 用户指南](#)》。安装完成后，AWS CLI 使用凭据和区域进行 [配置](#)。

或者，打开已 AWS CLI 安装和配置的 AWS CloudShell 终端。有关更多信息，请参阅 [AWS CloudShell 用户指南](#)。

AWS CLI 使用前面步骤中的频道 ARN 和 WEBRTC 端点为主参与者运行 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/123456789012 \  
--region "us-west-2"
```

成功运行后，返回空响应且不打印任何内容。

如果是观众参与者，请 AWS CLI 使用之前的频道 ARN 和 WEBRTC 端点在中运行以下 `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/123456789012 \  
--client-id "ExampleViewerClientID"  
--region "us-west-2"
```

成功运行后，返回空响应且不打印任何内容。

## AWS SDK

此代码片段向您展示了如何使用适用于 v2 的软件开发工具包调用 Kinesis Video Streams 和 WebRTC 信令通道的 Kinesis Video Streams 的 `JoinStorageSession` 或 `JoinStorageSessionAsViewer` API。AWS JavaScript 语法将与其他语法不同 AWS SDKs，但一般流程将相同。查看 [JoinStorageSession](#) 或的完整代码示例 [JoinStorageSessionAsViewer](#)。

创建 Kinesis Video Streams WebRTC 存储客户端。这是用于调用 `JoinStorageSession` 或的客户端，`JoinStorageSessionAsViewer` 与之前步骤中创建的 Kinesis Video Streams 客户端不同。

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

使用客户端为主参与者调用 `JoinStorageSession` API。

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

如果是观看者参与者，则使用客户端调用 `JoinStorageSessionAsViewer` API。

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

包含此代码示例的实时网页可在上找到[GitHub](#)。输入您的区域、AWS 凭证和信令频道的名称。展开 WebRTC 摄取和存储节点，取消选中自动确定摄取模式。

将手动覆盖切换为开启并选择显示按钮以手动调用 `JoinStorageSession` API and/or 显示按钮来手动调用 `JoinStorageSessionAsViewer` API。

当您选择“启动主服务器”或“启动查看器”时，在应用程序通过 `ConnectAsMaster` 或连接到信令后 `ConnectAsViewer`，将出现一个按钮，让存储会话启动与对等方的 WebRTC 连接，而不是应用程序在连接到信令后立即调用 API。

## 解决与存储会话连接的问题

本节提供有关设置和配置用于录制视频流的存储空间的疑难解答指南。

主题

- [控制和控制同行](#)
- [查看支持的编解码器](#)

### 控制和控制同行

在 WebRTC 中，控制对等体通过发送 SDP 报价来启动与受控对等体的连接。对于 peer-to-peer 会话，观看者参与者通过 Signaling 向主参与者发送报价来启动连接。连接到存储会话进行 WebRTC 摄取时，存

储会话是控制对等体。对于主参与者，他们仍然是受控参与者。但是，观看者参与者会从控制切换到受控。

致电[JoinStorageSession](#)或时 [JoinStorageSessionAsViewer](#)，所有参与者都必须回复 SDP 答案，并与存储会话交换 ICE 候选人。

有关序列图，请参见[the section called “了解 WebRTC 的摄取和存储”](#)。

## 查看支持的编解码器

在[发送 SDP 答案](#)并与存储会话[交换 ICE 候选](#)内容时，我们建议在消息correlationId中包含 a。在消息correlationId中包含允许存储会话返回statusResponse消息。这些消息将包含输入消息correlationId中的内容，允许您跟踪该消息statusResponse属于哪条消息。这样，您就可以立即收到有关您的 SDP 答案被拒绝的原因的反馈。

有关 correlationId 和 statusResponse 的更多信息，请参阅 [the section called “异步消息接收”](#)。

存储会话可能拒绝 SDP 答案的一个常见原因是存储会话无法接受答案中指定的编解码器。示例statusResponse可能如下所示：

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

在查看 SDP 答案内容时，请查看以开头的行a=rtpmap并验证编解码器是否与存储会话支持的编解码器相匹配。以下是包含opus音频和视频的SDP示例答案的片段。VP8

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

[JoinStorageSession](#)有关支持的编解码器的列表，请参阅。

# 安全性

云安全 AWS 是重中之重。作为 AWS 客户，您将受益于专为满足大多数安全敏感型组织的要求而构建的数据中心和网络架构。

安全是双方共同承担 AWS 的责任。[责任共担模式](#)将其描述为云的 安全性和云中的安全性：

- 云安全 — AWS 负责保护在 AWS 云中运行 AWS 服务的基础架构。AWS 还为您提供可以安全使用的服务。作为 [AWS 合规性计划](#)的一部分，我们的安全措施的有效性定期由第三方审计员进行测试和验证。要了解适用于 Kinesis Video Streams 的合规性计划，请参阅[合规性计划范围内的AWS 服务](#)。
- 云端安全-您的责任由您使用的 AWS 服务决定。您还需要对其他因素负责，包括您的数据的敏感性、您组织的要求以及适用的法律法规。

本文档可帮助您了解在 WebRTC 中使用 Amazon Kinesis Video Streams 时如何应用分担责任模型。以下主题向您展示了如何使用 WebRTC 配置 Amazon Kinesis Video Streams，以实现您的安全和合规目标。您还将学习如何使用其他 AWS 服务，这些服务可以帮助您通过 WebRTC 资源监控和保护您的 Amazon Kinesis Video Streams。

## 主题

- [使用 WebRTC 资源控制对亚马逊 Kinesis Video Streams 的访问权限 AWS Identity and Access Management](#)
- [Amazon Kinesis Video Streams with WebRTC 的合规性验证](#)
- [使用 WebRTC 实现亚马逊 Kinesis Video Streams 的弹性](#)
- [使用 WebRTC 实现 Kinesis Video Streams 中的基础设施安全](#)
- [使用 WebRTC 实现亚马逊 Kinesis Video Streams 的安全最佳实践](#)
- [WebRTC 加密](#)

## 使用 WebRTC 资源控制对亚马逊 Kinesis Video Streams 的访问权限 AWS Identity and Access Management

通过将 AWS Identity and Access Management (IAM) 与 Amazon Kinesis Video Streams 与 WebRTC 配合使用，您可以控制组织中的用户是否可以使用具有 WebRTC API 操作的特定 Kinesis Video Streams 执行任务，以及他们是否可以使用特定资源。AWS

有关 IAM 的更多信息，请参阅以下文档：

- [AWS Identity and Access Management \(IAM\)](#)
- [入门](#)
- [IAM 用户指南](#)

内容

- [策略语法](#)
- [API 操作](#)
- [亚马逊资源名称 \(ARNs\)](#)
- [向其他 IAM 账户授予访问 Kinesis 视频流的权限](#)
- [策略示例](#)

## 策略语法

IAM policy 是包含一个或多个语句的 JSON 文档。每个语句的结构如下：

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

组成语句的各个元素如下：

- **Effect**：此 effect 可以是 Allow 或 Deny。在默认情况下，IAM 用户没有使用资源和 API 操作的许可，因此，所有请求均会被拒绝。显式允许将覆盖默认规则。显式拒绝将覆盖任何允许。
- **Action**：action 是对其授予或拒绝权限的特定 API 操作。
- **Resource**：受操作影响的资源。要在语句中指定资源，您需要使用其 Amazon 资源名称 (ARN)。

- 条件：条件是可选的。它们可以用于控制策略生效的时间。

在创建和管理 IAM policy 时，您可能希望使用 [IAM Policy 生成器](#) 和 [IAM Policy Simulator](#)。

## API 操作

在 IAM policy 语句中，您可以从支持 IAM 的任何服务中指定任何 API 操作。对于 Kinesis Video Streams with WebRTC，使用以下前缀为 API 操作命名：kinesisvideo:。例如：kinesisvideo:CreateSignalingChannel、kinesisvideo:ListSignalingChannels 和 kinesisvideo:DescribeSignalingChannel。

要在单个语句中指定多项操作，请使用逗号将它们隔开，如下所示：

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

您也可以使用通配符指定多项操作。例如，您可以指定名称以单词“Get”开头的所有操作，如下所示：

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

要指定 Kinesis Video Streams 的所有操作，请按如下方式使用星号 (\*) 通配符：

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

有关 Kinesis Video Streams API 操作的完整列表，请参阅 [Kinesis Video Streams API 参考](#)。

## 亚马逊资源名称 (ARNs)

每个 IAM 政策声明都适用于您使用其指定的资源 ARNs。

请对 Kinesis Video Streams 使用以下 ARN 资源格式：

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

例如：

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

您可以使用获取频道的 ARN。 [DescribeSignalingChannel](#)

## 向其他 IAM 账户授予访问 Kinesis 视频流的权限

您可能需要向其他 IAM 账户授予对 Kinesis Video Streams with WebRTC 信令通道执行操作的权限。服务角色是由一项服务担任、代表您执行操作的 [IAM 角色](#)。IAM 管理员可以在 IAM 中创建、修改和删除服务角色。有关更多信息，请参阅《IAM 用户指南》中的 [创建向 AWS 服务委派权限的角色](#)。

### 策略示例

以下策略示例演示如何控制用户对 Kinesis Video Streams with WebRTC 通道的访问。

#### Example 1：允许用户从任何信令通道获取数据

此策略允许用户或组对任何 DescribeSignalingChannel 信令通道执行 GetSignalingChannelEndpoint、ListSignalingChannels 和 ListTagsForResource 操作。

#### JSON

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

#### Example 2：允许用户创建信令通道

此策略允许用户或组执行 CreateSignalingChannel 操作。

```
{
  "Statement": [
```

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

Example 3 : 允许用户访问所有 Kinesis Video Streams 和 Kinesis Video Streams with WebRTC 的资源

此策略允许用户或组对任何资源执行任何 Kinesis Video Streams 操作。此策略适用于管理员。

JSON

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

Example 4 : 允许用户从特定信令通道获取数据

此策略允许用户或组从特定信令通道获取数据。

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 的合规性验证

要了解是否属于特定合规计划的范围，请参阅AWS 服务“[按合规计划划分的范围](#)”，然后选择您感兴趣的合规计划。AWS 服务 有关一般信息，请参阅[AWS 合规计划AWS](#)。

您可以使用下载第三方审计报告 AWS Artifact。有关更多信息，请参阅中的“[下载报告](#)”中的“[AWS Artifact](#)”。

您在使用 AWS 服务 时的合规责任取决于您的数据的敏感性、贵公司的合规目标以及适用的法律和法规。AWS 提供了以下资源来帮助实现合规性：

- [Security Compliance & Governance](#)：这些解决方案实施指南讨论了架构考虑因素，并提供了部署安全性和合规性功能的步骤。
- [符合 HIPAA 要求的服务参考](#)：列出符合 HIPAA 要求的服务。并非所有 AWS 服务 人都符合 HIPAA 资格。
- [AWS 合规资源AWS](#) — 此工作簿和指南集可能适用于您所在的行业和所在地区。
- [AWS 客户合规指南](#) — 从合规角度了解责任共担模式。这些指南总结了保护的最佳实践，AWS 服务 并将指南映射到跨多个框架（包括美国国家标准与技术研究院 (NIST)、支付卡行业安全标准委员会 (PCI) 和国际标准化组织 (ISO) ) 的安全控制。
- [使用AWS Config 开发人员指南中的规则评估资源](#) — 该 AWS Config 服务评估您的资源配置在多大程度上符合内部实践、行业指导方针和法规。
- [AWS Security Hub](#)— 这 AWS 服务 提供了您内部安全状态的全面视图 AWS。Security Hub 通过安全控制措施评估您的 AWS 资源并检查其是否符合安全行业标准和最佳实践。有关受支持服务及控制措施的列表，请参阅 [Security Hub 控制措施参考](#)。
- [Amazon GuardDuty](#) — 它通过监控您的 AWS 账户环境中是否存在可疑和恶意活动，来 AWS 服务 检测您的工作负载、容器和数据面临的潜在威胁。GuardDuty 通过满足某些合规性框架规定的入侵检测要求，可以帮助您满足各种合规性要求，例如 PCI DSS。
- [AWS Audit Manager](#)— 这 AWS 服务 可以帮助您持续审计 AWS 使用情况，从而简化风险管理以及对法规和行业标准的合规性。

## 使用 WebRTC 实现亚马逊 Kinesis Video Streams 的弹性

AWS 全球基础设施是围绕 AWS 区域 可用区构建的。AWS 区域 提供多个物理分隔和隔离的可用区，这些可用区通过低延迟、高吞吐量和高度冗余的网络连接。您可以利用可用区设计和运行在可用区之间无中断地自动实现故障转移的应用程序和数据库。与传统的单个或多个数据中心基础结构相比，可用区具有更高的可用性、容错性和可扩展性。

有关 AWS 区域 和可用区的更多信息，请参阅[AWS 全球基础设施](#)。

## 使用 WebRTC 实现 Kinesis Video Streams 中的基础设施安全

作为一项托管服务，Kinesis Video Streams ( 包括其WebRTC功能 ) 受 AWS 亚马逊网络[服务](#)：安全流程概述白皮书中描述的全局网络安全程序的保护。

您可以使用 AWS 已发布的 API 调用通过网络访问 Kinesis Video Streams。客户端必须支持传输层安全性 ( TLS ) 1.2 或更高版本。我们建议使用 TLS 1.3 或更高版本。客户端还必须支持具有完全向前保密 ( PFS ) 的密码套件，例如 Ephemeral Diffie-Hellman ( DHE ) 或 Elliptic Curve Ephemeral Diffie-Hellman ( ECDHE )。大多数现代系统 ( 如 Java 7 及更高版本 ) 都支持这些模式。

此外，必须使用访问密钥 ID 和与 IAM 主体关联的秘密访问密钥来对请求进行签名。或者，您可以使用[AWS Security Token Service](#) ( AWS STS ) 生成临时安全凭证来对请求进行签名。

## 使用 WebRTC 实现亚马逊 Kinesis Video Streams 的安全最佳实践

Amazon Kinesis Video Streams ( 包含其 WebRTC 功能 ) 提供了您在开发和实施自己的安全策略时需要考虑的许多安全功能。以下最佳实践是一般指导原则，并不代表完整安全解决方案。这些最佳实践可能不适合环境或不满足环境要求，请将其视为有用的考虑因素而不是惯例。

有关远程设备的安全最佳实践，请参阅[设备代理的安全最佳实践](#)。

### 实施最低权限访问

在授予权限时，您可以决定谁获得哪些 Kinesis Video Streams 资源的哪些权限。您可以对这些资源启用希望允许的特定操作。因此，您应仅授予执行任务所需的权限。实施最低权限访问对于减小安全风险以及可能由错误或恶意意图造成的影响至关重要。

例如，向 Kinesis Video Streams 发送数据的创建者仅需要 PutMedia、GetStreamingEndpoint 和 DescribeStream。请勿向创建者应用程序授予所有操作 (\*) 或其他操作 ( 例如 GetMedia ) 的权限。

有关更多信息，请参阅 [Apply least-privilege permissions](#)。

## 使用 IAM 角色

创建者和客户端应用程序必须具有有效的凭证来访问 Kinesis 视频流。您不应将 AWS 证书直接存储在客户端应用程序或 Amazon S3 存储桶中。这些是不会自动轮换的长期凭证，如果它们受到损害，可能会对业务产生重大影响。

相反，您应该使用 IAM 角色来管理创建器和客户端应用程序的临时凭证以访问 Kinesis 视频流。在使用角色时，您不必使用长期凭证来访问其他资源。

有关更多信息，请参阅 IAM 用户指南中的以下主题：

- [IAM 角色](#)
- [角色的常见场景：用户、应用程序和服务](#)

## CloudTrail 用于监控 API 调用

带有 WebRTC 的 Kinesis Video Streams AWS CloudTrail 与一项服务集成，该服务提供用户、角色或服务在使用 WebRTC 的 AWS Kinesis Video Streams 中采取的操作的记录。

使用收集的信息 CloudTrail，您可以确定使用 WebRTC 向 Kinesis Video Streams 发出的请求、发出请求的 IP 地址、谁提出请求、何时发出请求以及其他详细信息。

有关更多信息，请参阅 [the section called “使用记录 API 调用 CloudTrail”](#)。

## WebRTC 加密

端到端加密是带有 WebRTC 的 Amazon Kinesis Video Streams 的必备功能，Kinesis Video Streams 在包括信令和媒体或数据流在内的所有组件上都强制使用端到端加密。无论通信是 peer-to-peer 通过 Kinesis Video Streams Video Streams TURN 端点还是中继，所有 WebRTC 通信都通过标准化加密协议进行安全加密。

使用安全 Websockets (WSS) 交换信令消息，使用数据报传输层安全 (DTLS) 对数据流进行加密，媒体流使用安全实时传输协议 (SRTP) 进行加密。

## 监控指标和 API 调用

监控是使用 WebRTC 和您的解决方案维护 Amazon Kinesis Video Streams 的可靠性、可用性和性能的重要组成部分。AWS 您应该从 AWS 解决方案的所有部分收集监控数据，以便在出现多点故障时可以更轻松地对其进行调试。不过，在开始监控 Kinesis Video Streams with WebRTC 之前，您应制定监控计划并在计划中回答下列问题：

- 监控目的是什么？
- 您将监控哪些资源？
- 监控这些资源的频率如何？
- 您将使用哪些监控工具？
- 谁负责执行监控任务？
- 出现错误时应通知谁？

在定义监控目标并创建监控计划后，下一步是在您的环境中建立 Kinesis Video Streams with WebRTC 正常性能的基准。您应该在不同时间和不同负载条件下测量 Kinesis Video Streams with WebRTC 性能。监控 Kinesis Video Streams with WebRTC 时，您应存储所收集的监控数据的历史记录。您可将当前 Kinesis Video Streams with WebRTC 性能与这些历史数据进行比较，这样有助于您确定性能的正常模式和异常模式，以及找出方法来应对可能出现的问题。

### 主题

- [使用 WebRTC 监控 Kinesis Video Streams](#)
- [使用记录 API 调用 AWS CloudTrail](#)

## 使用 WebRTC 监控 Kinesis Video Streams

您可以使用亚马逊通过 WebRTC 监控亚马逊 Kinesis Video Streams，亚马逊通过 WebRTC 收集来自 CloudWatch 亚马逊 Kinesis Video Streams 的原始数据，并将其处理为可读的、近乎实时的指标。这些统计数据会保存 15 个月，从而使您能够访问历史信息，并能够更好地了解您的 Web 应用程序或服务的执行情况。

亚马逊 Kinesis Video Streams 提供以下指标：

### 主题

- [信令指标](#)
- [TURN 指标](#)
- [WebRTC 摄取指标](#)

## 信令指标

本节提供有关如何使用日志监控和解决与信号相关的问题的信息。 CloudWatch

指标名称	描述	单位	Dimension s
Failure	如果维度中提到的操作返回 200 状态代码响应，则会发出“0”。否则为“1”。	计数	操作， Signaling ChannelName
延迟	衡量服务接收请求和返回响应所花费的时间。	毫秒	操作， Signaling ChannelName
MessagesTransferred.Count	一个频道发送和接收的消息总数。	计数	Signaling ChannelName

该Operation维度适用于以下内容 APIs：

- ConnectAsMaster
- ConnectAsViewer
- SendSdpOffer
- SendSdpAnswer
- SendCandidate
- SendAlexaOfferToMaster
- GetIceServerConfig
- Disconnect

## TURN 指标

本节提供有关如何使用 CloudWatch 日志监控和解决回合相关问题的信息。

指标名称	描述	单位	Dimensions
TURNConnected分钟	对于一分钟内用于流式传输数据的每个 TURN 分配都发出“1”。	计数	Signaling ChannelName

## WebRTC 摄取指标

本节提供有关如何使用日志监控和解决WebRTC摄取相关问题的信息。 CloudWatch

指标名称	描述	单位	Dimensions
Failure	如果维度中引用的操作返回 200 状态码，则会发出值 '0'；否则会发出值 '1'。	计数	操作， Signaling ChannelName
TotalBitrate	主参与者发送的总比特率	bps	操作， ChannelName
TotalPacketCount	在此期间，MASTER 参与者发送的数据包总数。	计数	操作， ChannelName

对于 WebRTC 摄取指标，Operation该维度适用于以下 API：。JoinStorageSession

## 使用记录 API 调用 AWS CloudTrail

带有 WebRTC 的 Amazon Kinesis Video Streams AWS CloudTrail与 WebRTC 集成，该服务提供用户、角色或 AWS 服务在 WebRTC 的亚马逊 Kinesis Video Streams 中采取的操作记录。CloudTrail 以 WebRTC 作为事件捕获亚马逊 Kinesis Video Streams 的所有 API 调用。捕获的调用包括来自

Amazon Kinesis Video Streams 控制台的调用以及对 Amazon Kinesis Video Streams with WebRTC API 操作的代码调用。如果您创建跟踪，则可以允许将 CloudTrail 事件持续传输到 Amazon S3 存储桶，包括通过 WebRTC 为亚马逊 Kinesis Video Streams 提供的事件。如果您不配置跟踪记录，则仍可在 CloudTrail 控制台的 Event history (事件历史记录) 中查看最新事件。使用收集的信息 CloudTrail，您可以确定通过 WebRTC 向 Amazon Kinesis Video Streams 发出的请求、发出请求的 IP 地址、谁提出请求、何时发出请求以及其他详细信息。

要了解更多信息 CloudTrail，包括如何配置和启用它，请参阅[AWS CloudTrail 用户指南](#)。

## 亚马逊 Kinesis Video 通过 WebRTC 直播 Kinesis Video Streams 和 CloudTrail

CloudTrail 在您创建 AWS 账户时已在您的账户上启用。当支持的事件活动出现在带有 WebRTC 的 Amazon Kinesis Video Streams 中时，该活动将 AWS 与其他服务事件一起记录在 CloudTrail 事件历史记录中。您可以在自己的 AWS 账户中查看、搜索和下载最近发生的事件。有关更多信息，请参阅[使用事件历史记录查看 CloudTrail 事件](#)。

要持续记录您的 AWS 账户中的事件，包括使用 WebRTC 的 Amazon Kinesis Video Streams 的事件，请创建跟踪。跟踪允许 CloudTrail 将日志文件传输到 Amazon S3 存储桶。预设情况下，在控制台中创建跟踪记录时，此跟踪记录应用于所有 AWS 区域。跟踪记录 AWS 分区中所有区域的事件，并将日志文件传送到您指定的 Amazon S3 存储桶。此外，您可以配置其他 AWS 服务，以进一步分析和处理 CloudTrail 日志中收集的事件数据。有关更多信息，请参阅下列内容：

- [创建跟踪概述](#)
- [CloudTrail 支持的服务和集成](#)
- [配置 Amazon SNS 通知 CloudTrail](#)
- [接收来自多个区域的 CloudTrail 日志文件和接收来自多个账户的 CloudTrail 日志文件](#)

Amazon Kinesis Video Streams with WebRTC 支持在 CloudTrail 日志文件中将以下操作记录为事件：

- [CreateSignalingChannel](#)
- [DeleteSignalingChannel](#)
- [DescribeSignalingChannel](#)
- [GetSignalingChannelEndpoint](#)
- [ListSignalingChannels](#)
- [ListTagsForResource](#)

- [TagResource](#)
- [UntagResource](#)
- [UpdateSignalingChannel](#)

每个事件或日志条目都包含有关生成请求的人员信息。身份信息有助于您确定以下内容：

- 请求是使用根证书还是 AWS Identity and Access Management (IAM) 用户凭证发出。
- 请求是使用角色还是联合用户的临时安全凭证发出的。
- 请求是否由其他 AWS 服务发出。

有关更多信息，请参阅 [CloudTrail userIdentity 元素](#)。

## 示例：日志文件条目

跟踪是一种配置，允许将事件作为日志文件传输到您指定的 Amazon S3 存储桶。CloudTrail 日志文件包含一个或多个日志条目。事件代表来自任何来源的单个请求，包括有关请求的操作、操作的日期和时间、请求参数等的信息。CloudTrail 日志文件不是公共 API 调用的有序堆栈跟踪，因此它们不会按任何特定顺序出现。

以下示例显示了演示该 [CreateSignalingChannel](#) 操作的 CloudTrail 日志条目。

```
{
  "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 参考

## 主题

- [WebSocket 终端节点 APIs](#)
- [异步消息接收](#)

## WebSocket 终端节点 APIs

以下是 Amazon Kinesis Video Streams with WebR WebSocket TC 终端节点：APIs

## 主题

- [ConnectAsMaster](#)
- [ConnectAsViewer](#)

## ConnectAsMaster

以主设备的身份连接到由终端节点指定的信令通道。任何 WebSocket 投诉库都可用于连接到从 GetSignalingChannelEndpoint API 调用中获得的安全 websocket (WSS) 端点。信令通道的 Amazon 资源名称 (ARN) 必须作为查询字符串参数提供。有单独的终端节点可用于作为主设备和查看器进行连接。如果多个客户端作为主设备连接到特定通道，则最近的请求优先。新的连接元数据将覆盖现有连接元数据。

## 请求

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

- X-Amz-ChannelARN - 信令通道的 ARN。
  - 类型：字符串
  - 长度限制：最小长度为 1。长度上限为 1024。
  - 模式：arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9\_.-]+/[0-9]+
  - 必需：是

## 响应

200 OK HTTP 状态代码以及一个空正文。

## 错误

- `InvalidArgumentException`

指定的参数超出其限制、不受支持或无法使用。有关更多信息，请参阅返回的消息。

HTTP 状态代码：400

- `AccessDeniedException`

未授权调用方访问给定的通道或令牌已过期。

HTTP 状态代码：403

- `ResourceNotFoundException`

通道不存在。

HTTP 状态代码：404

- `ClientLimitExceededException`

当以过高的速率调用 API 时。有关更多信息，请参阅[带有 WebRTC 服务配额的亚马逊 Kinesis Video Streams](#)和中的[错误重试和指数回退](#)。AWS

HTTP 状态代码：400

## 局限/限制

如果以过高的速率调用 API，此 API 将在账户级别受到限制。受到限制时返回错误以及 `ClientLimitExceededException`。

## 幂等

如果指定的 `clientId` 和通道已存在连接，则使用新信息更新连接元数据。

## 重试行为

这被视为新的 API 调用。

## 并发调用

允许并发调用，对于每个调用都会更新连接元数据。

## ConnectAsViewer

以查看器身份连接到由终端节点指定的信令通道。任何 WebSocket 兼容的库都可用于连接到从 `GetSignalingChannelEndpoint` API 调用中获得的安全 websocket (WSS) 端点。信令通道的 Amazon 资源名称 (ARN) 和客户端 ID 必须作为查询字符串参数提供。有单独的终端节点可用于作为主设备和查看器进行连接。如果存在与请求指定的 `ClientId` 相同的现有连接，则新连接优先。新信息将覆盖连接元数据。

## 请求

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

- X-Amz-ChannelARN - 信令通道的 ARN。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 1024
  - 模式：`arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`
  - 必需：是
- X-Amz-ClientId-客户端的唯一标识符。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：`^(?!(?i)AWS_.*)[a-zA-Z0-9_.-]`

### Note

X-Amz-ClientId 一开始不行 AWS\_。

- 必需：是

## 响应

200 OK HTTP 状态代码以及一个空正文。

## 错误

- `InvalidArgumentException`

指定的参数超出其限制、不受支持或无法使用。有关更多信息，请参阅返回的消息。

HTTP 状态代码：400

- `AccessDeniedException`

未授权调用方访问给定的通道或令牌已过期。

HTTP 状态代码：403

- `ResourceNotFoundException`

通道不存在。

HTTP 状态代码：404

- `ClientLimitExceededException`

当以过高的速率调用 API 时，或者连接到通道的查看器数量超过支持的最大数量时。有关更多信息，请参阅[带有 WebRTC 服务配额的亚马逊 Kinesis Video Streams](#)和中的[错误重试和指数回退](#)。

AWS

HTTP 状态代码：400

## 局限/限制

如果以过高的速率调用该 API，或当连接到该通道的查看器数量超过支持的最大数量时，会在账户级别限制该 API。受到限制时返回错误以及 `ClientLimitExceededException`。

## 幂等

如果指定的 `ClientId` 和通道已存在连接，则使用新信息更新连接元数据。

## 重试行为

这被视为新的 API 调用。

## 并发调用

允许并发调用，对于每个调用都会更新连接元数据。

## 异步消息接收

所有响应消息都作为事件异步传递给接收方（例如，SDP 提议或 SDP 应答传递）。以下是事件消息结构。

### 事件

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

- senderClientId-发件人客户端的唯一标识符。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：[a-zA-Z0-9\_.-]+
  - 必需：否
- messageType - 事件的类型。
  - 类型：ENUM
  - 有效类  
型：SDP\_OFFER、SDP\_ANSWER、ICE\_CANDIDATE、GO\_AWAY、RECONNECT\_ICE\_SERVER、STATUS
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：[a-zA-Z0-9\_.-]+
  - 必需：是
- messagePayload - 以 base-64 编码的消息内容。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 10K。
  - 必需：否

- `correlationId` - 状态所指的消息的唯一标识符。这是客户端消息（例如，SDP 提议、SDP 应答或 ICE 候选项）中提供的相同 `correlationId`。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必需：是
- `errorType` - 用于唯一标识错误的名称。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必需：否
- `statusCode` - 与响应的性质相对应的 HTTP 状态代码。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必需：否
- `description` - 解释状态的字符串描述。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 1K。
  - 必需：否

## SendSdpOffer

将提议发送给目标接收方。前提条件是客户端必须已经连接到从 `GetSignalingChannelEndpoint` API 获取的 `WebSocket` 端点。

如果发送方类型是查看器，则它会将提议发送给主设备。此外，没有必要指定 `RecipientClientId`，并将忽略为 `RecipientClientId` 指定的任何值。如果发送方类型为主设备，提议将发送到由 `RecipientClientId` 指定的目标查看器。在这种情况下，`RecipientClientId` 是必需的输入。

允许主设备客户端应用程序向任何查看器发送提议，而只允许查看器客户端应用程序将提议发送到主设备客户端应用程序。如果查看器客户端应用程序尝试向其他查看器客户端应用程序发送提议，则不会接受该请求。如果同一客户端的未完成提议尚未交付，则该提议将被新提议覆盖。

## 请求

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

- **action** - 正在发送的消息的类型。
  - 类型：ENUM
  - 有效值：SDP\_OFFER、SDP\_ANSWER、ICE\_CANDIDATE
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：[a-zA-Z0-9\_.-]+
  - 必需：是
- **recipientClientId**-收件人的唯一标识符。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：[a-zA-Z0-9\_.-]+
  - 必需：是
- **messagePayload** - 以 base-64 编码的消息内容。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 10K。
  - 必需：是
- **correlationId** - 消息的唯一标识符。此参数为可选参数。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：[a-zA-Z0-9\_.-]+
  - 必需：否

## 响应

如果信令后端成功接收消息，则不会返回任何响应。如果服务遇到错误，并且在请求中指定了 `correlationId`，则错误详细信息将作为 `STATUS_RESPONSE` 消息返回。有关更多信息，请参阅 [the section called “异步消息接收”](#)。

## 错误

- `InvalidArgumentException`

指定的参数超出其限制、不受支持或无法使用。有关更多信息，请参阅返回的消息。

HTTP 状态代码：400

- `ClientLimitExceededException`

当以过高的速率调用 API 时。有关更多信息，请参阅 [带有 WebRTC 服务配额的亚马逊 Kinesis Video Streams](#) 中的 [错误重试和指数退缩](#)。AWS

HTTP 状态代码：400

## 局限/限制

如果以过高的速率调用 API，此 API 将在账户级别受到限制。受到限制时返回错误以及 `ClientLimitExceededException`。

## 幂等

此 API 不是幂等的。

## 重试行为

这被视为新的 API 调用。

## 并发调用

允许并发调用。每次调用发送一次提议。

## SendSdpAnswer

将应答发送给目标接收方。前提条件是客户端必须已经连接到从 `GetSignalingChannelEndpoint` API 获取的 `WebSocket` 端点。

如果发送方类型是查看器，则它会将应答发送给主设备。此外，没有必要指定 `RecipientClientId`，并将忽略为 `RecipientClientId` 指定的任何值。如果发送方类型为主设备，应答将发送到由 `RecipientClientId` 指定的目标查看器。在这种情况下，`RecipientClientId` 是必需的输入。

允许主设备客户端应用程序向任何查看器发送应答，而只允许查看器客户端应用程序将应答发送到主设备客户端应用程序。如果查看器客户端应用程序尝试向其他查看器客户端应用程序发送应答，则不会接受该请求。如果同一客户端的未完成应答尚未交付，则该应答将被新应答覆盖。

## 请求

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

- `action` - 正在发送的消息的类型。
  - 类型：ENUM
  - 有效值：SDP\_OFFER、SDP\_ANSWER、ICE\_CANDIDATE
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必需：是
- `recipientClientId`-收件人的唯一标识符。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必需：是
- `messagePayload` - 以 base-64 编码的消息内容。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 10K。
  - 必需：是
- `correlationId` - 消息的唯一标识符。
  - 类型：字符串

- 长度限制：最小长度为 1。最大长度为 256。
- 模式：`[a-zA-Z0-9_.-]+`
- 必需：否

## 响应

如果信令后端成功接收消息，则不会返回任何响应。如果服务遇到错误，并且在请求中指定了 `correlationId`，则错误详细信息将作为 `STATUS_RESPONSE` 消息返回。有关更多信息，请参阅 [异步消息接收](#)。

## 错误

- `InvalidArgumentException`

指定的参数超出其限制、不受支持或无法使用。有关更多信息，请参阅返回的消息。

HTTP 状态代码：400

- `ClientLimitExceededException`

当以过高的速率调用 API 时返回。有关更多信息，请参阅 [带有 WebRTC 服务配额的亚马逊 Kinesis Video Streams](#) 中的 [错误重试和指数退缩](#)。AWS

HTTP 状态代码：400

## 局限/限制

如果以过高的速率调用 API，此 API 将在账户级别受到限制。使用 `ClientLimitExceededException` 进行限制时返回错误。

## 幂等

此 API 不是幂等的。

## 重试行为

这被视为新的 API 调用。

## 并发调用

允许并发调用。每次调用发送一次提议。

## SendIceCandidate

将 ICE 候选项发送给目标接收方。前提条件是客户端必须已经连接到从 `GetSignalingChannelEndpoint` API 获取的 WebSocket 端点。

如果发送者类型是查看器，则它会将 ICE 候选项发送给主设备。此外，没有必要指定 `RecipientClientId`，并将忽略为 `RecipientClientId` 指定的任何值。如果发送方类型为主设备，ICE 候选项将发送到由 `RecipientClientId` 指定的目标。在这种情况下，`RecipientClientId` 是必需的输入。

允许主设备客户端应用程序向任何查看器发送 ICE 候选项，而只允许查看器客户端应用程序将 ICE 候选项发送到主设备客户端应用程序。如果查看器客户端应用程序尝试向其他查看器客户端应用程序发送 ICE 候选项，则不会接受该请求。

### 请求

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

- `action` - 正在发送的消息的类型。
  - 类型：ENUM
  - 有效值：SDP\_OFFER、SDP\_ANSWER、ICE\_CANDIDATE
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必需：是
- `recipientClientId`-收件人的唯一标识符。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：`[a-zA-Z0-9_.-]+`
  - 必需：否
- `messagePayload` - 以 base-64 编码的消息内容。
  - 类型：字符串

- 长度限制：最小长度为 1。最大长度为 10K。
- 必需：是
- correlationId - 消息的唯一标识符。
  - 类型：字符串
  - 长度限制：最小长度为 1。最大长度为 256。
  - 模式：[a-zA-Z0-9\_.-]+
  - 必需：否

## 响应

如果信令后端成功接收消息，则不会返回任何响应。如果服务遇到错误，并且在请求中指定了 correlationId，则错误详细信息将作为 STATUS\_RESPONSE 消息返回。有关更多信息，请参阅 [异步消息接收](#)。

## 错误

- InvalidArgumentException

指定的参数超出其限制、不受支持或无法使用。有关更多信息，请参阅返回的消息。

HTTP 状态代码：400

- ClientLimitExceededException

当以过高的速率调用 API 时。有关更多信息，请参阅 [带有 WebRTC 服务配额的亚马逊 Kinesis Video Streams](#) 中的 [错误重试和指数退缩](#)。AWS

HTTP 状态代码：400

## 局限/限制

如果以过高的速率调用 API，此 API 将在账户级别受到限制。受到限制时返回错误以及 ClientLimitExceededException。

## 幂等

此 API 不是幂等的。

## 重试行为

这被视为新的 API 调用。

## 并发调用

允许并发调用。每次调用发送一次提议。

## Disconnect

客户端可以随时关闭连接。WebSocket兼容的库支持关闭功能。关闭连接后，服务会将客户端标记为针对特定信令通道处于脱机状态，并且不会尝试传递任何消息。同样的行为也适用于空闲连接超时的情况。

该服务还会向客户端发送断开连接指示，例如，在部署或服务器维护期间。以下是定义的指示消息：

- **GO\_AWAY**：此消息用于发起连接关闭。它使客户端能够正常处理之前的消息、断开连接，并根据需要重新连接到信令通道。
- **RECONNECT\_ICE\_SERVER**：此消息用于发起中继连接关闭，并使客户端能够正常断开连接，获取新的 ICE 服务器配置，并在需要时重新连接到中继服务器。

# Amazon Kinesis Video Streams with WebRTC 故障排除

使用以下信息来排查 Amazon Kinesis Video Streams with WebRTC 可能遇到的常见问题。

## 建立 peer-to-peer会话的问题

WebRTC 可以帮助缓解由于以下原因而出现的问题：

- 网络地址转换 ( NAT )
- 防火墙
- 对等设备之间的代理

WebRTC 提供了一个框架，可在对等设备连接期间帮助协商和维护连接。它还提供了一种在无法协商 peer-to-peer连接的情况下通过TURN服务器中继媒体的机制。

考虑到建立连接所需的所有组件，有必要了解一些可用来帮助解决会话建立相关问题的工具。

主题

- [会话描述协议 \(SDP\) 提议和应答](#)
- [评估 ICE 候选项的生成](#)
- [确定使用了哪些候选项来建立连接](#)
- [与冰有关的超时](#)

## 会话描述协议 (SDP) 提议和应答

会话描述协议 (SDP) 提议和应答将对等设备之间的 RTC 会话进行初始化。

要了解有关 SDP 协议的更多信息，请参阅[规范](#)。

- 选件由“观众”生成，他们希望通过WebRTC在Kinesis Video Streams中以“主人”的身份与连接到信令频道的对等方建立联系。
- 应答由提议的接收者生成。

提议和应答都是在客户端生成的，尽管它们可能包含迄今为止已经收集到的 ICE 候选项。

适用于 [C](#) 的 [Kinesis Video Streams WebRTC SDK](#) 包含一个简单的环境变量，您可以将其设置以记录 SDP。这有助于了解收到的提议和生成的应答。

SDPs 要 stdout 从 SDK 登录，请设置以下环境变量：`export DEBUG_LOG_SDP=TRUE`。您还可以使用该 `sdpOffer` 事件在 JavaScript 基于客户端的客户端中记录 SDP 的报价和答案。要查看此演示，请参阅 [GitHub](#)。

有关更多信息，请参阅 [the section called “使用 WebRTC 监控 Kinesis Video Streams”](#)。

如果未返回 SDP 答案，则可能是对等设备无法接受 SDP 提议，因为该提议不包含任何兼容的媒体编解码器。您将看到类似以下内容的日志：

```
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'..
```

在查看 SDP 提议的内容时，请查找以 `a=rtpmap` 开头的行，以查看请求的是哪些媒体编解码器。

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

如果您使用 Safari 作为查看器连接到发送 H.265 媒体的主机，则会遇到以下情况：

- `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.`

确认问题出在浏览器生成的 SDP 报价上。在 SDP 报价中，搜索以 H.265 开头的线路 `a=rtpmap`，然后检查是否有 H.265 的线路。它应如下所示：

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

如果该编解码器不存在，请在 Safari 设置中启用 WebRTC 的 H.265 编解码器。

在 Safari 浏览器顶部导航栏中，执行以下操作：

- 选择 Safari 浏览器 > 设置... > 高级。选中“显示 Web 开发者的功能”复选框。
- 选择“功能标志”。选中 WebRTC H265 编解码器复选框。

重新启动浏览器以使更改生效。

## 评估 ICE 候选项的生成

ICE 候选项由每个向 STUN 服务器发出调用的客户端生成。对于 Kinesis Video Streams with WebRTC 来说，STUN 服务器是 `stun:stun.kinesisvideo.{aws-region}.amazonaws.com:443`。

除了调用 STUN 服务器获取候选项外，客户端通常还会调用 TURN 服务器。他们进行此调用是为了在无法建立直接 peer-to-peer 连接时将中继服务器用作后备服务器。

您可以使用以下工具来生成 ICE 候选对象：

- [Trickle ICE WebRTC](#) 样本，其使用 Trickle ICE 来收集候选项
- [IceTest.Info](#)

使用这两个工具。您可以输入 STUN 和 TURN 服务器信息来收集候选项。

[要通过 WebRTC 获取 Kinesis Video Streams 的 TURN 服务器信息和必要凭证，您可以调用 API 操作。GetIceServerConfig](#)

以下 AWS CLI 调用演示了如何获取此信息以用于这两个工具。

```
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](#) 命令的输出如下所示：

```
{  
  "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 值获取 TURN 服务器列表，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
```

响应包含 TURN 服务器的详细信息，包括 TCP 和 UDP 的端点以及访问它们所需的凭证。

#### Note

响应中的 TTL 值决定了这些凭证的有效期限（以秒为单位）。在 Trickle [ICE WebRTC](#) 示例或 [Info IceTest](#) 中使用这些值，使用 Kinesis Video Streams 托管服务端点生成 ICE 候选值。

## 确定使用了哪些候选项来建立连接

了解成功建立会话时使用了哪些候选项可能会有所帮助。如果您的浏览器客户端正在运行已建立的会话，则可以使用内置的 `webrtc-internals` 实用程序在 Google Chrome 中确定这些信息。

在一个浏览器选项卡中打开 WebRTC 会话。

在另一个选项卡中，打开 chrome://webrtc-internals/。您可以在此选项卡中查看所进行会话的所有信息。

您将看到有关已建立连接的信息。例如：

► Create Dump  
Read stats From: (Standardized (promise-based) getStats() API)

Note: computed stats are in []. Experimental stats are marked with an \* at the end and do not show up in the getStats result.

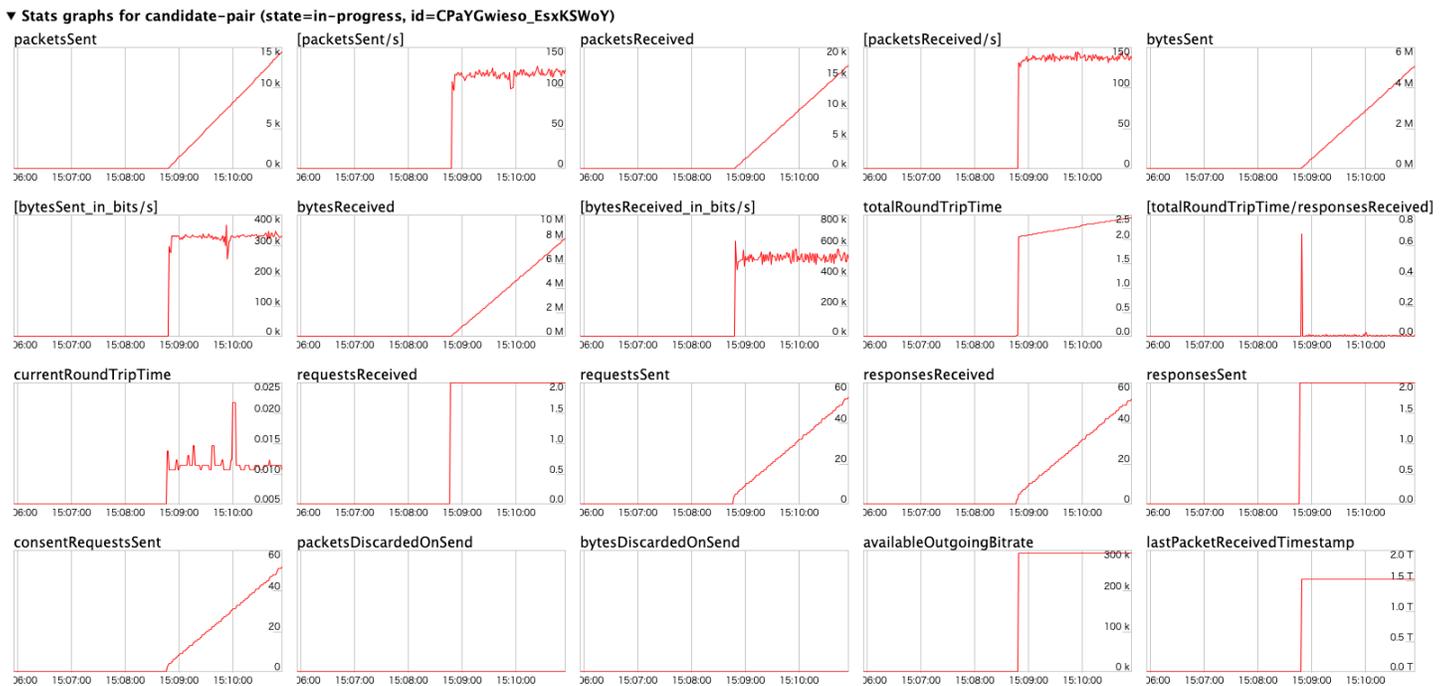
GetUserMedia Requests <https://awslabs.github.io/amazon-kinesis-video-streams-webrtc-sdk-js/examples/index.html#rid-6025>

```
https://awslabs.github.io/amazon-kinesis-video-streams-webrtc-sdk-js/examples/index.html, { iceServers: [stun:kinesisvideo.ap-northeast-1.amazonaws.com:443, turn:3-113-249-125-t-6618366e.kinesisvideo.ap-northeast-1.amazonaws.com:443?transport=udp, turns:3-113-249-125-t-6618366e.kinesisvideo.ap-northeast-1.amazonaws.com:443?transport=tcp, turn:18-183-42-89-t-6618366e.kinesisvideo.ap-northeast-1.amazonaws.com:443?transport=udp, turns:18-183-42-89-t-6618366e.kinesisvideo.ap-northeast-1.amazonaws.com:443?transport=tcp], iceTransportPolicy: all, bundlePolicy: balanced, rtcMuxPolicy: require, iceCandidatePoolSize: 0 }
```

ICE connection state: new => checking => connected  
Connection state: new => connecting => connected  
Signaling state: new => have-local-offer => stable  
ICE Candidate pair: [redacted]:46950 <=> [redacted]:35795  
► ICE candidate grid

State Tables

您还可以确认已建立连接的以下指标。



## 与冰有关的超时

中为 ICE 设置了默认超时值 [KvsRtcConfiguration](#)。对于大多数用户来说，默认值应该足够了，但是您可能需要对其进行调整，以提高通过较差的网络建立连接的机会。您可以在应用程序中配置这些默认值。

查看日志以了解默认设置：

```
2024-01-08 19:43:44.433 INFO iceAgentValidateKvsRtcConfig():
```

```
iceLocalCandidateGatheringTimeout: 10000 ms
iceConnectionCheckTimeout: 12000 ms
iceCandidateNominationTimeout: 12000 ms
iceConnectionCheckPollingInterval: 50 ms
```

如果您的网络质量较差并且想要提高连接几率，请尝试调整以下值：

- `iceLocalCandidateGatheringTimeout`-增加此超时限制，以收集更多潜在候选人尝试连接。目标是尝试所有可能的候选对，因此，如果您的网络状况不佳，请增加此限制以留出更多时间进行收集。

例如，如果主机候选人不起作用，需要尝试服务器反身 (srflx) 或中继候选人，则可能需要延长此超时时间。由于网络不佳，候选人聚集速度很慢，应用程序不想在此步骤上花费超过20秒。增加超时时间可以为收集潜在候选人尝试连接提供更多时间。

#### Note

我们建议该值应小于`iceCandidateNominationTimeout`，因为提名步骤需要有时间与新候选人合作。

- `iceConnectionCheckTimeout`-在不稳定或运行缓慢的网络中，数据包交换和绑定请求/响应需要时间，因此会增加此超时时间。增加此超时允许至少一对候选人尝试由另一对候选人提名。
- `iceCandidateNominationTimeout`-延长此超时时间以确保尝试使用本地中继候选的候选配对。

例如，如果收集第一个本地中继候选需要大约 15 秒，请将超时设置为大于 15 秒的值，以确保成功尝试与本地中继候选的候选配对。如果将该值设置为小于 15 秒，SDK 将在尝试潜在候选对时失败，从而导致连接建立失败。

#### Note

我们建议将此值设置为大于`iceLocalCandidateGatheringTimeout`，以使其生效。

- `iceConnectionCheckPollingInterval`-[每个规格此值默认为 50 毫秒](#)。更改此值会更改连接检查的频率，本质上还会更改 ICE 状态机转换的频率。

在具有良好系统资源的可靠、高性能的网络环境中，您可以降低该值以帮助更快地建立连接。增加该值可以帮助减少网络负载，但建立连接的速度可能会减慢。

**⚠ Important**

我们不建议更改此默认值。

# 使用 WebRTC 开发者指南记录亚马逊 Kinesis Video Streams 的历史记录

变更	说明	日期
<a href="#">修改后的内容组织</a>	使用 WebRTC 开发者指南修改了 Amazon Kinesis Video Streams 中的布局和组织。	2024 年 9 月 18 日
<a href="#">初次发布</a>	初次发布 Amazon Kinesis Video Streams with WebRTC 开发人员指南。 <a href="#">了解更多</a>	2019 年 11 月 4 日

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。