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

Table of Contents

什么是 Amazon Kinesis Video Streams with WebRTC 1
区域可用性 1
Kinesis Video Streams with WebRTC 定价 2
Accessing Kinesis Video Streams with WebRTC 3
Kinesis Video Streams with WebRTC：工作方式 4
Amazon Kinesis Video Streams with WebRTC 概念 4
WebRTC 技术概念 5
STUN、TURN 和 ICE 如何协同工作 5
Kinesis Video Streams with WebRTC 组件 6
WebRTC Websocket API 7
ConnectAsViewer 7
ConnectAsMaster 9
SendSdpOffer 11
SendSdpAnswer 13
SendIceCandidate 15
Disconnect 18
异步消息接收 18
配额 20
控制层面 API 服务配额 20
信令 API 服务配额 22
TURN 服务配额 23
WebRTC 提取服务限额 23
入门 24
设置一个 AWS 账户 24
注册获取 AWS 账户 24
创建具有管理权限的用户 25
创建 AWS账户密钥 26
创建信令通道 26
使用控制台创建信令通道 26
实时媒体流式传输 27
C 中适用于嵌入式设备的 WebRTC 开发工具包 27
WebRTC 软件开发工具包已推出 JavaScript 30
适用于 Android 的 WebRTC 开发工具包 34
适用于 iOS 的 WebRTC 开发工具包 41

WebRTC C SDK 的客户端指标 48
WebRTC 提取 66
API 操作 66
串流提取入门 66
创建信令通道 67
使用创建信令信道 AWS Management Console 67
使用创建信令信道 AWS CLI 67
创建流 67
使用创建直播 AWS Management Console 68
使用创建直播 AWS CLI 68
配置媒体提取和存储 68
提取媒体 69
从浏览器提取媒体 69
从 WebRTC C SDK 中提取媒体 70
查看媒体 71
安全性 72
使用 AWS Identity and Access Management控制对 Kinesis Video Streams with WebRTC 的访问 72
策略语法 73
适用于 Kinesis Video Streams with WebRTC 的操作 74
Kinesis Video Streams 的 Amazon 资源名称（ARN） 74
授予其他 IAM 账户访问 Kinesis 视频流的权限 75
示例策略 75
合规性验证 76
故障恢复能力 77
Kinesis Video Streams with WebRTC 中的基础设施安全 78
适用于 Kinesis Video Streams with WebRTC 的安全最佳实践 78
实施最低权限访问 78
使用 IAM 角色 79
使用 CloudTrail 监控 API 调用 79
WebRTC 加密 79
监控 80
通过 CloudWatch 监控 Kinesis Video Streams with WebRTC 80
信令指标 81
TURN 指标 82
使用 WebRTC API 调用记录 Kinesis Video Streams AWS CloudTrail 82

亚马逊 Kinesis Video 通过 WebRTC 直播 Kinesis Video Streams 和 CloudTrail 82
示例：Amazon Kinesis Video Streams with WebRTC 日志文件条目 83
故障排除 85
服务限额 85
联网要求 85
网络环境 86
建立会话相关问题 86
会话描述协议 (SDP) 提议和应答 87
评估 ICE 候选项的生成 88
确定使用了哪些候选项来建立连接 89
与冰有关的超时 90
调试正在进行的连接 92
文档历史记录 94
AWS 术语表 95
..... xcvi

什么是 Amazon Kinesis Video Streams with WebRTC

WebRTC 是一个开放的技术规范，它通过简单的 API 支持跨浏览器和移动应用程序的实时通信 (RTC)。它使用对等技术在互联对等设备间实时交换数据，并提供人与人交互所需的低延迟媒体流。WebRTC 规范包括一组 IETF 协议，除了用于可靠和安全的实时媒体和数据流式传输的协议规范之外，还包括用于建立对等连接的交互式连接建立、使用中继绕过 NAT 的遍历 (TURN) 和 NAT 的会话遍历实用工具 (STUN)。

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 Kinesis Video Streams with WebRTC，您可以轻松地构建用于下面这些用途的应用程序：进行实时对等媒体流式传输，或在摄像头 IoT 设备、Web 浏览器和移动设备之间针对各种使用案例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拥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：

- Kinesis Video Streams with WebRTC：工作方式
- C 中适用于嵌入式设备的 WebRTC 开发工具包
- 带有 WebRTC SDK 的 Kinesis Video Streams 适用于网络应用程序 JavaScript
- 适用于 Android 的 WebRTC 开发工具包
- 适用于 iOS 的 WebRTC 开发工具包
- 控制层面 API
- 数据层面 REST API
- 数据层面 Websocket API

区域可用性

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

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

Kinesis Video Streams with WebRTC 定价

有关 Kinesis Video Streams with WebRTC 定价的信息，请参阅 Amazon Kinesis Video Streams 定价。

Accessing Kinesis Video Streams with WebRTC

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

AWS Management Console

AWS Management Console入门

此控制台是可用于访问和使用 AWS服务（包括 Kinesis Video Streams with WebRTC）的基于浏览器的界面。

AWS SDK

AWS 提供了软件开发工具包 (SDK)，其中包含各种编程语言和平台的库和示例代码，例如，Java、Python、Ruby、.NET、iOS 和 Android 等。开发工具包提供便捷的方式来创建对 Kinesis Video Streams with WebRTC 的编程访问。有关 AWS 开发工具包的信息（包括如何下载及安装），请参阅适用于 Amazon Web Services 的工具。

Kinesis Video Streams with WebRTC HTTPS API

您可以使用 Kinesis Video Streams with WebRTC API（可让您直接向服务发布 API 请求）以编程方式访问 Kinesis Video Streams with WebRTC 和 AWS。有关更多信息，请参阅 Amazon Kinesis Video Streams API 参考。

Kinesis Video Streams with WebRTC：工作方式

主题

- Amazon Kinesis Video Streams with WebRTC 概念
- WebRTC 技术概念
- STUN、TURN 和 ICE 如何协同工作
- Kinesis Video Streams with WebRTC 组件
- WebRTC Websocket API

Amazon Kinesis Video Streams with WebRTC 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

使应用程序能够通过交换信令消息来发现、设置、控制和终止对等连接的一种资源。信令消息是指两个应用程序为了建立对等连接而互相交换的元数据。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。信令通道使应用程序能够采用一个主设备连接多个查看器的概念在一对多模型中互相连接。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。连接到此信令通道后，主设备和查看器应用程序可以相互发送信令消息，以建立对等连接来进行实时媒体流式传输。

对等

配置为通过 Kinesis Video Streams with WebRTC 实现实时双向流式传输的任何设备或应用程序（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。

WebRTC 技术概念

当您开始使用 Kinesis Video Streams with WebRTC 时，您还可以从了解 WebRTC 技术所包含的几个相互关联的协议和 API 中受益。

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 如何协同工作

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、安全 WebSocket 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。

A 和 B 交换 SDP 后，它们将执行一系列连接性检查。每个应用程序中的 ICE 算法从它在另一方的 SDP 中收到的列表中获取候选项 IP/端口对，并向其发送 STUN 请求。如果从另一个应用程序返回响应，发起方应用程序会认为检查成功，并将该 IP/端口对标记为有效的 ICE 候选项。

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。

Kinesis Video Streams with WebRTC 组件

Kinesis Video Streams with WebRTC 包括以下组件：

- 控制层面

  控制层面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 信令终端节点，这些终端节点允许应用程序彼此之间安全地连接，以实现对等的实时媒体流式传输。信令组件包括 Amazon Kinesis Video Signaling REST API 和一组 Websocket API。

- 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 终端节点，这些终端节点在应用程序无法通过对等方式流式传输媒体时，可通过云启用媒体中继。

- Kinesis Video Streams WebRTC 开发工具包

  这些是软件库，您可以在设备和应用程序客户端上下载、安装和配置这些库，以使具有 WebRTC 功能的摄像头 IoT 设备能够进行低延迟的对等媒体流式传输。这些开发工具包还使 Android、iOS 和 Web 应用程序客户端能够将 Kinesis Video Streams with WebRTC 信令、TURN 和 STUN 功能与任何符合 WebRTC 的移动设备或 Web 播放器集成。

  - C 中适用于嵌入式设备的 WebRTC 开发工具包
  - 带有 WebRTC SDK 的 Kinesis Video Streams 适用于网络应用程序 JavaScript
  - 适用于 Android 的 WebRTC 开发工具包
  - 适用于 iOS 的 WebRTC 开发工具包

WebRTC Websocket API

主题

- ConnectAsViewer
- ConnectAsMaster
- SendSdpOffer
- SendSdpAnswer
- SendIceCandidate
- Disconnect
- 异步消息接收

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 `GetSignalingEndpoint` API 调用中获得的端点。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Amazon Kinesis Video Streams with WebRTC 服务限额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。

ConnectAsMaster

以主设备的身份连接到由终端节点指定的信令通道。任何符合 WebSocket 的库均可用于连接到从 `GetSignalingChannelEndpoint` API 调用获得的终端节点。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 AWS中的 Amazon Kinesis Video Streams with WebRTC 服务限额和 错误重试和指数回退。

  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。

SendSdpOffer

将提议发送给目标接收方。先决条件是客户端必须已连接到从 `GetSignalingChannelEndpoint` API 获取的 WebSocket 终端节点。

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。长度上限为 256。
  - 模式：[a-zA-Z0-9_.-]+
  - 必需：是
- recipientClientId - 接收方的唯一标识符。
  - 类型：字符串
  - 长度限制：最小长度为 1。长度上限为 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。长度上限为 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 AWS中的 Amazon Kinesis Video Streams with WebRTC 服务限额和 错误重试和指数回退。

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。先决条件是客户端必须已连接到从 `GetSignalingChannelEndpoint` API 获取的 WebSocket 终端节点。

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。长度上限为 256。
  - 模式：[a-zA-Z0-9_.-]+
  - 必需：是
- recipientClientId - 接收方的唯一标识符。
  - 类型：字符串
  - 长度限制：最小长度为 1。长度上限为 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。长度上限为 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 AWS中的 Amazon Kinesis Video Streams with WebRTC 服务限额和 错误重试和指数回退。

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。先决条件是客户端必须已连接到从 `GetSignalingChannelEndpoint` API 获取的 WebSocket 终端节点。

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。长度上限为 256。
  - 模式：[a-zA-Z0-9_.-]+
  - 必需：是
- recipientClientId - 接收方的唯一标识符。
  - 类型：字符串
  - 长度限制：最小长度为 1。长度上限为 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。长度上限为 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 AWS中的 Amazon Kinesis Video Streams with WebRTC 服务限额和 错误重试和指数回退。

  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。符合 WebSocket 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。

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

Amazon Kinesis Video Streams with WebRTC 服务限额

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]（可以通过提交支持票证进行升级），要么是硬的 [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

- 控制层面 API 服务配额
- 信令 API 服务配额
- TURN 服务配额
- WebRTC 提取服务限额

控制层面 API 服务配额

以下部分介绍控制层面 API 的服务配额。

API	账户服务配额：请求	账户服务配额：通道	通道级服务配额	相关异常和说明
CreateSignalingChannel	50 TPS [s]	us-east-1 以及 us-west-2 - 每个区域每个账户 10,000 个通道；所有其他支持的区域		

API	账户服务配额：请求	账户服务配额：通道	通道级服务配额	相关异常和说明
		- 每个区域每个账户 5,000 个通道		
DeleteSignalingChannel	50 TPS [h]	不适用	5 TPS [h]	
DescribeMediaStorageConfiguration	50 TPS [h]		5 TPS [h]	
DescribeSignalingChannel	300 TPS [h]	不适用	5 TPS [h]	
GetSignalingChannelEndpoint	300 TPS [h]	不适用		
ListSignalingChannels	50 TPS [h]	不适用		
ListTagsForResource	50 TPS [h]	不适用	5 TPS [h]	
TagResource	50 TPS [h]	不适用	5 TPS [h]	
UntagResource	50 TPS [h]	不适用	5 TPS [h]	
UpdateMediaStorageConfiguration	10 TPS [h]		5 TPS [h]	

API	账户服务配额：请求	账户服务配额：通道	通道级服务配额	相关异常和说明
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Kinesis Video Streams with WebRTC：工作方式。

- ConnectAsMaster
  - API-每个频道 3 TPS (h)
  - 每个信令通道的最大主连接数-1 (h)
  - 连接时长限制-1 小时 (h)
  - 空闲连接超时-10 分钟 (h)
  - 当客户端收到来自服务器的GO_AWAY消息时，连接将在 1 分钟 (h) 的宽限期后终止
- ConnectAsViewer
  - API-每个频道 3 TPS (h)
  - 每个频道的最大观众连接数-10 (s)
  - 连接时长限制-1 小时 (h)
  - 空闲连接超时-10 分钟 (h)
  - 客户端收到来自服务器的GO_AWAY消息后，连接将在 1 分钟 (h) 的宽限期后终止
- Disconnect
  - 不适用
- GetIceServerConfig
  - API-每个信令通道 5 TPS (h)
- SendAlexaOffertoMaster
  - API-每个信令通道 5 TPS (h)
- SendICECandidate
  - API-每个 WebSocket 连接 20 TPS (h)

  - 消息有效载荷大小限制-10k (h)
- SendSDPAnswer
  - API-每个 WebSocket 连接 5 TPS (h)
  - 消息有效载荷大小限制-10k (h)
- SendSDPOffer
  - API-每个 WebSocket 连接 5 TPS (h)
  - 消息有效载荷大小限制-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Kinesis Video Streams with WebRTC：工作方式。

- 比特率-5Mbps (h)
- 凭证生命周期-5 分钟 (h)
- 分配数-每个信令信道 50 个 (h)

WebRTC 提取服务限额

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Amazon Kinesis Video Streams WebRTC 提取。

JoinStorageSession

- API：
  - 每个账户-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)

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。
- 创建一个 Kinesis Video Streams with WebRTC 信令通道。
- 使用带有 WebRTC SDK 的 Kinesis Video Streams 来配置主站和查看器，使其通过信令通道 peer-to-peer执行视频和音频流。

如果您是首次使用 Kinesis Video Streams with WebRTC，建议先阅读 Kinesis Video Streams with WebRTC：工作方式。

主题

- 设置 AWS 账户并创建管理员
- 创建信令通道
- 实时媒体流式传输

设置 AWS 账户并创建管理员

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/signup。
2. 按照屏幕上的说明进行操作。

在注册时，将接到一通电话，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。作为安全最佳实践，应为用户分配管理访问权限，并仅使用 root 用户来执行需要 root 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。在任何时候，您都可以通过转至 https://aws.amazon.com/ 并选择我的账户来查看当前的账户活动并管理您的账户。

创建具有管理权限的用户

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》中的以根用户身份登录。
2. 为您的根用户启用多重身份验证 (MFA)。

   有关说明，请参阅 I A M 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》中的启用 AWS IAM Identity Center。
2. 在 IAM 身份中心中，向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。

为其他用户分配访问权限

1. 在 IAM Identity Center 中，创建一个遵循应用最低权限权限的最佳实践的权限集。

   有关说明，请参阅《AWS IAM Identity Center 用户指南》中的创建权限集。
2. 将用户分配到群组，然后为该群组分配单点登录访问权限。

   有关说明，请参阅AWS IAM Identity Center 用户指南中的添加群组。

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

你可以使用 Kinesis Video Streams 控制台、 AWS API CreateSignalingChannel() 或 AWS CLI 来创建你的信令通道。

使用控制台创建信令通道

1. 登录 AWS Management Console 并打开 Amazon Kinesis Video Streams 控制台。
2. 在 Signaling channels (信令通道) 页面上，选择 Create signaling channel (创建信令通道)。
3. 在 Create a new signaling channel (创建新的信令通道) 页面上，输入信令通道的名称。保持默认 T ime-to-live (Ttl) 值 60 秒不变。
4. 选择 Create signaling channel (创建信令通道)。

5. 创建信令通道后，在通道的详细信息页面上查看其详细信息。

实时媒体流式传输

Kinesis Video Streams with WebRTC 包括以下开发工具包：

- C 中适用于嵌入式设备的 WebRTC 开发工具包
- 带有 WebRTC SDK 的 Kinesis Video Streams 适用于网络应用程序 JavaScript
- 适用于 Android 的 WebRTC 开发工具包
- 适用于 iOS 的 WebRTC 开发工具包

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。您可以使用这些示例进行低延迟、实时、双向音频和视频流式传输，以及在 Web/Android/iOS 应用程序或嵌入式设备的任何组合之间进行数据交换。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。

C 中适用于嵌入式设备的 WebRTC 开发工具包

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。

下载 C 语言版 Kinesis Video Streams With WebRTC 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/awslabs/amazon-kinesis-video-streams-webrtc-sdk-c.git
```

构建 C 语言版 Kinesis Video Streams with WebRTC 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 srtp`
   - 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 `gstreamer`，则无法构建 `kvsWebrtcClientMasterGstSample`。要确保它是（在 macOS 上）构建的，您必须运行：`brew install gstreamer gst-plugins-base gst-plugins-good`

运行 C 中的 WebRTC 开发工具包的示例

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示例 H264/Opus 帧（路径：/samples/h264 和 /samples/SampleFrames ）。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 GStreamer 管道发送示例 H264/Opus 帧。

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 Kinesis Video Streams with WebRTC 测试页面中的步骤打开 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 SDK 的 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

- 安装带有 WebRTC SDK 的 Kinesis Video Streams JavaScript
- 带有 WebR JavaScript TC 的 Kinesis Video Streams SDK 文档
- 使用 Kinesis Video Streams with WebRTC 测试页面
- 使用 WebRTC 编辑 Kinesis Video Streams 测试页面

安装带有 WebRTC SDK 的 Kinesis Video Streams JavaScript

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-w ebrtc。

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。

带有 WebR JavaScript TC 的 Kinesis Video Streams SDK 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 Kinesis Video Streams with WebRTC 测试页面

带有 WebRTC 的 Kinesis Video Streams 还托管一个测试页面，您可以使用该页面创建新的信令通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 测试页面位于 https://awslabs.github.io/-/examples/index.html。amazon-kinesis-video-streams webrtc-sdk-js

测试页的代码位于examples目录中。

主题

- 从测试页面流式传输到 AWS Management Console

- 从测试页面流式传输到测试页面

从测试页面流式传输到 AWS Management Console

1. 打开 Kinesis Video Streams with WebRTC 测试页面，然后完成以下操作：
   - 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 C reate 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。
   - 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。

   如有必要，允许访问您的摄像头和/或麦克风。
3. 在中打开 Kinesis Video Stream s 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。

从测试页面流式传输到测试页面

1. 打开 Kinesis Video Streams with WebRTC 测试页面并填写以下信息：

- 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 C reate 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。
- 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。

   如有必要，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 测试页面。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和之间正在交换媒体viewer。

使用 WebRTC 编辑 Kinesis Video Streams 测试页面

要出于开发目的编辑 SDK 和测试页面，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。

编辑测试页面

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/awslabs/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. 在你的浏览器中，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 Android 的 WebRTC 开发工具包

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**
>
> Kinesis Video Streams 在安卓系统上不支持 IPv6 地址。有关在安卓设备上禁用 IPv6 的更多信息和步骤，请参阅 https://support.surfshark.com/hc/en-us/articles/360011828279-H ow-to-disable-ipv6-on- Android-。

下载适用于 Android 的 WebRTC 开发工具包

要在 Android 中下载 WebRTC 开发工具包，请运行以下命令：

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

在 Android 中构建 WebRTC 开发工具包

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。当您看到构建错误时，请选择 Install missing SDK package(s) (安装缺少的开发工具包) 来安装所有必需的开发工具包，然后选择 Accept (接受) 并完成安装。
3. 配置 Amazon Cognito（用户池和身份池）设置。有关详细步骤，请参阅为 Android WebRTC 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 Android WebRTC 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、网络和 IoT 设备客户端之间的实时 WebRTC 音频/视频流式传输。

运行 Android 示例应用程序

完成以下步骤：

1. 在您的 Android 设备上，使用新的（先创建）或现有的 Amazon Cognito 账户打开AWSKinesisVideoWebRTCDemoApp并登录。
2. 在中 AWSKinesisVideoWebRTCDemoApp，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，您只能在中AWSKinesisVideoWebRTCDemoApp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 区域，然后选择您是要发送音频数据、视频数据，还是发送两者。
5. 要验证 peer-to-peer 直播，请执行以下任一操作：

> **Note**
>
> 确保您在本演示中使用的所有客户端上指定相同的信令通道名称、AWS 区域、查看器 ID 和 AWS 账户 ID。

- P 在两台 Android 设备之间进行eer-to-peer 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AWSKinesisVideoWebRTCDemoApp上以主模式打开（选择 S TART MASTER），开始新会话（信令信道）。

    > **Note**
    >
    > 目前，任何给定信令通道只能有一个主设备。

  - 在第二台 Android 设备AWSKinesisVideoWebRTCDemoApp上以查看器模式打开，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看到主设备的音频/视频数据。
- P 在嵌入式 SDK 主站和安卓设备查看器之间进行eer-to-peer 流式传输
  - 在摄像头设备上，下载、构建 C 中适用于嵌入式设备的 WebRTC 开发工具包 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。在此 Android 设备AWSKinesisVideoWebRTCDemoApp上以查看器模式打开，确认查看者可以看到嵌入式 SDK 主服务器的音频/视频数据。
- P 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eer-to-peer 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。在这台 Android 设备AWSKinesisVideoWebRTCDemoApp上以主模式打开（选择启动主模式），开始新的会话（信令信道）。
  - 下载、构建 带有 WebRTC SDK 的 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看到 Android 主设备的音频/视频。

为 Android WebRTC 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，然后使用以下网址打开 IAM 控制台：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。

适用于 iOS 的 WebRTC 开发工具包

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。

在 iOS 中下载 WebRTC 开发工具包

要在 iOS 中下载 WebRTC 开发工具包，请运行以下命令：

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

在 iOS 中构建 WebRTC 开发工具包

完成以下步骤：

1. 通过KinesisVideoWebRTCDemoApp.xcworkspace打开（路径 amazon-kinesis-video-streams：-/Swift/ .xcworkspace），将 iOS WebRTC SDK 导入 iOS 电脑上的 xCode 集成开发环境 (IDEwebrtc-sdk-ios)。AWSKinesisVideoWebRTCDemoApp
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 iOS WebRTC 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 Android WebRTC 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 Android WebRTC 开发工具包配置 Amazon Cognito中的步骤生成的值更新 `Constants.swift`。

6. 确保您的 iOS 设备已连接到运行 XCode 的 Mac 计算机。在 XCode 中，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、网络和 IoT 设备客户端之间的实时 WebRTC 音频/视频流式传输。

运行 iOS 示例应用程序

完成以下步骤：

1. 在您的 iOS 设备上，使用新的（先创建）或现有的 Amazon Cognito 账户打开AWSKinesisVideoWebRTCDemoApp并登录。

2. 在中 AWSKinesisVideoWebRTCDemoApp，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，您只能在中AWSKinesisVideoWebRTCDemoApp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 区域，然后选择您是要发送音频数据、视频数据，还是发送两者。
5. 要验证 peer-to-peer 直播，请执行以下任一操作：

> **Note**
>
> 确保您在本演示中使用的所有客户端上指定相同的信令通道名称、AWS 区域、查看器 ID 和 AWS 账户 ID。

- P 在两台 iOS 设备之间进行eer-to-peer 直播：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AWSKinesisVideoWebRTCDemoApp台 iOS 设备上以主模式打开（选择启动主模式），开始新会话（信令信道）。

    > **Note**
    >
    > 目前，任何给定信令通道只能有一个主设备。

  - 在第二台 iOS 设备AWSKinesisVideoWebRTCDemoApp上以查看器模式打开，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到主设备的音频/视频数据。
- P 在嵌入式 SDK 主设备和 iOS 设备查看器之间进行eer-to-peer 流式传输
  - 在摄像头设备上，下载、构建 C 中适用于嵌入式设备的 WebRTC 开发工具包 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。在此 iOS 设备AWSKinesisVideoWebRTCDemoApp上以查看器模式打开，并确认 iOS 查看器可以看到嵌入式 SDK 主服务器的音频/视频数据。
- P 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eer-to-peer 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。在此 iOS 设备AWSKinesisVideoWebRTCDemoApp上以主模式打开（选择启动主模式），开始新会话（信令信道）。
  - 下载、构建并以查看者带有 WebRTC SDK 的 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到 Android 主服务器的音频/视频。JavaScript

为 iOS WebRTC 开发工具包配置 Amazon Cognito

先决条件

- 我们建议使用 XCode 来检查、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，然后使用以下网址打开 IAM 控制台：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。

WebRTC 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
- WebRTC 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
cpApiCallLatency	计算控制面板 API 调用的延迟。使用指数移动平均线 (EMA) 进行计算。相关的调用包括：describeChannel、createChannel、getChannelEndpoint 和 deleteChannel。
dpApiCallLatency	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。相关的调用包括：getIceConfig。
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

指标	描述
	间。否则，会发出值为 0。这与信令客户端正常运行时间不同，因为连接断断续续，但是 signalingClientUptime 表示客户端对象本身。
numberOfMessagesSent	当对等设备发送提议、应答或 ICE 候选项时，此值会更新。
numberOfMessagesReceived	与 numberOfMessagesSent 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。
iceRefreshCount	当调用 getIceConfig 时，该值会递增。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。
numberOfRuntimeErrors	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。

指标	描述	
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。	

WebRTC 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 rtcPeerConnectionGetMetrics() 的第二个参数为 NULL，则返回列表中第一个收发器的数据。

rtcStatsObject 的定义可以在 Stats.h 中找到，RtcStats 的定义可以在 Include.h 中找到。

在 WebRTC C SDK 存储库的示例目录和 Kinesis Video Stream 演示存储库中可以找到这些 API 的示例用法和不同指标。

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

- 联网
- 媒体
- 数据通道

联网

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	

指标	描述	
	会通过 UDP 尝试 TURN，否则 ICE 会通过 TCP/TLS 尝试 TURN。如果 URI 不包含传输，ICE 会通过 UDP 和 TCP/TLS 尝试 TURN。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 (远程调试端口)	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。	

指标	描述
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。
packetsSent	发送的数据包数。这是在 .调用中的 `writeFrame` 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 incomingDataHandler 时都会更新此信息。
bytesSent	这是在 `writeFrame()` 调用 `iceAgentSendPacket`

指标	描述	
	() 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 incomingDataHandler 时都会更新此信息。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。	
lastPacketReceivedTimestamp	在 `incomingDataHandler()` 中接收数据时更新此信息。可以与 packetsReceived 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 `transceiverOnFrame()` 回调中。	
firstRequestTimestamp	当第一个 STUN 绑定请求在 `iceAgentSendStunPacket()` 中成功发出时记录此信息。它可以与 lastRequestTimestamp 和 requestsSent 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 `iceAgentSendStunPacket()` 中成功发出时都会记录此信息。	

指标	描述
lastResponseTimestamp	每次收到 STUN 绑定响应时都会记录此信息。
totalRoundTripTime	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。
currentRoundTripTime	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。
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。
packetsDiscardedOnSend	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。

指标	描述	
bytesDiscardedOnSend	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 https://www.w3.org/TR/webrtc-stats/#sentrtpstats-dict* 的一部分，包含在出站统计信息中。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。	

指标	描述	
encoderImplementation	它由应用程序层作为 RtcEncoderStats 对象的一部分进行更新。	
packetsDiscardedOnSend	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。	
bytesDiscardedOnSend	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_KIND_VIDEO 时，该值才会递增。	
hugeFramesSent	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）并使用应用程序设置的 RtcEncoderStats 中的 targetBitrate 来获得。	
framesEncoded	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。	
keyFramesEncoded	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。	

指标	描述
framesDiscardedOnSend	当由于 iceAgentSendPacket 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，如果在发送时由于错误而丢弃了任何数据包，则 framesDiscardedOnSend 会失败。
frameWidth	理想情况下，这表示最后一个编码帧的帧宽。目前，该值由应用程序作为 RtcEncoderStats* 的一部分设置为一个值，意义不大。
frameHeight	理想情况下，这表示最后一个编码帧的帧高。目前，该值由应用程序作为 RtcEncoderStats 的一部分设置为一个值，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，该值由应用程序作为 RtcEncoderStats 的一部分设置的，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。

指标	描述	
firCount	该值在收到 FIR 数据包 (onRtcpPacket->onRtcpFIRPacket) 时更新。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	该值在收到 PLI 数据包 (onRtcpPacket->onRtcpPLIPacket) 时更新。它表示在一帧或多帧中丢失了一定数量的编码视频数据。	
sliCount	该值将在收到 SLI 数据包 (onRtcpPacket->onRtcpSLIPacket) 时更新。它表示丢包影响单个帧的频率。	
qualityLimitationResolutionChanges	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。	

指标	描述	
bytesDiscardedOnSend	当由于 iceAgentSendPacket 调用失败而导致帧发送失败时，此信息会更新。帧由一组数据包组成，这些数据包又由字节组成，当前，如果在发送时由于错误而丢弃了任何数据包，bytesDiscardedOnSend 将失败。	
retransmittedPacketsSent	在接收 PLI/SLI/NACK 时重新传输的数据包数量。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。	
retransmittedBytesSent	在接收 PLI/SLI/NACK 时重新传输的字节数。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。	
totalEncodedBytesTarget	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。	

指标	描述	
totalPacketSendDelay	目前将其设置为 0，因为 iceAgentSendPacket 会立即发送数据包。	

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。	
totalRoundTripTime	计算出的往返时间总和	
fractionLost	表示自上一次发送方/接收方 reporfractionLost 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。	
roundTripTimeMeasurements	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。	
jitterBufferEmittedCount	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。	
lastPacketReceivedTimestamp	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。	

数据通道

数据通道指标

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。	
dataChannelIdentifier	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。	
state	查询统计信息时数据通道的状态。目前，支持的两种状态是 RTC_DATA_CHANNEL_STATE_CONNECTING（通道创建时）和 RTC_DATA_CHANNEL_STATE_OPEN（在 onOpen() 事件中设置）。	
messagesSent	当 SDK 通过数据通道发送消息时，计数器会更新。	
bytesSent	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。	
messagesReceived	该指标在 `onMessage()` 回调中递增。	

指标	描述	
bytesReceived	该指标在 `onMessage()` 回调中生成。	

Amazon Kinesis Video Streams WebRTC 提取

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 和云端 API 来实现实时流式传输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 Vide o Streams，然后使用我们的 API 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 API 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，还可以通过与 Amazon Rekognition Video 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

- API 操作
- WebRTC 提取和存储入门指南
- 创建 Kinesis Video Streams with WebRTC 信令通道
- 创建流
- 配置媒体提取和存储
- 提取媒体
- 查看 Kinesis Video Streams 流中的媒体

API 操作

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

- DescribeMappedResourceConfiguration
- DescribeMediaStorageConfiguration
- JoinStorageSession
- UpdateMediaStorageConfiguration

WebRTC 提取和存储入门指南

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 和云 API 的 step-by-step 说明，以实现

云端的实时流媒体和媒体接入。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”。

创建 Kinesis Video Streams with WebRTC 信令通道

有两种创建信令通道的方法，即 AWS Management Console 或。 AWS CLI

使用创建信令信道 AWS Management Console

1. 在中 AWS Management Console，打开 Kinesis Video Streams 控制台。
2. 左侧导航中，选择信令通道。

   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 T ime-to-live (Ttl) 值保留为 60 秒。
4. 选择 Create signaling channel (创建信令通道)。
5. 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。

   记下通道 ARN。

使用创建信令信道 AWS CLI

在中 AWS CLI，键入：

```
$ aws kinesisvideo create-signaling-channel --channel-name your-channel-name --region
 us-west-2
```

创建流

有两种创建直播的方法，即 AWS Management Console 或 AWS CLI。

> **Important**
>
> WebRTC 提取需要启用具有数据留存功能的 Amazon Kinesis Video Streams。

使用创建直播 AWS Management Console

1. 在中 AWS Management Console，打开 Kinesis Video Streams 控制台。
2. 在左侧导航窗格中，选择 Dashboard。

   选择 Create video stream (创建视频流)。
3. 在创建新视频流页面上，键入测试流的名称。

   使用默认配置。
4. 选择 Create video stream (创建视频流)。
5. 创建流后，请在视频流页面上查看详细信息。

   记录流 ARN。

使用创建直播 AWS CLI

在中 AWS CLI，键入：

```
$ aws kinesisvideo create-stream --stream-name your-stream-name --region us-west-2 --
data-retention-in-hours 24
```

配置媒体提取和存储

在中 AWS CLI，设置媒体存储配置。

如果您使用配置直播和/或频道，请复制并粘贴这些步骤中的资源 ARN。 AWS Management Console

如果您使用配置直播和/或频道，请执行以下操作来检索资源 ARNS。 AWS CLI

- 要检索频道 ARN，请键入：

  ```
  $ aws kinesisvideo describe-signaling-channel --channel-name your-channel-name --
  region us-west-2
  ```
- 要检索流 ARN，请键入：

  ```
  $ aws kinesisvideo describe-stream --stream-name your-stream-name --region us-west-2
  ```

获得 ARN 后，请设置媒体存储配置。类型：

```
$ aws kinesisvideo update-media-storage-configuration \
    --channel-arn your-arn \
    --media-storage-configuration \
        StreamARN="your-stream-arn",Status="ENABLED" \
    --region us-west-2
```

> **Important**
>
> 如果启用，StorageStatus则不再存在直接 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。

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

从浏览器提取媒体

> **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。 JavaScript
2. 完成以下信息：
   - KVS 端点。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。该示例应用程序支持临时凭证和长期凭证。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。
  - 信令通道。在通道名称字段中，键入您之前配置的信令通道的名称。有关更多信息，请参阅 the section called “配置媒体提取和存储”。
  - 跟踪。选择发送视频和发送音频。
  - WebRTC 摄取和存储。选择自动加入摄取和存储对等连接。
3. 选择启动主设备。

   如果使用 API 将信令通道配置为摄取，则示例应用程序将自动调用 DescribeMediaStorageConfigurationAPI 来启动 WebRT JoinStorageSessionC 摄取工作流程。

从 WebRTC C SDK 中提取媒体

按照 the section called “C 中适用于嵌入式设备的 WebRTC 开发工具包” 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 主示例

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。

查看 Kinesis Video Streams 流中的媒体

1. 打开 Amazon Kinesis Video Streams 媒体查看器。
2. 完成以下字段：
   - 区域。选择 us-west-2。
   - AWS 访问密钥
   - AWS 密钥
   - 流名称
   - 播放模式。选择实时。
3. 选择开始播放。

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 AWS Identity and Access Management控制对 Kinesis Video Streams with WebRTC 的访问
- Amazon Kinesis Video Streams with WebRTC 的合规性验证
- Kinesis Video Streams with WebRTC 中的弹性
- Kinesis Video Streams with WebRTC 中的基础设施安全
- 适用于 Kinesis Video Streams with WebRTC 的安全最佳实践
- WebRTC 加密

使用 AWS Identity and Access Management控制对 Kinesis Video Streams with WebRTC 的访问

通过将 AWS Identity and Access Management(IAM) 与 Amazon Kinesis Video Streams with WebRTC 配合使用，您可以控制组织中的用户能否使用特定的 Kinesis Video Streams with WebRTC API 操作执行某项任务，以及他们能否使用特定的 AWS资源。

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

目录

- 策略语法
- 适用于 Kinesis Video Streams with WebRTC 的操作
- Kinesis Video Streams 的 Amazon 资源名称（ARN）
- 授予其他 IAM 账户访问 Kinesis 视频流的权限
- Kinesis Video Streams with WebRTC 的示例策略

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)。

- Condition：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。

适用于 Kinesis Video Streams with WebRTC 的操作

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*"
```

若要指定所有 Kinesis Video Streams 操作，请使用 * (星号) 通配符，如下所示：

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

有关 Kinesis Video Streams API 操作的完整列表，请参阅 Kinesis Video Streams API 参考。

Kinesis Video Streams 的 Amazon 资源名称（ARN）

每个 IAM policy 语句适用于您使用资源的 ARN 指定的资源。

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 DescribeSignalingChannel 获取通道的 ARN。

授予其他 IAM 账户访问 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。

Kinesis Video Streams with WebRTC 的示例策略

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。

```
{
    "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。

```
{
    "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。

```
{
    "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：

- 安全与合规性快速入门指南 — 这些部署指南讨论了架构注意事项，并提供了在这些基础上 AWS 部署以安全性和合规性为重点的基准环境的步骤。
- 在 A@@ mazon Web Services 上构建 HIPAA 安全与合规架构 — 本白皮书描述了各公司如何使用 AWS 来创建符合 HIPAA 资格的应用程序。

> **Note**
>
> 并非所有 AWS 服务 人都符合 HIPAA 资格。有关更多信息，请参阅符合 HIPAA 要求的服务参考。

- 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、行业准则和法规。
- 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 通过安全控件评估您的 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件的列表，请参阅 Security Hub 控件参考。
- 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。

Kinesis Video Streams with WebRTC 中的弹性

AWS 全球基础设施围绕 AWS 区域 和可用区构建。AWS 区域 提供多个在物理上独立且隔离的可用区，这些可用区与延迟率低、吞吐量高且冗余性高的网络连接在一起。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设施相比，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。

Kinesis Video Streams with WebRTC 中的基础设施安全

作为一项托管服务，Kinesis Video Streams（包含其 WebRTC 功能）由 Amazon Web Services: Overview of Security Processes 白皮书中所述的 AWS全球网络安全流程提供保护。

您可以使用 AWS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。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 委托人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。

适用于 Kinesis Video Streams with WebRTC 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您的环境或不满足您的环境要求，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，请参阅什么是最低权限以及为什么需要它？

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。您不能将 AWS 凭证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
- 针对角色的常见情形：用户、应用程序和服务

使用 CloudTrail 监控 API 调用

Kinesis Video Streams with WebRTC 与 AWS CloudTrail相集成，后者是一项服务，可提供用户、角色或 AWS服务在 Kinesis Video Streams with WebRTC 中所采取的操作的记录。

使用 CloudTrail 收集的信息，您可以确定向 Kinesis Video Streams with WebRTC 发出了什么请求、发出请求的 IP 地址、何人发出的请求、请求的发出时间以及其他详细信息。

有关更多信息，请参阅the section called “使用 WebRTC API 调用记录 Kinesis Video Streams AWS 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

监控是保持 Amazon Kinesis Video Streams with WebRTC 和您的 AWS解决方案的可靠性、可用性和性能的重要方面。您应从 AWS 解决方案的所有部分收集监控数据，以便更轻松地调试出现的多点故障。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

- 通过 CloudWatch 监控 Kinesis Video Streams with WebRTC
- 使用 WebRTC API 调用记录 Kinesis Video Streams AWS CloudTrail

通过 CloudWatch 监控 Kinesis Video Streams with WebRTC

您可以使用 Amazon CloudWatch 监控 Kinesis Video Streams with WebRTC，Amazon CloudWatch 从Kinesis Video Streams with WebRTC 中收集原始数据，并将其转化为可读、接近实时的指标。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。

Kinesis Video Streams 提供以下指标：

主题

- 信令指标
- TURN 指标

信令指标

指标名称	维度	单位	描述	
Failure	操作，SignalingChannelName	计数	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”。	
延迟	操作，SignalingChannelName	毫秒	从服务接收请求到服务返回响应期间测量的时间。	
MessagesTransferred.Count	SignalingChannelName	计数	针对给定频道传输（发送和接收）的消息总数。	

`Operation` 维度可应用于以下任何 API：

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

TURN 指标

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

使用 WebRTC API 调用记录 Kinesis Video Streams 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。如果您未配置跟踪，您仍然可以在 CloudTrail控制台的 “事件历史记录” 中查看最新的事件。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

带有 WebRTC 的 Amazon Kinesis Video Streams 支持将以下操作作为事件记录在日志文件中：CloudTrail

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

每个事件或日记账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 用户身份元素。

示例：Amazon Kinesis Video Streams with WebRTC 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的顺序出现。

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"
}
```

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。

主题

- 服务限额
- 联网要求
- 网络环境
- 建立会话相关问题
- 调试正在进行的连接

服务限额

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。

自 2023 年 2 月起，无法将多个主设备连接到单个信令通道。有关 Athena 服务限额的额外信息，请参阅 配额。

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

作为 RTC 的一部分，对等体之间使用的协议PeerConnection 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` IP 的 ICE 候选对，例如 `192.168.*` 或 `10.0.0.*`。
> 这可能会导致一些（但不是全部）ICE 候选对失败。

建立会话相关问题

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。

- 提议由“查看器”生成，查看器希望在 Kinesis Video Streams with WebRTC 中作为“主设备”与连接到信令通道的对等设备建立联系。
- 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，你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。

要从 SDK 将 SDP 记录到 `stdout`，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 “通过 CloudWatch 监控 Kinesis Video Streams with WebRTC”。

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
...
```

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。

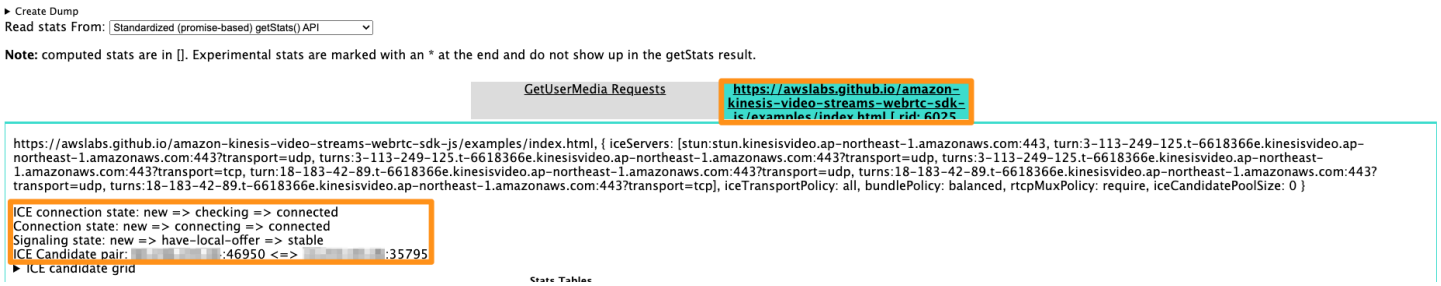
确定使用了哪些候选项来建立连接

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。

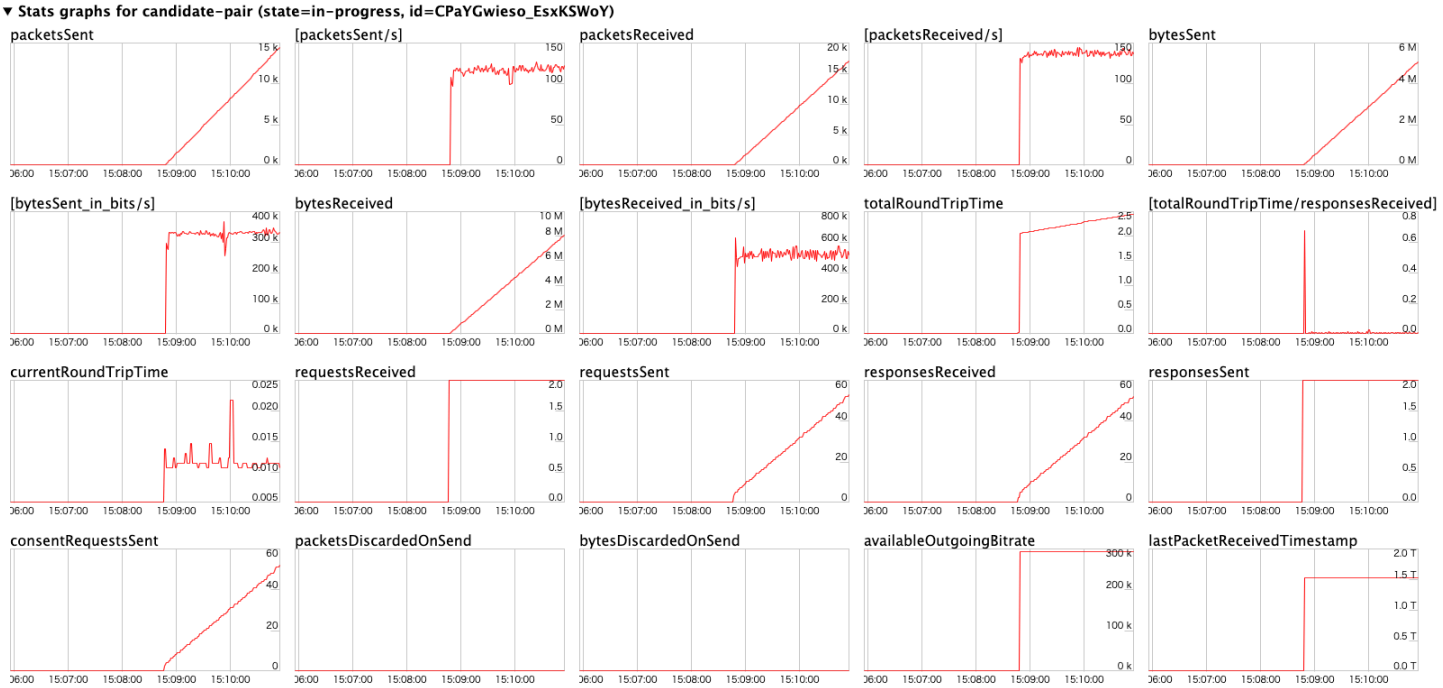
在一个浏览器选项卡中打开 WebRTC 会话。

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：



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，请将超时值设置为大于 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。增加该值可能有助于减少网络负载，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。

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

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

Amazon Kinesis Video Streams with WebRTC 开发人员指南的文档历史记录

变更	说明	日期
初次发布	初次发布 Amazon Kinesis Video Streams with WebRTC 开发人员指南。了解更多	2019 年 11 月 4 日

AWS 术语表

有关最新的 AWS 术语，请参阅《AWS 词汇表参考》中的 AWS 词汇表。

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。