



开发人员指南

Amazon Kinesis Video Streams



Amazon Kinesis Video Streams: 开发人员指南

Copyright © 2024 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

什么是 Kinesis Video Streams ?	1
区域可用性	2
你是首次使用 Kinesis Video Streams 吗?	3
系统要求	4
相机要求	4
经过测试的操作系统	5
SDK 存储要求	5
工作方式	6
API 和创建者库	7
Kinesis Video Streams	7
制作者库	9
视频播放	9
播放要求	10
使用 HLS 播放视频	12
使用 MPEG-DASH 播放视频	21
使用流式处理元数据	25
向 Kinesis 视频流添加元数据	26
使用 Kinesis 视频流中嵌入的元数据	27
流式传输元数据限制	28
数据模型	29
流标头元素	29
流式传输曲目数据	33
帧头元素	34
MKV 帧数据	35
开始使用	36
设置了账户	36
注册获取 AWS 账户	37
创建管理用户	37
创建密 AWS 账户 钥	38
创建 Kinesis 视频流	38
使用控制台创建视频流	38
使用创建视频流 AWS CLI	39
向 Amazon Kinesis 视频流发送数据	39
构建 SDK 和示例	39

运行示例将媒体上传到 Kinesis Video Streams	42
查看确认对象	44
使用媒体数据	44
在控制台中查看媒体	44
使用 HLS 使用媒体数据	44
边缘代理	45
亚马逊 Kinesis Video Streams 边缘代理 API 操作	46
监控 Amazon Kinesis Video Streams 边缘代理	46
在非AWS IoT Greengrass模式下部署	46
1. 安装依赖项。	47
2. 为您的 IP 摄像机 RTSP 网址创建资源	48
3. 创建 IAM 权限策略	50
4. 创建 IAM 角色	52
5. 创建AWS IoT角色别名	53
6. 创建 AWS IoT 策略	54
7. 创建AWS IoT事物并获取AWS IoT Core凭证	55
8. 构建并运行 Amazon Kinesis Video Streams 边缘代理	58
9. (可选) 安装代 CloudWatch 理	67
10. (可选) 运行 Amazon Kinesis Video Streams 边缘代理	70
部署到 AWS IoT Greengrass	73
1. 创建 Ubuntu 实例	73
2. 设置AWS IoT Greengrass核心设备	74
3. 为您的 IP 摄像机 RTSP 网址创建资源	75
4. 为 TES 角色添加权限	77
5. 安装密钥管理器组件	80
6. 在设备上部署 Amazon Kinesis Video Streams 边缘代理	82
7. (可选) 安装AWS IoT Greengrass日志管理器组件	90
常见问题解答	93
亚马逊 Kinesis Video Streams Video Streams Edge Agent 支持哪些操作系统？	93
亚马逊 Kinesis Video Streams Video Streams Edge Agent 是否支持 H.265 媒体？	94
亚马逊 Kinesis Video Streams Video Streams Edge Agent 能在 AL2 中运行吗？	94
我怎样才能能在AWS IoT事物或设备中运行多个直播？	94
发送StartEdgeConfigurationUpdate后如何编辑？	94
你有常见的例子ScheduleConfigs吗？	94
有最大直播限制吗？	95
如何重启出错的作业？	95

如何监控我的亚马逊 Kinesis Video Streams Edge Agent 的运行状况？	96
通过 VPC 流式传输视频	97
其他信息	97
VPC 终端节点程序	97
映像	100
开始使用 GetImages	100
亚马逊 S3 交付入门	100
UpdateImageGenerationConfiguration	101
DescribeImageGenerationConfiguration	103
制片人 MKV 标签	104
使用 Producer SDK 添加元数据标签PutEventMetaData	104
Limits	105
S3 对象元数据	105
S3 对象路径 (图片)	105
防止限制的 Amazon S3 URI 建议	106
通知	107
UpdateNotificationConfiguration	107
DescribeNotificationConfiguration	107
制作人 MKV 标签	104
制作人 MKV 标签的语法	104
MKV 标签限制	108
.....	108
.....	108
亚马逊 SNS 主题有效负载	108
查看您的亚马逊 SNS 消息	109
安全性	110
数据保护	110
什么是 Kinesis Video Streams 的服务器端加密？	111
成本、地区和性能注意事项	111
如何开始使用服务器端加密？	112
创建和使用用户生成的 KMS 密钥	112
使用用户生成的 KMS 密钥的权限	113
使用 IAM 控制对 Kinesis Video Streams 资源的访问权限	115
策略语法	115
Kinesis Video Streams 的操作	116
Kinesis Video Streams 的 Amazon 资源名称 (ARN)	116

向其他 IAM 账户授予访问 Kinesis 视频流的权限	117
示例策略	117
使用控制对 Kinesis Video Streams 资源的访问权限 AWS IoT	119
AWS IoT ThingName 作为直播名称	120
AWS IoT CertificateId 作为直播名称	125
使用 AWS IoT 凭据直播到硬编码的直播名称	127
监控	127
合规性验证	128
韧性	128
基础设施安全性	129
安全最佳实践	129
实施最低权限访问	129
使用 IAM 角色	130
CloudTrail 用于监控 API 调用	130
创建者库	131
Kinesis 视频直播制作人客户端	131
Kinesis 视频直播制作人库	132
相关主题	132
Java 创建者库	133
过程：使用 Java 创建者开发工具包	133
步骤 1：下载并配置代码	134
第 2 步：编写并检查代码	135
步骤 3：运行并验证代码	137
Android 创建者库	137
过程：使用 Android 创建者开发工具包	138
先决条件	138
步骤 1：下载并配置代码	141
步骤 2：检查代码	143
步骤 3：运行并验证代码	145
C++ 创建者库	146
物体模型	146
将媒体放到直播中	146
回调接口	147
过程：使用 C++ 创建者开发工具包	147
步骤 1：下载并配置代码	150
步骤 2：编写并检查代码	150

步骤 3：运行并验证代码	156
使用 C++ Produce SED SEDER SEDER	157
在 Docker 容器中使用 C++ Producer SDK 作为 gStreamer 插件	157
使用日志记录	157
C 创建者库	158
物体模型	158
将媒体放到直播中	159
过程：使用 C 创建者开发工具包	159
第 1 步：下载代码	162
步骤 2：编写和检查代码	162
第 3 步：运行代码	165
树莓派上的 C++ 制作人 SDK	167
先决条件	167
创建有权写入 Kinesis Video Streams 的 IAM 用户	168
加入你的 Raspberry Pi 到 Wi-Fi 网络	169
远程连接到你的 Raspberry Pi	170
配置树莓派摄像头	170
安装软件先决条件	171
下载并构建 Kinesis Video Streams C++ Producer SDK	172
将视频流式传输到您的 Kinesis 视频流并观看直播	172
参考	173
制作人 SDK 限制	174
错误代码参考	176
NAL 适配标志	218
创建者结构	219
流结构	221
回调	234
视频流解析器库	241
步骤：使用 Kinesis 视频流解析器库	241
先决条件	241
步骤 1：下载和配置代码	241
下一步	242
步骤 2：编写和检查代码	242
StreamingMkvReader	242
FragmentMetadataVisitor	243
OutputSegmentMerger	244

KinesisVideoExample	246
下一步	249
步骤 3：运行并验证代码	249
示例	250
示例：向 Kinesis Video Streams 发送数据	250
示例：从 Kinesis Video Streams 检索数据	250
示例：播放视频数据	250
先决条件	250
GStreamer	251
下载、构建和配置 gStreamer 元素	252
运行 gStreamer 元素	252
启动命令	253
在 Docker 容器中运行 gStreamer 元素	255
参数参考	258
PutMedia API	268
步骤 1：下载并配置代码	268
第 2 步：编写并检查代码	269
步骤 3：运行并验证代码	271
RTSP 和 Docker	272
先决条件	273
构建 Docker 镜像	273
运行 RTSP 示例应用程序	274
渲染器	275
先决条件	275
运行渲染器示例	276
工作方式	277
SageMaker	278
先决条件	279
创建应用程序	280
监视应用程序	282
扩展应用程序	283
清理应用程序	284
监控	286
使用监控指标 CloudWatch	286
CloudWatch 指标指导	298
使用以下方式监控 Amazon Kinesis Video Streams Edge Agent CloudWatch	302

CloudWatch 亚马逊 Kinesis Video Streams Edge Agent 的指标指南	304
使用 记录 CloudTrail API 调用	306
亚马逊 Kinesis Video Streams 和 CloudTrail	306
示例：亚马逊 Kinesis Video Streams Video Streams 日志文件条目	307
配额	312
控制平面 API 服务配额	312
媒体和存档媒体 API 服务配额	317
片段元数据和片段媒体配额	320
片段元数据配额	322
直播标签	323
故障排除	324
排查一般问题	324
延迟太高	324
排查 API 问题	325
错误：“Unknown options”	325
错误：“Unable to determine service/operation name to be authorized (无法确定要授权的服务/操作名称)”	325
错误：“Failed to put a frame in the stream (无法将帧放入流)”	326
错误：“服务在收到最终版本之前关闭 AckEvent 了连接”	326
错误：“STATUS_STORE_OUT_OF_MEMORY”	326
HLS 问题疑难解答	326
解决 Java 问题	326
启用 Java 日志	327
对制作器库问题进行故障排除	327
无法编译创建者开发工具包	328
视频流未显示在控制台中	328
在采用 GStreamer 演示应用程序流式处理数据时，出现错误：“Security token included in the request is invalid”	329
错误：“Failed to submit frame to Kinesis Video client”(无法将帧提交到 Kinesis 视频客户端)	329
GStreamer 应用程序停止运行，OS X 上显示消息 "streaming stopped, reason not-negotiated"	329
当在 Raspberry Pi 上的 GStreamer 演示应用程序中创建 Kinesis 视频客户端时，出现错误：“Failed to allocate heap”	330
当在 Raspberry Pi 上运行 GStreamer 演示应用程序时，出现错误：“Illegal Instruction”	330
摄像机在 Raspberry Pi 上加载失败	330

在 macOS High Sierra 上未找到摄像机	331
在 macOS High Sierra 上编译时，找不到 jni.h 文件	331
在运行 GStreamer 演示应用程序时出现 Curl 错误	331
Raspberry Pi 上运行时的时间戳/范围断言	332
Raspberry Pi 上的 gst_value_set_fraction_range_full 断言	332
Android 上的 STATUS_MKV_INVALID_ANNEXB_NALU_IN_FRAME_DATA (0x3200000d) 错误	332
已达到最大片段持续时间错误	332
使用 IoT 授权时出现“Invalid thing name passed (传递的事物名称无效)”错误	333
对流解析器库问题进行故障排除	333
无法从流中访问单个帧	333
片段解码错误	333
文档历史记录	335
API 参考	338
操作	338
Amazon Kinesis Video Streams	339
Amazon Kinesis Video Streams	458
Amazon Kinesis Video Streams 存档媒体	473
Amazon Kinesis Video	515
Amazon Kinesis Kinesis WebRTC	524
数据类型	528
Amazon Kinesis Video Streams	529
Amazon Kinesis Video Streams	569
Amazon Kinesis Video Streams 存档媒体	572
Amazon Kinesis Video Streams	590
Amazon Kinesis Video Streams	592
常见错误	592
常见参数	594
.....	dx cvi

什么是 Kinesis Video Streams ?

您可以使用完全AWS 服务托管的 Amazon Kinesis Video Streams 将直播视频从设备流式传输到AWS Cloud设备，或者构建用于实时视频处理或批处理视频分析的应用程序。

Kinesis Video Streams 不仅可以存储视频数据。您还可以用它来实时监控视频流，因为这些流在云中接收。您可以在中监控直播AWS Management Console，也可以开发自己的监控应用程序，使用 Kinesis Video Streams API 库来显示直播视频。

您可以使用 Kinesis Video Streams 捕获来自数百万个来源的大量实时视频数据，包括智能手机、安全摄像头、网络摄像头、嵌入在汽车中的摄像头、无人机和其他来源。您还可以发送非视频、时间序列化数据，例如音频数据、热成像、深度数据和雷达数据。当直播视频从这些来源流到 Kinesis 视频流时，您可以构建应用程序来实时访问数据 frame-by-frame，从而实现低延迟处理。Kinesis Video Streams 与来源无关。您可以使用[GStreamer](#)库从计算机的网络摄像头流式传输视频，也可以使用实时流式传输协议 (RTSP) 从网络上的摄像头流式传输视频。

您还可以将 Kinesis 视频流配置为在指定保留期内持久存储媒体数据。Kinesis Video Streams 会自动存储这些数据并在静态状态下对其进行加密。此外，Kinesis Video Streams 还根据制作者时间戳和摄取时间戳对存储的数据进行时间索引。您可以构建定期批处理视频数据的应用程序，也可以创建需要一次性访问不同用例的历史数据的应用程序。

您的自定义应用程序，无论是实时应用程序还是面向批处理的应用程序，都可以在 Amazon EC2 实例上运行。这些应用程序可能使用开源、深度学习算法来处理数据，或者使用与 Kinesis Video Streams 集成的第三方应用程序。

使用 Kinesis Video Streams 的好处包括以下几点：

- 连接数百万台设备并进行流式传输 — 您可以使用 Kinesis Video Streams 连接和流式传输来自数百万台设备（包括消费类智能手机、无人机和行车记录仪）的视频、音频和其他数据。您可以使用 Kinesis Video Streams 制作器库来配置您的设备，并以实时或媒体上传的 after-the-fact 形式可靠地进行流式传输。
- 持久存储、加密和索引数据 — 您可以将 Kinesis 视频流配置为在自定义保留期内持久存储媒体数据。Kinesis Video Streams 还根据制作人生成的时间戳或服务端的时间戳为存储的数据生成索引。您的应用程序可以使用时间索引检索流中的指定数据。
- 专注于管理应用程序而不是基础架构 — Kinesis Video Streams 是无服务器的，因此无需设置或管理基础架构。您无需担心底层基础设施的部署、配置或弹性扩展，因为您的数据流和使用应用程序的数量会不断增长和缩小。Kinesis Video Streams 会自动完成管理直播所需的所有管理和维护，因此您可以专注于应用程序，而不是基础架构。

- 在数据流上构建实时和批处理应用程序 — 您可以使用 Kinesis Video Streams 构建在实时数据流上运行的自定义实时应用程序，并创建批处理或一次性应用程序，这些应用程序可以在没有严格延迟要求的情况下对持久持久的数据进行操作。您可以构建、部署和管理自定义应用程序：开源（Apache MxNet、OpenCV）、本土解决方案或使用处理和分析直播AWS Marketplace的第三方解决方案。您可以使用 Kinesis Video Get Streams API 来构建多个并发应用程序，以实时或面向批处理的方式处理数据。
- 更安全地传输数据 — Kinesis Video Streams 在所有数据流经服务时以及保留数据时对其进行加密。Kinesis Video Streams 对来自设备的数据流强制执行基于传输层安全 (TLS) 的加密，并使用 () 对所有静AWS Key Management Service态数据进行加密。AWS KMS此外，您可以使用 AWS Identity and Access Management (IAM) 管理对数据的访问权限。
- 即@@ 用即付-有关更多信息，请参阅[AWS Pricing Calculator](#)。

区域可用性

亚马逊 Kinesis Video Streams 在以下地区推出：

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
亚太地区（东京）	ap-northeast-1

区域名称	AWS地区代码
加拿大 (中部)	ca-central-1
中国 (北京)	cn-north-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

你是首次使用 Kinesis Video Streams 吗？

如果你是首次使用 Kinesis Video Streams 的用户，我们建议你按顺序阅读以下章节：

1. [Kinesis Video Streams : 工作原理](#)— 了解 Kinesis Video Streams 的概念。
2. [亚马逊 Kinesis Video Streams 入门](#)— 设置你的账户并测试 Kinesis Video Streams。
3. [Kinesis 视频直播制作器库](#)— 了解如何创建 Kinesis Video Streams 制作器应用程序。
4. [Kinesis 视频直播解析器库](#)— 了解如何在 Kinesis Video Streams 消费者应用程序中处理传入的数据帧。
5. [亚马逊 Kinesis Video Streams 示例](#)— 查看更多关于你可以使用 Kinesis Video Streams 做什么的示例。

Kinesis Video Streams 系统要求

以下各节包含亚马逊 Kinesis Video Streams Amazon Kinesis Video Streams 的硬件、软件和存储要求。

主题

- [相机要求](#)
- [经过测试的操作系统](#)
- [SDK 存储要求](#)

相机要求

用于运行 Kinesis Video Streams Producer SDK 和样本的摄像机具有以下内存要求：

- 该开发工具包内容视图需要 16 MB 的内存。
- 示例应用程序的默认配置是 128 MiB 的内存。此值适用于网络连接良好而无需额外缓冲的创建者。如果网络连接不佳而需要更多缓冲，可通过将每秒帧率乘以帧内存大小来计算每秒缓冲的内存要求。有关分配内存的更多信息，请参阅 [StorageInfo](#)。

建议使用 USB 或 RTSP (实时流协议) 摄像机 (使用 H.264 编码数据)，因为这会消除 CPU 的编码工作负载。

目前，该演示应用程序不支持用于 RTSP 流式传输的用户数据报协议 (UDP)。将来会添加这种功能。

创建者开发工具包支持以下类型的摄像机：

- 网络摄像机。
- USB 摄像机。
- 采用 H.264 编码的摄像机 (首选)。
- 不采用 H.264 编码的摄像机。
- Raspberry Pi 摄像机模块。这是 Raspberry Pi 设备的首选摄像机，因为它连接到 GPU 进行视频数据传输，所以没有 CPU 处理开销。
- RTSP (网络) 摄像机。这些是首选摄像机，因为视频流已采用 H.264 进行编码。

经过测试的操作系统

我们已使用以下设备和操作系统测试了网络摄像机和 RTSP 摄像机：

- Mac mini
 - High Sierra
- MacBook 专业笔记本电脑
 - Sierra (10.12)
 - El Capitan (10.11)
- 运行 Ubuntu 16.04 的惠普笔记本电脑
- Ubuntu 17.10 (Docker 容器)
- Raspberry Pi 3

SDK 存储要求

安装 [Kinesis 视频直播制作人库](#) 的最小存储要求为 170 MB，而建议的存储要求为 512 MB。

Kinesis Video Streams : 工作原理

主题

- [Kinesis Video Streams API 和制作人库](#)
- [Kinesis Video Streams 回放](#)
- [在 Kinesis Video Streams 中使用流式传输元数据](#)
- [Kinesis Video Streams 数据模型](#)

您可以使用完全托管AWS 服务的 Amazon Kinesis Video Streams 将直播视频从设备流式传输到设备 AWS Cloud并对其进行持久存储。之后，您可以构建用于实时视频处理的应用程序或执行面向批处理的视频分析。

下图提供了Kinesis Video Streams 如何的概述。

该图演示了以下组件之间的交互：

- 制作者 — 任何将数据放入 Kinesis 视频流的来源。创建者可以是任何视频生成设备，例如，安保摄像头、随身摄像机、智能手机摄像头或车载摄像机。创建者还可以发送非视频数据，例如音频源、图像或雷达数据。

单个创建者可以生成一个或多个视频流。例如，摄像机可以将视频数据推送到一个 Kinesis 视频流，将音频数据推送到另一个视频流。

- Kinesis Video Streams Producer 库 — 一组可以在设备上安装和配置的软件和库。您可以使用这些库以不同的方式安全地连接和可靠地流式传输视频，包括实时、缓冲几秒钟后或作为after-the-fact媒体上传。
- Kinesis 视频流 — 一种资源，您可以使用它来传输实时视频数据、选择性地存储这些数据，并使这些数据既可以实时使用，也可以批量或一次性使用。在典型配置中，Kinesis 视频流只有一个制作者向其中发布数据。

流可以传输音频、视频和类似的时间编码的数据流，如深度感应源、雷达源等。您可以使用AWS Management Console或以编程方式使用 SDK 创建 Kinesis 视频流。AWS

多个独立应用程序可以parallel 使用 Kinesis 视频流。

- 消费者-从 Kinesis 视频流中获取片段和帧等数据，以进行查看、处理或分析。通常，这些消费者被称为 Kinesis Video Streams 应用程序。您可以编写用于实时消耗和处理 Kinesis Video Streams 中数据的应用程序，或者在不需要低延迟处理时存储数据并编制时间索引之后编写应用程序。您可以创建这些在 Amazon EC2 实例上运行。
- [Kinesis 视频直播解析器库](#)— 使 Kinesis Video Streams 应用程序能够以低延迟的方式可靠地从 Kinesis 视频流中获取媒体。此外，它将解析媒体中的帧边界，以便应用程序可以集中处理和分析帧本身。

Kinesis Video Streams API 和制作人库

Kinesis Video Streams 提供 API，供您创建和管理直播以及在流中读取或写入媒体数据。除了管理功能外，Kinesis Video Streams 控制台还支持直播和video-on-demand播放。Kinesis Video Streams 还提供了一组制作者库，您可以在应用程序代码中使用这些库从媒体源提取数据并上传到 Kinesis 视频流。

主题

- [Kinesis Video Streams](#)
- [制作者库](#)

Kinesis Video Streams

Kinesis Video Streams 提供用于创建和管理 Kinesis Video Streams 的 API。它还提供了用于在流中读取和写入媒体数据的 API，如下所示：

- 制作者 API — Kinesis Video Streams 提供了一个将媒体数据写入 Kinesis 视频流的 PutMedia API。在 PutMedia 请求中，创建者将发送一个媒体片段流。片段 是一系列独立帧。属于某个片段的帧不依赖其他片段中的任何帧。有关更多信息，请参阅[PutMedia](#)：

片段到达时，Kinesis Video Streams 会按递增顺序分配一个唯一的片段编号。它还将每个片段的制作者端和服务端时间戳存储为 Kinesis Video Streams 特定的元数据。

- 消费者 API — 消费者可以使用以下 API 从流中获取数据：
 - GetMedia - 在使用此 API 时，使用者必须标识正在启动的片段。之后，API 将按照将片段添加到流中的顺序返回片段 (按照片段号的递增顺序)。片段中的媒体数据将打包成一个结构化格式，例如 [Matroska \(MKV\)](#)。有关更多信息，请参阅[GetMedia](#)：

Note

GetMedia 了解片段所在位置 (存档到数据存储中或实时可用)。例如, 如果 GetMedia 确定已存档正在启动的片段, 则它会开始从数据存储返回片段。当它必须返回尚未存档的新片段时, GetMedia 切换到从内存流缓冲区读取片段。

这是一个持续使用者的示例, 它按照流提取片的顺序处理片段。

GetMedia 使视频处理应用程序能够先失败或落后, 然后保持同步, 而无需执行其他操作。通过使用 GetMedia, 应用程序可以处理已存档到数据存储中的数据, 并且在应用程序保持同步时, GetMedia 仍会在其到达时实时提供媒体数据。

- GetMediaFromFragmentList (和 ListFragments) - 批处理应用程序被视为离线使用者。离线使用者可能选择通过将 ListFragments 和 GetMediaFromFragmentList API 组合使用, 来明确提取视频的特定媒体片段或范围。ListFragments 和 GetMediaFromFragmentList 使应用程序能够标识特定时间范围或片段范围的视频片段, 然后按顺序或并行提取这些片段以进行处理。此方法适用于 MapReduce 应用程序套件, 该套件必须并行快速处理大量数据。

例如, 假设使用者需要处理一天的视频片段。使用者将执行以下操作:

1. 通过调用 ListFragments API 并指定时间范围以获取片段列表, 从而选择所需的片段集合。

API 将返回指定时间范围的所有片段中的元数据。元数据提供诸如片段编号、生产者端和服务端时间戳等信息。

2. 获取片段元数据列表并按任意顺序检索片段。例如, 要处理当天的所有片段, 使用者可以选择将列表拆分为子列表, 让工作人员 (例如, 多个 Amazon EC2 实例) 使用 parallel 提取碎片, 然后 parallel 处理它们。GetMediaFromFragmentList

下图显示了这些 API 调用期间片段和数据块的数据流。

当创建者发送 PutMedia 请求时, 它会在负载中发送媒体元数据, 然后发送一系列媒体数据片段。收到数据后, Kinesis Video Streams 将传入的媒体数据存储为 Kinesis Video Streams 区块。每个数据块均包含以下内容:

- 媒体元数据的副本
- 片段

- Kinesis Video Streams 特定的元数据；例如，片段编号以及服务器端和制作者端的时间戳

当消费者请求媒体元数据时，Kinesis Video Streams 会返回一个区块流，从您在请求中指定的片段编号开始。

如果您为流启用数据持久性，则在收到流中的片段后，Kinesis Video Streams 还会将该片段的副本保存到数据存储中。

制作者库

创建 Kinesis 视频流后，即可开始向该视频流发送数据。在应用程序代码中，您可以使用这些库从媒体源提取数据并上传到 Kinesis 视频流。有关可用创建者库的更多信息，请参阅 [Kinesis 视频直播制作者库](#)。

Kinesis Video Streams 回放

您可以使用以下方法查看 Kinesis 视频流：

- GetMedia— 您可以使用 GetMedia API 构建自己的应用程序来处理 Kinesis Video Streams。GetMedia 是一种低延迟的实时 API。要创建能使用的玩家 GetMedia，你必须自己构建。有关如何开发使用显示 Kinesis 视频流的应用程序的信息 GetMedia，请参阅 [视频流解析器库](#)
- HLS — [HTTP 直播 \(HLS\)](#) 是一种基于 HTTP 的行业标准媒体流通信协议。您可以使用 HLS 查看 Kinesis 视频流，用于实时播放或查看存档视频。

您可以使用 HLS 进行实时播放。延迟通常介于 3-5 秒之间，但可能介于 1-10 秒之间，具体取决于用例、玩家和网络条件。你可以使用第三方播放器（如 [Video.js](#) 或 [Google Shaka Player](#)），通过提供 HLS 流会话 URL 来以编程或手动方式显示视频流。你也可以通过在 [Apple Safari](#) 或 [Microsoft Edge](#) 浏览器的位置栏中输入 HLS 直播会话网址来播放视频。

- MPEG-DASH — 基于 [HTTP 的动态自适应流媒体 \(DASH\)](#)，也称为 MPEG-DASH，是一种自适应比特率流媒体协议，可通过互联网从传统 HTTP Web 服务器传输高质量的媒体内容。

您可以使用 MPEG-DASH 进行实时播放。延迟通常介于 3-5 秒之间，但可能介于 1-10 秒之间，具体取决于用例、玩家和网络条件。你可以通过编程或手动方式提供 MPEG-DASH 直播会话网址，使用第三方 [播放器（例如 dash.js 或 Google Shaka Player）](#) 来显示视频流。

- GetClip— 您可以使用 GetClip API 从指定时间范围内从指定视频流中下载包含已存档的点播媒体的片段（在 MP4 文件中）。有关详细信息，请参阅 [GetClip API 参考](#)。

主题

- [视频播放曲目要求](#)
- [使用 HLS 播放视频](#)
- [使用 MPEG-DASH 播放视频](#)

视频播放曲目要求

Amazon Kinesis Video Streams 支持以多种格式编码的媒体。如果您的 Kinesis 视频流使用的格式不受下面列出的四个 API 的支持，请使用[GetMedia](#)或 [GetMediaForFragmentList](#)，因为它们没有轨道类型限制。

主题

- [GetClip 要求](#)
- [GetDash StreamingSession 网址要求](#)
- [getHLS 网址要求 StreamingSession](#)
- [GetImages 要求](#)

GetClip 要求

相关此 API 的更多信息，请参阅 [GetClip](#)。

曲目 1 描述	轨道 1 编解码器 ID	曲目 2 描述	轨道 2 编解码器 ID
H.264 视频	V_MPEG/ISO/AVC	不适用	不适用
H.264 视频	V_MPEG/ISO/AVC	AAC 音频	A_AAC
H.264 视频	V_MPEG/ISO/AVC	G.711 音频 (仅限 A-Law)	A_MS/ACM
H.265 视频	V_MPEGH/ISO/HEVC	不适用	不适用
H.265 视频	V_MPEGH/ISO/HEVC	AAC 音频	A_AAC

GetDash StreamingSession 网址要求

相关此 API 的更多信息，请参阅 [GetDASHStreamingSessionURL](#)。

曲目 1 描述	轨道 1 编解码器 ID	曲目 2 描述	轨道 2 编解码器 ID
H.264 视频	V_MPEG/ISO/AVC	不适用	不适用
H.264 视频	V_MPEG/ISO/AVC	AAC 音频	A_AAC
H.264 视频	V_MPEG/ISO/AVC	G.711 音频 (仅限 A-Law)	A_MS/ACM
H.264 视频	V_MPEG/ISO/AVC	G.711 音频 (仅限 U-Law)	A_MS/ACM
AAC 音频	A_AAC	不适用	不适用
H.265 视频	V_MPEGH/ISO/HEVC	不适用	不适用
H.265 视频	V_MPEGH/ISO/HEVC	AAC 音频	A_AAC

getHLS 网址要求 StreamingSession

相关此 API 的更多信息，请参阅 [GetHLSStreamingSessionURL](#)。

HLS Mp4

曲目 1 描述	轨道 1 编解码器 ID	曲目 2 描述	轨道 2 编解码器 ID
H.264 视频	V_MPEG/ISO/AVC	不适用	不适用
H.264 视频	V_MPEG/ISO/AVC	AAC 音频	A_AAC
AAC 音频	A_AAC	不适用	不适用
H.265 视频	V_MPEGH/ISO/HEVC	不适用	不适用
H.265 视频	V_MPEGH/ISO/HEVC	AAC 音频	A_AAC

哈哈哈哈哈

曲目 1 描述	轨道 1 编解码器 ID	曲目 2 描述	轨道 2 编解码器 ID
H.264 视频	V_MPEG/ISO/AVC	不适用	不适用
H.264 视频	V_MPEG/ISO/AVC	AAC 音频	A_AAC
AAC 音频	A_AAC	不适用	不适用

GetImages 要求

相关此 API 的更多信息，请参阅 [GetImages](#)。

Note

GetImages 媒体应包含轨道 1 中的视频曲目。

使用 HLS 播放视频

[HTTP 直播 \(HLS\)](#) 是一种基于 HTTP 的行业标准媒体流通信协议。您可以使用 HLS 查看 Kinesis 视频流，用于实时播放或查看存档视频。

您可以使用 HLS 进行实时播放。延迟通常介于 3-5 秒之间，但可能介于 1-10 秒之间，具体取决于用例、玩家和网络条件。你可以使用第三方播放器（如 [Video.js](#) 或 [Google Shaka Player](#)），通过提供 HLS 流会话 URL 来以编程或手动方式显示视频流。你也可以通过在 [Apple Safari](#) 或 [Microsoft Edge](#) 浏览器的位置栏中输入 HLS 直播会话网址来播放视频。

[要使用 HLS 观看 Kinesis 视频流，请先使用 getHLS 网址创建直播会话。StreamingSession](#) 此操作将返回用于访问 HLS 会话的 URL（包含会话令牌）。您随后可以使用媒体播放器或独立应用程序中的 URL 来显示流。

Important

并非所有发送到 Kinesis Video Streams 的媒体都可以通过 HLS 播放。有关具体 [the section called "GetHLSStreamingSessionURL"](#) 的上传要求，请参阅。

主题

- [使用检索 HLS 直播会话网址 AWS CLI](#)
- [示例：在 HTML 中使用 HLS 和 JavaScript](#)
- [HLS 问题疑难解答](#)

使用检索 HLS 直播会话网址 AWS CLI

按照以下步骤使用为 Kinesis 视频流生成 HLS 直播会话网址。AWS CLI

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](#)。

检索 Kinesis 视频流的 HLS 网址端点。

1. 在终端中键入以下内容：

```
aws kinesisvideo get-data-endpoint \  
  --api-name GET_HLS_STREAMING_SESSION_URL \  
  --stream-name YourStreamName
```

你会收到如下所示的回复：

```
{  
  "DataEndpoint": "https://b-1234abcd.kinesisvideo.aws-region.amazonaws.com"  
}
```

2. 向返回的端点发出 HLS 直播会话 URL 请求。

Live

为了进行实时播放，HLS 媒体播放列表会根据最新媒体不断更新。当您在媒体播放器中播放此类会话时，用户界面通常会显示“实时”通知，没有用于在播放窗口中选择要显示的位置的滑块控件。

运行此命令时，请确保您正在将媒体上传到此流。

```
aws kinesis-video-archived-media get-hls-streaming-session-url \  
  --endpoint-url https://b-1234abcd.kinesisvideo.aws-region.amazonaws.com \  
  --stream-name YourStreamName
```

```
--stream-name YourStreamName \  
--playback-mode LIVE
```

Live replay

对于直播回放，从指定的开始时间开始播放。HLS 媒体播放列表还会在可用时不断更新最新媒体。会话将继续包含新摄取的媒体，直到会话到期或指定的结束时间（以先到者为准）。此模式非常有用，可以从检测到事件时开始播放，并继续直播截至会话创建时尚未收录的媒体。

确定开始时间戳。

在此示例中，我们使用以秒为单位的 Unix Epoch 时间。有关[时间戳](#)格式的更多信息，请参阅《AWS Command Line Interface 用户指南》中的“时间戳”部分。

有关转换工具，请参见 [UnixTime.org](https://unixtime.org)。

- 1708471800 等于 2024 年 2 月 20 日下午 3:30:00 GMT-08:00

在此示例中，我们没有指定结束时间戳，这意味着会话将继续包含新摄取的媒体，直到会话到期。

在LIVE_REPLAY播放模式和指定了[HLS 片段选择器](#)的情况下调用 GetHLSStreamingSessionURL API。

```
aws kinesis-video-archived-media get-hls-streaming-session-url \  
  --endpoint-url https://b-1234abcd.kinesisvideo.aws-region.amazonaws.com \  
  --stream-name YourStreamName \  
  --playback-mode LIVE_REPLAY \  
  --hls-fragment-selector \  
  "FragmentSelectorType=SERVER_TIMESTAMP,TimestampRange={StartTimestamp=1708471800}"
```

On-demand

对于按需播放，HLS 媒体播放列表包含由 HLS 片段选择器指定的媒体。在媒体播放器中播放此类会话时，用户界面通常会显示一个滑块控件，用于在播放窗口中选择要显示的位置。

要为直播的特定部分创建 URL，请先确定开始和结束时间戳。


在此示例中，我们使用以秒为单位的 Unix Epoch 时间。有关[时间戳](#)格式的更多信息，请参阅《AWS Command Line Interface 用户指南》中的“时间戳”部分。

有关转换工具，请参见 [UnixTime.org](https://unixtime.org)。

- 1708471800 等于 2024 年 2 月 20 日下午 3:30:00 GMT-08:00
- 1708471860 等于 2024 年 2 月 20 日下午 3:31:00 GMT-08:00

在 ON_DEMAND 播放模式和指定了 [HLS 片段选择器](#) 的情况下调用 GetHLSStreamingSessionURL API。


```
aws kinesis-video-archived-media get-hls-streaming-session-url \  
  --endpoint-url https://b-1234abcd.kinesisvideo.aws-region.amazonaws.com \  
  --stream-name YourStreamName \  
  --playback-mode ON_DEMAND \  
  --hls-fragment-selector \  
  
"FragmentSelectorType=SERVER_TIMESTAMP, TimestampRange={StartTimestamp=1708471800, EndTime
```

 Note

如文档中所述，时间戳必须在 24 小时之内。 [the section called “HLSTimestampRange”](#)

你会收到如下所示的回复：

```
{  
  "HLSStreamingSessionURL": "https://b-1234abcd.kinesisvideo.aws-  
region.amazonaws.com/hls/v1/getHLSMasterPlaylist.m3u8?SessionToken=CiAz...DkREGM~"  
}
```

 Important

请勿将此令牌共享或存储在未经授权的实体可以访问的地方。该令牌提供对直播内容的访问权限。使用与 AWS 凭证相同的措施来保护令牌。

您可以使用此网址和任何 HLS 播放器来观看 HLS 直播。

例如，使用 VLC 媒体播放器。

你也可以通过在 Apple Safari 或 Microsoft Edge 浏览器的位置栏中输入 HLS 直播会话网址来播放 HLS 直播。

示例：在 HTML 中使用 HLS 和 JavaScript

以下示例说明如何使用 AWS 适用于 JavaScript v2 的 SDK 检索 Kinesis 视频流的 HLS 直播会话并在网页中播放。该示例演示了如何在以下播放器中播放视频：

- [Video.js](#)
- [Google Shaka Player](#)
- [hls.js](#)

在中查看[完整的示例代码](#)和[托管网页](#) GitHub。

代码演练主题：

- [导入 JavaScript 适用于浏览器的 AWS SDK](#)
- [设置 Kinesis Video Streams 客户端](#)
- [检索 HLS 播放的端点](#)
- [设置 Kinesis Video Streams 存档媒体客户端](#)
- [检索 HLS 直播会话网址](#)
- [在网页上显示 HLS 直播](#)

导入 JavaScript 适用于浏览器的 AWS SDK

在网页中，添加以下脚本标签以将 JavaScript v2 版 AWS SDK 导入到项目中。

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/aws-sdk/2.490.0/aws-sdk.min.js"></script>
```

有关更多信息，请参阅 S [AWS DK](#) 中的 JavaScript 文档。

设置 Kinesis Video Streams 客户端

要使用 HLS 访问流媒体视频，请先创建和配置 Kinesis Video Streams 客户端。有关其他身份验证方法，[请参阅在 Web 浏览器中设置凭据](#)。

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

该应用程序将从 HTML 页面上的输入框检索必要值。

检索 HLS 播放的端点

使用 Kinesis Video Streams 客户端调用 [the section called “GetDataEndpoint” API](#) 来检索端点。

```
const getDataEndpointOptions = {
  StreamName: 'YourStreamName',
  APIName: 'GET_HLS_STREAMING_SESSION_URL'
};
const getDataEndpointResponse = await kinesisVideoClient
  .getDataEndpoint(getDataEndpointOptions)
  .promise();
const hlsDataEndpoint = getDataEndpointResponse.DataEndpoint;
```

此代码将端点存储在 hlsDataEndpoint 变量中。

设置 Kinesis Video Streams 存档媒体客户端

在 Kinesis Video Streams 存档媒体客户端的客户端配置中，指定您在上一步中获得的终端节点。

```
const archivedMediaClientConfig = {
  accessKeyId: 'YourAccessKey',
  secretAccessKey: 'YourSecretKey',
  region: 'us-west-2',
  endpoint: hlsDataEndpoint
}
const kinesisVideoArchivedMediaClient = new
  AWS.KinesisVideoArchivedMedia(archivedMediaClientConfig);
```

检索 HLS 直播会话网址

使用 Kinesis Video Streams 存档媒体客户端调用 [the section called “GetHLSStreamingSessionURL”](#) API 来检索 HLS 播放网址。

```
const getHLSStreamingSessionURLOptions = {
  StreamName: 'YourStreamName',
  PlaybackMode: 'LIVE'
};
const getHLSStreamingSessionURLResponse = await
  kinesisVideoArchivedMediaClient.getHLSStreamingSessionURL(getHLSStreamingSessionURLOptions);
const hlsUrl = getHLSStreamingSessionURLResponse.HLSStreamingSessionURL;
```

在网页上显示 HLS 直播

当您具有 HLS 流会话 URL 时，请将其提供到视频播放器。向视频播放器提供 URL 的方法是特定于所使用的播放器的。

Video.js

执行以下操作将 [Video.js](#) 及其 CSS 类导入到我们的浏览器脚本中：

```
<link rel="stylesheet" href="https://vjs.zencdn.net/6.6.3/video-js.css">
<script src="https://vjs.zencdn.net/6.6.3/video.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/videojs-contrib-hls/5.14.1/
videojs-contrib-hls.js"></script>
```

创建一个 video HTML 元素来显示视频：

```
<video id="videojs" class="player video-js vjs-default-skin" controls autoplay></
video>
```

将 HLS 网址设置为 HTML 视频元素的来源：

```
const playerElement = document.getElementById('videojs');
const player = videojs(playerElement);
player.src({
  src: hlsUrl,
  type: 'application/x-mpegURL'
});
player.play();
```

Shaka

执行以下操作将 [Google Shaka 播放器](#) 导入我们的浏览器脚本：

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/shaka-player/2.4.1/shaka-player.compiled.js"></script>
```

创建一个 video HTML 元素来显示视频：

```
<video id="shaka" class="player" controls autoplay></video>
```

创建一个指定视频元素的 Shaka 播放器并调用 load 方法：

```
const playerElement = document.getElementById('shaka');  
const player = new shaka.Player(playerElement);  
player.load(hlsUrl);
```

hls.js

执行以下操作将 [hls.js](#) 导入我们的浏览器脚本：

```
<script src="https://cdn.jsdelivr.net/npm/hls.js@latest"></script>
```

创建一个 video HTML 元素来显示视频：

```
<video id="hlsjs" class="player" controls autoplay></video>
```

创建一个 hls.js 播放器，给它一个 HLS 网址，然后叫它玩：

```
var playerElement = document.getElementById('hlsjs');  
var player = new Hls();  
player.loadSource(hlsUrl);  
player.attachMedia(playerElement);  
player.on(Hls.Events.MANIFEST_PARSED, function() {  
    video.play();  
});
```

HLS 问题疑难解答

本节介绍在 Kinesis Video Streams 中使用 HTTP 直播 (HLS) 时可能遇到的问题。

问题

- [检索 HLS 流会话 URL 成功，但在视频播放器中播放失败](#)
- [创建者与播放器之间的延迟过高](#)

检索 HLS 流会话 URL 成功，但在视频播放器中播放失败

当使用 `GetHLSStreamingSessionURL` 可以成功检索 HLS 流会话 URL，但将该 URL 提供给视频播放器后视频无法播放时，就会出现这种情况。

要排查这种情况，请尝试以下操作：

- 确定视频流是否在 Kinesis Video Streams 控制台中播放。考虑控制台显示的任何错误。
- 如果片段持续时间少于 1 秒，则增加到 1 秒。如果片段持续时间太短，则该服务可能会限制播放器，因为它过于频繁地请求视频片段。
- 验证每个 HLS 流会话 URL 是否在被唯一的用户使用。如果多个播放器正在使用单个 HLS 流会话 URL，服务可能会收到过多的请求并限制这些播放器。
- 确认您的播放器支持您为 HLS 直播会话指定的所有选项。为以下参数尝试其他值组合：
 - `ContainerFormat`
 - `PlaybackMode`
 - `FragmentSelectorType`
 - `DiscontinuityMode`
 - `MaxMediaPlaylistFragmentResults`

某些媒体播放器（如 HTML5 和手机播放器）通常仅支持 fMP4 容器格式的 HLS。其他媒体播放器（例如 Flash 和自定义播放器）可能只支持 MPEG TS 容器格式的 HLS。我们建议尝试使用该 `ContainerFormat` 参数以开始故障排除。

- 验证每个片段是否具有一致的轨道数。验证流中的片段在同时具有音频和视频轨道与只有视频轨道之间没有变化。另外，请确认每个轨道的片段之间的编码器设置（分辨率和帧速率）没有变化。

创建者与播放器之间的延迟过高

当捕获视频与在视频播放器中播放视频之间的延迟过高时，就会出现这种情况。

视频通过 HLS 逐个片段地播放。因此，延迟不能小于片段持续时间。延迟还包含缓冲和传输数据所需的时间。如果您的解决方案需要的延迟不到一秒，请考虑改用 `GetMedia API`。

您可以调整以下参数来减少总体延迟，但调整这些参数也可能降低视频质量或提高再缓冲率。

- 片段持续时间-片段持续时间是视频流中各分段之间的视频量，由视频编码器生成的关键帧频率控制。推荐值为 1 秒。具有较短的片段持续时间意味着在将视频数据传输到服务之前等待片段完成的时间会更少。较短的片段还有助于提高服务的处理速度。但是，如果片段持续时间过短，则可能增加播放器耗尽内容且必须停止和缓冲内容的可能性。如果片段持续时间短于 500 毫秒，创建者可能会创建过多请求，从而导致服务对其进行限制。
- 比特率 — 比特率较低的视频流读取、写入和传输所需的时间更少。但是，比特率较低的视频流通常具有较低的视频质量。
- 媒体播放列表中的片段数 — 对延迟敏感的播放器只能加载媒体播放列表中的最新片段。大多数玩家改为从最早的片段开始。通过减少播放列表中的片段数量，可以缩短先前片段和新片段之间的时间间隔。如果播放列表大小较小，则在向播放列表添加新片段时出现延迟，或者如果玩家获取更新的播放列表出现延迟，则在播放过程中可能会跳过片段。我们建议使用 3—5 个片段，并使用配置为仅加载播放列表中最新片段的播放器。
- 播放器缓冲区大小 — 大多数视频播放器都有可配置的最小缓冲持续时间，默认值通常为 10 秒。要最大程度地减少延迟，您可以将此值设置为 0 秒。但是，这样做意味着如果有任何延迟生成片段，则玩家会拒绝缓冲，因为玩家没有缓冲区来吸收延迟。
- 播放器“catch up” — 如果缓冲区已满，视频播放器通常不会自动将播放捕捉到视频缓冲区前面，比如延迟的片段导致片段积压播放。自定义播放器可通过删除帧或提高播放速度（例如，提高到 1.1 倍）以赶到缓冲区的前面来避免这一点。这会导致播放器赶上时出现震荡或加速，并且由于缓冲区大小被保持在较小水平，重新缓冲可能会更频繁。

使用 MPEG-DASH 播放视频

[要使用 MPEG-DASH 观看 Kinesis 视频流，请先使用 GetDash 网址创建直播会话。](#)

[StreamingSession](#) 此操作将返回用于访问 MPEG-DASH 会话的 URL（包含会话令牌）。您随后可以使用媒体播放器或独立应用程序中的 URL 来显示流。

Amazon Kinesis 视频流在通过 MPEG-DASH 提供视频时需要满足以下要求：

- 有关流式视频播放曲目的要求，请参阅[the section called “GetDash 网址 StreamingSession”](#)。
- 数据保留必须大于 0。
- 对于 H.264 格式的高级视频编码 (AVC) 和 H.265 格式的 HEVC，各个片段的视频轨道必须包含编解码器专用数据。有关更多信息，请参阅 [MPEG-4 规范 ISO/IEC 14496-15](#)。有关使流数据适应给定格式的信息，请参阅 [NAL 适应标志](#)。
- 各个片段的音频轨道（如果存在）必须包含 AAC 格式 ([AAC specification ISO/IEC 13818-7](#)) 或 [MS Wave 格式](#) 的编解码器专用数据。

示例：在 HTML 中使用 MPEG-DASH 和 JavaScript

以下示例说明如何检索 Kinesis 视频流的 MPEG-DASH 直播会话并在网页中播放。该示例演示了如何在以下播放器中播放视频：

- [Google Shaka Player](#)
- [dash.js](#)

主题

- [设置 Kinesis Video Streams 客户端以便播放 MPEG-DASH](#)
- [检索 Kinesis Video Streams 存档内容端点以便播放 MPEG-DASH](#)
- [检索 MPEG-DASH 流会话 URL](#)
- [通过 MPEG-DASH 播放显示流视频](#)
- [完整示例](#)

设置 Kinesis Video Streams 客户端以便播放 MPEG-DASH

要使用 MPEG-DASH 访问流媒体视频，请先创建并配置 Kinesis Video Streams 客户端（用于检索服务端点）和存档媒体客户端（用于检索 MPEG-DASH 直播会话）。该应用程序将从 HTML 页面上的输入框检索必要值。

```
var streamName = $('#streamName').val();

// Step 1: Configure SDK Clients
var options = {
  accessKeyId: $('#accessKeyId').val(),
  secretAccessKey: $('#secretAccessKey').val(),
  sessionToken: $('#sessionToken').val() || undefined,
  region: $('#region').val(),
  endpoint: $('#endpoint').val() || undefined
}
var kinesisVideo = new AWS.KinesisVideo(options);
var kinesisVideoArchivedContent = new AWS.KinesisVideoArchivedMedia(options);
```


检索 Kinesis Video Streams 存档内容端点以便播放 MPEG-DASH

启动客户端后，检索 Kinesis Video Streams 存档内容端点，以便您可以按如下方式检索 MPEG-DASH 直播会话网址：

```
// Step 2: Get a data endpoint for the stream
console.log('Fetching data endpoint');
kinesisVideo.getDataEndpoint({
  StreamName: streamName,
  APIName: "GET_DASH_STREAMING_SESSION_URL"
}, function(err, response) {
  if (err) { return console.error(err); }
  console.log('Data endpoint: ' + response.DataEndpoint);
  kinesisVideoArchivedContent.endpoint = new AWS.Endpoint(response.DataEndpoint);
```

检索 MPEG-DASH 流会话 URL

当您拥有存档内容端点后，请调用 [GetDash StreamingSession 网址](#) API 来检索 MPEG-DASH 直播会话网址，如下所示：

```
// Step 3: Get a Streaming Session URL
var consoleInfo = 'Fetching ' + protocol + ' Streaming Session URL';
console.log(consoleInfo);

if (protocol === 'DASH') {
  kinesisVideoArchivedContent.getDASHStreamingSessionURL({
    StreamName: streamName,
    PlaybackMode: $('#playbackMode').val(),
    DASHFragmentSelector: {
      FragmentSelectorType: $('#fragmentSelectorType').val(),
      TimestampRange: $('#playbackMode').val() === "LIVE" ? undefined : {
        StartTimestamp: new Date($('#startTimestamp').val()),
        EndTimestamp: new Date($('#endTimestamp').val())
      }
    },
    DisplayFragmentTimestamp: $('#displayFragmentTimestamp').val(),
    DisplayFragmentNumber: $('#displayFragmentNumber').val(),
    MaxManifestFragmentResults: parseInt($('#maxResults').val()),
    Expires: parseInt($('#expires').val())
  }, function(err, response) {
    if (err) { return console.error(err); }
```

```
console.log('DASH Streaming Session URL: ' + response.DASHStreamingSessionURL);
```

通过 MPEG-DASH 播放显示流视频

当您具有 MPEG-DASH 流会话 URL 时，请将其提供到视频播放器。向视频播放器提供 URL 的方法特定于您使用的播放器。

以下代码示例演示如何将流会话 URL 提供给 [Google Shaka](#) 播放器：

```
// Step 4: Give the URL to the video player.

//Shaka Player elements
<video id="shaka" class="player" controls autoplay></video>
<script src="https://cdnjs.cloudflare.com/ajax/libs/shaka-player/2.4.1/shaka-
player.compiled.js">
</script>
...

var playerName = $('#player').val();

if (playerName === 'Shaka Player') {
  var playerElement = $('#shaka');
  playerElement.show();

  var player = new shaka.Player(playerElement[0]);
  console.log('Created Shaka Player');

  player.load(response.DASHStreamingSessionURL).then(function() {
    console.log('Starting playback');
  });
  console.log('Set player source');
}
```

以下代码示例说明如何将流会话 URL 提供给 [dash.js](#) 播放器：

```
<!-- dash.js Player elements -->
<video id="dashjs" class="player" controls autoplay=""></video>
<script src="https://cdn.dashjs.org/latest/dash.all.min.js"></script>
```

```
...

var playerElement = $('#dashjs');
playerElement.show();

var player = dashjs.MediaPlayer().create();
console.log('Created DASH.js Player');

player.initialize(document.querySelector('#dashjs'), response.DASHStreamingSessionURL,
  true);
console.log('Starting playback');
console.log('Set player source');
}
```

完整示例

您可以在上[下载或查看已完成的示例代码](#) GitHub。

在 Kinesis Video Streams 中使用流式传输元数据

您可以使用 Amazon Kinesis Video Streams Producer SDK 在 Kinesis 视频流中嵌入单个片段级别的元数据。Kinesis Video Streams 中的元数据是一个可变的键值对。您可以使用它来描述片段的内容，嵌入必须与实际片段一起传输的相关传感器读数，或者满足其他自定义需求。元数据作为 [the section called “GetMedia”](#) 或 [the section called “GetMediaForFragmentList”](#) API 操作的一部分提供。在直播的整个保留期内，它与片段一起存储。您的消费应用程序可以使用基于元数据进行读取、处理和响应 [Kinesis 视频直播解析器库](#)。

可通过以下两种模式在数据流片段中嵌入元数据：

- 非持续 — 您可以根据已出现的业务特定标准，一次性或临时将元数据附加到流中的片段。例如，智能相机可以检测动作，并在将片段发送到其 Kinesis 视频流之前，向包含动作的相应片段添加元数据。您可以采用以下格式将元数据应用于片段：Motion = true。
- 持续-您可以根据持续需要将元数据附加到流中连续的片段。例如，智能相机将与其发送的所有片段关联的当前纬度和经度坐标发送到其 Kinesis 视频流。您可以采用以下格式将元数据应用于所有片段：Lat = 47.608013N ， Long = -122.335167W。

您可以根据应用程序的需求，将这两种模式的元数据同时附加到同一个片段中。嵌入的元数据可能包括检测到的对象、跟踪的活动、GPS 坐标或者要与数据流中的片段关联的任何其他自定义数据。元数据编码为键值字符串对。

主题

- [向 Kinesis 视频流添加元数据](#)
- [使用 Kinesis 视频流中嵌入的元数据](#)
- [流式传输元数据限制](#)

向 Kinesis 视频流添加元数据

您添加到 Kinesis 视频流中的元数据将建模为 MKV 标签，这些标签以键值对的形式实现。

元数据可以是临时的，如用于标记数据流内的事件，也可以是持久的，如用于识别发生给定事件的片段。永久性元数据项将保留，并应用于每个连续的片段，直到它被取消。

Note

使用[创建者库](#)添加的元数据项目不同于使用 [the section called “TagStream”](#)、[the section called “UntagStream”](#) 和 [the section called “ListTagsForStream”](#) 实现的数据流级别的标记 API。

流式传输元数据 API

您可以使用创建者开发工具包中的以下操作实现流式处理元数据。

PIC

```
PUBLIC_API STATUS putKinesisVideoFragmentMetadata(STREAM_HANDLE streamHandle,
    PCHAR name,
    PCHAR value,
    BOOL persistent);
```

C++ 创建者开发工具包

```
/**
 * Appends a "tag" or metadata - a key/value string pair into the stream.
 */
bool putFragmentMetadata(const std::string& name, const std::string& value, bool
    persistent = true);
```

Java 创建者开发工具包

您可以使用 Java Producer SDK 通过以下方式向添加元数据 `MediaSourceSink.onCodecPrivateData` :

```
void onFragmentMetadata(final @NonNull String metadataName, final @NonNull String
    metadataValue, final boolean persistent)
    throws KinesisVideoException;
```

永久和非持久元数据

对于非持久元数据，您可以添加多个具有相同名称的元数据项目。创建者开发工具包一直收集元数据队列中的元数据项目，直到将这些项目附加到下一个片段之前。在将元数据项目应用到数据流时，将清除元数据队列。要重复元数据，请再次调用 `putKinesisVideoFragmentMetadata` 或 `putFragmentMetadata`。

对于持久元数据，创建者开发工具包将按照处理非持久元数据的相同方式收集元数据队列中的元数据项目。但是，当元数据项被添加到下一个片段之前时，它们不会从队列中删除。

在将 `persistent` 设置为 `true` 时调用 `putKinesisVideoFragmentMetadata` 或 `putFragmentMetadata` 会出现以下行为：

- 调用 API 会将元数据项目置于队列中。当项目位于队列中时，会将元数据作为 MKV 标签添加到每个片段中。
- 使用与以前添加的元数据项目相同的名称和不同的值调用 API 将覆盖该项目。
- 使用空值调用 API 将从元数据队列中删除（取消）元数据项目。

使用 Kinesis 视频流中嵌入的元数据

要使用 Kinesis 视频流中的元数据，请使用以下实现：`MkvTagProcessor`

```
public interface MkvTagProcessor {
    default void process(MkvTag mkvTag, Optional<FragmentMetadata>
        currentFragmentMetadata) {
        throw new NotImplementedException("Default
        FragmentMetadataVisitor.MkvTagProcessor");
    }
    default void clear() {
```

```
        throw new NotImplementedException("Default
FragmentMetadataVisitor.MkvTagProcessor");
    }
}
}
```

在 [Kinesis 视频直播解析器库](#) 的 [FragmentMetadataVisitor](#) 类中可找到此接口。

`FragmentMetadataVisitor` 类包含 `MkvTagProcessor` 的实现：

```
public static final class BasicMkvTagProcessor implements
FragmentMetadataVisitor.MkvTagProcessor {
    @Getter
    private List<MkvTag> tags = new ArrayList<>();

    @Override
    public void process(MkvTag mkvTag, Optional<FragmentMetadata>
currentFragmentMetadata) {
        tags.add(mkvTag);
    }

    @Override
    public void clear() {
        tags.clear();
    }
}
```

`KinesisVideoRendererExample` 类包含演示如何使用 `BasicMkvTagProcessor` 的示例。以下示例将 `BasicMkvTagProcessor` 添加到应用程序的 `MediaProcessingArguments` 中：

```
if (renderFragmentMetadata) {
    getMediaProcessingArguments =
KinesisVideoRendererExample.GetMediaProcessingArguments.create(
    Optional.of(new FragmentMetadataVisitor.BasicMkvTagProcessor()));
}
```

在片段元数据送达时调用 `BasicMkvTagProcessor.process` 方法。您可以使用 `GetTags` 检索累积的元数据。要检索单个元数据项，请先调用 `clear` 以清除收集的元数据，然后再次检索元数据项目。

流式传输元数据限制

[the section called “片段元数据配额”](#) 有关向 Kinesis 视频流添加直播元数据所适用的限制的更多信息，[请参阅](#)

Kinesis Video Streams 数据模型

[创建者库](#) 和 [视频流解析器库](#) 以某种格式发送和接收视频数据，此格式支持随视频数据一起嵌入信息。此格式基于 Matroska (MKV) 规范。

[MKV 格式](#) 是适用于媒体数据的开放规范。亚马逊 Kinesis Video Streams 开发者指南中的所有库和代码示例都以 MKV 格式发送或接收数据。

[Kinesis 视频直播制作人库](#) 使用 `StreamDefinition` 和 `Frame` 类型生成 MKV 流标头、帧标题和帧数据。

有关完整 MKV 规范的信息，请参阅 [Matroska 规范](#)。

以下几节描述了由 [C++ 创建者库](#) 生成的 MKV 格式数据的组成部分。

主题

- [流标头元素](#)
- [流式传输曲目数据](#)
- [帧头元素](#)
- [MKV 帧数据](#)

流标头元素

`StreamDefinition` 使用以下 MKV 标头元素 (在 `StreamDefinition.h` 中定义)。

元素	描述	典型值
<code>stream_name</code>	与 Kinesis 视频流的名称相对应。	my-stream
<code>retention_period</code>	Kinesis Video Streams 保留直播数据的持续时间 (以小时为单位)。0 为不保留数据的直播指定。	24
<code>tags</code>	用户数据的键-值集合。此数据显示在 AWS Management Console 中，可由客户端应用程序	

元素	描述	典型值
	序读取以筛选或获取有关流的信息。	
kms_key_id	如果存在，则使用用户定义的 AWS KMS 密钥来加密流中的数据。如果不存在，则通过 Kinesis 提供的密钥 () 对数据进行加密。aws/kinesis-video	01234567-89ab-cdef-0123-456789ab
streaming_type	目前，唯一有效的流式处理类型是 STREAMING_TYPE_REALTIME 。	STREAMING_TYPE_REALTIME
content_type	用户定义的内容类型。对于要在控制台中播放的流视频数据，内容类型必须为 video/h264 。	video/h264
max_latency	当前未使用此值，应将其设置为 0。	0
fragment_duration	这是对片段应持续时间的估计，用于优化。实际的片段持续时间由流数据决定。	2
timecode_scale	表示帧时间戳使用的比例。默认为 1 毫秒。指定 0 还将分配 1 毫秒的默认值。此值可以介于 100 纳秒到 1 秒之间。 有关更多信息，请参阅 Matroska 文档 TimecodeScale 中的。	
key_frame_fragmentation	如果为 true，则流在收到关键帧时将启动一个新集群。	true

元素	描述	典型值
frame_timecodes	如果是true，Kinesis Video Streams 将使用接收到的帧的演示时间戳 (pts) 和解码时间戳 (dts) 值。如果是false，Kinesis Video Streams 将在收到帧时使用系统生成的时间值对其进行标记。	true
absolute_fragment_time	如果为 true，则集群时间码将解释为使用绝对时间 (例如，来自创建者的系统时钟)。如果为 false，则集群时间码将解释为相对于流的开始时间。	true
fragment_acks	如果是true，则在 Kinesis Video Streams 收到数据时发送确认 (ACK)。可使用 KinesisVideoStream FragmentAck 或 KinesisVideoStream ParseFragmentAck 回调接收确认。	true
restart_on_error	指示在发生流错误后是否应继续传输流。	true
nal_adaptation_flags	指示内容中是否有 NAL (网络抽象层) 改编或编解码器私有数据。有效标记包括 NAL_ADAPTATION_ANNEXB_NALS 和 NAL_ADAPTATION_ANNEXB_CPD_NALS 。	NAL_ADAPTATION_ANNEXB_NALS

元素	描述	典型值
frame_rate	内容帧速率的估计值。此值用于优化；实际帧速率由传入数据的速率决定。指定 0 将分配默认值 24。	24
avg_bandwidth_bps	内容带宽的估计值，以 Mbps 为单位。此值用于优化；实际速率由传入数据的带宽决定。例如，对于按 25 FPS 运行的 720p 分辨率视频流，您可预计平均带宽为 5 Mbps。	5
buffer_duration	要在创建者中缓冲的内容的持续时间。如果网络延迟较低，则可以降低此值。如果网络延迟很高，则增加此值可以防止帧在发送之前被丢弃，因为分配无法将帧放入较小的缓冲区。	
replay_duration	如果连接中断，视频数据流“倒带”的时间量。如果不考虑因连接丢失而导致的帧丢失，则此值可以为零。如果使用该应用程序可以移除冗余帧，则可以增加该值。此值应小于缓冲持续时间，否则将使用缓冲持续时间。	
connection_staleness	在未收到数据时保持连接的持续时间。	
codec_id	内容使用的编解码器。有关更多信息，请参阅 Matroska 规范中的 CodecID 。	V_MPEG2

元素	描述	典型值
track_name	用户定义的音轨名称。	my_track
codecPrivateData	编码器提供的数据，用于对许多下游使用者所需的帧数据（例如，以像素表示的帧宽度和高度）进行解码。在 C++ 创建者库 中，MkvStatic s.cpp 中的 gMkvTrack VideoBits 数组包含帧的像素宽度和高度。	
codecPrivateData大小	codecPrivateData 参数中数据的大小。	
track_type	流的轨道的类型。	MKV_TRACK_INFO_TYPE_AUDIO 或 MKV_TRACK_INFO_TYPE_VIDEO
segment_uuid	用户定义的片段 uuid (16 字节)。	
default_track_id	轨道的唯一非零编号。	1

流式传输曲目数据

StreamDefinition 使用以下 MKV 轨道元素 (在 StreamDefinition.h 中定义)。

元素	描述	典型值
track_name	用户定义的轨道名称。例如，“audio”代表音轨。	audio
codec_id	轨道的编解码器 ID。例如，“A_AAC”代表音轨。	A_AAC

元素	描述	典型值
cpd	编码器提供的数据用于对帧数据进行解码。帧数据可以包括帧的像素宽度和高度，许多下游使用者都需要帧数据。在 C++ 制作器库 中，MkvStatics.cpp 中的 gMkvTrackVideoBits 数组包括帧的像素宽度和高度。	
cpd_size	codecPrivateData 参数中数据的大小。	
track_type	轨道的类型。例如，您可以对音频使用 MKV_TRACK_INFO_TYPE_AUDIO 的枚举值。	MKV_TRACK_INFO_TYPE_AUDIO

帧头元素

Frame (在 mkvgen/Include.h 中的 KinesisVideoPic 程序包中定义) 将使用以下 MKV 标头：

- Frame Index：一个单调递增的值。
- Flags：帧的类型。有效值包括：
 - FRAME_FLAGS_NONE
 - FRAME_FLAG_KEY_FRAME：如果在流上设置 key_frame_fragmentation，关键帧将启动新的片段。
 - FRAME_FLAG_DISCARDABLE_FRAME：告知解码器可在解码速度较慢时放弃此帧。
 - FRAME_FLAG_INVISIBLE_FRAME：此数据块的持续时间为 0。
- 解码时间戳：解码此帧的时间戳。如果之前的帧依赖于此帧进行解码，则此时间戳可能早于前一帧的时间戳。该值相对于片段的开头。
- 演示时间戳：显示此帧的时间戳。该值相对于片段的开头。
- Duration：帧的播放持续时间。
- Size：帧数据的大小 (以字节为单位)

MKV 帧数据

`frame.frameData` 中的数据可能仅包含帧的媒体数据，或可能包含进一步嵌套的标头信息，具体取决于使用的编码架构。要在中显示AWS Management Console，必须使用 [H.264](#) 编解码器对数据进行编码，但是 Kinesis Video Streams 可以接收任何格式的时间序列化数据流。

亚马逊 Kinesis Video Streams 入门

本节介绍如何在 Amazon Kinesis Video Streams 中执行以下任务：

- 如果您尚未设置管理员 AWS 账户 并创建管理员（如果尚未这样做）。
- 创建 Kinesis 视频流。
- 从您的摄像机向 Kinesis 视频流发送数据，并在控制台中查看媒体。

如果你不熟悉 Amazon Kinesis Video Streams，我们建议你先[Kinesis Video Streams：工作原理](#)阅读。

Note

遵循入门示例不会对您 AWS 账户产生任何费用。有关您所在地区的数据成本，请参阅[亚马逊 Kinesis Video Streams](#) 定价。

主题

- [设置了账户](#)
- [创建 Kinesis 视频流](#)
- [向 Amazon Kinesis 视频流发送数据](#)
- [使用媒体数据](#)

设置了账户

在首次使用 Amazon Kinesis Video Streams 之前，请完成以下任务。

主题

- [注册获取 AWS 账户](#)
- [创建管理用户](#)
- [创建密 AWS 账户 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。

报名参加 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，请 [为管理用户分配管理访问权限](#)，并且只使用根用户执行 [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。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。

创建管理用户

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

创建管理用户

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 Identity Center 中，为管理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](#)。

作为管理用户登录

- 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](#)。

创建密 AWS 账户 钥

你需要 AWS 账户 密钥才能以编程方式访问亚马逊 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/](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”下选择“显示”。记录秘密访问密钥。

创建 Kinesis 视频流

本节介绍如何创建 Kinesis 视频流。

本节包含以下过程：

- [the section called “使用控制台创建视频流”](#)
- [the section called “使用创建视频流 AWS CLI”](#)

使用控制台创建视频流

1. 打开控制台，网址为<https://console.aws.amazon.com/kinesisvideo/home>。
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。
4. 选择 Create video stream (创建视频流)。
5. 亚马逊 Kinesis Video Streams 创建直播后，请查看页面YourStreamName上的详细信息。

使用创建视频流 AWS CLI

1. 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。
2. 在 AWS CLI中运行以下 Create-Stream 命令：

```
aws kinesismvideo create-stream --stream-name "YourStreamName" --data-retention-in-hours 24
```

该响应应该类似于以下内容：

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

向 Amazon Kinesis 视频流发送数据

本节介绍如何将媒体数据从摄像机发送到您在上一节中创建的 Kinesis 视频流。本节使用[C++ 创建者库](#)作为[GStreamer](#)插件。

要从各种操作系统上的各种设备发送媒体，本教程使用了 Kinesis Video Streams C++ 制作人库和 gStreamer，后者是一个开源媒体框架，用于标准化对摄像机和其他媒体源的访问。

主题

- [构建 SDK 和示例](#)
- [运行示例将媒体上传到 Kinesis Video Streams](#)
- [查看确认对象](#)

构建 SDK 和示例

您可以在计算机上或中构建 SDK 和示例 AWS Cloud9。请按照以下相应步骤操作。

Build on your computer

使用[自述文件](#)中的说明来构建生产者库和示例应用程序。

这包括：

- 安装依赖项
- 克隆存储库
- 使用 CMake 生成生成文件
- 使用 `make` 构建二进制文件

Build in AWS Cloud9

请按照以下步骤上传到中的 Kinesis Video Stream AWS Cloud9 s。您无需将任何内容下载到您的计算机上。

1. 在里面 AWS Management Console ，打开[AWS Cloud9](#)。

选择“创建环境”。

2. 在创建环境屏幕上，完成以下操作：

- 名称-键入新环境的名称。
- 平台——选择 Ubuntu Server 22.04 LTS。

您可以将其他字段保留为默认选项。

3. 创建环境后，在 Cloud9 IDE 列中选择“打开”。

在屏幕的中间下方区域，你会看到 `Admin:~/environment $`。这是 AWS Cloud9（亚马逊 EC2）终端。

Note

如果您不小心关闭了终端，请选择“窗口”，“新建终端”。

在终端中运行以下命令将音量更改为 20 GiB。

- a. 下载脚本。

```
wget https://awsj-iot-handson.s3-ap-northeast-1.amazonaws.com/kvs-workshop/resize_volume.sh
```

- b. 授予脚本执行权限。

```
chmod +x resize_volume.sh
```

- c. 运行脚本。

```
./resize_volume.sh
```

4. 获取有关您可以通过高级打包工具 (APT) 安装或更新的所有软件的最新信息。

此命令不会更新软件本身，但可以确保您的系统知道最新的可用版本。

```
sudo apt-get update
```

5. 安装 C++ 制作器 SDK 依赖项。

```
sudo apt-get install -y cmake m4 git build-essential pkg-config libssl-dev  
libcurl4-openssl-dev \  
liblog4cplus-dev libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev \  
gstreamer1.0-plugins-base-apps gstreamer1.0-plugins-bad gstreamer1.0-plugins-  
good \  
gstreamer1.0-plugins-ugly gstreamer1.0-tools
```

6. 使用 git 克隆 C++ 制作器开发工具包。

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

7. 准备一个构建目录。

```
cd amazon-kinesis-video-streams-producer-sdk-cpp  
mkdir build  
cd build
```

8. 使用 CMake 生成生成文件。

```
cmake .. -DBUILD_GSTREAMER_PLUGIN=TRUE -DBUILD_DEPENDENCIES=OFF
```

预期输出的结尾如下所示：

```
-- Build files have been written to: /home/ubuntu/environment/amazon-kinesis-  
video-streams-producer-sdk-cpp/build
```

9. 使用 make 编译 SDK 和示例应用程序，以及构建最终的可执行文件。

```
make
```

预期输出的结尾如下所示：

```
[100%] Linking CXX executable kvs_gstreamer_file_uploader_sample  
[100%] Built target kvs_gstreamer_file_uploader_sample
```

10. 确认示例文件已构建。列出当前目录中的文件：

```
ls
```

确认是否存在以下文件：

- kvs_gstreamer_sample
- libgstkvssink.so

运行示例将媒体上传到 Kinesis Video Streams

示例应用程序不支持 IMDS 凭证。在您的终端中，导出您的 IAM 用户或角色以及您的直播所在区域的 AWS 证书。

```
export AWS_ACCESS_KEY_ID=YourAccessKey  
export AWS_SECRET_KEY=YourSecretKey  
export AWS_DEFAULT_REGION=us-west-2
```

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
```

.mp4 files

下载一个.mp4 视频样本上传到 Kinesis Video Streams。

```
wget https://awsj-iot-handson.s3-ap-northeast-1.amazonaws.com/kvs-workshop/sample.mp4
```

使用您之前创建的直播的名称运行以下命令。如果您尚未创建直播，请参阅[the section called “创建 Kinesis 视频流”](#)。

```
./kvs_gstreamer_sample YourStreamName ./sample.mp4
```

Sample video from GStreamer

使用以下命令通过 gStreamer 生成视频。

告诉 gStreamer 在哪里可以找到 kvssink gStreamer 插件。在您的生成目录中，指定包含该libgstkvssink.so文件的文件夹的路径。

在您的构建目录中，运行以下命令：

```
export GST_PLUGIN_PATH=`pwd`
```

这个 gStreamer 管道生成具有标准测试模式的实时测试视频流，该视频流以每秒 10 帧的速度运行，分辨率为 640x480 像素。添加了一个显示当前系统时间和日期的叠加层。然后将视频编码为 H.264 格式，并且最多每 10 帧生成一次关键帧，从而使片段持续时间（也称为一组图片 (GoP) 大小）为 1 秒。kvssink 获取 H.264 编码的视频流，将其打包为 Matroska (MKV) 容器格式，然后将其上传到你的 Kinesis 视频流中。

运行以下命令：

```
gst-launch-1.0 -v videotestsrc is-live=true \  
! video/x-raw,framerate=10/1,width=640,height=480 \  
! clockoverlay time-format="%a %B %d, %Y %I:%M:%S %p" \  
! x264enc bframes=0 key-int-max=10 \  
! h264parse \  
! kvssink stream-name="YourStreamName"
```

要停止 GStreamer 管道，请选择终端窗口并按 CTRL+C。

Note

有关使用 gStreamer 插件流式传输来自摄像机的 RTSP 流或 USB 摄像头的视频的更多信息，请参阅 [示例：Kinesis Video Streams Producer SDK gStreamer 插件](#)

查看确认对象

在上传过程中，Kinesis Video Streams 会将确认对象发送回执行上传的客户端。您应该在命令输出中看到这些内容。一个例子如下所示：

```
{"EventType":"PERSISTED","FragmentTimecode":1711124585823,"FragmentNumber":"1234567890123456789"}
```

如果确认 EventType 是 PERSISTED，则表示 Kinesis Video Streams 已对这块媒体进行了持久存储和加密，以供检索、分析和长期存储。

有关致谢的更多信息，请参阅 [the section called “PutMedia”](#)。

使用媒体数据

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。有关使用 HLS 创建使用媒体数据的应用程序的信息，请参阅 [the section called “视频播放”](#)。

亚马逊 Kinesis Video Streams Edge Agent

Amazon Kinesis Video Streams 为连接客户场所的 IP 摄像机提供了一种高效、经济实惠的方式。借助 Amazon Kinesis Video Streams Edge Agent，您可以在本地录制和存储来自摄像机的视频，并按照客户定义的时间表将视频流式传输到云端，以进行长期存储、播放和分析处理。

Note

要访问亚马逊 Kinesis Video Streams Edge Agent，请填写[这份简短](#)的表格。

您可以下载 Amazon Kinesis Video Streams 边缘代理并将其部署到本地边缘计算设备上。您还可以轻松地将它们部署到在 Amazon EC2 实例上运行的 Docker 容器中。部署后，您可以使用 Amazon Kinesis Video Streams Video Streams API 来更新视频录制和云上传配置。该功能适用于任何可以通过 RTSP 协议进行直播的 IP 摄像机。它不需要在摄像机上部署任何额外的固件。

我们为亚马逊 Kinesis Video Streams Video Streams Edge Agent 提供以下安装选项：

- 作为AWS IoT Greengrass V2组件：您可以将 Amazon Kinesis Video Streams Edge Agent AWS IoT Greengrass 作为组件安装在AWS IoT Greengrass任何经过认证的设备上。有关 AWS IoT Greengrass 的更多信息，请参阅 [AWS IoT Greengrass Version 2 开发人员指南](#)。
- 开启AWS Snowball Edge：您可以在 Snowball Edge 设备上运行亚马逊 Kinesis Video Streams Edge Agent。要了解更多信息，请参阅 [AWS SnowballEdge 开发者指南](#)。
- 在本机AWS IoT部署中：您可以在任何计算实例上以原生方式安装 Amazon Kinesis Video Streams 边缘代理。Edge SDK [AWS IoT Core](#)用于通过管理边缘[the section called “Amazon Kinesis Video Streams”](#)。

要开始使用 Amazon Kinesis Video Streams Edge Agent，请继续执行以下相应步骤。

主题

- [亚马逊 Kinesis Video Streams 边缘代理 API 操作](#)
- [监控 Amazon Kinesis Video Streams 边缘代理](#)
- [在非AWS IoT Greengrass模式下运行 Amazon Kinesis Video Streams Edge Agent](#)
- [将 Amazon Kinesis Video Streams 边缘代理部署到 AWS IoT Greengrass](#)
- [亚马逊 Kinesis Video Streams Edge Agent 常见问题解答](#)

亚马逊 Kinesis Video Streams 边缘代理 API 操作

使用以下 API 操作配置 Amazon Kinesis Video Streams 边缘代理：

- [the section called “StartEdgeConfigurationUpdate”](#)
- [the section called “DescribeEdgeConfiguration”](#)
- [the section called “DeleteEdgeConfiguration”](#)
- [the section called “ListEdgeAgentConfigurations”](#)

监控 Amazon Kinesis Video Streams 边缘代理

要监控你的 Amazon Kinesis Video Streams Edge Agent，请参阅[the section called “使用以下方式监控 Amazon Kinesis Video Streams Edge Agent CloudWatch”](#)。

在非AWS IoT Greengrass模式下运行 Amazon Kinesis Video Streams Edge Agent

按照以下步骤运行AWS IoT带有 MQTT 的 Amazon Kinesis Video Streams Edge Agent 作为独立部署。

主题

- [步骤 1：在设备上安装必要的依赖项](#)
- [第 2 步：为 IP 摄像机 RTSP 网址创建 Amazon Kinesis Video Streams AWS Secrets Manager 和资源](#)
- [步骤 3：创建 IAM 权限策略](#)
- [步骤 4：创建 IAM 角色](#)
- [步骤 5：创建AWS IoT角色别名](#)
- [步骤 6：创建AWS IoT策略](#)
- [步骤 7：创建AWS IoT事物并获取凭证 AWS IoT Core](#)
- [第 8 步：构建并运行亚马逊 Kinesis Video Streams Edge Agent](#)
- [步骤 9：（可选）在设备上安装 CloudWatch 代理](#)
- [第 10 步：（可选）以原生进程身份运行 Amazon Kinesis Video Streams Edge Agent](#)

步骤 1：在设备上安装必要的依赖项

Note

有关支持的操作系统的列表，请参阅[the section called “亚马逊 Kinesis Video Streams Video Streams Edge Agent 支持哪些操作系统？”](#)。

在设备上安装依赖关系

1. 要运行 Amazon Kinesis Video Streams Edge Agent，请在您的设备上安装以下相应的库：

Ubuntu

类型：

```
wget -O- https://apt.corretto.aws/corretto.key | sudo apt-key add -  
sudo add-apt-repository 'deb https://apt.corretto.aws stable main'  
sudo apt-get update  
  
sudo apt-get install -y gcc libssl-dev libcurl4-openssl-dev liblog4cplus-dev \  
libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev \  
gstreamer1.0-plugins-base-apps gstreamer1.0-plugins-bad \  
gstreamer1.0-plugins-good gstreamer1.0-tools \  
unzip java-11-amazon-corretto-jdk maven
```

Amazon Linux 2

类型：

```
sudo yum update -y && sudo yum upgrade -y && sudo yum clean all -y  
sudo yum install -y gcc-c++ openssl-devel libcurl-devel gstreamer1* wget \  
java-11-amazon-corretto tar
```

log4cplus-2.1.0从源代码安装。

```
wget https://github.com/log4cplus/log4cplus/releases/download/REL_2_1_0/  
log4cplus-2.1.0.tar.gz  
tar -xzvf log4cplus-2.1.0.tar.gz  
cd log4cplus-2.1.0 && \  
mkdir build && \  
cd build && \  
cmake .. && \  
make && \  
make install && \  
cd .. && \  
rm -rf build
```

```
cd build && \  
cmake .. && \  
sudo make && \  
sudo make install
```

apache-maven-3.9.2从源代码安装。

```
wget https://dlcdn.apache.org/maven/maven-3/3.9.2/binaries/apache-maven-3.9.2-  
bin.tar.gz  
RUN tar -xzvf apache-maven-3.9.2-bin.tar.gz -C /opt
```

Important

如果您看到屏幕提示需要重新启动某些服务，请按 Enter 选择确定。

有关更多信息，请参阅 [Amazon Corretto 11 用户指南](#)。

2. 安装 AWS Command Line Interface。请参阅《[AWS Command Line Interface用户指南](#)》中的“[安装或更新最新版本](#)”的AWS CLI程序”。

第 2 步：为 IP 摄像机 RTSP 网址创建 Amazon Kinesis Video Streams AWS Secrets Manager 和资源

按照以下步骤创建中所需的直播和密钥AWS Secrets Manager。请先执行此步骤，因为您需要策略中已创建资源的 ARN。

创建 Amazon Kinesis Video Streams

使用AWS Management ConsoleAWS CLI、或 API 创建 Amazon Kinesis Video Streams。

在中AWS Management Console，打开 [Amazon Kinesis Video Streams](#) 控制台。在左侧导航栏中选择“视频流”。

有关更多信息，请参阅 [the section called “创建 Kinesis 视频流”](#)。

在 AWS Secrets Manager 中创建秘密

在中AWS Management Console，打开[AWS Secrets Manager控制台](#)。在左侧导航栏中选择“密钥”。

确认选择了相应的区域。

1. 选择存储新密钥。

a. 第 1 步：选择密钥类型

- 选择其他密钥类型。
- 在“键/值对”部分中，添加键值对。

键：MediaURI

Note

密钥必须是MediaURI。这区分大小写。如果输入不正确，则应用程序将无法运行。

值：*Your MediaURI*。

Example

示例：rtsp://<YourCameraIPAddress>:<YourCameraRTSPPort>/
YourCameraMediaURI。

- b. 步骤 2：配置密钥。给这个秘密起个名字。随心所欲地给它起个名字。
- c. 步骤 3：配置轮换-可选。请选择 Next (下一步)。
- d. 第 4 步：查看。选择 Store (存储)。

2. 如果您的密钥没有立即显示，请选择刷新按钮。

选择您的密钥的名称。记下秘密 ARN。

3. 对要从中进行直播的每个 MediaURI 重复此过程。

Note

该AWS网络封锁了一些公共的 RTSP 来源。您无法从 Amazon EC2 实例中访问它们，或者如果您在连接到 VPN 时处于非托管状态。

Important

您的摄像机 RTSP 网址应以 h.264 格式流式传输视频。片段持续时间不得超过中提及的限制。[the section called “制作人 SDK 限制”](#)。

亚马逊 Kinesis Video Streams Kinesis Streams 边缘代理仅支持视频。

运行 `gst-discoverer-1.0 Your RtspUrl` 以确保您的设备可以访问您的摄像头。

保存您创建的所有直播和密钥的 ARN。下一步需要这些。

步骤 3：创建 IAM 权限策略

按照以下步骤创建 IAM 策略。此权限策略允许对 AWS 资源进行选择访问控制（支持的操作的子集）。在本例中，AWS 资源是你希望 Amazon Kinesis Video Streams Edge Agent 直播到的视频流。这些资源还包括 Amazon Kinesis Video Streams Edge Agent 可以检索的 AWS Secrets Manager 机密。有关更多信息，请参阅 [IAM policy](#)。

使用 JSON 策略编辑器创建策略

1. 登录 AWS Management Console，然后通过以下网址打开 IAM 控制台：<https://console.aws.amazon.com/iam/>。
2. 在左侧导航窗格中，选择 Policies（策略）。

如果这是您首次选择策略，则会显示欢迎访问托管式策略页面。选择开始使用。

3. 在页面的顶部，选择创建策略。
4. 在策略编辑器部分，选择 JSON 选项。
5. 输入以下 JSON 策略文档：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:PutMetricData",
        "kinesisvideo:ListStreams",
        "iot:Connect",
        "iot:Publish",
        "iot:Subscribe",
        "iot:Receive"
      ]
    }
  ],
}
```

```

    "Resource": [
        "*"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "kinesisvideo:DescribeStream",
        "kinesisvideo:PutMedia",
        "kinesisvideo:TagStream",
        "kinesisvideo:GetDataEndpoint"
    ],
    "Resource": [
        "arn:aws:kinesisvideo:*:*:stream/streamName1/*",
        "arn:aws:kinesisvideo:*:*:stream/streamName2/*"
    ]
},
{
    "Effect": "Allow",
    "Action": "secretsmanager:GetSecretValue",
    "Resource": [
        "arn:aws:secretsmanager:*:*:secret:*",
        "arn:aws:secretsmanager:*:*:secret:*"
    ]
}
]
}

```

Note

将 `arn:aws:kinesisvideo:*:*:stream/streamName1/` 和 `arn:aws:kinesisvideo:*:*:stream/streamName2/` 替换为视频流的 ARN，并 `arn:aws:secretsmanager:*:*:secret:*` 替换为包含您在中创建的 MediaURI 密钥的 ARN。 [the section called “2. 为您的 IP 摄像机 RTSP 网址创建资源”](#) 使用 ARN 获取你希望 Amazon Kinesis Video Streams Edge Agent 访问的机密。

6. 选择下一步。

Note

您可以随时在可视化和 JSON 编辑器选项卡之间切换。不过，如果您进行更改或在可视化编辑器中选择下一步，IAM 可能会调整策略结构以针对可视化编辑器进行优化。有关更多信息，请参阅 IAM 用户指南中的[策略重组](#)。

7. 在查看并创建页面上，输入您正在创建的策略的策略名称和可选描述。查看此策略中定义的权限以查看策略授予的权限。
8. 选择创建策略可保存新策略。

步骤 4：创建 IAM 角色

您可以代入您在本步骤中创建的角色，以便从 AWS Security Token Service (AWS STS) 获取临时证书。AWS IoT 这是在执行来自亚马逊 Kinesis Video Streams Edge Agent 的凭证授权请求时完成的。

为 Amazon Kinesis Video Streams Video Streams (IAM 控制台) 创建服务角色

1. 登录 AWS Management Console，然后通过以下网址打开 IAM 控制台：<https://console.aws.amazon.com/iam/>。
2. 在 IAM 控制台的导航窗格中，选择角色，然后选择创建角色。
3. 选择自定义信任策略角色类型并粘贴以下策略：

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "Service": "credentials.iot.amazonaws.com"
    },
    "Action": "sts:AssumeRole"
  }
}
```

4. 选中您在中创建的 IAM 策略旁边的复选框 [the section called “3. 创建 IAM 权限策略”](#)。
5. 请选择 Next (下一步) 。
6. 输入角色名称或角色名称后缀，以帮助确定此角色的用途。

Example

示例：KvsEdgeAgentRole

7. (可选) 在描述中，输入新角色的描述。
8. (可选) 通过将标签作为键/值对附加到角色中。

有关在 IAM 中使用标签的更多信息，请参阅 [IAM 用户指南中的为 IAM 资源添加标签](#)。

9. 检查角色，然后选择 Create role。

步骤 5：创建AWS IoT角色别名

按照以下步骤为您AWS IoT在中创建的 IAM 角色创建角色别名 [the section called “4. 创建 IAM 角色”](#)。角色别名是指向 IAM 角色的替代数据模型。AWS IoT证书提供商请求必须包含角色别名，以指明要代入哪个 IAM 角色才能从 AWS Security Token Service (AWS STS) 获取临时证书。有关更多信息，请参阅 [如何使用证书获取安全令牌](#)。

创建AWS IoT角色别名

1. 登录AWS Management Console并打开AWS IoT Core控制台，[网址为 https://console.aws.amazon.com/iot/](https://console.aws.amazon.com/iot/)。
2. 确认选择了相应的区域。
3. 在左侧导航栏中，选择“安全”，然后选择“角色别名”。
4. 选择创建角色别名。
5. 输入角色别名的名称。

Example

示例：KvsEdgeAgentRoleAlias

6. 在角色下拉列表中，选择您在中创建的 IAM 角色 [the section called “4. 创建 IAM 角色”](#)。
7. 选择创建。在下一页上，您会看到一条注释，说明您的角色别名已成功创建。
8. 搜索并选择新创建的角色别名。记下角色别名 ARN。在下一步中，您需要将其用于AWS IoT策略。

步骤 6：创建AWS IoT策略

按照以下步骤创建将附加到设备证书的AWS IoT策略。这为AWS IoT权能授予权限，并允许使用证书假设角色别名。

通过AWS IoT Core策略，您可以控制对AWS IoT Core数据平面的访问。AWS IoT Core数据平面由可用于执行以下操作的操作组成：

- Connect 连接到AWS IoT Core消息代理
- 发送和接收 MQTT 消息
- 获取或更新事物的设备影子

有关更多信息，请参阅 [AWS IoT Core 策略](#)。

使用AWS IoT策略编辑器创建AWS IoT策略

1. 登录AWS Management Console并打开AWS IoT Core控制台，[网址为 https://console.aws.amazon.com/iot/](https://console.aws.amazon.com/iot/)。
2. 在左侧导航栏中，选择“安全”，然后选择“策略”。
3. 选择创建策略。
4. 输入策略的名称。

Example

策略名称的一个例子是 KvsEdgeAccessIotPolicy。

5. (可选) 通过以密钥值对的形式附加标签来向策略添加元数据。

有关在 IAM 中使用标签的更多信息，请参阅AWS IoT Core开发人员指南中的为[AWS IoT资源添加标签](#)。

6. 选择 JSON 选项卡。
7. 在 JSON 策略文档中，粘贴以下内容：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```



```
    "Action": [
      "iot:Connect",
      "iot:Publish",
      "iot:Subscribe",
      "iot:Receive"
    ],
    "Resource": [
      "*"
    ]
  },
  {
    "Effect": "Allow",
    "Action": [
      "iot:AssumeRoleWithCertificate"
    ],
    "Resource": "your-role-alias-arn"
  }
]
```

Note

*your-role-alias-arn*替换为您在中创建的角色别名的 ARN。 [the section called “5. 创建AWS IoT角色别名”](#)

8. 选择“创建”以保存您所做的工作。

步骤 7：创建AWS IoT事物并获取凭证 AWS IoT Core

此时你已经创建了：

- IAM 权限策略。请参阅[the section called “3. 创建 IAM 权限策略”](#)。
- 一个 IAM 角色，附加了权限策略。请参阅[the section called “4. 创建 IAM 角色”](#)。
- IAM AWS IoT 角色的角色别名。请参阅[the section called “5. 创建AWS IoT角色别名”](#)。
- 一种AWS IoT策略，目前与任何AWS资源无关。请参阅[the section called “6. 创建 AWS IoT 策略”](#)。

创建和注册AWS IoT事物并获取AWS IoT Core访问凭证

1. 将设备注册为AWS IoT事物并为该设备生成 X.509 证书。

- a. 登录AWS Management Console并打开AWS IoT Core控制台，[网址为 https://console.aws.amazon.com/iot/](https://console.aws.amazon.com/iot/)。
- b. 选择相应的地区。
- c. 在左侧导航栏中，选择所有设备，然后选择事物。
- d. 选择“创建事物”。
- e. 选择“创建单件事物”，然后选择“下一步”。

1. 第 1 步。指定事物属性

为你的事物键入一个名称，然后选择“下一步”。

2. 第 2 步。配置设备证书

选择“自动生成新证书（推荐）”，然后选择“下一步”。

3. 第 3 步。将策略附加到证书

搜索您在中创建的权限策略[the section called “6. 创建 AWS IoT 策略”](#)。

选中您的策略旁边的复选框，然后选择创建事物。

- f. 在出现的窗口中，下载以下文件：

- 设备证书。这是 X.509 证书。
- 公钥文件
- 私钥文件
- 亚马逊信任服务终端节点（RSA 2048 位密钥：Amazon Root CA 1）

记下每个文件的位置，以供后续步骤使用。

- g. 选择完成。在下一页上，您会看到一条注释，说明您的事物已成功创建。
- h. 将上面下载的文件传输到你的AWS IoT东西上（如果还没有）。

2. 获取您AWS账户的凭证提供商终端节点。

AWS CLI

运行以下命令：

```
aws iot describe-endpoint --endpoint-type iot:CredentialProvider
```

AWS Management Console

在中 [AWS CloudShell](#)，运行以下命令：

```
aws iot describe-endpoint --endpoint-type iot:CredentialProvider
```

请记住这些信息以供后续步骤使用。

3. 获取您AWS账户的设备数据端点。

AWS CLI

运行以下命令：

```
aws iot describe-endpoint --endpoint-type iot:Data-ATS
```

AWS Management Console

执行以下操作：

1. 登录AWS Management Console并打开AWS IoT Core控制台，[网址为 https://console.aws.amazon.com/iot/](https://console.aws.amazon.com/iot/)。
2. 在左侧导航栏中，选择“设置”。
3. 找到设备数据端点。

请记住这些信息以供后续步骤使用。

4. (可选) 验证您的证书是否正确生成。

运行以下命令以验证您的项目是否正确生成。

```
curl --header "x-amzn-iot-thingname:your-thing-name" \  
  --cert /path/to/certificateID-certificate.pem.crt \  
  --key /path/to/certificateID-private.pem.key \  
  --cacert /path/to/AmazonRootCA1.pem \  
  https://your-credential-provider-endpoint/role-aliases/your-role-alias-name/  
  credentials
```

有关更多信息，请参阅[如何使用证书获取安全令牌](#)。

第 8 步：构建并运行亚马逊 Kinesis Video Streams Edge Agent

构建并运行 Amazon Kinesis Video Streams 边缘代理

1. 使用提供给您链接下载tar文件。

如果你填写了 Amazon Kinesis Video Streams Edge Agent 意向表，请查看电子邮件中的下载链接。如果您尚未填写表格，请[在此处](#)填写。

2. 验证校验和。
3. 提取设备中的二进制文件和 jar。

类型：`tar -xvf kvs-edge-agent.tar.gz`。

解压缩后，您的文件夹结构将如下所示：

```
kvs-edge-agent/LICENSE
kvs-edge-agent/THIRD-PARTY-LICENSES
kvs-edge-agent/pom.xml
kvs-edge-agent/KvsEdgeComponent
kvs-edge-agent/KvsEdgeComponent/recipes
kvs-edge-agent/KvsEdgeComponent/recipes/recipe.yaml
kvs-edge-agent/KvsEdgeComponent/artifacts
kvs-edge-agent/KvsEdgeComponent/artifacts/aws.kinesisvideo.KvsEdgeComponent
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/edge_log_config
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/kvs-edge-agent.jar
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/libgstkvssink.so
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/libIngestorPipelineJNI.so
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/lib
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/lib/libcproducer.so
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/lib/libKinesisVideoProducer.so
```

Note

发行版文件夹名称的设置应反映最新的二进制版本号。例如，1.0.0 版本的文件夹名称将设置为 1.0.0。

4. 构建依赖关系 jar。

Note

随附的 jar `kvs-edge-agent.tar.gz` 没有依赖关系。使用以下步骤来构建这些库。

导航到包含 `kvs-edge-agent` 的文件夹 `pom.xml`。

键入 `mvn clean package`。

这将生成一个 jar 文件，其中包含亚马逊 Kinesis Video Streams Edge Agent 所需的 `kvs-edge-agent/target/libs.jar` 依赖项。

5. 将 `libs.jar` 放入包含组件构件的文件夹中。

键入 `mv ./target/libs.jar ./KvsEdgeComponent/artifacts/aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/`。

6. 使用前面步骤中的值设置环境变量。下表提供了变量的描述。

环境变量名	必需	描述
AWS_REGION	有	使用的区域。 示例：us-west -2
AWS_IOT_CA_CERT	有	用于通过 TLS 与后端服务建立信任的 CA 证书的文件路径。 示例： <code>/file/path/to/AmazonRootCA1.pem</code>
AWS_IOT_CORE_CERT	有	X.509 证书的文件路径。

环境变量名	必需	描述
		<p>示</p> <p>例：<code>/file/path/to/certificateID-certificate.pem.crt</code></p>
AWS_IOT_CORE_CREDENTIAL_ENDPOINT	有	<p>您AWS账户的AWS IoT Core 凭证终端节点提供商终端节点。</p> <p>示例：<code>credential-account-specific-prefix.credentials.iot.aws-region.amazonaws.com</code></p>
AWS_IOT_CORE_DATA_PLANE_ENDPOINT	有	<p>您AWS账户AWS IoT Core的数据平面终端节点。</p> <p>示例：<code>data-account-specific-prefix.iot.aws-region.amazonaws.com</code></p>
AWS_IOT_CORE_PRIVATE_KEY	有	<p>公钥/私钥对中使用的私钥的文件路径。有关更多信息，请参阅中的密钥管理AWS IoT。</p> <p>示</p> <p>例：<code>/file/path/to/certificateID-private.pem.key</code></p>

环境变量名	必需	描述
AWS_IOT_CORE_ROLE_ALIAS	有	<p>指向连接时要使用AWS的IAM 角色的角色别名的名称 AWS IoT Core。</p> <p>示例：kvs-edge-role-alias</p>
AWS_IOT_CORE_THING_NAME	有	<p>正在运行应用程序AWS IoT的事物的名称。</p> <p>示例：my-edge-device-thing</p>
GST_PLUGIN_PATH	有	<p>指向包含gstkvssink 和IngestorPipelineJNI 依赖于平台的库的文件夹的文件路径。让gStreamer 加载这些插件。有关更多信息，请参阅下载、构建和配置 gStreamer 元素。</p> <p>示例：<i>/download-location /kvs-edge-agent/KvsEdgeComponent/artifacts/aws.kinesisvideo.KvsEdgeComponent/ EdgeAgent Version /</i></p>

环境变量名	必需	描述
LD_LIBRARY_PATH	有	<p>指向包含cproducer和KinesisVideoProducer 依赖于平台的库的目录的文件路径。</p> <p>示例：<i>/download-location /kvs-edge-agent/KvsEdgeComponent/artifacts/aws.kinesisvideo.KvsEdgeComponent/ EdgeAgentVersion /lib/</i></p>
AWS_KVS_EDGE_CLOUD_WATCH_ENABLED	否	<p>确定 Amazon Kinesis Video Streams Video Streams Edge Agent 是否会将工作健康指标Amazon CloudWatch 发布到上。</p> <p>可接受的 值：TRUE/FALSE (不区分大小写)。FALSE如果未提供，则默认为。</p> <p>示例：假</p>

环境变量名	必需	描述
AWS_KVS_EDGE_LOG_LEVEL	否	<p>亚马逊 Kinesis Video Streams Video Streams Edge Agent 输出的记录级别。</p> <p>可接受的值：</p> <ul style="list-style-type: none"> • 关闭 • ALL • 致命 • ERROR • 警告 • 信息，默认（如果未提供） • 调试 • 跟踪 <p>示例：INFO</p>
AWS_KVS_EDGE_LOG_MAX_FILE_SIZE	否	<p>一旦日志文件达到此大小，就会发生翻转。</p> <ul style="list-style-type: none"> • 最小:0 • 最大：10000 • 默认值：20（如果未提供） • 单位：兆字节 (MB) <p>示例：5</p>

环境变量名	必需	描述
AWS_KVS_EDGE_LOG_OUTPUT_DIRECTORY	否	指向输出 Amazon Kinesis Video Streams Edge Agent 日志的目录的文件路径。./log 如果未提供，则默认为。 示例： <i>/file/path/</i>
AWS_KVS_EDGE_LOG_ROLLOVER_COUNT	否	删除前要保留的翻转日志的数量。 <ul style="list-style-type: none"> • 最小:1 • 最大值：100 • 默认值：10 (如果未提供) 示例：20
AWS_KVS_EDGE_RECORDING_DIRECTORY	否	指向录制媒体将被写入的目录的文件路径。如果未提供，则默认为当前目录。 示例： <i>/file/path/</i>
GST_DEBUG	否	指定要输出的 GStreamer 日志的级别。有关更多信息，请参阅 gStreamer 文档 。 示例：0

环境变量名	必需	描述
GST_DEBUG_FILE	否	指定 GStreamer 调试日志的输出文件。如果未设置，则调试日志将输出为标准错误。有关更多信息，请参阅 gStreamer 文档 。 示例： <code>/tmp/gstreamer-logging.log</code>

7. 清除 GStreamer 缓存。类型：

```
rm ~/.cache/gstreamer-1.0/registry.your-os-architecture.bin
```

有关更多信息，请参阅 [gStreamer 注册表文档](#)。

8. 准备并运行 java 命令。亚马逊 Kinesis Video Streams Edge Agent 接受以下参数：

Java 属性名称	必需	描述
java.library.path	否	指向包含gstkvssink 和IngestorPipelineJNI 依赖库的文件夹的文件路径。如果未提供，Amazon Kinesis Video Streams Edge Agent 将在当前目录中搜索它们。

Important

如果 Amazon Kinesis Video Streams Edge Agent 找不到这些文件，它将无法正常运行。

Java 属性名称	必需	描述
		示例： <i>/file/path/</i>

要设置这些，请-D*java-property-name=value*添加到用于运行 jar 的 java 命令中。

例如：

```
java -Djava.library.path=download-location/kvs-edge-agent/KvsEdgeComponent/
artifacts/aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion \
--add-opens java.base/jdk.internal.misc=ALL-UNNAMED \
-Dio.netty.tryReflectionSetAccessible=true \
-cp kvs-edge-agent.jar:libs.jar \
com.amazonaws.kinesisvideo.edge.controller.ControllerApp
```

Important

在的与的同一个目录中运行上面的 java 命令/*download-location/*
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/*EdgeAgentVersion*。

9. 使用向应用程序发送配置AWS CLI。

a. 创建一个新文件，*example-edge-configuration.json*。

将以下代码粘贴到该文件中。这是一个配置示例，每天从上午 9:00:00 到下午 4:59:59 (根据设备上的系统时间) 进行记录。AWS IoT它还会在每天晚上 7:00:00 至晚上 9:59:59 上传录制的媒体。

有关更多信息，请参阅 [the section called “StartEdgeConfigurationUpdate”](#)。

```
{
  "StreamARN": "arn:aws:kinesisvideo:your-region:your-account-id:stream/your-stream/0123456789012",
  "EdgeConfig": {
    "HubDeviceArn": "arn:aws:iot:your-region:your-account-id:thing/kvs-edge-agent-demo",
    "RecorderConfig": {
```

```
    "MediaSourceConfig": {
      "MediaUriSecretArn": "arn:aws:secretsmanager:your-region:your-account-id:secret:your-secret-dRbHJQ",
      "MediaUriType": "RTSP_URI"
    },
    "ScheduleConfig": {
      "ScheduleExpression": "0 0 9,10,11,12,13,14,15,16 ? * * *",
      "DurationInSeconds": 3599
    }
  },
  "UploaderConfig": {
    "ScheduleConfig": {
      "ScheduleExpression": "0 0 19,20,21 ? * * *",
      "DurationInSeconds": 3599
    }
  },
  "DeletionConfig": {
    "EdgeRetentionInHours": 15,
    "LocalSizeConfig": {
      "MaxLocalMediaSizeInMB": 2800,
      "StrategyOnFullSize": "DELETE_OLDEST_MEDIA"
    },
    "DeleteAfterUpload": true
  }
}
```

- b. 要将文件发送到 Amazon Kinesis Video Streams Edge Agent，请在 AWS CLI 键入以下内容：

```
aws kinesishvideo start-edge-configuration-update --cli-input-json
"file://example-edge-configuration.json"
```

10. 对亚马逊 Kinesis Video Streams Edge Agent 的每个直播重复上述步骤。

步骤 9：（可选）在设备上安装 CloudWatch 代理

Note

注意配 [CloudWatch 额](#)。

按照以下步骤安装和配置 CloudWatch 代理，使其自动将 Amazon Kinesis Video Streams Edge Agent CloudWatch 生成的日志上传到。

有关在您的设备上安装 CloudWatch 代理的[步骤](#)，请参阅《Amazon CloudWatch 用户指南》。

当系统提示您进行配置时，请选择以下配置之一。

Important

以下配置 `file_path` 中的假设使用了默认的日志输出位置。
使用的文件路径假设你正在从以下位置 `download-location/kvs-edge-agent/KvsEdgeComponent/artifacts/aws.kinesisvideo.KvsEdgeComponent/version` 运行 Amazon Kinesis Video Streams Edge Agent 。

- 要将 CloudWatch 代理配置为上传日志并发布设备 RAM 和 CPU 指标，请将以下内容粘贴到配置文件中。

```
{
  "agent": {
    "run_as_user": "ubuntu",
    "metrics_collection_interval": 60
  },
  "metrics": {
    "metrics_collected": {
      "mem": {
        "measurement": [
          "mem_used_percent"
        ],
        "append_dimensions": {
          "IotThing": "YourIotThingName"
        }
      },
    },
    "cpu": {
      "resources": [
        "*"
      ],
      "measurement": [
        "usage_active"
      ],
      "totalcpu": true,
```

```

        "append_dimensions": {
            "IotThing": "YourIotThingName"
        }
    }
},
"logs": {
    "logs_collected": {
        "files": {
            "collect_list": [
                {
                    "file_path": "download-location/kvs-edge-agent/KvsEdgeComponent/
artifacts/aws.kinesisvideo.KvsEdgeComponent/version/log/java_kvs.log",
                    "log_group_name": "/aws/kinesisvideo/EdgeRuntimeAgent",
                    "log_stream_name": "YourIotThingName-java_kvs.log"
                },
                {
                    "file_path": "download-location/kvs-edge-agent/KvsEdgeComponent/
artifacts/aws.kinesisvideo.KvsEdgeComponent/version/log/cpp_kvs_edge.log*",
                    "log_group_name": "/aws/kinesisvideo/EdgeRuntimeAgent",
                    "log_stream_name": "YourIotThingName-cpp_kvs_edge.log"
                },
                {
                    "file_path": "download-location/kvs-edge-agent/KvsEdgeComponent/
artifacts/aws.kinesisvideo.KvsEdgeComponent/version/log/cpp_kvs_streams.log*",
                    "log_group_name": "/aws/kinesisvideo/EdgeRuntimeAgent",
                    "log_stream_name": "YourIotThingName-cpp_kvs_streams.log"
                },
                {
                    "file_path": "download-location/kvs-edge-agent/KvsEdgeComponent/
artifacts/aws.kinesisvideo.KvsEdgeComponent/version/log/cpp_kvssink.log*",
                    "log_group_name": "/aws/kinesisvideo/EdgeRuntimeAgent",
                    "log_stream_name": "YourIotThingName-cpp_kvssink.log"
                }
            ]
        }
    }
}
}

```

- 要仅上传日志而不收集设备的 RAM 和 CPU，请使用以下配置：

```

{
    "logs": {

```

```
"logs_collected": {
  "files": {
    "collect_list": [
      {
        "file_path": "download-location/kvs-edge-agent/KvsEdgeComponent/
artifacts/aws.kinesisvideo.KvsEdgeComponent/version/log/java_kvs.log",
        "log_group_name": "/aws/kinesisvideo/EdgeRuntimeAgent",
        "log_stream_name": "YourIotThingName-java_kvs.log"
      },
      {
        "file_path": "download-location/kvs-edge-agent/KvsEdgeComponent/
artifacts/aws.kinesisvideo.KvsEdgeComponent/version/log/cpp_kvs_edge.log*",
        "log_group_name": "/aws/kinesisvideo/EdgeRuntimeAgent",
        "log_stream_name": "YourIotThingName-cpp_kvs_edge.log"
      },
      {
        "file_path": "download-location/kvs-edge-agent/KvsEdgeComponent/
artifacts/aws.kinesisvideo.KvsEdgeComponent/version/log/cpp_kvs_streams.log*",
        "log_group_name": "/aws/kinesisvideo/EdgeRuntimeAgent",
        "log_stream_name": "YourIotThingName-cpp_kvs_streams.log"
      },
      {
        "file_path": "download-location/kvs-edge-agent/KvsEdgeComponent/
artifacts/aws.kinesisvideo.KvsEdgeComponent/version/log/cpp_kvssink.log*",
        "log_group_name": "/aws/kinesisvideo/EdgeRuntimeAgent",
        "log_stream_name": "YourIotThingName-cpp_kvssink.log"
      }
    ]
  }
}
```

第 10 步：(可选) 以原生进程身份运行 Amazon Kinesis Video Streams Edge Agent

将 Amazon Kinesis Video Streams Video Streams Edge Agent 设置为系统服务。

systemd 是 Linux 设备上的系统和服務管理器。systemd 是管理该过程的推荐方法，因为如果应用程序遇到错误或运行该应用程序的设备断电，它将重启 Amazon Kinesis Video Streams Edge Agent。

执行以下操作：

将 Amazon Kinesis Video Streams Edge Agent 作为原生进程运行

1. 在中创建一个新文件/etc/systemd/system并将其命名*aws.kinesisvideo.edge-runtime-agent.service*。

粘贴以下内容：

```
[Unit]
Description=AWS Kinesis Video Streams edge agent
After=network.target
StartLimitBurst=3
StartLimitInterval=30

[Service]
Type=simple
Restart=on-failure
RestartSec=10
WorkingDirectory=/download-location/kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion
Environment="GST_PLUGIN_PATH=/download-location/kvs-edge-agent/KvsEdgeComponent/
artifacts/aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion"
Environment="LD_LIBRARY_PATH=/download-location/kvs-edge-agent/KvsEdgeComponent/
artifacts/aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/lib"
...
Environment="AWS_IOT_CORE_DATA_ATS_ENDPOINT=data-account-specific-prefix.iot.aws-
region.amazonaws.com"
ExecStart=/usr/lib/jvm/java-11-amazon-corretto/bin/java --add-opens java.base/
jdk.internal.misc=ALL-UNNAMED -Dio.netty.tryReflectionSetAccessible=true -cp kvs-
edge-agent.jar:libs.jar com.amazonaws.kinesisvideo.edge.controller.ControllerApp

[Install]
WantedBy=multi-user.target
```

有关systemd服务配置文件接受的参数的更多信息，请参阅[文档](#)。

Note

在...位置添加所需的环境变量，如中所述[the section called “8. 构建并运行 Amazon Kinesis Video Streams 边缘代理”](#)。

2. 重新加载服务文件以包含新服务。

键入 `sudo systemctl daemon-reload`。

3. 启动服务。

键入 `sudo systemctl start aws.kinesisvideo.edge-runtime-agent.service`。

4. 检查亚马逊 Kinesis Video Streams Video Streams Edge Agent 服务的状态以验证其是否正在运行。

键入 `sudo systemctl status aws.kinesisvideo.edge-runtime-agent.service`。

以下是您将看到的输出示例。

```
aws.kinesisvideo.edge-runtime-agent.service - AWS Kinesis Video Streams edge agent
Loaded: loaded (/etc/systemd/system/aws.kinesisvideo.edge-runtime-agent.service; disabled; vendor preset: enabled)
Active: active (running) since Thu 2023-06-08 19:15:02 UTC; 6s ago
Main PID: 506483 (java)
Tasks: 23 (limit: 9518)
Memory: 77.5M
CPU: 4.214s
CGroup: /system.slice/aws.kinesisvideo.edge-runtime-agent.service
        ##506483 /usr/lib/jvm/java-11-amazon-corretto/bin/java -cp kvs-edge-agent.jar:libs.jar com.amazonaws.kinesisvideo.edge.controller.ControllerApp
```

5. 检查日志中是否存在任何错误。

键入 `journalctl -e -u aws.kinesisvideo.edge-runtime-agent.service`。

6. 键入 `systemctl --help` 以查看用于管理流程的选项的完整列表 `systemctl`。

以下是管理 Amazon Kinesis Video Streams Edge Agent 的一些常用命令：

- 要重新启动，请键入 `sudo systemctl restart aws.kinesisvideo.edge-runtime-agent.service`。
- 要停止，请键入 `sudo systemctl stop aws.kinesisvideo.edge-runtime-agent.service`。
- 要在每次设备重启时自动启动，请键入 `sudo systemctl enable aws.kinesisvideo.edge-runtime-agent.service`。

将 Amazon Kinesis Video Streams 边缘代理部署到 AWS IoT Greengrass

按照以下步骤部署 Amazon Kinesis Video Streams Edge Agent AWS IoT Greengrass 来录制和上传来自 IP 摄像机的媒体。

主题

- [步骤 1：创建 Ubuntu Amazon EC2 实例](#)
- [步骤 2：在设备上设置 AWS IoT Greengrass V2 核心设备](#)
- [第 3 步：为 IP 摄像机 RTSP 网址创建 Amazon Kinesis Video Streams AWS Secrets Manager 和资源](#)
- [步骤 4：为令牌交换服务 \(TES\) 角色添加权限](#)
- [第 5 步：在设备上安装 S AWS IoT Greengrass secret Manager 组件](#)
- [第 6 步：在设备上部署 Amazon Kinesis Video Streams 边缘 AWS IoT Greengrass 代理组件](#)
- [步骤 7：\(可选 \) 在设备上安装 AWS IoT Greengrass 日志管理器组件](#)

步骤 1：创建 Ubuntu Amazon EC2 实例

执行以下操作创建 Ubuntu Amazon EC2 实例。

创建 Ubuntu 亚马逊 EC2 实例

1. 登录到 AWS Management Console 并打开 Amazon EC2 控制台 (<https://console.aws.amazon.com/ec2/>) 。

确认选择了相应的区域。

2. 选择启动实例。

填写以下字段：

- 名称-键入实例的名称。
- 应用程序和操作系统映像 (Amazon 计算机映像) -选择 Ubuntu。
- 实例类型-选择 t2.large。
- 密钥对登录-创建自己的密钥对。
- 网络设置-保留默认设置。

- 配置存储-将音量增加到 256 GiB。
 - 高级设置-保留默认设置。
3. 启动实例并通过 SSH 进入该实例。

执行以下操作：

1. 在左侧导航栏中选择实例，然后选择实例 ID。
 2. 选择右上角的 Connect。
 3. 选择 SSH 客户端，然后按照屏幕上的说明进行操作。
 4. 打开终端并导航到下载的 .pem 文件（可能在~/Downloads）。
 5. 首次执行这些步骤时，您将收到一条消息：“无法确定主机 (...) 的真实性。” 键入“是”。
4. 安装系统库以在实例上构建 Amazon Kinesis Video Streams 边缘代理。

```
wget -O- https://apt.corretto.aws/corretto.key | sudo apt-key add -
sudo add-apt-repository 'deb https://apt.corretto.aws stable main'

sudo apt-get update

sudo apt-get install -y gcc libssl-dev libcurl4-openssl-dev liblog4cplus-dev \
libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev \
gstreamer1.0-plugins-base-apps gstreamer1.0-plugins-bad \
gstreamer1.0-plugins-good gstreamer1.0-tools \
unzip java-11-amazon-corretto-jdk maven
```

Important

如果您看到屏幕提示需要重新启动某些服务，请按 Enter 选择确定。

有关更多信息，请参阅 [Amazon Corretto 11 用户指南](#)。

步骤 2：在设备上设置 AWS IoT Greengrass V2 核心设备

按照以下步骤在 Amazon EC2 实例上安装 AWS IoT Greengrass 核心 nucleus 软件。

设置 AWS IoT Greengrass 核心设备

1. 登录 <https://console.aws.amazon.com/iot/>。AWS Management Console

确认选择了相应的区域。

2. 在左侧导航栏中，选择 Greengrass 设备、核心设备。
3. 选择“设置一台核心设备”。
4. 完成屏幕上的步骤。

- 第 1 步：注册 Greengrass 核心设备。键入设备的名称。
- 步骤 2：添加到事物组以应用持续部署。选择“无群组”。
- 第 3 步：安装 Greengrass Core 软件。选择 Linux。
 - 步骤 3.1：在设备上安装 Java

Java 是作为其中的一部分安装的[the section called “1. 创建 Ubuntu 实例”](#)。如果您尚未安装 Java，请返回该步骤。

- 步骤 3.2：将AWS凭据复制到设备上

打开该bash/zsh选项并将导出命令粘贴到 Amazon EC2 实例中。

- 步骤 3.3：运行安装程序
 1. 在 Ubuntu Amazon EC2 实例中复制并运行下载安装程序并运行安装程序命令。

Note

运行安装程序命令将根据您在上一步中选择的名称自动更新。

2. 记下创建的令牌交换服务 (TES) 角色。稍后您将需要用到它。

Note

默认情况下，创建的角色名为 GreenGrassV2 TokenExchangeRole。

第 3 步：为 IP 摄像机 RTSP 网址创建 Amazon Kinesis Video Streams AWS Secrets Manager 和资源

按照以下步骤创建中所需的直播和密钥AWS Secrets Manager。请先执行此步骤，因为您需要策略中已创建资源的 ARN。

创建 Amazon Kinesis Video Streams

使用AWS Management Console、AWS CLI、或 API 创建 Amazon Kinesis Video Streams。

在中AWS Management Console，打开 [Amazon Kinesis Video Streams](#) 控制台。在左侧导航栏中选择“视频流”。

有关更多信息，请参阅 [the section called “创建 Kinesis 视频流”](#)。

在 AWS Secrets Manager 中创建秘密

在中AWS Management Console，打开 [AWS Secrets Manager](#) 控制台。在左侧导航栏中选择“密钥”。

确认选择了相应的区域。

1. 选择存储新密钥。

a. 第 1 步：选择密钥类型

- 选择其他密钥类型。
- 在“键/值对”部分中，添加键值对。

键：MediaURI

Note

密钥必须是MediaURI。这区分大小写。如果输入不正确，则应用程序将无法运行。

值：*Your MediaURI*。

Example

示例：`rtsp://<YourCameraIPAddress>:<YourCameraRTSPPort>/
YourCameraMediaURI`。

- b. 步骤 2：配置密钥。给这个秘密起个名字。随心所欲地给它起个名字。
 - c. 步骤 3：配置轮换-可选。请选择 Next (下一步)。
 - d. 第 4 步：查看。选择 Store (存储)。
2. 如果您的密钥没有立即显示，请选择刷新按钮。

选择您的密钥的名称。记下秘密 ARN。

3. 对要从中进行直播的每个 MediaURI 重复此过程。

Note

该AWS网络封锁了一些公共的 RTSP 来源。您无法从 Amazon EC2 实例中访问它们，或者如果您在连接到 VPN 时处于非托管状态。

Important

您的摄像机 RTSP 网址应以 h.264 格式流式传输视频。片段持续时间不得超过中提及的限制[the section called “制作人 SDK 限制”](#)。

亚马逊 Kinesis Video Streams Kinesis Streams 边缘代理仅支持视频。

运行 `gst-discoverer-1.0 Your RtspUrl` 以确保您的设备可以访问您的摄像头。

保存您创建的所有直播和密钥的 ARN。下一步需要这些。

步骤 4：为令牌交换服务 (TES) 角色添加权限

向拥有查看密钥权限的设备授予令牌交换服务 (TES) 角色。这是AWS Secrets ManagerAWS IoT Greengrass组件正常工作所必需的。

为 TES 角色添加权限

1. 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/>。
2. 在左侧导航栏中选择“角色”，然后搜索您在流程前面创建的 TES 角色。
3. 在添加权限下拉列表中，选择附加策略。
4. 选择创建策略。
5. 向下滚动并选择编辑。
6. 在策略编辑器中，选择 JSON 并编辑策略。

将该策略替换为以下内容：

Note

将`arn:aws:kinesisvideo:*:*:stream/streamName1/`
`*`和`arn:aws:kinesisvideo:*:*:stream/streamName2/*`替换为您在上一步中创建的直播的 ARN。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kinesisvideo:ListStreams"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kinesisvideo:DescribeStream",
        "kinesisvideo:PutMedia",
        "kinesisvideo:TagStream",
        "kinesisvideo:GetDataEndpoint"
      ],
      "Resource": [
        "arn:aws:kinesisvideo:*:*:stream/streamName1/*",
        "arn:aws:kinesisvideo:*:*:stream/streamName2/*"
      ]
    }
  ]
}
```

7. 在 Add tags (添加标签) 页面上，选择 Next: Review (下一步：审核)。
8. 命名您的策略，然后选择创建策略。

策略名称的一个示例是KvsEdgeAccessPolicy。

9. 关闭选项卡，然后返回到您向 TES 角色附加策略的选项卡。


选择刷新按钮，然后搜索新创建的策略。

选中该复选框并选择附加策略。

在下一个屏幕上，你会看到一条注释，上面写着策略已成功关联到角色。

10. 创建并附加另一个策略，这次是针对您的密钥。

将该策略替换为以下内容：

 Note

arn:aws:secretsmanager:*:*:secret:* 替换为包含您在中创建的 MediaURI 密钥的 ARN。 [the section called “3. 为您的 IP 摄像机 RTSP 网址创建资源”](#)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:GetSecretValue",
      "Resource": [
        "arn:aws:secretsmanager:*:*:secret:*",
        "arn:aws:secretsmanager:*:*:secret:*"
      ]
    }
  ]
}
```

11. 创建并附加另一个策略，这次是针对 Amazon CloudWatch 指标。将该策略替换为以下内容：

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

```
]
}
```

第 5 步：在设备上安装 S AWS IoT Greengrass secret Manager 组件

亚马逊 Kinesis Video Streams Edge Agent 要求 AWS IoT Greengrass 先在设备上安装 Secret Manager 组件。

安装密钥管理器组件

1. 登录 AWS Management Console 并打开 AWS IoT Core 控制台，[网址为 https://console.aws.amazon.com/iot/](https://console.aws.amazon.com/iot/)。确认选择了相应的区域。
2. 在左侧导航栏中，选择 Greengrass 设备，然后选择“部署”。

选择与我们在中创建的目标相同的部署 [the section called “2. 设置 AWS IoT Greengrass 核心设备”](#)。

3. 在右上角的“操作”下拉列表中，选择“修订”。

在出现的弹出窗口中，选择修订部署。

4. 完成以下各节：

- 步骤 1：指定目标。请选择 Next (下一步)。
- 步骤 2：选择组件。
 - 确认已选择 aws.greenGrass.cli 组件。请勿卸载此组件。
 - 切换“仅显示选定的组件”开关并搜索 aws.greengrass。SecretManager。
 - 选中 aws.greengrass 旁边的复选框。SecretManager，然后选择“下一步”。
- 步骤 3：配置组件。将 S AWS IoT Greengrass secret Manager 组件配置为从 AWS IoT Greengrass 环境中下载密钥。


选择 aws.greengrass。SecretManager 组件，然后选择配置组件。

在出现的屏幕中，更新“要合并的配置”框中的 AWS Secrets Manager ARN。

Note

arn:aws:secretsmanager:*:*:secret:* 替换为您在中创建的密钥的 ARN。 [the section called “3. 为您的 IP 摄像机 RTSP 网址创建资源”](#)

```
{
  "cloudSecrets": [
    {
      "arn": "arn:aws:secretsmanager:*:*:secret:*"
    },
    {
      "arn": "arn:aws:secretsmanager:*:*:secret:*"
    }
  ]
}
```

 Note

cloudSecrets是带有密钥的对象的列表arn。有关更多信息，请参阅《AWS IoT Greengrass Version 2开发者指南》中的“[密钥管理器配置](#)”部分。

完成后，选择“确认”，然后选择“下一步”。

- 步骤 4：配置高级设置。选择下一步。
 - 第 5 步：查看。选择部署。
5. 确认AWS Secrets Manager组件和权限已正确安装。

在 Ubuntu Amazon EC2 实例上，键入`sudo /greengrass/v2/bin/greengrass-cli component details --name aws.greengrass.SecretManager`以验证该组件是否已收到更新的配置。

6. 检查AWS IoT Greengrass核心日志。

键入`sudo less /greengrass/v2/logs/greengrass.log`。

查看部署错误。

如果出现错误，请修改部署以删除该`aws.greengrass.SecretManager`组件。

键入`sudo service greengrass restart`以重新启动AWS IoT Greengrass核心服务。

如果部署错误与缺少权限有关，请查看该[the section called “4. 为 TES 角色添加权限”](#)部分以确保 TES 角色具有适当的权限。然后，重复本节。

更新 Sec AWS IoT Greengrass ret Manager 组件上的密钥

Important

只有在AWS IoT Greengrass更新部署时，Secret Manager 组件才会获取和缓存机密。

要更新 Sec AWS IoT Greengrass ret Manager 组件上的密钥，请按照前面的步骤 1-6 进行操作，并进行以下更改。

步骤 3：配置组件。将 S AWS IoT Greengrass ecret Manager 组件配置为从AWS IoT Greengrass环境中下载密钥。

选择 aws.greengrass。SecretManager组件，然后选择配置组件。

在出现的屏幕中，粘贴[""]重置路径框，然后在要合并的配置框中更新 AWS Secrets Manager ARN。

有关更多信息，请参阅[重置更新](#)。

第 6 步：在设备上部署 Amazon Kinesis Video Streams 边缘AWS IoT Greengrass代理组件

在设备上部署 Amazon Kinesis Video Streams 边缘AWS IoT Greengrass代理组件

1. 使用提供的链接下载tar文件。

如果你填写了 Amazon Kinesis Video Streams Edge Agent 意向表，请查看电子邮件中的下载链接。如果您尚未填写表格，请[在此处](#)填写。

2. 验证校验和。
3. 提取设备中的二进制文件和 jar。

类型：`tar -xvf kvs-edge-agent.tar.gz`。

解压缩后，您的文件夹结构将如下所示：

```
kvs-edge-agent/LICENSE
kvs-edge-agent/THIRD-PARTY-LICENSES
kvs-edge-agent/pom.xml
kvs-edge-agent/KvsEdgeComponent
kvs-edge-agent/KvsEdgeComponent/recipes
kvs-edge-agent/KvsEdgeComponent/recipes/recipe.yaml
kvs-edge-agent/KvsEdgeComponent/artifacts
kvs-edge-agent/KvsEdgeComponent/artifacts/aws.kinesisvideo.KvsEdgeComponent
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/edge_log_config

kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/kvs-edge-agent.jar
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/libgstkvssink.so
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/libIngestorPipelineJNI.so
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/lib
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/lib/libcproducer.so
kvs-edge-agent/KvsEdgeComponent/artifacts/
aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/lib/libKinesisVideoProducer.so
```

Note

发行版文件夹名称的设置应反映最新的二进制版本号。例如，1.0.0 版本的文件夹名称将设置为 1.0.0。

4. 构建依赖关系 jar。

Note

kvs-edge-agent.tar.gz 中包含的 jar 没有依赖关系。使用以下步骤来构建这些库。

导航到包含kvs-edge-agent的文件夹pom.xml。

键入 `mvn clean package`。

这将生成一个 jar 文件，其中包含亚马逊 Kinesis Video Streams Edge Agent 所需的 `kvs-edge-agent/target/libs.jar` 依赖项。

5. 将 `libs.jar` 放入包含组件构件的文件夹中。

键入 `mv ./target/libs.jar ./KvsEdgeComponent/artifacts/aws.kinesisvideo.KvsEdgeComponent/EdgeAgentVersion/`。

6. 可选。配置属性。Amazon Kinesis Video Streams Edge Agent AWS IoT Greengrass 在模式下接受以下环境变量：

环境变量名	必需	描述
<code>AWS_REGION</code>	有	<p>使用的区域。</p> <p>示例：<code>us-west-2</code></p> <p>AWS IoT Greengrass 核心软件会自动为您设置此值。有关更多信息，请参阅《AWS IoT Greengrass Version 2 开发人员指南》中的组件环境变量参考主题。</p>
<code>GST_PLUGIN_PATH</code>	有	<p>指向包含 <code>gstkvssink</code> 和 <code>IngestorPipelineJNI</code> 依赖于平台的库的文件夹的文件路径。这让 GStreamer 可以加载这些插件。有关更多信息，请参阅下载、构建和配置 GStreamer 元素。</p> <p>示例：<code>/download-location/kvs-edge-agent/KvsEdgeComponent/art</code></p>

环境变量名	必需	描述
		<p>ifacts/aws.kinesis video.KvsEdgeComponent/<i>EdgeAgent Version</i> /</p>
LD_LIBRARY_PATH	有	<p>指向包含cproducer 和KinesisVideoProducer 依赖于平台的库的目录的文件路径。</p> <p>示例：<i>/download- location</i> /kvs- edge-agent/Kv sEdgeComponent/art ifacts/aws.kinesis video.KvsEdgeComponent/<i>EdgeAgent Version</i> /lib/</p>
AWS_KVS_EDGE_CLOUD WATCH_ENABLED	否	<p>确定 Amazon Kinesis Video Streams Video Streams Edge Agent 是否会将工作健康指标Amazon CloudWatch 发布到上。</p> <p>可接受的 值：TRUE/FALSE (不区分大小写)。FALSE如果未提供， 则默认为。</p> <p>示例：假</p>

环境变量名	必需	描述
AWS_KVS_EDGE_LOG_LEVEL	否	<p>亚马逊 Kinesis Video Streams Video Streams Edge Agent 输出的记录级别。</p> <p>可接受的值：</p> <ul style="list-style-type: none"> • 关闭 • ALL • 致命 • ERROR • 警告 • 信息，默认（如果未提供） • 调试 • 跟踪 <p>示例：INFO</p>
AWS_KVS_EDGE_LOG_MAX_FILE_SIZE	否	<p>一旦日志文件达到此大小，就会发生翻转。</p> <ul style="list-style-type: none"> • 最小:1 • 最大值：100 • 默认值：20（如果未提供） • 单位：兆字节 (MB) <p>示例：5</p>

环境变量名	必需	描述
AWS_KVS_EDGE_LOG_OUTPUT_DIRECTORY	否	指向输出 Amazon Kinesis Video Streams Edge Agent 日志的目录的文件路径。./log 如果未提供，则默认为。 示例： <i>/file/path/</i>
AWS_KVS_EDGE_LOG_ROLLOVER_COUNT	否	删除前要保留的翻转日志的数量。 <ul style="list-style-type: none"> • 最小:1 • 最大值：100 • 默认值：10 (如果未提供) 示例：20
AWS_KVS_EDGE_RECORDING_DIRECTORY	否	指向录制媒体将被写入的目录的文件路径。如果未提供，则默认为当前目录。 示例： <i>/file/path/</i>
GREENGRASS_ROOT_DIRECTORY	否	AWS IoT Greengrass 根目录的文件路径。 /greengrass/v2/ 如果未提供，则默认为。 示例： <i>/file/path/</i>
GST_DEBUG	否	指定要输出的 GStreamer 日志的级别。有关更多信息，请参阅 gStreamer 文档 。 示例：0

环境变量名	必需	描述
GST_DEBUG_FILE	否	指定 GStreamer 调试日志的输出文件。如果未设置，则调试日志将输出为标准错误。有关更多信息，请参阅 gStreamer 文档 。 示例： <code>/tmp/gstreamer-logging.log</code>

打开 `kvs-edge-agent/KvsEdgeComponent/recipes/recipe.yaml` 并修改运行脚本以添加上述任何环境变量。

Important

确保修改后的运行脚本不包含任何制表符。AWS IoT Greengrass 核心软件将无法读取食谱。

7. 部署 Amazon Kinesis Video Streams 边缘 AWS IoT Greengrass 代理组件。

类型：

```
sudo /greengrass/v2/bin/greengrass-cli deployment create \
  --recipeDir <download location>/kvs-edge-agent/KvsEdgeComponent/recipes/ \
  --artifactDir <download location>/kvs-edge-agent/KvsEdgeComponent/artifacts/ \
  --merge "aws.kinesisvideo.KvsEdgeComponent=EdgeAgentVersion"
```

有关更多信息，请参阅《AWS IoT Greengrass Version 2 开发人员指南》中的以下章节：

- [AWS IoT Greengrass CLI 命令](#)
- [将 AWS IoT Greengrass 组件部署到设备](#)

8. 使用向应用程序发送配置 AWS CLI。

- 创建一个新文件，`example-edge-configuration.json`。

将以下代码粘贴到该文件中。这是一个配置示例，每天从上午 9:00:00 到下午 4:59:59（根据设备上的系统时间）进行记录。AWS IoT 它还会在每天晚上 7:00:00 至晚上 9:59:59 上传录制的媒体。

有关更多信息，请参阅 [the section called “StartEdgeConfigurationUpdate”](#)。

```
{
  "StreamARN": "arn:aws:kinesisvideo:your-region:your-account-id:stream/your-stream/0123456789012",
  "EdgeConfig": {
    "HubDeviceArn": "arn:aws:iot:your-region:your-account-id:thing/kvs-edge-agent-demo",
    "RecorderConfig": {
      "MediaSourceConfig": {
        "MediaUriSecretArn": "arn:aws:secretsmanager:your-region:your-account-id:secret:your-secret-dRbHJQ",
        "MediaUriType": "RTSP_URI"
      },
      "ScheduleConfig": {
        "ScheduleExpression": "0 0 9,10,11,12,13,14,15,16 ? * * *",
        "DurationInSeconds": 3599
      }
    },
    "UploaderConfig": {
      "ScheduleConfig": {
        "ScheduleExpression": "0 0 19,20,21 ? * * *",
        "DurationInSeconds": 3599
      }
    },
    "DeletionConfig": {
      "EdgeRetentionInHours": 15,
      "LocalSizeConfig": {
        "MaxLocalMediaSizeInMB": 2800,
        "StrategyOnFullSize": "DELETE_OLDEST_MEDIA"
      },
      "DeleteAfterUpload": true
    }
  }
}
```

- b. 在中键入以下内容将文件发送AWS CLI到亚马逊 Kinesis Video Streams Edge Agent :

```
aws kinesishvideo start-edge-configuration-update --cli-input-json
"file://example-edge-configuration.json"
```

9. 对亚马逊 Kinesis Video Streams Edge Agent 的每个直播重复上述步骤。

步骤 7：（可选）在设备上安装AWS IoT Greengrass日志管理器组件

Note

注意配[CloudWatch](#)额。

按照以下步骤将 Amazon Kinesis Video Streams Video Streams Edge Agent 日志配置 CloudWatch 为使用日志管理器AWS IoT Greengrass组件自动上传到。

安装AWS IoT Greengrass日志管理器组件

1. 确认AWS IoT Greengrass设备角色具有[相应的权限](#)。
 - 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. 在左侧导航栏中单击“角色”。
 - c. 选择在中创建的 TES 角色的名称[the section called “2. 设置AWS IoT Greengrass核心设备”](#)。如有必要，请使用搜索栏。
 - d. 选择 GreengrassV2TokenExchangeRoleAccess 策略。
 - e. 选择 JSON 选项卡并验证策略是否如下所示：


```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogStreams",
        "s3:GetBucketLocation"
      ]
    }
  ]
}
```

```

    ],
    "Resource": "*"
  }
]
}

```

- f. 如果该GreengrassV2TokenExchangeRoleAccess策略不存在，或者缺少某些必需的权限，请使用这些权限创建一个新的 IAM 策略，并将其附加到中创建的 TES 角色 [the section called “2. 设置AWS IoT Greengrass核心设备”](#)。
2. 登录AWS Management Console并打开AWS IoT Core控制台，[网址为 https://console.aws.amazon.com/iot/](https://console.aws.amazon.com/iot/)。确认选择了相应的区域。
 3. 在左侧导航栏中，选择 Greengrass 设备，然后选择“部署”。
选择与您在中创建的目标相同的部署 [the section called “2. 设置AWS IoT Greengrass核心设备”](#)。
 4. 在右上角选择操作，然后选择修订。
在出现的弹出窗口中，选择修订部署。
 5. 完成以下各节：
 - a. 步骤 1：指定目标。请选择 Next (下一步)。
 - b. 步骤 2：选择组件。
 - i. 验证 aws.greenGrass.cli 组件和 aws.greengrass.SecretManager 组件仍处于选中状态。

 Important

不要卸载这些组件。

- ii. 切换“仅显示选定的组件”开关并搜索 aws.greengrass.LogManager。
 - iii. 选中 aws.greengrass 旁边的复选框。LogManager，然后选择“下一步”。
- c. 步骤 3：配置组件。配置AWS IoT Greengrass日志管理器组件以上传由 Amazon Kinesis Video Streams Edge Agent 生成的日志。

选择 aws.greengrass.LogManager 组件，然后选择配置组件。

在出现的屏幕中，将以下日志管理器配置粘贴到要合并的配置框中。

```
{
```

```

"logsUploaderConfiguration": {
  "componentLogsConfigurationMap": {
    "aws.kinesisvideo.KvsEdgeComponent/java_kvs.log": {
      "diskSpaceLimit": "100",
      "diskSpaceLimitUnit": "MB",
      "logFileDirectoryPath": "/greengrass/v2/work/
aws.kinesisvideo.KvsEdgeComponent/log",
      "logFileRegex": "java_kvs.log\\w*"
    },
    "aws.kinesisvideo.KvsEdgeComponent/cpp_kvs_edge.log": {
      "diskSpaceLimit": "100",
      "diskSpaceLimitUnit": "MB",
      "logFileDirectoryPath": "/greengrass/v2/work/
aws.kinesisvideo.KvsEdgeComponent/log",
      "logFileRegex": "cpp_kvs_edge.log\\w*"
    },
    "aws.kinesisvideo.KvsEdgeComponent/cpp_kvssink.log": {
      "diskSpaceLimit": "100",
      "diskSpaceLimitUnit": "MB",
      "logFileDirectoryPath": "/greengrass/v2/work/
aws.kinesisvideo.KvsEdgeComponent/log",
      "logFileRegex": "cpp_kvssink.log\\w*"
    },
    "aws.kinesisvideo.KvsEdgeComponent/cpp_kvs_streams.log": {
      "diskSpaceLimit": "100",
      "diskSpaceLimitUnit": "MB",
      "logFileDirectoryPath": "/greengrass/v2/work/
aws.kinesisvideo.KvsEdgeComponent/log",
      "logFileRegex": "cpp_kvs_streams.log\\w*"
    }
  }
},
"periodicUploadIntervalSec": "1"
}

```

⚠ Important

上述配置logFileDirectoryPath中的假设使用了默认的日志输出位置。

Note

有关日志管理器配置的每个参数的更多信息，请参阅《AWS IoT Greengrass Version 2开发人员指南》的“[日志管理器](#)”部分。

完成后，选择“确认”，然后选择“下一步”。

- d. 步骤 4：配置高级设置。选择下一步。
 - e. 第 5 步：查看。选择部署。
6. 确认AWS日志管理器组件和权限已正确安装。
 7. 在 Ubuntu Amazon EC2 实例上，键入`sudo /greengrass/v2/bin/greengrass-cli component details --name aws.greengrass.LogManager`以验证组件已收到更新的配置。
 8. 检查AWS IoT Greengrass核心日志。

键入 `sudo less /greengrass/v2/logs/greengrass.log`。

查看部署错误。

如果出现错误，请修改部署以删除该`aws.greengrass.LogManager`组件。

键入`sudo service greengrass restart`以重新启动AWS IoT Greengrass核心服务。

如果部署错误与缺少权限有关，请查看[the section called “4. 为 TES 角色添加权限”](#)以确保 TES 角色具有适当的权限。然后，重复本节。

亚马逊 Kinesis Video Streams Edge Agent 常见问题解答

以下是亚马逊 Kinesis Video Streams Edge Agent 服务的一些常见问题。

亚马逊 Kinesis Video Streams Video Streams Edge Agent 支持哪些操作系统？

亚马逊 Kinesis Video Streams Kinesis Streams 边缘代理目前支持以下操作系统：

Ubuntu

- 22.x
 - AMD64
- 18.x
 - ARM

AL2

- amzn2
 - AMD64 amazonlinux : 2.0.20210219.0-amd64 (Snowball)

亚马逊 Kinesis Video Streams Video Streams Edge Agent 是否支持 H.265 媒体？

亚马逊 Kinesis Video Streams Edge Agent 仅支持 H.264 基本直播。

亚马逊 Kinesis Video Streams Video Streams Edge Agent 能在 AL2 中运行吗？

是。

我怎样才能能在AWS IoT事物或设备中运行多个直播？

[the section called “StartEdgeConfigurationUpdate”](#)向相同HubDeviceArn但不同的亚马逊 Kinesis Video Streams/ AWS Secrets Manager ARN 发送另一个。

发送**StartEdgeConfigurationUpdate**后如何编辑？

HubDeviceArn使用相同的 Amazon Kinesis Video Streams ARN 发送相同内容的更新[the section called “StartEdgeConfigurationUpdate”](#)。当应用程序收到来自 Amazon Kinesis Video Streams 的消息时，它会覆盖该流的先前配置。届时将发生变化。

你有常见的例子**ScheduleConfigs**吗？

Amazon Kinesis Video Streams Edge Agent 使用其运行设备的系统时间。

描述	ScheduleExpression	DurationInSeconds
全天候录制，每小时上传	## ScheduleConfig#	
每天上午 9:00:00-下午 4:59:59	0 0 9-16 **# *	3599
9:00:00 AM - 4:59:59 PM weekdays	0 0 9-16# * 2-6 * 0 0 9-16 ? * 2,3,4,5,6 *	3599 3599
	0 0 9-16 ? * MON-FRI *	3599
	0 0 9-16 ? * MON,TUE,W ED,THU,FRI *	3599
周末上午 9:00:00-下午 4:59:59	0 0 9-16 ? * SAT,SUN *	3599
工作日晚上 10:00:00-晚上 11:59:59	0 0 22,23 ? * MON-FRI *	3599
每天上午 9:00:00-上午 10:00:00	0 0 9 * * ? *	3600
每天下午 4:00:00-下午 5:59:59	0 0 16-17 * * ? *	3599

有关更多示例，请参阅 [Quartz 文档](#)。

有最大直播限制吗？

目前，亚马逊 Kinesis Video Streams Edge Agent 的硬限制为每台设备 16 个直播。使用 [the section called “DeleteEdgeConfiguration”](#) API 从设备中删除直播。使用更新同一直播的配置 [the section called “StartEdgeConfigurationUpdate”](#) 不会增加设备的直播数量。

如何重启出错的作业？

如果遇到错误，Amazon Kinesis Video Streams Edge Agent 将尝试重启作业。但是，如果出现某些错误（例如配置错误），则必须手动重启作业。

要确定哪些作业需要手动重启，请参阅中的FatalError指标。[the section called “使用以下方式监控 Amazon Kinesis Video Streams Edge Agent CloudWatch”](#)

重新发送[the section called “StartEdgeConfigurationUpdate”](#)以重新启动直播的作业。

如何监控我的亚马逊 Kinesis Video Streams Edge Agent 的运行状况？

有关更多信息，请参阅 [the section called “使用以下方式监控 Amazon Kinesis Video Streams Edge Agent CloudWatch”](#)。

通过 VPC 流式传输视频

该测试版已在欧洲（巴黎）区域 eu-west-3 提供预览版。要访问这些组件和我们的入门指南，[请给我们发送电子邮件](#)。

Amazon Kinesis Video Streams VPC 终端节点服务允许您通过亚马逊网络流式传输和消费视频，而无需通过公共互联网传输任何数据。

要申请访问权限，请[通过电子邮件向我们发送](#)以下信息：

- 账户 ID
- 直播 ARN
- VPC ID

Note

我们最多可能需要一周的时间才能将您添加到服务中。

如果您过去未使用过 VPC 终端节点，请查看以下信息以熟悉该概念：

- [AWS PrivateLink背景](#)
- [VPC 入门指南](#)

其他信息

在您加入测试版后，我们将通过电子邮件向您发送有关此功能的其他信息的链接。

VPC 终端节点程序

限额

主要配额差异是：

- 降低所有带宽 API 的配额 (2 mbps)：
 - PutMedia

- GetMedia
- GetMediaForFragmentList
- 每位客户允许 10 个直播

创建端点

一旦你被允许上市，你就会收到 Amazon Kinesis Video Streams 的 VPC 终端节点服务名称。它会看起来像 `com.amazonaws.region.kinesisvideo`。

使用亚马逊 [VPC 控制台](#) 或 [AWS Command Line Interface AWS CLI \(\)](#) 为 Amazon Kinesis Video Streams 创建接口 VPC 终端节点。

在中 AWS CLI，键入以下内容：

```
aws ec2 create-vpc-endpoint \  
--vpc-id customer-provided-vpc-id \  
--service-name com.amazonaws.eu-west-2.kinesisvideo \  
--private-dns-enabled
```

Important

您的 VPC 内的流量将使用私有 DNS 通过终端节点进行路由。如果您不启用此功能，则需要实现自己的 DNS 逻辑。有关私有 DNS 的更多信息，请参阅 [AWS PrivateLink 文档](#)。

有关该 AWS CLI 选项的更多信息，请参阅 [create-vpc-endpoint](#)。

控制对端点的访问

您可以将终端节点策略附加到控制对 Amazon Kinesis Video Streams 的访问权限的 VPC 终端节点。该策略指定以下信息：

- 可以执行操作的委托人，
- 可以执行的操作，以及
- 可以对其执行操作的资源。

有关更多信息，请参阅 AWS PrivateLink 指南中的 [使用终端节点策略控制对具有 VPC 终端节点的服务的访问](#)。

以下是 Amazon Kinesis Video Streams 的终端节点策略示例。当连接到终端节点时，此策略将拒绝所有委托人访问所有资源上列出的PutMedia操作。

```
{
  "Statement": [
    {
      "Principal": "*",
      "Effect": "Deny",
      "Action": [
        "kinesisvideo:PutMedia"
      ],
      "Resource": "*"
    }
  ]
}
```

Kinesis 视频流中的图片

您可以使用亚马逊 Kinesis Video Streams API 和 SDK 来帮助您从视频流中提取图像。您可以将这些图像用于增强的播放应用程序，例如缩略图或增强的擦除功能，也可以用于机器学习管道。Kinesis Video Streams 通过 API 提供按需图像提取，或从采集的视频中的元数据标签中自动提取图像。

有关使用 Kinesis Video Streams 管理的图像支持的信息，请参阅：

- [按需生成图像 \(GetImages\)](#)-此 API 允许客户从 Kinesis Video Streams 中存储的视频中提取单张图像或多张图像。
- [自动生成图像 \(S3 交付 \)](#) -将 Kinesis Video Streams 配置为根据上传视频中的标签自动实时从视频数据中提取图像，并将图像传送到客户指定的 S3 存储桶。

主题

- [GetImages 入门](#)
- [亚马逊 S3 交付入门](#)

GetImages 入门

对图像的托管支持提供了一种完全托管的方式，可以从流式传输和存储在 Kinesis Video Streams 中的视频数据中获取图像。您可以使用图像运行机器学习 (ML) 工作负载，例如人、宠物或车辆检测。图像还可用于向回放添加交互元素，例如动作事件的图像预览和视频片段的擦除。

有关该的更多信息Get Images功能，请参见[GetImages](#)在亚马逊 Kinesis 视频直播存档媒体API 参考指南。

亚马逊 S3 交付入门

当前，客户运行和管理自己的图像转码管道，以创建用于各种目的的图像，例如清理、图像预览、在图像上运行 ML 模型等。Kinesis Video Streams 提供了对图像进行转码和传送的功能。Kinesis Video Streams 将根据标签自动实时从视频数据中提取图像，并将图像传送到客户指定的 S3 存储桶。

UpdateImageGenerationConfiguration

要设置 Kinesis 视频流以允许向 Amazon S3 生成图像，请执行以下操作：

1. 创建一个S3 存储桶用于基于使用新 API 在 SDK 中添加的标签生成图像。请注意S3 网址，这是下一步更新流的图像生成配置时所必需的。
2. 创建一个名为的 JSON 文件update-image-generation-input.json使用以下内容作为输入。

```
{
  "StreamName": "TestStream",
  "ImageGenerationConfiguration": {
    {
      "Status": "ENABLED",
      "DestinationConfig": {
        {
          "DestinationRegion": "us-east-1",
          "Uri": "s3://bucket-name"
        },
        "SamplingInterval": 200,
        "ImageSelectorType": "PRODUCER_TIMESTAMP",
        "Format": "JPEG",
        "FormatConfig": {
          "JPEGQuality": "80"
        },
        "WidthPixels": 320,
        "HeightPixels": 240
      }
    }
  }
}
```

你可以使用AWS CLI调用[UpdateImageGenerationConfiguration](#)用于添加先前创建的 Amazon S3 ARN 并将状态更改为 API 操作ENABLED。

```
aws kinesishvideo update-image-generation-configuration \
--cli-input-json file://./update-image-generation-input.json \
```

请求：

```
UpdateImageGenerationConfiguration HTTP/1.1
```

```
Method: 'POST'
```

```
Path: '/updateImageGenerationConfiguration'
Body: {
  StreamName: 'String', // Optional. Either stream name or arn should be passed
  StreamArn: 'String', // Optional. Either stream name or arn should be passed
  ImageGenerationConfiguration : {
    // required
    Status: 'Enum', // ENABLED | DISABLED,
    ImageSelectorType: 'Enum', // SERVER_TIMESTAMP | PRODUCER_TIMESTAMP..
    DestinationConfig: {
      DestinationRegion: 'String',
      Uri: string,
    },
    SamplingInterval: 'Number'//
    Format: 'Enum', // JPEG | PNG
    // Optional parameters
    FormatConfig: {
      'String': 'String',
    },
    WidthPixels: 'Number', // 1 - 3840 (4k).
    HeightPixels: 'Number' // 1 - 2160 (4k).
  }
}
```

回应 :

```
HTTP/1.1 200
Content-type: application/json
Body: {
}
```

Note

更新图像生成配置后，启动图像生成工作流程至少需要 1 分钟。等待至少 1 分钟后再调用PutMedia更新通话之后。

DescribeImageGenerationConfiguration

要查看已经为直播设置的图像生成配置，客户可以制作DescribeImageGenerationConfiguration请求，如下所示。

请求:

```
DescribeImageGenerationConfiguration HTTP/1.1
```

```
Method: 'POST'
Path: '/describeImageGenerationConfiguration'
Body: {
  StreamName: 'String', // Optional. Either stream name or arn should be passed
  StreamArn: 'String', // Optional. Either stream name or arn should be passed
}
```

响应:

```
HTTP/1.1 200
Content-type: application/json
Body: {
  ImageGenerationConfiguration : {
    Status: 'Enum',
    ImageSelectorType: 'Enum', // SERVER_TIMESTAMP | PRODUCER_TIMESTAMP
    DestinationConfig: {
      DestinationRegion: 'String'
      Uri: 'string',
    },
    SamplingInterval: 'Number',
    Format: 'Enum',
    FormatConfig: {
      'String': 'String',
    },
    WidthPixels: 'Number',
    HeightPixels: 'Number'
  }
}
```

要了解有关... 的更多信息DescribeImageGenerationConfiguration功能，请参见[DescribeImageGenerationConfiguration](#) 在亚马逊 Kinesis 视频直播开发者指南。

制片人 MKV 标签

您可以使用 Kinesis Video Streams Producer SDK 通过在 SDK 中公开 API 操作来标记感兴趣的特定片段。有关标签的示例，请参见[这段代码](#)。调用此 API 后，SDK 将添加一组预定义的 MKV 标签以及片段数据。Kinesis Video Streams 将识别这些特殊的 MKV 标签，并根据该流的图像处理配置启动图像生成工作流程。

与 Amazon S3 图像生成标签一起提供的任何片段元数据都将另存为 Amazon S3 元数据。

制作者 MKV 标签的语法

```
|+ Tags
| + Tag
| // MANDATORY: Predefined MKV tag to trigger image generation for the fragment
| + Simple
| + Name: AWS_KINESISVIDEO_IMAGE_GENERATION

| // OPTIONAL: S3 prefix which will be set as prefix for generated image.
| + Simple
| + Name: AWS_KINESISVIDEO_IMAGE_PREFIX
| + String: image_prefix_in_s3 // 256 bytes max m

| // OPTIONAL: Key value pairs that will be persisted as S3 Image object metadata.
| + Simple
| + Name: CUSTOM_KEY_1 // Max 128 bytes
| + String: CUSTOM_VALUE_1 // Max 256 bytes
| + Simple
| + Name: CUSTOM_KEY_2 // Max 128 bytes
| + String: CUSTOM_VALUE_2 // Max 256 bytes
```

使用 Producer SDK 添加元数据标签PutEventMetaData

这个PutEventMetaData函数追加与事件关联的 MKV 文件。PutEventMetaData需要两个参数。第一个参数是一个事件，其值来自STREAM_EVENT_TYPE枚举。第二个参数，[pStreamEventMetadata](#)，是可选的，可用于将其他元数据作为键值对包括在内。最多可以添加五对元数据的键值对。

Limits

下表列出了与元数据标签相关的限制。如果元数据标签限制是可调整的，您可以通过您的客户经理申请增加。

限制	最大值	可调整
图像前缀长度	256	否
可选的元数据密钥长度	128	否
可选的元数据值长度	256	否
可选元数据的最大数量	10	是

S3 对象元数据

默认情况下，Kinesis Video Streams 将设置片段编号，制片人，以及服务器时间戳作为 Amazon S3 对象元数据生成的图像。如果在 MKV 标签中指定了任何其他片段数据，则这些标签也将添加到 Amazon S3 对象元数据中。以下示例显示了 Amazon S3 对象元数据的正确语法。

```
{
  // KVS S3 object metadata
  x-amz-meta-aws_kinesisvideo_fragment_number : 'string',
  x-amz-meta-aws_kinesisvideo_producer_timestamp: 'number',
  x-amz-meta-aws_kinesisvideo_server_timestamp: 'number',

  // Optional key value pair sent as part of the MKV tags
  custom_key_1: custom_value_1,
  custom_key_2: custom_value_2,
}
```

S3 对象路径 (图片)

以下列表显示了对对象路径的正确格式并描述了路径中的每个元素。

格式：

ImagePrefix_## ID_StreamName_ImageTimecode_## ID. #####

1. ImagePrefix-的价值AWS_KINESISVIDEO_IMAGE_PREFIX。
- 2.AccountID -创建直播时使用的账户 ID。
3. StreamName-为其生成图像的流的名称。
4. ImageTimecode-生成图像的片段中的纪元时间码。
5. RandomID-随机 GUID。
6. file-extension-基于所请求的图像格式的 JPG 或 PNG。

防止限制的 Amazon S3 URI 建议

如果您向 Amazon S3 写入数千张图片，则存在限制的风险。有关更多信息，请参见[S3 前缀放置请求限制](#)。

Amazon S3 前缀的起始限制为每秒 3,500 个 PUT 请求，随着时间的推移，唯一前缀的 PUT 限制将逐渐增加。避免使用日期和时间作为 Amazon S3 前缀。时间编码的数据一次只能影响一个前缀，并且还会定期更改，从而使之前的前缀扩展无效。为了实现更快、一致的 Amazon S3 扩展，我们建议在 Amazon S3 目标 URI 中添加一个随机前缀，例如十六进制代码或 UUID。例如，十六进制代码前缀自然会将您的请求随机拆分为 16 个不同的前缀（每个唯一十六进制字符的前缀），这将允许在 Amazon S3 自动扩展后每秒 56,000 个 PUT 请求。

Kinesis Video Streams 中的通知

当媒体片段可供使用时，Kinesis Video Streams 会 Amazon Simple Notification Service 使用（亚马逊 SNS）通知通知客户。以下主题说明了如何开始使用通知。

UpdateNotificationConfiguration

使用此 API 操作更新直播的通知信息。有关该UpdateNotificationConfiguration功能的更多信息，请参阅[UpdateNotificationConfiguration](#) 《亚马逊 Kinesis Video Streams 开发者指南》。

Note

更新通知配置后，至少需要一分钟才能启动通知。至少等待一分钟，然后在更新调用PutMedia后调用。

DescribeNotificationConfiguration

使用此 API 来描述附加到直播的通知配置。有关该DescribeNotificationConfiguration功能的更多信息，请参阅[DescribeNotificationConfiguration](#) 《亚马逊 Kinesis Video Streams 开发者指南》。

制作人 MKV 标签

你可以使用 Kinesis Video Streams Producer SDK 通过在 SDK 中公开 API 操作来标记感兴趣的特定片段。请[在此段代码中](#)查看其工作原理示例。调用此 API 后，SDK 将添加一组预定义的 MKV 标签以及片段数据。Kinesis Video Streams 将识别这些特殊的 MKV 标签，并为标记的片段启动通知。

随通知 MKV 标签一起提供的任何片段元数据都将作为 Amazon SNS 主题有效负载的一部分发布。

制作人 MKV 标签的语法

```
|+ Tags
| + Tag
| // MANDATORY: Predefined MKV tag to trigger the notification for the fragment
| + Simple
| + Name: AWS_KINESISVIDEO_NOTIFICATION
```

```

|   + String
| // OPTIONAL: Key value pairs that will be sent as part of the Notification payload
|   + Simple
|     + Name: CUSTOM_KEY_1 // Max 128 bytes
|     + String:CUSTOM_VALUE_1 // Max 256 bytes
|   + Simple
|     + Name: CUSTOM_KEY_2 // Max 128 bytes
|     + String: CUSTOM_VALUE_2 // Max 256 bytes

```

MKV 标签限制

下表列出了与元数据标签相关的限制。如果元数据标签限制是可调整的，您可以通过您的客户经理申请增加限制。

限制	最大值	可调整
可选的元数据密钥长度	128	不支持
可选的元数据值长度	256	不支持
可选元数据的最大数量	10	是

亚马逊 SNS 主题有效负载

通过上一个工作流程启动的任何通知都将传送 Amazon SNS 主题有效负载，如以下示例所示。此示例是一条 Amazon SNS 消息，它是在使用来自 Amazon Simple Queue Service（亚马逊 SQS）队列的通知数据之后出现的。

```

{
  "Type" : "Notification",
  "MessageId" : Message ID,
  "TopicArn" : SNS ARN,
  "Subject" : "Kinesis Video Streams Notification",
  "Message" : "{\"StreamArn\": \"Stream Arn\", \"FragmentNumber\": \"Fragment Number\",
  \"FragmentStartProducerTimestamp\": \"FragmentStartProducerTimestamp\",
  \"FragmentStartServerTimestamp\": \"FragmentStartServerTimestamp\",
  \"NotificationType\": \"PERSISTED\", \"NotificationPayload\": {\" CUSTOM_KEY_1:
  \"CUSTOM_VALUE_1,

```

```
    \CUSTOM_KEY_2:\CUSTOM_VALUE_2}}",
  "Timestamp" : "2022-04-25T18:36:29.194Z",
  "SignatureVersion" : Signature Version,
  "Signature" : Signature,
  "SigningCertURL" : Signing Cert URL,
  "UnsubscribeURL" : Unsubscribe URL
}
```

```
Subject: "Kinesis Video Streams Notification"
Message:
{
  "StreamArn":Stream Arn,
  "FragmentNumber":Fragment Number,
  "FragmentStartProducerTimestamp":Fragment Start Producer Timestamp,
  "FragmentStartServerTimestamp":Fragment Start Server Timestamp,
  "NotificationType":"PERSISTED",
  "NotificationPayload":{
    CUSTOM_KEY_1:CUSTOM_VALUE_1,
    CUSTOM_KEY_2:CUSTOM_VALUE_2
  }
}
```

查看您的亚马逊 SNS 消息

您无法直接从 Amazon SNS 主题中读取消息，因为没有 API 可以这样做。要查看消息，请在 SQS 队列中订阅 SNS 主题，或者选择任何其他 [Amazon SNS 支持的目的地](#)。但是，查看消息的最有效方法是使用 Amazon SQS。

使用亚马逊 SQS 查看您的亚马逊 SNS 消息

1. 创建 A [amazon SQS 队列](#)。
2. 从中 AWS Management Console，打开下方设置为目的地的 Amazon SNS 主题。NotificationConfiguration
3. 选择“创建订阅”，然后选择在第一步中创建的 Amazon SQS 队列。
4. 在启用通知配置并在片段中添加通知 MKV 标签的情况下运行会PutMedia话。
5. 在亚马逊 SQS 控制台中选择亚马逊 SQS 队列，然后为亚马逊 SQS 队列选择发送和接收消息。
6. 轮询留言。此命令应显示PutMedia会话生成的所有通知。有关轮询的信息，请参阅 [Amazon SQS 短期和长期投票](#)。

亚马逊 Kinesis Video Streams 中的安全

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。

本文档可帮助你了解在使用 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配置 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 服务来帮助您监控和保护您的 Kinesis Video Streams 资源。

主题

- [Kinesis Video Streams 中的数据保护](#)
- [使用 IAM 控制对 Kinesis Video Streams 资源的访问权限](#)
- [使用控制对 Kinesis Video Streams 资源的访问权限 AWS IoT](#)
- [监控 Amazon Kinesis Video Streams](#)
- [亚马逊 Kinesis Video Streams 的合规性验证](#)
- [亚马逊 Kinesis Video Streams 的弹性](#)
- [Kinesis Video Streams 中的基础设施安全](#)
- [Kinesis Video Streams 的安全最佳实践](#)

Kinesis Video Streams 中的数据保护

您可以使用 () 密钥使用服务器端加密 (SSE)，通过对 Amazon Kinesis Video Streams 中的静态数据进行加密，从而满足严格的数据管理要求。AWS Key Management Service AWS KMS

主题

- [什么是 Kinesis Video Streams 的服务器端加密？](#)
- [成本、地区和性能注意事项](#)
- [如何开始使用服务器端加密？](#)
- [创建和使用用户生成的 KMS 密钥](#)
- [使用用户生成的 KMS 密钥的权限](#)

什么是 Kinesis Video Streams 的服务器端加密？

服务器端加密是 Kinesis Video Streams 中的一项功能，它使用您指定的在数据处于静止状态之前自动对其 AWS KMS 进行加密。数据在写入 Kinesis Video Streams 流存储层之前会对其进行加密，从存储中取回数据后会被解密。因此，在 Kinesis Video Streams 服务中，您的数据始终处于静态加密状态。

借助服务器端加密，您的 Kinesis 视频流制作者和使用者无需管理 KMS 密钥或加密操作。如果启用了数据保留，则您的数据在进入和离开 Kinesis Video Streams 时会自动加密，因此您的静态数据将被加密。AWS KMS 提供了服务器端加密功能使用的所有密钥。AWS KMS 简化了 Kinesis Video Streams Kinesis Video Streams 的 KMS 密钥的使用，该密钥 AWS 由导入到服务 AWS KMS 中的用户指定的密钥管理。AWS KMS

成本、地区和性能注意事项

当你应用服务器端加密时，你需要支付 AWS KMS API 使用量和密钥费用。与自定义 AWS KMS 密钥不同，(Default) aws/kinesis-videoKMS 密钥是免费提供的。但是，你仍然必须支付 Kinesis Video Streams 代表你产生的 API 使用费用。

API 使用费用适用于每个 KMS 密钥，包括自定义密钥。AWS KMS 费用会随着您在数据创建者和使用者身上使用的用户凭证数量而变化，因为每个用户凭证都需要对每个用户凭证进行唯一的 API 调用。AWS KMS

下面按资源介绍各项费用：

键

- AWS 由 (别名 aws/kinesis-video =) 管理的 Kinesis Video Streams 的 KMS 密钥不收费。
- 用户生成的 KMS 密钥需要 AWS KMS key 付费。有关更多信息，请参阅[AWS Key Management Service 定价](#)。

AWS KMS API 使用情况

生成新数据加密密钥或检索现有加密密钥的 API 请求会随着流量的增加而增加，并且需要支付 AWS KMS 使用成本。有关更多信息，请参阅[AWS Key Management Service 定价：用量](#)。

即使保留期设置为 0（无保留期），Kinesis Video Streams 也会生成密钥请求。

按地区划分的服务器端加密的可用性

在所有提供 Kinesis Video Streams 的地方，都可以对 Kinesis 视频流 AWS 区域 进行服务器端加密。

如何开始使用服务器端加密？

Kinesis Video Streams 始终启用服务器端加密。如果在创建直播时未指定用户提供的密钥，则使用默认密钥（由 Kinesis Video Streams 提供）。

在创建 Kinesis 视频流时，必须将用户提供的 KMS 密钥分配给 Kinesis 视频流。以后您无法使用 [UpdateStream](#) API 为直播分配不同的密钥。

您可以通过两种方式为 Kinesis 视频流分配用户提供的 KMS 密钥：

- 在中创建 Kinesis 视频流时 AWS Management Console，请在创建新视频流页面的加密选项卡中指定 KMS 密钥。
- 使用 [CreateStream](#) API 创建 Kinesis 视频流时，请在参数中指定密钥 ID。KmsKeyId

创建和使用用户生成的 KMS 密钥

本节介绍如何创建和使用您自己的 KMS 密钥，而不是使用由 Amazon Kinesis Video Streams 管理的密钥。

创建用户生成的 KMS 密钥

有关如何创建自己的密钥的信息，请参阅 AWS Key Management Service 开发者指南中的[创建密钥](#)。为账户创建密钥后，Kinesis Video Streams 服务会在 KMS 主密钥列表中返回这些密钥。

使用用户生成的 KMS 密钥

向您的使用者、生产者和管理员应用正确的权限后，您就可以在自己的密钥 AWS 账户 或其他密钥中使用自定义 KMS 密钥 AWS 账户。您账户中的所有 KMS 密钥都显示在控制台的 KMS 主密钥列表中。

要使用位于其他账户中的自定义 KMS 密钥，您必须拥有使用这些密钥的权限。此外，您必须使用 `CreateStream` API 创建流。您不能在控制台中创建的直播中使用来自不同账户的 KMS 密钥。

Note

在执行 `PutMedia` 或 `GetMedia` 操作之前，不会访问 KMS 密钥。这会产生以下结果：

- 如果您指定的密钥不存在，则 `CreateStream` 操作会成功，但对流 `PutMedia` 的 `GetMedia` 操作将失败。
- 如果您使用提供的密钥 (`aws/kinesis-video`)，则在执行第一个 `PutMedia` 或 `GetMedia` 操作之前，该密钥不会出现在您的账户中。

使用用户生成的 KMS 密钥的权限

在对用户生成的 KMS 密钥使用服务器端加密之前，必须将 KMS 密钥策略配置为允许对流进行加密以及对流记录进行加密和解密。有关 AWS KMS 权限的示例和更多信息，请参阅 [AWS KMS API 权限：操作和资源参考](#)。

Note

使用默认服务密钥进行加密不需要应用自定义 IAM 权限。

在使用用户生成的 KMS 密钥之前，请验证您的 Kinesis 视频流制作者和使用者 (IAM 委托人) 是否为主密钥策略 AWS KMS 中的用户。否则，与流相关的读写操作会失败，这可能最终导致数据丢失、处理延迟或应用程序挂起。您可以使用 IAM policy 来管理 KMS 密钥的权限。有关更多信息，请参阅 [将 IAM 策略与一起使用 AWS KMS](#)。

制作者权限示例

您的 Kinesis 视频流制作者必须获得以下许可：`kms:GenerateDataKey`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
    "Action": [
      "kms:GenerateDataKey"
    ],
    "Resource": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  {
    "Effect": "Allow",
    "Action": [
      "kinesis-video:PutMedia",
    ],
    "Resource": "arn:aws:kinesis-video:*:123456789012:MyStream"
  }
]
```

消费者权限示例

您的 Kinesis 视频流用户必须获得以下许可：`kms:Decrypt`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kinesis-video:GetMedia",
      ],
      "Resource": "arn:aws:kinesis-video:*:123456789012:MyStream"
    }
  ]
}
```

使用 IAM 控制对 Kinesis Video Streams 资源的访问权限

您可以将 AWS Identity and Access Management (IAM) 与 Amazon Kinesis Video Streams 配合使用，以控制组织中的用户是否可以使用特定的 Kinesis Video Streams API 操作执行任务，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](#)
- [Kinesis Video Streams 的操作](#)
- [Kinesis Video Streams 的 Amazon 资源名称 \(ARN \)](#)
- [向其他 IAM 账户授予访问 Kinesis 视频流的权限](#)
- [Kinesis Video Streams 的策略示例](#)

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：

- 效果-效果可以是Allow或Deny。默认情况下 用户没有使用资源和 API 操作的权限，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。
- 操作 — 操作是您授予或拒绝权限的特定 API 操作。
- 资源-受操作影响的资源。要在语句中指定资源，必须使用其 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 策略时，我们建议您使用 [IAM 策略生成器](#) 和 [IAM 策略模拟器](#)。

Kinesis Video Streams 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，请使用以下前缀和 API 操作kinesisvideo:的名称：。例如：kinesisvideo:CreateStream、kinesisvideo:ListStreams 和 kinesisvideo:DescribeStream。

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:stream/stream-name/code
```

例如：

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

您可以使用获取直播的 ARN。 [DescribeStream](#)

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 账户授予权限才能对 Kinesis 视频流执行操作。下面概述了向各个账户授予对视频流的访问权的常规步骤：

1. 获取您要向其授予对流执行操作的权限的账户的 12 位数账户 ID (例如，111111111111)。
2. 在拥有流的账户中创建一个托管策略，此策略支持您希望授予的访问权级别。有关 Kinesis Video Streams 资源的策略示例，[示例策略](#)请参阅下一节中的。
3. 创建角色，并指定要为其授予权限的账户。然后附加您在上一步中创建的策略。
4. 创建允许对您在上一步中创建的角色AssumeRole执行操作的托管策略。例如，此角色可能如下所示：

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::123456789012:role/CustomRole"
  }
}
```

有关授予跨账户访问权限的 step-by-step 说明，请参阅[AWS 账户 使用 IAM 角色委派访问权限](#)。

Kinesis Video Streams 的策略示例

以下示例策略演示了如何控制用户对 Kinesis Video Streams 的访问权限

Example 1：允许用户从任何 Kinesis 视频流中获取数据

此策略允许用户或群组对任何 Kinesis 视频流执

行DescribeStreamGetDataEndpointGetMediaListStreams、、、和ListTagsForStream操作。此策略适用于可从任何视频流获取数据的用户。

```
{
  "Version": "2012-10-17",
```

```
"Statement": [  
  {  
    "Effect": "Allow",  
    "Action": [  
      "kinesisvideo:Describe*",  
      "kinesisvideo:Get*",  
      "kinesisvideo:List*"  
    ],  
    "Resource": "*"  
  }  
]  
}
```

Example 2 : 允许用户创建 Kinesis 视频流并向其写入数据

此策略允许用户或组执行 CreateStream 和 PutMedia 操作。此策略适用于可创建视频流并向该流发送数据的安保摄像头。

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

Example 3 : 允许用户完全访问所有 Kinesis Video Streams 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 : 允许用户向特定的 Kinesis 视频流写入数据

此策略允许用户或组将数据写入特定的视频流。此策略适用于可将数据发送到单个流的设备。

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

使用控制对 Kinesis Video Streams 资源的访问权限 AWS IoT

本节介绍如何允许设备（例如摄像机）仅向一个特定的 Kinesis 视频流发送音频和视频数据。您可以使用 AWS IoT 证书提供商和 AWS Identity and Access Management (IAM) 角色来执行此操作。

设备可以使用 X.509 证书通过 TLS 双向身份验证协议 AWS IoT 进行连接。其他 AWS 服务（例如 Kinesis Video Streams）不支持基于证书的身份验证，但可以使用签名版本 4 格式的凭据 AWS AWS 进行调用。签名版本 4 算法通常要求呼叫者拥有访问密钥 ID 和私有访问密钥。AWS IoT 有一个凭据提供程序，允许您使用内置的 X.509 证书作为唯一的设备身份来验证 AWS 请求（例如，对 Kinesis Video Streams Video Streams 的请求）。这样就无需在设备上存储访问密钥 ID 和私有访问密钥。

凭证提供者使用 X.509 证书对客户端（在本例中为要向视频流发送数据的摄像机上运行的 Kinesis Video Streams SDK）进行身份验证，并颁发临时的有限权限安全令牌。您可以使用该令牌对任何 AWS 请求（在本例中为对 Kinesis Video Streams 的调用）进行签名和身份验证。有关更多信息，请参阅[授权直接呼叫 AWS 服务](#)。

这种验证摄像机向 Kinesis Video Streams 发出的请求的方法要求您创建和配置 IAM 角色并为该角色附加相应的 IAM 策略，AWS IoT 以便证书提供者可以代表您担任该角色。

有关的更多信息 AWS IoT，请参阅[AWS IoT Core 文档](#)。有关 IAM 的更多信息，请参阅[AWS Identity and Access Management \(IAM\)](#)。

主题

- [AWS IoT ThingName 作为直播名称](#)
- [AWS IoT CertificateId 作为直播名称](#)
- [使用 AWS IoT 凭据直播到硬编码的直播名称](#)

AWS IoT ThingName 作为直播名称

主题

- [步骤 1：创建 AWS IoT 事物类型和 AWS IoT 事物](#)
- [步骤 2：创建由代入的 IAM 角色 AWS IoT](#)
- [步骤 3：创建和配置 X.509 证书](#)
- [第 4 步：使用你的 Kinesis 视频流测试 AWS IoT 凭证](#)
- [第 5 步：在摄像机的文件系统上部署 AWS IoT 证书和凭证，并将数据流式传输到视频流](#)

步骤 1：创建 AWS IoT 事物类型和 AWS IoT 事物

在中 AWS IoT，事物是特定设备或逻辑实体的表示。在本例中，你要配置资源级访问控制的 AWS IoT 东西代表你的 Kinesis 视频流。要创建事物，首先必须创建 AWS IoT 事物类型。您可以使用 AWS IoT 事物类型来存储与相同事物类型关联的所有事物共有的描述和配置信息。

1. 以下示例命令可以创建事物类型 `kvs_example_camera`：

```
aws --profile default iot create-thing-type --thing-type-name kvs_example_camera >
iot-thing-type.json
```

2. 以下示例命令创建 `kvs_example_camera_stream` 的事物类型为：

```
aws --profile default iot create-thing --thing-name kvs_example_camera_stream --
thing-type-name kvs_example_camera > iot-thing.json
```

步骤 2：创建由代入的 IAM 角色 AWS IoT

IAM 角色与用户类似，因为角色是一种 AWS 身份，其权限策略决定了该身份可以做什么和不能做什么 AWS。任何需要角色的人都可以代入该角色。当您代入角色时，它会为您提供角色会话的临时安全凭证。

在执行来自客户端的凭证授权请求时，可以代入您在此步骤中创建的角色 AWS IoT 来从安全令牌服务 (STS) 获取临时证书。在本例中，客户端是在您的相机上运行的 Kinesis Video Streams SDK。

执行以下步骤来创建和配置此 IAM 角色：

1. 创建一个 IAM 角色。

以下示例命令可创建一个名为 KVSCameraCertificateBasedIAMRole 的 IAM 角色：

```
aws --profile default iam create-role --role-name KVSCameraCertificateBasedIAMRole
--assume-role-policy-document 'file://iam-policy-document.json' > iam-role.json
```

您可以将以下信任策略 JSON 用于 iam-policy-document.json：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "credentials.iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. 接下来，将权限策略附加到您之前创建的 IAM 角色。此权限策略允许对 AWS 资源进行选择性访问控制（支持的操作的子集）。在这种情况下，AWS 资源就是您希望摄像机发送数据的视频流。换句话说，所有配置步骤完成后，您的摄像机将只能向该视频流发送数据。

```
aws --profile default iam put-role-policy --role-name
KVSCameraCertificateBasedIAMRole --policy-name KVSCameraIAMPolicy --policy-
document 'file://iam-permission-document.json'
```

您可以对 iam-permission-document.json 使用以下 IAM 策略 JSON：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```

        "Effect": "Allow",
        "Action": [
            "kinesisvideo:DescribeStream",
            "kinesisvideo:PutMedia",
            "kinesisvideo:TagStream",
            "kinesisvideo:GetDataEndpoint"
        ],
        "Resource": "arn:aws:kinesisvideo:*:*:stream/${credentials-
iot:ThingName}/*"
    }
]
}

```

请注意，此政策仅授权对占位符 (`${credentials-iot:}`) 指定的视频流 (AWS 资源) 执行指定操作。ThingName 当 AWS IoT 凭证提供者在请求中发送视频流名称 ThingName 时，此占位符会取用 `thin AWS IoT g` 属性的值。

3. 接下来，为您的 IAM 角色创建角色别名。角色别名是一个指向 IAM 角色的备用数据模型。AWS IoT 证书提供商请求必须包含角色别名，以指明要代入哪个 IAM 角色才能从 STS 获取临时证书。

以下示例命令可以创建一个称作 `KvsCameraIoTRoleAlias` 的角色别名。

```

aws --profile default iot create-role-alias --role-alias KvsCameraIoTRoleAlias --
role-arn $(jq --raw-output '.Role.Arn' iam-role.json) --credential-duration-seconds
3600 > iot-role-alias.json

```

4. 现在，您可以使用角色别名创建 AWS IoT 允许使用证书代入角色的策略 (附加证书后)。

以下示例命令为 AWS IoT 被调用创建策略 `KvsCameraIoTPolicy`。

```

aws --profile default iot create-policy --policy-name KvsCameraIoTPolicy --policy-
document 'file://iot-policy-document.json'

```

您可以使用以下命令来创建 `iot-policy-document.json` 文档 JSON：

```

cat > iot-policy-document.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [

```

```

        "iot:AssumeRoleWithCertificate"
    ],
    "Resource": "$(jq --raw-output '.roleAliasArn' iot-role-alias.json)"
}
]
}
EOF

```

步骤 3：创建和配置 X.509 证书

设备之间的通信（您的视频流）通过使用 X.509 证书受到保护。AWS IoT

1. 创建证书，您必须将之前创建的策略附加到 AWS IoT 该证书。

```

aws --profile default iot create-keys-and-certificate --set-as-active --
certificate-pem-outfile certificate.pem --public-key-outfile public.pem.key --
private-key-outfile private.pem.key > certificate

```

2. 将 AWS IoT（之前 KvsCameraIoTPolicy 创建的）的策略附加到此证书。

```

aws --profile default iot attach-policy --policy-name KvsCameraIoTPolicy --target
$(jq --raw-output '.certificateArn' certificate)

```

3. 将你的 AWS IoT 东西 (kvs_example_camera_stream) 附加到你刚刚创建的证书上：

```

aws --profile default iot attach-thing-principal --thing-name
kvs_example_camera_stream --principal $(jq --raw-output '.certificateArn'
certificate)

```

4. 要通过 AWS IoT 凭证提供商授权请求，您需要凭 AWS IoT 证端点，该端点是您的 AWS 账户 ID 所独有的。您可以使用以下命令来获取 AWS IoT 凭据端点。

```

aws --profile default iot describe-endpoint --endpoint-type iot:CredentialProvider
--output text > iot-credential-provider.txt

```

5. 除了之前创建的 X.509 证书外，您还必须拥有 CA 证书，才能通过 TLS 与后端服务建立信任。您可以使用以下命令获取 CA 证书：

```

curl --silent 'https://www.amazontrust.com/repository/SFSRootCAG2.pem' --output
cacert.pem

```

第 4 步：使用你的 Kinesis 视频流测试 AWS IoT 凭证

现在，您可以测试到目前为止已设置的 AWS IoT 凭证。

1. 首先，创建一个要用于测试此配置的 Kinesis 视频流。

Important

使用与您在上一步中创建 AWS IoT 的事物名称相同的名称创建视频流 (`kvs_example_camera_stream`)。

```
aws kinesismedia create-stream --data-retention-in-hours 24 --stream-name
kvs_example_camera_stream
```

2. 接下来，致电 AWS IoT 证书提供商以获取临时证书：

```
curl --silent -H "x-amzn-iot-thingname:kvs_example_camera_stream" --cert
certificate.pem --key private.pem.key https://IOT_GET_CREDENTIAL_ENDPOINT/role-
aliases/KvsCameraIoTRoleAlias/credentials --cacert ./cacert.pem > token.json
```

Note

你可以使用以下命令获取 `IOT_GET_CREDENTIAL_ENDPOINT`：

```
IOT_GET_CREDENTIAL_ENDPOINT=`cat iot-credential-provider.txt`
```

输出 JSON 包含访问密钥、密钥和会话令牌，你可以用它们来访问 Kinesis Video Streams。

3. 在测试中，您可以使用这些凭据为 `kvs_example_camera_stream` 示例视频 `DescribeStream` 流调用 Kinesis Video Streams API。

```
AWS_ACCESS_KEY_ID=$(jq --raw-output '.credentials.accessKeyId' token.json)
AWS_SECRET_ACCESS_KEY=$(jq --raw-output '.credentials.secretAccessKey' token.json)
AWS_SESSION_TOKEN=$(jq --raw-output '.credentials.sessionToken' token.json) aws
kinesisvideo describe-stream --stream-name kvs_example_camera_stream
```

第 5 步：在摄像机的文件系统中部署 AWS IoT 证书和凭证，并将数据流式传输到视频流

Note

本节中的步骤描述了从正在使用 Kinesis 的摄像机向 Kinesis 视频流发送媒体。[the section called “C++ 创建者库”](#)

1. 将之前步骤中生成的 X.509 证书、私钥和 CA 证书复制到相机的文件系统。指定这些文件的存储路径、角色别名以及用于运行 `gst-launch-1.0` 命令或示例应用程序的 AWS IoT 凭据端点。
2. 以下示例命令使用 AWS IoT 证书授权向 Kinesis Video Streams 发送视频：

```
gst-launch-1.0 rtspsrc location=rtsp://YourCameraRtspUrl short-header=TRUE !
  rtph264depay ! video/x-h264,format=avc,alignment=au ! h264parse ! kvssink stream-
  name="kvs_example_camera_stream" aws-region="YourAWSRegion" iot-certificate="iot-
  certificate,endpoint=credential-account-specific-prefix.credentials.iot.aws-
  region.amazonaws.com,cert-path=/path/to/certificate.pem,key-path=/path/to/
  private.pem.key,ca-path=/path/to/cacert.pem,role-aliases=KvsCameraIoTRoleAlias"
```

AWS IoT CertificateId 作为直播名称

要通过某 AWS IoT 物来表示您的设备（例如您的摄像头），但授权不同的直播名称，则可以使用该 AWS IoT `certificateId` 属性作为直播名称，并使用为直播提供 Kinesis Video Stream AWS IoT s 权限。完成此操作的步骤与前面概述的步骤类似，但有一些改动。

- 按如下方式修改您的 IAM 角色 (`iam-permission-document.json`) 的权限策略：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kinesisvideo:DescribeStream",
        "kinesisvideo:PutMedia",
        "kinesisvideo:TagStream",
        "kinesisvideo:GetDataEndpoint"
      ]
    }
  ],
}
```

```

        "Resource": "arn:aws:kinesisvideo:*:*:stream/${credentials-
iot:AwsCertificateId}/*"
    }
]
}

```

Note

资源 ARN 将证书 ID 用作流名称的占位符。当您使用证书 ID 作为直播名称时，IAM 权限将起作用。从证书中获取证书 ID，这样您就可以在以下描述流 API 调用中将其用作直播名称。

```
export CERTIFICATE_ID=`cat certificate | jq --raw-output '.certificateId'`
```

- 使用 Kinesis Video Streams describe-stream CLI 命令验证此更改：

```

AWS_ACCESS_KEY_ID=$(jq --raw-output '.credentials.accessKeyId' token.json)
AWS_SECRET_ACCESS_KEY=$(jq --raw-output '.credentials.secretAccessKey' token.json)
AWS_SESSION_TOKEN=$(jq --raw-output '.credentials.sessionToken' token.json) aws
kinesisvideo describe-stream --stream-name ${CERTIFICATE_ID}

```

- 在 Kinesis Video Streams C AWS IoT ++ SDK [的示例应用程序中将](#)证书 ID 传递给凭证提供者：

```

credential_provider =
make_unique<IotCertCredentialProvider>(iot_get_credential_endpoint,
    cert_path,
    private_key_path,
    role_alias,
    ca_cert_path,
    certificateId);

```

Note

请注意，您要将 thingname 传递给 AWS IoT 凭证提供商。您可以使用 getenv 将 thingname 传递给演示应用程序，就像传递其他属性一样。AWS IoT 运行示例应用程序时，将证书 ID 用作命令行参数中的流名称。

使用 AWS IoT 凭据直播到硬编码的直播名称

要通过某 AWS IoT 事物展示你的设备（例如你的摄像头），但授权直播到特定的亚马逊 Kinesis 视频流，请使用向直播提供 Amazon Kinesis Video Streams 权限。AWS IoT 该过程与前面的章节类似，但有一些改动。

按如下方式修改您的 IAM 角色 (`iam-permission-document.json`) 的权限策略：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kinesisvideo:DescribeStream",
        "kinesisvideo:PutMedia",
        "kinesisvideo:TagStream",
        "kinesisvideo:GetDataEndpoint"
      ],
      "Resource": "arn:aws:kinesisvideo:*:*:stream/YourStreamName/*"
    }
  ]
}
```

将前面步骤中生成的 X.509 证书、私钥和 CA 证书复制到相机的文件系统中。

指定这些文件的存储路径、角色别名、AWS IoT 事物名称以及用于运行 `gst-launch-1.0` 命令或示例应用程序的 AWS IoT 凭据端点。

以下示例命令使用 AWS IoT 证书授权将视频发送到 Amazon Kinesis Video Streams：

```
gst-launch-1.0 rtspsrc location=rtsp://YourCameraRtspUrl short-header=TRUE !
rtph264depay ! video/x-h264,format=avc,alignment=au ! h264parse ! kvssink
stream-name="YourStreamName" aws-region="YourAWSRegion" iot-certificate="iot-
certificate,endpoint=credential-account-specific-prefix.credentials.iot.aws-
region.amazonaws.com,cert-path=/path/to/certificate.pem,key-path=/path/to/
private.pem.key,ca-path=/path/to/cacert.pem,role-aliases=KvsCameraIoTRoleAlias,iot-
thing-name=YourThingName"
```

监控 Amazon Kinesis Video Streams

Kinesis Video Streams 为您的直播流提供监控功能。有关更多信息，请参阅 [监控](#)。

亚马逊 Kinesis Video Streams 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，并提供了部署以安全性和合规性为重点 AWS 的基准环境的步骤。
- 在 [Amazon Web Services 上构建 HIPAA 安全与合规性](#) — 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](#)。
- [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。

亚马逊 Kinesis Video Streams 的弹性

AWS 全球基础设施是围绕 AWS 区域和可用区构建的。AWS 区域提供多个物理隔离和隔离的可用区，这些可用区通过低延迟、高吞吐量和高度冗余的网络相连。利用可用区，您可以设计和操作在可用

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 中的基础设施安全

作为一项托管服务，Amazon Kinesis Video Streams 受 AWS [亚马逊网络服务：安全流程概述白皮书](#)中描述的[全球网络安全程序](#)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。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 委托人关联的私有访问密钥对请求进行签名。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。

Kinesis Video Streams 的安全最佳实践

Amazon Kinesis Video Streams 提供了许多安全功能，供您在制定和实施自己的安全策略时考虑。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[什么是最低权限以及为什么需要它？](#)

使用 IAM 角色

制作者和客户端应用程序必须具有有效的凭据才能访问 Kinesis Video Streams。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。这些是长期凭证，不会自动轮换，如果遭到泄露，可能会对业务产生重大影响。

相反，您应该使用 IAM 角色来管理您的制作者和客户端应用程序访问 Kinesis Video Streams 的临时证书。使用角色时，无需使用长期证书（例如用户名和密码或访问密钥）即可访问其他资源。

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](#)

CloudTrail 用于监控 API 调用

Kinesis Video Streams AWS CloudTrail 与一项服务配合使用，可记录用户、角色或用户在 Kinesis Video Streams 中采取的操作。

您可以使用收集的信息 CloudTrail 来确定向 Kinesis Video Streams 发出的请求、发出请求的 IP 地址、谁发出了请求、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 CloudTrail API 调用”](#)。

Kinesis 视频直播制作入库

亚马逊 Kinesis Video Streams Producer 库是 Kinesis Video Streams 制作者 SDK 中的一组库 客户端使用库和 SDK 构建设备端应用程序，以便安全地连接到 Kinesis Video Streams，并使用流媒体数据在控制台或客户端应用程序中实时查看。

媒体数据可通过以下方式进行流式传输：

- 实时
- 缓冲几秒钟后
- 媒体上传后

创建 Kinesis Video Streams 流后，即可开始向其发送数据。您可以使用 SDK 创建应用程序代码，从媒体源提取视频数据（称为帧）并将其上传到 Kinesis Video Streams。这些应用程序也称为创建者应用程序。

创建者库包含以下组件：

- [Kinesis 视频直播制作入客户端](#)
- [Kinesis 视频直播制作入库](#)

Kinesis 视频直播制作入客户端

Kinesis Video Streams 制作入客户端包括单类 `KinesisVideoClient` 类。该类管理媒体源，从来源接收数据，并在数据从媒体源流向 Kinesis Video Streams 时管理直播生命周期。它还提供了 `MediaSource` 用于定义 Kinesis Video Streams 与您的专有硬件和软件之间交互的接口。

媒体源几乎可以是任何内容。例如，您可以使用摄像头媒体源或麦克风媒体源。媒体源不仅限于音频源和视频源。例如，数据日志可能是文本文件，但仍可作为数据流发送。您的手机上也可以安装多个摄像头以便同时流式处理数据。

要从这些源中的任意一个获取数据，可以实施 `MediaSource` 接口。此接口支持我们未提供内置支持的其他情况。例如，你可以选择向 Kinesis Video Streams 发送以下内容：

- 诊断数据流（例如，应用程序日志和事件）
- 来自红外线摄像头、雷达或深度摄像头的的数据

Kinesis Video Streams 不为摄像机等媒体制作设备提供内置实现。要从这些设备提取数据，您必须实施代码，从而创建您自己的自定义媒体源实现。然后，您可以明确地将您的自定义媒体源注册到 KinesisVideoClient，它将数据上传到 Kinesis Video Streams。

Kinesis Video Streams 制作者客户端可用于 Java 和安卓应用程序。有关更多信息，请参阅 [使用 Java 创建者库](#) 和 [使用 Android 创建者库](#)。

Kinesis 视频直播制作人库

Kinesis Video Streams 制作者库包含在 Kinesis Video Streams 制作者客户端中。该库也可以直接供那些想要更深入地与 Kinesis Video Streams 集成的人使用。它支持与具有专有操作系统的设备、网络堆栈或有限的设备上资源进行集成。

Kinesis Video Streams 制作者库实现了用于直播到 Kinesis Video Streams 的状态机。它提供回调钩子，这需要您提供自己的传输实施，并显式处理传入和传出服务的每条消息。

出于以下原因，你可以选择直接使用 Kinesis Video Streams 制作者库：

- 要在其上运行应用程序的设备没有 Java 虚拟机。
- 您希望使用非 Java 语言编写应用程序代码。
- 由于内存和处理能力等限制，你想减少代码的开销并将其限制在最低抽象级别。

目前，Kinesis Video Streams 制作者库可用于安卓、C、C++ 和 Java 应用程序。有关更多信息，请参阅以下支持的语言相关话题。

相关主题

[使用 Java 创建者库](#)

[使用 Android 创建者库](#)

[使用 C++ 创建者库](#)

[使用 C 创建者库](#)

[在 Raspberry Pi 上使用 C++ 创建者开发工具包](#)

使用 Java 创建者库

您可以使用亚马逊 Kinesis Video Streams 提供的 Java Producer 库以最少的配置编写应用程序代码，将媒体数据从设备发送到 Kinesis 视频流。

执行以下步骤将您的代码与 Kinesis Video Streams 集成，以便您的应用程序可以开始将数据流式传输到您的 Kinesis 视频流：

1. 创建 `KinesisVideoClient` 对象的实例。
2. 通过提供媒体源信息创建 `MediaSource` 对象。例如，当创建摄像头媒体源时，您需要提供相应信息，例如，识别摄像头并指定摄像头所用编码方面的信息。

如果要开始流式处理，您必须创建自定义媒体源。

3. 将媒体源注册到 `KinesisVideoClient`。

将媒体源注册到 `KinesisVideoClient` 后，每当数据对媒体源可用时，都会随数据一起调用 `KinesisVideoClient`。

过程：使用 Java 创建者开发工具包

此过程演示如何在 Java 应用程序中使用 Kinesis Video Streams Java Producer Client 向您的 Kinesis 视频流发送数据。

这些步骤不需要您具备摄像头或麦克风等媒体源。相反，出于测试目的，该代码会生成包含一系列字节的示例帧。在您从摄像头和麦克风等实际源发送媒体数据时，您可以使用相同的编码模式。

该过程包括以下步骤：

- [下载和配置代码](#)
- [编写和检查代码](#)
- [运行和验证代码](#)

先决条件

- 在示例代码中，您可以通过指定在凭证配置文件中设置的配置文件来提供 AWS 证书。如果尚未执行此操作，请先设置凭证配置文件。有关更多信息，请参阅中的[设置用于开发的 AWS 凭据和区域AWS SDK for Java](#)。

Note

Java 示例使用 `SystemPropertiesCredentialsProvider` 对象获取您的证书。提供程序从 `aws.accessKeyId` 和 `aws.secretKey` Java 系统属性检索这些凭证。您可以在 Java 开发环境中设置这些系统属性。有关如何设置 Java 系统属性的信息，请参阅特定集成开发环境 (IDE) 的文档。

- 你 `NativeLibraryPath` 必须包含你的 `KinesisVideoProducerJNI` 文件，[网址为 `https://github.com/aws-labs/amazon-kinesis-video-streams-producer-sdk-cpp`](https://github.com/aws-labs/amazon-kinesis-video-streams-producer-sdk-cpp)。此文件的文件扩展名取决于您的操作系统：
 - `KinesisVideoProducer` 适用于 Linux 的 `jni.so`
 - `KinesisVideoProducer` 适用于 macOS 的 `jni.dylib`
 - `KinesisVideoProducerJNI.dll` 适用于 Windows

Note

[适用于 macOS、Ubuntu、Windows 和 Raspbian 的预建库可 `src/main/resources/lib` 在 `https://github.com/aws-labs/-git` 中找到。](https://github.com/aws-labs/amazon-kinesis-video-streams-producer-sdk-java) `amazon-kinesis-video-streams-producer-sdk-java` 对于其他环境，请编译 [C++ 创建者库](#)。

步骤 1：下载并配置 Java 制作器库代码

在 Java 创建者库过程的这一部分中，您需要下载 Java 示例代码、将项目导入到 Java IDE 中并配置库位置。

有关此示例的先决条件和其他详细信息，请参阅[使用 Java 创建者库](#)。

1. 创建一个目录，然后从 GitHub 存储库中克隆示例源代码。

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

2. 打开你使用的 Java 集成开发环境 (IDE) (例如 Eclipse [se](#) 或 [Intel JetBrains li J ID EA](#))，然后导入你下载的 Apache Maven 项目：
 - 在 IntelliJ IDEA 中：选择 Import。导航到下载的程序包的根目录中的 `pom.xml` 文件。

- 在 Eclipse 中：选择 File、Import、Maven、Existing Maven Projects。然后，导航到 kinesis-video-java-demo 目录。

有关更多信息，请参阅您的 IDE 的相应文档。

3. Java 示例代码使用当前的 AWS 凭据。要使用不同的凭证配置文件，请在 DemoAppMain.java 中找到以下代码：

```
final KinesisVideoClient kinesisVideoClient = KinesisVideoJavaClientFactory
    .createKinesisVideoClient(
        Regions.US_WEST_2,
        AuthHelper.getSystemPropertiesCredentialsProvider());
```

将该代码更改为以下内容：

```
final KinesisVideoClient kinesisVideoClient = KinesisVideoJavaClientFactory
    .createKinesisVideoClient(
        Regions.US_WEST_2,
        new ProfileCredentialsProvider("credentials-profile-name"));
```

有关更多信息，请参阅 AWS SDK for Java 参考中的 [ProfileCredentialsProvider](#)。

下一个步骤

[the section called “第 2 步：编写并检查代码”](#)

第 2 步：编写并检查代码

在 [Java Producer 库过程](#) 的这一部分中，您将编写并检查在上一节中下载的 Java 示例代码。

Java 测试应用程序 ([DemoAppMain](#)) 会显示以下编码模式：

- 创建 KinesisVideoClient 的实例。
- 创建 MediaSource 的实例。
- 将 MediaSource 注册到客户端。
- 开始流式处理。启动 MediaSource，它就会开始向客户端发送数据。

以下各节提供了详细信息。

创建的实例 KinesisVideoClient

您可以通过调用 `KinesisVideoClient` 操作来创建 `createKinesisVideoClient` 对象。

```
final KinesisVideoClient kinesisVideoClient = KinesisVideoJavaClientFactory
    .createKinesisVideoClient(
        Regions.US_WEST_2,
        AuthHelper.getSystemPropertiesCredentialsProvider());
```

`KinesisVideoClient` 需要凭证以进行身份验证，才能进行网络调用。您将传入一个 `SystemPropertiesCredentialsProvider` 实例，它会读取凭证文件中默认配置文件的 `AWSCredentials`：

```
[default]
aws_access_key_id = ABCDEFGHIJKLMOPQRSTU
aws_secret_access_key = AbCd1234EfGh5678IjKl9012MnOp3456QrSt7890
```

创建的实例 MediaSource

要向 Kinesis 视频流发送字节，必须生成数据。Amazon Kinesis Video Streams 提供了 `MediaSource` 代表数据源的接口。

例如，Kinesis Video Streams Java 库提供了 `ImageFileMediaSource` 该接口的实现 `MediaSource`。该类仅从一系列媒体文件中读取数据，而不是 Kinesis 视频流，但你可以用它来测试代码。

```
final MediaSource bytesMediaSource = createImageFileMediaSource();
```

MediaSource 向客户端注册

将您创建的媒体源注册到 `KinesisVideoClient`，使其能够识别该客户端（并且可向客户端发送数据）。

```
kinesisVideoClient.registerMediaSource(mediaSource);
```

启动媒体源

启动媒体源，使其可以开始生成数据并将其发送到客户端。

```
bytesMediaSource.start();
```

后续步骤

[the section called “步骤 3：运行并验证代码”](#)

步骤 3：运行并验证代码

要运行 Java Producer [库的 Java](#) 测试工具，请执行以下操作。

1. 选择 DemoAppMain。
2. 选择“运行”，“运行 DemoAppMain”。
3. 将您的凭证添加到此应用程序的 JVM 自变量：
 - 对于非临时 AWS 证书：“-Daws.accessKeyId={YourAwsAccessKey} -Daws.secretKey={YourAwsSecretKey} -Djava.library.path={NativeLibraryPath}”
 - 对于临时 AWS 证书：“-Daws.accessKeyId={YourAwsAccessKey} -Daws.secretKey={YourAwsSecretKey} -Daws.sessionToken={YourAwsSessionToken} -Djava.library.path={NativeLibraryPath}”
4. 登录 AWS Management Console 并打开 [Kinesis Video Streams](#) 控制台。

在 Manage Streams 页面中选择您的流。

5. 示例视频将在嵌入式播放器中播放。可能需要等待一小段积累帧的时间 (标准带宽和处理器条件下最多十秒)，视频才会出现。

该代码示例会创建一个流。代码中的 MediaSource 启动时，它会开始将示例帧发送到 KinesisVideoClient。然后，客户端会将数据发送到您的 Kinesis 视频流。

使用 Android 创建者库

您可以使用 Amazon Kinesis Video Streams 提供的安卓制作器库来编写应用程序代码，只需最少的配置，即可将媒体数据从安卓设备发送到 Kinesis 视频流。

执行以下步骤将您的代码与 Kinesis Video Streams 集成，以便您的应用程序可以开始将数据流式传输到您的 Kinesis 视频流：

1. 创建 KinesisVideoClient 对象的实例。
2. 通过提供媒体源信息创建 MediaSource 对象。例如，当创建摄像头媒体源时，您需要提供相应信息，例如，识别摄像头并指定摄像头所用编码方面的信息。

如果要开始流式处理，您必须创建自定义媒体源。

过程：使用 Android 创建者开发工具包

此过程演示如何在安卓应用程序中使用 Kinesis Video Streams Android Producer 客户端向你的 Kinesis 视频流发送数据。

该过程包括以下步骤：

- [the section called “先决条件”](#)
- [the section called “步骤 1：下载并配置代码”](#)
- [the section called “步骤 2：检查代码”](#)
- [the section called “步骤 3：运行并验证代码”](#)

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/](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。

步骤 1：下载并配置 Android 制作器库代码

在 Android 创建者库过程的这一部分中，需要下载 Android 示例代码并在 Android Studio 中打开该项目。

有关此示例的先决条件和其他详细信息，请参阅[使用 Android 创建者库](#)。

1. 创建一个目录，然后 AWS Mobile SDK for Android 从 GitHub 存储库中克隆该目录。

```
$ git clone https://github.com/aws-labs/aws-sdk-android-samples
```

2. 打开 [Android Studio](#)。
3. 在起始屏幕中，选择 Open an existing Android Studio project。
4. 导航到 aws-sdk-android-samples/AmazonKinesisVideoDemoApp 目录，选择 OK。
5. 打开 AmazonKinesisVideoDemoApp/src/main/res/raw/awsconfiguration.json 文件。

在 CredentialsProvider 节点中，提供 [先决条件](#) 部分中设置身份池过程中的身份池 ID，并提供您的 AWS 区域（例如 **us-west-2**）。

在 CognitoUserPool 节点中，提供 [先决条件](#) 部分中设置用户池过程中的应用程序客户端密钥、应用程序客户端 ID 和池 ID，并提供您的 AWS 区域（例如 **us-west-2**）。

6. awsconfiguration.json 文件类似如下：

```
{
  "Version": "1.0",
  "CredentialsProvider": {
    "CognitoIdentity": {
      "Default": {
        "PoolId": "us-west-2:01234567-89ab-cdef-0123-456789abcdef",
        "Region": "us-west-2"
      }
    }
  },
  "IdentityManager": {
    "Default": {}
  },
  "CognitoUserPool": {
    "Default": {
      "AppClientSecret": "abcdefghijklmnopqrstuvwxyz0123456789abcdefghijklmnop",
      "AppClientId": "0123456789abcdefghijklmnop",
      "PoolId": "us-west-2_qRsTuVwXy",
      "Region": "us-west-2"
    }
  }
}
```


7. AmazonKinesisVideoDemoApp/src/main/java/com/amazonaws/kinesisvideo/demoapp/KinesisVideoDemoApp.java使用您的地区更新（在以下示例中，将其设置为 US_WEST_2）：

```
public class KinesisVideoDemoApp extends Application {
    public static final String TAG = KinesisVideoDemoApp.class.getSimpleName();
    public static Regions KINESIS_VIDEO_REGION = Regions.US_WEST_2;
```

有关 AWS 区域 常量的信息，请参阅[区域](#)。

后续步骤

[the section called “步骤 2：检查代码”](#)

步骤 2：检查代码

在 [Android 创建者库过程](#)的这一部分中，需要检查示例代码。

Android 测试应用程序 (AmazonKinesisVideoDemoApp) 显示以下编码模式：

- 创建 KinesisVideoClient 的实例。
- 创建 MediaSource 的实例。
- 开始流式处理。启动MediaSource，它就会开始向客户端发送数据。

以下各节提供了详细信息。

创建的实例 KinesisVideoClient

您可以通过调用 [KinesisVideoClient](#) 操作来创建 [createKinesisVideoClient](#) 对象。

```
mKinesisVideoClient = KinesisVideoAndroidClientFactory.createKinesisVideoClient(
    getActivity(),
    KinesisVideoDemoApp.KINESIS_VIDEO_REGION,
    KinesisVideoDemoApp.getCredentialsProvider());
```

KinesisVideoClient 需要凭证以进行身份验证，才能进行网络调用。您传入了一个实例AWSCredentialsProvider，该实例会从您在上一节中修改awsconfiguration.json的文件中读取您的 Amazon Cognito 证书。

创建的实例 MediaSource

要向 Kinesis 视频流发送字节，必须生成数据。Amazon Kinesis Video Streams 提供了 [MediaSource](#) 代表数据源的接口。

例如，Kinesis Video Streams 安卓库提供了 [AndroidCameraMediaSource](#) 该接口的实现MediaSource。此类从设备的某个摄像头读取数据。

下面的代码示例 (摘自 [fragment/StreamConfigurationFragment.java](#) 文件) 创建媒体源配置：

```
private AndroidCameraMediaSourceConfiguration getCurrentConfiguration() {
    return new AndroidCameraMediaSourceConfiguration(
        AndroidCameraMediaSourceConfiguration.builder()
            .withCameraId(mCamerasDropdown.getSelectedItem().getCameraId())

            .withEncodingMimeType(mMimeTypeDropdown.getSelectedItem().getMimeType())

            .withHorizontalResolution(mResolutionDropdown.getSelectedItem().getWidth())

            .withVerticalResolution(mResolutionDropdown.getSelectedItem().getHeight())
                .withCameraFacing(mCamerasDropdown.getSelectedItem().getCameraFacing())
                .withIsEncoderHardwareAccelerated(

mCamerasDropdown.getSelectedItem().isEncoderHardwareAccelerated())
                .withFrameRate(FRAMERATE_20)
                .withRetentionPeriodInHours(RETENTION_PERIOD_48_HOURS)
                .withEncodingBitRate(BITRATE_384_KBPS)
                .withCameraOrientation(-
mCamerasDropdown.getSelectedItem().getCameraOrientation())

            .withNalAdaptationFlags(StreamInfo.NalAdaptationFlags.NAL_ADAPTATION_ANNEXB_CPD_AND_FRAME_NALS)
                .withIsAbsoluteTimecode(false));
}
```

下面的代码示例 (摘自 [fragment/StreamingFragment.java](#) 文件) 创建媒体源：

```
mCameraMediaSource = (AndroidCameraMediaSource) mKinesisVideoClient
```

```
.createMediaSource(mStreamName, mConfiguration);
```

启动媒体源

启动媒体源，以便开始生成数据并将数据发送到客户端。下面的代码示例摘自 [fragment/StreamingFragment.java](#) 文件：

```
mCameraMediaSource.start();
```

后续步骤

[the section called “步骤 3：运行并验证代码”](#)

步骤 3：运行并验证代码

要运行 [Android 创建者库](#) 的 Android 示例应用程序，请执行以下操作。

1. 连接 Android 设备。
2. 依次选择 Run、Run...、Edit configurations...。
3. 选择加号图标 (+)，安卓应用程序。在名称字段中，输入 **AmazonKinesisVideoDemoApp**。在“模块”下拉列表中，选择 AmazonKinesisVideoDemoApp。选择确定。
4. 选择 Run、Run。
5. 在 Select Deployment Target 屏幕中，选择连接的设备，然后选择 OK。
6. 在设备上的 AWSKinesisVideoDemoApp 应用程序中，选择创建新帐户。
7. 为 USERNAME、Password、Given name、Email address 和 Phone number 输入值，然后选择 Sign up。

Note

这些值具有以下约束：

- 密码：必须包含大小写字母、数字和特殊字符。您可以在 [Amazon Cognito](#) 控制台的用户池页面中更改这些限制。
- 电子邮件地址：必须是有效地址，您才能收到确认码。
- Phone number (电话号码)：必须采用以下格式：**+<Country code><Number>**，例如 **+12065551212**。

8. 输入您通过电子邮件收到的验证码，然后选择确认。选择确定。
9. 在下一页上，保留默认值，然后选择直播。
10. 登录 AWS Management Console 并打开美国西部（俄勒冈）地区的 [Kinesis Video Streams](#) 控制台。

在 Manage Streams 页面上，选择 demo-stream。

11. 流视频将在嵌入式播放器中播放。可能需要等待一小段积累帧的时间（标准带宽和处理器条件下最多十秒），视频才会出现。

Note

如果设备的屏幕发生旋转（例如，从纵向到横向），则应用程序会停止流视频。

该代码示例会创建一个流。代码中的 `MediaSource` 启动后，就开始将帧从摄像头发送到 `KinesisVideoClient`。然后，客户端将数据发送到名为 `demo-stream` 的 Kinesis 视频流。

使用 C++ 创建者库

您可以使用 Amazon Kinesis Video Streams 提供的 C++ 制作者库来编写应用程序代码，将媒体数据从设备发送到 Kinesis 视频流。

物体模型

C++ 库提供以下对象来管理向 Kinesis 视频流发送数据：

- `KinesisVideoProducer`：包含有关您的媒体来源和 AWS 凭证的信息，并维护回传以报告 Kinesis Video Streams 事件。
- `KinesisVideoStream`：代表 Kinesis 视频流。包含有关视频流参数的信息，例如名称、数据保留期和媒体内容类型。

将媒体放到直播中

您可以使用 C++ 库提供的方法（例如 `PutFrame`）将数据放入 `KinesisVideoStream` 对象。随后，该库将管理数据的内部状态，这可包含以下任务：

- 执行身份验证。

- 监视网络延迟。如果延迟太高，库可能会选择丢弃帧。
- 跟踪正在进行的流式处理的状态。

回调接口

此层公开一组回调接口，使其能够与应用程序层进行通信。这些回调接口包括：

- 服务回调接口 (CallbackProvider)：该库在创建流、获取流描述和删除流时调用通过此接口获得的事件。
- 客户端就绪状态或存储不足事件接口 (ClientCallbackProvider)：当客户端准备就绪或检测到可用存储空间或内存不足时，该库将对此接口调用事件。
- 流事件回调接口 (StreamCallbackProvider)：当发生流事件 (例如，进入准备就绪状态的流、丢弃的帧或流错误) 时，该库将对此接口调用事件。

Kinesis Video Streams 为这些接口提供默认实现。您也可以提供自己的自定义实现，例如，如果您需要自定义网络逻辑或想要向用户界面公开存储空间不足的情况。

有关创建者库中的回调的更多信息，请参阅[制作人 SDK 回调](#)。

过程：使用 C++ 创建者开发工具包

此过程演示如何在 C++ 应用程序中使用 Kinesis Video Streams 客户端和媒体源向 Kinesis 视频流发送数据。

该过程包括以下步骤：

- [步骤 1：下载并配置代码](#)
- [步骤 2：编写和检查代码](#)
- [步骤 3：运行和验证代码](#)

先决条件

- 凭证：在示例代码中，您可以通过指定在证书配置文件中设置的配置文件来提供AWS证书。如果尚未执行此操作，请先设置凭证配置文件。

有关更多信息，请参阅[设置AWS证书和开发区域](#)。

- 证书存储集成：Kinesis Video Streams 制作者库必须与其调用的服务建立信任。这是通过验证公共证书存储中的证书颁发机构 (CA) 来完成的。对于基于 Linux 的模型，此存储位于 `/etc/ssl/` 目录中。

从以下位置将证书下载到您的证书存储：

<https://www.amazontrust.com/repository/SFSRootCAG2.pem>

- 为 macOS 安装以下构建依赖项：
 - [Autoconf 2.69](#) (许可证 GPLv3+/Autoconf：GNU GPL 版本 3 或更高版本)
 - [CMake 3.7 或 3.8](#)
 - [Pkg-Config](#)
 - [Flex 2.5.35 Apple \(flex-31\) 或更高版本](#)
 - [Bison 2.4](#) (GNU 许可证)
 - [Automake 1.15.1](#) (GNU 许可证)
 - GNU Libtool (Apple Inc. 版本 cctools-898)
 - xCode (macOS) / clang / gcc (xcode-选择版本 2347)
 - Java 开发工具包 (JDK) (用于 Java JNI 编译)
 - [Lib-Pkg](#)
- 为 Ubuntu 安装以下构建依赖项 (对版本命令的响应已截断)：
 - 安装 Git：`sudo apt-get install git`

```
$ git --version
git version 2.14.1
```

- 安装 [CMake](#)：`sudo apt-get install cmake`

```
$ cmake --version
cmake version 3.9.1
```

- 安装 Libtool：`sudo apt-get install libtool`

```
2.4.6-2
```

- 安装 libtool-bin：`sudo apt-get install libtool-bin`

```
$ libtool --version
libtool (GNU libtool) 2.4.6
```

```
Written by Gordon Matzigkeit, 1996
```

- 安装 GNU Automake : `sudo apt-get install automake`

```
$ automake --version
automake (GNU automake) 1.15
```

- 安装 GNU Bison : `sudo apt-get install bison`

```
$ bison -V
bison (GNU Bison) 3.0.4
```

- 安装 G++ : `sudo apt-get install g++`

```
g++ --version
g++ (Ubuntu 7.2.0-8ubuntu3) 7.2.0
```

- 安装 curl : `sudo apt-get install curl`

```
$ curl --version
curl 7.55.1 (x86_64-pc-linux-gnu) libcurl/7.55.1 OpenSSL/1.0.2g zlib/1.2.11
libidn2/2.0.2 libpsl/0.18.0 (+libidn2/2.0.2) librtmp/2.3
```

- 安装 pkg-config : `sudo apt-get install pkg-config`

```
$ pkg-config --version
0.29.1
```

- 安装 Flex : `sudo apt-get install flex`

```
$ flex --version
flex 2.6.1
```

- 安装 OpenJDK : `sudo apt-get install openjdk-8-jdk`

```
$ java -version
openjdk version "1.8.0_171"
```

- 设置 JAVA_HOME 环境变量 : `export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/`
- 运行构建脚本 : `./install-script`

下一步

[步骤 1：下载并配置 C++ 创建者库代码](#)

步骤 1：下载并配置 C++ 生产人员库代码

有关如何下载和配置 C++ 制作者库的信息，请参阅 [Amazon Kinesis Video Streams CPP Producer、gStreamer 插件和 JNI](#)。

有关此示例的先决条件和更多信息，请参阅[使用 C++ 生产者库](#)。

下一步

[步骤 2：编写和检查代码](#)

步骤 2：编写并检查代码

在 [C++ 创建者库过程](#)的这一部分中，您需要在 C++ 测试框架中检查该代码（`tst/ProducerTestFixture.h` 和其他文件）。您在上一部分中已下载该代码。

平台独立的 C++ 示例演示了以下编码模式：

- 创建一个实例 `KinesisVideoProducer` 来访问 Kinesis Video Streams。
- 创建 `KinesisVideoStream` 的实例。AWS 账户如果尚不存在同名视频流，这将在您中创建 Kinesis 视频流。
- 对于每个数据帧，当其可用时，对 `KinesisVideoStream` 调用 `putFrame` 以将其发送到流。

以下章节提供有关这种编码模式的更多信息，请参阅以下章节的更多信息，请参阅以下章节

创建的实例 `KinesisVideoProducer`

您可以通过调用 `KinesisVideoProducer::createSync` 方法来创建 `KinesisVideoProducer` 对象。以下示例在 `ProducerTestFixture.h` 文件中创建 `KinesisVideoProducer`：

```
kinesis_video_producer_ = KinesisVideoProducer::createSync(move(device_provider_),
    move(client_callback_provider_),
    move(stream_callback_provider_),
    move(credential_provider_),
    defaultRegion_);
```


createSync 方法采用以下参数：

- 一个 DeviceInfoProvider 对象，此对象返回一个包含有关设备或存储配置的信息的 DeviceInfo 对象。

Note

您可以使用 deviceInfo.storageInfo.storageSize 参数配置内容存储大小。您的内容流共享内容存储。要确定存储大小要求，请将平均帧大小乘以为所有流存储最大持续时间的帧数。然后再乘以 1.2（考虑碎片整理）。例如，假设您的应用程序具有以下配置：

- 三个流
- 3 分钟的最大持续时间
- 每个流为 30 帧/秒 (FPS)
- 每个帧的大小为 10000 KB

此应用程序的内容存储要求为 $3 \text{ (直播)} * 3 \text{ (分钟)} * 60 \text{ (一分钟内的秒)} * 10000 \text{ (kb)} * 1.2 \text{ (碎片整理容量)} = 194.4 \text{ Mb} \sim 200 \text{ Mb}$ 。

- 一个 ClientCallbackProvider 对象，此对象返回报告客户端特定的事件的函数指针。
- 一个 StreamCallbackProvider 对象，此对象返回在发生流特定的事件时将回调的函数指针。
- 一个 CredentialProvider 对象，提供对 AWS 凭证环境变量的访问。
- AWS 区域（“us-west-2”）。从区域确定服务终端节点。

创建的实例 KinesisVideoStream

您可以通过调用带 StreamDefinition 参数的 KinesisVideoProducer::CreateStream 方法来创建 KinesisVideoStream 对象。该示例在 ProducerTestFixture.h 文件中创建 KinesisVideoStream，轨道类型为视频，轨道 ID 为 1：

```
auto stream_definition = make_unique<StreamDefinition>(stream_name,
                                                    hours(2),
                                                    tags,
                                                    "",
                                                    STREAMING_TYPE_REALTIME,
                                                    "video/h264",
                                                    milliseconds::zero(),
                                                    seconds(2),
                                                    milliseconds(1),
                                                    true,
```

```
        true,  
        true);  
return kinesis_video_producer_->createStream(move(stream_definition));
```

StreamDefinition 对象具有以下字段：

- 流名称。
- 数据保留期。
- 流的标记。使用者应用程序可使用这些标记来查找正确的流或获取有关流的更多信息。也可以在 AWS Management Console 中查看这些标记。
- 流的 AWS KMS 加密密钥。有关更多信息，请参阅对 [Kinesis Video Streams 使用的服务器端加密的服务器端加密中的使用服务器端加密的服务器端加密中的使用](#)
- 流式处理类型。目前唯一有效的值是 STREAMING_TYPE_REALTIME。
- 媒体内容类型。
- 媒体延迟。此值当前未使用，应设置为 0。
- 每个片段的播放持续时间。
- 媒体时间码标度。
- 媒体是否使用关键帧片段。
- 媒体是否使用时间码。
- 媒体是否使用绝对片段时间。

在 Kinesis 视频流中添加音轨

您可以使用以下的 addTrack 方法将音轨详细信息添加到视频轨道流定义中 StreamDefinition：

```
stream_definition->addTrack(DEFAULT_AUDIO_TRACKID, DEFAULT_AUDIO_TRACK_NAME,  
    DEFAULT_AUDIO_CODEC_ID, MKV_TRACK_INFO_TYPE_AUDIO);
```

addTrack 方法需要以下参数：

- 曲目 ID（作为音频 ID）。该值应为唯一的非零值。
- 用户定义的轨道名称（例如，音轨的“音频”）。
- 此轨道的编解码器 ID（例如，音轨“A_AAC”）。
- 曲目类型（例如，使用 MKV_TRACK_INFO_TYPE_AUDIO 的枚举值作为音频）。

如果有用于音轨的编解码器专用数据，可在调用 `addTrack` 函数时传递。您还可以在创建 `KinesisVideoStream` 对象后发送编解码器的私有数据，同时在中调用 `start` 方法。

在 Kinesis 视频流中存储 FRAES

您可以使用 `KinesisVideoStream::putFrame` 传入包含标题和媒体数据的 `Frame` 对象将媒体放入 Kinesis 视频流。此示例调用 `ProducerApiTest.cpp` 文件中的 `putFrame`：

```
frame.duration = FRAME_DURATION_IN_MICROS * HUNDREDS_OF_NANOS_IN_A_MICROSECOND;
frame.size = sizeof(frameBuffer_);
frame.frameData = frameBuffer_;
memset(frame.frameData, 0x55, frame.size);

while (!stop_producer_) {
    // Produce frames
    timestamp = std::chrono::duration_cast<std::chrono::nanoseconds>(
        std::chrono::system_clock::now().time_since_epoch()).count() /
    DEFAULT_TIME_UNIT_IN_NANOS;
    frame.index = index++;
    frame.decodingTs = timestamp;
    frame.presentationTs = timestamp;

    // Key frame every 50th
    frame.flags = (frame.index % 50 == 0) ? FRAME_FLAG_KEY_FRAME : FRAME_FLAG_NONE;
    ...

    EXPECT_TRUE(kinesis_video_stream->putFrame(frame));
}
```

Note

上一个 C++ 创建者示例发送测试数据缓冲区。在实际应用中，您应从媒体源（例如摄像机）的帧数据中获取帧缓冲区和大小。

`Frame` 对象具有以下字段：

- 帧索引。这应是一个单调递增的值。
- 与帧关联的标记。例如，如果编码器已配置为生成关键帧，则将为此帧分配 `FRAME_FLAG_KEY_FRAME` 标记。
- 解码时间戳。

- 演示时间戳。
- 帧的持续时间 (最多 100 ns)。
- 帧大小 (以字节为单位)。
- 帧数据。

有关帧格式的更多信息，请参见 [Kinesis Video Streams 数据模型](#)。

将 a 放KinesisVideoFrame入特定的曲目中 KinesisVideoStream

您可以使用该PutFrameHelper类将帧数据放入特定的轨道中。首先，调用 getFrameData Buffer 以获取指向其中一个预先分配的缓冲区的指针，以填充数据。KinesisVideoFrame然后，您可以调用 putFrameMulti Track 将与布尔值KinesisVideoFrame一起发送，以指示帧数据的类型。如果是视频数据，请使用 true；如果帧包含音频数据，则使用 false。putFrameMultiTrack 方法使用排队机制来确保 MKV 片段保持单调递增的帧时间戳，并且任意两个片段不重叠。例如，片段第一帧的 MKV 时间戳应始终大于前一片段最后一帧的 MKV 时间戳。

PutFrameHelper具有以下字段：

- 队列中的最大音频帧数。
- 队列中的最大视频帧数。
- 为单个音频帧分配的大小。
- 为单个视频帧分配的大小。

指标和指标日志

C++ 创建者开发工具包包括指标和指标日志记录的功能。

您可以使用getKinesisVideoMetrics和 getKinesisVideoStreamMetrics API 操作来检索有关 Kinesis Video Streams 和您的活跃直播的信息。

以下代码来自 kinesis-video-pic/src/client/include/com/amazonaws/kinesis/video/client/Include.h 文件。

```
/**
 * Gets information about the storage availability.
 *
 * @param 1 CLIENT_HANDLE - the client object handle.
```

```
* @param 2 PKinesisVideoMetrics - OUT - Kinesis Video metrics to be filled.
*
* @return Status of the function call.
*/
PUBLIC_API STATUS getKinesisVideoMetrics(CLIENT_HANDLE, PKinesisVideoMetrics);

/**
* Gets information about the stream content view.
*
* @param 1 STREAM_HANDLE - the stream object handle.
* @param 2 PStreamMetrics - Stream metrics to fill.
*
* @return Status of the function call.
*/
PUBLIC_API STATUS getKinesisVideoStreamMetrics(STREAM_HANDLE, PStreamMetrics);
```

`getKinesisVideoMetrics` 填入的 `PClientMetrics` 对象包含以下信息：

- `contentStoreSize`：内容存储（用于存储流数据的内存）的总大小（以字节为单位）。
- `contentStoreAvailable`大小：内容存储中的可用内存，以字节为单位。
- `contentStoreAllocated`大小：内容存储中分配的内存。
- `totalContentViews`大小：用于内容视图的总内存。内容视图是内容存储中的一系列信息索引。
- `totalFrameRate`：所有活动流中每秒的总帧数。
- `totalTransferRate`：所有数据流中发送的每秒总比特数 (bps)。

`getKinesisVideoStreamMetrics` 填入的 `PStreamMetrics` 对象包含以下信息：

- `currentViewDuration`：内容视图的头部（帧编码时）和当前位置（帧数据发送到 Kinesis Video Streams 时）之间的差值，以 100 纳秒为单位。
- `overallViewDuration`：内容视图的头部（对帧进行编码时）与尾部（当帧从内存中刷新时，要么是因为超过了为内容视图分配的总空间，要么是因为从 Kinesis Video Streams 接收到 `PersistedAck` 消息，并且已知存在的帧被刷新）之间的差异（以 100 纳秒为单位）。
- `currentViewSize`：从头部（对帧进行编码时）到当前位置（帧发送到 Kinesis Video Streams 时）的内容视图的大小（以字节为单位）。
- `overallViewSize`：内容视图的总大小（以字节为单位）。
- `currentFrameRate`：上次测量的流速率，以每秒帧数为单位。

- `currentTransferRate` : 上次测量的流速率，以每秒字节数为单位。

分解

如果要发送缓冲区中的剩余字节并等待 ACK，您可以使用 `stopSync` :

```
kinesis_video_stream->stopSync();
```

或者，您可以调用 `stop` 来结束流式传输：

```
kinesis_video_stream->stop();
```

停止流式传输后，您可以通过调用以下 API 释放流：

```
kinesis_video_producer_->freeStream(kinesis_video_stream);
```

下一步

[the section called “步骤 3：运行并验证代码”](#)

步骤 3：运行并验证代码

要运行和验证 [C++ 生产者库过程](#)的代码，请参阅以下特定于操作系统的说明：

- [Linux](#)
- [macOS](#)
- [Windows](#)
- [Raspberry](#)

您可以通过在亚马逊CloudWatch控制台中查看与您的直播相关的指标来监控直播流量，例如 `PutMedia.IncomingBytes`。

使用 C++ Produce SED SEDER SEDER

[GStreamer](#) 是一个流行的媒体框架，多个摄像机和视频源使用它通过组合模块化插件来创建自定义媒体管道。Kinesis Video Streams gStreamer 插件简化了现有 GStreamer 媒体管道与 Kinesis Video Streams 的集成。

有关使用 C++ 创建者开发工具包作为 GStreamer 插件的信息，请参阅[示例：Kinesis Video Streams Producer SDK gStreamer 插件](#)。

在 Docker 容器中使用 C++ Producer SDK 作为 gStreamer 插件

[GStreamer](#) 是一个流行的媒体框架，多个摄像机和视频源使用它通过组合模块化插件来创建自定义媒体管道。Kinesis Video Streams gStreamer 插件简化了现有 GStreamer 媒体管道与 Kinesis Video Streams 的集成。

此外，使用 [Docker](#) 创建 gStreamer 流水线可以标准化 Kinesis Video Streams 的操作环境，从而简化应用程序的构建和运行。

有关使用 C++ 创建者开发工具包作为 Docker 容器中的 GStreamer 插件的信息，请参阅[在 Docker 容器中运行 gStreamer 元素](#)。

在 C++ 制作者 SDK 中使用日志记录

您可以在 `kvs_log_configuration` 文件夹的 `kinesis-video-native-build` 文件中为 C++ 创建者开发工具包应用程序配置日志记录。

以下示例显示了默认配置文件的第一行，它会将应用程序配置为将 DEBUG 级日志条目写入到 AWS Management Console：

```
log4cplus.rootLogger=DEBUG, KvsConsoleAppender
```

您可以将日志记录级别设置为 INFO 以减少详细日志记录。

要将应用程序配置为将日志条目写入日志文件，请将文件中的第一行更新为以下内容：

```
log4cplus.rootLogger=DEBUG, KvsConsoleAppender, KvsFileAppender
```

这会将应用程序配置为将日志条目写入到 `kvs.log` 文件夹中的 `kinesis-video-native-build/log`。

要更改日志文件位置，请使用新路径更新以下行：

```
log4cplus.appender.KvsFileAppender.File=../Log/kvs.Log
```

Note

如果 DEBUG 级日志记录被写入到某个文件，日志文件可能快速用尽设备上的可用存储空间。

使用 C 创建者库

您可以使用 C Producer 库提供的 Amazon Kinesis Video Streams 编写应用程序代码，将媒体数据从设备发送到 Kinesis 视频流。

物体模型

Kinesis Video Streams C Producer Library 基于一个名为平台独立代码库 (PIC) 的通用组件，该组件可 GitHub 在 <https://github.com/awslabs/amazon-kinesis-video-streams-pic/> 上找到。PIC 包含基础组件的独立于平台的业务逻辑。Kinesis Video Streams C Producer Library 为 PIC 添加了额外的 API 层，允许特定场景和平台的回调和事件。Kinesis Video Streams C Producer 库在 PIC 之上构建了以下组件：

- 设备信息提供者-公开可以直接提供给 PIC API 的 DeviceInfo 结构。您可以配置一组提供商，包括应用程序场景优化提供商，这些提供程序可以根据应用程序处理的流数量和类型以及根据可用 RAM 量配置的所需缓冲量来优化内容存储。
- 流信息提供者-公开可以直接提供给 PIC API 的 StreamInfo 结构。有一组特定于应用程序类型和常见流媒体场景类型的提供商。其中包括视频、音频、音频和视频多轨等提供商。这些场景中的每一个都有默认值，您可以根据应用程序的要求进行自定义。
- 回调提供者-公开可以直接提供给 PIC API 的 ClientCallbacks 结构。这包括一组用于联网（基于 curl 的 API 回调）、授权（AWS 凭证 API）和错误时重试流式传输的回调提供商。回调提供商 API 需要配置许多参数，例如 AWS 区域和授权信息。这可以通过使用 IoT 证书或使用 AWSAccessKeyIdSecretKey、或来完成 SessionToken。如果您的应用程序需要进一步处理特定回调以实现某些应用程序特定的逻辑，则可以通过自定义回调来增强回调提供程序。
- FrameOrderCoordinator— 帮助处理多轨场景的音频和视频同步。它具有默认行为，您可以对其进行自定义以处理应用程序的特定逻辑。它还简化了 PIC Frame 结构中的帧元数据打包，然后将其提交到下层 PIC API。对于非多轨道场景，此组件直接传递到 PIC putFrame API。

C 库提供以下对象来管理向 Kinesis 视频流发送数据的过程：

- KinesisVideoClient— 包含有关您的设备的信息，并维护回传以报告 Kinesis Video Streams 事件。
- KinesisVideoStream— 表示有关视频流参数的信息，例如名称、数据保留期和媒体内容类型。

将媒体放到直播中

您可以使用 C 库提供的方法（例如 PutKinesisVideoFrame）将数据放入 KinesisVideoStream 对象。随后，该库将管理数据的内部状态，这可包含以下任务：

- 执行身份验证。
- 监视网络延迟。如果延迟太高，库可能会选择丢弃帧。
- 跟踪正在进行的流式处理的状态。

过程：使用 C 创建者开发工具包

此过程演示如何在 C 应用程序中使用 Kinesis Video Streams 客户端和媒体源将 H.264 编码的视频帧发送到您的 Kinesis 视频流。

该过程包括以下步骤：

- [步骤 1：下载 CI Producer 库代码](#)
- [步骤 2：编写和检查代码](#)
- [第 3 步：运行代码](#)

先决条件

- 证书-在示例代码中，您可以通过指定在证书配置文件中设置的配置文件来提供 AWS 凭证。如果尚未执行此操作，请先设置凭证配置文件。

有关更多信息，请参阅[设置 AWS 证书和开发区域](#)。

- 证书存储集成 — Kinesis Video Streams 制作者库必须与其调用的服务建立信任。这是通过验证公共证书存储中的证书颁发机构 (CA) 来完成的。对于基于 Linux 的模型，此存储位于 /etc/ssl/ 目录中。

从以下位置将证书下载到您的证书存储：

<https://www.amazontrust.com/repository/SFSRootCAG2.pem>

- 为 macOS 安装以下构建依赖项：
 - [Autoconf 2.69](#) (许可证 GPLv3+/Autoconf : GNU GPL 版本 3 或更高版本)
 - [CMake 3.7 或 3.8](#)
 - [Pkg-Config](#)
 - [Flex 2.5.35 Apple \(flex-31\) 或更高版本](#)
 - [Bison 2.4](#) (GNU 许可证)
 - [Automake 1.15.1](#) (GNU 许可证)
 - GNU Libtool (Apple Inc. 版本 cctools-898)
 - xCode (macOS) / clang / gcc (xcode-选择版本 2347)
 - Java 开发工具包 (JDK) (用于 Java JNI 编译)
 - [Lib-Pkg](#)
- 为 Ubuntu 安装以下构建依赖项 (对版本命令的响应已截断)：
 - 安装 Git : `sudo apt-get install git`

```
$ git --version
git version 2.14.1
```

- 安装 [CMake](#) : `sudo apt-get install cmake`

```
$ cmake --version
cmake version 3.9.1
```

- 安装 Libtool : `sudo apt-get install libtool`

```
2.4.6-2
```

- 安装 libtool-bin : `sudo apt-get install libtool-bin`

```
$ libtool --version
libtool (GNU libtool) 2.4.6
Written by Gordon Matzigkeit, 1996
```

- 安装 GNU Automake : `sudo apt-get install automake`

```
automake (GNU automake) 1.15
```

- 安装 GNU Bison : `sudo apt-get install bison`

```
$ bison -V
bison (GNU Bison) 3.0.4
```

- 安装 G++ : `sudo apt-get install g++`

```
g++ --version
g++ (Ubuntu 7.2.0-8ubuntu3) 7.2.0
```

- 安装 curl : `sudo apt-get install curl`

```
$ curl --version
curl 7.55.1 (x86_64-pc-linux-gnu) libcurl/7.55.1 OpenSSL/1.0.2g zlib/1.2.11
libidn2/2.0.2 libpsl/0.18.0 (+libidn2/2.0.2) librtmp/2.3
```

- 安装 pkg-config : `sudo apt-get install pkg-config`

```
$ pkg-config --version
0.29.1
```

- 安装 Flex : `sudo apt-get install flex`

```
$ flex --version
flex 2.6.1
```

- 安装 OpenJDK : `sudo apt-get install openjdk-8-jdk`

```
$ java -version
openjdk version "1.8.0_171"
```

- 设置 JAVA_HOME 环境变量 : `export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/`
- 运行构建脚本 : `./install-script`

下一步

[步骤 1 : 下载 CI Producer 库代码](#)

步骤 1：下载 C Producer 库代码

在本节中，您将下载低级别库。有关此示例的先决条件和其他详细信息，请参阅[使用 C 创建者库](#)。

1. 创建一个目录，然后从GitHub存储库克隆示例源代码。

```
git clone --recursive https://github.com/aws-labs/amazon-kinesis-video-streams-producer-c.git
```

Note

如果您错过使用 `--recursive` 运行 `git` 克隆，请在 `amazon-kinesis-video-streams-producer-c/open-source` 目录中运行 `git submodule update --init`。你还必须安装 `pkg-config`、`automake`、`CMake` 和构建环境。欲了解更多信息，请参阅 <https://github.com/aws-labs/amazon-kinesis-video-streams-producer-README.md> `c.git` 中的。

2. 在您选定的集成开发环境 (IDE) (例如 [Eclipse](#)) 中打开您的代码。

下一步

[步骤 2：编写和检查代码](#)

步骤 2：编写和检查代码

在本节中，您将在 <https://github.com/aws-labs/amazon-kinesis-video-streams-producer-KvsVideoOnlyStreamingSample.c> 存储 `samples` 库的文件夹中检查示例应用程序的代码。GitHub 您在上一步中已下载该代码。此示例演示如何使用 C Producer 库将文件夹内的 H.264 编码视频帧发送 `samples/h264SampleFrames` 到您的 Kinesis 视频流。

此示例应用程序包含三个部分：

- 初始化和配置：
 - 初始化和配置特定于平台的媒体管道。
 - `KinesisVideoStream` 为管道初始化和配置 `KinesisVideoClient` 和，设置回调，集成特定场景身份验证，提取和提交编解码器私有数据，并将数据流设置为 `READY` 状态。

- 主循环：
 - 使用时间戳和标志从媒体管道获取帧。
 - 将框架提交给KinesisVideoStream。
- 分解：
 - 停止 (同步) KinesisVideoStream、释放KinesisVideoStream、释KinesisVideoClient放。

此示例应用程序完成以下任务：

- 调用 createDefaultDeviceInfo API 来创建 deviceInfo 对象，其中包含有关设备或存储配置的信息。

```
// default storage size is 128MB. Use setDeviceInfoStorageSize after create to change
storage size.
CHK_STATUS(createDefaultDeviceInfo(&pDeviceInfo));
// adjust members of pDeviceInfo here if needed
pDeviceInfo->clientInfo.loggerLogLevel = LOG_LEVEL_DEBUG;
```

- 调用 createRealtimeVideoStreamInfoProvider API 以创建 StreamInfo 对象。

```
CHK_STATUS(createRealtimeVideoStreamInfoProvider(streamName,
DEFAULT_RETENTION_PERIOD, DEFAULT_BUFFER_DURATION, &pStreamInfo));
// adjust members of pStreamInfo here if needed
```

- 调用 createDefaultCallbacksProviderWithAwsCredentials API 以基于静态AWS凭证创建默认回调提供商。

```
CHK_STATUS(createDefaultCallbacksProviderWithAwsCredentials(accessKey,
                                                             secretKey,
                                                             sessionToken,
                                                             MAX_UINT64,
                                                             region,
                                                             cacertPath,
                                                             NULL,
                                                             NULL,
                                                             FALSE,
```

```
&pClientCallbacks));
```

- 调用 `createKinesisVideoClient` API 创建包含设备存储相关信息的 `KinesisVideoClient` 对象，并维护回调以报告 Kinesis Video Streams 事件。

```
CHK_STATUS(createKinesisVideoClient(pDeviceInfo, pClientCallbacks, &clientHandle));
```

- 调用 `createKinesisVideoStreamSync` API 以创建 `KinesisVideoStream` 对象。

```
CHK_STATUS(createKinesisVideoStreamSync(clientHandle, pStreamInfo, &streamHandle));
```

- 设置示例帧并调用 `PutKinesisVideoFrame` API 将该帧发送到 `KinesisVideoStream` 对象。

```
// setup sample frame
MEMSET(frameBuffer, 0x00, frameSize);
frame.frameData = frameBuffer;
frame.version = FRAME_CURRENT_VERSION;
frame.trackId = DEFAULT_VIDEO_TRACK_ID;
frame.duration = HUNDREDS_OF_NANOS_IN_A_SECOND / DEFAULT_FPS_VALUE;
frame.decodingTs = defaultGetTime(); // current time
frame.presentationTs = frame.decodingTs;

while(defaultGetTime() > streamStopTime) {
    frame.index = frameIndex;
    frame.flags = fileIndex % DEFAULT_KEY_FRAME_INTERVAL == 0 ?
FRAME_FLAG_KEY_FRAME : FRAME_FLAG_NONE;
    frame.size = sizeof(frameBuffer);

    CHK_STATUS(readFrameData(&frame, frameFilePath));

    CHK_STATUS(putKinesisVideoFrame(streamHandle, &frame));
    defaultThreadSleep(frame.duration);

    frame.decodingTs += frame.duration;
    frame.presentationTs = frame.decodingTs;
    frameIndex++;
    fileIndex++;
}
```

```
    fileIndex = fileIndex % NUMBER_OF_FRAME_FILES;  
}
```

- 分解：

```
CHK_STATUS(stopKinesisVideoStreamSync(streamHandle));  
CHK_STATUS(freeKinesisVideoStream(&streamHandle));  
CHK_STATUS(freeKinesisVideoClient(&clientHandle));
```

下一步

[第 3 步：运行代码](#)

第 3 步：运行代码

要运行并验证 [C 创建者库过程](#) 的代码，请执行以下操作：

1. 运行以下命令在[下载的 C SDK](#) 中创建一个build目录，然后cmake从中启动：

```
mkdir -p amazon-kinesis-video-streams-producer-c/build;  
cd amazon-kinesis-video-streams-producer-c/build;  
cmake ..
```

您可以将以下选项传递给 `cmake ..`

- `-DBUILD_DEPENDENCIES`-是否从源代码构建依赖库。
- `-DBUILD_TEST=TRUE`-构建单元和集成测试。可能有助于确认对您的设备的支持。

```
./tst/webrtc_client_test
```

- `-DCODE_COVERAGE`-启用覆盖范围报告。
- `-DCOMPILER_WARNINGS`-启用所有编译器警告。
- `-DADDRESS_SANITIZER`-用... 构建AddressSanitizer。
- `-DMEMORY_SANITIZER`-用... 构建MemorySanitizer。

- `-DTHREAD_SANITIZER`-用... 构建ThreadSanitizer。
 - `-DUNDEFINED_BEHAVIOR_SANITIZER`-用... 构建UndefinedBehaviorSanitizer。
 - `-DALIGNED_MEMORY_MODEL` -为仅对齐内存模型的设备构建。默认为 OFF。
2. 导航到您刚刚在上一步中创建的build目录，然后运行make以构建 WebRTC C SDK 及其提供的示例。

```
make
```

3. 示例应用程序kinesis_video_cproducer_video_only_sample将文件夹内的 h.264 编码视频帧发送samples/h264SampleFrames到 Kinesis Video Streams。以下命令将视频帧循环发送到 Kinesis Video Streams，持续十秒钟：

```
./kinesis_video_cproducer_video_only_sample YourStreamName 10
```

如果要从另一个文件夹（例如MyH264FramesFolder）发送 H.264 编码的帧，请使用以下参数运行示例：

```
./kinesis_video_cproducer_video_only_sample YourStreamName 10 MyH264FramesFolder
```

4. 若要启用详细日志，请定义 `HEAP_DEBUG` 并通过取消 `CMakeList.txt` 中相应行的注释进行 `LOG_STREAMING C` 定义。

您可以在 IDE 中的调试输出中监控测试套件的进度。您还可以通过在亚马逊CloudWatch控制台中查看与您的直播相关的指标来监控直播流量，例如PutMedia.IncomingBytes。

Note

由于测试框架仅发送空字节的帧，因此，控制台不会将数据显示为视频流。

在 Raspberry Pi 上使用 C++ 创建者开发工具包

Raspberry Pi 是一个小型经济型计算机，可用于教授和学习基本计算机编程技能。本教程介绍如何在 Raspberry Pi 设备上设置和使用 Amazon Kinesis Video Streams C++ Producer SDK。这些步骤还包括如何使用 GStreamer 演示应用程序验证安装。

主题

- [先决条件](#)
- [创建有权写入 Kinesis Video Streams 的 IAM 用户](#)
- [加入你的 Raspberry Pi 到 Wi-Fi 网络](#)
- [远程连接到你的 Raspberry Pi](#)
- [配置树莓派摄像头](#)
- [安装软件先决条件](#)
- [下载并构建 Kinesis Video Streams C++ Producer SDK](#)
- [将视频流式传输到您的 Kinesis 视频流并观看直播](#)

先决条件

在您的 Raspberry Pi 上安装 C++ 创建者开发工具包之前，请确保满足以下先决条件：

- 使用以下配置的 Raspberry Pi 设备：
 - 面板版本：3 B 型号或更高版本。
 - 连接的摄像机模块。
 - 至少具有 8 GB 容量的 SD 卡。
 - 已安装 Raspbian 操作系统 (内核版本 4.9 或更高版本)。你可以从 Raspberry Pi [网站下载最新的 Raspberry Pi 操作系统 \(以前称为 Raspbian\) 镜像](#)。按照 Raspberry Pi 说明[在 SD 卡上安装下载的映像](#)。
- 还有一个 AWS 账户 Kinesis 视频流。有关更多信息，请参阅 [Kinesis Video Streams 入门](#)。

Note

默认情况下，C++ 制作器 SDK 使用美国西部 (俄勒冈 us-west-2) () 区域。要使用默认设置，请在美国西部 (俄勒冈) 地区 AWS 区域 创建您的 Kinesis 视频流。

要在 Kinesis 视频流中使用不同的区域，请执行以下操作之一：

- 将以下环境变量设置为您的区域（例如，*us-east-1*）：

```
export AWS_DEFAULT_REGION=us-east-1
```

创建有权写入 Kinesis Video Streams 的 IAM 用户

如果您尚未这样做，请设置一个有权写入 Kinesis 视频流的 AWS Identity and Access Management (IAM) 用户。

这些步骤旨在帮助您快速开始使用 AWS 访问密钥（access key pair）。设备可以使用 X.509 证书进行连接。AWS IoT 有关如何将设备配置为使用基于证书的身份验证的更多信息，请参阅 [the section called “使用控制对 Kinesis Video Streams 资源的访问权限 AWS IoT”](#)。

1. 登录 AWS Management Console 并打开 IAM 控制台，[网址为 https://console.aws.amazon.com/iam/](https://console.aws.amazon.com/iam/)。
2. 在左侧的导航菜单中，选择用户。
3. 要创建新的用户，请选择添加用户。
4. 为用户提供一个描述性的用户名，例如 **kinesis-video-raspberry-pi-producer**。
5. 在访问类型下面，选择编程访问。
6. 选择下一步：权限。
7. 在“为 kinesis-video-raspberry-pi-producer 设置权限”下，选择“直接附加现有策略”。
8. 选择 创建策略。将在新的 Web 浏览器选项卡中打开创建策略页。
9. 选择 JSON 选项卡。
10. 将以下 JSON 策略复制并粘贴到文本区域中。此策略允许您的用户创建数据并将其写入 Kinesis 视频流。

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "kinesisvideo:DescribeStream",
      "kinesisvideo:CreateStream",
      "kinesisvideo:GetDataEndpoint",
      "kinesisvideo:PutMedia"
    ]
  }]
}
```

```
    ],
    "Resource": [
      "*"
    ]
  }]
}
```

11. 选择查看策略。
12. 为策略提供名称，例如 **kinesis-video-stream-write-policy**。
13. 选择 创建策略。
14. 在浏览器中返回到添加用户选项卡，然后选择刷新。
15. 在搜索框中，键入创建的策略的名称。
16. 在列表中选中新策略旁边的复选框。
17. 选择 Next: Review (下一步: 审核)。
18. 选择 Create user。
19. 控制台将显示新用户的访问密钥 ID。选择显示以显示秘密访问密钥。记录这些值；在配置应用程序时，需要使用这些值。

加入你的 Raspberry Pi 到 Wi-Fi 网络

您可以在无管控 模式下使用 Raspberry Pi，即，未连接键盘、显示器或网络电缆。如果使用连接的显示器和键盘，请转到 [配置树莓派摄像头](#)。

1. 在您的计算机上，创建一个名为 `wpa_supplicant.conf` 的文件。
2. 复制以下文本并将其粘贴到 `wpa_supplicant.conf` 文件中：

```
country=US
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

network={
  ssid="Your Wi-Fi SSID"
  scan_ssid=1
  key_mgmt=WPA-PSK
  psk="Your Wi-Fi Password"
}
```

与 `ssid` 和 `psk` 值替换为您的 Wi-Fi 网络的信息。

3. 将 `wpa_supplicant.conf` 文件复制到 SD 卡中。必须将其复制到 boot 卷的根目录中。
4. 将 SD 卡插入到 Raspberry Pi，然后打开设备电源。它将加入您的 Wi-Fi 网络并启用 SSH。

远程连接到你的 Raspberry Pi

您可以在无管控模式下远程连接到您的 Raspberry Pi。如果将 Raspberry Pi 与连接的显示器和键盘一起使用，请转到 [配置树莓派摄像头](#)。

1. 在远程连接到您的 Raspberry Pi 设备之前，请执行以下操作之一以确定其 IP 地址：
 - 如果您有权访问您的网络的 Wi-Fi 路由器，请查看连接的 Wi-Fi 设备。查找名为 Raspberry Pi 的设备以查找您的设备的 IP 地址。
 - 如果您无权访问您的网络的 Wi-Fi 路由器，您可以使用其他软件查找您的网络上的设备。[Fing](#) 是一个常用的应用程序，它适用于 Android 和 iOS 设备。您可以使用该应用程序的免费版本查找您的网络上的设备的 IP 地址。
2. 如果您知道 Raspberry Pi 设备的 IP 地址，您可以使用任何终端应用程序进行连接。
 - 在 macOS 或 Linux 上，使用 `ssh`：

```
$ ssh pi@<IP address>
```

- 在 Windows 上，使用 [PuTTY](#)，这是一个适用于 Windows 的免费 SSH 客户端。

对于新安装的 Raspbian，用户名为 **pi**，密码为 **raspberry**。建议您[更改默认密码](#)。

配置树莓派摄像头

按照以下步骤将 Raspberry Pi 摄像机配置为将视频从设备发送到 Kinesis 视频流。

1. 打开编辑器以使用以下命令更新 `modules` 文件：

```
$ sudo nano /etc/modules
```

2. 将以下行添加到文件结尾 (如果尚未存在)：

```
bcm2835-v4l2
```

3. 保存该文件并退出编辑器 (Ctrl-X)。

4. 重启 Raspberry Pi :

```
$ sudo reboot
```

5. 如果远程进行连接，在设备重启时，请通过终端应用程序再次连接到该设备。

6. 打开 raspi-config:

```
$ sudo raspi-config
```

7. 选择“接口选项”、“旧版相机”。在旧版本的 Raspbian 操作系统中，此菜单选项可能位于“接口选项”“摄像头”下。

启用摄像机 (如果尚未启用)，并在出现提示时重启。

8. 键入以下命令以验证摄像机是否正常工作：

```
$ raspistill -v -o test.jpg
```

如果您的相机配置正确，则此命令会从相机捕获图像，将其保存到名为的文件中test.jpg，并显示信息性消息。

安装软件先决条件

C++ 创建者开发工具包要求在 Raspberry Pi 上安装以下必备软件。

1. 更新软件包列表并安装构建 SDK 所需的库。键入以下命令：

```
$ sudo apt update
$ sudo apt install -y \
  automake \
  build-essential \
  cmake \
  git \
  gstreamer1.0-plugins-base-apps \
  gstreamer1.0-plugins-bad \
  gstreamer1.0-plugins-good \
  gstreamer1.0-plugins-ugly \
  gstreamer1.0-tools \
  gstreamer1.0-omx-generic \
  libcurl4-openssl-dev \
  libgstreamer1.0-dev \
```

```
libgstreamer-plugins-base1.0-dev \  
liblog4cplus-dev \  
libssl-dev \  
pkg-config
```

2. 将以下 PEM 文件复制到 `/etc/ssl/cert.pem` 中：

```
$ sudo curl https://www.amazontrust.com/repository/AmazonRootCA1.pem -o /etc/ssl/  
AmazonRootCA1.pem  
$ sudo chmod 644 /etc/ssl/AmazonRootCA1.pem
```

下载并构建 Kinesis Video Streams C++ Producer SDK

你可以按照以下步骤下载和构建 Kinesis Video Streams C++ Producer SDK。此方法需要更长的构建时间，具体取决于网络连接和处理器速度。

1. 下载 SDK 类型：

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

2. 准备一个构建目录。类型：

```
$ mkdir -p amazon-kinesis-video-streams-producer-sdk-cpp/build  
$ cd amazon-kinesis-video-streams-producer-sdk-cpp/build
```

3. 构建 SDK 和示例应用程序。根据您的构建的 Raspberry Pi 型号，首次运行可能需要几个小时：

```
$ cmake .. -DBUILD_GSTREAMER_PLUGIN=ON -DBUILD_DEPENDENCIES=FALSE  
$ make
```

将视频流式传输到您的 Kinesis 视频流并观看直播

1. 要运行示例应用程序，您需要具有以下信息：

- 在[先决条件](#)一节中创建的流名称。
- 在[创建有权写入 Kinesis Video Streams 的 IAM 用户](#)中创建的账户凭证 (访问密钥 ID 和秘密访问密钥)。

2. 使用以下命令运行示例应用程序。将占位符替换为您的环境值。

```
$ export GST_PLUGIN_PATH=Directory Where You Cloned the SDK/amazon-kinesis-video-streams-producer-sdk-cpp/build
$ export AWS_DEFAULT_REGION=AWS Region i.e. us-east-1
$ export AWS_ACCESS_KEY_ID=Access Key ID
$ export AWS_SECRET_ACCESS_KEY=Secret Access Key
$ ./kvs_gstreamer_sample Your Stream Name
```

3. 如果示例应用程序退出library not found时出现错误，请键入以下命令以验证该项目是否已正确链接到其开源依赖关系：

```
$ gst-inspect-1.0 kvssink
```

4. 打开 [Kinesis Video Streams](#) 控制台。
5. 选择创建的流的流名称。

将在控制台中显示从 Raspberry Pi 发送的视频流。

直播播放时，你可以试用 Kinesis Video Streams 控制台的以下功能：

- 在 Video preview (视频预览) 部分中，使用导航控件后退和快进流。
- 在 Stream info (流信息) 部分中，查看流的编解码器、分辨率和比特率。有意在 Raspberry Pi 上设置较低分辨率和比特率值，以便在本教程中最大限度减少使用的带宽。要查看正在为您的直播创建的 Amazon CloudWatch 指标，请在中选择查看直播指标 CloudWatch。
- 在 Data retention period (数据保留期) 下面，可以看到视频流保留 1 天。您可以编辑该值并将其设置为 No data retention (不保留数据)，或者设置 1 天到几年之间的值。

在服务器端加密下，请注意，您的数据是使用由 AWS Key Management Service (AWS KMS) 维护的密钥进行静态加密的。

创建者开发工具包参考

本节包含 [Kinesis 视频直播制作入库](#) 的限制、错误代码和其他参考信息。

主题

- [制作人 SDK 限制](#)
- [错误代码参考](#)

- [网络抽象层 \(NAL\) 适配标记参考](#)
- [制作人 SDK 结构](#)
- [Kinesis 视频流结构](#)
- [制作人 SDK 回调](#)

制作人 SDK 限制

下表包含[创建者库](#)中的当前值限制。

Note

设置这些值之前，必须验证您的输入。该开发工具包不会验证这些限制，超出限制时不会发生运行时错误。

值	限制	注意事项
最大流数	128	创建者对象可以创建的最大流数。这是一个软限制 (您可以请求增加值)。它保证了制作人不会意外递归地创建直播。
最大设备名称长度	128 个字符	
最大标记数	每个流 50 个	
最大流名称长度	256 个字符	
最小存储大小	10 MiB = 10 * 1024 * 1024 字节	
最大存储大小	10 GiB = 10 * 1024 * 1024 * 1024 字节	
最大根目录路径长度	4,096 个字符	
最大身份验证信息长度	10000 字节	

值	限制	注意事项
最大 URI 字符串长度	10,000 个字符	
最大标记名称长度	128 个字符	
最大标记值长度	1,024 个字符	
最小安全令牌期限	30 秒	
安全令牌宽限期	40 minutes	如果指定的持续时间更长，则仅限于此值。
保留期	0 或大于 1 小时	0 表示不保留。
最小集群持续时间	1 秒	该值是以 100 纳秒为单位指定的，这是 SDK 标准。
最大集群持续时间	30 秒	该值是以 100 纳秒为单位指定的，这是 SDK 标准。后端 API 可以强制缩短集群持续时间。
最大片段大小	50 MB	有关更多信息，请参阅 Kinesis Video Streams 服务配额 ：
最大片段持续时间	20 秒	有关更多信息，请参阅 Kinesis Video Streams 服务配额 ：
最大连接持续时间	45 minutes	后端在该时间后关闭连接。SDK 在该时间内轮换令牌并建立新连接。
最大 ACK 段长度	1,024 个字符	发送到 ACK 解析器函数的确认的最大段长度。
最大内容类型字符串长度	128 个字符	
最大编解码器 ID 字符串长度	32 个字符	
最大音轨名称字符串长度	32 个字符	

值	限制	注意事项
最大编解码器私有数据长度	1 MiB = 1 * 1024 * 1024 字节	
最小时间码标度值长度	100 纳秒	在生成的 MKV 集群中表示帧时间戳的最小时间码标度值。该值是以 100 纳秒为增量指定的，这是 SDK 标准。
最大时间码标度值长度	1 秒	在生成的 MKV 集群中表示帧时间戳的最大时间码标度值。该值是以 100 纳秒为增量指定的，这是 SDK 标准。
最小内容视图项目数	10	
最小缓冲区持续时间	20 秒	该值是以 100 纳秒为增量指定的，这是 SDK 标准。
最大更新版本长度	128 个字符	
最大 ARN 长度	1024 个字符	
最大片段序列长度	128 个字符	
最大保留期	10 岁	

错误代码参考

此部分包含 [创建者库](#) 的错误和状态代码信息。

有关常见问题解决方法的信息，请参阅 [对 Kinesis Video Streams 进行故障排除](#)

主题

- [PutFrame 回调返回的错误和状态码-平台独立代码 \(PIC\)](#)
- [PutFrame 回调返回的错误和状态码-C 制作人库](#)

PutFrame 回调返回的错误和状态码-平台独立代码 (PIC)

以下各节包含平台独立代码 (PIC) 中PutFrame操作的回调返回的错误和状态信息。

主题

- [客户端库返回的错误和状态码](#)
- [持续时间库返回的错误和状态码](#)
- [公共库返回的错误和状态码](#)
- [堆库返回的错误和状态码](#)
- [MKVGen 库返回的错误和状态码](#)
- [Trace 库返回的错误和状态码](#)
- [Utils 库返回的错误和状态码](#)
- [View 库返回的错误和状态码](#)

客户端库返回的错误和状态码

下表包含 Kinesis Video Client Streams 库中的方法返回的错误和状态信息。

代码	消息	描述	推荐操作
0x52000001	STATUS_MAX_STREAM_COUNT	已达到最大流数。	在 DeviceInfo 中指定更大的最大流数，该数量是在 制作人 SDK 限制 中指定的。
0x52000002	STATUS_MIN_STREAM_COUNT	最小流数错误。	在中指定大于零的流的最大数量DeviceInfo。
0x52000003	STATUS_INVALID_DEVICE_NAME_LENGTH	设备名称长度无效。	请参阅中指定的以字符为单位的最大设备名称长度 制作人 SDK 限制 。
0x52000004	STATUS_INVALID_DEVICE_INFO_VERSION	DeviceInfo 结构版本无效。	指定正确的当前结构版本。

代码	消息	描述	推荐操作
0x52000005	STATUS_MAX_TAG_COUNT	已达到最大标记数。	请参阅中指定的当前最大标签数 制作人 SDK 限制 。
0x52000006	STATUS_DEVICE_FINGERPRINT_LENGTH		
0x52000007	STATUS_INVALID_CALLBACKS_VERSION	Callbacks 结构版本无效。	指定正确的当前结构版本。
0x52000008	STATUS_INVALID_STREAM_INFO_VERSION	StreamInfo 结构版本无效。	指定正确的当前结构版本。
0x52000009	STATUS_INVALID_STREAM_NAME_LENGTH	流名称长度无效。	请参阅中指定的以字符为单位的最大直播名称长度 制作人 SDK 限制 。
0x5200000a	STATUS_INVALID_STORAGE_SIZE	指定的存储大小无效。	存储大小 (字节) 必须在 制作人 SDK 限制 中指定的限制内。
0x5200000b	STATUS_INVALID_ROOT_DIRECTORY_LENGTH	根目录字符串长度无效。	请参阅中指定的最大根目录路径长度 制作人 SDK 限制 。
0x5200000c	STATUS_INVALID_SPILL_RATIO	溢出率无效。	以 0—100 之间的百分比表示溢出率。
0x5200000d	STATUS_INVALID_STORAGE_INFO_VERSION	StorageInfo 结构版本无效。	指定正确的当前结构版本。

代码	消息	描述	推荐操作
0x5200000e	STATUS_INVALID_STREAM_STATE	流处于不允许执行当前操作的状态。	最常见的是，当 SDK 无法达到执行请求的操作所需的状态时，就会发生此错误。例如，如果 <code>GetStreamingEndpoint</code> API 调用失败，并且客户端应用程序忽略该失败并继续将帧放入流中，则会出现该错误。
0x5200000f	STATUS_SERVICE_CALLBACKS_MISSING	对于某些必需函数， <code>Callbacks</code> 结构具有缺少的函数入口点。	验证强制回调是否已在客户端应用程序中实现。此错误仅暴露于平台独立代码 (PIC) 客户端。C++ 和其他更高级别的包装程序满足这些调用的要求。
0x52000010	STATUS_SERVICE_CALLBACK_NOT_AUTHORIZED_ERROR	未授权。	验证安全令牌、证书、安全令牌集成和到期。验证令牌是否具有正确的关联权限。对于 Kinesis Video Streams 示例应用程序，请验证环境变量设置是否正确。
0x52000011	STATUS_DESCRIBE_STREAM_CALL_FAILED	<code>DescribeStream</code> API 失败。	在 <code>DescribeStream</code> API 重试失败后，将返回该错误。PIC 客户端在停止重试后返回此错误。

代码	消息	描述	推荐操作
0x52000012	STATUS_INVALID_DESCRIPTOR_RESPONSE	DescribeStreamResponse 结构无效。	传递给 DescribeStreamResultEvent 的结构为 Null 或包含无效的项目，例如，Null Amazon 资源名称 (ARN)。
0x52000013	STATUS_STREAM_IS_BEING_DELETED_ERROR	正在删除流。	API 失败是由正在删除的流造成的。确认在使用该流时，没有其他进程正在尝试删除该流。
0x52000014	STATUS_SERVICE_CALL_INVALID_ARG_ERROR	为服务调用指定的参数无效。	当服务调用参数无效或 SDK 遇到无法解释的错误时，后端会返回此错误。
0x52000015	STATUS_SERVICE_CALL_DEVICE_NOT_FOUND_ERROR	找不到设备。	确认设备在使用过程中未被删除。
0x52000016	STATUS_SERVICE_CALL_DEVICE_NOT_PROVISIONED_ERROR	未预置设备。	验证设备是否已配置。
0x52000017	STATUS_SERVICE_CALL_RESOURCE_NOT_FOUND_ERROR	找不到从服务中返回的常规资源。	在服务找不到资源 (例如，流) 时，将会出现该错误。它可能在不同上下文中具有不同的含义，但可能的原因是在创建流之前使用 API。使用 SDK 确认直播已先创建。

代码	消息	描述	推荐操作
0x52000018	STATUS_INVALID_AUTH_LEN	身份验证信息长度无效。	请参阅 制作人 SDK 限制 中指定的当前值。
0x52000019	STATUS_CREATE_STREAM_CALL_FAILED	CreateStream API 调用失败。	有关操作失败原因的更详细信息，请参阅错误字符串。
0x5200002a	STATUS_GET_STREAMING_TOKEN_CALL_FAILED	GetStreamingToken 调用失败。	有关操作失败原因的更详细信息，请参阅错误字符串。
0x5200002b	STATUS_GET_STREAMING_ENDPOINT_CALL_FAILED	GetStreamingEndpoint API 调用失败。	有关操作失败原因的更详细信息，请参阅错误字符串。
0x5200002c	STATUS_INVALID_URI_LEN	从 GetStreamingEndpoint API 中返回的 URI 字符串长度无效。	请参阅 制作人 SDK 限制 中指定的当前最大值。
0x5200002d	STATUS_PUT_STREAM_CALL_FAILED	PutMedia API 调用失败。	有关操作失败原因的更详细信息，请参阅错误字符串。

代码	消息	描述	推荐操作
0x5200002e	STATUS_STORE_OUT_OF_MEMORY	内容存储内存不足。	内容存储是在流之间共享的，应具有足够的容量以存储所有流的最大持续时间内的内容 + ~20% (考虑碎片整理)。请切勿使存储溢出。请选择与累计存储大小和容许的延迟对应的每个流的最大持续时间值。我们建议在框架从内容视图窗口中掉出来时将其丢掉，而不是直接放置 (内容存储内存压力)。这是因为丢帧会启动直播压力通知回调。然后，应用程序可以调整上游媒体组件 (如编码器) 以降低比特率，删除帧或执行相应的操作。
0x5200002f	STATUS_NO_MORE_DATA_AVAILABLE	流当前没有更多的可用数据。	在媒体管道生成帧的速度比网络线程使用发送到服务的帧的速度慢时，这可能是一个有效的结果。更高级别的客户端 (例如 C++、Java 或 Android) 不会看到此警告，因为它是在内部处理的。
0x52000030	STATUS_INVALID_TAG_VERSION	Tag 结构版本无效。	指定正确的当前结构版本。
0x52000031	STATUS_SERVICE_CALL_UNKNOWN_ERROR	从网络堆栈中返回未知或常规错误。	有关更详细信息，请参阅日志。

代码	消息	描述	推荐操作
0x52000032	STATUS_SERVICE_CALL_RESOURCE_IN_USE_ERROR	正在使用资源。	这是从服务中返回的。有关更多信息，请参阅 Kinesis Video Streams API 参考。
0x52000033	STATUS_SERVICE_CALL_CLIENT_LIMIT_ERROR	客户端限制。	这是从服务中返回的。有关更多信息，请参阅 Kinesis Video Streams API 参考。
0x52000034	STATUS_SERVICE_CALL_DEVICE_LIMIT_ERROR	设备限制。	这是从服务中返回的。有关更多信息，请参阅 Kinesis Video Streams API 参考。
0x52000035	STATUS_SERVICE_CALL_STREAM_LIMIT_ERROR	流限制。	这是从服务中返回的。有关更多信息，请参阅 Kinesis Video Streams API 参考。
0x52000036	STATUS_SERVICE_CALL_RESOURCE_DELETED_ERROR	已删除或正在删除资源。	这是从服务中返回的。有关更多信息，请参阅 Kinesis Video Streams API 参考。
0x52000037	STATUS_SERVICE_CALL_TIMEOUT_ERROR	服务调用超时。	调用特定服务 API 导致超时。验证您的网络连接是否有效。PIC 将自动重试该操作。
0x52000038	STATUS_STREAM_READ_CALLBACK_FAILED	流就绪通知。	该通知从 PIC 发送到客户端，表示已创建异步流。

代码	消息	描述	推荐操作
0x52000039	STATUS_DE VICE_TAGS _COUNT_NO N_ZERO_TAGS_NULL	指定的标记无效。	标签数不为零，但标签为空。验证标签是否已指定或计数为零。
0x5200003a	STATUS_IN VALID_STR EAM_DESCR IPTION_VERSION	StreamDes cription 结构版 本无效。	指定正确的当前结构版本。
0x5200003b	STATUS_IN VALID_TAG _NAME_LEN	标记名称长度无效。	请参阅 制作人 SDK 限制 中指定的标记名称限制。
0x5200003c	STATUS_IN VALID_TAG _VALUE_LEN	标记值长度无效。	请参阅 制作人 SDK 限制 中指定的标记值限制。
0x5200003d	STATUS_TA G_STREAM_ CALL_FAILED	TagResource API 失败。	TagResource API 调用失败。检查有效的网络连接。有关该失败的更多信息，请参阅日志。
0x5200003e	STATUS_IN VALID_CUS TOM_DATA	调用 PIC API 的自 定义数据无效。	在 PIC API 调用中指定的自定义数据无效。仅在直接使用 PIC 的客户端中可能会发生这种情况。
0x5200003f	STATUS_IN VALID_CRE ATE_STREA M_RESPONSE	CreateStr eamResponse 结构无效。	结构或其成员字段无效 (即，ARN 为 Null 或大于 制作人 SDK 限制 中指定的值)。

代码	消息	描述	推荐操作
0x52000040	STATUS_CLIENT_AUTH_CALL_FAILED	客户端身份验证失败。	多次重试后，PIC 未能获得正确的身份验证信息 (AccessKey Id 或 SecretAccessKey)。检查身份验证集成。示例应用程序使用环境变量将凭证信息传入到 C++ 创建者库。
0x52000041	STATUS_GET_CLIENT_TOKEN_CALL_FAILED	获取安全令牌调用失败。	仅在直接使用 PIC 的客户端中可能会发生这种情况。在重试一定次数后，调用失败并出现该错误。
0x52000042	STATUS_CLIENT_PROVISION_CALL_FAILED	预置错误。	未实现配置。
0x52000043	STATUS_CREATE_CLIENT_CALL_FAILED	无法创建创建者客户端。	在客户端创建失败时，在重试一定次数后，PIC 返回一个常规错误。
0x52000044	STATUS_CLIENT_READ_Y_CALLBACK_FAILED	无法将创建者客户端置于就绪状态。	这是在 PIC 无法变为就绪状态时 PIC 状态机返回的。有关根本原因的更多信息，请参阅日志。
0x52000045	STATUS_TAG_CLIENT_CALL_FAILED	创建者客户端的 TagResource 失败。	创建者客户端的 TagResource API 调用失败。有关根本原因的更多信息，请参阅日志。

代码	消息	描述	推荐操作
0x52000046	STATUS_INVALID_CREATE_DEVICE_RESPONSE	设备/创建者创建失败。	更高级别的软件开发工具包 (例如 C++ 或 Java) 尚未实现设备或制作者创建 API。直接使用 PIC 的客户端可以使用结果通知指示失败。
0x52000047	STATUS_ACK_TIMESTAMP_NOT_IN_VIEW_WINDOW	在视图中不显示收到的 ACK 的时间戳。	如果与收到的 ACK 对应的帧没有位于内容视图窗口中, 则会出现该错误。通常, 如果 ACK 传输速度较慢, 则会发生这种情况。可以将其解释为警告, 并指示下行链路速度较慢。
0x52000048	STATUS_INVALID_FRAGMENT_ACK_VERSION	FragmentAck 结构版本无效。	指定正确的当前 FragmentAck 结构版本。
0x52000049	STATUS_INVALID_TOKEN_EXPIRATION	安全令牌到期日期无效。	安全令牌到期后的绝对时间戳应大于当前时间戳, 并带有宽限期。有关宽限期的限制, 请参阅 制作人 SDK 限制 。
0x5200004a	STATUS_END_OF_STREAM	流结束 (EOS) 指示符。	在 GetStreamData API 调用中, 指示当前上传处理会话已结束。如果会话结束或出现错误, 或者会话令牌已过期并且正在轮换会话, 则会发生这种情况。

代码	消息	描述	推荐操作
0x5200004b	STATUS_DUPLICATE_STREAM_NAME	重复的流名称。	多个流不能具有相同的流名称。为流选择唯一的名称。
0x5200004c	STATUS_INVALID_RETENTION_PERIOD	保留期无效。	在 StreamInfo 结构中指定的保留期无效。有关有效的保留期值范围的信息，请参阅 制作人 SDK 限制 。
0x5200004d	STATUS_INVALID_ACK_KEY_START	无效 FragmentAck 。	无法解析片段 ACK 字符串。键开始指示符无效。片段 ACK 字符串可能已损坏。它可以自行更正，并且可以将该错误视为警告。
0x5200004e	STATUS_INVALID_DUPLICATE_KEY_NAME	无效 FragmentAck 。	无法解析片段 ACK 字符串。多个键具有相同的名称。片段 ACK 字符串可能已损坏。它可以自行更正，并且可以将该错误视为警告。
0x5200004f	STATUS_INVALID_ACK_INVALID_VALUE_START	无效 FragmentAck 。	无法解析片段 ACK 字符串，因为键值开始指示符无效。片段 ACK 字符串可能已损坏。它可以自行更正，并且可以将该错误视为警告。

代码	消息	描述	推荐操作
0x52000050	STATUS_INVALID_ACK_INVALID_VALUE_END	无效 FragmentAck 。	无法解析片段 ACK 字符串，因为键值结束指示符无效。片段 ACK 字符串可能已损坏。它可以自行更正，并且可以将该错误视为警告。
0x52000051	STATUS_INVALID_PARSED_ACK_TYPE	无效 FragmentAck 。	无法解析片段 ACK 字符串，因为指定的 ACK 类型无效。
0x52000052	STATUS_STREAM_HAS_BEEN_STOPPED	已停止流。	已停止流，但仍在将帧放入流中。
0x52000053	STATUS_INVALID_STREAM_METRICS_VERSION	StreamMetrics 结构版本无效。	指定正确的当前 StreamMetrics 结构版本。
0x52000054	STATUS_INVALID_CLIENT_METRICS_VERSION	ClientMetrics 结构版本无效。	指定正确的当前 ClientMetrics 结构版本。
0x52000055	STATUS_INVALID_CLIENT_READY_STATE	创建者初始化无法达到就绪状态。	在创建者客户端初始化期间，无法达到就绪状态。有关更多信息，请参阅日志。
0x52000056	STATUS_STATE_MACHINE_STATE_NOT_FOUND	内部状态机错误。	不是公开可见的错误。

代码	消息	描述	推荐操作
0x52000057	STATUS_INVALID_FRAGMENT_ACK_TYPE	在 FragmentAck 结构中指定的 ACK 类型无效。	FragmentAck 结构应包含公共标头中定义的 ACK 类型。
0x52000058	STATUS_INVALID_STREAM_READY_STATE	内部状态机转换错误。	不是公开可见的错误。
0x52000059	STATUS_CLIENT_FREE_BEFORE_STREAM	在释放创建者后释放流对象。	尝试在释放创建者对象后释放流对象。仅在直接使用 PIC 的客户端中可能会发生这种情况。
0x5200005a	STATUS_ALLOCATION_SIZE_SMALLER_THAN_REQUESTED	内部存储错误。	内部错误，表示来自内容存储的实际分配大小小于已打包的帧和片段的大小。
0x5200005b	STATUS_VIEW_ITEM_SIZE_GREATER_THAN_ALLOCATION	内部存储错误。	内容视图中存储的分配大小大于内容存储中的分配大小。
0x5200005c	STATUS_ACK_ERR_STREAM_READ_ERROR	流读取错误 ACK。	ACK 从后端返回的错误，表示流读取或解析错误。在后端无法检索流时，通常会发生这种情况。自动重新流式传输通常可以纠正该错误。

代码	消息	描述	推荐操作
0x5200005d	STATUS_ACK_ERR_FRAGMENT_SIZE_REACHED	已达到最大片段大小。	在 制作人 SDK 限制 中定义了最大片段大小 (字节)。该错误表示具有非常大的帧，或者没有用于创建可管理大小的片段的关键帧。检查编码器设置并验证关键帧是否正确生成。对于具有较高密度的流，请将编码器配置为生成具有较短持续时间的片段以管理最大大小。
0x5200005e	STATUS_ACK_ERR_FRAGMENT_DURATION_REACHED	已达到最大片段持续时间。	在 制作人 SDK 限制 中定义了最大片段持续时间。该错误表示每秒具有非常少的帧，或者没有用于创建可管理持续时间的片段的关键帧。检查编码器设置并验证关键帧是否按固定间隔正确生成。
0x5200005f	STATUS_ACK_ERR_CONNECTION_DURATION_REACHED	已达到最大连接持续时间。	Kinesis Video Streams 强制执行中指定的最大连接持续时间。 制作人 SDK 限制 在达到最大值之前，Producer SDK 会自动轮换直播或令牌。使用 SDK 的客户不应收到此错误。
0x52000060	STATUS_ACK_ERR_FRAGMENT_TIMESTAMP_NOT_MONOTONIC	时间码不是单调递增的。	Producer SDK 强制使用时间戳，因此使用 SDK 的客户端不应收到此错误。

代码	消息	描述	推荐操作
0x52000061	STATUS_AC K_ERR_MUL TI_TRACK_MKV	在 MKV 中具有多个音轨。	Producer SDK 强制使用单轨流，因此使用 SDK 的客户端不应收到此错误。
0x52000062	STATUS_AC K_ERR_INV ALID_MKV_DATA	MKV 数据无效。	后端 MKV 解析器在解析流时遇到错误。如果直播在过渡期间损坏，使用 SDK 的客户端可能会遇到此错误。如果缓冲压力迫使 SDK 丢掉部分传输的尾帧，也会发生这种情况。在后一种情况下，我们建议您要么降低 FPS 和分辨率，提高压缩率，要么（如果有“爆发”的网络）允许更大的内容存储和缓冲持续时间以适应临时压力。
0x52000063	STATUS_AC K_ERR_INV ALID_PROD UCER_TIMESTAMP	创建者时间戳无效。	如果创建者时钟大幅漂移到将来的时间，服务将返回该错误 ACK。更高级别的 SDK (例如，Java 或 C++) 使用某种版本的系统时钟以满足从 PIC 中回调当前时间的要求。验证系统时钟设置是否正确。直接使用 PIC 的客户端应验证其回调函数是否返回了正确的时间戳。

代码	消息	描述	推荐操作
0x52000064	STATUS_ACK_ERR_STREAM_NOT_ACTIVE	非活动流。	在流未处于“活动”状态时，对后端 API 进行了调用。如果客户端创建流并立即继续将帧推入流中，则会发生这种情况。SDK 通过状态机和恢复机制处理这种情况。
0x52000065	STATUS_ACK_ERR_KMS_KEY_ACCESS_DENIED	AWS KMS 访问被拒绝错误。	在账户没有指定密钥的访问权限时返回。
0x52000066	STATUS_ACK_ERR_KMS_KEY_DISABLED	AWS KMS 密钥已禁用。	已禁用指定的密钥。
0x52000067	STATUS_ACK_ERR_KMS_KEY_VALIDATION_ERROR	AWS KMS 密钥验证错误。	常规验证错误。有关更多信息，请参阅 AWS Key Management Service API 参考 。
0x52000068	STATUS_ACK_ERR_KMS_KEY_UNAVAILABLE	AWS KMS key 不可用。	密钥不可用。有关更多信息，请参阅 AWS Key Management Service API 参考 。
0x52000069	STATUS_ACK_ERR_KMS_KEY_INVALID_USAGE	KMS 密钥的使用无效。	未配置为在此上下文中使用。AWS KMS key 有关更多信息，请参阅 AWS Key Management Service API 参考 。

代码	消息	描述	推荐操作
0x5200006a	STATUS_AK_ERR_KMS_KEY_INVALID_STATE	AWS KMS 状态无效。	有关更多信息，请参阅 AWS Key Management Service API 参考 。
0x5200006b	STATUS_AK_ERR_KMS_KEY_NOT_FOUND	找不到 KMS 密钥。	找不到密钥。有关更多信息，请参阅 AWS Key Management Service API 参考 。
0x5200006c	STATUS_AK_ERR_STREAM_DELETED	已删除或正在删除流。	正在由另一个应用程序或通过 AWS Management Console 删除流。
0x5200006d	STATUS_AK_ERR_ACK_INTERNAL_ERROR	内部错误。	常规服务内部错误。
0x5200006e	STATUS_AK_ERR_FRAGMENT_ARCHIVAL_ERROR	片段存档错误。	在服务无法持久保留片段并为其编制索引时返回。虽然非常少见，但可能会出于各种原因发生这种情况。默认情况下，SDK 再次尝试发送片段。
0x5200006f	STATUS_AK_ERR_UNKNOWN_ACK_ERROR	未知错误。	服务返回未知错误。
0x52000070	STATUS_MISSING_ERR_ACK_ID	缺少 ACK 信息。	ACK 解析器已完成解析，但缺少 FragmentAck 信息。

代码	消息	描述	推荐操作
0x52000071	STATUS_INVALID_ACK_SEGMENT_LEN	ACK 段长度无效。	为 ACK 解析器指定的 ACK 段字符串具有无效的长度。有关更多信息，请参阅 制作 SDK 限制 ：
0x52000074	STATUS_MAXIMUM_FRAGMENT_METADATA_COUNT	已向片段中添加最大数目的元数据项目。	Kinesis 视频流最多可以向片段添加 10 个元数据项，方法是向片段添加非永久项目，也可以向元数据队列中添加永久项目。有关更多信息，请参阅 在 Kinesis Video Streams 中使用流式传输元数据 ：
0x52000075	STATUS_ACK_FRAGMENT_METADATA_LIMIT_REACHED	已达到限制（最大元数据计数、元数据名称长度或元数据值长度）。	创建者开发工具包限制元数据项目的数量和大小。除非更改了 Producer SDK 代码中的限制，否则不会发生此错误。有关更多信息，请参阅 在 Kinesis Video Streams 中使用流式传输元数据 ：
0x52000076	STATUS_BLOCKING_PUT_INTERRUPTED_STREAM_TERMINATED	未实现。	

代码	消息	描述	推荐操作
0x52000077	STATUS_INVALID_METADATA_NAME	元数据名称无效。	元数据名称不能以字符串“AWS”开头。如果发生此错误，则不会将元数据项添加到片段或元数据队列中。有关更多信息，请参阅 在 Kinesis Video Streams 中使用流式传输元数据 ：
0x52000078	STATUS_END_OF_FRAGMENT_FRAME_INVALID_STATE	片段帧的结尾处于无效状态。	片段的结尾不应以碎片流的形式 non-key-frame 发送。
0x52000079	STATUS_TRACK_INFO_MISSING	缺少轨道信息。	曲目编号必须大于零且与曲目 ID 相匹配。
0x5200007a	STATUS_MAX_TRACK_COUNT_EXCEEDED	超出最大轨道数。	每个直播最多可以有三首曲目。
0x5200007b	STATUS_OFFLINE_MODE_WITH_ZERO_RETENTION	离线流式处理模式保留时间设置为零。	离线直播模式的保留时间不应设置为零。
0x5200007c	STATUS_ACK_TRACK_NUMBER_MISMATCH	错误 ACK 的轨道编号不匹配。	
0x5200007d	STATUS_ACK_FRAMES_MISSING_FOR_TRACK	轨道缺少帧。	

代码	消息	描述	推荐操作
0x5200007e	STATUS_ACK_ERR_MORE_THAN_ALLOWED_TRACKS_FOUND	超出允许的最大轨道数。	
0x5200007f	STATUS_UPLOAD_HANDLE_ABORTED	上传处理已中止。	
0x52000080	STATUS_INVALID_CERT_PATH_LENGTH	证书路径长度无效。	
0x52000081	STATUS_DUPLICATE_TRACK_ID_FOUND	找到重复的轨道ID。	
0x52000082	STATUS_INVALID_CLIENT_INFO_VERSION		
0x52000083	STATUS_INVALID_CLIENT_ID_STRING_LENGTH		
0x52000084	STATUS_SETTING_KEY_FRAME_FLAG_WHILE_USING_EOFR		

代码	消息	描述	推荐操作
0x52000085	STATUS_MAX_FRAME_TIMESTAMP_DELTA_TRACKS_EXCEEDED		
0x52000086	STATUS_STREAM_SHUTTING_DOWN		
0x52000087	STATUS_CLIENT_SHUTTING_DOWN		
0x52000088	STATUS_PUBLISH_MEDIA_LAST_PERSISTENT_ACK_NOT_RECEIVED		
0x52000089	STATUS_NON_ALIGNED_HEAP_WITHIN_CONTENT_STORE_ALLOCATORS		
0x5200008a	STATUS_MULTIPLE_CONSECUTIVE_EOFR		
0x5200008b	STATUS_DUPLICATE_STREAM_EVENT_TYPE		

代码	消息	描述	推荐操作
0x5200008c	STATUS_STREAM_NOT_STARTED		
0x5200008d	STATUS_INVALID_IMAGE_PREFIX_LENGTH		
0x5200008e	STATUS_INVALID_IMAGE_METADATA_KEY_LENGTH		
0x5200008f	STATUS_INVALID_IMAGE_METADATA_VALUE_LENGTH		

持续时间库返回的错误和状态码

下表包含Duration库中方法返回的错误和状态信息。

代码	消息
0xFFFFFFFFFFFFFFFF	INVALID_DURATION_VALUE

公共库返回的错误和状态码

下表包含Common库中方法返回的错误和状态信息。

Note

对于很多 API，这些错误和状态信息代码是相同的。

代码	消息	描述
0x00000001	STATUS_NULL_ARG	为必需参数传递了 NULL。
0x00000002	STATUS_INVALID_ARG	为参数指定的值无效。
0x00000003	STATUS_INVALID_ARG_LEN	指定的参数长度无效。
0x00000004	STATUS_NOT_ENOUGH_MEMORY	无法分配足够的内存。
0x00000005	STATUS_BUFFER_TOO_SMALL	指定的缓冲区大小太小。
0x00000006	STATUS_UNEXPECTED_EOF	到达意外的文件末尾。
0x00000007	STATUS_FORMAT_ERROR	遇到无效的格式。
0x00000008	STATUS_INVALID_HANDLE_ERROR	句柄值无效。
0x00000009	STATUS_OPEN_FILE_FAILED	无法打开文件。
0x0000000a	STATUS_READ_FILE_FAILED	无法从文件中读取。
0x0000000b	STATUS_WRITE_TO_FILE_FAILED	无法写入到文件中。
0x0000000c	STATUS_INTERNAL_ERROR	通常不会出现的内部错误，可能表示 SDK 或服务 API 错误。
0x0000000d	STATUS_INVALID_OPERATION	具有无效的操作，或者不允许执行该操作。

代码	消息	描述
0x0000000e	STATUS_NOT_IMPLEMENTED	未实现该功能。
0x0000000f	STATUS_OPERATION_TIMED_OUT	操作已超时。
0x00000010	STATUS_NOT_FOUND	找不到所需的资源。
0x00000011	STATUS_CREATE_THREAD_FAILED	创建话题失败。
0x00000012	STATUS_THREAD_NOT_ENOUGH_RESOURCES	资源不足，无法创建另一个线程，或者遇到了系统对线程数量施加的限制。
0x00000013	STATUS_THREAD_INVALID_ARG	指定的话题属性无效，或者其他话题已在等待加入此话题。
0x00000014	STATUS_THREAD_PERMISSIONS	没有权限设置线程属性中指定的调度策略和参数。
0x00000015	STATUS_THREAD_DEADLOCKED	检测到死锁或者加入的线程指定了调用线程。
0x00000016	STATUS_THREAD_DOES_NOT_EXIST	找不到具有指定话题 ID 的话题。
0x00000017	STATUS_JOIN_THREAD_FAILED	线程联接操作返回未知或一般错误。
0x00000018	STATUS_WAIT_FAILED	已超过等待条件变量的最长时间。
0x00000019	STATUS_CANCEL_THREAD_FAILED	线程取消操作返回未知或一般错误。
0x0000001a	STATUS_THREAD_IS_NOT_JOINABLE	在不可连接的线程上请求线程加入操作。

代码	消息	描述
0x0000001b	STATUS_DETACH_THREAD_FAILED	线程分离操作返回未知或一般错误。
0x0000001c	STATUS_THREAD_ATTR_INIT_FAILED	无法初始化线程属性对象。
0x0000001d	STATUS_THREAD_ATTR_SET_STACK_SIZE_FAILED	未能为线程属性对象设置堆栈大小。
0x0000001e	STATUS_MEMORY_NOT_FREED	仅在测试中使用。表示未释放所有请求的内存。
0x10000015	STATUS_INVALID_ALLOCATION_SIZE	
0x10000016	STATUS_HEAP_REALLOC_ERROR	
0x10000017	STATUS_HEAP_FILE_HEAP_FILE_CORRUPT	

堆库返回的错误和状态码

下表包含Heap库中方法返回的错误和状态信息。

代码	消息	描述
0x01000001	STATUS_HEAP_FLAGS_ERROR	指定的标记组合无效。
0x01000002	STATUS_HEAP_NOT_INITIALIZED	在初始化堆之前尝试执行一个操作。

代码	消息	描述
0x01000003	STATUS_HEAP_CORRUPTED	堆已损坏或已覆盖保护频段 (在调试模式下)。客户端代码中的缓冲区溢出可能导致堆损坏。
0x01000004	STATUS_HEAP_VRAM_LIB_MISSING	VRAM (视频 RAM) 用户或内核模式库无法加载或缺失。检查底层平台是否支持 VRAM 分配。
0x01000005	STATUS_HEAP_VRAM_LIB_REOPEN	无法打开 VRAM 库。
0x01000006	STATUS_HEAP_VRAM_INIT_FUNC_SYMBOL	无法加载 INIT 函数导出。
0x01000007	STATUS_HEAP_VRAM_ALLOC_FUNC_SYMBOL	无法加载 ALLOC 函数导出。
0x01000008	STATUS_HEAP_VRAM_FREE_FUNC_SYMBOL	无法加载 FREE 函数导出。
0x01000009	STATUS_HEAP_VRAM_LOCK_FUNC_SYMBOL	无法加载 LOCK 函数导出。
0x0100000a	STATUS_HEAP_VRAM_UNLOCK_FUNC_SYMBOL	无法加载 UNLOCK 函数导出。
0x0100000b	STATUS_HEAP_VRAM_UNINIT_FUNC_SYMBOL	无法加载 UNINIT 函数导出。
0x0100000c	STATUS_HEAP_VRAM_GETMAX_FUNC_SYMBOL	无法加载 GETMAX 函数导出。
0x0100000d	STATUS_HEAP_DIRECT_MEM_INIT	无法初始化混合堆中的主堆池。

代码	消息	描述
0x0100000e	STATUS_HEAP_VRAM_INIT_FAILED	VRAM 动态初始化失败。
0x0100000f	STATUS_HEAP_LIBRAR Y_FREE_FAILED	无法取消分配并释放 VRAM 库。
0x01000010	STATUS_HEAP_VRAM_A LLOC_FAILED	VRAM 分配失败。
0x01000011	STATUS_HEAP_VRAM_F REE_FAILED	VRAM 释放失败。
0x01000012	STATUS_HEAP_VRAM_M AP_FAILED	VRAM 映射失败。
0x01000013	STATUS_HEAP_VRAM_U NMAP_FAILED	VRAM 取消映射失败。
0x01000014	STATUS_HEAP_VRAM_U NINIT_FAILED	VRAM 取消初始化失败。

MKVGen 库返回的错误和状态码

下表包含MKVGen库中方法返回的错误和状态信息。

代码	消息	描述/建议的操作
0x32000001	STATUS_MKV_INVALID _FRAME_DATA	Frame 数据结构的成员无效。确保时长、大小和帧数据有效且在中指定的限制范围内 制作 人 SDK 限制 。
0x32000002	STATUS_MKV_INVALID _FRAME_TIMESTAMP	帧时间戳无效。计算的 PTS (演示时间戳) 和 DTS (解码时间戳) 大于或等于片段开始帧的时间戳。这表

代码	消息	描述/建议的操作
		示潜在的媒体管道重新启动或编码器稳定性问题。有关故障排除信息，请参阅 错误：“Failed to submit frame to Kinesis Video client” (无法将帧提交到 Kinesis 视频客户端)。
0x32000003	STATUS_MKV_INVALID_CLUSTER_DURATION	指定的片段持续时间无效。有关更多信息，请参阅 制作人 SDK 限制 ：
0x32000004	STATUS_MKV_INVALID_CONTENT_TYPE_LENGTH	内容类型字符串长度无效。有关更多信息，请参阅 制作人 SDK 限制 ：
0x32000005	STATUS_MKV_NUMBER_TOO_BIG	尝试编码的数字太大，无法使用 EBML (可扩展二进制元语言) 格式表示。不会向 SDK 客户端显示该错误。
0x32000006	STATUS_MKV_INVALID_CODEC_ID_LENGTH	编解码器 ID 字符串长度无效。有关更多信息，请参阅 制作人 SDK 限制 ：
0x32000007	STATUS_MKV_INVALID_TRACK_NAME_LENGTH	音轨名称字符串长度无效。有关更多信息，请参阅 制作人 SDK 限制 ：
0x32000008	STATUS_MKV_INVALID_CODEC_PRIVATE_LENGTH	编解码器私有数据长度无效。有关更多信息，请参阅 制作人 SDK 限制 ：
0x32000009	STATUS_MKV_CODEC_PRIVATE_NULL	编解码器私有数据 (CPD) 为空，而 CPD 大小大于零。

代码	消息	描述/建议的操作
0x3200000a	STATUS_MKV_INVALID_TIMECODE_SCALE	时间码标度值无效。有关更多信息，请参阅 制作人 SDK 限制 ：
0x3200000b	STATUS_MKV_MAX_FRAME_TIMECODE	帧时间码大于最大值。有关更多信息，请参阅 制作人 SDK 限制 ：
0x3200000c	STATUS_MKV_LARGE_FRAME_TIMECODE	已达到最大帧时间码。MKV 格式使用有符号 16 位表示帧相对于集群开头的的时间码。如果无法表示帧时间码，则会生成错误。该错误表示选择的时间码标度不正确或集群持续时间太长，因此，表示帧时间码将导致有符号 16 位空间溢出。
0x3200000d	STATUS_MKV_INVALID_ANNEXB_NALU_IN_FRAME_DATA	遇到无效的 Annex-B 启动代码。例如，指定了 Annex-B 适配标记，并且代码遇到包含超过三个零的无效开始序列。有效的 Annex-B 格式应具有“模拟防护”序列，以便对字节流中包含三个或更多零的序列进行转义。有关更多信息，请参阅 MPEG 规范。有关 Android 上的此错误的信息，请参阅 Android 上的 STATUS_MKV_INVALID_ANNEXB_NALU_IN_FRAME_DATA (0x3200000d) 错误 。

代码	消息	描述/建议的操作
0x3200000e	STATUS_MKV_INVALID_AVCC_NALU_IN_FRAME_DATA	指定自适应 AVCC 标志时，AVCC NALU 打包无效。验证字节流是否为有效的 AVCC 格式。有关更多信息，请参阅 MPEG 规范。
0x3200000f	STATUS_MKV_BOTH_ANNEXB_AND_AVCC_SPECIFIED	同时指定了改编 AVCC 和 Annex-B 的 NALU。指定其中的一个 NAL，或者不指定任何 NAL。
0x32000010	STATUS_MKV_INVALID_ANNEXB_NALU_IN_CPD	在指定了适配 Annex-B 标记时，CPD 的 Annex-B 格式无效。验证 CPD 格式是否有效 Annex-B。如果不是，则移除 CPD Annex-B 适应标志。
0x32000011	STATUS_MKV_PTS_DTS_ARE_NOT_SAME	Kinesis Video Streams 强制片段起始帧的 PTS (演示时间戳) 和 DTS (解码时间戳) 相同。这些是片段开头的关键帧。
0x32000012	STATUS_MKV_INVALID_H264_H265_CPD	无法解析 H264/H265 编解码器私有数据。
0x32000013	STATUS_MKV_INVALID_H264_H265_SPS_WIDTH	无法从编解码器私有数据中提取宽度。
0x32000014	STATUS_MKV_INVALID_H264_H265_SPS_HEIGHT	无法从编解码器私有数据中提取高度。
0x32000015	STATUS_MKV_INVALID_H264_H265_SPS_NALU	H264/H265 SPS NALU 无效。

代码	消息	描述/建议的操作
0x32000016	STATUS_MKV_INVALID_BIH_CPD	编解码器私有数据中的位图信息标头格式无效。
0x32000017	STATUS_MKV_INVALID_HEVC_NALU_COUNT	高效高级音频编码 (HEVC) 网络抽象层单元 (NALU) 计数无效。
0x32000018	STATUS_MKV_INVALID_HEVC_FORMAT	HEVC 格式无效。
0x32000019	STATUS_MKV_HEVC_SPS_NALU_MISSING	序列参数集 (SPS) 中缺少 HEVC NALU。
0x3200001a	STATUS_MKV_INVALID_HEVC_SPS_NALU_SIZE	HEVC SPS NALU 大小无效。
0x3200001b	STATUS_MKV_INVALID_HEVC_SPS_CHROMA_FORMAT_IDC	色度格式 IDC 无效。
0x3200001c	STATUS_MKV_INVALID_HEVC_SPS_RESERVED	HEVC 保留 SPS 无效。
0x3200001d	STATUS_MKV_MIN_ANNEX_B_CPD_SIZE	AnnexBb 编解码器私有测试版的最小值大小。对于 H264，该值必须等于或大于 11。对于 H265，该值必须等于或大于 15。
0x3200001e	STATUS_MKV_ANNEXB_CPD_MISSING_NALUS	Annex-B NALU 中缺少编解码器专用数据。
0x3200001f	STATUS_MKV_INVALID_ANNEXB_CPD_NALUS	Annex-B NALU 中的编解码器封闭测试版无效。
0x32000020	STATUS_MKV_INVALID_TAG_NAME_LENGTH	标记名称长度无效。有效值大于 0 且小于 128。

代码	消息	描述/建议的操作
0x32000021	STATUS_MKV_INVALID_TAG_VALUE_LENGTH	标记值长度无效。有效值大于零且小于 256。
0x32000022	STATUS_MKV_INVALID_GENERATOR_STATE_TAGS	生成器状态标签无效。
0x32000023	STATUS_MKV_INVALID_AAC_CPD_SAMPLING_FREQUENCY_INDEX	AAC 编解码器专用数据采样频率索引无效。
0x32000024	STATUS_MKV_INVALID_AAC_CPD_CHANNEL_CONFIG	AAC 编解码器专用数据通道配置无效。
0x32000025	STATUS_MKV_INVALID_AAC_CPD	AAC 编解码器专用数据无效。
0x32000026	STATUS_MKV_TRACK_INFO_NOT_FOUND	找不到轨道信息。
0x32000027	STATUS_MKV_INVALID_SEGMENT_UUID	片段 UUID 无效。
0x32000028	STATUS_MKV_INVALID_TRACK_UID	轨道 UID 无效。
0x32000029	STATUS_MKV_INVALID_CLIENT_ID_LENGTH	
0x3200002a	STATUS_MKV_INVALID_AMS_ACM_CPD	
0x3200002b	STATUS_MKV_MISSING_SPS_FROM_H264_CPD	

代码	消息	描述/建议的操作
0x3200002c	STATUS_MKV_MISSING_PPS_FROM_H264_CPD	
0x3200002d	STATUS_MKV_INVALID_PARENT_TYPE	

Trace 库返回的错误和状态码

下表包含Trace库中方法返回的错误和状态信息。

代码	消息
0x10100001	STATUS_MIN_PROFILER_BUFFER

Utils 库返回的错误和状态码

下表包含Utils库中方法返回的错误和状态信息。

代码	消息
0x40000001	STATUS_INVALID_BASE64_ENCODE
0x40000002	STATUS_INVALID_BASE
0x40000003	STATUS_INVALID_DIGIT
0x40000004	STATUS_INT_OVERFLOW
0x40000005	STATUS_EMPTY_STRING
0x40000006	STATUS_DIRECTORY_OPEN_FAILED
0x40000007	STATUS_PATH_TOO_LONG
0x40000008	STATUS_UNKNOWN_DIR_ENTRY_TYPE
0x40000009	STATUS_REMOVE_DIRECTORY_FAILED

代码	消息
0x4000000a	STATUS_REMOVE_FILE_FAILED
0x4000000b	STATUS_REMOVE_LINK_FAILED
0x4000000c	STATUS_DIRECTORY_ACCESS_DENIED
0x4000000d	STATUS_DIRECTORY_MISSING_PATH
0x4000000e	STATUS_DIRECTORY_ENTRY_STAT_ERROR
0x4000000f	STATUS_STRFTIME_FAILED
0x40000010	STATUS_MAX_TIMESTAMP_FORMAT_STR_LEN_EXCEEDED
0x40000011	STATUS_UTIL_MAX_TAG_COUNT
0x40000012	STATUS_UTIL_INVALID_TAG_VERSION
0x40000013	STATUS_UTIL_TAGS_COUNT_NON_ZERO_TAGS_NULL
0x40000014	STATUS_UTIL_INVALID_TAG_NAME_LEN
0x40000015	STATUS_UTIL_INVALID_TAG_VALUE_LEN
0x4000002a	STATUS_EXPONENTIAL_BACKOFF_INVALID_STATE
0x4000002b	STATUS_EXPONENTIAL_BACKOFF_RETRIES_EXHAUSTED
0x4000002c	STATUS_THREADPOOL_MAX_COUNT
0x4000002d	STATUS_THREADPOOL_INTERNAL_ERROR
0x40100001	STATUS_HASH_KEY_NOT_PRESENT

代码	消息
0x40100002	STATUS_HASH_KEY_ALREADY_PRESENT
0x40100003	STATUS_HASH_ENTRY_ITERATION_ABORT
0x41000001	STATUS_BIT_READER_OUT_OF_RANGE
0x41000002	STATUS_BIT_READER_INVALID_SIZE
0x41100001	STATUS_TIMER_QUEUE_STOP_SCHEDULING
0x41100002	STATUS_INVALID_TIMER_COUNT_VALUE
0x41100003	STATUS_INVALID_TIMER_PERIOD_VALUE
0x41100004	STATUS_MAX_TIMER_COUNT_REACHED
0x41100005	STATUS_TIMER_QUEUE_SHUTDOWN
0x41200001	STATUS_SEMAPHORE_OPERATION_AFTER_SHUTDOWN
0x41200002	STATUS_SEMAPHORE_ACQUIRE_WHEN_LOCKED
0x41300001	STATUS_FILE_LOGGER_INDEX_FILE_INVALID_SIZE

View 库返回的错误和状态码

下表包含View库中方法返回的错误和状态信息。

代码	消息	描述
0x30000001	STATUS_MIN_CONTENT_VIEW_ITEMS	指定的内容视图项目数无效。有关更多信息，请参阅 制作人 SDK 限制 ：
0x30000002	STATUS_INVALID_CONTENT_VIEW_DURATION	指定的内容视图持续时间无效。有关更多信息，请参阅 制作人 SDK 限制 ：
0x30000003	STATUS_CONTENT_VIEW_NO_MORE_ITEMS	尝试获取的位置超过标头。
0x30000004	STATUS_CONTENT_VIEW_INVALID_INDEX	指定的索引无效。
0x30000005	STATUS_CONTENT_VIEW_INVALID_TIMESTAMP	具有无效的时间戳，或者时间戳发生重叠。帧解码时间戳应大于或等于前一帧的时间戳，再加上前一帧的持续时间： $\text{DTS}(n) \geq \text{DTS}(n-1) + \text{Duration}(n-1)$ 。该错误通常表示“不稳定”编码器。编码器生成一组突发编码帧，并且其时间戳小于帧内的持续时间。或者，流配置为使用 SDK 时间戳，并且发送帧的速度比帧持续时间快。为了帮助解决编码器中的一些“抖动”问题，请在 <code>StreamInfo.StreamCaps</code> 结构中指定较小的帧持续时间。例如，如果视频流为 25 FPS，则每帧的持续时间为 40 毫秒。但是，为了处理编码器的“抖动”，我们建议您使用该帧持续时间的一半（20 毫

代码	消息	描述
		秒)。某些流要求更精确地控制错误检测时间。
0x30000006	STATUS_INVALID_CONTENT_VIEW_LENGTH	指定的内容视图项目数据长度无效。

PutFrame 回调返回的错误和状态码-C 制作人库

以下部分包含 C producer 库中PutFrame操作的回调返回的错误和状态信息。

代码	消息	描述	推荐操作
0x15000001	STATUS_STOP_CALLBACK_CHAIN	回调链已停止。	
0x15000002	STATUS_MAX_CALLBACK_CHAIN	已达到最大回调链数。	
0x15000003	STATUS_INVALID_PLATFORM_CALLBACKS_VERSION	PlatformCallbacks 结构版本无效。	指定正确的当前结构版本。
0x15000004	STATUS_INVALID_PRODUCER_CALLBACKS_VERSION	ProducerCallbacks 结构版本无效。	指定正确的当前结构版本。
0x15000005	STATUS_INVALID_STREAM_CALLBACKS_VERSION	StreamCallbacks 结构版本无效。	指定正确的当前结构版本。
0x15000006	STATUS_INVALID_AUTH_CALLBACKS_VERSION	AuthCallbacks 结构版本无效。	指定正确的当前结构版本。

代码	消息	描述	推荐操作
	H_CALLBACKS_VERSION		
0x15000007	STATUS_INVALID_API_CALLBACKS_VERSION	ApiCallbacks 结构版本无效。	指定正确的当前结构版本。
0x15000008	STATUS_INVALID_AWS_CREDENTIALS_VERSION	AwsCredentials 结构版本无效。	指定正确的当前结构版本。
0x15000009	STATUS_MAXIMUM_REQUEST_HEADER_COUNT	已达到最大请求标头数。	
0x1500000a	STATUS_MAXIMUM_REQUEST_HEADER_NAME_LEN	已达到最大请求标头名称长度。	
0x1500000b	STATUS_MAXIMUM_REQUEST_HEADER_VALUE_LEN	已达到最大请求标头值长度。	
0x1500000c	STATUS_INVALID_API_CALL_RETURN_JSON	API 调用的返回 JSON 无效。	
0x1500000d	STATUS_CURL_INIT_FAILED	Curl 初始化失败。	
0x1500000e	STATUS_CURL_LIBRARY_INIT_FAILED	Curl lib 初始化失败。	

代码	消息	描述	推荐操作
0x1500000f	STATUS_INVALID_DESCRIBE_STREAM_RETURN_JSON	的返回 JSON 无效的 DescribeStream。	
0x15000010	STATUS_HMAC_GENERATION_ERROR	HMAC 生成错误。	
0x15000011	STATUS_IOT_FAILED	物联网授权失败。	
0x15000012	STATUS_MAXIMUM_ROLE_ALIAS_LENGTH_EXCEEDED	已达到最大角色别名长度。	请指定较短的别名长度。
0x15000013	STATUS_MAXIMUM_USER_AGENT_NAME_POSTFIX_LENGTH_EXCEEDED	已达到最大代理名称后缀长度。	
0x15000014	STATUS_MAXIMUM_CUSTOM_USER_AGENT_LENGTH_EXCEEDED	已达到最大客户用户代理长度。	
0x15000015	STATUS_INVALID_USER_AGENT_LENGTH	无效的用户代理长度。	
0x15000016	STATUS_INVALID_ENDPOINT_CACHING_PERIOD	无效的终端节点缓存期。	指定小于 24 小时的缓存期。

代码	消息	描述	推荐操作
0x15000017	STATUS_IOT_EXPIRATION_OCCURS_IN_PAST	IoT 过期时间戳发生在过去。	
0x15000018	STATUS_IOT_EXPIRATION_PARSING_FAILED	物联网过期解析失败。	
0x15000019	STATUS_DUPLICATE_PRODUCER_CALLBACK_FREE_FUNC		
0x1500001a	STATUS_DUPLICATE_STREAM_CALLBACK_FREE_FUNC		
0x1500001b	STATUS_DUPLICATE_AUTH_CALLBACK_FREE_FUNC		
0x1500001c	STATUS_DUPLICATE_API_CALLBACK_FREE_FUNC		
0x1500001d	STATUS_FILE_LOGGER_INDEX_FILE_TOO_LARGE		

代码	消息	描述	推荐操作
0x1500001e	STATUS_MAX_IOT_THING_NAME_LENGTH		
0x1500001f	STATUS_IOT_CREATE_LWS_CONTEXT_FAILED		
0x15000020	STATUS_INVALID_CERT_PATH		
0x15000022	STATUS_FILE_CREDENTIAL_OPEN_FILE_FAILED		
0x15000023	STATUS_FILE_CREDENTIAL_INVALID_FILE_LENGTH		
0x15000024	STATUS_FILE_CREDENTIAL_INVALID_FILE_FORMAT		
0x15000026	STATUS_STREAM_BEING_SHUTDOWN		

代码	消息	描述	推荐操作
0x15000027	STATUS_CLIENT_BEING_SHUTDOWN		
0x15000028	STATUS_CONTINUOUS_RETRY_RESET_FAILED		
0x16000001	STATUS_CURL_PERFORMANCE_FAILED		
0x16000002	STATUS_IOT_INVALID_RESPONSE_LENGTH		
0x16000003	STATUS_IOT_NULL_AWS_CREDS		
0x16000004	STATUS_IOT_INVALID_URI_LEN		
0x16000005	STATUS_TIMESTAMP_STRING_UNRECOGNIZED_FORMAT		

网络抽象层 (NAL) 适配标记参考

本节包含有关 `StreamInfo.NalAdaptationFlags` 枚举的可用标记的信息。

应用程序中的[基本流](#)可以采用 Annex-B 或 AVCC 格式：

- Annex-B 格式使用两个字节的零划分 [NALU \(网络抽象层单元\)](#)，后跟一或三个字节的零，然后是数字 1 (称为起始码，例如 00000001)。
- AVCC 格式也对 NALU 进行了包装，但每个 NALU 前都有一个表示 NALU 大小 (通常为 4 个字节) 的值。

许多编码器生成 Annex-B 比特流格式。一些较高级的比特流处理器 (例如播放引擎或 [中的](#) Media Source Extensions (MSE)AWS Management Console 播放器) 则为其帧使用 AVCC 格式。

编解码器私有数据 (CPD) 是面向 H.264 编解码器的 SPS/PPS (序列参数集/图片参数集)，也可以采用 Annex-B 或 AVCC 格式。但是，对 CPD 而言，格式与前面描述的有所不同。

这些标记告知 SDK 将 NALU 与帧数据和 CPD 的 AVCC 或 Annex-B 适配，如下所示：

标记	适配
NAL_ADAPTATION_FLANG_NONE	没有改编。
NAL_ADAPTATION_ANNEXB_NALS	将 Annex-B Nalus 改编为 AVCC NalU。
NAL_ADAPTATION_AVCC_NALS	将 AVCC Nalus 改编成附件 B NalU。
NAL_ADAPTATION_ANNEXB_CPD_NALS	将编解码器私有数据的 Annex-B Nalus 改编为 AVCC 格式的 nalU。
NAL_ADAPTATION_ANNEXB_CPD_AND_FRAME_NALS	调整编解码器的 Annex-B Nalus，将私有数据帧转换为 AVCC 格式的 nalU。

有关 NALU 类型的更多信息，请参阅 RFC 3984 中的 [Section 1.3: Network Abstraction Layer Unit Types](#)。

制作人 SDK 结构

本节包含有关可用于向 Kinesis Video Streams Producer 对象提供数据的结构的信息。

主题

- [DeviceInfo/DefaultDeviceInfoProvider](#)
- [StorageInfo](#)

DeviceInfo/DefaultDeviceInfoProvider

DeviceInfo和DefaultDeviceInfoProvider对象控制 Kinesis Video Streams Producer 对象的行为。

成员字段

- version-一个整数值，用于确保在当前版本的代码库中使用正确的结构版本。当前版本使用 DEVICE_INFO_CURRENT_VERSION 宏指定。
- name — 设备的人类可读名称。
- tagCount/tags — 当前未使用。
- StreamCount — 设备可以处理的最大直播数量。这会为最初指向流对象的指针预分配存储，但实际流对象将稍后创建。默认值为 16 个流，但可在 DefaultDeviceInfoProvider.cpp 文件中更改此数字。
- storageInfo : 描述主存储配置的对象。有关更多信息，请参阅[StorageInfo](#) :

StorageInfo

指定 Kinesis Video Streams 的主存储配置。

默认实施基于低片段快速堆实施，这一实施针对流式处理进行了优化。它使用 MEMALLOC 分配器，可以在指定平台上覆盖。一些平台具有虚拟内存分配，没有物理页分配的支持。由于使用了内存，虚拟页由物理页提供支持。在存储利用不充分时，这会导致整个系统面临内存不足压力。

根据以下公式计算默认存储大小。DefragmentationFactor 应设置为 1.2 (20%)。

```
Size = NumberOfStreams * AverageFrameSize * FramesPerSecond * BufferDurationInSeconds *  
DefragmentationFactor
```

在以下示例中，设备具有音频和视频流。音频流每秒采样 512 次，平均样本 100 字节。视频流每秒 25 帧，平均 10000 字节。每个流的缓冲时长为 3 分钟。

```
Size = (512 * 100 * (3 * 60) + 25 * 10000 * (3 * 60)) * 1.2 = (9216000 + 45000000) *  
1.2 = 65059200 = ~ 66MB.
```

如果设备有更多可用内存，我们建议您在存储空间中添加更多内存，以避免出现严重的碎片。

验证存储大小是否足以容纳编码复杂度高（当帧大小因高动态而变大时）或带宽较低时的所有流的完整缓冲区。如果生产者达到内存压力，它就会发出存储溢出压力回调 (`StorageOverflowPressureFunc`)。但是，当内容存储中没有可用内存时，它会丢弃正在推送到 Kinesis Video Streams 的帧并显示错误 `STATUS_STORE_OUT_OF_MEMORY = 0x5200002e` ()。有关更多信息，请参阅[客户端库返回的错误和状态码](#)：如果应用程序确认 (ACK) 不可用或者持久 ACK 延迟，也会出现这种情况。在这种情况下，在前一帧开始丢失之前，缓冲区将填充到“缓冲持续时间”容量。

成员字段

- `version`-一个整数值，用于确保在当前版本的代码库中使用正确的结构版本。
- `StorageType` — `DEVICE_STORAGE_TYPE` 一个枚举，用于指定存储的底层支持和实现。目前唯一支持的值是 `DEVICE_STORAGE_TYPE_IN_MEM`。未来的实施中将支持 `DEVICE_STORAGE_TYPE_HYBRID_FILE`，指示存储回退到由文件支持的内容存储。
- `StorageSize`-要预分配的存储大小（以字节为单位）。最小分配大小为 10 MB，最大为 10 GB。（在未来实施由文件支持的内容存储时，这会有所更改。）
- `spillRatio` — 一个整数值，表示要从直接内存存储类型 (RAM) 与二级溢出存储（文件存储）中分配的存储空间的百分比。当前未使用。
- `rootDirectory`：由文件支持的内容存储所在目录的路径。当前未使用。

Kinesis 视频流结构

您可以使用以下结构向 Kinesis 视频流的实例提供数据。

主题

- [StreamDefinition/StreamInfo](#)
- [ClientMetrics](#)
- [StreamMetrics](#)

StreamDefinition/StreamInfo

C++ 层中的 `StreamDefinition` 对象在独立于平台的代码中封装 `StreamInfo` 对象，并在构造函数中提供一些默认值。

成员字段

字段	数据类型	描述	默认值
stream_name	string	可选的流名称。有关流名称长度的更多信息，请参阅 制作人 SDK 限制 。每个流应具有唯一的名称。	如果未指定名称，则将随机生成一个名称。
retention_period	duration<uint64_t, ratio<3600>>	流的保留期，以秒为单位。指定 0 表示不保留。	3600 (1 小时)
tags	const map<string, string>*	包含用户信息的键-值对的映射。如果流已有一组标志，则新标志将附加到现有一组标志之后。	无标签
kms_key_id	string	要用于加密流的 AWS KMS 密钥 ID。有关更多信息，请参阅 Kinesis Video Streams 中的数据保护 ：	默认 KMS 密钥 (aws/kinesis-video)。
streaming_type	STREAMING_TYPE 枚举	STREAMING_TYPE_REALTIME 是唯一受支持的值。	
content_type	string	流的内容格式。Kinesis Video Streams 控制台可以播放 video/h264 该格式的内容。	video/h264

字段	数据类型	描述	默认值
max_latency	duration< uint64_t, milli>	数据流的最大延迟，以毫秒为单位。当缓冲持续时间超过此时间量时，将调用流延迟压力回调（如果指定）。指定 0 表示将不调用流延迟压力回调。	milliseconds::zero()
fragment_duration	duration< uint64_t>	所需的片段持续时间，以秒为单位。此值与 key_frame_fragmentation 值结合使用。如果此值为 false，Kinesis Video Streams 将在该持续时间过去后在关键帧上生成片段。例如，高级音频编码 (AAC) 音频流将每个帧作为关键帧。指定 key_frame_fragmentation = false 将导致在经过此持续时间之后，在关键帧上片段化，生成 2 秒的片段。	2

字段	数据类型	描述	默认值
timecode_scale	duration<uint64_t, milli>	MKV 时间码标度以毫秒为单位，这指定 MKV 集簇中帧的时间码粒度。MKV 帧时间码始终相对于集簇的开始。MKV 使用一个有符号的 16 位值 (0-32767) 来表示集簇内的时间码 (片段)。验证帧时间码是否可以用给定的时间码比例来表示。默认时间码标度值 1 毫秒确保可以表示的最大帧为 32767 ms ~ = 32 秒。这是在 Kinesis Video Streams 服务配额 指定的最大片段持续时间 (10 秒) 内。	1
key_frame_fragmentation	bool	是否在关键帧上生成片段。如果为 true，则开发工具包将在每次出现关键帧时生成片段的开始。如果 false，Kinesis Video Streams fragmentation 至少会等待，然后在紧随其后的关键帧上生成一个新的片段。	true

字段	数据类型	描述	默认值
frame_timecodes	bool	是否使用帧时间码或者使用当前时间回调生成时间戳。许多编码器不在帧中生成时间戳。因此，指定false此参数可确保帧在放入 Kinesis Video Streams 时带有时间戳。	true
absolute_fragment_times	bool	Kinesis Video Streams 使用 MKV 作为其底层打包机制。MKV 规范具有严格的帧时间码，相对于集簇（片段）的开始位置，但集簇时间码可以是绝对的，也可以相对于流的开始时间。如果时间戳是相对的，PutMedia 服务 API 调用将使用可选的流开始时间戳并调整集群时间戳。该服务始终存储片段及其绝对时间戳。	true
fragment_acks	bool	是否接收应用程序级片段 ACK（确认）。	true 意味着开发工具包接收 ACK 并相应操作。
restart_on_error	bool	是否在出现特定错误时重启。	true 意味着开发工具包在出现任意错误时将尝试重新启动流。

字段	数据类型	描述	默认值
recalculate_metrics	bool	是否重新计算指标。每个检索度量的调用可以重新计算这些值，以获取最新的“正在运行”值，这可能会导致较小的 CPU 影响。在极低功耗/容量的设备上，可能需要将此值设置为 false 以避免占用 CPU 周期。否则，我们不建议使用 false 此值。	true

字段	数据类型	描述	默认值
nal_adaptation_flags	uint32_t	<p>指定网络抽象层单元 (NALU) 自适应标志。如果比特流为 H.264 编码，则可以将其按原始数据或包装为 NALU 进行处理。后者可以为 Annex-B 或 AVCC 格式。大多数基本流制作者和使用者 (读取编码器和解码器) 都使用 Annex-B 格式，因为它具有诸如错误恢复等优点。高级系统使用 AVCC 格式，这是 MPEG、HLS、DASH 等的默认格式。控制台播放使用浏览器的 MSE (Media Source Extensions) 来解码和播放使用 AVCC 格式的流。对于 H.264 (以及对于 M-JPEG 和 H.265)，开发工具包提供自适应功能。</p> <p>许多基本流采用以下格式。在此示例中，Ab 是 Annex-B 开始代码 (001 或 0001)。</p> <pre>Ab(Sps)Ab (Pps)Ab(I- frame)Ab(P/B- frame) Ab(P/B-fr ame)... Ab(Sps)Ab</pre>	默认情况下，帧数据和编解码器私有数据均为从 Annex-B 格式更改为 AVCC 格式。

字段	数据类型	描述	默认值
		<p>(Pps)Ab(I-frame)Ab(P/B-frame) Ab(P/B-frame)</p> <p>对于 H.264，编解码器私有数据 (CPD) 位于 SPS (序列参数集) 和 PPS (图像参数集) 参数中，并且可以调整为 AVCC 格式。除非媒体管道单独提供 CPD，否则应用程序可以从帧中提取 CPD。它可以通过查找第一个 IDR 帧 (其中应包含 SPS 和 PPS)，提取两个 NALU (它们是 Ab(Sps)Ab(Pps))，然后将其设置在 CPD 中来实现。StreamDefinition</p> <p>有关更多信息，请参阅 NAL 适配标志：</p>	
frame_rate	uint32_t	预期的帧率。此值用于更好地计算缓冲需求。	25
avg_bandwidth_bps	uint32_t	流的预期平均带宽。此值用于更好地计算缓冲需求。	4 * 1024 * 1024

字段	数据类型	描述	默认值
buffer_duration	duration<uint64_t>	流缓冲持续时间，以秒为单位。SDK 将帧保留在内容存储中的时间最长buffer_duration，之后随着窗口向前移动，之前的帧会被丢弃。如果被丢弃的帧尚未发送到后端，则会调用丢弃的帧回调。如果当前缓冲区持续时间大于max_latency，则将调用流延迟压力回调。收到片段持久ACK时，缓冲区将修剪到下一个片段开始位置。这指示内容已持久保留到云中，因此不再需要在本地设备上存储内容。	120

字段	数据类型	描述	默认值
replay_duration	duration< uint64_t>	如果启用了重新启动，则在出错期间向后滚动当前读取器进行重播的持续时间（以秒为单位）。回滚操作将在缓冲开始位置停止（在刚启动流式传输时，或者出现持久 ACK 时）。回滚将尝试停留在指示片段开始的关键帧上。如果“导致重启”的错误并不表示主机已死亡（主机仍处于活动状态且其内部缓冲区中包含帧数据），则回滚将在最后收到的 ACK 帧处停止。然后，它滚动到下一个关键帧，因为整个片段已经存储在主机内存中。	40
connection_staleness	duration< uint64_t>	如果 SDK 未收到缓冲 ACK，则调用直播陈旧回调的时间（以秒为单位）。它表示帧是从设备发送的，但后端没有确认这些帧。这表明在中间跳转时或者负载均衡器上出现了断开的连接。	30

字段	数据类型	描述	默认值
codec_id	string	MKV 音轨的编解码器 ID。	V_MPEG4/ISO/AVC
track_name	string	MKV 音轨名称。	kinesis_video
codecPrivateData	unsigned char*	编解码器私有数据 (CPD) 缓冲区。如果媒体管道在流启动之前具有 CPD 的相关信息，则可以在 StreamDefinition.codecPrivateData 中发送。此时将复制位，在创建流的调用之后可以重用或释放缓冲区。但是，如果创建流时数据不可用，则可以在KinesisVideoStream.start(cpd) 函数的其中一个重载中对其进行设置。	null
codecPrivateData大小	uint32_t	编解码器私有数据缓冲区大小。	0

ClientMetrics

通过调用来填充ClientMetrics对象getKinesisVideoMetrics。

成员字段

字段	数据类型	描述
version	UINT32	结构的版本，在 CLIENT_METRICS_CURRENT_VERSION 宏中定义。
contentStoreSize	UINT64	整体内容存储大小，以字节为单位。这是在 DeviceInfo.StorageInfo.storageSize 中指定的值。
contentStoreAvailable大小	UINT64	当前可用存储大小（以字节为单位）。
contentStoreAllocated大小	UINT64	当前分配的大小。由于内部记账和内存存储的实施，分配大小 + 可用大小应略小于总存储大小。
totalContentViews大小	UINT64	所有流的所有内容视图的已分配内存大小。这不计入存储大小。此内存使用 MEMALLOC 宏分配，可以覆盖该值以提供自定义分配器。
totalFrameRate	UINT64	在所有流上观察到的总帧率。
totalTransferRate	UINT64	在所有流上观察到的总流速率，以每秒字节数为单位。

StreamMetrics

通过调用来填充StreamMetrics对象getKinesisVideoMetrics。

成员字段

字段	数据类型	描述
version	UINT32	结构的版本，在 STREAM_METRICS_CURRENT_VERSION 宏中定义。
currentViewDuration	UINT64	累积帧的时间长度。在快速联网的情况下，此持续时间要么为零，要么为帧持续时间（在传输帧时）。如果持续时间长于中max_latency 指定的时间StreamDefinition，则会调用直播延迟回调（如果已指定）。持续时间以 100 纳秒为单位指定，这是 PIC 层的默认时间单位。
overallViewDuration	UINT64	整体查看持续时间。如果直播配置为没有 ACK 或持久性，则该值会随着帧放入 Kinesis 视频流而增长，并变为等于buffer_duration 中的。StreamDefinition 启用 ACK 并收到持久的 ACK 后，缓冲区会被修剪到下一个关键帧。这是因为 ACK 时间戳表示整个片段的开头。持续时间以 100 纳秒为单位指定，这是 PIC 层的默认时间单位。
currentViewSize	UINT64	当前缓冲区的大小，以字节为单位。
overallViewSize	UINT64	整体视图大小，以字节为单位。

字段	数据类型	描述
currentFrameRate	UINT64	当前流观察到的帧率。
currentTransferRate	UINT64	当前流观察到的传输速率，以每秒字节数为单位。

制作人 SDK 回调

亚马逊 Kinesis Video Streams Video Streams Producer SDK 中的类和方法不维护自己的流程。相反，它们使用传入的函数调用和事件来安排用于与应用程序通信的回调。

应用程序可以使用两种回调模式来与开发工具包交互：

- [CallbackProvider](#)— 此对象公开了从独立于平台的代码 (PIC) 组件到应用程序的所有回调。此模式允许使用全部功能，但这也意味着实施必须处理 C++ 层中的所有公有 API 方法和签名。
- [StreamCallbackProvider](#) 和 [ClientCallbackProvider](#) — 这些对象公开了特定于流和客户端特定的回调，SDK 的 C++ 层公开了其余的回调。这是用于与创建者开发工具包交互的首选回调模式。

下图说明了回调对象的对象模型：

在上图中，[DefaultCallbackProvider](#) 派生自 [CallbackProvider](#) (后者公开 PIC 中的所有回调) 并包含 [StreamCallbackProvider](#) 和 [ClientCallbackProvider](#)。

本主题包含下列部分：

- [ClientCallbackProvider](#)
- [StreamCallbackProvider](#)
- [ClientCallbacks 结构](#)
- [用于重试直播的回调实现](#)

ClientCallbackProvider

[ClientCallbackProvider](#) 对象可公开客户端级别的回调函数。[ClientCallbacks](#) 中介绍了这些函数的详细信息。

回调方法：

- `getClientReadyCallback`— 报告客户端的就绪状态。
- `getStorageOverflowPressureCallback`— 报告存储溢出或压力。当存储利用率下降到 `STORAGE_PRESSURE_NOTIFICATION_THRESHOLD` 值（该值为总体存储大小的 5%）以下时，将调用此回调。有关更多信息，请参阅[StorageInfo](#)：

StreamCallbackProvider

`StreamCallbackProvider` 对象可公开流级别的回调函数。

回调方法：

- `getDroppedFragmentReportCallback`：报告已丢弃片段。
- `getDroppedFrameReportCallback`— 报告丢帧。
- `getFragmentAckReceivedCallback`— 报告已收到该流的片段 ACK。
- `getStreamClosedCallback`— 报告直播关闭情况。
- `getStreamConnectionStaleCallback`— 报告过时的连接状况。在这种情况下，生产者正在向服务发送数据，但没有收到确认。
- `getStreamDataAvailableCallback`— 报告数据流中可用。
- `getStreamErrorReportCallback`— 报告直播错误情况。
- `getStreamLatencyPressureCallback`— 报告流延迟情况，即累积的缓冲区大小大于该 `max_latency` 值。有关更多信息，请参阅[StreamDefinition/StreamInfo](#)：
- `getStreamReadyCallback`: —报告直播就绪状态。
- `getStreamUnderflowReportCallback`— 报告直播下溢情况。此函数目前未使用，保留供将来使用。

有关源代码 `StreamCallbackProvider`，请参阅 [StreamCallbackProvider.h](#)。

ClientCallbacks 结构

该 `ClientCallbacks` 结构包含 PIC 在发生特定事件时调用的回调函数入口点。此结构还在 `CALLBACKS_CURRENT_VERSION` 字段中包含版本信息，还有一个 `customData` 字段用于提供各个回调函数返回的用户定义数据。

客户端应用程序可将 `this` 指针用于 `custom_data` 字段，以在运行时将成员函数映射到静态 `ClientCallback` 函数，如以下代码示例所示：

```

STATUS TestStreamCallbackProvider::streamClosedHandler(UINT64 custom_data,
    STREAM_HANDLE stream_handle, UINT64 stream_upload_handle) {
    LOG_INFO("Reporting stream stopped.");

    TestStreamCallbackProvider* streamCallbackProvider =
        reinterpret_cast<TestStreamCallbackProvider*> (custom_data);
    streamCallbackProvider->streamClosedHandler(...);
}

```

事件

功能	描述	类型
CreateDeviceFunc	目前未在后端实施。在从 Java 或 C++ 调用时，此调用将失败。其他客户端执行特定于平台的初始化。	后端 API
CreateStreamFunc	在创建流时调用。	后端 API
DescribeStreamFunc	在调用 DescribeStream 时调用。	后端 API
GetStreamingEndpointFunc	在调用 GetStreamingEndpoint 时调用。	后端 API
GetStreamingTokenFunc	在调用 GetStreamingToken 时调用。	后端 API
PutStreamFunc	在调用 PutStream 时调用。	后端 API
TagResourceFunc	在调用 TagResource 时调用。	后端 API
CreateMutexFunc	创建同步互斥锁。	同步
FreeMutexFunc	释放互斥锁。	同步
LockMutexFunc	锁定同步互斥锁。	同步

功能	描述	类型
TryLockMutexFunc	尝试锁定互斥锁。当前未实施。	同步
UnlockMutexFunc	解锁互斥锁。	同步
ClientReadyFunc	在客户端进入就绪状态时调用。	Notification
DroppedFrameReportFunc	在丢弃帧时报告。	Notification
DroppedFragmentReportFunc	在丢弃片段时报告。此函数目前未使用，保留供将来使用。	Notification
FragmentAckReceivedFunc	在收到片段 ACK (缓冲、接收、持久存在和错误) 时调用。	Notification
StorageOverflowPressureFunc	在存储利用率下降到 STORAGE_PRESSURE_NOTIFICATION_THRESHOLD 值 (该值定义为总体存储大小的 5%) 以下时调用。	Notification
StreamClosedFunc	在流式处理其余帧的最后几个比特时调用。	Notification
StreamConnectionStaleFunc	在流进入过时连接状态时调用。在这种情况下，创建者会向服务发送数据，但收不到确认。	Notification
StreamDataAvailableFunc	在流数据可用时调用。	Notification

功能	描述	类型
StreamErrorReportFunc	在出现流错误时调用。在这种情况下，PIC 会自动关闭流。	Notification
StreamLatencyPressureFunc	在流进入延迟状况（即累积缓冲区大小大于 <code>max_latency</code> 值时）调用。有关更多信息，请参阅 StreamDefinition/StreamInfo ：	Notification
StreamReadyFunc	在流进入就绪状态时调用。	Notification
StreamUnderflowReportFunc	此函数目前未使用，保留供将来使用。	Notification
DeviceCertToTokenFunc	以令牌形式返回连接证书。	平台集成
GetCurrentTimeFunc	返回当前时间。	平台集成
GetDeviceCertificateFunc	返回设备证书。此函数目前未使用，保留供将来使用。	平台集成
GetDeviceFingerprintFunc	返回设备指纹。此函数目前未使用，保留供将来使用。	平台集成
GetRandomNumberFunc	返回 0 和 <code>RAND_MAX</code> 之间的随机数。	平台集成
GetSecurityTokenFunc	返回传递给与后端 API 通信的函数的安全令牌。该实施可以指定序列化的 <code>AccessKeyId</code> 、 <code>SecretKeyId</code> 和会话令牌。	平台集成

功能	描述	类型
LogPrintFunc	记录带有标签和日志级别的一行文本。有关更多信息，请参阅PlatformUtils.h：	平台集成

对于上表中的平台集成函数，最后一个参数是 ServiceCallContext 结构，该结构具有以下字段：

- version：结构的版本。
- callAfter：调用函数等待的绝对时间。
- timeout：操作超时，以 100 纳秒为单位。
- customData：要传递回客户端的用户定义的值。
- pAuthInfo：调用的凭证。有关更多信息，请参阅下面的 (__AuthInfo) 结构。

使用 __AuthInfo 结构提供授权信息，该信息可以是序列化凭证或特定于提供商的身份验证令牌。此结构具有以下字段：

- version：__AuthInfo 结构的版本。
- type：用于定义凭证类型（证书或安全令牌）的 AUTH_INFO_TYPE 值。
- data：包含身份验证信息的字节数组。
- size：data 参数的大小。
- expiration：凭证的过期时间（以 100 纳秒为单位）。

用于重试直播的回调实现

Kinesis Video Producer 开发工具包通过回调函数提供流式处理的状态。我们建议您实现以下回调机制，以从直播期间遇到的任何短暂网络问题中恢复过来。

- 直播延迟压力回调-此回调机制在 SDK 遇到直播延迟情况时启动。这在累计缓冲区大小大于 MAX_LATENCY 值时发生。在创建流后，流式处理应用程序会将 MAX_LATENCY 设置为 60 秒（默认值）。此回调的典型实施是重置连接。你可以根据需要在 <https://github.com/aws-labs/amazon-kinesis-video-streams-producer-sdk-cpp/blob/master/src/src/sroce/kinesis-video-c-producer.c> 上使用示例实现。StreamLatencyStateMachine 请注意，无法将由于网络中断而未交付的帧存储到辅助存储器中进行回填。

- 直播失效回调 ——当创建者可以向 Amazon Kinesis Data Streams 服务 (上行链路) 发送数据，但无法及时获取确认 (缓冲后的 ACK) (默认为 60 秒) 时，就会启动此回调。根据网络设置，可以启动直播延迟压力回调或直播陈旧回调，或者两者兼而有之。与流延迟压力回调重试实施相似，典型实施是重置连接并启动新的连接以进行流式处理。你可以根据需要在 <https://github.com/aws-labs/amazon-kinesis-video-streams-producer-c/blob/master/src/src/ConnectionStaleStateMachine.surce/.c> 上使用示例实现。
- 直播错误回调-当 SDK 在调用 KVS API 服务时遇到网络连接超时或其他错误时，将启动此回调。
- 丢帧回调-当由于网络速度慢或直播错误导致存储空间已满时，会启动此回调。如果网络速度导致丢帧，则可以增加存储大小、减小视频帧大小或帧速率以匹配网络速度。

Kinesis 视频直播解析器库

Kinesis 视频流解析器库是一组工具，您可以在 Java 应用程序中使用这些工具在 Kinesis 视频流中使用 MKV 数据。

该库包含以下工具：

- [StreamingMkvReader](#)：此类从视频流中读取指定的 MKV 元素。
- [FragmentMetadataVisitor](#)：此类在元数据中检索片段 (媒体元素) 和音轨 (包含音频或字幕等媒体信息的单个数据流)。
- [OutputSegmentMerger](#)：此类可合并视频流中的连续片段或数据块。
- [KinesisVideoExample](#)：这是一个示例应用程序，显示了如何使用 Kinesis 视频流解析器库。

该库还包括介绍如何使用这些工具的测试。

步骤：使用 Kinesis 视频流解析器库

该过程包括以下步骤：

- [the section called “步骤 1：下载和配置代码”](#)。
- [the section called “步骤 2：编写和检查代码”](#)。
- [the section called “步骤 3：运行并验证代码”](#)。

先决条件

必须具备以下条件才能检查和使用 Kinesis 视频流解析器库：

- Amazon Web Services (AWS) 账户。如果您还没有 AWS 账户，请参阅[the section called “注册获取 AWS 账户”](#)。
- Java 集成开发环境 (IDE)，例如 [Eclipse Java Neon](#) 或 [JetBrains IntelliJ IDEA](#)。

步骤 1：下载和配置代码

在此部分中，您下载 Java 库和测试代码，并将项目导入 Java IDE 中。

有关此过程的先决条件及其他详细信息，请参阅 [视频流解析器库](#)。

1. 创建一个目录并从GitHub存储库中克隆库源代码 (<https://github.com/aws/amazon-kinesis-video-streams-parser-library>) 。

```
$ git clone https://github.com/aws/amazon-kinesis-video-streams-parser-library
```

2. 打开你正在使用的 Java IDE (例如 [Eclipse](#) 或 [Intelli J IDE](#)) ，然后导入你下载的 Apache Maven 项目：

- 在 Eclipse 中：依次选择 File、Import、Maven、Existing Maven Projects ，并导航到 kinesis-video-streams-parser-lib 文件夹。
- 在 IntelliJ Idea 中：选择 Import。导航到下载的程序包的根目录中的 pom.xml 文件。

有关更多信息，请参阅相关的 IDE 文档。

下一步

[the section called “步骤 2：编写和检查代码”](#)。

步骤 2：编写和检查代码

在此部分中，您将检查 Java 库和测试代码，并了解如何在您自己的代码中使用该库中的工具。

Kinesis 视频流解析器库包含以下工具：

- [StreamingMkvReader](#)
- [FragmentMetadataVisitor](#)
- [OutputSegmentMerger](#)
- [KinesisVideoExample](#)

StreamingMkvReader

此类以非阻止方式从流中读取指定的 MKV 元素。

以下代码示例 (来自 `FragmentMetadataVisitorTest`) 说明如何创建 `Streaming MkvReader` 并使用它从名为 `inputStream` 的输入流中检索 `MkvElement` 对象：

```

StreamingMkvReader mkvStreamReader =
    StreamingMkvReader.createDefault(new
InputStreamParserByteSource(inputStream));
    while (mkvStreamReader.mightHaveNext()) {
        Optional<MkvElement> mkvElement = mkvStreamReader.nextIfAvailable();
        if (mkvElement.isPresent()) {
            mkvElement.get().accept(fragmentVisitor);
            ...
        }
    }
}

```

FragmentMetadataVisitor

该类检索片段（媒体元素）的元数据，并跟踪包含媒体信息的单个数据流，例如编解码器私有数据、像素宽度或像素高度。

以下代码示例（来自 `FragmentMetadataVisitorTest` 文件）说明如何使用 `FragmentMetadataVisitor` 检索 `MkvElement` 对象中的数据：

```

FragmentMetadataVisitor fragmentVisitor = FragmentMetadataVisitor.create();
StreamingMkvReader mkvStreamReader =
    StreamingMkvReader.createDefault(new InputStreamParserByteSource(in));
int segmentCount = 0;
while(mkvStreamReader.mightHaveNext()) {
    Optional<MkvElement> mkvElement = mkvStreamReader.nextIfAvailable();
    if (mkvElement.isPresent()) {
        mkvElement.get().accept(fragmentVisitor);
        if
(MkvTypeInfos.SIMPLEBLOCK.equals(mkvElement.get().getElementMetaData().getTypeInfo()))
{
            MkvDataElement dataElement = (MkvDataElement) mkvElement.get();
            Frame frame =
((MkvValue<Frame>)dataElement.getValueCopy()).getVal();
            MkvTrackMetadata trackMetadata =
fragmentVisitor.getMkvTrackMetadata(frame.getTrackNumber());
            assertTrackAndFragmentInfo(fragmentVisitor, frame, trackMetadata);
        }
        if
(MkvTypeInfos.SEGMENT.equals(mkvElement.get().getElementMetaData().getTypeInfo())) {
            if (mkvElement.get() instanceof MkvEndMasterElement) {
                if (segmentCount < continuationTokens.size()) {

```


以下代码示例 (来自 `FragmentMetadataVisitorTest` 文件) 说明如何使用 `OutputSegmentMerger` 合并来自名为 `inputBytes` 的字节数组的音轨元数据：

```
FragmentMetadataVisitor fragmentVisitor = FragmentMetadataVisitor.create();

ByteArrayOutputStream outputStream = new ByteArrayOutputStream();

OutputSegmentMerger outputSegmentMerger =
    OutputSegmentMerger.createDefault(outputStream);

CompositeMkvElementVisitor compositeVisitor =
    new TestCompositeVisitor(fragmentVisitor, outputSegmentMerger);

final InputStream in = TestResourceUtil.getTestInputStream("output_get_media.mkv");

StreamingMkvReader mkvStreamReader =
    StreamingMkvReader.createDefault(new InputStreamParserByteSource(in));

while (mkvStreamReader.mightHaveNext()) {
    Optional<MkvElement> mkvElement = mkvStreamReader.nextIfAvailable();
    if (mkvElement.isPresent()) {
        mkvElement.get().accept(compositeVisitor);
        if
(MkvTypeInfos.SIMPLEBLOCK.equals(mkvElement.get().getElementMetaData().getTypeInfo()))
        {
            MkvDataElement dataElement = (MkvDataElement) mkvElement.get();
            Frame frame = ((MkvValue<Frame>) dataElement.getValueCopy()).getVal();
            Assert.assertTrue(frame.getFrameData().limit() > 0);
            MkvTrackMetadata trackMetadata =
fragmentVisitor.getMkvTrackMetadata(frame.getTrackNumber());
            assertTrackAndFragmentInfo(fragmentVisitor, frame, trackMetadata);
        }
    }
}
```

上一个示例显示以下编码模式：

- 创建 [FragmentMetadataVisitor](#) 以从流中检索元数据。
- 创建一个输出流来接收合并的元数据。
- 创建 `OutputSegmentMerger`，传入到 `ByteArrayOutputStream` 中。
- 创建包含两个访问者的 `CompositeMkvElementVisitor`。
- 创建指向指定文件的 `InputStream`。

- 将输入数据中的每个元素合并到输出流中。

KinesisVideoExample

这是一个示例应用程序，显示了如何使用 Kinesis 视频流解析器库。

该类执行以下操作：

- 创建 Kinesis 视频流。如果已存在具有给定名称的流，则删除流并重新创建。
- 调用[PutMedia](#)将视频片段流式传输到 Kinesis 视频流。
- 关于[GetMedia](#)从 Kinesis 视频流中流出视频片段的调用。
- 使用 [StreamingMkvReader](#) 解析流中返回的片段，使用 [FragmentMetadataVisitor](#) 记录片段。

删除流并重新创建

以下代码示例（来自StreamOps.java文件）删除给定的 Kinesis 视频流：

```
//Delete the stream
amazonKinesisVideo.deleteStream(new
    DeleteStreamRequest().withStreamARN(streamInfo.get().getStreamARN()));
```

以下代码示例（来自StreamOps.java文件）创建了具有指定名称的 Kinesis 视频流：

```
amazonKinesisVideo.createStream(new CreateStreamRequest().withStreamName(streamName)
    .withDataRetentionInHours(DATA_RETENTION_IN_HOURS)
    .withMediaType("video/h264"));
```

打电话 PutMedia

以下代码示例（来自PutMediaWorker.java文件）调用[PutMedia](#)流：

```
putMedia.putMedia(new PutMediaRequest().withStreamName(streamName)
    .withFragmentTimecodeType(FragmentTimecodeType.RELATIVE)
    .withProducerStartTimestamp(new Date())
    .withPayload(inputStream), new PutMediaAckResponseHandler() {
    ...
});
```


打电话 GetMedia

以下代码示例 (来自 `GetMediaWorker.java` 文件) 调用 [GetMedia](#) 流 :

```
GetMediaResult result = videoMedia.getMedia(new
    GetMediaRequest().withStreamName(streamName).withStartSelector(startSelector));
```

解析结果 GetMedia

本节介绍如何使用 [StreamingMkvReader](#)、[FragmentMetadataVisitor](#) 和 [CompositeMkvElementVisitor](#) 解析、保存到文件以及记录 [GetMedia](#) 返回的数据。

读取 with [GetMedia](#) 的输出 [StreamingMkvReader](#)

以下代码示例 (来自 `GetMediaWorker.java` 文件) 创建 [StreamingMkvReader](#) 并使用它来解析 [GetMedia](#) 操作的结果 :

```
StreamingMkvReader mkvStreamReader = StreamingMkvReader.createDefault(new
    InputStreamParserByteSource(result.getPayload()));
log.info("StreamingMkvReader created for stream {} ", streamName);
try {
    mkvStreamReader.apply(this.elementVisitor);
} catch (MkvElementVisitException e) {
    log.error("Exception while accepting visitor {}", e);
}
```

在前面的代码示例中, [StreamingMkvReader](#) 从 [GetMedia](#) 结果的负载中检索 `MkvElement` 对象。在下一节中, 将元素传递给 [FragmentMetadataVisitor](#)。

使用以下方法检索片段 [FragmentMetadataVisitor](#)

下面的代码示例 (摘自 `KinesisVideoExample.java` 和 `StreamingMkvReader.java` 文件) 创建 [FragmentMetadataVisitor](#)。然后, 将 [StreamingMkvReader](#) 迭代的 `MkvElement` 对象传递给使用 `accept` 方法的访问者。

摘自 `KinesisVideoExample.java` :

```
FragmentMetadataVisitor fragmentMetadataVisitor = FragmentMetadataVisitor.create();
```

摘自 *StreamingMkvReader.java* :

```
if (mkvElementOptional.isPresent()) {
    //Apply the MkvElement to the visitor
    mkvElementOptional.get().accept(elementVisitor);
}
```

记录元素并将其写入文件

下面的代码示例 (摘自 *KinesisVideoExample.java* 文件) 创建以下对象，并将它们作为 *GetMediaProcessingArguments* 函数返回值的一部分返回：

- 写入系统日志的 *LogVisitor* (*MkvElementVisitor* 的扩展)。
- 将传入数据写入 MKV 文件的 *OutputStream*。
- 缓冲发往 *OutputStream* 的数据的 *BufferedOutputStream*。
- 将 *GetMedia* 结果中的连续元素与相同音轨和 EBML 数据合并的 [the section called “OutputSegmentMerger”](#)。
- A *CompositeMkvElementVisitor* 将 [FragmentMetadataVisitor](#) [the section called “OutputSegmentMerger”](#)、和组成 *LogVisitor* 为单个元素的访客。

```
//A visitor used to log as the GetMedia stream is processed.
LogVisitor logVisitor = new LogVisitor(fragmentMetadataVisitor);

//An OutputSegmentMerger to combine multiple segments that share track and ebml
metadata into one
//mkv segment.
OutputStream fileOutputStream =
Files.newOutputStream(Paths.get("kinesis_video_example_merged_output2.mkv"),
    StandardOpenOption.WRITE, StandardOpenOption.CREATE);
BufferedOutputStream outputStream = new BufferedOutputStream(fileOutputStream);
OutputSegmentMerger outputSegmentMerger =
OutputSegmentMerger.createDefault(outputStream);

//A composite visitor to encapsulate the three visitors.
CompositeMkvElementVisitor mkvElementVisitor =
    new CompositeMkvElementVisitor(fragmentMetadataVisitor,
outputSegmentMerger, logVisitor);
```

```
return new GetMediaProcessingArguments(outputStream, logVisitor,  
mkvElementVisitor);
```

然后将媒体处理参数传递到 `GetMediaWorker`，然后传递给，后者在单独的 `ExecutorService` 线程上执行 worker：

```
GetMediaWorker getMediaWorker = GetMediaWorker.create(getRegion(),  
getCredentialsProvider(),  
getStreamName(),  
new StartSelector().withStartSelectorType(StartSelectorType.EARLIEST),  
amazonKinesisVideo,  
getMediaProcessingArgumentsLocal.getMkvElementVisitor());  
executorService.submit(getMediaWorker);
```

下一步

[the section called “步骤 3：运行并验证代码”](#)

步骤 3：运行并验证代码

Kinesis 视频流解析器库包含供您在自己的项目中使用的工具。项目包含针对工具的单元测试，您可以运行此工具验证您的安装。

库中包含以下单元测试：

- mkv
 - ElementSizeAndOffsetVisitorTest
 - MkvValueTest
 - StreamingMkvReaderTest
- utilities
 - FragmentMetadataVisitorTest
 - OutputSegmentMergerTest

亚马逊 Kinesis Video Streams 示例

以下代码示例演示了如何使用 Kinesis Video Streams API :

示例：向 Kinesis Video Streams 发送数据

- [示例：Kinesis Video Streams Producer SDK gStreamer 插件](#)：演示如何构建用作 GStreamer 目标的 Kinesis Video Streams Producer SDK。
- [在 Docker 容器中运行 gStreamer 元素](#)：演示如何使用预先构建的 Docker 镜像将 RTSP 视频从 IP 摄像机发送到 Kinesis Video Streams。
- [示例：从 RTSP 来源进行流式传输](#)：演示如何构建自己的 Docker 镜像以及如何将 RTSP 视频从 IP 摄像机发送到 Kinesis Video Streams。
- [示例：使用 PutMedia API 向 Kinesis Video Streams 发送数据](#)：演示如何使用 API 将已经采用容器格式 (MKV) 的数据发送到 Kinesis Video Streams [PutMedia](#)。[使用 Java 创建者库](#)

示例：从 Kinesis Video Streams 检索数据

- [KinesisVideoExample](#)：演示如何使用 Kinesis Video Streams 解析器库解析和记录视频片段。
- [示例：解析和渲染 Kinesis Video Streams 片段](#)：演示如何使用 jCodec 和 jF rame 解析和渲染 Kinesis 视频流片段。
- [the section called “SageMaker”](#)：显示一种解决方案，该解决方案使用 Amazon SageMaker 来确定某些对象何时出现在视频流中。

示例：播放视频数据

- [示例：在 HTML 中使用 HLS 和 JavaScript](#)：演示如何检索 Kinesis 视频流的 HLS 直播会话并在网页中播放。

先决条件

- 在示例代码中，您可以通过指定在凭据配置文件中设置的配置文件或在集成开发环境 (IDE) 的 Java 系统属性中提供凭据来提供凭据。AWS 如果您还没有这样做，请先设置您的凭据。有关更多信息，请参阅[设置用于开发的 AWS 凭据和区域](#)。

- 建议您使用 Java IDE 来查看和运行代码，例如下列项之一：
 - [Eclipse Java Neon](#)
 - [JetBrains IntelliJ IDEA](#)

示例：Kinesis Video Streams Producer SDK gStreamer 插件

本主题介绍如何构建用作 GStreamer 插件的 Amazon Kinesis Video Streams Producer SDK。

主题

- [下载、构建和配置 gStreamer 元素](#)
- [运行 gStreamer 元素](#)
- [GStreamer 启动命令示例](#)
- [在 Docker 容器中运行 gStreamer 元素](#)
- [gStreamer 元素参数参考](#)

[GStreamer](#) 是一种流行的媒体框架，可供多个摄像机和视频源使用，通过组合模块化插件来创建自定义媒体管道。Kinesis Video Streams gStreamer 插件简化了你现有 gStreamer 媒体管道与 Kinesis Video Streams 的集成。集成 gStreamer 后，您可以将来自网络摄像头或实时流媒体协议 (RTSP) 摄像机的视频流式传输到 Kinesis Video Streams，以便进行实时或以后的播放、存储和进一步分析。

gStreamer 插件将 Kinesis Video Streams Producer SDK 提供的功能封装在 gStreamer sink 元素中，从而自动管理将你的视频流传输到 Kinesis Video Streams Video Streams 的过程。kvssinkGStreamer 框架提供标准的托管环境，用于构建源自摄像机或其他视频源的媒体流，以供进一步处理、渲染或存储。

GStreamer 管道通常由源（视频摄像头）与接收器元素（用于呈现视频的播放器或用于脱机检索的存储）之间的链接组成。在本示例中，您使用创建者开发工具包元素作为视频源（网络摄像头或 IP 摄像机）的接收器或媒体目标。然后，封装 SDK 的插件元素将视频流发送到 Kinesis Video Streams。

本主题介绍如何构建 GStreamer 媒体管道，该管道能够将来自视频源（例如网络摄像头或 RTSP 流）的视频流式传输到 Kinesis Video Streams，通常通过中间编码阶段（使用 H.264 编码）连接到 Kinesis Video Streams。当您的视频流作为 Kinesis 视频流可用时，您可以使用 Kinesis 视频流解析器库对视频流进行进一步处理、播放、存储或分析。

下载、构建和配置 gStreamer 元素

GStreamer 插件示例包含在 Kinesis Video Streams C++ Producer SDK 中。有关该开发工具包的先决条件和下载信息，请参阅[步骤 1：下载并配置 C++ 生产人员库代码](#)。

您可以在 macOS、Ubuntu、Raspberry Pi 或 Windows 中将创建者开发工具包 GStreamer 接收器构建为动态库。GStreamer 插件位于您的 build 目录中。要加载此插件，它必须位于您的插件中 GST_PLUGIN_PATH。运行以下命令：

```
export GST_PLUGIN_PATH=`pwd`/build
```

Note

在 macOS 上，在 Docker 容器中运行 GStreamer 时，只能对来自网络摄像机的视频进行流式处理。在 macOS 上的 Docker 容器中，不支持对来自 USB 摄像机的视频进行流式处理。

运行 gStreamer 元素

要使用 Kinesis Video Streams Producer SDK 元素作为接收器运行 gStreamer，请使用命令 `gst-launch-1.0` 使用适合 GStreamer 插件使用的设置。例如，对于 Linux 系统上的 v4l2 设备使用 [v4l2src](#)，或对于 RTSP 设备使用 [rtspsrc](#)。指定 `kvssink` 作为向创建者开发工具包发送视频的接收器（管道的最终目标）。

`kvssink` 元素具有以下必需参数：

- **stream-name**— 目标 Kinesis 视频流的名称。
- **storage-size**— 设备的存储大小（以千字节为单位）。有关配置设备存储的信息，请参阅[StorageInfo](#)。
- **access-key**— 用于 AWS 访问 Kinesis Video Streams 的访问密钥。您必须提供此参数或 `credential-path`。
- **secret-key**— 用于访问 Kinesis Video Streams 的 AWS 密钥。您必须提供此参数或 `credential-path`。
- **credential-path**— 包含您访问 Kinesis Video Streams 的凭据的文件路径。有关轮换证书的更多信息，请参阅[管理 IAM 用户的访问密钥](#)。您必须提供此参数或 `access-key` 和 `secret-key`。

有关 `kvssink` 可选参数的信息，请参阅[gStreamer 元素参数参考](#)。

有关 gStreamer 插件和参数的最新信息，请参阅 [gStreamer 插件](#)，或使用以下命令列出选项：

```
gst-inspect-1.0 kvssink
```

如果构建失败或 GST_PLUGIN_PATH 设置不正确，则输出将如下所示：

```
No such element or plugin 'kvssink'
```

GStreamer 启动命令示例

以下示例演示如何使用 GStreamer 插件从不同类型的设备流式传输视频。

示例 1：在 Ubuntu 上直播来自 RTSP 摄像头的视频

以下命令将使用 [rtspsrc](#) GStreamer 插件在 Ubuntu 上创建一个流自网络 RTSP 摄像机的 GStreamer 管道：

```
gst-launch-1.0 -v rtspsrc location="rtsp://YourCameraRtspUrl" short-header=TRUE !  
rtph264depay ! h264parse ! kvssink stream-name="YourStreamName" storage-size=128
```

示例 2：在 Ubuntu 上对来自 USB 摄像头的视频进行编码和流式传输

以下命令在 Ubuntu 上创建一个 GStreamer 管道，该管道以 H.264 格式对来自 USB 摄像头的直播进行编码，然后将其流式传输到 Kinesis Video Streams。此示例使用 [v4l2](#) src gStreamer 插件。

```
gst-launch-1.0 v4l2src do-timestamp=TRUE device=/dev/video0 ! videoconvert ! video/x-  
raw,format=I420,width=640,height=480,framerate=30/1 ! x264enc bframes=0 key-int-max=45  
bitrate=500 ! video/x-h264,stream-format=avc,alignment=au,profile=baseline ! kvssink  
stream-name="YourStreamName" storage-size=512 access-key="YourAccessKey" secret-  
key="YourSecretKey" aws-region="YourAWSRegion"
```

示例 3：在 Ubuntu 上流式传输来自 USB 摄像头的预编码视频

以下命令在 Ubuntu 上创建 GStreamer 管道，该管道将摄像机已经以 H.264 格式编码的视频流式传输到 Kinesis Video Streams。此示例使用 [v4l2](#) src gStreamer 插件。

```
gst-launch-1.0 v4l2src do-timestamp=TRUE device=/dev/video0 ! h264parse ! video/x-  
h264,stream-format=avc,alignment=au ! kvssink stream-name="plugin" storage-size=512  
access-key="YourAccessKey" secret-key="YourSecretKey" aws-region="YourAWSRegion"
```

示例 4：在 macOS 上流式传输来自网络摄像机的视频

以下命令在 macOS 上创建一个 gStreamer 管道，该管道通过网络摄像机将视频流式传输到 Kinesis Video Streams。此示例使用 [rtspsrc](#) GStreamer 插件。

```
gst-launch-1.0 rtspsrc location="rtsp://YourCameraRtspUrl" short-header=TRUE !
  rtph264depay ! video/x-h264, format=avc,alignment=au ! kvssink stream-
  name="YourStreamName" storage-size=512 access-key="YourAccessKey" secret-
  key="YourSecretKey" aws-region="YourAWSRegion"
```

示例 5：在 Windows 上流式传输来自网络摄像机的视频

以下命令在 Windows 上创建一个 gStreamer 管道，用于将视频从网络摄像机流式传输到 Kinesis Video Streams。此示例使用 [rtspsrc](#) GStreamer 插件。

```
gst-launch-1.0 rtspsrc location="rtsp://YourCameraRtspUrl" short-header=TRUE !
  rtph264depay ! video/x-h264, format=avc,alignment=au ! kvssink stream-
  name="YourStreamName" storage-size=512 access-key="YourAccessKey" secret-
  key="YourSecretKey" aws-region="YourAWSRegion"
```

示例 6：在 Raspberry Pi 上直播来自摄像头的视频

以下命令在 Raspberry Pi 上创建一个 gStreamer 管道，用于将视频流式传输到 Kinesis Video Streams。此示例使用 [v4l2src](#) gStreamer 插件。

```
gst-launch-1.0 v4l2src do-timestamp=TRUE device=/dev/video0 ! videoconvert !
  video/x-raw,format=I420,width=640,height=480,framerate=30/1 !
  omxh264enc control-rate=1 target-bitrate=5120000 periodicity-
  idr=45 inline-header=FALSE ! h264parse ! video/x-h264,stream-
  format=avc,alignment=au,width=640,height=480,framerate=30/1,profile=baseline ! kvssink
  stream-name="YourStreamName" access-key="YourAccessKey" secret-key="YourSecretKey"
  aws-region="YourAWSRegion"
```

示例 7：在 Raspberry Pi 上直播来自摄像机的视频并指定区域

以下命令在 Raspberry Pi 上创建 GStreamer 管道，该管道将视频流式传输到美国东部（弗吉尼亚北部）地区的 Kinesis Video Streams。此示例使用 [v4l2src](#) gStreamer 插件。

```
gst-launch-1.0 v4l2src do-timestamp=TRUE device=/dev/video0 ! videoconvert !
  video/x-raw,format=I420,width=640,height=480,framerate=30/1 !
```



```
omxh264enc control-rate=1 target-bitrate=5120000 periodicity-  
idr=45 inline-header=FALSE ! h264parse ! video/x-h264,stream-  
format=avc,alignment=au,width=640,height=480,framerate=30/1,profile=baseline ! kvssink  
stream-name="YourStreamName" access-key="YourAccessKey" secret-key="YourSecretKey"  
aws-region="YourAWSRegion"
```

示例 8：在 Raspberry Pi 和 Ubuntu 中同时直播音频和视频

了解在 Raspberry-Pi 和 Ubuntu 中如何[运行 `gst-launch-1.0` 命令开始对音频和视频进行流式处理](#)。

示例 9：在 macOS 中同时传输音频和视频

了解在 MacOS 中如何[运行 `gst-launch-1.0` 命令开始对音频和视频进行流式处理](#)。

示例 10：上传包含音频和视频的 MKV 文件

了解如何[运行 `gst-launch-1.0` 命令来上传包含音频和视频的 MKV 文件](#)。

在 Docker 容器中运行 gStreamer 元素

Docker 是一个使用容器来开发、部署和运行应用程序的平台。使用 Docker 创建 GStreamer 管道可以标准化 Kinesis Video Streams 的操作环境，从而简化了应用程序的构建和使用。

要安装和配置 Docker，请参阅以下内容：

- [Docker 下载说明](#)
- [开始使用 Docker](#)

安装 Docker 后，你可以使用命令从亚马逊弹性容器注册表下载 Kinesis Video Streams C++ Producer SDK（以及 gStreamer 插件）。`docker pull`

要将 Kinesis Video Streams Producer SDK 元素作为 Docker 容器中的接收器运行 gStreamer，请执行以下操作：

主题

- [对你的 Docker 客户端进行身份验证](#)
- [下载用于 Ubuntu、macOS、Windows 或 Raspberry Pi 的 Docker 映像](#)
- [运行 Docker 镜像](#)

对你的 Docker 客户端进行身份验证

将您的 Docker 客户端验证到要从中提取镜像的 Amazon ECR 注册表。您必须为使用的每个注册表获取身份验证令牌。代币的有效期为 12 小时。有关更多信息，请参阅 Amazon Elastic Container Registry 用户指南中的[注册表身份验证](#)。

Example 向 Amazon ECR 进行身份验证

```
aws ecr get-login-password --region us-west-2 | docker login -u AWS --password-stdin
https://546150905175.dkr.ecr.us-west-2.amazonaws.com
```

如果成功，输出将打印 Login Succeeded。

下载用于 Ubuntu、macOS、Windows 或 Raspberry Pi 的 Docker 映像

根据您的操作系统，使用以下命令之一将 Docker 映像下载到您的 Docker 环境：

下载用于 Ubuntu 的 Docker 映像

```
sudo docker pull 546150905175.dkr.ecr.us-west-2.amazonaws.com/kinesis-video-producer-
sdk-cpp-amazon-linux:latest
```

下载用于 macOS 的 Docker 映像

```
sudo docker pull 546150905175.dkr.ecr.us-west-2.amazonaws.com/kinesis-video-producer-
sdk-cpp-amazon-linux:latest
```

下载用于 Windows 的 Docker 映像

```
docker pull 546150905175.dkr.ecr.us-west-2.amazonaws.com/kinesis-video-producer-sdk-
cpp-amazon-windows:latest
```

下载用于 Raspberry Pi 的 Docker 映像

```
sudo docker pull 546150905175.dkr.ecr.us-west-2.amazonaws.com/kinesis-video-producer-
sdk-cpp-raspberry-pi:latest
```

要验证是否已成功添加镜像，请使用以下命令：

```
docker images
```

运行 Docker 镜像

根据您的操作系统，使用以下命令之一运行 Docker 映像：

在 Ubuntu 上运行 Docker 镜像

```
sudo docker run -it --network="host" --device=/dev/video0 546150905175.dkr.ecr.us-west-2.amazonaws.com/kinesis-video-producer-sdk-cpp-amazon-linux /bin/bash
```

在 macOS 上运行 Docker 镜像

```
sudo docker run -it --network="host" 546150905175.dkr.ecr.us-west-2.amazonaws.com/kinesis-video-producer-sdk-cpp-amazon-linux /bin/bash
```

在 Windows 上运行 Docker 镜像

```
docker run -it 546150905175.dkr.ecr.us-west-2.amazonaws.com/kinesis-video-producer-sdk-cpp-windows <AWS_ACCESS_KEY_ID> <AWS_SECRET_ACCESS_KEY> <RTSP_URL> <STREAM_NAME>
```

在树莓派上运行 Docker 镜像

```
sudo docker run -it --device=/dev/video0 --device=/dev/vchiq -v /opt/vc:/opt/vc 546150905175.dkr.ecr.us-west-2.amazonaws.com/kinesis-video-producer-sdk-cpp-raspberry-pi /bin/bash
```

Docker 启动容器并显示命令提示符，供您在容器中使用命令。

在该容器中，使用以下命令设置环境变量：

```
export LD_LIBRARY_PATH=/opt/awssdk/amazon-kinesis-video-streams-producer-sdk-cpp/kinesis-video-native-build/downloads/local/lib:$LD_LIBRARY_PATH
export PATH=/opt/awssdk/amazon-kinesis-video-streams-producer-sdk-cpp/kinesis-video-native-build/downloads/local/bin:$PATH
export GST_PLUGIN_PATH=/opt/awssdk/amazon-kinesis-video-streams-producer-sdk-cpp/kinesis-video-native-build/downloads/local/lib:$GST_PLUGIN_PATH
```

使用适合您设备的 `gst-launch-1.0` 命令从摄像机开始直播。

有关使用 `gst-launch-1.0` 命令连接到本地网络摄像机或网络 RTSP 摄像机的示例，请参阅[启动命令](#)。

gStreamer 元素参数参考

要向 Amazon Kinesis Video Streams Video Streams Producer SDK 发送视频，请将视频kvssink指定为管道的接收方或最终目的地。此参考提供了有关 kvssink 必需参数和可选参数的信息。有关更多信息，请参阅[the section called “GStreamer”](#)。

主题

- [the section called “向提供凭证 kvssink”](#)
- [the section called “提供一个区域给 kvssink”](#)
- [the section called “kvssink必需的参数”](#)
- [the section called “kvssink可选参数”](#)

向提供凭证 kvssink

要允许 kvssink gStreamer 元素向其发出请求 AWS，请提供 AWS 凭证供其在调用 Amazon Kinesis Video Streams 服务时使用。凭证提供商链按以下顺序查找证书：

1. AWS IoT 证书

要设置 AWS IoT 凭证，请参阅[the section called “使用控制对 Kinesis Video Streams 资源的访问权限 AWS IoT”](#)。

iot-credentials参数值必须以以下# = ###iot-certificate,##### key = value 对的列表。

键	必需	描述
ca-path	是	用于通过 TLS 与后端服务建立信任的 CA 证书的文件路径。 Example 示例： <code>/file/path/to/certificate.pem</code>
cert-path	是	X.509 证书的文件路径。

键	必需	描述
		<p>Example</p> <p>示 例：<code>/file/path/to/certificateID -certificate.pem.crt</code></p>
endpoint	是	<p>您 AWS 账户的 AWS IoT Core 凭证终端节点提供商终端节点。请参阅 《AWS IoT 开发人员指南》。</p> <p>Example</p> <p>示例：<code>credential-account-specific-prefix .credentials.iot. aws-region .amazonaws.com</code></p>
key-path	是	<p>公钥/私钥对中使用的私钥的文件路径。</p> <p>Example</p> <p>示 例：<code>/file/path/to/certificateID -private.pem.key</code></p>
role-aliases	是	<p>指向连接时要使用 AWS 的 IAM 角色的角色别名的名称 AWS IoT Core。</p> <p>Example</p> <p>示例：<code>KvsCamera IoTRoleAlias</code></p>

键	必需	描述
iot-thing-name	否	<p>iot-thing-name 是可选项。如果iot-thing-name未提供，则使用stream-name 参数值。</p> <p>Example</p> <p>示例：<i>kvs_examp le_camera</i></p>

Example

示例：

```
gst-launch-1.0 -v ... ! kvssink stream-name="YourStream" aws-region="YourRegion"
  iot-certificate="iot-certificate,endpoint=credential-account-specific-  
prefix.credentials.iot.aws-region.amazonaws.com,cert-path=certificateID-  
certificate.pem.crt,key-path=certificateID-private.pem.key,ca-  
path=certificate.pem,role-aliases=YourRoleAlias,iot-thing-name=YourThingName"
```

2. 环境变量

要kvssink使用来自环境的凭证，请设置以下环境变量：

环境变量名	必需	描述
AWS_ACCESS_KEY_ID	是	用于 AWS 访问亚马逊 Kinesis Video Streams 的访问密钥。
AWS_SECRET_ACCESS_KEY	是	与访问 AWS 密钥关联的密钥。
AWS_SESSION_TOKEN	否	如果您直接使用 AWS STS 操作中的临时安全证书，则指定所需的会话令牌值。

设置环境变量会更改使用的值，直到 Shell 会话结束或直到您将该变量设置为其他值。要使变量在 future 会话中保持不变，请在 shell 的启动脚本中对其进行设置。

3. `access-key` , `secret-key`参数

要直接将凭据指定为kvssink参数，请设置以下参数：

kvssink参数名	必需	描述
<code>access-key</code>	是	用于 AWS 访问亚马逊 Kinesis Video Streams 的访问密钥。
<code>secret-key</code>	是	与访问 AWS 密钥关联的密钥。
<code>session-token</code>	否	如果您直接使用 AWS STS 操作中的临时安全证书，则指定所需的会话令牌值。

Example

使用静态凭证：

```
gst-launch-1.0 -v ... ! kvssink stream-name="YourStream" aws-region="YourRegion"
access-key="AKIDEXAMPLE" secret-key="SKEEXAMPLE"
```

Example

使用临时证书：

```
gst-launch-1.0 -v ... ! kvssink stream-name="YourStream" aws-region="YourRegion"
access-key="AKIDEXAMPLE" secret-key="SKEEXAMPLE" session-token="STEXAMPLE"
```

4. 凭证文件

Important

如果您选择了前面的方法之一，则无法使用该`credential-filekvssink`参数。

kvssink参数名	必需	描述
credential-file	是	包含特定格式凭据的文本文件的路径。

文本文件必须包含以下格式之一的凭据：

- 全权证书 *YourAccessKeyYourSecretKey*
- 凭证 *YourAccessKey## YourSecretKeySessionToken*

Example

示例：您的 *credentials.txt* 文件位于 `/home/ubuntu` 并包含以下内容：

```
CREDENTIALS AKIDEXAMPLE 2023-08-10T22:43:00Z SKEXAMPLE STEXAMPLE
```

要在中使用它kvssink，请键入：

```
gst-launch-1.0 -v ... ! kvssink stream-name="YourStream" aws-region="YourRegion"
credential-file="/home/ubuntu/credentials.txt"
```

Note

将来的到期时间应至少为 $5 + 30 + 3 = 38$ 秒。宽限期定义为中的 `IOT_CREDENTIAL_FETCH_GRACE_PERIOD` 变量 [IotCredentialProvider.h](#)。如果您启动时凭证过于接近到期时间kvssink，则会收到错误代码 `0x52000049 - STATUS_INVALID_TOKEN_EXPIRATION`。

Important

kvssink不会修改凭据文件。如果您使用的是临时证书，则证书文件必须在到期时间减去宽限期之前由外部来源更新。

提供一个区域给 kvssink

以下是区域查询顺序：

1. `AWS_DEFAULT_REGION`首先审查环境变量。如果已设置，则使用该区域来配置客户端。
2. `aws-region`接下来将查看参数。如果已设置，则使用该区域来配置客户端。
3. 如果前面的方法均未使用，`kvssink`则默认为`us-west-2`。

kvssink必需的参数

除了提供凭证和区域外，该`kvssink`元素还具有以下必需参数：

`stream-name`-目标亚马逊 Kinesis 视频流的名称。

kvssink可选参数

`kvssink` 元素具有以下可选参数。有关这些参数的更多信息，请参阅 [Kinesis 视频流结构](#)。

参数	描述	单位/类型	默认
<code>absolute-fragment-times</code>	是否使用绝对片段时间。	布尔值	<code>true</code>
<code>access-key</code>	<p>用于 AWS 访问 Kinesis Video Streams 的访问密钥。</p> <p>您必须设置 AWS 凭据或提供此参数。要提供此信息，请键入以下内容：</p> <pre>\$ export AWS_ACCES S_KEY_ID=</pre>		
<code>avg-bandwidth-bps</code>	流的预期平均带宽。	每秒字节数	4194304
<code>aws-region</code>	AWS 区域 要使用的。	字符串	<code>us-west-2</code>

参数	描述	单位/类型	默认
	<p>Note</p> <p>您也可以为该区域提供AWS_DEFAULT_REGION环境变量。如果同时设置了环境变量和 kvssink 参数，则环境变量优先。</p> <p>Important</p> <p>us-west-2 如果未另行指定，则该区域将默认为。</p>		
buffer-duration	流缓冲持续时间。	秒	180
codec-id	流的编解码器 ID。	字符串	"V_MPEG4/ISO/AVC"
connection-staleness	之后调用直播陈旧回调的时间。	秒	60
content-type	流的内容类型。	字符串	"video/h264"
fragment-acks	是否使用片段 ACK。	布尔值	true
fragment-duration	所需的片段持续时间。	毫秒	2000

参数	描述	单位/类型	默认
framerate	预期的帧率。	每秒帧数	25
frame-timecodes	是否使用帧时间码或者使用当前时间回调生成时间戳。	布尔值	true
key-frame-fragmentation	是否在关键帧上生成片段。	布尔值	true
log-config	日志配置路径。	字符串	"/kvs_log_configuration"
max-latency	流的最大延迟。	秒	60
recalculate-metrics	是否重新计算指标。	布尔值	true
replay-duration	启用重新启动时，在出错时回滚当前阅读器以重放的持续时间。	秒	40
restart-on-error	发生错误时是否重新启动。	布尔值	true
retention-period	保留流的时间长度。	小时	2
rotation-period	密钥轮换周期。有关更多信息，请参阅 轮换客户主密钥 。	秒	2400

参数	描述	单位/类型	默认
secret-key	<p>用于访问 Kinesis Video Streams 的 AWS 密钥。</p> <p>您必须设置 AWS 凭据或提供此参数。</p> <pre>\$ export AWS_SECRE T_ACCESS_ KEY_ID=</pre>		
session-token	如果您直接使用 AWS STS 操作中的临时安全证书，则指定所需的会话令牌值。		
storage-size	设备存储大小，以 MegaBytes (MB) 为单位。有关配置设备存储的信息，请参阅 StorageInfo 。	MegaBytes	128
streaming-type	<p>流式处理类型。有效值包括：</p> <ul style="list-style-type: none"> • 0：实时 • 1：几乎实时（当前不支持） • 2：离线 	枚举 GstKvsSinkStreamingType	0：实时
timecode-scale	MKV 时间码标度。	毫秒	1
track-name	MKV 音轨名称。	字符串	"kinesis_video"

参数	描述	单位/类型	默认
iot-certificate	<p>AWS IoT 要在kvssink元素中使用的证书。</p> <p>iot-certificate 接受以下键和值：</p> <div data-bbox="472 575 792 1083" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin: 10px 0;"> <p> Note</p> <p>iot-thing-name 是可选的。如果iot-thing-name 未提供，则使用stream-name 参数值。</p> </div> <ul style="list-style-type: none"> • endpoint=iotcredentialsproviderendpoint • cert-path=/localdirectorypath/to/certificate • key-path=/localdirectorypath/to/private/key 	字符串	无

参数	描述	单位/类型	默认
	<ul style="list-style-type: none"> • <code>ca-path=/localdir/ectorypath/to/ca-cert</code> • <code>role-aliases =role-aliases</code> • <code>iot-thing-name=YourIotThingName</code> 		

示例：使用 PutMedia API 向 Kinesis Video Streams 发送数据

此示例演示如何使用 [PutMedia API](#)。它显示了如何发送已经是容器格式 (MKV) 的数据。如果您的数据在发送之前必须汇编成容器格式（例如，如果您要将摄像机视频数据组合成帧），请参阅[Kinesis 视频直播制作人库](#)。

Note

该PutMedia操作仅在 C++ 和 Java 软件开发工具包中可用。这是因为连接、数据流和确认采用全双工管理。其他语言不支持它。

此示例包括以下步骤：

- [步骤 1：下载并配置代码](#)
- [第 2 步：编写并检查代码](#)
- [步骤 3：运行并验证代码](#)

步骤 1：下载并配置代码

按照以下步骤下载 Java 示例代码，将项目导入 Java IDE，配置库位置，然后配置代码以使用您的 AWS 凭据。

1. 创建目录并从 GitHub 存储库中克隆示例源代码。PutMedia 示例是 [Java 创建者库](#) 的一部分。

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

2. 打开你正在使用的 Java IDE (例如 Eclipse [se](#) 或 [Intelli J ID EA](#))，然后导入你下载的 Apache Maven 项目：

- 在 Eclipse 中：依次选择 File (文件)、Import (导入)、Maven 和 Existing Maven Projects (现有的 Maven 项目)，然后导航到下载的程序包的根目录。选择 pom.xml 文件。
- 在 IntelliJ Idea 中：选择 Import。导航到下载的程序包的根目录中的 pom.xml 文件。

有关更多信息，请参阅相关的 IDE 文档。

3. 更新项目，以便 IDE 可以发现您导入的库。

- 对于 IntelliJ IDEA，请执行下列操作之一：
 - a. 打开项目的 lib 目录的上下文菜单 (右键单击)，选择 Add as library。
 - b. 选择“文件”，然后选择“项目结构”。
 - c. 在 Project Settings 下，选择 Modules。
 - d. 在 Sources (源) 选项卡中，将 Language Level (语言级别) 设置为 7 或更大值。
- 对于 Eclipse，执行以下操作：
 - a. 打开项目的上下文菜单 (右键单击)，并依次选择 Properties、Java Build Path 和 Source。然后执行以下操作：
 1. 在 Source 选项卡中，双击 Native library location。
 2. 在 Native Library Folder Configuration 向导中，选择 Workspace。
 3. 在 Native Library Folder 选择项中，选择项目中的 lib 目录。
 - b. 打开项目的上下文菜单 (右键单击)，选择 Properties。然后执行以下操作：
 1. 在 Libraries 选项卡上，选择 Add Jars。
 2. 在 JAR 选择 向导中，选择项目的 lib 目录中的所有 .jar。

第 2 步：编写并检查代码

PutMedia API 示例 (PutMediaDemo) 显示以下编码模式：

主题

- [创建 PutMediaClient](#)
- [对媒体进行流式处理并暂停线程](#)

此部分中的代码示例来自 PutMediaDemo 类。

创建 PutMediaClient

创建PutMediaClient对象需要以下参数：

- PutMedia 终端节点的 URI。
- 一个 InputStream，指向要流式处理的 MKV 文件。
- 流名称。此示例使用已在 [使用 Java 创建者库](#) (my-stream) 中创建的流。要使用其他流，请更改以下参数：

```
private static final String STREAM_NAME="my-stream";
```

Note

PutMediaAPI 示例未创建直播。你必须使用测试应用程序[使用 Java 创建者库](#)、Kinesis Video Streams 控制台或 AWS CLI。

- 当前时间戳。
- 时间码类型。此示例使用 RELATIVE，指示时间戳相对于容器的开始时间。
- 一个 AWSKinesisVideoV4Signer 对象，验证收到的数据包是否由授权发件人发送。
- 最大上游带宽 (以 Kbps 为单位)。
- 一个 AckConsumer 对象，用于接收“数据包已接收确认”。

以下代码创建 PutMediaClient 对象：

```
/* actually URI to send PutMedia request */
final URI uri = URI.create(KINESIS_VIDEO_DATA_ENDPOINT + PUT_MEDIA_API);

/* input stream for sample MKV file */
final InputStream inputStream = new FileInputStream(MKV_FILE_PATH);

/* use a latch for main thread to wait for response to complete */
final CountDownLatch latch = new CountDownLatch(1);
```



```
/* a consumer for PutMedia ACK events */
final AckConsumer ackConsumer = new AckConsumer(latch);

/* client configuration used for AWS SigV4 signer */
final ClientConfiguration configuration = getClientConfiguration(uri);

/* PutMedia client */
final PutMediaClient client = PutMediaClient.builder()
    .putMediaDestinationUri(uri)
    .mkvStream(inputStream)
    .streamName(STREAM_NAME)
    .timestamp(System.currentTimeMillis())
    .fragmentTimeCodeType("RELATIVE")
    .signWith(getKinesisVideoSigner(configuration))
    .upstreamKbps(MAX_BANDWIDTH_KBPS)
    .receiveAcks(ackConsumer)
    .build();
```

对媒体进行流式处理并暂停线程

在创建客户端后，此示例使用 `putMediaInBackground` 开始异步流式处理。随后，主线程将暂停并显示 `latch.await`，直到 `AckConsumer` 返回，此时客户端已关闭。

```
/* start streaming video in a background thread */
    client.putMediaInBackground();

    /* wait for request/response to complete */
    latch.await();

    /* close the client */
    client.close();
```

步骤 3：运行并验证代码

若要运行 PutMedia API 示例，请执行以下操作：

1. 在 Kinesis Video Streams 控制台 `my-stream` 中或使用 AWS CLI 创建名为的直播。
2. 将工作目录更改为 Java 创建者开发工具包目录：

```
cd /<YOUR_FOLDER_PATH_WHERE_SDK_IS_DOWNLOADED>/amazon-kinesis-video-streams-  
producer-sdk-java/
```

3. 编译 Java 开发工具包演示应用程序：

```
mvn package
```

4. 在 /tmp 目录中创建临时文件名：

```
jar_files=$(mktemp)
```

5. 创建从本地存储库到文件的依赖项的类路径字符串：

```
mvn -Dmdep.outputFile=$jar_files dependency:build-classpath
```

6. 将 LD_LIBRARY_PATH 环境变量的值设置如下：

```
export LD_LIBRARY_PATH=/<YOUR_FOLDER_PATH_WHERE_SDK_IS_DOWNLOADED>/amazon-kinesis-  
video-streams-producer-sdk-cpp/kinesis-video-native-build/downloads/local/lib:  
$LD_LIBRARY_PATH  
$ classpath_values=$(cat $jar_files)
```

7. 按如下所示从命令行运行演示，并提供您的 AWS 凭据：

```
java -classpath target/kinesisvideo-java-demo-1.0-SNAPSHOT.jar:$classpath_values -  
Daws.accessKeyId=${ACCESS_KEY} -Daws.secretKey=${SECRET_KEY} -Djava.library.path=/  
opt/amazon-kinesis-video-streams-producer-sdk-cpp/kinesis-video-native-build  
com.amazonaws.kinesisvideo.demoapp.DemoAppMain
```

8. 打开 [Kinesis Video Streams](#) 控制台，然后在“管理直播”页面上选择您的直播。该视频将在 Video Preview 窗格中播放。

示例：从 RTSP 来源进行流式传输

[C++ 创建者库](#) 包含连接 RTSP（实时流协议）网络摄像机的 [Docker](#) 容器的定义。使用 Docker 可以标准化 Kinesis Video Streams 的操作环境，从而简化应用程序的构建和使用。

以下过程演示如何设置和使用此 RTSP 演示应用程序。

主题

- [先决条件](#)
- [构建 Docker 镜像](#)
- [运行 RTSP 示例应用程序](#)

先决条件

要运行 Kinesis Video Streams RTSP 示例应用程序，必须具备以下条件：

- Docker：有关安装和使用 Docker 的信息，请参阅以下链接：
 - [Docker 下载说明](#)
 - [开始使用 Docker](#)
- RTSP 网络摄像机源：有关推荐摄像机的信息，请参阅[系统要求](#)。

构建 Docker 镜像

首先，构建演示应用程序将在其中运行的 Docker 镜像。

1. 克隆 Amazon Kinesis Video Streams 演示存储库。

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

2. 切换到包含 Dockerfile 的目录。在本例中，它是 [docker-rtsp](#) 目录。

```
cd amazon-kinesis-video-streams-demos/producer-cpp/docker-rtsp/
```

3. 使用以下命令构建 Docker 镜像。此命令创建镜像并将其标记为 `rtspdockertest`。

```
docker build -t rtspdockertest .
```

4. 运行 `docker images` 并搜索标记为的图像 ID `rtspdockertest`。

例如，在下面的示例输出中，IMAGE ID 是 `54f0d65f69b2`。

REPOSITORY	TAG	IMAGE ID	CREATED	PLATFORM	SIZE
rtspdockertest	latest	54f0d65f69b2	10 minutes ago	linux/arm64	653.1 MiB
					292.4 MiB

稍后您将需要这个。

运行 RTSP 示例应用程序

您可以从 Docker 容器内部或外部运行 RTSP 示例应用程序。请按照以下相应说明进行操作。

主题

- [在 Docker 容器中](#)
- [在 Docker 容器外面](#)

在 Docker 容器中

运行 RTSP 示例应用程序

1. 使用以下命令启动 Amazon Kinesis Video Streams Docker 容器：

```
docker run -it YourImageId /bin/bash
```

2. 要启动示例应用程序，请提供您的 AWS 证书、Amazon Kinesis 视频流的名称以及 RTSP 网络摄像机的网址。

Important

如果您使用的是临时证书，则还需要提供您的AWS_SESSION_TOKEN。参见下面的第二个示例。

```
export AWS_ACCESS_KEY_ID=YourAccessKeyId  
export AWS_SECRET_ACCESS_KEY_ID=YourSecretKeyId  
export AWS_DEFAULT_REGION=YourAWSRegion  
./kvs_gstreamer_sample YourStreamName YourRtspUrl
```

临时证书：

```
export AWS_ACCESS_KEY_ID=YourAccessKeyId  
export AWS_SECRET_ACCESS_KEY_ID=YourSecretKeyId  
export AWS_SESSION_TOKEN=YourSessionToken
```

```
export AWS_DEFAULT_REGION=YourAWSRegion
./kvs_gstreamer_sample YourStreamName YourRtspUrl
```

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

观看直播。

4. 要退出 Docker 容器，请关闭终端窗口或键入 `exit`。

在 Docker 容器外面

在 Docker 容器外部，使用以下命令：

```
docker run -it YourImageId /bin/bash -c "export AWS_ACCESS_KEY_ID=YourAccessKeyId;  
export AWS_SECRET_ACCESS_KEY=YourSecretKeyId; export  
AWS_SESSION_TOKEN=YourSessionToken; export AWS_DEFAULT_REGION=YourAWSRegion; ./  
kvs_gstreamer_sample YourStreamName YourRtspUrl"
```

示例：解析和渲染 Kinesis Video Streams 片段

[视频流解析器库](#)包含一个名为的演示应用程序 `KinesisVideoRendererExample`，用于演示解析和渲染 Amazon Kinesis 视频流片段。该示例使用 [JCodec](#) 解码使用 [示例：Kinesis Video Streams Producer SDK gStreamer 插件](#) 应用程序提取的 H.264 编码的帧。在使用 JCodec 对帧进行解码后，将使用 [JFrame](#) 渲染可见的图像。

该示例说明了如何执行以下操作：

- 使用 `GetMedia` API 从 Kinesis 视频流中检索帧并渲染该视频流以供观看。
- 在自定义应用程序中查看直播的视频内容，而不是使用 Kinesis Video Streams 控制台。

您还可以使用此示例中的类来查看未编码为 H.264 的 Kinesis 视频流内容，例如在显示之前不需要解码的 JPEG 文件流。

以下过程说明了如何设置和使用渲染器演示应用程序。

先决条件

要检查和使用渲染器示例库，您必须满足以下条件：

- 亚马逊 Web Services (AWS) 账户。如果你还没有 AWS 账户，请参阅 [Kinesis Video Streams 入门](#)。
- [Java 集成开发环境 \(IDE\)](#)，例如 [Eclipse Java Neon](#) 或 [IntelliJ | JetBrains IDEA](#)。

运行渲染器示例

1. 创建一个目录，然后从 GitHub 存储库中克隆示例源代码。

```
git clone https://github.com/aws/amazon-kinesis-video-streams-parser-library
```

2. 打开您正使用的 Java IDE (例如，[Eclipse](#) 或 [IntelliJ IDEA](#))，并导入您下载的 Apache Maven 项目：
 - 在 Eclipse 中：选择 File、Import、Maven、Existing Maven Projects。导航到 `kinesis-video-streams-parser-lib` 目录。
 - 在 IntelliJ Idea 中：选择 Import。导航到下载的程序包的根目录中的 `pom.xml` 文件。

Note

如果 IntelliJ 找不到您的依赖项，则可能需要执行以下操作：

- 干净生成：选择 File (文件)、Settings (设置)、Build, Execution, Deployment (生成、执行、部署)、Compiler (编译器)。确认已选中“重建时清除输出目录”，然后选择“构建，生成项目”。
- 重新导入项目：打开该项目的上下文菜单 (右键单击)，选择 Maven、重新导入。

有关更多信息，请参阅相关的 IDE 文档。

3. 从您的 Java IDE 中，打开 `src/test/java/com.amazonaws.kinesisvideo.parser/examples/KinesisVideoRendererExampleTest`。
4. 从该文件中删除 `@Ignore` 指令。
5. 使用您的 Kinesis 视频流的名称更新 `.stream` 参数。
6. 运行 `KinesisVideoRendererExample` 测试。

工作方式

该示例应用程序说明了如何执行以下操作：

- [发送 MKV 数据](#)
- [将 MKV 片段解析成帧](#)
- [解码并显示画面](#)

发送 MKV 数据

该示例从rendering_example_video.mkv文件中发送示例 MKV 数据，PutMedia用于将视频数据发送到名为render-example-stream的流。

该应用程序创建一个 PutMediaWorker：

```
PutMediaWorker putMediaWorker = PutMediaWorker.create(getRegion(),
    getCredentialsProvider(),
    getStreamName(),
    inputStream,
    streamOps.amazonKinesisVideo);
executorService.submit(putMediaWorker);
```

有关 PutMediaWorker 类的信息，请参阅[视频流解析器库](#)文档中的 [打电话 PutMedia](#)。

将 MKV 片段解析成帧

然后，该示例使用 GetMediaWorker 从流中检索和解析 MKV 片段：

```
GetMediaWorker getMediaWorker = GetMediaWorker.create(getRegion(),
    getCredentialsProvider(),
    getStreamName(),
    new StartSelector().withStartSelectorType(StartSelectorType.EARLIEST),
    streamOps.amazonKinesisVideo,
    getMediaProcessingArgumentsLocal.getFrameVisitor());
executorService.submit(getMediaWorker);
```

有关 GetMediaWorker 类的更多信息，请参阅[视频流解析器库](#)文档中的 [打电话 GetMedia](#)。

解码并显示画面

然后，该示例使用 [JFrame](#) 解码并显示帧。

以下代码示例来自于 `KinesisVideoFrameViewer` 类，该类扩展了 `JFrame`：

```
public void setImage(BufferedImage bufferedImage) {  
    image = bufferedImage;  
    repaint();  
}
```

该图像显示为 [java.awt.image 的实例](#)。 [BufferedImage](#)。有关说明如何使用 `BufferedImage` 的示例，请参阅[读取/加载图像](#)。

示例：使用识别视频流中的对象 SageMaker

此示例演示如何创建用于 [SageMaker](#) 识别特定对象何时出现在 Amazon Kinesis 视频流中的解决方案。SageMaker 是供开发人员和数据科学家快速轻松地构建、训练和部署机器学习模型的托管平台。

该示例由一个包含应用程序功能的 [Docker](#) 容器和一个自动部署应用程序资源的 [AWS CloudFormation](#) 模板组成。AWS

该 AWS CloudFormation 模板创建了以下资源：

- 使用运行库软件的 [AWS Fargate](#) 计算引擎的 [亚马逊弹性容器服务 \(Amazon ECS\)](#) Container Service 集群。
- 一个 [Amazon DynamoDB 表](#)，用于维护在 Fargate 任务上运行的工作程序的检查点和相关状态。
- 一种 [Kinesis 数据流](#)，用于捕获从中生成的推理输出。SageMaker
- 一个解析其输出的 [AWS Lambda SageMaker 函数](#)。
- [AWS Identity and Access Management \(IAM\)](#) 资源，用于提供跨服务的访问权限。
- 用于监控应用程序的 [Amazon CloudWatch](#) 资源。

该应用程序与任何处理数据的 SageMaker 端点兼容。此示例包含使用示例对象识别算法模板创建 SageMaker 端点的说明。您可以根据应用程序的使用案例和要求修改或替换算法。

主题

- [先决条件](#)
- [创建应用程序](#)
- [监视应用程序](#)
- [扩展应用程序](#)

- [清理应用程序](#)

先决条件

示例应用程序具有以下先决条件：

- [SageMaker](#)
- [Kinesis 视频流](#)
- [服务相关角色](#)

SageMaker

此示例需要 SageMaker 笔记本电脑。有关创建笔记本的信息，请参阅 Amazon SageMaker 开发者指南中的[创建笔记本实例](#)。在创建笔记本时，请注意以下内容：

- 将 Amazon_JumpStart_Object_Detection.ipynb 示例（从 Jupyter 控制台的“SageMaker 示例”选项卡的“Amazon 算法简介”部分）添加到笔记本中。
- 创建亚马逊简单存储服务 (Amazon S3) Service 存储桶，并在添加示例时在“先决条件”步骤中提供其名称。
- 创建笔记本后，在 SageMaker 控制台上选择 Endpoint 配置，然后记下端点名称。

Kinesis 视频流

此示例需要一个或多个包含实时视频数据的 Kinesis Video Streams。有关创建 Kinesis 视频流并从摄像机向其发送数据的信息，请参阅[GStreamer](#)记下你的 Kinesis 视频直播名称。

服务相关角色

此示例要求您的账户具有执行 Fargate 操作的服务相关角色。新 AWS 账户 增默认启用此角色。如果您在创建应用程序时看到以下错误，则必须启用服务相关角色：

```
Unable to assume the service linked role. Please verify that the ECS service linked role exists
```

要启用服务相关角色，请运行以下命令：

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

创建应用程序

要创建示例应用程序，请使用 AWS CloudFormation 和提供的模板。

AWS CloudFormation 用于创建应用程序

1. 使用以下链接之一登录 AWS Management Console 并打开 AWS CloudFormation 控制台 AWS 区域。该链接将为您的区域启动正确的堆栈：

- [在亚太地区 \(悉尼\) 区域 \(ap-southeast-2 \) 上线](#)
- [在亚太地区 \(东京\) 区域 \(ap-northeast-1 \) 上线](#)
- [在欧洲 \(法兰克福\) 地区 \(eu-central-1 \) 上线](#)
- [在欧洲 \(爱尔兰\) 地区 \(eu-west-1 \) 上线](#)
- [在美国东部 \(弗吉尼亚北部\) 区域 \(us-east-1 \) 上线](#)
- [在美国西部 \(俄勒冈\) 区域 \(us-west-2 \) 上线](#)

2. 在创建堆栈页面上，提供以下值：

- 为堆栈指定一个唯一的名称（例如，###-KVS-SageMaker）。
- 提供您在上一 SageMaker 节中创建的终端节点名称（不是终端节点 ARN）。
- 提供您的 Kinesis 视频流的名称。如果您有多个 Kinesis 视频流，请用引号提供直播名称，并用逗号分隔。
- 其余设置保持原样。

选择下一步。

3. 在选项页面上，保留原来的设置。

4. 选中“我确认 AWS CloudFormation 可能会创建 IAM 资源”复选框。选择下一步。

AWS CloudFormation 创建应用程序。

下表列出了使用此 AWS CloudFormation 模板创建堆栈时 Docker 容器使用的几个参数。它们的值是在模板的 SSM 资源中预定义的，但您可以按需对其进行自定义。

资源名称	默认值	描述
------	-----	----

inferenceInterval	6	The sampling ratio for video frames that are sent to the SageMaker endpoint. The default value of 6 means that only 1 out of every 6 frames support inferencing on I-Frames. The default value of 6 means that only 1 out of every 6 frames sent to the SageMaker endpoint.
sageMakerTaskQueueSize	5000	The size of the queue that maintains the pending requests to the SageMaker endpoint. The size of the queue is affected by 'inferenceInterval' and 'sageMakerTaskThreadPoolSize'. If SageMaker inference takes longer, requests are buffered in this queue.
sageMakerTaskThreadPoolSize	20	Number of threads that's used to concurrently carry out SageMaker inference requests.
sageMakerTaskTimeoutInMilli	20000	The maximum duration accepted for a single request (or a retry request) to the SageMaker endpoint.
sageMakerTaskThreadPoolName	SageMakerThreadPool-%d	The name of the threadpool that's sending requests to the SageMaker endpoint.

要自定义这些参数的值，请在创建堆栈页面上选择模板网址来下载模板，然后在模板的Params部分中找到这些参数，如下所示：AWS CloudFormation

```
Params:
  Type: AWS::SSM::Parameter
  Properties:
    Name:
      Ref: AppName
    Description: "Configuration for SageMaker app"
    Type: String
    Value:
      Fn::Sub: |
        {"streamNames":["${StreamNames}], "tagFilters":
        [${TagFilters}], "sageMakerEndpoint":"${SageMakerEndpoint}",
        "endPointAcceptContentType": "${EndPointAcceptContentType}",

        "kdsStreamName":"${Kds}", "inferenceInterval":6, "sageMakerTaskQueueSize":5000,
        "sageMakerTaskThreadPoolSize":20, "sageMakerTaskTimeoutInMilli":20000,
        "sageMakerTaskThreadPoolName": "SageMakerThreadPool-%d"}
```

监视应用程序

该 AWS CloudFormation 模板创建的应用程序包括一个 Amazon CloudWatch 控制面板和一个用于监控应用程序指标和事件的 CloudWatch 日志流。

应用程序控制面板

该应用程序包括一个用于监控应用程序指标的 CloudWatch 仪表板。要查看应用程序仪表板，请打开 CloudWatch 控制台 <https://console.aws.amazon.com/cloudwatch/>，然后在左侧导航栏中选择仪表板。

KVS-SageMaker-Driver--KvsSageMakerIntegration aws-region #####该控制面板显示以下信息：

- 帧指标：处理视频流、向 SageMaker 端点发送帧以及写入 Kinesis 数据流的指标，该数据流将 SageMaker 笔记本与处理 SageMaker 推理输出结果的 AWS Lambda 函数连接起来。
- IngestToProcessLatency：将视频帧摄入到 Kinesis Video Streams 服务与应用程序接收该帧之间的时间差。
- 当前租赁总额：应用程序被授予使用租约从 Kinesis 视频流中读取内容的权限。此指标显示活动租赁数。该应用程序为每个 Kinesis 视频流使用一个租约，并使用一个租约用于流之间的同步。
- 租赁同步指标：权限租赁同步的频率和持续时间。
- LeaseCount 每个工作线程：SageMaker 工作线程之间的租约分配。
- 工作线程数：处理流的 SageMaker 工作线程的数量。Amazon ECS 集群中的每项任务都有一个工作线程在运行。一个工作线程可以处理多个流。
- ECS 服务利用率：Amazon ECS 集群的使用率指标。
- KinesisDataStream：Kinesis 数据流的使用指标。
- SageMaker：SageMaker 笔记本执行的操作。
- Lambda：处理笔记本输出的 Lambda 函数的数量和持续时间。SageMaker

如果这些图表中的任何信息指示有操作问题（例如值持续增长而不是保持稳定），请参阅以下部分，了解如何查看应用程序日志来确定问题。

CloudWatch 日志

该应用程序包括两个 CloudWatch 日志：

主题

- [应用程序日志](#)
- [Lambda 函数日志](#)

应用程序日志

您可以使用应用程序日志监控应用程序事件和出错情况。如果您需要就问题联系产品支持，也可以使用此日志。

查看应用程序日志

1. 打开 Amazon ECS 控制台，[网址为 https://console.aws.amazon.com/ecs](https://console.aws.amazon.com/ecs)。
2. 选择 KVS-Sagemaker-Driver 集群。
3. 在“服务”选项卡中选择####- SageMakerDriverService 服务。
4. 选择日志选项卡。

应用程序日志显示初始化、配置和租赁活动等事件。

Lambda 函数日志

您可以使用 Lambda 函数日志来跟踪成功的对象识别。

要读取 Lambda 日志

1. 打开 AWS Lambda 控制台，[网址为 https://console.aws.amazon.com/lambda](https://console.aws.amazon.com/lambda)。
2. 为您的应用程序选择 Lambda 函数。Lambda 函数名称采用以下格式：

```
stack-name-LambdaFunction-A1B2C3D4E5F6G
```

3. 选择监控面板。
4. 选择“查看登录信息” CloudWatch。

应用程序 CloudWatch 日志显示成功识别 Kinesis 视频流中的对象以及其他应用程序事件。

扩展应用程序

您可以通过修改您在 AWS CloudFormation 模板窗口中提供的值来为应用程序添加自定义功能，如下所示：

- `EndPointAcceptContentType` : 如果您的 SageMaker 端点不接受 JPG 格式的帧，则可以更改此值。支持以下格式：
 - `image/jpeg`
 - `image/png`
 - `image/bmp`
 - `image/gif`
 - `application/x-image`
- `LambdaFunctionBucket` , `LambdaFunctionKey` : 提供的设置使用处理 SageMaker 输出并将其写入 CloudWatch 日志的 AWS Lambda 函数。如果您想将 SageMaker 输出发送到其他地方，则可以提供自己的 Lambda 函数。
- 标签筛选条件 : 如果您有使用 [the section called “TagStream”](#) 操作标记的流，则可以指定要处理的流的标签。例如，如果您有两个流具有 `Location` 键，其值分别为 `Front` 和 `Parking`，您应使用以下条目，筛选为仅使用这些流：

```
{"key": "Location", "values": ["Front", "Parking"]}
```

清理应用程序

在您完成了为本教程创建的应用程序之后，建议您删除不希望保留的任何资源，以免持续产生费用。

1. SageMaker 终端节点 : 如果您为本教程创建了 SageMaker 终端节点，而不是使用现有的终端节点，请删除该终端节点。在 SageMaker 控制面板中，选择端点配置。选择您创建的终端节点，然后选择操作、删除。确认删除操作。
2. SageMaker 笔记本 : 在 SageMaker 控制台上，选择笔记本实例。选择您创建的笔记本，然后选择操作、停止。当笔记本的状态显示为已停止时，选择操作、删除。确认删除操作。

Note

有关清理 SageMaker 资源的更多信息，请参阅[SageMaker 开发者指南](#)中的[清理](#)。

3. SageMaker 执行策略 : 在 IAM 控制台的导航窗格中，选择策略。选择您为本教程创建的策略。策略的名称类似于以下内容: `AmazonSageMaker-ExecutionPolicy-timestamp`。

选择 Policy actions (策略操作)、Delete (删除)。确认删除操作。

4. SageMaker 执行角色：在 IAM 控制台的导航窗格中，选择角色。选择您为本教程创建的角色。该角色的名称类似于以下名称：AmazonSageMaker-ExecutionRole-*timestamp*。

选择删除角色。确认删除操作。

5. AWS CloudFormation stack：在 AWS CloudFormation 控制台上，选择您为本教程创建的堆栈。依次选择操作和删除堆栈。确认删除操作。
6. Amazon S3 存储桶：在 Amazon S3 控制台上，选择您创建的用于存储 SageMaker 资产的存储桶。选择 Delete（删除）。输入存储桶名称，然后选择确认以确认删除。
7. Kinesis 视频流：在 Kinesis Video Streams 控制台上，选择你为应用程序创建的视频流。选择 Delete。确认删除操作。

监控 Amazon Kinesis Video Streams

监控是维护 Amazon Kinesis Video Streams 和 AWS 您的解决方案的可靠性、可用性和性能的重要组成部分。我们建议从 AWS 解决方案的所有部分收集监控数据，以帮助您在出现多点故障时进行调试。在开始监控 Amazon Kinesis Video Streams 之前，我们建议您制定一份包含以下问题答案的监控计划：

- 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？

在您定义了监控目标并制定了监控计划之后，下一步是为环境中正常的 Amazon Kinesis Video Streams 性能建立基准。您应该衡量 Amazon Kinesis Video Streams 在不同时间和不同负载条件下的性能。在监控 Amazon Kinesis Video Streams 时，请存储您收集的监控数据的历史记录。您可以将当前的 Amazon Kinesis Video Streams 性能与历史数据进行比较，以帮助您识别正常的性能模式和性能异常，并设计解决可能出现的问题的方法。

主题

- [使用以下方式监控 Amazon Kinesis Video Streams 指标 CloudWatch](#)
- [使用以下方式监控 Amazon Kinesis Video Streams Edge Agent CloudWatch](#)
- [使用记录亚马逊 Kinesis Video Streams Video Streams API 调用 AWS CloudTrail](#)

使用以下方式监控 Amazon Kinesis Video Streams 指标 CloudWatch

您可以使用亚马逊监控 Kinesis 视频流 CloudWatch，亚马逊会收集来自亚马逊 Kinesis Video Streams 的原始数据，并将其处理为可读的、近乎实时的指标。这些统计数据记录的时间为 15 个月，因此您可以访问历史信息并更好地了解您的 Web 应用程序或服务的性能。

在[亚马逊 Kinesis Video Stream](#)s 控制台中，您可以通过两种方式 CloudWatch 查看亚马逊 Kinesis 视频流的指标：

- 在“控制面板”页面中，在“当前区域的账户级指标”部分中选择“视频流”选项卡。
- 在视频流的详细信息页面中，选择 Monitoring (监控) 选项卡。

亚马逊 Kinesis Video Streams 提供以下指标：

指标	描述
ArchivedFragmentsConsumed.Media	所有 API 消耗的片段媒体配额点数。有关配额积分概念的解释，请参阅 the section called “片段元数据和片段媒体配额” 。 单位：计数
ArchivedFragmentsConsumed.Metadata	所有 API 消耗的片段元数据配额点的数量。有关配额积分概念的解释，请参阅 the section called “片段元数据和片段媒体配额” 。 单位：计数
PutMedia.Requests	给定直播PutMedia的 API 请求数。 单位：计数
PutMedia.IncomingBytes	作为流的一部分接收PutMedia的字节数。 单位：字节
PutMedia.IncomingFragments	作为直播一部分收到的完整片段PutMedia的数量。 单位：计数
PutMedia.IncomingFrames	作为直播一部分接收PutMedia的完整帧数。 单位：计数
PutMedia.ActiveConnections	与服务主机的连接总数。 单位：计数
PutMedia.ConnectionErrors	为直播建立PutMedia连接时出现的错误。


指标	描述
	单位：计数
PutMedia.FragmentIngestionLatency	Amazon Kinesis Video Streams 接收片段的第一个字节和最后一个字节之间的时间差。 单位：毫秒
PutMedia.FragmentPersistLatency	从接收和存档完整片段数据起所花费的时间。 单位：计数
PutMedia.Latency	请求与建立连接时的 HTTP 响应之间的时间差。 InletService 单位：计数
PutMedia.BufferingAckLatency	Amazon Kinesis Video Streams 接收新片段的第一个字节与为该片段发送缓冲 ACK 之间的时间差。 单位：毫秒
PutMedia.ReceivedAckLatency	Amazon Kinesis Video Streams 收到新片段的最后一个字节与为该片段发送已收到的 ACK 之间的时间差。 单位：毫秒
PutMedia.PersistedAckLatency	Amazon Kinesis Video Streams 接收新片段的最后一个字节与为该片段发送持久的 ACK 之间的时间差。 单位：毫秒
PutMedia.ErrorAckCount	在直播中发送的错误 ACK PutMedia 的数量。 单位：计数
PutMedia.Success	成功写入的每个片段为 1；每个失败的片段为 0。该指标的平均值表示发送的完整有效片段数。 单位：计数

指标	描述
GetMedia.Requests	给定直播GetMedia的 API 请求数。 单位：计数
GetMedia.OutgoingBytes	作为给定流的 GetMedia API 的一部分从服务发送的总字节数。 单位：字节
GetMedia.OutgoingFragments	直播时发送的片段数量。GetMedia 单位：计数
GetMedia.OutgoingFrames	在给定直播中发送GetMedia的帧数。 单位：计数
GetMedia.MillisBehindNow	当前服务器时间戳和上次发送片段的服务器时间戳之间的时差。 单位：毫秒
GetMedia.ConnectionErrors	未成功建立的连接数。 单位：计数
GetMedia.Success	每个成功发送的片段为 1；每个失败的片段为 0。平均值表示成功率。 <div data-bbox="777 1444 818 1482"></div> Note 故障包含 400 (用户) 错误和 500 (系统) 错误。有关启用请求和响应摘要 (包括 AWS 请求 ID) 的更多信息, 请参阅 请求/响应摘要记录 。 单位：计数

指标	描述
GetMediaForFragmentList.OutgoingBytes	<p>作为给定流的 GetMediaForFragmentList API 的一部分从服务发送的总字节数。</p> <p>单位：字节</p>
GetMediaForFragmentList.OutgoingFragments	<p>作为给定流的 GetMediaForFragmentList API 的一部分从服务发送的片段总数。</p> <p>单位：计数</p>
GetMediaForFragmentList.OutgoingFrames	<p>作为给定流的 GetMediaForFragmentList API 的一部分从服务发送的帧总数。</p> <p>单位：计数</p>
GetMediaForFragmentList.Requests	<p>给定直播 GetMediaForFragmentList 的 API 请求数。</p> <p>单位：计数</p>
GetMediaForFragmentList.Success	<p>每个成功发送的片段为 1；每个失败的片段为 0。平均值表示成功率。</p> <div data-bbox="748 1186 1507 1499" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin: 10px 0;"> <p> Note</p> <p>故障包含 400 (用户) 错误和 500 (系统) 错误。有关启用请求和响应摘要 (包括 AWS 请求 ID) 的更多信息, 请参阅请求/响应摘要记录。</p> </div> <p>单位：计数</p>
ListFragments.Latency	<p>ListFragments API 的延迟需要给定的直播名称。</p> <p>单位：毫秒</p>

指标	描述
ListFragments.Requests	<p>给定直播ListFragments 的 API 请求数。</p> <p>单位：计数</p>
ListFragments.Success	<p>每个成功的请求为 1；每个失败的请求为 0。平均值表示成功率。</p> <div data-bbox="748 512 1508 827" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin: 10px 0;"> <p> Note</p> <p>故障包含 400 (用户) 错误和 500 (系统) 错误。有关启用请求和响应摘要 (包括 AWS 请求 ID) 的更多信息, 请参阅请求/响应摘要记录。</p> </div> <p>单位：计数</p>
GetHLSStreamingSessionURL.Latency	<p>GetHLSStreamingSessionURL API 的延迟需要给定的直播名称。</p> <p>单位：毫秒</p>
GetHLSStreamingSessionURL.Requests	<p>给定直播GetHLSStreamingSessionURL 的 API 请求数。</p> <p>单位：计数</p>

指标	描述
GetHLSStreamingSessionURL.Success	<p>每个成功的请求为 1；每个失败的请求为 0。平均值表示成功率。</p> <div style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin: 10px 0;"> <p> Note</p> <p>故障包含 400 (用户) 错误和 500 (系统) 错误。有关启用请求和响应摘要 (包括 AWS 请求 ID) 的更多信息, 请参阅请求/响应摘要记录。</p> </div> <p>单位: 计数</p>
GetHLSMasterPlaylist.Latency	<p>GetHLSMasterPlaylist API 的延迟需要给定的直播名称。</p> <p>单位: 毫秒</p>
GetHLSMasterPlaylist.Requests	<p>给定直播GetHLSMasterPlaylist 的 API 请求数。</p> <p>单位: 计数</p>
GetHLSMasterPlaylist.Success	<p>每个成功的请求为 1；每个失败的请求为 0。平均值表示成功率。</p> <div style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin: 10px 0;"> <p> Note</p> <p>故障包含 400 (用户) 错误和 500 (系统) 错误。有关启用请求和响应摘要 (包括 AWS 请求 ID) 的更多信息, 请参阅请求/响应摘要记录。</p> </div> <p>单位: 计数</p>

指标	描述
GetHLSMediaPlaylist.Latency	GetHLSMediaPlaylist API 的延迟需要给定的直播名称。 单位：毫秒
GetHLSMediaPlaylist.Requests	给定直播GetHLSMediaPlaylist 的 API 请求数。 单位：计数
GetHLSMediaPlaylist.Success	每个成功的请求为 1；每个失败的请求为 0。平均值表示成功率。 <div data-bbox="748 768 1507 1083" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px;"> <p> Note</p> <p>故障包含 400 (用户) 错误和 500 (系统) 错误。有关启用请求和响应摘要 (包括 AWS 请求 ID) 的更多信息, 请参阅请求/响应摘要记录。</p> </div> 单位：计数
GetMP4InitFragment.Latency	GetMP4InitFragment API 的延迟需要给定的直播名称。 单位：毫秒
GetMP4InitFragment.Requests	给定直播GetMP4InitFragment 的 API 请求数。 单位：计数

指标	描述
GetMP4InitFragment.Success	<p>每个成功的请求为 1；每个失败的请求为 0。平均值表示成功率。</p> <div data-bbox="748 352 1508 667"><p> Note</p><p>故障包含 400 (用户) 错误和 500 (系统) 错误。有关启用请求和响应摘要 (包括 AWS 请求 ID) 的更多信息, 请参阅请求/响应摘要记录。</p></div> <p>单位: 计数</p>
GetMP4MediaFragment.Latency	<p>GetMP4MediaFragment API 的延迟需要给定的直播名称。</p> <p>单位: 毫秒</p>
GetMP4MediaFragment.Requests	<p>给定直播GetMP4MediaFragment 的 API 请求数。</p> <p>单位: 计数</p>
GetMP4MediaFragment.Success	<p>每个成功的请求为 1；每个失败的请求为 0。平均值表示成功率。</p> <div data-bbox="748 1360 1508 1675"><p> Note</p><p>故障包含 400 (用户) 错误和 500 (系统) 错误。有关启用请求和响应摘要 (包括 AWS 请求 ID) 的更多信息, 请参阅请求/响应摘要记录。</p></div> <p>单位: 计数</p>

指标	描述
GetMP4MediaFragment.OutgoingBytes	<p>作为给定流的 GetMP4MediaFragment API 的一部分从服务发送的总字节数。</p> <p>单位：字节</p>
GetTSFragment.Latency	<p>GetTSFragment API 的延迟需要给定的直播名称。</p> <p>单位：毫秒</p>
GetTSFragment.Requests	<p>给定直播GetTSFragment 的 API 请求数。</p> <p>单位：计数</p>
GetTSFragment.Success	<p>每个成功的请求为 1；每个失败的请求为 0。平均值表示成功率。</p> <div data-bbox="748 884 1507 1192" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin: 10px 0;"> <p> Note</p> <p>故障包含 400 (用户) 错误和 500 (系统) 错误。有关启用请求和响应摘要 (包括 AWS 请求 ID) 的更多信息, 请参阅请求/响应摘要记录。</p> </div> <p>单位：计数</p>
GetTSFragment.OutgoingBytes	<p>作为给定流的 GetTSFragment API 的一部分从服务发送的总字节数。</p> <p>单位：字节</p>
GetDASHStreamingSessionURL.Latency	<p>GetDASHStreamingSessionURL API 的延迟需要给定的直播名称。</p> <p>单位：毫秒</p>

指标	描述
GetDASHStreamingSessionURL. Requests	给定直播GetDASHStreamingSessionURL 的 API 请求数。 单位：计数
GetDASHStreamingSessionURL. Success	每个成功的请求为 1；每个失败的请求为 0。平均值表示成功率。 <div data-bbox="748 562 1507 873" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px;"><p> Note</p><p>故障包含 400 (用户) 错误和 500 (系统) 错误。有关启用请求和响应摘要 (包括 AWS 请求 ID) 的更多信息, 请参阅请求/响应摘要记录。</p></div> 单位：计数
GetDASHManifest.Latency	GetDASHManifest API 的延迟需要给定的直播名称。 单位：毫秒
GetDASHManifest.Requests	给定直播GetDASHManifest 的 API 请求数。 单位：计数

指标	描述
GetDASHManifest.Success	<p>每个成功的请求为 1；每个失败的请求为 0。平均值表示成功率。</p> <div data-bbox="748 352 1508 667"><p> Note</p><p>故障包含 400 (用户) 错误和 500 (系统) 错误。有关启用请求和响应摘要 (包括 AWS 请求 ID) 的更多信息, 请参阅请求/响应摘要记录。</p></div> <p>单位: 计数</p>
GetClip.Latency	<p>GetClip API 的延迟需要给定的视频流名称。</p> <p>单位: 毫秒</p>
GetClip.Requests	<p>给定视频流 GetClip 的 API 请求数。</p> <p>单位: 计数</p>
GetClip.Success	<p>每个成功的请求为 1；每个失败的请求为 0。平均值表示成功率。</p> <div data-bbox="748 1262 1508 1577"><p> Note</p><p>故障包含 400 (用户) 错误和 500 (系统) 错误。有关启用请求和响应摘要 (包括 AWS 请求 ID) 的更多信息, 请参阅请求/响应摘要记录。</p></div> <p>单位: 计数</p>

指标	描述
GetClip.OutgoingBytes	作为给定视频流 GetClip API 的一部分从服务发送的总字节数。 单位：字节

CloudWatch 指标指导

CloudWatch 指标可以帮助找到以下问题的答案：

主题

- [数据是否会到达亚马逊 Kinesis Video Streams 服务？](#)
- [为什么 Amazon Kinesis Video Streams 服务无法成功提取数据？](#)
- [为什么从 Amazon Kinesis Video Streams 服务读取数据的速率不能与从制作者发送的数据的速率相同？](#)
- [为什么控制台中没有视频，或者为什么视频播放出现延迟？](#)
- [什么是实时数据读取延迟，以及为何客户端会滞后于流头？](#)
- [客户端是否从 Kinesis 视频流中读取数据，读取速率是多少？](#)
- [为什么客户端无法从 Kinesis 视频流中读取数据？](#)

数据是否会到达亚马逊 Kinesis Video Streams 服务？

相关指标：

- PutMedia.IncomingBytes
- PutMedia.IncomingFragments
- PutMedia.IncomingFrames

操作项：

- 如果这些指标有所下降，请检查您的应用程序是否仍在向服务发送数据。
- 检查网络带宽。如果您的网络带宽不足，可能会降低服务接收数据的速率。

为什么 Amazon Kinesis Video Streams 服务无法成功提取数据？

相关指标：

- `PutMedia.Requests`
- `PutMedia.ConnectionErrors`
- `PutMedia.Success`
- `PutMedia.ErrorAckCount`

操作项：

- 如果增加了 `PutMedia.ConnectionErrors`，请查看生产者客户端收到的 HTTP 响应和错误代码，以了解在建立连接时发生了哪些错误。
- 如果出现下降 `PutMedia.Success` 或增加 `PutMedia.ErrorAckCount`，请查看服务发送的 ack 响应中的 ack 错误代码，以了解数据摄取失败的原因。有关更多信息，请参阅 [AckErrorCode.Values](#)。

为什么从 Amazon Kinesis Video Streams 服务读取数据的速率不能与从制作者发送的数据的速率相同？

相关指标：

- `PutMedia.FragmentIngestionLatency`
- `PutMedia.IncomingBytes`

操作项：

- 如果这些指标有所下降，请检查您的连接的网络带宽。低带宽连接可能导致数据到达服务的速率较低。

为什么控制台中没有视频，或者为什么视频播放出现延迟？

相关指标：

- `PutMedia.FragmentIngestionLatency`
- `PutMedia.FragmentPersistLatency`
- `PutMedia.Success`

- `ListFragments.Latency`
- `PutMedia.IncomingFragments`

操作项：

- 如果网络带宽增加`PutMedia.FragmentIngestionLatency`或减少`PutMedia.IncomingFragments`，请检查网络带宽以及数据是否仍在发送中。
- 如果有漏洞`PutMedia.Success`，请检查`ack`错误代码。有关更多信息，请参阅 [AckErrorCode.Values](#)。
- 如果`PutMedia.FragmentPersistLatency`或增加`ListFragments.Latency`，则很可能遇到了服务问题。如果这种情况持续很长时间，请咨询您的客户服务联系人，看看您的服务是否存在问题。

什么是实时数据读取延迟，以及为何客户端会滞后于流头？

相关指标：

- `GetMedia.MillisBehindNow`
- `GetMedia.ConnectionErrors`
- `GetMedia.Success`

操作项：

- 如果流量增加`GetMedia.ConnectionErrors`，则由于频繁尝试重新连接到直播，消费者可能会在阅读直播方面落后。请查看针对 `GetMedia` 请求所返回的 HTTP 响应/错误代码。
- 如果流量下降`GetMedia.Success`，则可能是由于服务无法将数据发送给消费者，这将导致连接中断，并导致消费者重新连接，从而导致消费者落后于直播的头部。
- 如果带宽增加`GetMedia.MillisBehindNow`，请查看您的带宽限制，以查看是否因为带宽较低而导致以较慢的速率接收数据。

客户端是否从 Kinesis 视频流中读取数据，读取速率是多少？

相关指标：

- `GetMedia.OutgoingBytes`

- `GetMedia.OutgoingFragments`
- `GetMedia.OutgoingFrames`
- `GetMediaForFragmentList.OutgoingBytes`
- `GetMediaForFragmentList.OutgoingFragments`
- `GetMediaForFragmentList.OutgoingFrames`

操作项：

- 这些指标表示读取实时和存档数据的速率。

为什么客户端无法从 Kinesis 视频流中读取数据？

相关指标：

- `GetMedia.ConnectionErrors`
- `GetMedia.Success`
- `GetMediaForFragmentList.Success`
- `PutMedia.IncomingBytes`

操作项：

- 如果增加了 `GetMedia.ConnectionErrors`，请查看 `GetMedia` 请求返回的 HTTP 响应和错误代码。有关更多信息，请参阅 [AckErrorCode.Values](#)。
- 如果您正在尝试读取最新或实时数据，`PutMedia.IncomingBytes` 请检查是否有数据进入数据流，以便服务发送给消费者。
- 如果出现下降 `GetMedia.Success` 或 `GetMediaForFragmentList.Success`，则可能是由于服务无法将数据发送给消费者。如果这种情况持续很长时间，请咨询您的客户服务联系人，看看您的服务是否存在问题。

使用以下方式监控 Amazon Kinesis Video Streams Edge Agent CloudWatch

您可以使用亚马逊监控 Amazon Kinesis Video Streams Edge CloudWatch Agent，该代理收集原始数据并将其处理为可读的近乎实时的指标。这些统计数据记录的时间为15个月。利用这些历史信息，您可以更好地了解您的网络应用程序或 Amazon Kinesis Video Streams Edge Agent 服务的性能。

要查看指标，请执行以下操作：

1. 登录 AWS Management Console 并打开 CloudWatch 控制台，[网址为 https://console.aws.amazon.com/cloudwatch/](https://console.aws.amazon.com/cloudwatch/)。
2. 在左侧导航栏的“指标”下，选择“所有指标”。
3. 选择“浏览”选项卡，然后选择EdgeRuntimeAgent自定义命名空间。

Amazon Kinesis Video Streams Edge Agent 在命名空间下发布以下指标：EdgeRuntimeAgent

尺寸	省/自治区/直辖市	描述
直播名称，RecordJob	Running	运行时RecordJob 持续发布。 单位：无。只要RecordJob 处于此状态，“1”就会持续发布。
	FatalError	如果出现RecordJob 致命错误，则发布。 单位：无。此事件发生时，“1”仅发布一次。 <div style="border: 1px solid #ccc; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p> Note 有关其他信息，请参阅日志。</p> </div>
	已完成	在 a 完成RecordJob 时发布。 单位：无。此事件发生时，“1”仅发布一次。
直播名称，UploadJob	Running	运行时UploadJob 持续发布。 单位：无。只要UploadJob 处于此状态，“1”就会持续发布。

尺寸	省/自治区/直辖市	描述
	FatalError	<p>如果出现UploadJob 致命错误，则发布。</p> <p>单位：无。此事件发生时，“1”仅发布一次。</p> <div style="border: 1px solid #00a0e3; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> Note 有关其他信息，请参阅日志。</p> </div>
	已完成	<p>完成后UploadJob 发布。</p> <p>单位：无。此事件发生时，“1”仅发布一次。</p>
流名称	PercentageSpaceUsed	<p>这是在 Amazon Kinesis Video Streams Edge Agent 配置中为录制媒体分配的总空间中使用的百分比。请参阅the section called “LocalSizeConfig”了解更多信息。</p> <p>单位：百分比（比例为 0—1）。</p>
事务名称	还活着	<p>每分钟从 Amazon Kinesis Video Streams Video Streams Edge Agent 上发布一次，无论其上运行任何配置。</p> <p>这可以用来了解 Amazon Kinesis Video Streams Edge Agent 是否处于活动状态并准备好接受配置。</p> <p>单位：无。“1”每分钟发布一次。</p>
	RecordJobs.HealthyJobCount	<p>Amazon Kinesis Video Streams Edge Agent 上正在运行和计划录制的作业总数。</p> <p>单位：计数。</p>
	UploadJobs.HealthyJobCount	<p>亚马逊 Kinesis Video Streams Edge Agent 上正在运行和计划的上传任务总数。</p> <p>单位：计数。</p>

尺寸	省/自治区/直辖市	描述
	RecordJobs.UnhealthyJobCount	当前出错的记录作业总数。 单位：计数。
	UploadJobs.UnhealthyJobCount	当前出错的上传任务总数。 单位：计数。
	RecordJobs.RunningJobCount	活跃运行的记录作业总数。 单位：计数。
	UploadJobs.RunningJobCount	正在运行的上传任务总数。 单位：计数。
	RecordJobs.EdgeConfigCount	亚马逊 Kinesis Video Streams Edge Agent 上正在处理的记录配置总数。 单位：计数。
	UploadJobs.EdgeConfigCount	亚马逊 Kinesis Video Streams Edge Agent 上正在处理的上传配置总数。 单位：计数。

CloudWatch 亚马逊 Kinesis Video Streams Edge Agent 的指标指南

CloudWatch 指标对于寻找以下问题的答案非常有用：

主题

- [亚马逊 Kinesis Video Streams Video Streams Edge Agent 有足够的录制空间吗？](#)
- [亚马逊 Kinesis Video Streams Video Streams 边缘代理还活着吗？](#)
- [有没有不健康的工作？](#)
- [有工作需要外部干预吗？](#)

亚马逊 Kinesis Video Streams Video Streams Edge Agent 有足够的录制空间吗？

相关指标：PercentageSpaceUsed

操作：无需执行任何操作。

亚马逊 Kinesis Video Streams Video Streams 边缘代理还活着吗？

相关指标：Alive

操作：如果您在任何时候停止接收此指标，则意味着 Amazon Kinesis Video Streams Edge Agent 遇到了以下一项或多项情况：

- 应用程序运行时问题：内存或其他资源限制、错误等
- 代理在关闭、崩溃或终止时正在运行的 AWS IoT 设备
- AWS IoT 设备没有网络连接

有没有不健康的工作？

相关指标：

- RecordJobs.UnhealthyJobCount
- UploadJobs.UnhealthyJobCount

操作：检查日志并查找FatalError指标。

- 如果存在该**FatalError**指标，则会遇到致命错误，您需要手动重启作业。在使用手动重启作业之前，StartEdgeConfigurationUpdate请检查日志并修复问题。
- 如果该FatalError指标不存在，则会遇到暂时（非致命）错误，Amazon Kinesis Video Streams Edge Agent 正在重试该作业。

Note

要让代理重新尝试严重错误的作业，请使用 [the section called “StartEdgeConfigurationUpdate”](#)

有工作需要外部干预吗？

相关指标：

- `PercentageSpaceUsed`— 如果超过一定值，则录制作业将暂停并仅在空间可用时恢复（当媒体保留期满时）。您可以发送更高版本的更新配置，`MaxLocalMediaSizeInMB`以便立即更新作业。
- `RecordJob.FatalError/UploadJob.FatalError`— 调查代理的日志并再次发送配置以恢复作业。

操作：使用配置进行 API 调用，以重启遇到此问题的作业。

使用记录亚马逊 Kinesis Video Streams Video Streams API 调用 AWS CloudTrail

Amazon Kinesis Video Streams AWS CloudTrail 与该服务配合使用，可记录用户、角色或用户在 Amazon Kinesis Video Streams 中采取的操作。CloudTrail 将亚马逊 Kinesis Video Streams 的所有 API 调用捕获为事件。捕获的调用包括来自亚马逊 Kinesis Video Streams 控制台的调用和对亚马逊 Kinesis Video Streams 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，包括亚马逊 Kinesis Video Streams 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，您可以确定向 Amazon Kinesis Video Streams 发出的请求、发出请求的 IP 地址、谁发出了请求、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 Streams 和 CloudTrail

CloudTrail 在您创建 AWS 账户时已在您的账户上启用。当 Amazon Kinesis Video Streams Video Streams 中出现支持的事件活动时 CloudTrail，该活动将 AWS 与其他服务事件一起记录在事件历史记录中。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，包括亚马逊 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 支持将以下操作作为事件记录 CloudTrail 在日志文件中：

- [CreateStream](#)
- [DeleteStream](#)
- [DescribeStream](#)
- [GetDataEndpoint](#)
- [ListStreams](#)
- [ListTagsForStream](#)
- [TagStream](#)
- [UntagStream](#)
- [UpdateDataRetention](#)
- [UpdateStream](#)

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：

- 请求是使用根用户凭证还是 用户凭证发出的
- 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：亚马逊 Kinesis Video Streams Video Streams 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。一个事件表示来自任何源的一个请求，包括有关所请求的操作、操作的日期和时间、请求参数等方面的信息。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[CreateStream](#)操作的 CloudTrail 日志条目。

```
{
  "Records": [
    {
      "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": "2018-05-25T00:16:31Z",
      "eventSource": "kinesisvideo.amazonaws.com",
      "eventName": "CreateStream",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "127.0.0.1",
      "userAgent": "aws-sdk-java/unknown-version Linux/x.xx",
      "requestParameters": {
        "streamName": "VideoStream",
        "dataRetentionInHours": 2,
        "mediaType": "mediaType",
        "kmsKeyId": "arn:aws:kms::us-east-1:123456789012:alias",
        "deviceName": "my-device"
      },
      "responseElements": {
        "streamARN": "arn:aws:kinesisvideo:us-east-1:123456789012:stream/VideoStream/12345"
      },
      "requestID": "db6c59f8-c757-11e3-bc3b-57923b443c1c",
      "eventID": "b7acfc0-6ca9-4ee1-a3d7-c4e8d420d99b"
    },
    {
      "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": "2018-05-25:17:06Z",
      "eventSource": "kinesisvideo.amazonaws.com",
```

```

    "eventName": "DeleteStream",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "aws-sdk-java/unknown-version Linux/x.xx",
    "requestParameters": {
      "streamARN": "arn:aws:kinesisvideo:us-east-1:012345678910:stream/
VideoStream/12345",
      "currentVersion": "keqrjeqkj9"
    },
    "responseElements": null,
    "requestID": "f0944d86-c757-11e3-b4ae-25654b1d3136",
    "eventID": "0b2f1396-88af-4561-b16f-398f8eaea596"
  },
  {
    "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": "2014-04-19T00:15:02Z",
    "eventSource": "kinesisvideo.amazonaws.com",
    "eventName": "DescribeStream",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "aws-sdk-java/unknown-version Linux/x.xx",
    "requestParameters": {
      "streamName": "VideoStream"
    },
    "responseElements": null,
    "requestID": "a68541ca-c757-11e3-901b-cbcfe5b3677a",
    "eventID": "22a5fb8f-4e61-4bee-a8ad-3b72046b4c4d"
  },
  {
    "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": "2014-04-19T00:15:03Z",
    "eventSource": "kinesisvideo.amazonaws.com",
    "eventName": "GetDataEndpoint",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "aws-sdk-java/unknown-version Linux/x.xx",
    "requestParameters": {
        "streamName": "VideoStream",
        "apiName": "LIST_FRAGMENTS"
    }
},
"responseElements": null,
"requestID": "a6e6e9cd-c757-11e3-901b-cbcfe5b3677a",
"eventID": "dcd2126f-c8d2-4186-b32a-192dd48d7e33"
},
{
    "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": "2018-05-25T00:16:56Z",
    "eventSource": "kinesisvideo.amazonaws.com",
    "eventName": "ListStreams",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "aws-sdk-java/unknown-version Linux/x.xx",
    "requestParameters": {
        "maxResults": 100,
        "streamNameCondition": {"comparisonValue": "MyVideoStream"
comparisonOperator": "BEGINS_WITH"}}
    },
    "responseElements": null,
    "requestID": "e9f9c8eb-c757-11e3-bf1d-6948db3cd570",
    "eventID": "77cf0d06-ce90-42da-9576-71986fec411f"
}
]
}

```


Kinesis Video Streams 服务配额

Kinesis Video Streams 有以下服务配额：

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] 。


控制平面 API 服务配额

以下部分介绍控制平面 API 的服务配额。TPS 代表每秒的交易次数。

当达到账户级别或资源级别的请求限制时，会抛出 `ClientLimitExceededException`

控制平面 API 服务配额

API	账户限制：申请	账号限制：直播	流级别限制	相关的例外情况和注意事项
the section called “CreateStream”	50 TPS [s]	在美国东部 (弗吉尼亚北部) 和美国西部 (俄勒冈) 地区每个账户 10000 个直播。在所有其他支持的区域，每个账户 5000 个直播。		设备、CLI、SDK 驱动的和控制台均可调用此 API。如果流尚不存在，则只有一个 API 调用会成功。

 **Note**

可将此限制提

API	账户限制：申请	账号限制：直播	流级别限制	相关的例外情况和注意事项
		<p>高至每个账户 100,000 个（或更多个）流。通过以下网址登录 AWS Management Console： console.aws.amazon.com/， 并提交服务限制提高案例让 Kinesis Video Streams 请求提高此限制。</p>		

API	账户限制：申请	账号限制：直播	流级别限制	相关的例外情况和注意事项
the section called “DeleteEdgeConfiguration”	10 TPS [h]	不适用	1 TPS [h]	
the section called “DeleteStream”	50 TPS [h]	不适用	5 TPS [h]	
the section called “DescribeEdgeConfiguration”	50 TPS [h]	不适用	5 TPS [h]	
the section called “DescribeImageGenerationConfiguration”	50 TPS [h]	不适用	5 TPS [h]	
the section called “DescribeMappedResourceConfiguration”	50 TPS [h]	不适用	5 TPS [h]	

API	账户限制：申请	账号限制：直播	流级别限制	相关的例外情况和注意事项
the section called “DescribeNotificationConfiguration”	50 TPS [h]	不适用	5 TPS [h]	
the section called “DescribeStream”	300 TPS [h]	不适用	5 TPS [h]	
the section called “GetDataEndpoint”	300 TPS [h]	不适用	5 TPS [h]	每 45 分钟调用一次，旨在刷新大多数 PutMedia/GetMedia 使用案例的流令牌。如果应用程序在失败时重新加载缓存数据终端节点，那么这些终端节点是安全的。
the section called “ListEdgeAgentConfigurations”	50 TPS [h]	不适用	不适用	
the section called “ListStreams”	50 TPS [h]	不适用		
the section called “ListTagsForStream”	50 TPS [h]	不适用	5 TPS [h]	

API	账户限制：申请	账号限制：直播	流级别限制	相关的例外情况和注意事项
the section called “StartEdgeConfigurationUpdate”	10 TPS [h]	不适用	1 TPS [h]	
the section called “TagStream”	50 TPS [h]	不适用	5 TPS [h]	
the section called “UntagStream”	50 TPS [h]	不适用	5 TPS [h]	
the section called “UpdateDataRetention”	50 TPS [h]	不适用	5 TPS [h]	
the section called “UpdateImageGenerationConfiguration”	50 TPS [h]	不适用	5 TPS [h]	
the section called “UpdateNotificationConfiguration”	50 TPS [h]	不适用	5 TPS [h]	

API	账户限制：申请	账号限制：直播	流级别限制	相关的例外情况和注意事项
the section called “UpdateStream”	50 TPS [h]	不适用	5 TPS [h]	

媒体和存档媒体 API 服务配额

以下部分介绍媒体和存档媒体 API 的服务配额。

当达到账户级别或资源级别的请求限制时，会抛出 `ClientLimitExceededException`

达到连接级别限制时，会引发 `ConnectionLimitExceededException`。

达到片段级别限制时，会引发以下错误或确认：

- 为低于最低持续时间的片段返回 `MIN_FRAGMENT_DURATION_REACHED` 确认。
- 会为高于最高持续时间的片段返回 `MAX_FRAGMENT_DURATION_REACHED` 确认。
- 会为高于最大数据大小的片段返回 `MAX_FRAGMENT_SIZE` 确认。
- 如果在 `GetMediaForFragmentList` 操作中达到片段限制，则会引发 `FragmentLimitExceeded` 异常。

数据层面 API 服务配额

API	流级别限制	连接级别限制	带宽限制	片段级别限制	相关的例外情况和注意事项
the section called “PutMedia”	5 TPS [h]	1 [s]	12.5 MB/秒，或每个直播 100 Mbps [s]	<ul style="list-style-type: none"> • 最小片段持续时间：1 秒 [h] • 最大片段持续时间：20 秒 [h] 	典型 <code>PutMedia</code> 请求包含几秒钟的数据，因而导致每个流较低的 TPS。如果有多个并发连接超过配额，则接受最后一个连接。

API	流级别限制	连接级别限制	带宽限制	片段级别限制	相关的例外情况和注意事项
				<ul style="list-style-type: none"> • 最大片段大小：50 MB [h] • 轨道的最大数量：3 [s] • 每秒发送的最大片段：5 [h] • 片段元数据最大限制：10 个标签 [h] 	
the section called “GetClip”	不适用	不适用	100 MB 大小限制 [h]	最大片段数：200 [h]	
the section called “GetDASHStreamingSessionURL”	25 TPS [h]	不适用	不适用	不适用	
the section called “GetHLSStreamingSessionURL”	25 TPS [h]	不适用	不适用	不适用	

API	流级别限制	连接级别限制	带宽限制	片段级别限制	相关的例外情况和注意事项
the section called “GetImages”	不适用	不适用	100 MB [h]	不适用	<p>每次请求的最大图像数为 100 [h]。</p> <div data-bbox="1133 394 1507 760" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px;"> <p> Note</p> <p>的最小值 <code>SamplingInterval</code> 为 200 毫秒 (ms)，即每秒 5 张图像。</p> </div>
the section called “GetMedia”	5 TPS [h]	3 [s]	25 MB/秒或 200 Mbps [s]	每秒发送的最大片段：6 [h]	<p>一个独特的使用客户端不需要超过两三个 TPS，因为在建立连接后，应用程序应该持续读取。</p> <p>如果典型片段约为 5 MB，则此限制意味着每个 Kinesis 视频流约为 75 Mbps。此类流将具有流的最大传入比特率 2 倍的传出比特率。</p> <div data-bbox="1133 1339 1507 1612" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px;"> <p> Note</p> <p>GetMedia 不用于 HLS/DASH 播放。</p> </div>

API	流级别限制	连接级别限制	带宽限制	片段级别限制	相关的例外情况和注意事项
the section called “GetMediaForFragmentList”	不适用	5 [s]	25 MB/秒或 200 Mbps [s]	最大片段数：1000 [h]	可以同时调用五个基于片段的消费应用程序。GetMediaForFragmentList 后续连接将被拒绝。

视频播放协议 API 服务配额

API	会话级别限制	片段级别限制
GetDash ManifestPlaylist	5 TPS [h]	每个播放列表的最大片段数：5000 [h]
getHLS MasterPlaylist	5 TPS [h]	不适用
getHLS MediaPlaylist	5 TPS [h]	每个播放列表的最大片段数：5000 [h]
getMP4 InitFragment	5 TPS [h]	不适用
getMP4 MediaFragment	20 TPS [h]	不适用
GetTSFragment	20 TPS [h]	不适用

片段元数据和片段媒体配额

Kinesis Video [Streams 访问存档媒体的 API](#) 是根据请求的片段数量而不是 API 调用次数进行限制的。API 的速率受片段元数据数量和请求的片段媒体数量的限制。片段元数据和片段媒体配额按流计算。换句话说，对一个流中的片段元数据或媒体的请求不适用于另一个流的配额。但是，在给定的流中，每个配额都是在多个 API 之间共享的。这意味着，对于给定的流，跨不同 API 的片段请求消耗的配额相同。当超过直播的片段元数据或片段媒体配额时，API 会返回 `ClientLimitExceededException`。下表显示了 API 如何使用这两种类型的配额。对于这些表中的第二列，假设如果某个流的配额为 N，则意味着 API 在该流的配额类型中有 N 个积分可供消耗。GetClipAPI 出现在两个表中。

片段元数据配额消耗

API	每次请求消耗的配额积分	共享配额 (N)
the section called “ListFragments”	MaxResults 参数的值	每个直播每秒 10000 个配额积分 [h]
the section called “GetClip”	生成的片段中的片段数	
GetHLSMediaPlaylist	MaxMediaPlaylistFragmentResults 参数的值	
GetDASHManifest	MaxManifestFragmentResults 参数的值	
the section called “GetImages”	值为 400 + 请求的最大图像数	

片段媒体配额消耗

API	每次请求消耗的配额积分	共享配额 (N)
the section called “GetMediaForFragmentList”	片段参数中的片段数	每个直播每秒 500 个配额积分 [h]
the section called “GetClip”	生成的片段中的片段数	
GetMP4MediaFragment	1	
GetTSFragment	1	
the section called “GetImages”	请求的最大图像数	

例如，如果配额为每秒 500 个片段媒体，则支持特定流的以下调用模式：

- 每秒 5 个请求，每个片段中GetClip有 100 个片段。
- 每秒 100 个请求，每个片段中GetClip有 5 个片段。
- 每秒 2 个请求，每个片段中GetClip有 100 个片段，每个片段每秒 3 个请求。GetMediaForFragmentList
- 每秒 400 个请求发送到GetMP4MediaFragment，每秒 100 个请求GetTSFragment。

这些配额对每个直播可以支持的 HLS 和 MPEG-DASH 会话数量有重要影响。媒体播放器在给定时间可以使用的 HLS 和 DASH 会话数量没有限制。因此，播放应用程序不要允许同时使用太多会话，这一点很重要。以下两个示例描述了如何确定可以支持的并发播放会话数量：

示例 1：直播

在直播场景中，HLS 的持续时间为 1 秒的片段、音频和视频轨道，MaxMediaPlaylistFragmentResults 设置为 5，媒体播放器通常 GetHLSMediaPlaylist 每秒进行两次调用。一个调用是为了获取最新的视频元数据，另一个是为了获取相应的音频元数据。这两个调用各消耗五个片段元数据配额点。它 GetMP4MediaFragment 每秒还会拨打两个电话：一个呼叫最新的视频，另一个呼叫相应的音频。每次调用都会消耗一个片段媒体令牌，因此总共消耗两个令牌。

在这种情况下，最多可以支持 250 个并发播放会话。在 250 个会话中，此场景每秒消耗 2,500 个片段元数据配额点（远低于 10,000 个配额），每秒消耗 500 个片段媒体配额点。

示例 2：按需播放

在以往事件的点播播放场景中，MPEG-DASH（音频和视频轨道，MaxManifestFragmentResults 设置为 1,000），媒体播放器通常在会话开始时调用 GetDASHManifest 一次（消耗 1,000 个片段元数据配额点），并以每秒 5 次（消耗 5 个片段媒体配额点）的 GetMP4MediaFragment 速度调用，直到所有片段都加载完毕。在这种情况下，每秒最多可以启动 10 个新会话（正好是每秒 10,000 个片段元数据的配额），最多可以有 100 个会话以每秒 5 的速度主动加载片段媒体（正好是每秒 500 个片段媒体的配额）。

您可以使

用 ArchivedFragmentsConsumed.Metadata 和 ArchivedFragmentsConsumed.Media 分别监控片段元数据和片段媒体配额点的使用情况。有关监控的信息，请参见 [监控](#)。

片段元数据配额

以下服务配额适用于向 Kinesis 视频流中的片段添加片段元数据：

- 您最多可在片段前附加 10 个元数据项目。
- 片段元数据名称的最大长度可为 128 个字节。
- 片段元数据值的最大长度可为 256 个字节。
- 片段元数据名称不能以字符串“AWS”开头。如果添加此类元数据项目，则 PIC 中的 putFragmentMetadata 方法将返回 STATUS_INVALID_METADATA_NAME 错误（错误代码 0x52000077）。然后，您的应用程序可以忽略该错误（PIC 不添加元数据项目）或响应该错误。

直播标签

这些元数据键值对适用于整个 Kinesis Video Streams 资源，而不是 Kinesis 视频流中包含的单个片段。

每个 Kinesis 视频流最多支持 50 个标签。

[the section called “TagStream”](#)有关直播标签键和值的限制，请参阅。

对 Kinesis Video Streams 进行故障排除

使用以下信息来解决亚马逊 Kinesis Video Streams 遇到的常见问题。

主题

- [排查一般问题](#)
- [排查 API 问题](#)
- [HLS 问题疑难解答](#)
- [解决 Java 问题](#)
- [对制作器库问题进行故障排除](#)
- [对流解析器库问题进行故障排除](#)

排查一般问题

本节介绍您在使用 Kinesis Video Streams 时可能遇到的一般问题。

问题

- [延迟太高](#)

延迟太高

延迟可能是由发送到 Kinesis Video Streams 服务的片段持续时间造成的。降低创建者与服务器间延迟的一种方法是配置媒体管道，以缩短片段持续时间。

要减少每个片段中发送的帧数，请减少以下值 `kinesis_video_gstreamer_sample_app.cpp`：

```
g_object_set(G_OBJECT (data.encoder), "bframes", 0, "key-int-max", 45, "bitrate", 512,
NULL);
```

Note

由于视频渲染的内部实现，Mozilla Firefox 浏览器中的延迟较高。

排查 API 问题

本节介绍您在使用 Kinesis Video Streams 时可能遇到的 API 问题。

问题

- [错误：“Unknown options”](#)
- [错误：“Unable to determine service/operation name to be authorized \(无法确定要授权的服务/操作名称 \)”](#)
- [错误：“Failed to put a frame in the stream \(无法将帧放入流 \)”](#)
- [错误：“服务在收到最终版本之前关闭 AckEvent 了连接”](#)
- [错误：“STATUS_STORE_OUT_OF_MEMORY”](#)

错误：“Unknown options”

下列错误可导致 GetMedia 和 GetMediaForFragmentList 失败：

```
Unknown options: <filename>.mkv
```

如果您配置的 output 类型为 `json`，AWS CLI 则会出现此错误。AWS CLI 使用默认输出类型 (`none`) 重新配置。[有关配置的信息 AWS CLI，请参阅《AWS CLI 命令参考》中的 `configure`。](#)

错误：“Unable to determine service/operation name to be authorized (无法确定要授权的服务/操作名称)”

下列错误可导致 GetMedia 失败：

```
Unable to determine service/operation name to be authorized
```

如果未正确指定终端节点，则可能会发生此错误。获取终端节点时，请务必在调用中包含以下参数，具体取决于要 GetDataEndpoint 调用的 API：

```
--api-name GET_MEDIA
--api-name PUT_MEDIA
--api-name GET_MEDIA_FOR_FRAGMENT_LIST
--api-name LIST_FRAGMENTS
```

错误：“Failed to put a frame in the stream (无法将帧放入流)”

下列错误可导致 PutMedia 失败：

```
Failed to put a frame in the stream
```

如果连接或权限不适用于服务，则可能会发生此错误。在中运行以下命令 AWS CLI，并验证是否可以检索直播信息：

```
aws kinesismedia describe-stream --stream-name StreamName --endpoint https://  
ServiceEndpoint.kinesisvideo.region.amazonaws.com
```

如果呼叫失败，请参阅[AWS CLI 故障排除](#)以了解更多信息。

错误：“服务在收到最终版本之前关闭 AckEvent 了连接”

下列错误可导致 PutMedia 失败：

```
com.amazonaws.SdkClientException: Service closed connection before final AckEvent was  
received
```

如果未正确实施 PushbackInputStream，则可能会发生此错误。验证unread()方法的实现是否正确。

错误：“STATUS_STORE_OUT_OF_MEMORY”

下列错误可导致 PutMedia 失败：

```
The content store is out of memory.
```

当内容存储没有分配到足够的大小时，会发生此错误。要增加内容存储的大小，请增加 StorageInfo.storageSize 的值。有关更多信息，请参阅 [StorageInfo](#)。

HLS 问题疑难解答

如果您的视频流无法正确播放，请参阅[the section called “HLS 问题疑难解答”](#)。

解决 Java 问题

本节介绍如何解决在使用 Kinesis Video Streams 时遇到的常见 Java 问题。

问题

- [启用 Java 日志](#)

启用 Java 日志

要解决有关 Java 示例和库的问题，启用和检查调试日志会很有帮助。要启用调试日志，请执行以下操作：

1. 在 dependencies 节点中，将 log4j 添加到 pom.xml 文件：

```
<dependency>
  <groupId>log4j</groupId>
  <artifactId>log4j</artifactId>
  <version>1.2.17</version>
</dependency>
```

2. 在 target/classes 目录中，创建一个名为 log4j.properties 的文件，该文件包含以下内容：

```
# Root logger option
log4j.rootLogger=DEBUG, stdout

# Redirect log messages to console
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.Target=System.out
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:
%L - %m%n

log4j.logger.org.apache.http.wire=DEBUG
```

之后，调试日志打印至 IDE 控制台。

对制作器库问题进行故障排除

此部分介绍了在采用 [创建者库](#) 时可能会遇到的问题。

问题

- [无法编译创建者开发工具包](#)

- [视频流未显示在控制台中](#)
- [在采用 GStreamer 演示应用程序流式处理数据时，出现错误：“Security token included in the request is invalid”](#)
- [错误：“Failed to submit frame to Kinesis Video client”\(无法将帧提交到 Kinesis 视频客户端\)](#)
- [GStreamer 应用程序停止运行，OS X 上显示消息 "streaming stopped, reason not-negotiated"](#)
- [当在 Raspberry Pi 上的 GStreamer 演示应用程序中创建 Kinesis 视频客户端时，出现错误：“Failed to allocate heap”](#)
- [当在 Raspberry Pi 上运行 GStreamer 演示应用程序时，出现错误：“Illegal Instruction”](#)
- [摄像机在 Raspberry Pi 上加载失败](#)
- [在 macOS High Sierra 上未找到摄像机](#)
- [在 macOS High Sierra 上编译时，找不到 jni.h 文件](#)
- [在运行 GStreamer 演示应用程序时出现 Curl 错误](#)
- [Raspberry Pi 上运行时的时间戳/范围断言](#)
- [Raspberry Pi 上的 gst_value_set_fraction_range_full 断言](#)
- [Android 上的 STATUS_MKV_INVALID_ANNEXB_NALU_IN_FRAME_DATA \(0x3200000d\) 错误](#)
- [已达到最大片段持续时间错误](#)
- [使用 IoT 授权时出现“Invalid thing name passed \(传递的事物名称无效\)”错误](#)

无法编译创建者开发工具包

验证所需的库是否在您的路径中。要验证这一点，请使用以下命令：

```
$ env | grep LD_LIBRARY_PATH
LD_LIBRARY_PATH=/home/local/awslabs/amazon-kinesis-video-streams-producer-sdk-cpp/
kinesis-video-native-build/downloads/local/lib
```

视频流未显示在控制台中

要在控制台中显示视频流，它必须使用 H.264 以 AvCC 格式编码。如果未显示您的流，请验证以下内容：

- 如果原始流使用 Annex-B 格式，则您的 [NAL 适配标志](#) 设置为 NAL_ADAPTATION_ANNEXB_NALS | NAL_ADAPTATION_ANNEXB_CPD_NALS。这是 StreamDefinition 构造函数中的默认值。

- 您是否正确提供了编解码器私有数据。对于 H.264，这是序列参数集 (SPS) 和图片参数集 (PPS)。根据您的媒体源，此数据可从媒体源中单独检索或编码到帧中。

许多基本流采用以下格式，其中 Ab 是 Annex-B 启动代码 (001 或 0001)：

```
Ab(Sps)Ab(Pps)Ab(I-frame)Ab(P/B-frame) Ab(P/B-frame)... Ab(Sps)Ab(Pps)Ab(I-frame)Ab(P/B-frame) Ab(P/B-frame)
```

如果 H.264 以 SPS 和 PPS 的形式出现在直播中，CPD (编解码器私有数据) 可以适应 aVCC 格式。除非媒体管道单独给出 CPD，否则应用程序可以通过查找第一个 Idr 帧 (其中应包含 SPS 和 PPS) 从帧中提取 CPD，提取两个 NALU (将是 Ab (Sps) Ab (Pps)) 并将其设置在 CPD 中，然后将其设置在 CPD 中。StreamDefinition

在采用 GStreamer 演示应用程序流式处理数据时，出现错误："Security token included in the request is invalid"

如果发生此错误，则您的凭证存在问题。请验证以下内容：

- 如果您使用的是临时凭证，则您必须指定一个会话令牌。
- 请验证您的临时凭证没有过期。
- 请验证您已设置了适当的权限。
- 在 macOS 上，请验证您没有在 Keychain 中缓存凭证。

错误："Failed to submit frame to Kinesis Video client"(无法将帧提交到 Kinesis 视频客户端)

如果发生此错误，则表明源流中的时间戳设置不正确。尝试以下操作：

- 使用最新的开发工具包示例，它可能包含能够修复您的问题的更新。
- 将高质量视频流设置为更高的比特率，如果摄像机支持，则修复源流中的任何抖动。

GStreamer 应用程序停止运行，OS X 上显示消息 "streaming stopped, reason not-negotiated"

OS X 上的流式处理可能停止，并显示以下消息：

```
Debugging information: gstbasesrc.c(2939): void gst_base_src_loop(GstPad *) (): /
GstPipeline:test-pipeline/GstAutoVideoSrc:source/GstAVFVideoSrc:source-actual-src-
avfvide:
streaming stopped, reason not-negotiated (-4)
```

一种可能的解决方法是从`gst_caps_new_simple`调用中删除帧速率参数`kinesis_video_gstreamer_sample_app.cpp`：

```
GstCaps *h264_caps = gst_caps_new_simple("video/x-h264",
                                         "profile", G_TYPE_STRING, "baseline",
                                         "stream-format", G_TYPE_STRING, "avc",
                                         "alignment", G_TYPE_STRING, "au",
                                         "width", GST_TYPE_INT_RANGE, 320, 1920,
                                         "height", GST_TYPE_INT_RANGE, 240, 1080,
                                         "framerate", GST_TYPE_FRACTION_RANGE, 0,
                                         1, 30, 1,
                                         NULL);
```

当在 Raspberry Pi 上的 GStreamer 演示应用程序中创建 Kinesis 视频客户端时，出现错误：“Failed to allocate heap”

GStreamer 示例应用程序尝试分配 512 MB 的 RAM，这可能不适用于您的系统。您可以通过降低 `KinesisVideoProducer.cpp` 中的以下值来减少此分配：

```
device_info.storageInfo.storageSize = 512 * 1024 * 1024;
```

当在 Raspberry Pi 上运行 GStreamer 演示应用程序时，出现错误：“Illegal Instruction”

如果您在运行 GStreamer 演示时遇到以下错误，请确认您已为正确的设备版本编译应用程序。（例如，当您在 Raspberry Pi 2 上运行时，请确认你没有针对 Raspberry Pi 3 进行编译。）

```
INFO - Initializing curl.
Illegal instruction
```

摄像机在 Raspberry Pi 上加载失败

要检查摄像机是否已加载，请运行以下命令：

```
$ ls /dev/video*
```

如果未找到，请运行以下命令：

```
$ vcgencmd get_camera
```

该输出值应该类似于以下内容：

```
supported=1 detected=1
```

如果驱动程序无法检测到摄像机，请执行以下操作：

1. 检查物理摄像机相机设置并验证其是否已正确连接。
2. 运行以下命令以升级固件：

```
$ sudo rpi-update
```

3. 重启设备。
4. 运行以下命令以加载驱动程序：

```
$ sudo modprobe bcm2835-v4l2
```

5. 验证是否已检测到摄像机：

```
$ ls /dev/video*
```

在 macOS High Sierra 上未找到摄像机

在 macOS High Sierra 上，如果有多个摄像机可用，则演示应用程序无法找到摄像机。

在 macOS High Sierra 上编译时，找不到 jni.h 文件

要纠正该错误，请将 Xcode 安装更新为最新的版本。

在运行 GStreamer 演示应用程序时出现 Curl 错误

要解决在运行 GStreamer 演示应用程序时出现的 Curl 错误，请将[该证书文件](#)复制到 `/etc/ssl/cert.pem` 中。

Raspberry Pi 上运行时的时间戳/范围断言

如果时间戳范围断言发生在运行时，请更新固件并重启设备：

```
$ sudo rpi-update
$ sudo reboot
```

Raspberry Pi 上的 `gst_value_set_fraction_range_full` 断言

如果 `uv4l` 服务正在运行，则将显示以下断言：

```
gst_util_fraction_compare (numerator_start, denominator_start, numerator_end,
denominator_end) < 0' failed
```

如果发生这种情况，请停止 `uv4l` 服务并重新启动应用程序。

Android 上的 `STATUS_MKV_INVALID_ANNEXB_NALU_IN_FRAME_DATA (0x3200000d)` 错误

如果 [NAL 适配标志](#) 对于媒体流不正确，则会出现以下错误：

```
putKinesisVideoFrame(): Failed to put a frame with status code 0x3200000d
```

如果发生此错误，请为您的媒体提供正确的 `.withNalAdaptationFlags` 标记（例如，`NAL_ADAPTATION_ANNEXB_CPD_NALS`）。在 [Android 创建者库](#) 的以下行中提供此标记：

[https://github.com/aws-labs/ aws-sdk-android-samples /blob/master/ /src/main/java/com/amazonaws/kinesisvideo/demoapp/fragmen AmazonKinesisVideoDemoApp t/.java #L169 StreamConfigurationFragment](https://github.com/aws-labs/aws-sdk-android-samples/blob/master/src/main/java/com/amazonaws/kinesisvideo/demoapp/fragmen AmazonKinesisVideoDemoApp t/.java #L169 StreamConfigurationFragment)

已达到最大片段持续时间错误

此错误在流中的媒体片段超出最大片段持续时间限制时发生。请参阅本 [the section called “媒体和存档媒体 API 服务配额”](#) 节中的最大片段持续时间限制。

要解决这个问题，请尝试以下操作：

- 如果您使用的是网络摄像机/USB 摄像机，请执行下列操作之一：
 - 如果您使用的是基于关键帧的分段，请将编码器设置为在 10 秒内提供关键帧。

- 如果您没有使用基于关键帧的分段，则在定义直播时[步骤 2：编写并检查代码](#)，请将最大片段持续时间限制设置为小于 10 秒的值。
- 如果您在 GStreamer 管道中使用软件编码器（如 x264），则可以在 10 秒钟内将该 key-int-max 属性设置为一个值。例如，设置为 60，key-int-max 将 fps 设置为 30，每隔 2 秒启用一次关键帧。
- 如果您使用的是 RPI 摄像机，请将关键帧间隔属性设置为小于 10 秒。
- 如果您使用的是 IP (RTSP) 摄像机，请将 GOP 大小设置为 60。

使用 IoT 授权时出现“Invalid thing name passed (传递的事物名称无效)”错误

要避免在使用物联网凭据进行授权时出现此错误 (HTTP Error 403: Response: {"message": "Invalid thing name passed"}), 请确保 stream-name (kvssink 元素的必填参数) 的值与的值相同 iot-thingname。有关更多信息，请参阅 [gStreamer 元素参数参考](#)。

对流解析器库问题进行故障排除

此部分介绍了在采用 [视频流解析器库](#) 时可能会遇到的问题。

问题

- [无法从流中访问单个帧](#)
- [片段解码错误](#)

无法从流中访问单个帧

要在使用者应用程序中访问来自流媒体源的单个帧，请验证您的直播是否包含正确的编解码器私有数据。有关流中数据格式的信息，请参阅 [数据模型](#)。

[要了解如何使用编解码器私有数据访问帧，请参阅 GitHub 网站上的以下测试文件：
KinesisVideoRendererExampleTest.java](#)

片段解码错误

如果您的片段未采用浏览器支持的 H.264 格式和级别进行正确编码，则在控制台中播放流时，您可能会看到以下错误：

```
Fragment Decoding Error
```

```
There was an error decoding the video data. Verify that the stream contains valid H.264 content
```

如果出现此错误，请确认以下几点：

- 帧的分辨率与编解码器私有数据中指定的分辨率匹配。
- 已编码的帧的 H.264 配置文件和级别与编解码器私有数据中指定的配置文件和级别匹配。
- 浏览器支持配置文件/级别组合。最新的浏览器支持所有配置文件和级别组合。
- 时间戳准确且采用正确顺序，并且未创建任何重复的时间戳。
- 您的应用程序使用 H.264 格式对帧数据进行编码。

Amazon Kinesis Video Streams 的文档历史记录

下表描述了自上次发行 Amazon Kinesis Video Streams 以来对文档所做的重要更改。

- 最新 API 版本：2017-11-29
- 最新文档更新：2023 年 6 月 27 日

更改	说明	日期
亚马逊 Kinesis 视频直播 Edge Agent 边缘到云的连接	新功能发布。有关更多信息，请参阅 边缘代理 ：	2023年6月27日
入门：向 Kinesis 视频流发送数据	将媒体数据从摄像机发送到 Kinesis 视频流的基本教程。有关更多信息，请参阅 向 Amazon Kinesis 视频流发送数据 ：	2019 年 1 月 21 日
用于与 SageMaker 集成的库模板	Kinesis Video Streams 的示例应用程序，SageMaker用于识别某些对象何时出现在亚马逊 Kinesis 视频流中。有关更多信息，请参阅 SageMaker ：	2018 年 11 月 19 日
流式处理元数据	您可以使用 Producer SDK 在 Kinesis 视频流中嵌入元数据。有关更多信息，请参阅 在 Kinesis Video Streams 中使用流式传输元数据 ：	2018 年 9 月 28 日
C++ 创建者开发工具包日志记录	您可以为 C++ 创建者开发工具包应用程序配置日志记录。有关更多信息，请参阅 在 C++ 制作者 SDK 中使用日志记录 ：	2018 年 7 月 18 日
HLS 视频流	现在，您可以使用 HTTP 直播观看 Kinesis 视频流。有关更	2018 年 7 月 13 日

更改	说明	日期
	多信息，请参阅 Kinesis Video Streams 回放 ：	
从 RTSP 源进行流式处理	在 Docker 容器中运行并从 RTSP 源流式传输视频的 Kinesis Video Streams 示例应用程序。有关更多信息，请参阅 RTSP 和 Docker ：	2018 年 6 月 20 日
C++ 创建者开发工具包 GStreamer 插件	演示如何生成 C++ 创建者库 ，以用作 GStreamer 目标。有关更多信息，请参阅 GStreamer ：	2018 年 15 月 6 日
创建者开发工具包回调参考文档	Kinesis 视频直播制作人库 使用的回调的参考文档。有关更多信息，请参阅 制作人 SDK 回调 ：	2018 年 12 月 6 日
系统要求	创建者设备和开发工具包的内存和存储要求文档。有关更多信息，请参阅 Kinesis Video Streams 系统要求 ：	2018 年 5 月 30 日
CloudTrail支持	用于监控 API CloudTrail 使用情况的文档。有关更多信息，请参阅 使用记录亚马逊 Kinesis Video Streams Video Streams API 调用 AWS CloudTrail ：	2018 年 5 月 24 日
创建者开发工具包结构参考文档	Kinesis 视频直播制作人库 所用结构的参考文档。有关更多信息，请参阅 制作人 SDK 结构 和 Kinesis 视频流结构 。	2018 年 5 月 7 日

更改	说明	日期
渲染器示例文档	Renderer 示例应用程序的文档，该文档显示了如何解码和显示 Kinesis 视频流中的帧。有关更多信息，请参阅 示例：解析和渲染 Kinesis Video Streams 片段 ：	2018 年 3 月 15 日
创建者开发工具包限制参考文档	有关 C++ 创建者库 中的操作限制的信息。有关更多信息，请参阅 制作人 SDK 限制 ：	2018 年 3 月 13 日
监控	有关使用亚马逊监控 Kinesis Video Streams 指标和 API 调用的信息 CloudWatch。AWS CloudTrail 有关更多信息，请参阅 监控 Amazon Kinesis Video Streams ：	2018 年 2 月 5 日
网络抽象层 (NAL) 适配标记参考	有关在使用流视频时设置 NAL 适配标记的信息。有关更多信息，请参阅 NAL 适配标志 ：	2018 年 1 月 15 日
Android 支持流视频	Kinesis Video Streams 现在支持来自安卓设备的流媒体视频。有关更多信息，请参阅 Android 创建者库 ：	2018 年 1 月 12 日
Kinesis 视频示例文档	Kinesis Video 示例应用程序的文档，其中显示了如何在应用程序 Kinesis 视频直播解析器库 中使用。有关更多信息，请参阅 KinesisVideoExample ：	2018 年 1 月 9 日
已发布的 Kinesis 视频流文档	这是 Amazon Kinesis Video Streams 开发人员指南的初始版本。	2017 年 11 月 29 日

API 参考

此节点下的部分包含 API 参考文档。使用左侧窗格中的目录转到不同的 API 参考部分。

操作

亚马逊 Kinesis 视频直播支持以下操作：

- [CreateSignalingChannel](#)
- [CreateStream](#)
- [DeleteEdgeConfiguration](#)
- [DeleteSignalingChannel](#)
- [DeleteStream](#)
- [DescribeEdgeConfiguration](#)
- [DescribeImageGenerationConfiguration](#)
- [DescribeMappedResourceConfiguration](#)
- [DescribeMediaStorageConfiguration](#)
- [DescribeNotificationConfiguration](#)
- [DescribeSignalingChannel](#)
- [DescribeStream](#)
- [GetDataEndpoint](#)
- [GetSignalingChannelEndpoint](#)
- [ListEdgeAgentConfigurations](#)
- [ListSignalingChannels](#)
- [ListStreams](#)
- [ListTagsForResource](#)
- [ListTagsForStream](#)
- [StartEdgeConfigurationUpdate](#)
- [TagResource](#)
- [TagStream](#)
- [UntagResource](#)

- [UntagStream](#)
- [UpdateDataRetention](#)
- [UpdateImageGenerationConfiguration](#)
- [UpdateMediaStorageConfiguration](#)
- [UpdateNotificationConfiguration](#)
- [UpdateSignalingChannel](#)
- [UpdateStream](#)

亚马逊 Kinesis Video Streams Media 支持以下操作：

- [GetMedia](#)
- [PutMedia](#)

亚马逊 Kinesis Video Streams 存档媒体支持以下操作：

- [GetClip](#)
- [GetDASHStreamingSessionURL](#)
- [GetHLSStreamingSessionURL](#)
- [GetImages](#)
- [GetMediaForFragmentList](#)
- [ListFragments](#)

亚马逊 Kinesis 视频信号频道支持以下操作：

- [GetIceServerConfig](#)
- [SendAlexaOfferToMaster](#)

亚马逊 Kinesis Video WebRTC Storage 支持以下操作：

- [JoinStorageSession](#)

Amazon Kinesis Video Streams

亚马逊 Kinesis 视频直播支持以下操作：

- [CreateSignalingChannel](#)
- [CreateStream](#)
- [DeleteEdgeConfiguration](#)
- [DeleteSignalingChannel](#)
- [DeleteStream](#)
- [DescribeEdgeConfiguration](#)
- [DescribeImageGenerationConfiguration](#)
- [DescribeMappedResourceConfiguration](#)
- [DescribeMediaStorageConfiguration](#)
- [DescribeNotificationConfiguration](#)
- [DescribeSignalingChannel](#)
- [DescribeStream](#)
- [GetDataEndpoint](#)
- [GetSignalingChannelEndpoint](#)
- [ListEdgeAgentConfigurations](#)
- [ListSignalingChannels](#)
- [ListStreams](#)
- [ListTagsForResource](#)
- [ListTagsForStream](#)
- [StartEdgeConfigurationUpdate](#)
- [TagResource](#)
- [TagStream](#)
- [UntagResource](#)
- [UntagStream](#)
- [UpdateDataRetention](#)
- [UpdateImageGenerationConfiguration](#)
- [UpdateMediaStorageConfiguration](#)
- [UpdateNotificationConfiguration](#)
- [UpdateSignalingChannel](#)
- [UpdateStream](#)

CreateSignalingChannel

服务: Amazon Kinesis Video Streams

创建信令信道。

CreateSignalingChannel 是一个异步操作。

请求语法

```
POST /createSignalingChannel HTTP/1.1
Content-type: application/json
```

```
{
  "ChannelName": "string",
  "ChannelType": "string",
  "SingleMasterConfiguration": {
    "MessageTtlSeconds": number
  },
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ]
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

ChannelName

您正在创建的信令通道的名称。此名称对于每个 AWS 账户 和 AWS 区域 必须唯一。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：[a-zA-Z0-9_.-]+

必需：是

ChannelType

您正在创建的信令通道的类型。目前唯一支持的通道类型为 SINGLE_MASTER。

类型：字符串

有效值：SINGLE_MASTER | FULL_MESH

必需：否

SingleMasterConfiguration

包含SINGLE_MASTER频道类型配置的结构。

类型：[SingleMasterConfiguration](#) 对象

必需：否

Tags

您要与此频道关联的一组标签（键值对）。

类型：[Tag](#) 对象数组

数组成员：最少 0 项。最多 50 项。

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "ChannelARN": "string"
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

ChannelARN

已创建频道的亚马逊资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

AccountChannelLimitExceededException

您已达到该区域AWS 账户中此活动的信令通道的最大限制。

HTTP 状态代码：400

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceInUseException

如果输入StreamARN或ChannelARN输入已映射到其他 Kinesis Video Stream 资源，或者提供的输入StreamARN或未ChannelARN处于“活动”状态，请尝试以下方法之一：`CLOUD_STORAGE_MODE`

1. 用于确定给定频道的直播映射到什么的 DescribeMediaStorageConfiguration API。
2. 用于确定给定直播映射到哪个频道的 DescribeMappedResourceConfiguration API。
3. DescribeStream或 DescribeSignalingChannel API，用于确定资源状态。

HTTP 状态代码：400

TagsPerResourceExceededLimitException

您已超出可以与资源关联的标签上限。一个 Kinesis 视频流最多可以支持 50 个标签。

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

CreateStream

服务: Amazon Kinesis Video Streams

创建新的 Kinesis 视频流。

创建新流时，Kinesis Video Streams 会为其分配一个版本号。更改流的元数据时，Kinesis Video Streams 会更新该版本。

CreateStream 是一个异步操作。

有关服务工作方式的信息，请参阅[工作方式](#)。

您必须有 KinesisVideo:CreateStream 操作的权限。

请求语法

```
POST /createStream HTTP/1.1
Content-type: application/json

{
  "DataRetentionInHours": number,
  "DeviceName": "string",
  "KmsKeyId": "string",
  "MediaType": "string",
  "StreamName": "string",
  "Tags": {
    "string" : "string"
  }
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

[DataRetentionInHours](#)

您希望在流中保留数据的小时数。Kinesis Video Streams 将数据保留在与流关联的数据存储中。

默认值为 0，表示流不保留数据。

当该DataRetentionInHours值为 0 时，使用者仍然可以使用保留在服务主机缓冲区中的碎片，该缓冲区的保留时间限制为 5 分钟，保留内存限制为 200 MB。当达到任一限制时，片段将从缓冲区中移除。

类型：整数

有效范围：最小值为 0。

必需：否

DeviceName

正在写入流的设备的名称。

Note

在当前的实现中，Kinesis Video Streams 没有使用这个名称。

类型：字符串

长度限制：最小长度为 1。最大长度为 128。

模式：[a-zA-Z0-9_.-]+

必需：否

KmsKeyId

你希望 Kinesis Video Streams 用来加密直播数据的 AWS Key Management Service (AWS KMS) 密钥的 ID。

如果未指定密钥 ID，则使用默认的 Kinesis 视频管理密钥 () AWS/kinesisvideo。

有关更多信息，请参阅[DescribeKey](#)。

类型：字符串

长度限制：最小长度为 0。最大长度为 2048。

模式：.+

必需：否

MediaType

直播的媒体类型。直播的使用者可以在处理直播时使用此信息。有关媒体类型的更多信息，请参阅[媒体类型](#)。如果您选择指定 MediaType，请参阅[命名要求](#)以获取指南。

有效值示例包括 “video/h264” 和 “video/h264 , audio/aac”。

此参数是可选的；默认值为 null（或在 JSON 中为空）。

类型：字符串

长度限制：最小长度为 1。最大长度为 128。

模式：`[\w\-\.\+]+/[\w\-\.\+]+(,[\w\-\.\+]+/[\w\-\.\+]+)*`

必需：否

StreamName

您正在创建的直播的名称。

直播名称是直播的标识符，并且对于每个账户和地区都必须是唯一的。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：是

Tags

要与指定直播关联的标签列表。每个标签都是一个键值对（该值是可选的）。

类型：字符串到字符串映射

映射条目：最多 50 项。

密钥长度限制：最小长度为 1。最大长度为 128。

密钥模式：`^[{\p{L}}{\p{Z}}{\p{N}}_.:/=+\-@]*$`

值长度限制：最小长度为 0。长度上限为 256。

价值模式：`[{\p{L}}{\p{Z}}{\p{N}}_.:/=+\-@]*`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "StreamARN": "string"
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

StreamARN

流的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccountStreamLimitExceededException

为该账户创建的直播数量过高。

HTTP 状态代码：400

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为您已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

DeviceStreamLimitExceededException

未实现。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

InvalidDeviceException

未实现。

HTTP 状态代码：400

ResourceInUseException

如果输入StreamARN或ChannelARN输入已映射到其他 Kinesis Video Stream 资源，或者提供的输入StreamARN或未ChannelARN处于“活动”状态，请尝试以下方法之

一：CLOUD_STORAGE_MODE

1. 用于确定给定频道的直播映射到什么的 DescribeMediaStorageConfiguration API。
2. 用于确定给定直播映射到哪个频道的 DescribeMappedResourceConfiguration API。
3. DescribeStream或 DescribeSignalingChannel API，用于确定资源状态。

HTTP 状态代码：400

TagsPerResourceExceededLimitException

您已超出可以与资源关联的标签上限。一个 Kinesis 视频流最多可以支持 50 个标签。

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)

- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DeleteEdgeConfiguration

服务: Amazon Kinesis Video Streams

一种异步 API，用于从边缘代理中删除直播的现有边缘配置以及相应的媒体。

当您调用此 API 时，同步状态将设置为 DELETING。删除过程开始，在此过程中，活动的边缘作业将停止，所有媒体将从边缘设备中删除。删除的时间会有所不同，具体取决于存储的媒体总量。如果删除过程失败，则同步状态将更改为 DELETE_FAILED。您将需要重试删除。

成功完成删除过程后，将无法再访问边缘配置。

Note

此 API 不在 AWS 非洲 (开普敦) 区域 af-south-1 中提供。

请求语法

```
POST /deleteEdgeConfiguration HTTP/1.1
Content-type: application/json

{
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

StreamARN

流的 Amazon 资源名称 (ARN)。指定 StreamName 或 StreamARN。

类型：字符串

长度限制：最小长度为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

要从中删除边缘配置的流的名称。指定StreamName或StreamARN。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为您已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

StreamEdgeConfigurationNotFoundException

当 Amazon Kinesis 视频流找不到您指定的直播边缘配置时，会呈现异常。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DeleteSignalingChannel

服务: Amazon Kinesis Video Streams

删除指定的信令信道。DeleteSignalingChannel是一个异步操作。如果您未指定频道的当前版本，则会删除最新版本。

请求语法

```
POST /deleteSignalingChannel HTTP/1.1
Content-type: application/json

{
  "ChannelARN": "string",
  "CurrentVersion": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

ChannelARN

您要删除的信令通道的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：是

CurrentVersion

您要删除的信令通道的当前版本。您可以通过调用DescribeSignalingChannel或ListSignalingChannels API 操作来获取当前版本。

类型：字符串

长度限制：最小长度为 0。最大长度为 64。

模式：`[a-zA-Z0-9]+`

必需：否

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为您已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceInUseException

如果输入 StreamARN 或 ChannelARN 输入已映射到其他 Kinesis Video Stream 资源，或者所提供的输入 StreamARN 或未 ChannelARN 处于“活动”状态，请尝试以下方法之

一：CLOUD_STORAGE_MODE

1. 用于确定给定频道的直播映射到什么的 DescribeMediaStorageConfiguration API。

2. 用于确定给定直播映射到哪个频道的 `DescribeMappedResourceConfiguration` API。
3. `DescribeStream`或 `DescribeSignalingChannel` API，用于确定资源状态。

HTTP 状态代码：400

ResourceNotFoundException

Amazon Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

VersionMismatchException

您指定的直播版本不是最新版本。要获取最新版本，请使用 [DescribeStream](#)API。

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DeleteStream

服务: Amazon Kinesis Video Streams

删除 Kinesis 视频流和该视频流中包含的数据。

此方法将流标记为删除，并使流中的数据立即无法访问。

为确保在删除直播之前拥有最新版本，您可以指定直播版本。Kinesis Video Streams 为每个直播分配一个版本。当你更新直播时，Kinesis Video Streams 会分配一个新的版本号。要获取最新的直播版本，请使用 DescribeStream API。

此操作需要 KinesisVideo:DeleteStream 操作权限。

请求语法

```
POST /deleteStream HTTP/1.1
Content-type: application/json

{
  "CurrentVersion": "string",
  "StreamARN": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

CurrentVersion

可选：您要删除的直播版本。

指定版本作为安全措施，以确保您删除的直播正确。要获取直播版本，请使用 DescribeStream API。

如果未指定，CreationTime则在删除直播之前仅选中。

类型：字符串

长度限制：最小长度为 0。最大长度为 64。

模式：`[a-zA-Z0-9]+`

必需：否

StreamARN

您要删除的直播的亚马逊资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：是

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

NotAuthorizedException

呼叫者无权执行此操作。

HTTP 状态代码：401

ResourceInUseException

如果输入StreamARN或ChannelARN输入已映射到其他 Kinesis Video Stream 资源，或者提供的输入StreamARN或未ChannelARN处于“活动”状态，请尝试以下方法之一：CLOUD_STORAGE_MODE

1. 用于确定给定频道的直播映射到什么的 DescribeMediaStorageConfiguration API。
2. 用于确定给定直播映射到哪个频道的 DescribeMappedResourceConfiguration API。
3. DescribeStream或 DescribeSignalingChannel API，用于确定资源状态。

HTTP 状态代码：400

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

VersionMismatchException

您指定的直播版本不是最新版本。要获取最新版本，请使用 [DescribeStreamAPI](#)。

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)

- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DescribeEdgeConfiguration

服务: Amazon Kinesis Video Streams

描述使用 `StartEdgeConfigurationUpdate` API 设置的直播边缘配置以及边缘代理录制器和上传者作业的最新状态。使用此 API 获取配置状态，以确定配置是否与 Edge Agent 同步。使用此 API 来评估边缘代理的运行状况。

Note

此 API 不在 AWS 非洲 (开普敦) 区域 `af-south-1` 中提供。

请求语法

```
POST /describeEdgeConfiguration HTTP/1.1
```

```
Content-type: application/json
```

```
{  
  "StreamARN": "string",  
  "StreamName": "string"  
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

StreamARN

流的 Amazon 资源名称 (ARN)。指定 `StreamName` 或 `StreamARN`。

类型：字符串

长度限制：最小长度为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

要更新其边缘配置的直播的名称。指定StreamName或StreamARN。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "CreationTime": number,
  "EdgeAgentStatus": {
    "LastRecorderStatus": {
      "JobStatusDetails": "string",
      "LastCollectedTime": number,
      "LastUpdatedTime": number,
      "RecorderStatus": "string"
    },
    "LastUploaderStatus": {
      "JobStatusDetails": "string",
      "LastCollectedTime": number,
      "LastUpdatedTime": number,
      "UploaderStatus": "string"
    }
  },
  "EdgeConfig": {
    "DeletionConfig": {
      "DeleteAfterUpload": boolean,
      "EdgeRetentionInHours": number,
      "LocalSizeConfig": {
        "MaxLocalMediaSizeInMB": number,
        "StrategyOnFullSize": "string"
      }
    }
  }
}
```

```

    },
    "HubDeviceArn": "string",
    "RecorderConfig": {
      "MediaSourceConfig": {
        "MediaUriSecretArn": "string",
        "MediaUriType": "string"
      },
      "ScheduleConfig": {
        "DurationInSeconds": number,
        "ScheduleExpression": "string"
      }
    },
    "UploaderConfig": {
      "ScheduleConfig": {
        "DurationInSeconds": number,
        "ScheduleExpression": "string"
      }
    }
  },
  "FailedStatusDetails": "string",
  "LastUpdatedTime": number,
  "StreamARN": "string",
  "StreamName": "string",
  "SyncStatus": "string"
}

```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

CreationTime

首次创建直播边缘配置的时间戳。

类型：时间戳

EdgeAgentStatus

一个对象，其中包含边缘代理的录制器和上传者作业的最新状态详细信息。使用此信息来确定边缘代理的当前运行状况。

类型：[EdgeAgentStatus](#) 对象

[EdgeConfig](#)

对直播边缘配置的描述，该配置将用于与 Edge Agent IoT Greengrass 组件同步。Edge Agent 组件将在您所在地的 IoT 中心设备设置上运行。

类型：[EdgeConfig](#) 对象

[FailedStatusDetails](#)

对生成的故障状态的描述。

类型：字符串

[LastUpdatedTime](#)

上次更新直播边缘配置的时间戳。

类型：时间戳

[StreamARN](#)

流的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

[StreamName](#)

从中更新 Edge 配置的流的名称。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

[SyncStatus](#)

边缘配置更新的最新状态。

类型：字符串

有效值：SYNCING | ACKNOWLEDGED | IN_SYNC | SYNC_FAILED | DELETING | DELETE_FAILED | DELETING_ACKNOWLEDGED

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceNotFoundException

Amazon Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

StreamEdgeConfigurationNotFoundExcpion

当 Amazon Kinesis 视频流找不到您指定的直播边缘配置时，会呈现异常。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DescribeImageGenerationConfiguration

服务: Amazon Kinesis Video Streams

获取给定 Kinesis 视频流的 `ImageGenerationConfiguration`

请求语法

```
POST /describeImageGenerationConfiguration HTTP/1.1
Content-type: application/json

{
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

StreamARN

要从中检索图像生成配置的 Kinesis 视频流的亚马逊资源名称 (ARN)。必须指定 `StreamName` 或 `StreamARN`。

类型：字符串

长度限制：最小长度为 1。长度上限为 1024。

模式：arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：否

StreamName

要从中检索图像生成配置的流的名称。必须指定 `StreamName` 或 `StreamARN`。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "ImageGenerationConfiguration": {
    "DestinationConfig": {
      "DestinationRegion": "string",
      "Uri": "string"
    },
    "Format": "string",
    "FormatConfig": {
      "string": "string"
    },
    "HeightPixels": number,
    "ImageSelectorType": "string",
    "SamplingInterval": number,
    "Status": "string",
    "WidthPixels": number
  }
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

[ImageGenerationConfiguration](#)

包含 Kinesis 视频流 (KVS) 图像交付所需信息的结构。如果此结构为空，则该配置将从流中删除。

类型：[ImageGenerationConfiguration](#) 对象

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为您已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版 SDK](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DescribeMappedResourceConfiguration

服务: Amazon Kinesis Video Streams

返回有关直播的最新信息。输入中streamARN应提供streamName或。

请求语法

```
POST /describeMappedResourceConfiguration HTTP/1.1
Content-type: application/json
```

```
{
  "MaxResults": number,
  "NextToken": "string",
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

MaxResults

响应中返回的最大结果数。

类型：整数

有效范围：固定值为 1。

必需：否

NextToken

在您的下一个请求中提供的令牌，以获得另一批结果。

类型：字符串

长度约束：最小长度为 0。最大长度为 512。

模式：`[a-zA-Z0-9+/=]*`

必需：否

StreamARN

流的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

流的名称。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "MappedResourceConfigurationList": [
    {
      "ARN": "string",
      "Type": "string"
    }
  ],
  "NextToken": "string"
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

[MappedResourceConfigurationList](#)

封装或包含媒体存储配置属性的结构。

类型：[MappedResourceConfigurationListItem](#) 对象数组

数组成员：最少 0 项。最多 1 项。

[NextToken](#)

NextToken 请求中用于获取下一组结果的令牌。

类型：字符串

长度约束：最小长度为 0。最大长度为 512。

模式：`[a-zA-Z0-9+/=]*`

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为您已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DescribeMediaStorageConfiguration

服务: Amazon Kinesis Video Streams

返回有关该频道的最新信息。在输入ChannelARN中指定ChannelName或。

请求语法

```
POST /describeMediaStorageConfiguration HTTP/1.1
Content-type: application/json

{
  "ChannelARN": "string",
  "ChannelName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

ChannelARN

该频道的亚马逊资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：否

ChannelName

通道的名称。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "MediaStorageConfiguration": {
    "Status": "string",
    "StreamARN": "string"
  }
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

[MediaStorageConfiguration](#)

一种封装或包含媒体存储配置属性的结构。

类型：[MediaStorageConfiguration](#) 对象

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceNotFoundException

Amazon Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DescribeNotificationConfiguration

服务: Amazon Kinesis Video Streams

获取给定 Kinesis 视频流的。NotificationConfiguration

请求语法

```
POST /describeNotificationConfiguration HTTP/1.1
Content-type: application/json
```

```
{
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

[StreamARN](#)

您要从中检索通知配置的 Kinesis 视频流的亚马逊资源名称 (ARN)。您必须指定或 StreamArn。StreamName

类型：字符串

长度限制：最小长度为 1。长度上限为 1024。

模式：arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：否

[StreamName](#)

要从中检索通知配置的流的名称。必须指定StreamName或StreamARN。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "NotificationConfiguration": {
    "DestinationConfig": {
      "Uri": "string"
    },
    "Status": "string"
  }
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

[NotificationConfiguration](#)

包含通知所需信息的结构。如果结构为空，则配置将从流中删除。

类型：[NotificationConfiguration](#) 对象

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DescribeSignalingChannel

服务: Amazon Kinesis Video Streams

返回有关信令通道的最新信息。您必须指定要描述的频道的名称或亚马逊资源名称 (ARN)。

请求语法

```
POST /describeSignalingChannel HTTP/1.1
Content-type: application/json

{
  "ChannelARN": "string",
  "ChannelName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

ChannelARN

您要描述的信令信道的 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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：否

ChannelName

您要描述的信令通道的名称。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "ChannelInfo": {
    "ChannelARN": "string",
    "ChannelName": "string",
    "ChannelStatus": "string",
    "ChannelType": "string",
    "CreationTime": number,
    "SingleMasterConfiguration": {
      "MessageTtlSeconds": number
    },
    "Version": "string"
  }
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

ChannelInfo

一种封装指定信令通道的元数据和属性的结构。

类型：[ChannelInfo](#) 对象

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版 SDK](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DescribeStream

服务: Amazon Kinesis Video Streams

返回有关指定直播的最新信息。必须指定StreamName或StreamARN。

请求语法

```
POST /describeStream HTTP/1.1
Content-type: application/json

{
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

StreamARN

流的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：否

StreamName

流的名称。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：[a-zA-Z0-9_.-]+

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "StreamInfo": {
    "CreationTime": number,
    "DataRetentionInHours": number,
    "DeviceName": "string",
    "KmsKeyId": "string",
    "MediaType": "string",
    "Status": "string",
    "StreamARN": "string",
    "StreamName": "string",
    "Version": "string"
  }
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

[StreamInfo](#)

描述直播的对象。

类型：[StreamInfo](#) 对象

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

NotAuthorizedException

呼叫者无权执行此操作。

HTTP 状态代码：401

ResourceNotFoundException

Amazon Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

GetDataEndpoint

服务: Amazon Kinesis Video Streams

获取用于读取或写入的指定流的端点。在应用程序中使用此端点从指定的流中读取（使用 `GetMedia` 或 `GetMediaForFragmentList` 操作）或向其写入（使用 `PutMedia` 操作）。

Note

返回的端点未附加 API 名称。客户端需要将 API 名称添加到返回的端点。

在请求中，通过 `StreamName` 或指定直播 `StreamARN`。

请求语法

```
POST /getDataEndpoint HTTP/1.1
Content-type: application/json

{
  "APIName": "string",
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

APIName

要获取终端节点的 API 操作的名称。

类型：字符串

有效值：PUT_MEDIA | GET_MEDIA | LIST_FRAGMENTS |
GET_MEDIA_FOR_FRAGMENT_LIST | GET_HLS_STREAMING_SESSION_URL |
GET_DASH_STREAMING_SESSION_URL | GET_CLIP | GET_IMAGES

必需：是

StreamARN

您想要获取终端节点的流的 Amazon 资源名称 (ARN)。您必须在请求StreamName中指定此参数或一个。

类型：字符串

长度限制：最小长度为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

您要获取其终端节点的直播的名称。您必须在请求StreamARN中指定此参数或一个。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "DataEndpoint": "string"
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

[DataEndpoint](#)

终端节点值。要从流中读取数据或向其写入数据，请在应用程序中指定此端点。

类型：字符串

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

NotAuthorizedException

呼叫者无权执行此操作。

HTTP 状态代码：401

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)

- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

GetSignalingChannelEndpoint

服务: Amazon Kinesis Video Streams

为指定的信令通道提供一个用于发送和接收消息的端点。此 API 使用 `SingleMasterChannelEndpointConfiguration` 输入参数，该参数由 `Protocols` 和 `Role` 属性组成。

`Protocols` 用于确定通信机制。例如，如果您指定 WSS 为协议，则此 API 会生成一个安全的 websocket 端点。如果您指定 HTTPS 为协议，则此 API 会生成一个 HTTPS 终端节点。

`Role` 决定消息传送权限。MASTER 角色会导致此 API 生成一个端点，客户端可以使用该端点与频道上的任何观众进行通信。VIEWER 角色会导致此 API 生成一个端点，客户端只能使用该端点与通信 MASTER。

请求语法

```
POST /getSignalingChannelEndpoint HTTP/1.1
Content-type: application/json

{
  "ChannelARN": "string",
  "SingleMasterChannelEndpointConfiguration": {
    "Protocols": [ "string" ],
    "Role": "string"
  }
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

[ChannelARN](#)

您要为其获取终端节点的信令通道的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：是

[SingleMasterChannelEndpointConfiguration](#)

包含SINGLE_MASTER频道类型端点配置的结构。

类型：[SingleMasterChannelEndpointConfiguration](#) 对象

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "ResourceEndpointList": [
    {
      "Protocol": "string",
      "ResourceEndpoint": "string"
    }
  ]
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

[ResourceEndpointList](#)

指定信令通道的端点列表。

类型：[ResourceEndpointListItem](#) 对象数组

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceInUseException

如果输入StreamARN或ChannelARN输入已映射到其他 Kinesis Video Stream 资源，或者提供的输入StreamARN或未ChannelARN处于“活动”状态，请尝试以下方法之一：`CLOUD_STORAGE_MODE`

1. 用于确定给定频道的直播映射到什么的 `DescribeMediaStorageConfiguration` API。
2. 用于确定给定直播映射到哪个频道的 `DescribeMappedResourceConfiguration` API。
3. `DescribeStream`或 `DescribeSignalingChannel` API，用于确定资源状态。

HTTP 状态代码：400

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)

- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ListEdgeAgentConfigurations

服务: Amazon Kinesis Video Streams

返回与指定 Edge Agent 关联的边缘配置数组。

在请求中，您必须指定边缘代理 HubDeviceArn。

Note

此 API 不在 AWS 非洲 (开普敦) 区域 af-south-1 中提供。

请求语法

```
POST /listEdgeAgentConfigurations HTTP/1.1
Content-type: application/json
```

```
{
  "HubDeviceArn": "string",
  "MaxResults": number,
  "NextToken": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

HubDeviceArn

边缘代理的“物联网 (IoT) 事物” 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:[a-z\d-]+:iot:[a-z0-9-]+:[0-9]+:thing/[a-zA-Z0-9_.-]+

必需：是

MaxResults

响应中要返回的最大边缘配置数。默认为 5。

类型：整数

有效范围：最小值为 1。最大值为 10。

必需：否

NextToken

如果您指定此参数，则当ListEdgeAgentConfigurations操作的结果被截断时，调用将在响应NextToken中返回。要获得另一批边缘配置，请在下次请求中提供此令牌。

类型：字符串

长度约束：最小长度为 0。最大长度为 512。

模式：`[a-zA-Z0-9+/=]*`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "EdgeConfigs": [
    {
      "CreationTime": number,
      "EdgeConfig": {
        "DeletionConfig": {
          "DeleteAfterUpload": boolean,
          "EdgeRetentionInHours": number,
          "LocalSizeConfig": {
            "MaxLocalMediaSizeInMB": number,
            "StrategyOnFullSize": "string"
          }
        },
        "HubDeviceArn": "string",
        "RecorderConfig": {
```

```

    "MediaSourceConfig": {
      "MediaUriSecretArn": "string",
      "MediaUriType": "string"
    },
    "ScheduleConfig": {
      "DurationInSeconds": number,
      "ScheduleExpression": "string"
    }
  },
  "UploaderConfig": {
    "ScheduleConfig": {
      "DurationInSeconds": number,
      "ScheduleExpression": "string"
    }
  },
  "FailedStatusDetails": "string",
  "LastUpdatedTime": number,
  "StreamARN": "string",
  "StreamName": "string",
  "SyncStatus": "string"
}
],
"NextToken": "string"
}

```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

[EdgeConfigs](#)

对单个流的边缘配置的描述。

类型：[ListEdgeAgentConfigurationsEdgeConfig](#) 对象数组

[NextToken](#)

如果响应被截断，则调用将返回带有给定令牌的此元素。要获取下一批边缘配置，请在下一个请求中使用此令牌。

类型：字符串

长度约束：最小长度为 0。最大长度为 512。

模式：`[a-zA-Z0-9+/=]*`

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为您已超过允许的客户端调用次数限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

NotAuthorizedException

呼叫者无权执行此操作。

HTTP 状态代码：401

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ListSignalingChannels

服务: Amazon Kinesis Video Streams

返回一个ChannelInfo对象数组。每个对象都描述了一个信令信道。要仅检索满足特定条件的频道，可以指定ChannelNameCondition。

请求语法

```
POST /listSignalingChannels HTTP/1.1
Content-type: application/json

{
  "ChannelNameCondition": {
    "ComparisonOperator": "string",
    "ComparisonValue": "string"
  },
  "MaxResults": number,
  "NextToken": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

ChannelNameCondition

可选：仅返回满足特定条件的频道。

类型：[ChannelNameCondition](#) 对象

必需：否

MaxResults

响应中要返回的最大频道数。默认值为 500。

类型：整数

有效范围：最小值为 1。最大值为 10000。

必需：否

[NextToken](#)

如果您指定此参数，则当ListSignalingChannels操作的结果被截断时，调用将在响应NextToken中返回。要获得另一批频道，请在下次请求中提供此令牌。

类型：字符串

长度约束：最小长度为 0。最大长度为 512。

模式：`[a-zA-Z0-9+/=]*`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "ChannelInfoList": [
    {
      "ChannelARN": "string",
      "ChannelName": "string",
      "ChannelStatus": "string",
      "ChannelType": "string",
      "CreationTime": number,
      "SingleMasterConfiguration": {
        "MessageTtlSeconds": number
      },
      "Version": "string"
    }
  ],
  "NextToken": "string"
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

ChannelInfoList

ChannelInfo 对象数组。

类型：[ChannelInfo](#) 对象数组

NextToken

如果响应被截断，则调用会返回带有令牌的此元素。要获取下一批直播，请在下一个请求中使用此令牌。

类型：字符串

长度约束：最小长度为 0。最大长度为 512。

模式：`[a-zA-Z0-9+/=]*`

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ListStreams

服务: Amazon Kinesis Video Streams

返回一个StreamInfo对象数组。每个对象都描述了一个流。要仅检索满足特定条件的直播，可以指定StreamNameCondition。

请求语法

```
POST /listStreams HTTP/1.1
Content-type: application/json

{
  "MaxResults": number,
  "NextToken": "string",
  "StreamNameCondition": {
    "ComparisonOperator": "string",
    "ComparisonValue": "string"
  }
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

MaxResults

响应中要返回的最大直播数量。默认值为 10,000。

类型：整数

有效范围：最小值为 1。最大值为 10000。

必需：否

NextToken

如果您指定此参数，则当ListStreams操作的结果被截断时，调用将在响应NextToken中返回。要获得另一批直播，请在下次请求中提供此令牌。

类型：字符串

长度约束：最小长度为 0。最大长度为 512。

模式：`[a-zA-Z0-9+/=]*`

必需：否

StreamNameCondition

可选：仅返回满足特定条件的直播。目前，您只能将直播名称的前缀指定为条件。

类型：[StreamNameCondition](#) 对象

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "NextToken": "string",
  "StreamInfoList": [
    {
      "CreationTime": number,
      "DataRetentionInHours": number,
      "DeviceName": "string",
      "KmsKeyId": "string",
      "MediaType": "string",
      "Status": "string",
      "StreamARN": "string",
      "StreamName": "string",
      "Version": "string"
    }
  ]
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

NextToken

如果响应被截断，则调用会返回带有令牌的此元素。要获取下一批直播，请在下一个请求中使用此令牌。

类型：字符串

长度约束：最小长度为 0。最大长度为 512。

模式：`[a-zA-Z0-9+/=]*`

StreamInfoList

StreamInfo 对象数组。

类型：[StreamInfo](#) 对象数组

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)

- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ListTagsForResource

服务: Amazon Kinesis Video Streams

返回与指定信令通道关联的标签列表。

请求语法

```
POST /ListTagsForResource HTTP/1.1
Content-type: application/json

{
  "NextToken": "string",
  "ResourceARN": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

[NextToken](#)

如果您指定此参数并且 ListTagsForResource 调用结果被截断，则响应中会包含一个令牌，您可以在下一个请求中使用该令牌来获取下一批标签。

类型：字符串

长度约束：最小长度为 0。最大长度为 512。

模式：`[a-zA-Z0-9+/=]*`

必需：否

[ResourceARN](#)

您要列出标签的信令通道的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：是

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "NextToken": "string",
  "Tags": {
    "string" : "string"
  }
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

[NextToken](#)

如果您指定此参数并且 `ListTagsForResource` 调用结果被截断，则响应中会包含一个令牌，您可以在下一个请求中使用该令牌来获取下一组标签。

类型：字符串

长度约束：最小长度为 0。最大长度为 512。

模式：`[a-zA-Z0-9+/=]*`

[Tags](#)

与指定信令通道关联的标签键和值的映射。

类型：字符串到字符串映射

映射条目：最多 50 项。

密钥长度限制：最小长度为 1。最大长度为 128。

密钥模式：`^([\p{L}\p{Z}\p{N}_.:/+\\-@]*)$`

值长度限制：最小长度为 0。长度上限为 256。

价值模式：`[\p{L}\p{Z}\p{N}_.:/+\\-@]*`

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为您已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceNotFoundException

Amazon Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)

- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ListTagsForStream

服务: Amazon Kinesis Video Streams

返回与指定直播关联的标签列表。

在请求中，您必须指定StreamName或StreamARN。

请求语法

```
POST /listTagsForStream HTTP/1.1
Content-type: application/json
```

```
{
  "NextToken": "string",
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

NextToken

如果您指定此参数并且ListTagsForStream调用结果被截断，则响应中会包含一个令牌，您可以在下一个请求中使用该令牌来获取下一批标签。

类型：字符串

长度约束：最小长度为 0。最大长度为 512。

模式：[a-zA-Z0-9+/=]*

必需：否

StreamARN

您要为其列出标签的直播的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

您要为其列出标签的直播名称。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "NextToken": "string",
  "Tags": {
    "string" : "string"
  }
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

NextToken

如果您指定此参数并且 ListTags 调用结果被截断，则响应中会包含一个令牌，您可以在下一个请求中使用该令牌来获取下一组标签。

类型：字符串

长度约束：最小长度为 0。最大长度为 512。

模式：`[a-zA-Z0-9+/=]*`

Tags

与指定流关联的标签键和值的映射。

类型：字符串到字符串映射

映射条目：最多 50 项。

密钥长度限制：最小长度为 1。最大长度为 128。

密钥模式：`^([\p{L}\p{Z}\p{N}_.:/=+\-@]*)$`

值长度限制：最小长度为 0。长度上限为 256。

价值模式：`[\p{L}\p{Z}\p{N}_.:/=+\-@]*`

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

InvalidResourceFormatException

的格式StreamARN无效。

HTTP 状态代码：400

NotAuthorizedException

呼叫者无权执行此操作。

HTTP 状态代码 : 401

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码 : 404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

StartEdgeConfigurationUpdate

服务: Amazon Kinesis Video Streams

一种异步 API，用于更新直播的现有边缘配置。Kinesis Video Stream 会将直播的边缘配置与在你所在地设置的物联网中心设备上运行的 Edge Agent IoT Greengrass 组件同步。同步的时间可能有所不同，具体取决于集线器设备的连接。SyncStatus将在确认边缘配置后进行更新，并与 Edge Agent 同步。

如果首次调用此 API，则将为该直播创建新的边缘配置，并将同步状态设置为SYNCING。在再次使用此 API 之前，您必须等待同步状态达到终端状态SYNC_FAILED，例如:IN_SYNC、或。如果您在同步过程中调用此 API，则ResourceInUseException会抛出。将重试直播的边缘配置和 Edge Agent 的连接，持续 15 分钟。15 分钟后，状态将变为SYNC_FAILED状态。

要将边缘配置从一台设备移动到另一台设备，请使用[DeleteEdgeConfiguration](#)删除当前的边缘配置。然后，您可以使用更新的中心设备 ARN StartEdgeConfigurationUpdate 进行调用。

Note

此 API 不在AWS非洲 (开普敦) 区域 af-south-1 中提供。

请求语法

```
POST /startEdgeConfigurationUpdate HTTP/1.1
Content-type: application/json
```

```
{
  "EdgeConfig": {
    "DeletionConfig": {
      "DeleteAfterUpload": boolean,
      "EdgeRetentionInHours": number,
      "LocalSizeConfig": {
        "MaxLocalMediaSizeInMB": number,
        "StrategyOnFullSize": "string"
      }
    },
    "HubDeviceArn": "string",
    "RecorderConfig": {
      "MediaSourceConfig": {
        "MediaUriSecretArn": "string",
        "MediaUriType": "string"
      }
    }
  }
}
```

```
    "ScheduleConfig": {
      "DurationInSeconds": number,
      "ScheduleExpression": "string"
    }
  },
  "UploaderConfig": {
    "ScheduleConfig": {
      "DurationInSeconds": number,
      "ScheduleExpression": "string"
    }
  }
},
"StreamARN": "string",
"StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

EdgeConfig

调用更新过程所需的边缘配置详细信息。

类型：[EdgeConfig](#) 对象

必需：是

StreamARN

流的 Amazon 资源名称 (ARN)。指定StreamName或StreamARN。

类型：字符串

长度限制：最小长度为 1。长度上限为 1024。

模式：arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：否

StreamName

要更新其边缘配置的直播的名称。指定StreamName或StreamARN。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "CreationTime": number,
  "EdgeConfig": {
    "DeletionConfig": {
      "DeleteAfterUpload": boolean,
      "EdgeRetentionInHours": number,
      "LocalSizeConfig": {
        "MaxLocalMediaSizeInMB": number,
        "StrategyOnFullSize": "string"
      }
    },
    "HubDeviceArn": "string",
    "RecorderConfig": {
      "MediaSourceConfig": {
        "MediaUriSecretArn": "string",
        "MediaUriType": "string"
      },
      "ScheduleConfig": {
        "DurationInSeconds": number,
        "ScheduleExpression": "string"
      }
    },
    "UploaderConfig": {
      "ScheduleConfig": {
        "DurationInSeconds": number,
        "ScheduleExpression": "string"
      }
    }
  }
}
```

```
    }  
  }  
},  
"FailedStatusDetails": "string",  
"LastUpdatedTime": number,  
"StreamARN": "string",  
"StreamName": "string",  
"SyncStatus": "string"  
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

[CreationTime](#)

首次创建直播边缘配置的时间戳。

类型：时间戳

[EdgeConfig](#)

对直播边缘配置的描述，该配置将用于与 Edge Agent IoT Greengrass 组件同步。Edge Agent 组件将在您所在地的 IoT 中心设备设置上运行。

类型：[EdgeConfig](#) 对象

[FailedStatusDetails](#)

对生成的故障状态的描述。

类型：字符串

[LastUpdatedTime](#)

上次更新直播边缘配置的时间戳。

类型：时间戳

[StreamARN](#)

流的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

StreamName

从中更新 Edge 配置的流名称。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

SyncStatus

直播边缘配置的当前同步状态。当您调用此 API 时，同步状态将设置为 SYNCING 状态。使用 DescribeEdgeConfiguration API 获取边缘配置的最新状态。

类型：字符串

有效值：SYNCING | ACKNOWLEDGED | IN_SYNC | SYNC_FAILED | DELETING | DELETE_FAILED | DELETING_ACKNOWLEDGED

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

NoDataRetentionException

流数据保留时间（以小时为单位）等于零。

HTTP 状态代码：400

ResourceInUseException

如果输入StreamARN或ChannelARN输入已映射到其他 Kinesis Video Stream 资源，或者提供的输入StreamARN或未ChannelARN处于“活动”状态，请尝试以下方法之一：

—：CLOUD_STORAGE_MODE

1. 用于确定给定频道的直播映射到什么的 DescribeMediaStorageConfiguration API。
2. 用于确定给定直播映射到哪个频道的 DescribeMappedResourceConfiguration API。
3. DescribeStream或 DescribeSignalingChannel API，用于确定资源状态。

HTTP 状态代码：400

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

TagResource

服务: Amazon Kinesis Video Streams

向信令通道添加一个或多个标签。标签是一个键值对（该值是可选的），您可以定义并分配给AWS资源。如果您指定的标签已经存在，则标签值将替换为您在请求中指定的值。有关更多信息，请参阅[《成本管理用户指南》AWS Billing and Cost Management](#)和[《成本管理用户指南》中的使用成本分配标签](#)。

请求语法

```
POST /TagResource HTTP/1.1
Content-type: application/json
```

```
{
  "ResourceARN": "string",
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ]
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

ResourceARN

您要向其添加标签的信令通道的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：是

Tags

要与指定信令通道关联的标签列表。每个标签都是一个键-值对。

类型：[Tag](#) 对象数组

数组成员：最少 1 项。最多 50 项。

必需：是

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

TagsPerResourceExceededLimitException

您已超出可以与资源关联的标签上限。一个 Kinesis 视频流最多可以支持 50 个标签。

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版 SDK](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

TagStream

服务: Amazon Kinesis Video Streams

向直播添加一个或多个标签。标签是一个键值对（该值是可选的），您可以定义并分配给AWS资源。如果您指定的标签已经存在，则标签值将替换为您在请求中指定的值。有关更多信息，请参阅《[成本管理用户指南](#)》[AWS Billing and Cost Management](#)和《[成本管理用户指南](#)》中的[使用成本分配标签](#)。

您必须提供StreamName或StreamARN。

此操作需要 KinesisVideo:TagStream 操作权限。

一个 Kinesis 视频流最多可以支持 50 个标签。

请求语法

```
POST /tagStream HTTP/1.1
Content-type: application/json

{
  "StreamARN": "string",
  "StreamName": "string",
  "Tags": {
    "string" : "string"
  }
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

[StreamARN](#)

您要为其添加一个或多个标签的资源的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：否

StreamName

您要为其添加一个或多个标签的直播名称。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

Tags

要与指定直播关联的标签列表。每个标签都是一个键值对（该值是可选的）。

类型：字符串到字符串映射

映射条目：最多 50 项。

密钥长度限制：最小长度为 1。最大长度为 128。

密钥模式：`^[a-zA-Z0-9_./=-@]*$`

值长度限制：最小长度为 0。长度上限为 256。

价值模式：`[a-zA-Z0-9_./=-@]*`

必需：是

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

InvalidResourceFormatException

的格式StreamARN无效。

HTTP 状态代码：400

NotAuthorizedException

呼叫者无权执行此操作。

HTTP 状态代码：401

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

TagsPerResourceExceededLimitException

您已超出可以与资源关联的标签上限。一个 Kinesis 视频流最多可以支持 50 个标签。

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)

- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

UntagResource

服务: Amazon Kinesis Video Streams

从信令通道中移除一个或多个标签。在请求中，仅指定一个或多个标签密钥；不要指定值。如果您指定的标签密钥不存在，则会将其忽略。

请求语法

```
POST /UntagResource HTTP/1.1
Content-type: application/json

{
  "ResourceARN": "string",
  "TagKeyList": [ "string" ]
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

ResourceARN

您要从中移除标签的信令通道的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：是

TagKeyList

要移除的标签的密钥列表。

类型：字符串数组

数组成员：最少 1 项。最多 50 项。

长度限制：最小长度为 0。最大长度为 128。

模式：`^([\p{L}\p{Z}\p{N}_.:/=+\-@]*)$`

必需：是

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

UntagStream

服务: Amazon Kinesis Video Streams

从直播中移除一个或多个标签。在请求中，仅指定一个或多个标签密钥；不要指定值。如果您指定的标签密钥不存在，则会将其忽略。

在请求中，您必须提供StreamName或StreamARN。

请求语法

```
POST /untagStream HTTP/1.1
Content-type: application/json

{
  "StreamARN": "string",
  "StreamName": "string",
  "TagKeyList": [ "string" ]
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

StreamARN

您要从移除标签的直播的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：否

StreamName

要从移除标签的直播名称。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

[TagKeyList](#)

要移除的标签的密钥列表。

类型：字符串数组

数组成员：最少 1 项。最多 50 项。

长度限制：最小长度为 0。最大长度为 128。

模式：`^([\p{L}\p{Z}\p{N}_.:/=+\-@]*)$`

必需：是

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

InvalidResourceFormatException

的格式StreamARN无效。

HTTP 状态代码：400

NotAuthorizedException

呼叫者无权执行此操作。

HTTP 状态代码：401

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

UpdateDataRetention

服务: Amazon Kinesis Video Streams

按您指定的值增加或减少数据流的数据保留期。要指明您是要延长还是缩短数据保留期，请在请求正文中指定 `Operation` 参数。在请求中，您必须指定 `StreamName` 或 `StreamARN`。

此操作需要 `KinesisVideo:UpdateDataRetention` 操作权限。

更改数据保留期会对流中的数据产生如下影响：

- 如果延长了数据保留期，则现有数据将在新的保留期内保留。例如，如果数据保留期从一小时延长到七小时，则所有现有数据将保留七个小时。
- 如果缩短了数据保留期，则现有数据将在新的保留期内保留。例如，如果数据保留期从七小时缩短到一小时，则所有现有数据将保留一小时，并且任何超过一小时的数据都将立即删除。

请求语法

```
POST /updateDataRetention HTTP/1.1
Content-type: application/json

{
  "CurrentVersion": "string",
  "DataRetentionChangeInHours": number,
  "Operation": "string",
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

CurrentVersion

您要更改其保留期的直播版本。要获取版本，请调用 `DescribeStream` 或 `ListStreams` API。

类型：字符串

长度限制：最小长度为 0。最大长度为 64。

模式：`[a-zA-Z0-9]+`

必需：是

DataRetentionChangeInHours

调整当前留存时间所依据的小时数。您指定的值将与当前值相加或减去，具体视情况而定。`operation`

数据保留的最小值为 0，最大值为 87600（十年）。

类型：整数

有效范围：最小值为 1。

必需：是

Operation

表示您是要延长还是缩短保留期。

类型：字符串

有效值：`INCREASE_DATA_RETENTION` | `DECREASE_DATA_RETENTION`

必需：是

StreamARN

您要更改其保留期的直播的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

您要更改其保留期的直播名称。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

NotAuthorizedException

呼叫者无权执行此操作。

HTTP 状态代码：401

ResourceInUseException

如果输入StreamARN或ChannelARN输入已映射到其他 Kinesis Video Stream 资源，或者提供的输入StreamARN或未ChannelARN处于“活动”状态，请尝试以下方法之一：`CLOUD_STORAGE_MODE`

1. 用于确定给定频道的直播映射到什么的 DescribeMediaStorageConfiguration API。
2. 用于确定给定直播映射到哪个频道的 DescribeMappedResourceConfiguration API。
3. DescribeStream 或 DescribeSignalingChannel API，用于确定资源状态。

HTTP 状态代码：400

ResourceNotFoundException

Amazon Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

VersionMismatchException

您指定的直播版本不是最新版本。要获取最新版本，请使用 [DescribeStreamAPI](#)。

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

UpdateImageGenerationConfiguration

服务: Amazon Kinesis Video Streams

更新StreamInfo和ImageProcessingConfiguration字段。

请求语法

```
POST /updateImageGenerationConfiguration HTTP/1.1
Content-type: application/json
```

```
{
  "ImageGenerationConfiguration": {
    "DestinationConfig": {
      "DestinationRegion": "string",
      "Uri": "string"
    },
    "Format": "string",
    "FormatConfig": {
      "string": "string"
    },
    "HeightPixels": number,
    "ImageSelectorType": "string",
    "SamplingInterval": number,
    "Status": "string",
    "WidthPixels": number
  },
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

ImageGenerationConfiguration

包含 KVS 图像交付所需信息的结构。如果结构为空，则配置将从流中删除。

类型：[ImageGenerationConfiguration](#) 对象

必需：否

StreamARN

您要从中更新图像生成配置的 Kinesis 视频流的亚马逊资源名称 (ARN)。必须指定StreamName或StreamARN。

类型：字符串

长度限制：最小长度为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

用于更新图像生成配置的流的名称。必须指定StreamName或StreamARN。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

NoDataRetentionException

流数据保留时间（以小时为单位）等于零。

HTTP 状态代码：400

ResourceInUseException

如果输入StreamARN或ChannelARN输入已映射到其他 Kinesis Video Stream 资源，或者提供的输入StreamARN或未ChannelARN处于“活动”状态，请尝试以下方法之一：CLOUD_STORAGE_MODE

1. 用于确定给定频道的直播映射到什么的 DescribeMediaStorageConfiguration API。
2. 用于确定给定直播映射到哪个频道的 DescribeMappedResourceConfiguration API。
3. DescribeStream或 DescribeSignalingChannel API，用于确定资源状态。

HTTP 状态代码：400

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

UpdateMediaStorageConfiguration

服务: Amazon Kinesis Video Streams

将 a 关联 SignalingChannel 到直播以存储媒体。您可以指定两种信令模式：

- 如果启用 StorageStatus 用，则数据将存储在 StreamARN 提供的中。为了让 WebRTC Ingestion 正常运行，直播必须启用数据保留。
- 如果禁用，StorageStatus 则不会存储任何数据，也不需要该 StreamARN 参数。

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。

请求语法

```
POST /updateMediaStorageConfiguration HTTP/1.1
```

```
Content-type: application/json
```

```
{
  "ChannelARN": "string",
  "MediaStorageConfiguration": {
    "Status": "string",
    "StreamARN": "string"
  }
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

ChannelARN

该频道的亚马逊资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：是

MediaStorageConfiguration

一种封装或包含媒体存储配置属性的结构。

类型：[MediaStorageConfiguration](#) 对象

必需：是

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

NoDataRetentionException

流数据保留时间（以小时为单位）等于零。

HTTP 状态代码：400

ResourceInUseException

如果输入StreamARN或ChannelARN输入已映射到其他 Kinesis Video Stream 资源，或者所提供的输入StreamARN或未ChannelARN处于“活动”状态，请尝试以下方法之一：`CLOUD_STORAGE_MODE`

1. 用于确定给定频道的直播映射到什么的 `DescribeMediaStorageConfiguration` API。
2. 用于确定给定直播映射到哪个频道的 `DescribeMappedResourceConfiguration` API。
3. `DescribeStream`或 `DescribeSignalingChannel` API，用于确定资源状态。

HTTP 状态代码：400

ResourceNotFoundException

Amazon Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

UpdateNotificationConfiguration

服务: Amazon Kinesis Video Streams

更新直播的通知信息。

请求语法

```
POST /updateNotificationConfiguration HTTP/1.1
Content-type: application/json

{
  "NotificationConfiguration": {
    "DestinationConfig": {
      "Uri": "string"
    },
    "Status": "string"
  },
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

[NotificationConfiguration](#)

包含通知所需信息的结构。如果结构为空，则配置将从流中删除。

类型：[NotificationConfiguration](#) 对象

必需：否

[StreamARN](#)

您要从其中更新通知配置的 Kinesis 视频流的亚马逊资源名称 (ARN)。必须指定 `StreamName` 或 `StreamARN`。

类型：字符串

长度限制：最小长度为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

要从中更新通知配置的直播的名称。必须指定StreamName或StreamARN。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

NoDataRetentionException

流数据保留时间（以小时为单位）等于零。

HTTP 状态代码：400

ResourceInUseException

如果输入StreamARN或ChannelARN输入已映射到其他 Kinesis Video Stream 资源，或者所提供的输入StreamARN或未ChannelARN处于“活动”状态，请尝试以下方法之一：

—：CLOUD_STORAGE_MODE

1. 用于确定给定频道的直播映射到什么的 DescribeMediaStorageConfiguration API。
2. 用于确定给定直播映射到哪个频道的 DescribeMappedResourceConfiguration API。
3. DescribeStream或 DescribeSignalingChannel API，用于确定资源状态。

HTTP 状态代码：400

ResourceNotFoundException

Amazon Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)

- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

UpdateSignalingChannel

服务: Amazon Kinesis Video Streams

更新现有的信令信道。这是一个异步操作，需要一段时间才能完成。

如果该MessageTtlSeconds值已更新（增加或减少），则仅适用于更新后通过此渠道发送的新消息。根据之前的MessageTtlSeconds值，现有消息仍过期。

请求语法

```
POST /updateSignalingChannel HTTP/1.1
Content-type: application/json

{
  "ChannelARN": "string",
  "CurrentVersion": "string",
  "SingleMasterConfiguration": {
    "MessageTtlSeconds": number
  }
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

ChannelARN

您要更新的信令通道的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：是

CurrentVersion

您要更新的信令通道的当前版本。

类型：字符串

长度限制：最小长度为 0。最大长度为 64。

模式：`[a-zA-Z0-9]+`

必需：是

SingleMasterConfiguration

包含要更新的信令信道SINGLE_MASTER类型的配置的结构。

类型：[SingleMasterConfiguration](#) 对象

必需：否

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：401

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为您已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceInUseException

如果输入StreamARN或ChannelARN输入已映射到其他 Kinesis Video Stream 资源，或者提供的输入StreamARN或未ChannelARN处于“活动”状态，请尝试以下方法之一：`CLOUD_STORAGE_MODE`

1. 用于确定给定频道的直播映射到什么的 `DescribeMediaStorageConfiguration` API。
2. 用于确定给定直播映射到哪个频道的 `DescribeMappedResourceConfiguration` API。
3. `DescribeStream`或 `DescribeSignalingChannel` API，用于确定资源状态。

HTTP 状态代码：400

ResourceNotFoundException

亚马逊 Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

VersionMismatchException

您指定的直播版本不是最新版本。要获取最新版本，请使用 [DescribeStreamAPI](#)。

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)

- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

UpdateStream

服务: Amazon Kinesis Video Streams

更新直播元数据，例如设备名称和媒体类型。

您必须提供直播名称或直播的 Amazon 资源名称 (ARN)。

要确保在更新直播之前拥有最新版本，可以指定直播版本。Kinesis Video Streams 为每个直播分配一个版本。当你更新直播时，Kinesis Video Streams 会分配一个新的版本号。要获取最新的直播版本，请使用 DescribeStream API。

UpdateStream 是一种异步操作，需要一段时间才能完成。

请求语法

```
POST /updateStream HTTP/1.1
Content-type: application/json

{
  "CurrentVersion": "string",
  "DeviceName": "string",
  "MediaType": "string",
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

CurrentVersion

您要更新其元数据的直播的版本。

类型：字符串

长度限制：最小长度为 0。最大长度为 64。

模式：[a-zA-Z0-9]+

必需：是

DeviceName

正在写入流的设备的名称。

Note

在当前的实现中，Kinesis Video Streams 没有使用这个名称。

类型：字符串

长度限制：最小长度为 1。最大长度为 128。

模式：[a-zA-Z0-9_.-]+

必需：否

MediaType

直播的媒体类型。用于MediaType向直播的使用者指定流中包含的内容类型。有关媒体类型的更多信息，请参阅[媒体类型](#)。如果您选择指定MediaType，请参阅[命名要求](#)。

要在主机上播放视频，必须指定正确的视频类型。例如，如果流中的视频是 H.264，则指定video/h264为。MediaType

类型：字符串

长度限制：最小长度为 1。最大长度为 128。

模式：[\w\-\.\+]+/[\w\-\.\+]+(,[\w\-\.\+]+/[\w\-\.\+]+)*

必需：否

StreamARN

您要更新其元数据的直播的 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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：否

StreamName

要更新其元数据的直播的名称。

直播名称是直播的标识符，并且对于每个账户和地区都必须是唯一的。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：[a-zA-Z0-9_.-]+

必需：否

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

NotAuthorizedException

呼叫者无权执行此操作。

HTTP 状态代码：401

ResourceInUseException

如果输入StreamARN或ChannelARN输入已映射到其他 Kinesis Video Stream 资源，或者提供的输入StreamARN或未ChannelARN处于“活动”状态，请尝试以下方法之一：`CLOUD_STORAGE_MODE`

1. 用于确定给定频道的直播映射到什么的 `DescribeMediaStorageConfiguration` API。
2. 用于确定给定直播映射到哪个频道的 `DescribeMappedResourceConfiguration` API。
3. `DescribeStream`或 `DescribeSignalingChannel` API，用于确定资源状态。

HTTP 状态代码：400

ResourceNotFoundException

Amazon Kinesis Video Streams 找不到你指定的直播。

HTTP 状态代码：404

VersionMismatchException

您指定的直播版本不是最新版本。要获取最新版本，请使用 [DescribeStream](#)API。

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

Amazon Kinesis Video Streams

Amazon Kinesis Video Streams 支持的操作：

- [GetMedia](#)
- [PutMedia](#)

GetMedia

服务：Amazon Kinesis Video Streams Media

使用此 API 从 Kinesis 视频流中检索媒体内容。在请求中，您可以识别直播名称或流 Amazon 资源名称 (ARN) 以及起始区块。然后，Kinesis Video Streams 按片段编号的顺序返回区块流。

Note

您必须先调用 `GetDataEndpoint` API 才能获取终端节点。然后使用 `--endpoint-url` 参数将 `GetMedia` 请求发送到此端点。

当您在直播中放置媒体数据（片段）时，Kinesis Video Streams 会将每个传入的片段和相关元数据存储所谓的“块”中。有关更多信息，请参阅 [PutMedia](#)。GetMedia API 会从您在请求中指定的区块开始返回由这些区块组成的流。

使用 GetMedia API 时适用以下限制：

- 每个直播客户端每秒 GetMedia 最多可以呼叫五次。
- 在会话期间，Kinesis Video Streams 以高达每秒 25 兆字节（或每秒 200 兆比特）的速率发送媒体数据。GetMedia

Note

GetMedia HTTP 响应状态码将立即返回，但如果没有已摄取的片段可供播放，则读取 HTTP 响应有效负载将在 3 秒后超时。

Note

如果在调用 Kinesis Video Streams 媒体 API 后出现错误，则除了 HTTP 状态代码和响应正文外，还会包含以下信息：

- `x-amz-ErrorTypeHTTP` 标头 — 除了 HTTP 状态码提供的错误类型外，还包含更具体的错误类型。
- `x-amz-RequestIdHTTP` 标头 — 如果你想向报告问题 AWS，如果给出请求编号，支持团队可以更好地诊断问题。

HTTP 状态码和 `ErrorType` 标头都可用于对错误是否可重试以及在什么条件下做出编程决策，并提供有关客户端程序员可能需要采取哪些操作才能成功重试的信息。有关更多信息，请参阅本主题底部的错误部分以及[常见错误](#)。

请求语法

```
POST /getMedia HTTP/1.1
Content-type: application/json

{
  "StartSelector": {
    "AfterFragmentNumber": "string",
    "ContinuationToken": "string",
    "StartSelectorType": "string",
    "StartTimestamp": number
  },
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求体

请求接受采用 JSON 格式的以下数据。

[StartSelector](#)

标识要从指定流中获取的起始块。

类型：[StartSelector](#) 对象

必需：是

[StreamARN](#)

你想从哪里获取媒体内容的直播的 ARN。如果未指定 `streamARN`，则必须指定 `streamName`。

类型：字符串

长度限制：长度下限为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

您要从中获取媒体内容的 Kinesis 视频流名称。如果未指定 `streamName`，则必须指定 `streamARN`。

类型：字符串

长度约束：最小长度为 1。最大长度为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
Content-Type: ContentType
```

```
Payload
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

响应将返回以下 HTTP 标头。

ContentType

所请求媒体的内容类型。

长度限制：长度下限为 1。长度上限为 128。

模式：`^[a-zA-Z0-9_\. \-]+$`

响应将以下内容作为 HTTP 正文返回。

Payload

Kinesis Video Streams 返回的有效载荷是来自指定直播的一系列块。有关区块的更多信息，请参阅 [PutMedia](#)。Kinesis Video Streams 在通话中返回 GetMedia 的区块还包括以下其他 Matroska (MKV) 标签：

- `AWS_KINESISVIDEO_CONTINUATION_TOKEN` (UTF-8 字符串) -如果您的 GetMedia 呼叫终止，则可以在下一个请求中使用此延续令牌来获取最后一个请求终止的下一个区块。
- `AWS_KINESISVIDEO_MILLIS_BEHIND_NOW` (UTF-8 字符串) -客户端应用程序可以使用此标签值来确定响应中返回的区块与流中的最新区块相差多远。
- `AWS_KINESISVIDEO_FRAGMENT_NUMBER`-分块中返回的片段编号。
- `AWS_KINESISVIDEO_SERVER_TIMESTAMP`-片段的服务器时间戳。
- `AWS_KINESISVIDEO_PRODUCER_TIMESTAMP`-片段的制作者时间戳。

如果发生错误，将显示以下标签：

- `AWS_KINESISVIDEO_ERROR_CODE`-导致停止的错误的字符串描述。 GetMedia
- `AWS_KINESISVIDEO_ERROR_ID`：错误的整数代码。

错误代码如下：

- 3002-写入直播时出错
- 4000-未找到请求的片段
- 4500-直播的 KMS 密钥被拒绝访问
- 4501-Stream 的 KMS 密钥已禁用
- 4502-直播的 KMS 密钥存在验证错误
- 4503-直播中指定的 KMS 密钥不可用
- 4504-直播中指定的 KMS 密钥的使用无效
- 4505-直播中指定的 KMS 密钥的状态无效
- 4506-找不到直播中指定的 KMS 密钥
- 5000-内部错误

错误

有关所有操作返回的常见错误的信息，请参阅 [常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为您已超过允许的客户端调用次数限制。稍后再尝试拨打电话。

HTTP 状态代码：400

ConnectionLimitExceededException

Kinesis Video Streams 已限制该请求，因为您已超过允许的客户端连接限制。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

InvalidEndpointException

呼叫者使用了错误的端点将数据写入流。收到此类异常后，用户必须在APIName设置为的情况下调GetDataEndpoint用，PUT_MEDIA并使用响应中的端点来调用下一个PutMedia调用。

HTTP 状态代码：400

NotAuthorizedException

调用者无权对给定直播执行操作，或者令牌已过期。

HTTP 状态代码：401

ResourceNotFoundException

状态码：404，给定名称的直播不存在。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS SDK 中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)

- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

PutMedia

服务：Amazon Kinesis Video Streams Media

使用此 API 向 Kinesis 视频流发送媒体数据。

Note

您必须先调用 `GetDataEndpoint` API 才能获取终端节点。然后使用 [--endpoint-url 参数](#) 将 `PutMedia` 请求发送到此端点。

在请求中，您可以使用 HTTP 标头提供参数信息，例如直播名称、时间戳以及时间戳值是绝对值还是相对于制作人开始录制的时间。您使用请求正文发送媒体数据。Kinesis Video Streams 仅支持 Matroska (MKV) 容器格式，用于使用此 API 发送媒体数据。

您可以使用以下选项来使用此 API 发送数据：

- 实时发送媒体数据：例如，安全摄像机可以在生成帧时实时发送帧。这种方法最大限度地减少了视频录制和通过线路发送的数据之间的延迟。这被称为连续生产者。在这种情况下，消费者应用程序可以实时或在需要时读取流。
- 离线发送媒体数据（分批）：例如，机身摄像机可能会录制数小时的视频并将其存储在设备上。稍后，当您将摄像机连接到坞站端口时，摄像机可以启动 `PutMedia` 会话，将数据发送到 Kinesis 视频流。在这种情况下，延迟不是问题。

使用此 API 时，请注意以下注意事项：

- 您必须指定 `streamName` 或 `streamARN`，但不能同时指定两者。
- 为了能够在主机上或通过 HLS 播放媒体，每个片段的轨道 1 应包含 h.264 编码的视频，片段元数据中的 `CodeCid` 应为 “V_MPEG/ISO/AVC”，片段元数据应包含 AVCC 格式的 h.264 编解码器私有数据。或者，每个片段的轨道 2 应包含 AAC 编码的音频，片段元数据中的 `CodeCid` 应为 “A_AAC”，片段元数据应包含 AAC 编解码器私有数据。
- 您可能会发现，使用单个长时间运行的 `PutMedia` 会话并在有效负载中发送大量媒体数据片段会更容易。对于收到的每个片段，Kinesis Video Streams 都会发送一个或多个致谢。潜在的网络考虑因素可能会导致您无法在生成所有这些确认时获得这些确认。
- 您可以选择多个连续 `PutMedia` 会话，每个会话的片段更少，以确保您实时获得来自服务的所有确认。

Note

如果您在多个同步PutMedia会话中向同一个流发送数据，则媒体片段将在流中交错存放。你应该确保在你的应用场景中这是可以的。

使用 PutMedia API 时适用以下限制：

- 每个直播客户端每秒PutMedia最多可以呼叫五次。
- 一个客户端每秒最多可以发送五个片段。
- Kinesis Video Streams 在会话期间以高达 12.5 MB/秒或 100 Mbps 的速率读取媒体数据。PutMedia

请注意以下限制。在这些情况下，Kinesis Video Streams 会在响应中发送错误确认。

- 不允许使用时间码跨度超过允许的最大限制且包含超过 50 MB 数据的片段。
- 不允许包含超过三首曲目的片段。每个片段中的每个帧的轨道编号必须与片段标题中定义的轨道编号相同。此外，对于片段标题中定义的每个轨道，每个片段必须至少包含一个帧。
- 对于片段元数据中定义的每个轨道，每个片段必须至少包含一个帧。
- 片段中最早的帧时间戳必须晚于前一个片段中的最新帧时间戳。
- 包含多个 MKV 片段或包含不允许的 MKV 元素（例如track*）的 MKV 流也会导致错误确认。

Kinesis Video Streams 将每个传入的片段和相关元数据存储于所谓的“块”中。片段元数据包括以下内容：

- 请求开始时提供的 MKV 标头 PutMedia
- 以下 Kinesis Video Streams 特定于该片段的元数据：
 - server_timestamp-Kinesis Video Streams 开始接收片段的时间戳。
 - producer_timestamp-时间戳，制作人开始录制片段的时间。Kinesis Video Streams 使用请求中收到的三条信息来计算该值。
 - 与片段一起在请求正文中接收的片段时间码值。
 - 两个请求标头：producerStartTimestamp（制作人开始录制时）和fragmentTimeCodeType（有效载荷中的片段时间码是绝对的还是相对的）。

然后，Kinesis Video Streams 会根据producer_timestamp按如下方式计算片段的：

如果 `fragmentTimeCodeType` 是相对的，那么

`producer_timestamp = producerStartTimeStamp + 片段时间码`

如果 `fragmentTimeCodeType` 是绝对的，那么

`producer_timestamp = 片段时间码 (转换为毫秒)`

- 由 Kinesis Video Streams 分配的唯一片段编号。

Note

当你提出 `GetMedia` 请求时，Kinesis Video Streams 会返回这些区块的直播。客户端可以根据需要处理元数据。

Note

此操作仅适用于适用于 Java 的 AWS SDK。其他语言的 AWS SDK 不支持它。

Note

Kinesis Video Streams 在通过 API 进行摄取和存档期间不会解析和验证编解码器的私有数据。`PutMedia` 通过 HLS API 消费直播时，KVS 会从编解码器私有数据中提取并验证必要的信息，用于 MPEG-TS 和 MP4 片段打包。

Note

如果在调用 Kinesis Video Streams 媒体 API 后出现错误，则除了 HTTP 状态代码和响应正文外，还会包含以下信息：

- `x-amz-ErrorTypeHTTP` 标头 — 除了 HTTP 状态码提供的错误类型外，还包含更具体的错误类型。
- `x-amz-RequestIdHTTP` 标头 — 如果你想向报告问题 AWS，如果给出请求编号，支持团队可以更好地诊断问题。

HTTP 状态码和 `ErrorType` 标头都可用于对错误是否可重试以及在什么条件下做出编程决策，并提供有关客户端程序员可能需要采取哪些操作才能成功重试的信息。有关更多信息，请参阅本主题底部的错误部分以及[常见错误](#)。

请求语法

```
POST /putMedia HTTP/1.1
x-amzn-stream-name: StreamName
x-amzn-stream-arn: StreamARN
x-amzn-fragment-timecode-type: FragmentTimecodeType
x-amzn-producer-start-timestamp: ProducerStartTimestamp
```

Payload

URI 请求参数

请求使用以下 URI 参数。

[FragmentTimecodeType](#)

您将此值作为 `x-amzn-fragment-timecode-type` HTTP 标头进行传递。

表示片段（有效负载、HTTP 请求正文）中的时间码是绝对的还是相对的。`producerStartTimestamp` Kinesis Video Streams 使用此信息来计算 `producer_timestamp` 请求中收到的片段的，如 API 概述中所述。

有效值：ABSOLUTE | RELATIVE

必需：是

[ProducerStartTimestamp](#)

您将此值作为 `x-amzn-producer-start-timestamp` HTTP 标头进行传递。

这是制作人开始录制媒体的制作人时间戳（不是请求中特定片段的时间戳）。

[StreamARN](#)

您将此值作为 `x-amzn-stream-arn` HTTP 标头进行传递。

您要在其中写入媒体内容的 Kinesis 视频流的亚马逊资源名称 (ARN)。如果未指定 `streamARN`，则必须指定 `streamName`。

长度限制：长度下限为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

StreamName

您将此值作为 `x-amzn-stream-name` HTTP 标头进行传递。

您要在其中写入媒体内容的 Kinesis 视频流的名称。如果未指定 `streamName`，则必须指定 `streamARN`。

长度限制：最小长度为 1。最大长度为 256。

模式：`[a-zA-Z0-9_.-]+`

请求正文

请求接受以下二进制数据。

Payload

要写入 Kinesis 视频流的媒体内容。在当前的实现中，Kinesis Video Streams 仅支持带有单个 MKV 片段的 Matroska (MKV) 容器格式。一个区段可以包含一个或多个集群。

Note

每个 MKV 集群都映射到 Kinesis 视频流片段。无论您选择哪个集群持续时间，都将成为片段持续时间。

响应语法

```
HTTP/1.1 200
```

Payload

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

响应将以下内容作为 HTTP 正文返回。

Payload

Kinesis Video Streams 成功收到PutMedia请求后，该服务将验证请求标头。然后，该服务开始读取有效负载，并首先发送 HTTP 200 响应。

然后，该服务返回一个流，其中包含一系列由换行符分隔的 JSON Acknowledgement 对象（对象）。确认是在发送媒体数据的同一连接上接收的。一个PutMedia请求可能有许多确认信息。每个键值对Acknowledgement由以下键值对组成：

- AckEventType-确认所代表的事件类型。
 - 缓冲：Kinesis Video Streams 已开始接收片段。当收到片段数据的第一个字节时，Kinesis Video Streams 会发送第一个缓冲确认。
 - 已收到：Kinesis Video Streams 收到了整个片段。如果您未将流配置为保留数据，则生产者可以在收到此确认后停止缓冲片段。
 - 已@@@ 保存：Kinesis Video Streams 已将片段保存完毕（例如，保存到亚马逊 S3）。如果您将流配置为保留数据，则会收到此确认。收到此确认后，制作者可以停止缓冲片段。
 - 错误：Kinesis Video Streams 在处理片段时遇到了错误。您可以查看错误代码并确定下一步的操作方案。
 - 闲置：会PutMedia话正在进行中。但是，Kinesis Video Streams 目前没有接收数据。Kinesis Video Streams 在最后一次收到数据后最长 30 秒内定期发送此确认。如果在 30 秒内没有收到任何数据，Kinesis Video Streams 将关闭请求。

Note

这种确认可以帮助生产者确定PutMedia连接是否处于活动状态，即使它没有发送任何数据。

- FragmentTimecode-发送确认的片段时间码。

如果为 Idle，则可能缺少AckEventType该元素。

- FragmentNumber-Kinesis Video Streams 生成的已发送确认的片段编号。
- ErrorIdand ErrorCode-如果AckEventType是Error，则此字段提供相应的错误代码。以下是错误 ID 及其相应的错误代码和错误消息的列表：
 - 4000-STREAM_READ_ERROR-读取数据流时出错。
 - 4001-MAX_FRAGMENT_SIZE_REACH-片段大小超过允许的最大限制 50 MB。

- 4002-MAX_FRAGMENT_DURATION_REACH-片段持续时间大于允许的最大限制。
- 4003-MAX_CONNECTION_DURATION_DURATION_REACH-连接持续时间大于允许的最大阈值。
- 4004-FRAGMENT_TIMECODE_LESSER_THAN_PREVIOR-片段时间码小于之前的时间码（在通话中，你不能乱序发送片段）。PutMedia
- 4005-MORE_THAN_ALLOWED_TRACKS_FOUND-在 MKV 中发现了不止一首曲目。（已弃用）
- 4006-INVALID_MKV_DATA-无法将输入流解析为有效的 MKV 格式。
- 4007-INVALID_PRODUCER_TIMESTAMP-生产者时间戳无效。
- 4008-STREAM_NOT_ACTIVE-直播已不存在（已删除）。
- 4009-FRAGMENT_METADATA_LIMIT_REACH-已达到片段元数据限制。请参阅开发者指南的“[限制](#)”部分。
- 4010-TRACK_NUMBER_MISMATCH-MKV 帧中的曲目编号与 MKV 标题中的曲目不匹配。
- 4011-FRAMES_MISSING_FOR_TRACK-该片段不包含 MKV 标题中至少一条轨道的任何帧。
- 4012-INVALID_FRAGMENT_METADATA-片段元数据名称不能以字符串开头。 AWS_
- 4500-KMS_KEY_ACCESS_DENIED-访问直播中指定的 KMS 密钥被拒绝。
- 4501-KMS_KEY_DISABLED-直播中指定的 KMS 密钥已禁用。
- 4502-KMS_KEY_VALIDATION_ERROR-直播中指定的 KMS 密钥验证失败。
- 4503-KMS_KEY_UNFILABLE-直播中指定的 KMS 密钥不可用。
- 4504-KMS_KEY_INVALID_USAGE-直播中指定的 KMS 密钥的使用无效。
- 4505-KMS_KEY_INVALID_STATE-直播中指定的 KMS 密钥处于无效状态。
- 4506-KMS_KEY_NOT_FOUND-找不到直播指定的 KMS 密钥。
- 5000-内部错误-内部服务错误。
- 5001-ARCHIVAL_ERROR-Kinesis Video Streams 未能将片段保存到数据存储中。

Note

生产者在为长时间运行的PutMedia请求发送有效负载时，应阅读响应以进行确认。由于中间代理服务器上的缓冲，生产者可能会同时收到大量确认。想要及时收到确认的制作者可以在每个PutMedia请求中发送更少的片段。

错误

有关所有操作的常见错误信息，请参阅[常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为您已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

ConnectionLimitExceededException

Kinesis Video Streams 已限制该请求，因为您已超过允许的客户端连接限制。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

InvalidEndpointException

呼叫者使用了错误的端点将数据写入流。收到此类异常后，用户必须在APIName设置为的情况下调用GetDataEndpoint用，PUT_MEDIA并使用响应中的端点来调用下一个PutMedia调用。

HTTP 状态代码：400

NotAuthorizedException

调用者无权对给定直播执行操作，或者令牌已过期。

HTTP 状态代码：401

ResourceNotFoundException

状态码：404，给定名称的直播不存在。

HTTP 状态代码：404

示例

确认格式

确认的格式如下：

```
{
  Acknowledgement : {
    "EventType": enum
    "FragmentTimecode": Long,
    "FragmentNumber": Long,
    "ErrorId" : String
  }
}
```

另请参阅

有关在特定语言的 AWS SDK 中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [AWS 适用于 .NET 的 SDK](#)
- [AWS 适用于 C++ 的 SDK](#)
- [AWS 适用于 Go 的 SDK](#)
- [AWS 适用于 Java 的 SDK V2](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [AWS 适用于 PHP 的 SDK V3](#)
- [AWS Python 软件开发工具包](#)
- [AWS 适用于 Ruby V3 的 SDK](#)

Amazon Kinesis Video Streams 存档媒体

Amazon Kinesis Video Streams 存档媒体支持以下操作：

- [GetClip](#)
- [GetDASHStreamingSessionURL](#)
- [GetHLSStreamingSessionURL](#)
- [GetImages](#)
- [GetMediaForFragmentList](#)
- [ListFragments](#)

GetClip

服务：Amazon Kinesis Video Streams Archived Media

在指定时间范围内从指定视频流中下载包含已存档的点播媒体的 MP4 文件（片段）。

StreamName 和 StreamArn 参数都是可选的，但在调用此 API 操作时，必须指定 StreamName 或 StreamArn。

Note

您必须先调用 GetDataEndpoint API 才能获取终端节点。然后使用 [--endpoint-url 参数](#) 将 GetClip 请求发送到此端点。

Amazon Kinesis 视频流在通过 MP4 提供数据方面具有以下要求：

- [视频播放曲目要求](#)。
- 数据保留必须大于 0。
- 对于 H.264 格式的高级视频编码 (AVC) 和 H.265 格式的 HEVC，各个片段的视频轨道必须包含编解码器专用数据。有关更多信息，请参阅 [MPEG-4 规范 ISO/IEC 14496-15](#)。有关使流数据适应给定格式的信息，请参阅 [NAL 适应标志](#)。
- 各个片段的音频轨道（如果存在）必须包含 AAC 格式 ([AAC specification ISO/IEC 13818-7](#)) 或 [MS Wave 格式](#) 的编解码器专用数据。

您可以通过监控 GetClip.OutgoingBytes Amazon CloudWatch 指标来监控传出的数据量。有关使用 CloudWatch 监控 Kinesis Video Streams 的信息，[请参阅监控 Kinesis 视频流](#)。有关定价信息，[请参阅 Amazon Kinesis Video Streams 定价](#)。传出 AWS 数据需收费。

请求语法

```
POST /getClip HTTP/1.1
Content-type: application/json

{
  "ClipFragmentSelector": {
    "FragmentSelectorType": "string",
    "TimestampRange": {
      "EndTimeStamp": number,
      "StartTimeStamp": number
    }
  }
}
```

```
    }  
  },  
  "StreamARN": "string",  
  "StreamName": "string"  
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求体

请求接受采用 JSON 格式的以下数据。

[ClipFragmentSelector](#)

请求的片段的时间范围和时间戳的来源。

类型：[ClipFragmentSelector](#) 对象

必需：是

[StreamARN](#)

要检索媒体片段的直播的 Amazon 资源名称 (ARN)。

您必须指定或 StreamArn。 StreamName

类型：字符串

长度限制：长度下限为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

[StreamName](#)

要为其检索媒体片段的直播的名称。

您必须指定或 StreamArn。 StreamName

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

响应语法

```
HTTP/1.1 200
Content-Type: ContentType
```

```
Payload
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

响应将返回以下 HTTP 标头。

ContentType

请求的片段中媒体的内容类型。

长度限制：长度下限为 1。长度上限为 128。

模式：`^[a-zA-Z0-9_\.\\-]+$`

响应将以下内容作为 HTTP 正文返回。

Payload

传统 MP4 文件，其中包含来自指定视频流的媒体片段。输出将包含指定开始时间戳的前 100 MB 或前 200 个片段。有关更多信息，请参阅 [Kinesis Video Streams 限制](#)。

错误

有关所有操作返回的常见错误的信息，请参阅 [常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已经超过了限制。稍后再尝试拨打电话。有关限制的信息，请参阅 [Kinesis Video Streams 限制](#)。

HTTP 状态代码：400

InvalidArgumentException

指定参数超出其限制、不受支持或无法使用。

HTTP 状态代码：400

InvalidCodecPrivateDataException

视频流中至少一条轨道中的编解码器私有数据对此操作无效。

HTTP 状态代码：400

InvalidMediaFrameException

无法根据指定的编解码器解析请求片段中的一个或多个帧。

HTTP 状态代码：400

MissingCodecPrivateDataException

在视频流的至少一条轨道中未发现编解码器的私有数据。

HTTP 状态代码：400

NoDataRetentionException

GetImages请求的直播不保留数据 (即 a DataRetentionInHours 为 0) 。

HTTP 状态代码：400

NotAuthorizedException

状态码：403，调用者无权对给定直播执行操作，或者令牌已过期。

HTTP 状态代码：401

ResourceNotFoundException

GetImages当 Kinesis Video Streams 找不到你指定的直播时，将引发此错误。

GetHLSStreamingSessionURL如果请求PlaybackMode的会话在请求的时间范围内LIVE_REPLAY没有片段，ON_DEMAND或者在过去 30 秒内没有片段的流请求PlaybackMode的LIVE会话为或时，则会GetDASHStreamingSessionURL抛出此错误。

HTTP 状态代码：404

UnsupportedStreamMediaTypeException

无法根据播放会话的第一个片段中轨道的编解码器 ID 来确定媒体的类型 (例如 h.264 或 h.265 视频或 AAC 或 G.711 音频)。轨道 1 的编解码器 ID 应为 V_MPEG/ISO/AVC，轨道 2 的编解码器 ID 也应为 (可选)。A_AAC

HTTP 状态代码 : 400

另请参阅

有关在特定语言的 AWS SDK 中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [AWS 适用于 .NET 的 SDK](#)
- [AWS 适用于 C++ 的 SDK](#)
- [AWS 适用于 Go 的 SDK](#)
- [AWS 适用于 Java 的 SDK V2](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [AWS 适用于 PHP 的 SDK V3](#)
- [AWS Python 软件开发工具包](#)
- [AWS 适用于 Ruby V3 的 SDK](#)

GetDASHStreamingSessionURL

服务：Amazon Kinesis Video Streams Archived Media

检索直播的 MPEG 动态自适应流媒体通过 HTTP (DASH) 网址。然后，您可以在媒体播放器中打开 URL 来查看直播内容。

StreamName和StreamARN参数都是可选的，但在调用此 API 操作StreamARN时必须指定StreamName或。

Amazon Kinesis 视频流在通过 MPEG-DASH 提供数据时具有以下要求：

- [视频播放曲目要求](#)。
- 数据保留必须大于 0。
- 对于 H.264 格式的高级视频编码 (AVC) 和 H.265 格式的 HEVC，各个片段的视频轨道必须包含编解码器专用数据。有关更多信息，请参阅 [MPEG-4 规范 ISO/IEC 14496-15](#)。有关使流数据适应给定格式的信息，请参阅 [NAL 适应标志](#)。
- 各个片段的音频轨道（如果存在）必须包含 AAC 格式 ([AAC specification ISO/IEC 13818-7](#)) 或 [MS Wave 格式](#) 的编解码器专用数据。

以下过程展示了如何在 Kinesis Video Streams 中使用 MPEG-DASH：

1. 调用 GetDataEndpoint API 获取终端节点。然后使用 [--endpoint-url 参数](#) 将 [GetDASHStreamingSessionURL](#) 请求发送到此端点。
2. 使用检索 MPEG-DASH 网址。GetDASHStreamingSessionURL Kinesis Video Streams 创建了一个 MPEG-DASH 直播会话，用于使用 MPEG-DASH 协议访问直播中的内容。GetDASHStreamingSessionURL 返回会话的 MPEG-DASH 清单（使用 MPEG-DASH 进行直播所需的根资源）的经过身份验证的 URL（包括加密的会话令牌）。

Note

请勿将此令牌共享或存储在未经授权的实体可以访问的地方。该令牌提供对直播内容的访问权限。使用与 AWS 凭证相同的措施来保护令牌。

通过清单提供的媒体仅包含请求的直播、时间范围和格式。没有其他媒体数据（例如请求的窗口之外的帧或备用比特率）可用。

3. 向支持 MPEG-DASH 协议的媒体播放器提供 MPEG-DASH 清单的 URL (包含加密的会话令牌)。Kinesis Video Streams 通过清单 URL 提供初始化片段和媒体片段。初始化片段包含流的编解码器私有数据，以及设置视频或音频解码器和渲染器所需的其他数据。媒体片段包含编码的视频帧或编码的音频样本。
4. 媒体播放器会收到经过身份验证的 URL，并正常请求流元数据和媒体数据。当媒体播放器请求数据时，它会调用以下操作：
 - `getDashManifest`：检索 MPEG DASH 清单，其中包含您要播放的媒体的元数据。
 - `getMP4InitFragment`：检索 MP4 初始化片段。媒体播放器通常会在加载任何媒体片段之前加载初始化片段。此片段包含“fytp”和“moov”MP4 原子，以及初始化媒体播放器解码器所需的子原子。

初始化片段与 Kinesis 视频流中的片段不对应。它仅包含直播和相应轨道的编解码器私有数据，媒体播放器需要这些数据来解码媒体帧。

- `FetchMP4MediaFragment`：检索 MP4 媒体片段。这些片段包含“moof”和“mdat”MP4 原子及其子原子，包含编码片段的媒体帧及其时间戳。

Note

在流媒体会话中提供第一个媒体片段后，任何不包含相同编解码器私有数据的片段都会导致在加载这些不同的媒体片段时返回错误。因此，编解码器的私有数据不应在会话中的片段之间发生变化。这也意味着，如果直播中的片段从只有视频变为同时包含音频和视频，则会话将失败。

通过此操作检索到的数据是可计费的。有关详细信息，请参阅[定价](#)。

Note

有关适用于 MPEG-DASH 会话的限制，请参阅[Kinesis Video Streams 限制](#)。

您可以通过监控 `GetMP4MediaFragment.OutgoingBytes` Amazon CloudWatch 指标来监控媒体播放器消耗的数据量。有关使用 CloudWatch 监控 Kinesis Video Streams 的信息，请参阅[监控 Kinesis 视频流](#)。有关定价信息，请参阅[Amazon Kinesis Video Streams 定价](#)[AWS 和定价](#)。HLS 会话和传出 AWS 数据均需收费。

有关 HLS 的更多信息，请参阅[Apple 开发者网站上的 HTTP 直播](#)。

⚠ Important

如果在调用 Kinesis Video Streams 存档媒体 API 后出现错误，则除了 HTTP 状态代码和响应正文外，还会包含以下信息：

- `x-amz-ErrorTypeHTTP` 标头 — 除了 HTTP 状态码提供的错误类型外，还包含更具体的错误类型。
- `x-amz-RequestIdHTTP` 标头 — 如果您想向支持团队报告问题，如果给出请求编号，则可以更好地诊断问题。AWS

HTTP 状态码和 `ErrorType` 标头都可用于对错误是否可重试以及在什么条件下做出编程决策，并提供有关客户端程序员可能需要采取哪些操作才能成功重试的信息。

有关更多信息，请参阅本主题底部的错误部分以及[常见错误](#)。

请求语法

```
POST /getDASHStreamingSessionURL HTTP/1.1
Content-type: application/json
```

```
{
  "DASHFragmentSelector": {
    "FragmentSelectorType": "string",
    "TimestampRange": {
      "EndTimestamp": number,
      "StartTimestamp": number
    }
  },
  "DisplayFragmentNumber": "string",
  "DisplayFragmentTimestamp": "string",
  "Expires": number,
  "MaxManifestFragmentResults": number,
  "PlaybackMode": "string",
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求体

请求接受采用 JSON 格式的以下数据。

[DASHFragmentSelector](#)

所请求片段的时间范围和时间戳的来源。

如果为ON_DEMAND或，则此参数PlaybackMode为必填项LIVE_REPLAY。如果是，则此参数是可选 PlaybackMode 的LIVE。如果PlaybackMode是LIVE，则FragmentSelectorType可以设置，但TimestampRange不应设置。如果PlaybackMode为ON_DEMAND或LIVE_REPLAY，则TimestampRange必须同时设置FragmentSelectorType和。

类型：[DASHFragmentSelector](#) 对象

必需：否

[DisplayFragmentNumber](#)

在清单文件中，片段是根据其在会话中的序列号来识别的。如果设置 DisplayFragmentNumber 为ALWAYS，则 Kinesis Video Streams 片段编号将添加到清单文件中的每个 S 元素中，属性名为“kvs: fn”。这些片段编号可用于日志记录或与其他 API 一起使用（例如GetMedia和GetMediaForFragmentList）。要利用这些自定义属性，必须使用自定义 MPEG-DASH 媒体播放器。

默认值为 NEVER。

类型：字符串

有效值：ALWAYS | NEVER

必需：否

[DisplayFragmentTimestamp](#)

根据 MPEG-DASH 规范，清单文件中片段的挂钟时间可以使用清单本身中的属性得出。但是，通常，兼容 MPEG-DASH 的媒体播放器无法正确处理媒体时间轴中的空白。Kinesis Video Streams 调整清单文件中的媒体时间轴，以允许播放不连续的媒体。因此，从清单文件中得出的挂钟时间可能不准确。如果设置 DisplayFragmentTimestamp 为ALWAYS，则将精确的片段时间戳添加到清单文件中的每个 S 元素中，属性名为“kvs: ts”。要利用此自定义属性，必须使用自定义 MPEG-DASH 媒体播放器。

默认值为 NEVER。如果 [DASHFragmentSelector](#) 是 SERVER_TIMESTAMP，则时间戳将是服务器启动时间戳。同样，如果 [DASHFragmentSelector](#) 是 PRODUCER_TIMESTAMP，则时间戳将是生产者的开始时间戳。

类型：字符串

有效值：ALWAYS | NEVER

必需：否

[Expires](#)

请求的会话到期之前的时间（以秒为单位）。此值可以介于 300（5 分钟）和 43200（12 小时）之间。

会话到期后，不能对该会话进行任何新的调用 `GetMP4InitFragment`、`GetMP4MediaFragment` 或 `GetDashManifest`

默认值为 300（5 分钟）。

类型：整数

有效范围：最小值为 300。最大值为 43200。

必需：否

[MaxManifestFragmentResults](#)

MPEG-DASH 清单中返回的最大片段数。

如果 `PlaybackMode` 为 LIVE，则返回最新的片段，直至该值。如果 `PlaybackMode` 为 ON_DEMAND，则返回最旧的片段，不超过这个最大数目。

当实时 MPEG-DASH 清单中可用的片段数量较多时，视频播放器通常会在开始播放之前缓冲内容。增加缓冲区大小会增加播放延迟，但会降低播放期间发生重新缓冲的可能性。我们建议实时 MPEG-DASH 清单至少包含 3 个片段，最多 10 个片段。

如果为 LIVE 或，则默认为 5 个片段，如果 `PlaybackMode` 为 LIVE_REPLAY，则 `PlaybackMode` 默认为 1,000 ON_DEMAND。

1,000 个片段的最大值对应于包含 1 秒片段的直播中超过 16 分钟的视频，对应于包含 10 秒片段的直播中超过 2 个半小时的视频。

类型：长整型

有效范围：最小值为 1。最大值为 5000。

必需：否

[PlaybackMode](#)

是检索实时、实时重播还是存档的按需数据。

这三种类型的会话的特点包括：

- **LIVE**：对于此类会话，MPEG-DASH 清单会在可用的最新片段时不断更新。我们建议媒体播放器每隔一秒钟检索新的清单。在媒体播放器中播放此类会话时，用户界面通常会显示“实时”通知，没有用于在播放窗口中选择要显示的位置的滑块控件。

Note

在LIVE模式下，即使片段之间存在间隙（也就是说，如果缺少片段），最新的可用片段也会包含在 MPEG-DASH 清单中。这样的间隙可能会导致媒体播放器停止播放或导致播放跳跃。在此模式下，如果片段早于播放列表中的最新片段，则不会将其添加到 MPEG-DASH 清单中。如果在将后续片段添加到清单后丢失的片段变为可用，则不会添加较旧的片段，也不会填补空白。

- **LIVE_REPLAY**：对于此类会话，MPEG-DASH 清单的更新方式与LIVE模式更新方式类似，不同之处在于它首先包含给定开始时间的片段。片段不是在摄取时添加片段，而是在下一个片段的持续时间过去时添加片段。例如，如果会话中的片段长度为两秒，则每两秒钟就会向清单中添加一个新片段。此模式非常有用，可以从检测到事件时开始播放，并继续直播截至会话创建时尚未收录的媒体。此模式还可用于流式传输先前存档的媒体，而不受该ON_DEMAND模式下 1,000 个片段限制的限制。
- **ON_DEMAND**：对于此类会话，MPEG-DASH 清单包含会话的所有片段，但不超过中指定的数字。MaxManifestFragmentResults每次会话只能检索一次清单。在媒体播放器中播放此类会话时，用户界面通常会显示一个滑块控件，用于在播放窗口中选择要显示的位置。

在所有播放模式下，如果FragmentSelectorType是PRODUCER_TIMESTAMP，如果有多个片段的开始时间戳相同，则片段编号较大的片段（即较新的片段）将包含在 MPEG-DASH 清单中。其他片段不包括在内。具有不同时间戳但持续时间重叠的片段仍包含在 MPEG-DASH 清单中。这可能会导致媒体播放器出现意外行为。

默认值为 LIVE。

类型：字符串

有效值：LIVE | LIVE_REPLAY | ON_DEMAND

必需：否

StreamARN

要检索 MPEG-DASH 清单网址的直播的亚马逊资源名称 (ARN)。

必须指定StreamName或StreamARN。

类型：字符串

长度限制：长度下限为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

要检索 MPEG-DASH 清单网址的直播名称。

必须指定StreamName或StreamARN。

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

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "DASHStreamingSessionURL": "string"
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回以下数据。

[DASHStreamingSessionURL](#)

媒体播放器可用于检索 MPEG-DASH 清单的网址（包含会话令牌）。

类型：字符串

错误

有关所有操作返回的常见错误的信息，请参阅 [常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过限制。稍后再尝试拨打电话。有关限制的信息，请参阅 [Kinesis Video Streams](#) 限制。

HTTP 状态代码：400

InvalidArgumentException

指定参数超出其限制、不受支持或无法使用。

HTTP 状态代码：400

InvalidCodecPrivateDataException

视频流中至少一条轨道中的编解码器私有数据对此操作无效。

HTTP 状态代码：400

MissingCodecPrivateDataException

在视频流的至少一条轨道中未发现编解码器的私有数据。

HTTP 状态代码：400

NoDataRetentionException

GetImages 请求的直播不保留数据（即 a DataRetentionInHours 为 0）。

HTTP 状态代码：400

NotAuthorizedException

状态码：403，调用者无权对给定直播执行操作，或者令牌已过期。

HTTP 状态代码：401

ResourceNotFoundException

GetImages当 Kinesis Video Streams 找不到你指定的直播时，将引发此错误。

GetHLSStreamingSessionURL如果请求PlaybackMode的会话在请求的时间范围内LIVE_REPLAY没有片段，ON_DEMAND或者在过去 30 秒内没有片段的流请求PlaybackMode的LIVE会话为或时，则会GetDASHStreamingSessionURL抛出此错误。

HTTP 状态代码：404

UnsupportedStreamMediaTypeException

无法根据播放会话的第一个片段中轨道的编解码器 ID 来确定媒体的类型（例如 h.264 或 h.265 视频或 AAC 或 G.711 音频）。轨道 1 的编解码器 ID 应为V_MPEG/ISO/AVC，轨道 2 的编解码器 ID 也应为（可选）。A_AAC

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS SDK 中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [AWS 适用于 .NET 的 SDK](#)
- [AWS 适用于 C++ 的 SDK](#)
- [AWS 适用于 Go 的 SDK](#)
- [AWS 适用于 Java 的 SDK V2](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [AWS 适用于 PHP 的 SDK V3](#)
- [AWS Python 软件开发工具包](#)
- [AWS 适用于 Ruby V3 的 SDK](#)

GetHLSStreamingSessionURL

服务：Amazon Kinesis Video Streams Archived Media

检索直播的 HTTP 直播 (HLS) 网址。然后，您可以在浏览器或媒体播放器中打开 URL 来查看直播内容。

StreamName和StreamARN参数都是可选的，但在调用此 API 操作StreamARN时必须指定StreamName或。

要通过 HLS 提供数据，Amazon Kinesis 视频流需要满足以下要求：

- [视频播放曲目要求](#)。
- 数据保留必须大于 0。
- [每个片段的视频轨道必须包含 H.264 格式的高级视频编码 \(AVC\) 或 H.265 格式的 HEVC \(MPEG-4 规范 ISO/IEC 14496-15 \) 中的编解码器私有数据](#)。有关使流数据适应给定格式的信息，请参阅 [NAL 适应标志](#)。
- 每个片段的音轨 (如果存在) 必须包含 AAC 格式的编解码器私有数据 (AAC [规范 ISO/IEC 13818-7](#)) 。

Kinesis Video Streams HLS 会话包含分散的 MPEG-4 形式 (也称为 fmP4 或 CMAF) 或 MPEG-2 形式 (也称为 TS 块，HLS 规范也支持这种片段)。有关 HLS 片段类型的更多信息，请参阅 [HLS 规范](#)。

以下过程说明如何将 HLS 与 Kinesis Video Streams 配合使用：

1. 调用 GetDataEndpoint API 获取终端节点。然后使用 [--endpoint-url 参数](#) 将[GetHLSStreamingSessionURL](#)请求发送到此端点。
2. 使用GetHLSStreamingSessionURL检索 HLS 网址。Kinesis Video Streams 创建一个 HLS 直播会话，用于使用 HLS 协议访问直播中的内容。GetHLSStreamingSessionURL返回会话的 HLS 主播放列表 (使用 HLS 进行直播所需的根资源) 的经过身份验证的 URL (包括加密的会话令牌)。

Note

请勿将此令牌共享或存储在未经授权的实体可以访问的地方。该令牌提供对直播内容的访问权限。使用与 AWS 凭证相同的措施来保护令牌。

通过播放列表提供的媒体仅包含请求的直播、时间范围和格式。没有其他媒体数据（例如请求的窗口之外的帧或备用比特率）可用。

3. 向支持 HLS 协议的媒体播放器提供 HLS 主播放列表的 URL（包含加密的会话令牌）。Kinesis Video Streams 通过主播放列表网址提供 HLS 媒体播放列表、初始化片段和媒体片段。初始化片段包含流的编解码器私有数据，以及设置视频或音频解码器和渲染器所需的其他数据。媒体片段包含 H.264 编码的视频帧或 AAC 编码的音频样本。
4. 媒体播放器会收到经过身份验证的 URL，并正常请求流元数据和媒体数据。当媒体播放器请求数据时，它会调用以下操作：
 - `getHLSMasterPlaylist`：检索 HLS 主播放列表，其中包含每首曲目的 `GetHLSMediaPlaylist` 动作网址以及媒体播放器的其他元数据，包括估计的比特率和分辨率。
 - `getHLSMediaPlaylist`：检索 HLS 媒体播放列表，其中包含用于通过操作访问 MP4 初始化片段的网址，以及用于通过 `GetMP4InitFragment` 操作访问 MP4 媒体片段的网址。`GetMP4MediaFragmentHLS` 媒体播放列表还包含有关播放器需要播放的直播的元数据，例如是 `PlaybackModeLIVE` 是 `ON_DEMAND`。对于具有以下内容的会话，HLS 媒体播放列表通常是静态 `PlaybackType` 的。`ON_DEMAND` HLS 媒体播放列表会不断更新，其中包含会话的新片段 `PlaybackType`。LIVE 视频曲目和音轨（如果适用）有一个不同的 HLS 媒体播放列表，其中包含特定曲目的 MP4 媒体网址。
 - `getMP4InitFragment`：检索 MP4 初始化片段。媒体播放器通常会在加载任何媒体片段之前加载初始化片段。此片段包含“`fytp`”和“`moov`”MP4 原子，以及初始化媒体播放器解码器所需的原子。

初始化片段与 Kinesis 视频流中的片段不对应。它仅包含直播和相应轨道的编解码器私有数据，媒体播放器需要这些数据来解码媒体帧。

- `获取 MP4MediaFragment`：检索 MP4 媒体片段。这些片段包含“`moof`”和“`mdat`”MP4 原子及其子原子，包含编码片段的媒体帧及其时间戳。

Note

对于 HLS 直播会话，支持更改轨内编解码器私有数据 (CPD)。在直播会话中提供第一个媒体片段后，片段可以包含每首曲目的 CPD 更改。因此，会话中的片段在 CPD 中可能具有不同的分辨率、比特率或其他信息，而不会中断播放。但是，加载这些不同的媒体片段时，对曲目编解码器或曲目编解码器格式所做的任何更改都可能返回错误。例如，如果流中的片段从只有视频变为同时包含音频和视频，或者将 AAC 音轨更改为 ALAW 音轨，则直播将失败。对于每个直播会话，只允许更改 500 个 CPD。

通过此操作检索到的数据是可计费的。有关信息，请参阅 [定价](#)。

- GettsFragment：检索包含直播中所有曲目的初始化和媒体数据的 MPEG TS 片段。

Note

如果ContainerFormat是MPEG_TS，则使用此 API 代替GetMP4InitFragment和GetMP4MediaFragment来检索流媒体。

通过此操作检索到的数据是可计费的。有关更多信息，请参阅 [Kinesis Video Streams 定价](#)。

直播会话网址不得在玩家之间共享。如果多个媒体播放器共享会话，该服务可能会限制该会话。有关连接限制，请参阅 [Kinesis Video Streams 限制](#)。

您可以通过监控 GetMP4MediaFragment.OutgoingBytes Amazon CloudWatch 指标来监控媒体播放器消耗的数据量。有关使用 CloudWatch 监控 Kinesis Video Streams 的信息，[请参阅监控 Kinesis 视频流](#)。有关定价信息，请参阅[亚马逊 Kinesis Video Streams 定价](#)[AWS 和定价](#)。HLS 会话和传出 AWS 数据均需收费。

有关 HLS 的更多信息，请参阅 [Apple 开发者网站上的 HTTP 直播](#)。

Important

如果在调用 Kinesis Video Streams 存档媒体 API 后出现错误，则除了 HTTP 状态代码和响应正文外，还会包含以下信息：

- x-amz-ErrorTypeHTTP 标头 — 除了 HTTP 状态码提供的错误类型外，还包含更具体的错误类型。
- x-amz-RequestIdHTTP 标头 — 如果你想向报告问题 AWS，如果给出请求编号，支持团队可以更好地诊断问题。

HTTP 状态码和 ErrorType 标头都可用于对错误是否可重试以及在什么条件下做出编程决策，并提供有关客户端程序员可能需要采取哪些操作才能成功重试的信息。

有关更多信息，请参阅本主题底部的错误部分以及[常见错误](#)。

请求语法

```
POST /getHLSStreamingSessionURL HTTP/1.1
Content-type: application/json

{
  "ContainerFormat": "string",
  "DiscontinuityMode": "string",
  "DisplayFragmentTimestamp": "string",
  "Expires": number,
  "HLSFragmentSelector": {
    "FragmentSelectorType": "string",
    "TimestampRange": {
      "EndTimestamp": number,
      "StartTimestamp": number
    }
  },
  "MaxMediaPlaylistFragmentResults": number,
  "PlaybackMode": "string",
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求体

请求接受采用 JSON 格式的以下数据。

ContainerFormat

指定应使用哪种格式来包装媒体。指定FRAGMENTED_MP4容器格式会将媒体打包为 MP4 片段 (fMP4 或 CMAF)。这是推荐的包装，因为包装开销最小。另一个容器格式选项是MPEG_TS。HLS 自 MPEG TS 区块发布以来一直支持，有时是旧版 HLS 播放器唯一支持的封装。MPEG TS 的封装开销通常为 5-25%。这意味着 MPEG TS 需要的带宽和成本通常比 fMP4 高 5-25%。

默认值为 FRAGMENTED_MP4。

类型：字符串

有效值：FRAGMENTED_MP4 | MPEG_TS

必需：否

[DiscontinuityMode](#)

指定何时将标记片段之间不连续性的标记添加到媒体播放列表中。

媒体播放器通常会根据每个片段的时间戳来建立要播放的媒体内容的时间表。这意味着，如果片段之间存在任何重叠或间隙（如果设置[HLSFragmentSelector](#)为，则通常如此SERVER_TIMESTAMP），则媒体播放器时间轴在某些地方的片段之间也会有小间隙，并且会覆盖其他位置的帧。媒体播放器时间轴中的间隙可能会导致播放停滞，而重叠可能会导致播放抖动。当片段之间存在不连续标记时，媒体播放器应重置时间轴，从而在上一个片段之后立即播放下一个片段。

支持以下模式：

- ALWAYS：在 HLS 媒体播放列表中的每个片段之间都放置了一个不连续标记。ALWAYS如果片段时间戳不准确，建议使用值。
- NEVER：任何地方都没有放置不连续标记。建议使用值，NEVER以确保媒体播放器时间轴最准确地映射到制作者时间戳。
- ON_DISCONTINUITY：在间隙或重叠超过 50 毫秒的片段之间放置不连续标记。对于大多数播放场景，建议使用值，ON_DISCONTINUITY这样只有在媒体时间轴出现重大问题（例如缺少片段）时，才会重置媒体播放器时间轴。

默认值为ALWAYS何时设置[HLSFragmentSelector](#)为SERVER_TIMESTAMP，NEVER何时设置为PRODUCER_TIMESTAMP。

类型：字符串

有效值：ALWAYS | NEVER | ON_DISCONTINUITY

必需：否

[DisplayFragmentTimestamp](#)

指定片段开始时间戳何时应包含在 HLS 媒体播放列表中。通常，媒体播放器将播放头的位置报告为相对于播放会话中第一个片段开始的时间。但是，当开始时间戳包含在 HLS 媒体播放列表中时，某些媒体播放器可能会根据片段时间戳将当前播放头报告为绝对时间。这对于创建向观众显示媒体挂钟时间的播放体验非常有用。

默认值为 NEVER。如果 [HLSFragmentSelector](#) 是 SERVER_TIMESTAMP，则时间戳将是服务器启动时间戳。同样，如果 [HLSFragmentSelector](#) 是 PRODUCER_TIMESTAMP，则时间戳将是生产者的开始时间戳。

类型：字符串

有效值：ALWAYS | NEVER

必需：否

[Expires](#)

请求的会话到期之前的时间（以秒为单位）。此值可以介于 300（5 分钟）和 43200（12 小时）之间。

会话过期后，无法对该会话进行任何新的调

用 `GetHLSMasterPlaylist`、`GetHLSMediaPlaylist`、`GetMP4InitFragment`、`GetMP4MediaFragment` 或 `GetTSFragment`。

默认值为 300（5 分钟）。

类型：整数

有效范围：最小值为 300。最大值为 43200。

必需：否

[HLSFragmentSelector](#)

所请求片段的时间范围和时间戳的来源。

如果 `PlaybackMode` 为 `ON_DEMAND` 或 `LIVE_REPLAY`，则需要此参数 `LIVE_REPLAY`。如果是，则此参数是可选 `PlaybackMode` 的 `LIVE`。如果 `PlaybackMode` 是 `LIVE`，则 `FragmentSelectorType` 可以设置，但 `TimestampRange` 不应设置。如果 `PlaybackMode` 为 `ON_DEMAND` 或 `LIVE_REPLAY`，则 `TimestampRange` 必须同时设置 `FragmentSelectorType` 和 `TimestampRange`。

类型：[HLSFragmentSelector](#) 对象

必需：否

[MaxMediaPlaylistFragmentResults](#)

HLS 媒体播放列表中返回的最大片段数。

如果PlaybackMode为LIVE，则返回最新的片段，直至该值。如果PlaybackMode为ON_DEMAND，则返回最旧的片段，不超过这个最大数目。

当 HLS 直播媒体播放列表中有更多片段时，视频播放器通常会在开始播放之前缓冲内容。增加缓冲区大小会增加播放延迟，但会降低播放期间发生重新缓冲的可能性。我们建议直播的 HLS 媒体播放列表至少包含 3 个片段，最多 10 个片段。

如果为LIVE或，则默认为 5 个片段，如果PlaybackMode为LIVE_REPLAY，则PlaybackMode默认为 1,000 ON_DEMAND。

5,000 个片段的最大值对应于包含 1 秒片段的直播上超过 80 分钟的视频，对应于包含 10 秒片段的直播上超过 13 小时的视频。

类型：长整型

有效范围：最小值为 1。最大值为 5000。

必需：否

PlaybackMode

是检索实时、实时重播还是存档的按需数据。

这三种类型的会话的特点包括：

- **LIVE**：对于此类会话，HLS 媒体播放列表会不断更新最新片段。我们建议媒体播放器每隔一秒钟检索一个新的播放列表。在媒体播放器中播放此类会话时，用户界面通常会显示“实时”通知，没有用于在播放窗口中选择要显示的位置的滑块控件。

Note

在LIVE模式下，即使片段之间存在间隙（也就是说，如果缺少片段），最新的可用片段也会包含在 HLS 媒体播放列表中。这样的间隙可能会导致媒体播放器停止播放或导致播放跳跃。在此模式下，如果片段早于播放列表中的最新片段，则不会将其添加到 HLS 媒体播放列表中。如果在将后续片段添加到播放列表后丢失的片段变为可用，则不会添加较旧的片段，也不会填补空白。

- **LIVE_REPLAY**：对于此类会话，HLS 媒体播放列表的更新方式与更新LIVE模式的方式类似，不同之处在于它首先包含给定开始时间的片段。片段不是在摄取时添加片段，而是在下一个片段的持续时间过去时添加片段。例如，如果会话中的片段长度为两秒，则每两秒钟就会向媒体播放列表中添加一个新片段。此模式非常有用，可以从检测到事件时开始播放，并继续直播截至会话创

建时尚未收录的媒体。此模式还可用于流式传输先前存档的媒体，而不受该ON_DEMAND模式下 1,000 个片段限制的限制。

- **ON_DEMAND**：对于此类会话，HLS 媒体播放列表包含会话的所有片段，但不超过 `MaxMediaPlaylistFragmentResults` 指定的数字。每个会话只能检索一次播放列表。在媒体播放器中播放此类会话时，用户界面通常会显示一个滑块控件，用于在播放窗口中选择要显示的位置。

在所有播放模式下，如果 `FragmentSelectorType` 是 `PRODUCER_TIMESTAMP`，如果有多个片段的开始时间戳相同，则片段数最大（即最新片段）的片段将包含在 HLS 媒体播放列表中。其他片段不包括在内。具有不同时间戳但持续时间重叠的片段仍包含在 HLS 媒体播放列表中。这可能会导致媒体播放器出现意外行为。

默认值为 LIVE。

类型：字符串

有效值：LIVE | LIVE_REPLAY | ON_DEMAND

必需：否

StreamARN

要检索 HLS 主播放列表网址的直播的亚马逊资源名称 (ARN)。

必须指定 `StreamName` 或 `StreamARN`。

类型：字符串

长度限制：长度下限为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

要检索 HLS 主播放列表网址的直播名称。

必须指定 `StreamName` 或 `StreamARN`。

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

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "HLSStreamingSessionURL": "string"
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回以下数据。

[HLSStreamingSessionURL](#)

媒体播放器可用于检索 HLS 主播放列表的 URL（包含会话令牌）。

类型：字符串

错误

有关所有操作返回的常见错误的信息，请参阅 [常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过限制。稍后再尝试拨打电话。有关限制的信息，请参阅 [Kinesis Video Streams 限制](#)。

HTTP 状态代码：400

InvalidArgumentException

指定参数超出其限制、不受支持或无法使用。

HTTP 状态代码：400

InvalidCodecPrivateDataException

视频流中至少一条轨道中的编解码器私有数据对此操作无效。

HTTP 状态代码：400

MissingCodecPrivateDataException

在视频流的至少一条轨道中未发现编解码器的私有数据。

HTTP 状态代码：400

NoDataRetentionException

GetImages请求的直播不保留数据（即 a DataRetentionInHours 为 0）。

HTTP 状态代码：400

NotAuthorizedException

状态码：403，调用者无权对给定直播执行操作，或者令牌已过期。

HTTP 状态代码：401

ResourceNotFoundException

GetImages当 Kinesis Video Streams 找不到你指定的直播时，将引发此错误。

GetHLSStreamingSessionURL如果请求PlaybackMode的会话在请求的时间范围内LIVE_REPLAY没有片段，ON_DEMAND或者在过去 30 秒内没有片段的流请求PlaybackMode的LIVE会话为或时，则会GetDASHStreamingSessionURL抛出此错误。

HTTP 状态代码：404

UnsupportedStreamMediaTypeException

无法根据播放会话的第一个片段中轨道的编解码器 ID 来确定媒体的类型（例如 h.264 或 h.265 视频或 AAC 或 G.711 音频）。轨道 1 的编解码器 ID 应为V_MPEG/ISO/AVC，轨道 2 的编解码器 ID 也应为（可选）。A_AAC

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS SDK 中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [AWS 适用于 .NET 的 SDK](#)
- [AWS 适用于 C++ 的 SDK](#)
- [AWS 适用于 Go 的 SDK](#)
- [AWS 适用于 Java 的 SDK V2](#)
- [AWS JavaScript V3 版 SDK](#)
- [AWS 适用于 PHP 的 SDK V3](#)
- [AWS Python 软件开发工具包](#)
- [AWS 适用于 Ruby V3 的 SDK](#)

GetImages

服务：Amazon Kinesis Video Streams Archived Media

检索与给定时间范围、采样间隔和图像格式配置的每个时间戳对应的图像列表。

Note

您必须先调用 `GetDataEndpoint` API 才能获取终端节点。然后使用 `--endpoint-url` 参数将 `GetImages` 请求发送到此端点。

[视频播放曲目要求。](#)

请求语法

```
POST /getImages HTTP/1.1
Content-type: application/json

{
  "EndTimeStamp": number,
  "Format": "string",
  "FormatConfig": {
    "string" : "string"
  },
  "HeightPixels": number,
  "ImageSelectorType": "string",
  "MaxResults": number,
  "NextToken": "string",
  "SamplingInterval": number,
  "StartTimestamp": number,
  "StreamARN": "string",
  "StreamName": "string",
  "WidthPixels": number
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求体

请求接受采用 JSON 格式的以下数据。

EndTimeStamp

要生成的图像范围的结束时间戳。如果介于StartTimeStamp和之间的时间范围大EndTimeStamp于 300 秒StartTimeStamp，您将收到IllegalArgumentException。

类型：时间戳

必需：是

Format

将用于对图像进行编码的格式。

类型：字符串

有效值：JPEG | PNG

必需：是

FormatConfig

键值对结构的列表，其中包含可在生成图像时应用的额外参数。FormatConfig密钥是JPEGQuality，它表示用于生成图像的 JPEG 质量密钥。该FormatConfig值接受介于 1 到 100 之间的整数。如果该值为 1，则将生成质量较低且压缩效果最佳的图像。如果该值为 100，则将生成质量最好、压缩率更低的图像。如果未提供任何值，则JPEGQuality密钥的默认值将设置为 80。

类型：字符串到字符串映射

映射条目：最多 1 个物品。

有效密钥：JPEGQuality

值长度限制：最小长度为 0。最大长度为 256。

价值模式： $^[a-zA-Z_0-9]^+$

必需：否

HeightPixels

与WidthPixels参数一起使用的输出图像的高度。当同时提供HeightPixels和WidthPixels参数时，图像将被拉伸以适合指定的纵横比。如果仅提供了HeightPixels参数，则将使用其原始纵横比来计算该WidthPixels比率。如果两个参数均未提供，则将返回原始图像尺寸。

类型：整数

有效范围：最小值为 1。最大值为 2160。

必需：否

ImageSelectorType

用于生成图像的服务器或制作者时间戳的来源。

类型：字符串

有效值：PRODUCER_TIMESTAMP | SERVER_TIMESTAMP

必需：是

MaxResults

API 返回的最大图像数。

Note

每个 API 响应的默认限制为 25 张图片。如果提供的值MaxResults大于此值，则页面大小将为 25。任何其他结果都将进行分页。

类型：长整型

有效范围：最小值为 1。最大值为 100。

必需：否

NextToken

一种标记，用于指定从何处开始对下一组图像进行分页。这是之前截GetImages:NextToken断的响应。

类型：字符串

长度限制：长度下限为 1。最大长度为 4096。

模式：`[a-zA-Z0-9+/\]{0,2}`

必需：否

SamplingInterval

需要从流中生成图像的时间间隔，以毫秒 (ms) 为单位。可以提供的最小值为 200 ms (每秒 5 张图像)。如果时间戳范围小于采样间隔，则startTimestamp将返回的图像 (如果有)。

类型：整数

必需：是

StartTimestamp

应从中生成图像的起点。这StartTimestamp必须在包含的时间戳范围内，才能返回图像。

类型：时间戳

必需：是

StreamARN

要从中检索图像的直播的 Amazon 资源名称 (ARN)。必须指定StreamName或StreamARN。

类型：字符串

长度限制：长度下限为 1。长度上限为 1024。

模式：arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+

必需：否

StreamName

要从中检索图像的流的名称。必须指定StreamName或StreamARN。

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

WidthPixels

与HeightPixels参数一起使用的输出图像的宽度。当同时提供WidthPixels和HeightPixels参数时，图像将被拉伸以适合指定的纵横比。如果

只提供了WidthPixels参数或者只提供了参数，则ValidationException会抛出。HeightPixels如果两个参数都未提供，则将返回流中的原始图像大小。

类型：整数

有效范围：最小值为 1。最大值为 3840。

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "Images": [
    {
      "Error": "string",
      "ImageContent": "string",
      "TimeStamp": number
    }
  ],
  "NextToken": "string"
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回以下数据。

Images

从视频流生成的图像列表。如果在给定时间戳内没有可用的媒体，则NO_MEDIA错误将在输出中列出。如果在生成图像时出现错误，则MEDIA_ERROR将在输出中列为图像丢失的原因。

类型：[Image](#) 对象数组

NextToken

请求中用于获取更多图像的加密令牌。

类型：字符串

长度限制：长度下限为 1。最大长度为 4096。

模式：`[a-zA-Z0-9+/\]{0,2}`

错误

有关所有操作返回的常见错误的信息，请参阅 [常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过限制。稍后再尝试拨打电话。有关限制的信息，请参阅 [Kinesis Video Streams 限制](#)。

HTTP 状态代码：400

InvalidArgumentException

指定参数超出其限制、不受支持或无法使用。

HTTP 状态代码：400

NotAuthorizedException

状态码：403，调用者无权对给定直播执行操作，或者令牌已过期。

HTTP 状态代码：401

ResourceNotFoundException

GetImages 当 Kinesis Video Streams 找不到你指定的直播时，将引发此错误。

GetHLSStreamingSessionURL 如果请求 PlaybackMode 的会话在请求的时间范围内 LIVE_REPLAY 没有片段，ON_DEMAND 或者在过去 30 秒内没有片段的流请求 PlaybackMode 的 LIVE 会话为或时，则会 GetDASHStreamingSessionURL 抛出此错误。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS SDK 中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [AWS 适用于 .NET 的 SDK](#)

- [AWS 适用于 C++ 的 SDK](#)
- [AWS 适用于 Go 的 SDK](#)
- [AWS 适用于 Java 的 SDK V2](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [AWS 适用于 PHP 的 SDK V3](#)
- [AWS Python 软件开发工具包](#)
- [AWS 适用于 Ruby V3 的 SDK](#)

GetMediaForFragmentList

服务：Amazon Kinesis Video Streams Archived Media

从 Amazon Kinesis 视频流中的存档数据中获取片段列表（按片段号指定）的媒体。

Note

您必须先调用 `GetDataEndpoint` API 才能获取终端节点。然后使用 [--endpoint-url 参数](#) 将 `GetMediaForFragmentList` 请求发送到此端点。

有关限制，请参阅 [Kinesis Video Streams](#) 限制。

Important

如果在调用 Kinesis Video Streams 存档媒体 API 后出现错误，则除了 HTTP 状态代码和响应正文外，还会包含以下信息：

- `x-amz-ErrorTypeHTTP` 标头 — 除了 HTTP 状态码提供的错误类型外，还包含更具体的错误类型。
- `x-amz-RequestIdHTTP` 标头 — 如果你想向报告问题 AWS，如果给出请求编号，支持团队可以更好地诊断问题。

HTTP 状态码和 `ErrorType` 标头都可用于对错误是否可重试以及在什么条件下做出编程决策，并提供有关客户端程序员可能需要采取哪些操作才能成功重试的信息。

有关更多信息，请参阅本主题底部的“错误”部分以及 [常见错误](#)。

请求语法

```
POST /getMediaForFragmentList HTTP/1.1
```

```
Content-type: application/json
```

```
{
  "Fragments": [ "string" ],
  "StreamARN": "string",
  "StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求体

请求接受采用 JSON 格式的以下数据。

Fragments

要检索媒体的片段数量列表。您可以使用检索这些值[ListFragments](#)。

类型：字符串数组

数组成员：最少 1 个项目。最多 1000 项。

长度限制：长度下限为 1。长度上限为 128。

模式：`^[0-9]+$`

必需：是

StreamARN

要从中检索片段媒体的流的 Amazon 资源名称 (ARN)。指定此参数或[StreamName](#)参数。

类型：字符串

长度限制：长度下限为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

要从中检索片段媒体的流的名称。指定此参数或[StreamARN](#)参数。

类型：字符串

长度约束：最小长度为 1。最大长度为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
Content-Type: ContentType

Payload
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

响应将返回以下 HTTP 标头。

ContentType

所请求媒体的内容类型。

长度限制：长度下限为 1。长度上限为 128。

模式：`^[a-zA-Z0-9_\.\-]+$`

响应将以下内容作为 HTTP 正文返回。

Payload

Kinesis Video Streams 返回的有效载荷是来自指定流的一系列块。有关区块的信息，请参阅[PutMedia](#)。Kinesis Video Streams 在通话中返回 `GetMediaForFragmentList` 的区块还包括以下其他 Matroska (MKV) 标签：

- `AWS_KINESISVIDEO_FRAGMENT_NUMBER`-分块中返回的片段编号。
- `AWS_KINESISVIDEO_SERVER_SIDE_TIMESTAMP`-片段的服务器端时间戳。
- `AWS_KINESISVIDEO_PRODUCER_SIDE_TIMESTAMP`——片段的制作方时间戳。

如果发生异常，将包括以下标签：

- `AWS_KINESISVIDEO_FRAGMENT_NUMBER`-引发异常的片段的编号。
- `AWS_KINESISVIDEO_EXCEPTION_ERROR_CODE`-错误的整数代码。
- `AWS_KINESISVIDEO_EXCEPTION_MESSAGE`-异常的文字描述。

错误

有关所有操作返回的常见错误的信息，请参阅 [常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已经超过了限制。稍后再尝试拨打电话。有关限制的信息，请参阅 [Kinesis Video Streams 限制](#)。

HTTP 状态代码：400

InvalidArgumentException

指定参数超出其限制、不受支持或无法使用。

HTTP 状态代码：400

NotAuthorizedException

状态码：403，调用者无权对给定直播执行操作，或者令牌已过期。

HTTP 状态代码：401

ResourceNotFoundException

GetImages 当 Kinesis Video Streams 找不到你指定的直播时，将引发此错误。

GetHLSStreamingSessionURL 如果请求 PlaybackMode 的会话在请求的时间范围内 LIVE_REPLAY 没有片段，ON_DEMAND 或者在过去 30 秒内没有片段的流请求 PlaybackMode 的 LIVE 会话为或时，则会 GetDASHStreamingSessionURL 抛出此错误。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS SDK 中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)

- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ListFragments

服务：Amazon Kinesis Video Streams Archived Media

返回存档数据中指定流和时间戳范围内的[Fragment](#)对象列表。

列出片段最终是一致的。这意味着，即使生产者收到确认片段已保存，也可能不会立即从对的请求中返回结果。ListFragments但是，通常在不到一秒钟的时间内即可获得结果。

Note

您必须先调用 GetDataEndpoint API 才能获取终端节点。然后使用 [--endpoint-url 参数](#) 将ListFragments请求发送到此端点。

Important

如果在调用 Kinesis Video Streams 存档媒体 API 后出现错误，则除了 HTTP 状态代码和响应正文外，还会包含以下信息：

- x-amz-ErrorTypeHTTP 标头 — 除了 HTTP 状态码提供的错误类型外，还包含更具体的错误类型。
- x-amz-RequestIdHTTP 标头 — 如果你想向报告问题AWS，如果给出请求编号，支持团队可以更好地诊断问题。

HTTP 状态码和 ErrorType 标头都可用于对错误是否可重试以及在什么条件下做出编程决策，并提供有关客户端程序员可能需要采取哪些操作才能成功重试的信息。有关更多信息，请参阅本主题底部的错误部分以及[常见错误](#)。

请求语法

```
POST /listFragments HTTP/1.1
Content-type: application/json

{
  "FragmentSelector": {
    "FragmentSelectorType": "string",
    "TimestampRange": {
      "EndTimestamp": number,
```

```
    "StartTimeStamp": number
  }
},
"MaxResults": number,
"NextToken": "string",
"StreamARN": "string",
"StreamName": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求体

请求接受采用 JSON 格式的以下数据。

[FragmentSelector](#)

描述要返回的片段范围的时间戳范围和时间戳来源。

Note

只有在 API 中 NextToken 未传递时，才需要这样做。

类型：[FragmentSelector](#) 对象

必需：否

[MaxResults](#)

要返回的片段总数。如果可用片段的总数大于中指定的值 `max-results`，则输出中会提供 [ListFragments:NextToken](#)，您可以使用它来恢复分页。

默认值是 100。

类型：长整型

有效范围：最小值为 1。最大值为 1000。

必需：否

NextToken

指定从何处开始分页的令牌。这是[ListFragments:NextToken](#)来自之前截断的响应。

类型：字符串

长度限制：长度下限为 1。最大长度为 4096。

模式：`[a-zA-Z0-9+/\+=]{0,2}`

必需：否

StreamARN

要从中检索片段列表的流的 Amazon 资源名称 (ARN)。指定此参数或StreamName参数。

类型：字符串

长度限制：长度下限为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

要从中检索片段列表的流的名称。指定此参数或StreamARN参数。

类型：字符串

长度约束：最小长度为 1。最大长度为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "Fragments": [
    {
```

```
    "FragmentLengthInMilliseconds": number,
    "FragmentNumber": "string",
    "FragmentSizeInBytes": number,
    "ProducerTimestamp": number,
    "ServerTimestamp": number
  }
],
"NextToken": "string"
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回以下数据。

Fragments

流中符合选择条件的存档 [Fragment](#) 对象的列表。结果没有特定的顺序，即使是跨页面也是如此。

如果流中没有符合选择器条件的片段，则返回一个空列表。

类型：[Fragment](#) 对象数组

NextToken

如果返回的列表被截断，则该操作会返回此令牌以用于检索下一页的结果。当没有更多结果可返回 null 时，该值即为该值。

类型：字符串

长度限制：长度下限为 1。最大长度为 4096。

模式：`[a-zA-Z0-9+/\]+= {0,2}`

错误

有关所有操作返回的常见错误的信息，请参阅 [常见错误](#)。

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过限制。稍后再尝试拨打电话。有关限制的信息，请参阅 [Kinesis Video Streams 限制](#)。

HTTP 状态代码：400

InvalidArgumentException

指定参数超出其限制、不受支持或无法使用。

HTTP 状态代码：400

NotAuthorizedException

状态码：403，调用者无权对给定直播执行操作，或者令牌已过期。

HTTP 状态代码：401

ResourceNotFoundException

GetImages当 Kinesis Video Streams 找不到你指定的直播时，将引发此错误。

GetHLSStreamingSessionURL如果请求PlaybackMode的会话在请求的时间范围内LIVE_REPLAY没有片段，ON_DEMAND或者在过去 30 秒内没有片段的流请求PlaybackMode的LIVE会话为或时，则会GetDASHStreamingSessionURL抛出此错误。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS SDK 中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版 SDK](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

Amazon Kinesis Video

Amazon Kinesis 视频信号频道支持以下操作：

- [GetIceServerConfig](#)
- [SendAlexaOfferToMaster](#)

GetIceServerConfig

服务: Amazon Kinesis Video Signaling Channels

注意：在使用此 API 之前，您必须调用 `GetSignalingChannelEndpoint` API 来请求 HTTPS 终端节点。然后，您可以在 `GetIceServerConfig` API 请求中指定终端节点和区域。

获取交互式连接机构 (ICE) 服务器配置信息，包括可用于配置 WebRTC 连接的 URI、用户名和密码。ICE 组件使用此配置信息来设置 WebRTC 连接，包括使用 NAT (TURN) 中继服务器周围的中继遍历进行身份验证。

TURN 是一种用于改善 peer-to-peer 应用程序连接性的协议。通过提供基于云的中继服务，TURN 可确保即使一个或多个对等体无法直接 peer-to-peer 连接，也可以建立连接。有关更多信息，请参阅[用于访问 TURN 服务的 REST API](#)。

您可以调用此 API 来建立回退机制，以防任何一个对等体无法通过信令通道建立直接 peer-to-peer 连接。要调用此 API，您必须指定信令通道的 Amazon 资源名称 (ARN)。

请求语法

```
POST /v1/get-ice-server-config HTTP/1.1
Content-type: application/json
```

```
{
  "ChannelARN": "string",
  "ClientId": "string",
  "Service": "string",
  "Username": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

ChannelARN

用于在已配置的对等体之间进行 peer-to-peer 连接的信令信道的 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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：是

ClientId

查看者的唯一标识符。在信令信道内必须是唯一的。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

Service

指定所需的服务。当前，TURN 是唯一的有效值。

类型：字符串

有效值：TURN

必需：否

Username

要与凭证关联的可选用户 ID。

类型：字符串

长度限制：最小长度为 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：否

响应语法

```
HTTP/1.1 200
```



```
Content-type: application/json

{
  "IceServerList": [
    {
      "Password": "string",
      "Ttl": number,
      "Uris": [ "string" ],
      "Username": "string"
    }
  ]
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

[IceServerList](#)

ICE 服务器信息对象列表。

类型：[IceServer](#) 对象数组

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

ClientLimitExceededException

您的请求已被限制，因为您已超出允许的客户端呼叫上限。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

InvalidClientException

指定的客户机无效。

HTTP 状态代码：400

NotAuthorizedException

呼叫者无权执行此操作。

HTTP 状态代码：401

ResourceNotFoundException

未找到指定的资源。

HTTP 状态代码：404

SessionExpiredException

如果客户端会话已过期。客户端连接后，会话的有效期为 45 分钟。客户端应重新连接到频道才能继续发送/接收消息。

HTTP 状态代码：400

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

SendAlexaOfferToMaster

服务: Amazon Kinesis Video Signaling Channels

Note

在使用此 API 之前，您必须调用 `GetSignalingChannelEndpoint` API 来获取终端节点。然后，您可以在 `SendAlexaOfferToMaster` API 请求中指定终端节点和区域。

此 API 允许您将支持 WebRTC 的设备与 Alexa 显示设备连接起来。调用时，它会向主对等体发送 Alexa 会话描述协议 (SDP) 提议。一旦主站连接到指定的信令信道，报价就会立即交付。此 API 返回来自连接的主服务器的 SDP 答案。如果主服务器未连接到信令通道，则会发出重新传送请求，直到消息过期。

请求语法

```
POST /v1/send-alex-a-offer-to-master HTTP/1.1
Content-type: application/json

{
  "ChannelARN": "string",
  "MessagePayload": "string",
  "SenderClientId": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求正文

请求接受采用 JSON 格式的以下数据。

ChannelARN

Alexa 和主对等体通信的信令通道的亚马逊资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：是

MessagePayload

base64 编码的 SDP 提供内容。

类型：字符串

长度限制：最小长度为 0。最大长度为 10000。

模式：`[a-zA-Z0-9+/=]+`

必需：是

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 0。长度上限为 256。

模式：`[a-zA-Z0-9_.-]+`

必需：是

响应语法

```
HTTP/1.1 200
Content-type: application/json

{
  "Answer": "string"
}
```

响应元素

如果此操作成功，则该服务将会发送回 HTTP 200 响应。

服务以 JSON 格式返回的以下数据。

[Answer](#)

base64 编码的 SDP 答案内容。

类型：字符串

长度限制：最小长度为 0。最大长度为 10000。

错误

有关所有操作返回的常见错误的信息，请参阅[常见错误](#)。

ClientLimitExceededException

您的请求已被限制，因为您已超出允许的客户端呼叫上限。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

NotAuthorizedException

呼叫者无权执行此操作。

HTTP 状态代码：401

ResourceNotFoundException

未找到指定的资源。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [适用于 .NET 的 AWS SDK](#)
- [适用于 C++ 的 AWS SDK](#)

- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [适用于 PHP V3 的 AWS SDK](#)
- [适用于 Python 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

Amazon Kinesis Kinesis WebRTC

Amazon Kinesis Video WebRTC WebRTC WebRTC WebR

- [JoinStorageSession](#)

JoinStorageSession

服务：Amazon Kinesis Video WebRTC Storage

Note

在使用此 API 之前，您必须调用该 `GetSignalingChannelEndpoint` API 来请求 WebRTC 终端节点。然后，您可以在 `JoinStorageSession` API 请求中指定终端节点和区域。

作为输入频道的视频制作设备，加入正在进行的单向视频和/或多路音频 WebRTC 会话。如果该频道没有现有会话，则需要创建一个新的直播会话，并且必须提供信令频道的 Amazon 资源名称 (ARN)。

目前，对于该 `SINGLE_MASTER` 类型，视频制作设备能够将音频和视频媒体同时摄入到流中。只有视频制作设备才能加入会话并录制媒体。

Important

目前，WebRTC 摄取需要音频和视频轨道。

主参与者通过 WebRTC 协商连接后，摄取的媒体会话将存储在 Kinesis 视频流中。然后，多个观众可以通过我们的播放 API 播放实时媒体。

客户还可以将现有的 Kinesis Video Streams 功能，HLS 例如 DASH 回放、图像生成等，用于摄取的 WebRTC 媒体。

Note

目前不支持 S3 图像传输。

Note

假设只有一个视频制作设备客户端可以与该频道的会话相关联。如果多个客户端作为视频制作设备加入特定频道的会话，则最新的客户端请求优先。

其他信息

- 等性-此 API 不是等性的。
- 重试行为-这算作新的 API 调用。
- 并发呼叫-允许并发呼叫。每次调用发送一次提议。

请求语法

```
POST /joinStorageSession HTTP/1.1
Content-type: application/json

{
  "channelArn": "string"
}
```

URI 请求参数

该请求不使用任何 URI 参数。

请求体

请求接受采用 JSON 格式的以下数据。

[channelArn](#)

信令通道的 Amazon 资源名称 (ARN) 。

类型：字符串

模式：`^arn:(aws[a-zA-Z-]*):kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+$`

必需：是

响应语法

```
HTTP/1.1 200
```

响应元素

如果此操作成功，则该服务会发送回带有空 HTTP 正文的 HTTP 200 响应。

错误

有关所有操作的常见错误信息，请参阅[常见错误](#)。

AccessDeniedException

您没有执行此操作所需的权限。

HTTP 状态代码：403

ClientLimitExceededException

Kinesis Video Streams 已限制该请求，因为你已超过允许的客户端调用限制。稍后再尝试拨打电话。

HTTP 状态代码：400

InvalidArgumentException

此输入参数的值无效。

HTTP 状态代码：400

ResourceNotFoundException

未找到指定的资源。

HTTP 状态代码：404

另请参阅

有关在特定语言的 AWS SDK 中使用此 API 的更多信息，请参阅以下内容：

- [AWS 命令行界面](#)
- [AWS 适用于 .NET 的 SDK](#)
- [AWS 适用于 C++ 的 SDK](#)
- [AWS 适用于 Go 的 SDK](#)
- [AWS 适用于 Java 的 SDK V2](#)
- [AWS JavaScript V3 版软件开发工具包](#)
- [AWS 适用于 PHP 的 SDK V3](#)
- [AWS Python 软件开发工具包](#)

- [AWS 适用于 Ruby V3 的 SDK](#)

数据类型

亚马逊 Kinesis 视频流支持以下数据类型：

- [ChannelInfo](#)
- [ChannelNameCondition](#)
- [DeletionConfig](#)
- [EdgeAgentStatus](#)
- [EdgeConfig](#)
- [ImageGenerationConfiguration](#)
- [ImageGenerationDestinationConfig](#)
- [LastRecorderStatus](#)
- [LastUploaderStatus](#)
- [ListEdgeAgentConfigurationsEdgeConfig](#)
- [LocalSizeConfig](#)
- [MappedResourceConfigurationListItem](#)
- [MediaSourceConfig](#)
- [MediaStorageConfiguration](#)
- [NotificationConfiguration](#)
- [NotificationDestinationConfig](#)
- [RecorderConfig](#)
- [ResourceEndpointListItem](#)
- [ScheduleConfig](#)
- [SingleMasterChannelEndpointConfiguration](#)
- [SingleMasterConfiguration](#)
- [StreamInfo](#)
- [StreamNameCondition](#)
- [Tag](#)
- [UploaderConfig](#)

亚马逊 Kinesis Video Streams Media 支持以下数据类型：

- [StartSelector](#)

亚马逊 Kinesis Video Streams 存档媒体支持以下数据类型：

- [ClipFragmentSelector](#)
- [ClipTimestampRange](#)
- [DASHFragmentSelector](#)
- [DASHTimestampRange](#)
- [Fragment](#)
- [FragmentSelector](#)
- [HLSFragmentSelector](#)
- [HLSTimestampRange](#)
- [Image](#)
- [TimestampRange](#)

亚马逊 Kinesis 视频信号通道支持以下数据类型：

- [IceServer](#)

亚马逊 Kinesis Video WebRTC Storage 支持以下数据类型：

Amazon Kinesis Video Streams

亚马逊 Kinesis 视频流支持以下数据类型：

- [ChannelInfo](#)
- [ChannelNameCondition](#)
- [DeletionConfig](#)
- [EdgeAgentStatus](#)
- [EdgeConfig](#)
- [ImageGenerationConfiguration](#)
- [ImageGenerationDestinationConfig](#)

- [LastRecorderStatus](#)
- [LastUploaderStatus](#)
- [ListEdgeAgentConfigurationsEdgeConfig](#)
- [LocalSizeConfig](#)
- [MappedResourceConfigurationListItem](#)
- [MediaSourceConfig](#)
- [MediaStorageConfiguration](#)
- [NotificationConfiguration](#)
- [NotificationDestinationConfig](#)
- [RecorderConfig](#)
- [ResourceEndpointListItem](#)
- [ScheduleConfig](#)
- [SingleMasterChannelEndpointConfiguration](#)
- [SingleMasterConfiguration](#)
- [StreamInfo](#)
- [StreamNameCondition](#)
- [Tag](#)
- [UploaderConfig](#)

ChannelInfo

服务：Amazon Kinesis Video Streams

一种封装信道元数据和属性的结构。

目录

ChannelARN

信令通道的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

ChannelName

信令通道的名称。

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

ChannelStatus

信令通道的当前状态。

类型：字符串

有效值：CREATING | ACTIVE | UPDATING | DELETING

必需：否

ChannelType

信令通道的类型。

类型：字符串

有效值：SINGLE_MASTER | FULL_MESH

必需：否

CreationTime

创建信令通道的时间。

类型：Timestamp

必需：否

SingleMasterConfiguration

包含SINGLE_MASTER通道类型配置的结构。

类型：[SingleMasterConfiguration](#) 对象

必需：否

Version

信令通道的当前版本。

类型：字符串

长度限制：最小长度为 1。最大长度为 64。

模式：`[a-zA-Z0-9]+`

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ChannelNameCondition

服务：Amazon Kinesis Video Streams

ListSignalingChannelsAPI 的可选输入参数。如果在调用时指定了此参数ListSignalingChannels，则 API 仅返回满足中ChannelNameCondition指定条件的频道。

目录

ComparisonOperator

比较运算符。目前，您只能指定BEGINS_WITH运算符，该运算符会查找名称以给定前缀开头的信号通道。

类型：字符串

有效值：BEGINS_WITH

必需：否

ComparisonValue

一个值得比较的值。

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

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DeletionConfig

服务：Amazon Kinesis Video Streams

从 Edge Agent 中删除数据流连接所需的配置详细信息。

目录

DeleteAfterUpload

该boolean值用于指示媒体上传到 Kinesis Video Stream 云端后是否要将其标记为删除。如果将任何删除配置值设置为true，例如已达到或的限制EdgeRetentionInHours，则可以删除媒体文件。MaxLocalMediaSizeInMB

由于默认值设置为true，因此请配置上传者时间表，使媒体文件在最初上传到AWS云端之前不会被删除。

类型：布尔值

必需：否

EdgeRetentionInHours

您希望在 Edge Agent 上将数据保留在流中的小时数。保留时间的默认值为 720 小时，即 30 天。

类型：整数

有效范围：最小值为 1。最大值为 720。

必需：否

LocalSizeConfig

删除边缘配置所需的本地大小值。

类型：[LocalSizeConfig](#) 对象

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)

- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

EdgeAgentStatus

服务：Amazon Kinesis Video Streams

包含 Edge 代理的录制器和上传者作业的最新状态详细信息的对象。使用此信息来确定 Edge 代理的当前运行状况。

目录

LastRecorderStatus

直播边缘录制作业的最新状态。

类型：[LastRecorderStatus](#) 对象

必需：否

LastUploaderStatus

直播边缘到云端上传任务的最新状态。

类型：[LastUploaderStatus](#) 对象

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

EdgeConfig

服务：Amazon Kinesis Video Streams

直播边缘配置的描述，该配置将用于与 Edge Agent IoT Greengrass 组件同步。Edge Agent 组件将在您的场所的 IoT 中心设备设置上运行。

目录

HubDeviceArn

“物联网 (IoT) 物联网 (IoT) 物联网” 大事记。

类型：字符串

长度限制：最小长度为 1。长度上限为 1024。

模式：`arn:[a-z\d-]+:iot:[a-z0-9-]+:[0-9]+:thing/[a-zA-Z0-9_.-]+`

必需：是

RecorderConfig

录制器配置由本地 `MediaSourceConfig` 详细信息组成，这些详细信息用作访问摄像机上流式传输的本地媒体文件的凭据。

类型：[RecorderConfig](#) 对象

必需：是

DeletionConfig

删除配置由用于删除的保留时间 (`EdgeRetentionInHours`) 和本地大小配置 (`LocalSizeConfig`) 详细信息组成。

类型：[DeletionConfig](#) 对象

必需：否

UploaderConfig

上传器配置包含用于将录制的媒体文件从 Edge Agent 上传到 Kinesis Video Stream 的 `ScheduleExpression` 详细信息。

类型：[UploaderConfig](#) 对象

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ImageGenerationConfiguration

服务：Amazon Kinesis Video Streams

包含 KVS 图像交付所需信息的结构。如果为 null，则配置将从流中删除。

目录

DestinationConfig

包含向客户提供图像所需的信息的结构。

类型：[ImageGenerationDestinationConfig](#) 对象

必需：是

Format

可接受的图像格式。

类型：字符串

有效值：JPEG | PNG

必需：是

ImageSelectorType

用于生成图像的服务器或生产者时间戳的来源。

类型：字符串

有效值：SERVER_TIMESTAMP | PRODUCER_TIMESTAMP

必需：是

SamplingInterval

需要从流中生成图像的时间间隔（以毫秒 (ms) 为单位）。可以提供的最小值为 200 ms。如果时间戳范围小于采样间隔，则图像来自 StartTimestamp 如果可用，将退回。

类型：整数

有效范围：最小值为 3000。最大值为 20000。

必需：是

Status

表示是否ContinuousImageGenerationConfigurationsAPI 已启用或已禁用。

类型：字符串

有效值：ENABLED | DISABLED

必需：是

FormatConfig

键值对结构的列表，其中包含生成图像时可以应用的额外参数。这个FormatConfig关键是JPEGQuality，它表示用于生成图像的 JPEG 质量密钥。这个FormatConfig值接受从 1 到 100 的整数。如果值为 1，则生成的图像质量较差，压缩效果最佳。如果该值为 100，则生成质量最好、压缩程度较低的图像。如果未提供任何值，则为的默认值JPEGQuality密钥将设置为 80。

类型：字符串到字符串映射

地图条目：最大数量为 1 个项目。

有效密钥：JPEGQuality

值长度约束：最小长度为 0。长度上限为 256。

价值模式：`^[a-zA-Z_0-9]+`

必需：否

HeightPixels

与输出图像结合使用的输出图像的高度WidthPixels参数。当两者都有HeightPixels和WidthPixels提供了参数，将对图像进行拉伸以适合指定的长宽比。如果只有HeightPixels提供了参数，其原始纵横比将用于计算WidthPixels比率。如果两个参数都未提供，则将返回原始图像尺寸。

类型：整数

有效范围：最小值为 1。最大值为 2160。

必需：否

WidthPixels

与输出图像结合使用的输出图像的宽度HeightPixels参数。当两者都有WidthPixels和HeightPixels提供了参数，将对图像进行拉伸以适合指定的长宽比。如果只

有WidthPixels提供了参数，其原始纵横比将用于计算HeightPixels比率。如果两个参数都未提供，则将返回原始图像尺寸。

类型：整数

有效范围：最小值为 1。最大值为 3840。

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ImageGenerationDestinationConfig

服务：Amazon Kinesis Video Streams

包含向客户交付图像所需信息的结构。

目录

DestinationRegion

交付图像的 S3 桶的AWS区域。这DestinationRegion必须与直播所在的区域相匹配。

类型：字符串

最大长度限制：最小长度为 9。最大最大长度为 14。

模式：`^[a-z]+(-[a-z]+)?-[a-z]+-[0-9]$`

必需：是

Uri

统一资源标识符 (URI)，用于标识图像的传送位置。

类型：字符串

长度限制：最小长度为 1。最大长度为 255。

模式：`^[a-zA-Z_0-9]+:(//)?(^[^/]+)/?(^[^*]*)$`

必需：是

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

LastRecorderStatus

服务：Amazon Kinesis Video Streams

直播边缘录制作业的最新状态。

目录

JobStatusDetails

记录器作业最新状态的描述。

类型：字符串

必需：否

LastCollectedTime

上次执行记录器作业和将媒体存储到本地磁盘的时间戳。

类型：Timestamp

必需：否

LastUpdatedTime

上次更新记录器状态的时间戳。

类型：Timestamp

必需：否

RecorderStatus

最新记录器作业的状态。

类型：字符串

有效值：SUCCESS | USER_ERROR | SYSTEM_ERROR

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

LastUploaderStatus

服务：Amazon Kinesis Video Streams

直播边缘到云端上传任务的最新状态。

目录

JobStatusDetails

对上传者作业最新状态的描述。

类型：字符串

必需：否

LastCollectedTime

上次执行上传任务和将媒体收集到云端的时间戳。

类型：Timestamp

必需：否

LastUpdatedTime

上次更新上传者状态的时间戳。

类型：Timestamp

必需：否

UploaderStatus

最新上传者任务的状态。

类型：字符串

有效值：SUCCESS | USER_ERROR | SYSTEM_ERROR

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ListEdgeAgentConfigurationsEdgeConfig

服务：Amazon Kinesis Video Streams

对单个数据流边缘配置的描述。

目录

CreationTime

直播首次创建 Edge 配置时的时间戳。

类型：Timestamp

必需：否

EdgeConfig

直播边缘配置的描述，该配置将用于与 Edge Agent IoT Greengrass 组件同步。Edge Agent 组件将在您所在地的 IoT 中心设备设置上运行。

类型：[EdgeConfig](#) 对象

必需：否

FailedStatusDetails

对生成的故障状态的描述。

类型：字符串

必需：否

LastUpdatedTime

直播最后一次更新 edge 配置的时间戳。

类型：Timestamp

必需：否

StreamARN

流的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

流的名称。

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

SyncStatus

直播边缘配置的当前同步状态。

类型：字符串

有效值：SYNCING | ACKNOWLEDGED | IN_SYNC | SYNC_FAILED | DELETING | DELETE_FAILED | DELETING_ACKNOWLEDGED

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

LocalSizeConfig

服务：Amazon Kinesis Video Streams

配置详细信息包括要在 Edge Agent 上为流存储的最大媒体大小 (StrategyOnFullSize)，以及达到流的最大大小时应使用的策略 ()。MaxLocalMediaSizeInMB

目录

MaxLocalMediaSizeInMB

您要在 Edge Agent 上存储流媒体的最大总体大小。

类型：整数

有效范围：最小值为 65。最大值为 2。

必需：否

StrategyOnFullSize

达到直播MaxLocalMediaSizeInMB限制时要执行的策略。

类型：字符串

有效值：DELETE_OLDEST_MEDIA | DENY_NEW_MEDIA

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

MappedResourceConfigurationListItem

服务：Amazon Kinesis Video Streams

一种封装或包含媒体存储配置属性的结构。

目录

ARN

请记住 Kinesis 流资源的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

Type

kinesis 视频流的关联资源的类型。

类型：字符串

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

MediaSourceConfig

服务：Amazon Kinesis Video Streams

配置详细信息包括访问流式传输到摄像机的媒体文件所需 (`MediaUriSecretArn` 和 `MediaUriType`) 的凭据。

目录

MediaUriSecretArn

S AWS secrets Manager ARN，用于存储摄像机的用户名和密码，或本地媒体文件位置。

类型：字符串

长度限制：最小长度为 20。最大长度为 2048。

模式：`arn:[a-z\d-]+:secretsmanager:[a-z0-9-]+:[0-9]+:secret:[a-zA-Z0-9_.-]+`

必需：是

MediaUriType

统一资源标识符 (URI) 类型。该 `FILE_URI` 值可用于流式传输本地媒体文件。

Note

预览仅支持 `RTSP_URI` 媒体源 URI 格式。

类型：字符串

有效值：`RTSP_URI` | `FILE_URI`

必需：是

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)

- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

MediaStorageConfiguration

服务: Amazon Kinesis Video Streams

一种封装或包含媒体存储配置属性的结构。

- 如果启用 `StorageStatus`，则数据将存储在 `StreamARN` 提供的中。为了让 WebRTC Ingestion 正常运行，直播必须启用数据保留。
- 如果禁用，`StorageStatus` 则不会存储任何数据，也不需要该 `StreamARN` 参数。

目录

Status

媒体存储配置的状态。

类型：字符串

有效值：ENABLED | DISABLED

必需：是

StreamARN

流的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)

- [适用于 Ruby V3 的 AWS SDK](#)

NotificationConfiguration

服务：Amazon Kinesis Video Streams

包含 KVS 图像传送通知信息的结构。如果此参数为空，则配置删除流。

目录

DestinationConfig

向客户发送通知所需的目的地信息。

类型：[NotificationDestinationConfig](#) 对象

必需：是

Status

指示是启用还是禁用的通知配置。

类型：字符串

有效值：ENABLED | DISABLED

必需：是

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

NotificationDestinationConfig

服务：Amazon Kinesis Video Streams

包含向客户发送通知所需信息的结构。

目录

Uri

统一资源标识符 (URI)，用于标识图像的传送位置。

类型：字符串

长度限制：最小长度为 1。最大长度为 255。

模式：`^[a-zA-Z_0-9]+:(//)?([/]+)/?([^*]*)$`

必需：是

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

RecorderConfig

服务：Amazon Kinesis Video Streams

录像机配置由本地MediaSourceConfig详细信息组成，这些详细信息用作访问摄像机上流式传输的本地媒体文件的凭据。

目录

MediaSourceConfig

配置详细信息，包括访问流式传输到摄像机的媒体文件所需的凭据（MediaUriSecretArn和MediaUriType）。

类型：[MediaSourceConfig](#) 对象

必需：是

ScheduleConfig

由ScheduleExpression和DurationInMinutes详细信息组成的配置，用于指定从摄像机或本地媒体文件录制到 Edge Agent 的时间安排。如果未提供该ScheduleExpression属性，则 Edge Agent 将始终设置为录制模式。

类型：[ScheduleConfig](#) 对象

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ResourceEndpointListItem

服务：Amazon Kinesis Video Streams

描述由 GetSignalingChannelEndpoint API 返回的信号通道端点的对象。

媒体服务器端点将对应于WEBRTC协议。

目录

Protocol

GetSignalingChannelEndpointAPI 返回的信号通道的协议。

类型：字符串

有效值：WSS | HTTPS | WEBRTC

必需：否

ResourceEndpoint

GetSignalingChannelEndpointAPI 返回的信号通道的终端节点。

类型：字符串

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ScheduleConfig

服务：Amazon Kinesis Video Streams

此 API 使您能够指定摄像机或本地媒体文件应在 Edge Agent 上录制的时长。ScheduleConfig 由 ScheduleExpression 和 DurationInMinutes 属性组成。

如果 ScheduleConfig 未提供 RecorderConfig，则 Edge Agent 将始终设置为录制模式。

如果 ScheduleConfig 未提供 UploaderConfig，则 Edge Agent 将定期上传（每 1 小时）。

目录

DurationInSeconds

录制媒体的总时长。如果提供了该 ScheduleExpression 属性，则还应指定该 DurationInSeconds 属性。

类型：整数

有效范围：最小值为 60。最大值为 36。

必需：是

ScheduleExpression

Quartz cron 表达式，用于安排将作业从摄像机或本地媒体文件录制到 Edge Agent。如果未提供 ScheduleExpression，则 Edge Agent 将始终设置为录制模式。RecorderConfig

有关 Quartz 的更多信息，请参阅 [Cron Trigger 教程](#) 页面以了解有效表达式及其用法。

类型：字符串

长度限制：最小长度为 11。长度上限为 100。

模式：`[^\n]{11,100}`

必需：是

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)

- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

SingleMasterChannelEndpointConfiguration

服务：Amazon Kinesis Video Streams

一个包含SINGLE_MASTER频道类型的端点配置的对象。

目录

Protocols

此属性用于确定通过此SINGLE_MASTER信号信道进行通信的性质。如果WSS已指定，则此 API 将返回 websocket 端点。如果HTTPS已指定，则此 API 将返回一个HTTPS端点。

类型：字符串数组

数组成员：最少 1 项。最多 5 项。

有效值：WSS | HTTPS | WEBRTC

必需：否

Role

此属性用于确定此SINGLE_MASTER信号通道中的消息权限。如果MASTER已指定，此 API 将返回一个端点，客户端可以使用该端点从该信号通道上的任何观众那里接收报价并向其发送答案。VIEWER如果指定，此 API 将返回一个端点，客户端只能使用该端点向该信号通道上的另一个MASTER客户端发送报价。

类型：字符串

有效值：MASTER | VIEWER

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

SingleMasterConfiguration

服务：Amazon Kinesis Video Streams

包含控制面板权限的SINGLE_MASTER结构。

目录

MessageTtlSeconds

信令通道在丢弃未送达的消息之前保留这些消息的时间段。

类型：整数

有效范围：最小值为 5。最大值为 120。

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

StreamInfo

服务：Amazon Kinesis Video Streams

权限以创建 Kinesis 视频流。

目录

CreationTime

指明直播创建时间的时间戳。

类型：Timestamp

必需：否

DataRetentionInHours

流保留数据的时间（以小时为单位）。

类型：整数

有效范围：最小值为 0。

必需：否

DeviceName

与流关联的设备的名称。

类型：字符串

长度限制：最小长度为 1。最大长度为 128。

模式：[a-zA-Z0-9_.-]+

必需：否

KmsKeyId

Kinesis Video Streams 用于对流上数据进行加密的 AWS Key Management Service (AWS KMS) 密钥 ID。

类型：字符串

长度限制：最小长度为 1。最大长度为 2048。

模式：.+

必需：否

MediaType

流的 MediaType。

类型：字符串

长度限制：最小长度为 1。最大长度为 128。

模式：`[\w\-\.\+]+/[\w\-\.\+]+(,[\w\-\.\+]+/[\w\-\.\+]+)*`

必需：否

Status

视频流。

类型：字符串

有效值：CREATING | ACTIVE | UPDATING | DELETING

必需：否

StreamARN

流的 Amazon 资源名称 (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:[a-z\d-]+:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

必需：否

StreamName

流的名称。

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

Version

流的版本。

类型：字符串

长度限制：最小长度为 1。最大长度为 64。

模式：`[a-zA-Z0-9]+`

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

StreamNameCondition

服务：Amazon Kinesis Video Streams

指定列出直播时必须满足的流才能返回的条件（参见 `ListStreams` API）。条件具有比较运算和值。目前，您只能指定 `BEGINS_WITH` 运算符，该运算符会查找名称以给定前缀开头的流。

目录

ComparisonOperator

比较运算符。目前，您只能指定 `BEGINS_WITH` 运算符，该运算符会查找名称以给定前缀开头的流。

类型：字符串

有效值：`BEGINS_WITH`

必需：否

ComparisonValue

一个值得比较的值。

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

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

Tag

服务：Amazon Kinesis Video Streams

与指定信号通道关联的一个键/值对。

目录

Key

与指定信令信道关联的标签的密钥。

类型：字符串

长度限制：最小长度为 1。最大长度为 128。

模式：`^([\p{L}\p{Z}\p{N}_.:/=+\-@]*)$`

必需：是

Value

与指定信令信道关联的标签的值。

类型：字符串

长度约束：最小长度为 0。长度上限为 256。

模式：`[\p{L}\p{Z}\p{N}_.:/=+\-@]*`

必需：是

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

UploaderConfig

服务：Amazon Kinesis Video Streams

由ScheduleExpression和DurationInMinutes详细信息组成的配置，用于指定从摄像机或本地媒体文件录制到 Edge Agent 的时间安排。如果ScheduleConfig未提供UploaderConfig，则 Edge Agent 将定期上传（每 1 小时）。

目录

ScheduleConfig

由ScheduleExpression和DurationInMinutes详细信息组成的配置，用于指定从摄像机或本地媒体文件录制到 Edge Agent 的时间安排。如果此ScheduleConfig处未提供UploaderConfig，则 Edge Agent 将定期上传（每 1 小时）。

类型：[ScheduleConfig](#) 对象

必需：是

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

Amazon Kinesis Video Streams

Amazon Kinesis Video Streams 媒体支持以下数据类型：

- [StartSelector](#)

StartSelector

服务：Amazon Kinesis Video Streams Media

识别 Kinesis 视频流中您希望 GetMedia API 开始返回媒体数据的区块。您可以通过以下几种方式设置 Condition 子句：

- 选择最新（或最旧）的区块。
- 确定特定的区块。您可以通过提供片段号或时间戳（服务器或生产者）来识别特定的块。
- 每个区块的元数据都包含一个作为 Matroska (MKV) 标签 () 的延续令牌。AWS_KINESISVIDEO_CONTINUATION_TOKEN 如果您之前的 GetMedia 请求终止，则可以在下一个 GetMedia 请求中使用此标签值。然后，API 开始返回从上一个 API 结束的地方开始的区块。

目录

StartSelectorType

识别 Kinesis 视频流中您要从中开始获取数据的片段。

- 现在-从直播中的最新区块开始。
- 最早-从直播中最早的可用区块开始。
- FRAGMENT_NUMBER-从特定片段之后的区块开始。还必须指定 AfterFragmentNumber 参数。
- PRODUCER_TIMESTAMP 或 SERVER_TIMESTAMP-从包含具有指定生产者或服务器时间戳的片段的区块开始。您可以通过添加 StartTimestamp 来指定时间戳。
- CONTINUATION_TOKEN-使用指定的延续标记读取。

Note

如果您选择“现在”、“最早”或“CONTINUATION_TOKEN”作为 startSelectorType，则无需在中提供任何其他信息。startSelector

类型：字符串

有效值：FRAGMENT_NUMBER | SERVER_TIMESTAMP | PRODUCER_TIMESTAMP | NOW | EARLIEST | CONTINUATION_TOKEN

必需：是

AfterFragmentNumber

指定您希望 GetMedia API 从哪里开始返回片段的片段编号。

类型：字符串

长度限制：最小长度为 1。最大长度为 128。

模式：`^[0-9]+$`

必需：否

ContinuationToken

Kinesis Video Streams 在之前的 GetMedia 响应中返回的延续令牌。然后，GetMediaAPI 从延续令牌标识的区块开始。

类型：字符串

长度限制：最小长度为 1。最大长度为 128。

模式：`^[a-zA-Z0-9_\.\\-]+$`

必需：否

StartTimestamp

时间戳值。如果您选择 PRODUCER_TIMESTAMP 或 SERVER_TIMESTAMP 作为必填值。startSelectorType 然后，GetMediaAPI 从包含具有指定时间戳的片段的区块开始。

类型：Timestamp

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

Amazon Kinesis Video Streams 存档媒体

Amazon Kinesis Video Streams 存档媒体支持以下数据类型：

- [ClipFragmentSelector](#)
- [ClipTimestampRange](#)
- [DASHFragmentSelector](#)
- [DASHTimestampRange](#)
- [Fragment](#)
- [FragmentSelector](#)
- [HLSFragmentSelector](#)
- [HLSTimestampRange](#)
- [Image](#)
- [TimestampRange](#)

ClipFragmentSelector

服务：Amazon Kinesis Video Streams Archived Media

描述一系列片段的时间戳范围和时间戳来源。

对具有重复生产者时间戳的片段进行重复数据消除。这意味着，如果制作者正在生成一个片段流，其生产者时间戳与实际时钟时间大致相等，则该片段将包含所请求的时间戳范围内的所有片段。如果某些片段是在相同的时间范围和截然不同的时间点内摄取的，则只返回最早摄入的片段集合。

目录

FragmentSelectorType

要使用的时间戳的来源（服务器或生产者）。

类型：字符串

有效值：PRODUCER_TIMESTAMP | SERVER_TIMESTAMP

必需：是

TimestampRange

要返回的时间戳范围。

类型：[ClipTimestampRange](#) 对象

必需：是

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

ClipTimestampRange

服务：Amazon Kinesis Video Streams Archived Media

返回片段的时间戳范围。

目录

EndTimeStamp

所请求媒体的时间戳范围的结束。

此值必须位于指定值的 24 小时内StartTimeStamp，并且必须晚于该StartTimeStamp值。如果FragmentSelectorType for 请求是SERVER_TIMESTAMP，则此值必须是过去的值。

此范围包含起始值。将与片段EndTimeStamp的（起始）时间戳进行比较。在该值之前开始并在该EndTimeStamp值之后继续超过该值的片段包含在会话中。

类型：Timestamp

必需：是

StartTimeStamp

返回片段的时间戳范围内的起始时间戳。

只有恰好在或之后开始的片段才StartTimeStamp会包含在会话中。

在会话中不包括在此之前开始StartTimeStamp并持续过后的片段。如

果FragmentSelectorType是SERVER_TIMESTAMP，则StartTimeStamp必须晚于流头。

类型：Timestamp

必需：是

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DASHFragmentSelector

服务：Amazon Kinesis Video Streams Archived Media

包含所请求媒体的时间戳范围和时间戳的来源。

目录

FragmentSelectorType

所请求媒体的时间戳来源。

当设置FragmentSelectorType为 PRODUCER_TIMESTAMP [getDash StreamingSession URL: PlaybackMode](#) 为ON_DEMAND或时LIVE_REPLAY，媒体播放列表中包含制作者时间戳的第一个片段包含TimestampRange在指定[FragmentSelector](#):内。此外，还包括在第一个片段（直到TimestampRange [getDash StreamingSession URL: MaxManifestFragmentResults](#) 值）之后立即采集的带有生产者时间戳的片段。

对具有重复生产者时间戳的片段进行重复数据消除。这意味着，如果生产者正在生成一个片段流，其生产者时间戳与真实时钟时间大致相等，则 MPEG-DASH 清单将包含请求的时间戳范围内的所有片段。如果某些片段是在相同的时间范围和截然不同的时间点内摄取的，则只返回最早摄入的片段集合。

如果设置FragmentSelectorType为 [getDash StreamingSession URL: PlaybackMode](#) 是LIVE，则在 MP4 片段和重复数据删除中使用生产者时间戳。PRODUCER_TIMESTAMP但是，最近根据服务器时间戳采集的片段包含在 MPEG-DASH 清单中。这意味着，即使过去摄取的片段具有带有现在值的制作者时间戳，它们也不会包含在 HLS 媒体播放列表中。

默认为 SERVER_TIMESTAMP。

类型：字符串

有效值：PRODUCER_TIMESTAMP | SERVER_TIMESTAMP

必需：否

TimestampRange

所请求媒体的时间戳范围的开始和结束。

如果PlaybackType是，则不应存在此值LIVE。

类型：[DASHTimestampRange](#) 对象

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

DASHTimestampRange

服务：Amazon Kinesis Video Streams Archived Media

所请求媒体的时间戳范围的开始和结束。

如果PlaybackType是，则不应存在此值LIVE。

中的值DASHTimestampRange是包容性的。恰好在开始时间或之后开始的片段包含在会话中。在开始时间之前开始并在开始时间之后继续延续的片段不包含在会话中。

目录

EndTimeStamp

所请求媒体的时间戳范围的结束。此值必须位于指定值的 24 小时内StartTimestamp，并且必须晚于该StartTimestamp值。

如果FragmentSelectorType for 请求是SERVER_TIMESTAMP，则此值必须是过去的值。

该EndTimeStamp值对于模式是必填的，但对于ON_DEMANDLIVE_REPLAY模式是可选的。如果未设置为LIVE_REPLAY模式，则会话将继续包含新摄入的片段，直到会话过期。EndTimeStamp

Note

此范围包含起始值。将与片段EndTimeStamp的（起始）时间戳进行比较。在该值之前开始并在该EndTimeStamp值之后继续超过该值的片段包含在会话中。

类型：Timestamp

必需：否

StartTimestamp

所请求媒体的时间戳范围的起始值。

如果指定了该DASHTimestampRange值，则该StartTimestamp值为必填值。

只有恰好在或之后开始的片段才StartTimestamp会包含在会话中。

在会话中不包括在此之前开始StartTimestamp并持续过后的片段。如

果FragmentSelectorType是SERVER_TIMESTAMP，则StartTimestamp必须晚于流头。

类型：Timestamp

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

Fragment

服务：Amazon Kinesis Video Streams Archived Media

表示一段视频或其他以时间分隔的数据。

内容

FragmentLengthInMilliseconds

与片段关联的播放时长或其他时间值。

类型：长整型

必需：否

FragmentNumber

片段的唯一标识符。该值会根据摄取顺序单调增加。

类型：字符串

长度限制：长度下限为 1。长度上限为 128。

模式：`^[0-9]+$`

必需：否

FragmentSizeInBytes

片段的总大小，包括有关片段和所含媒体数据的信息。

类型：长整型

必需：否

ProducerTimestamp

来自生产者对应于片段的时间戳，以毫秒为单位。

类型：时间戳

必需：否

ServerTimestamp

AWS服务器上与片段对应的时间戳，以毫秒为单位。

类型：时间戳

必需：否

另请参阅

有关在特定语言的 AWS SDK 中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

FragmentSelector

服务：Amazon Kinesis Video Streams Archived Media

描述一系列片段的时间戳范围和时间戳来源。

只有开始时间大于等于给定开始时间并且小于等于结束时间的片段才会被返回。例如，如果一个流包含具有以下开始时间戳的片段：

- 00:00:00
- 00:00:02
- 00:00:04
- 00:00:06

开始时间为 00:00:01 且结束时间为 00:00:04 的片段选择器范围将返回开始时间为 00:00:02 和 00:00:04 的片段。

目录

FragmentSelectorType

要使用的时间戳的来源（服务器或生产者）。

类型：字符串

有效值：PRODUCER_TIMESTAMP | SERVER_TIMESTAMP

必需：是

TimestampRange

要返回的时间戳范围。

类型：[TimestampRange](#) 对象

必需：是

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)

- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

HLSFragmentSelector

服务：Amazon Kinesis Video Streams Archived Media

包含所请求媒体的时间戳范围和时间戳的来源。

目录

FragmentSelectorType

所请求媒体的时间戳来源。

当设置FragmentSelectorType为PRODUCER_TIMESTAMP且 [getHLS StreamingSession URL: PlaybackMode](#) 为ON_DEMAND或时LIVE_REPLAY，媒体播放列表中包含制作者时间戳的第一个片段包含在指TimestampRange定[FragmentSelector](#):内。此外，还包括生产者时间戳位于第一个片段之后立即TimestampRange采集的片段（直到 [getHLS StreamingSession URL: v MaxMediaPlaylistFragmentResults alue](#) ）。

对具有重复生产者时间戳的片段进行重复数据删除。这意味着，如果制作者正在生成带有制作者时间戳与真实时钟时间大致相等的片段流，则 HLS 媒体播放列表将包含所请求的时间戳范围内的所有片段。如果某些片段是在相同的时间范围和截然不同的时间点内摄取的，则只返回最早摄入的片段集合。

如果设置FragmentSelectorType为 PRODUCER_TIMESTAMP [getH StreamingSession L URL: PlaybackMode](#) 为LIVE，则在 MP4 片段和重复数据删除中使用生产者时间戳。但是最近根据服务器时间戳采集的片段包含在HLS媒体播放列表中。这意味着，即使过去摄取的片段具有带有现在值的制作者时间戳，它们也不会包含在 HLS 媒体播放列表中。

默认为 SERVER_TIMESTAMP。

类型：字符串

有效值：PRODUCER_TIMESTAMP | SERVER_TIMESTAMP

必需：否

TimestampRange

所请求媒体的时间戳范围的开始和结束。

如果PlaybackType是，则不应存在此值LIVE。

类型：[HLSTimestampRange](#) 对象

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

HLSTimestampRange

服务：Amazon Kinesis Video Streams Archived Media

所请求媒体的时间戳范围的开始和结束。

如果PlaybackType是，则不应存在此值LIVE。

目录

EndTimeStamp

所请求媒体的时间戳范围的结束。此值必须位于指定值的 24 小时内StartTimeStamp，并且必须晚于该StartTimeStamp值。

如果FragmentSelectorType for 请求是SERVER_TIMESTAMP，则此值必须是过去的值。

该EndTimeStamp值对于模式是必填的，但对于ON_DEMANDLIVE_REPLAY模式是可选的。如果未设置为LIVE_REPLAY模式，则会话将继续包含新摄入的片段，直到会话过期。EndTimeStamp

Note

此范围包含起始值。将与片段EndTimeStamp的（开始）时间戳进行比较。在该值之前开始并继续超过该EndTimeStamp值的片段包含在会话中。

类型：Timestamp

必需：否

StartTimeStamp

所请求媒体的时间戳范围的起始值。

如果指定了该HLSTimestampRange值，则该StartTimeStamp值为必填值。

只有恰好在或之后开始的片段才StartTimeStamp会包含在会话中。

在会话中不包括在此之前开始StartTimeStamp并持续过后的片段。如

果FragmentSelectorType是SERVER_TIMESTAMP，则StartTimeStamp必须晚于流头。

类型：Timestamp

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

Image

服务：Amazon Kinesis Video Streams Archived Media

包含TimestampError、和的结构ImageContent。

目录

Error

由于不可尝试的错误而无法提取所提供时间戳的图像时显示的错误消息。在以下情况下，将返回错误：

- 指定的媒体不存在Timestamp。
- 指定时间的媒体不允许提取图像。在这种情况下，媒体仅为音频，或者收录了错误的媒体。

类型：字符串

有效值：NO_MEDIA | MEDIA_ERROR

必需：否

ImageContent

采用 Base64 编码的Image对象的属性。

类型：字符串

长度限制：最小长度为 1。最大长度上限为 10。

必需：否

TimeStamp

用于从视频流中提取图像的Image对象的属性。此字段用于管理图像上的空白或更好地了解分页窗口。

类型：Timestamp

必需：否

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

TimestampRange

服务：Amazon Kinesis Video Streams Archived Media

返回片段的时间戳范围。

目录

EndTimeStamp

返回片段的时间戳范围内的结束时间戳。

类型：Timestamp

必需：是

StartTimeStamp

返回片段的时间戳范围内的起始时间戳。

类型：Timestamp

必需：是

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

Amazon Kinesis Video Streams

Amazon Video Streams

- [IceServer](#)

IceServer

服务：Amazon Kinesis Video Signaling Channels

ICE 服务器连接数据的结构。

目录

Password

登录 ICE 服务器的密码。

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

Ttl

以秒为单位的时间长度，在此期间内，和密码是否有效。

类型：整数

有效范围：最小值为 30。最大值为 86400。

必需：否

Uris

一个 URI 数组，采用 [I-D 中指定的形式](#)。 [petithuguenin-behave-turn-uris](#)规格。这些 URI 提供可用于访问 TURN 服务器的不同地址和/或协议。

类型：字符串数组

长度限制：最小长度为 1。长度上限为 256。

必需：否

Username

用于登录 ICE 服务器的用户名。

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

另请参阅

有关在特定语言的 AWS 软件开发工具包中使用此 API 的更多信息，请参阅以下内容：

- [适用于 C++ 的 AWS SDK](#)
- [适用于 Go 的 AWS SDK](#)
- [适用于 Java V2 的 AWS SDK](#)
- [适用于 Ruby V3 的 AWS SDK](#)

Amazon Kinesis Video Streams

Amazon Kinesis Video Video Streams Video Streams Video

常见错误

本部分列出了所有 AWS 服务的常见 API 操作错误。对于特定于此服务的 API 操作的错误，请参阅该 API 操作的主题。

AccessDeniedException

您没有足够的访问权限，无法执行该操作。

HTTP 状态代码：400

IncompleteSignature

请求签名不符合 AWS 标准。

HTTP 状态代码：400

InternalFailure

由于未知错误、异常或故障，请求处理失败。

HTTP 状态代码：500

InvalidAction

所请求的操作无效。验证操作是否已正确键入。

HTTP 状态代码：400

InvalidClientId

在我们的记录中没有所提供的 X.509 证书或 AWS 访问密钥 ID。

HTTP 状态代码：403

NotAuthorized

您无权执行此操作。

HTTP 状态代码：400

OptInRequired

AWS 访问密钥 ID 需要订阅服务。

HTTP 状态代码：403

RequestExpired

请求到达服务的时间超过请求上的日期戳或请求到期日期 (如针对预签名 URL) 15 分钟，或者请求上的日期戳离到期还有 15 分钟以上。

HTTP 状态代码：400

ServiceUnavailable

由于服务器发生临时故障而导致请求失败。

HTTP 状态代码：503

ThrottlingException

由于请求限制而导致请求被拒绝。

HTTP 状态代码：400

ValidationError

输入未能满足 AWS 服务指定的约束。

HTTP 状态代码：400

常见参数

以下列表包含所有操作用于使用查询字符串对 Signature Version 4 请求进行签名的参数。任何特定于操作的参数都列在该操作的主题中。有关 Signature Version 4 的更多信息，请参阅《IAM 用户指南》中的[签署 AWS API 请求](#)。

Action

要执行的操作。

类型：字符串。

必需：是

Version

编写请求所针对的 API 版本，格式为 YYYY-MM-DD。

类型：字符串。

必需：是

X-Amz-Algorithm

您用于创建请求签名的哈希算法。

条件：当您在查询字符串中而不是 HTTP 授权标头中包括身份验证信息时，请指定此参数。

类型：字符串

有效值：AWS4-HMAC-SHA256

必需：条件

X-Amz-Credential

凭证范围值，该值是一个字符串，其中包含您的访问密钥、日期、您要定位的区域、您请求的服务以及终止字符串（“aws4_request”）。值采用以下格式表示：access_key/YYYYMMDD/region/service/aws4_request。

有关更多信息，请参阅《IAM 用户指南》中的[创建已签名的 AWS API 请求](#)。

条件：当您在查询字符串中而不是 HTTP 授权标头中包括身份验证信息时，请指定此参数。

类型：字符串

必需：条件

X-Amz-Date

用于创建签名的日期。格式必须为 ISO 8601 基本格式 (YYYYMMDD'T'HHMMSS'Z')。例如，以下日期时间是有效的 X-Amz-Date 值：20120325T120000Z。

条件：X-Amz-Date 对于所有请求都是可选的；它可以用于覆盖对请求签名所使用的日期。如果以 ISO 8601 基本格式指定 Date 标头，则不需要 X-Amz-Date。使用 X-Amz-Date 时，它始终会覆盖 Date 标头的值。有关更多信息，请参阅《IAM 用户指南》中的 [AWS API 请求签名的元素](#)。

类型：字符串

必需：条件

X-Amz-Security-Token

通过调用 AWS Security Token Service (AWS STS) 获得的临时安全令牌。有关支持来自 AWS STS 的临时安全凭证的服务列表，请参阅《IAM 用户指南》中的 [使用 IAM 的 AWS 服务](#)。

条件：如果您使用来自 AWS STS 的临时安全凭证，则必须包含安全令牌。

类型：字符串

必需：条件

X-Amz-Signature

指定从要签名的字符串和派生的签名密钥计算的十六进制编码签名。

条件：当您在查询字符串中而不是 HTTP 授权标头中包括身份验证信息时，请指定此参数。

类型：字符串

必需：条件

X-Amz-SignedHeaders

指定作为规范请求的一部分包含的所有 HTTP 标头。有关指定已签名标头的更多信息，请参阅《IAM 用户指南》中的 [创建已签名的 AWS API 请求](#)。

条件：当您在查询字符串中而不是 HTTP 授权标头中包括身份验证信息时，请指定此参数。

类型：字符串

必需：条件

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。