Amazon Kinesis Video Streams

개발자 안내서

Amazon Kinesis Video Streams: 개발자 안내서

Table of Contents

Amazon Kinesis 비디오 스트림란 무엇입니까?

Amazon Kinesis 비디오 스트림는 라이브 비디오를 디바이스에서 AWS 클라우드로 스트리밍하거나 실시간 비디오 처리 또는 배치 중심 비디오 분석을 위한 애플리케이션을 빌드하는 데 사용할 수 있는 완전 관리형 AWS 서비스입니다.

Kinesis 비디오 스트림는 그저 비디오 데이터용 스토리지만은 아닙니다. 이것을 사용하여 클라우드에서 비디오 스트림을 수신하여 볼 수 있습니다. AWS Management Console에서 라이브 스트림을 모니터링하거나 라이브 비디오를 표시하기 위해 Kinesis 비디오 스트림 API 라이브러리를 사용하는 자체 모니터링 애플리케이션을 개발할 수 있습니다.

Kinesis 비디오 스트림를 사용하여 스마트폰, 보안 카메라, 웹캠, 차량 내장 카메라, 드론 등을 포함한 수많은 소스로부터 방대한 양의 라이브 비디오 데이터를 캡처할 수 있습니다. 오디오 데이터, 열 화상, 깊이 데이터, RADAR 데이터 등과 같은 비영상 시계열 데이터도 전송할 수 있습니다. 라이브 비디오가 이들 소스로부터 Kinesis 비디오 스트림으로 스트리밍되기 때문에 지연 시간이 짧은 처리에 대해 실시간으로 데이터에 프레임별로 액세스할 수 있는 애플리케이션을 빌드할 수 있습니다. Kinesis 비디오 스트림는 소스와 무관합니다. 따라서 GStreamer (p. 105) 라이브러리를 사용하여 컴퓨터의 웹캠에서나 RTSP를 사용하여 네트워크의 카메라에서나 비디오를 스트리밍할 수 있습니다.

미디어 데이터를 정해진 보존 기간 동안 안정적으로 저장할 수 있도록 Kinesis 비디오 스트림을 구성할 수도 있습니다. Kinesis 비디오 스트림는 이런 데이터를 자동으로 저장하고 저장 중에 암호화합니다. 또한 Kinesis 비디오 스트림는 생산자 타임스탬프와 수집 타임스탬프를 기준으로 저장된 데이터에 대해 타임 인덱싱을 합니다. 주기적으로 비디오 데이터에 대한 배치 처리를 수행하는 애플리케이션을 빌드하거나 다양한 사용 사례로 역사적 데이터에 대한 임시 액세스를 요구하는 애플리케이션을 생성할 수 있습니다.

실시간이든 배치 중심이든 사용자 지정 애플리케이션이 Amazon EC2 인스턴스를 실행할 수 있습니다. 이러한 애플리케이션은 오픈 소스 딥러닝 알고리즘을 사용하여 데이터를 처리하거나 Kinesis 비디오 스트림와 통합되는 타사 애플리케이션을 사용할 수 있습니다.

Kinesis 비디오 스트림 사용의 이점은 다음과 같습니다.

- 수많은 디바이스에 대한 연결 및 스트리밍 - Kinesis 비디오 스트림를 사용하면 스마트폰, 드론, 대시 캠 등과 같은 수많은 디바이스에 연결하여 그로부터 비디오, 오디오 및 기타 형식의 데이터를 스트리밍할 수 있습니다. Kinesis 비디오 스트림 생산자 라이브러리를 사용하여 디바이스를 구성하고, 실시간으로 또는 사후 미디어 업로드 시에 안정적으로 스트리밍할 수 있습니다.
- 안정적인 데이터 저장, 암호화 및 인덱싱- 미디어 데이터를 정해진 보존 기간 동안 안정적으로 미디어 데이터를 저장할 수 있도록 Kinesis 비디오 스트림을 구성할 수 있습니다. 또한 Kinesis 비디오 스트림는 생산자 생성 타임스탬프 또는 서비스 측 타임스탬프를 기준으로 저장된 데이터에 대해 인덱스를 생성합니다. 애플리케이션이 타임 인덱스를 사용하여 스트림에서 지정된 데이터를 쉽게 검색할 수 있습니다.
- 인프라 대신 애플리케이션 관리에 집중 - Kinesis 비디오 스트림는 서버 없이 작동하기 때문에 설정 또는 관리할 인프라가 없습니다. 데이터 스트림 및 소비 애플리케이션 수가 증가하거나 축소함에 따라 기반 인프라의 배포, 구성 또는 탄력적 규모 조정에 대해 걱정할 필요가 없습니다. Kinesis 비디오 스트림는 스트림 관리에 필요한 모든 관리 및 유지를 자동으로 수행하기 때문에 IT 부서 직원들이 인프라 대신 애플리케이션에 집중할 수 있습니다.
- 데이터 스트림 기반 애플리케이션 실시간 빌드 및 배치 - Kinesis 비디오 스트림를 사용하여 라이브 데이터 스트림에서 작동하는 사용자 지정 실시간 애플리케이션을 빌드하고, 엄격한 지연 시간 요건 없이 안정적으로 유지되는 데이터에서 작동하는 배치 혹은 임시 애플리케이션을 생성할 수 있습니다. 오픈 소스(Apache MXNet, OpenCV), 자체 혹은 타사 솔루션과 같은 사용자 지정 애플리케이션을 AWS Marketplace를 통해 빌드, 배포 및 관리하여 스트림을 처리하고 분석할 수 있습니다. Kinesis 비디오 스트림 `Get` API를 사용하면 데이터를 실시간 혹은 배치 중심으로 처리하는 복수의 동시 실행 애플리케이션을 빌드할 수 있습니다.

- 보안성이 보다 뛰어난 데이터 스트리밍 - Kinesis 비디오 스트림는 데이터가 서비스를 통과하고 서비스가 데이터를 유지할 때 모든 데이터를 암호화합니다. Kinesis 비디오 스트림는 디바이스로부터 스트리밍되는 데이터에 대해 TLS(전송 계층 보안) 기반 암호화를 시행하고, AWS Key Management Service (AWS KMS)를 사용하여 저장 중에 있는 모든 데이터를 암호화합니다. 또한 AWS Identity and Access Management(IAM)를 사용하여 데이터에 대한 액세스를 관리할 수 있습니다.
- 종량제 요금 - 자세한 내용은 AWS 요금을 참조하십시오.

Kinesis 비디오 스트림는 다양한 업종 상황에 적합합니다. 예:

- 스마트 시티 이니셔티브 - 교통 카메라의 비디오 스트림을 Kinesis 비디오 스트림으로 푸시하고 실시간 교통 상황을 추적할 수 있는 소비자 애플리케이션을 작성합니다. 역사적 데이터에 대한 배치 처리 작업을 통해 신규 건축에서 기인하는 변화나 교통 경로 변화를 파악할 수도 있습니다.
- 가계 부문 - 가정의 보안 카메라, 육아 모니터 및 가전 기기에 내장된 기타 카메라를 연결하여 소비자의 안전을 도모하고 생활을 보다 생산적이고 유쾌하게 만드는 환경을 구축할 수 있습니다.
- 소매점의 지능형 기능 - 매장 내 여러 카메라로부터 영상을 캡처하여 고객 수 계수를 자동화하고, 매장 열지도를 생성하여 매장 내 소비자 활동을 파악하고, 레이아웃을 최적화하고 디스플레이를 효과적으로 배치하고, 소비자 만족도를 향상하는 데 도움이 되는 기타 작업을 수행합니다.
- 산업 자동화 - 트럭이 창고를 출입할 때 자동으로 감지하는 차량 번호판 판독기를 사용합니다. 창고에서 팔레트를 계수하고 자재 및 상품의 이동을 추적합니다. 열 카메라를 사용하여 산업용 기계의 과열을 감지하고 알려 예방적 유지 보수를 촉진하고 작업장을 안전하게 유지합니다.

Kinesis 비디오 스트림을 처음 사용하십니까?

Kinesis 비디오 스트림를 처음 사용하는 경우, 먼저 이어지는 단원을 순서대로 읽어보기를 권장합니다.

1. Amazon Kinesis 비디오 스트림: 사용 방법 (p. 5) - Kinesis 비디오 스트림 개념에 대한 내용.
2. Kinesis 비디오 스트림 시작하기 (p. 23) - 계정 설정 및 Kinesis 비디오 스트림 테스트 방법.
3. Kinesis 비디오 스트림 생산자 라이브러리 (p. 27) - Kinesis 비디오 스트림 생산자 애플리케이션 생성에 관한 내용.
4. Kinesis Video Stream Parser Library (p. 86) - Kinesis 비디오 스트림 소비자 애플리케이션에서의 수신 데이터 프레임 처리에 관한 내용.
5. Amazon Kinesis 비디오 스트림 예제 (p. 93) - Kinesis 비디오 스트림로 할 수 있는 작업의 예.

Kinesis 비디오 스트림 시스템 요구 사항

다음 단원에는 Amazon Kinesis 비디오 스트림용 하드웨어, 소프트웨어 및 스토리지 요구사항이 나와 있습니다.

항목

카메라 요구 사항

Kinesis 비디오 스트림 Producer SDK와 샘플을 실행하는 데 사용되는 카메라에는 다음 메모리 요구 사항이 있습니다.

- SDK 콘텐츠 보기에는 16MB 메모리가 필요합니다.
- 샘플 애플리케이션 구성은 512MB입니다. 이 값은 네트워크 연결이 양호하고 추가 버퍼링 요구 사항이 없는 생산자에게 적합합니다. 네트워크 연결이 약하고 추가 버퍼링이 필요한 경우 초당 프레임 속도에 프레임 메모리 크기를 곱하여 버퍼링의 초당 메모리 요구 사항을 계산할 수 있습니다. 메모리 할당에 대한 자세한 내용은 StorageInfo (p. 71)를 참조하십시오.

CPU에서 인코딩 워크로드를 제거할 수 있도록 H.264를 사용하여 데이터를 인코딩하는 USB 또는 RTSP(실시간 스트리밍 프로토콜) 카메라를 사용하는 것이 좋습니다.

현재로서는 데모 애플리케이션이 RTSP 스트리밍을 위한 UDP(User Datagram Protocol)를 지원하지 않습니다. 향후 이 기능을 추가할 예정입니다.

생산자 SDK는 다음 두 가지 유형의 카메라를 지원합니다.

- 웹 카메라.
- USB 카메라.
- H.264 인코딩을 지원하는 카메라(기본 설정).
- H.264 인코딩을 지원하지 않는 카메라.
- Raspberry Pi 카메라 모듈. 이것은 Raspberry Pi 디바이스에서 기본 설정입니다. 왜냐하면 비디오 데이터 전송을 위해 GPU에 연결되므로 CPU 처리에 오버헤드가 들지 않기 때문입니다.
- RTSP(네트워크) 카메라. 이 카메라는 비디오 스트림이 이미 H.264로 인코딩되어 있으므로 기본 설정입니다.

테스트된 카메라

Kinesis 비디오 스트림로 다음 USB 카메라를 테스트했습니다.

- Logitech 1080p
- Logitech C930
- Logitech C920(H.264)
- Logitech Brio(4K)

- SVPRO USB 카메라 170도 어안 렌즈 광각 1080P 2mp Sony IMX322 HD H.264 30fps 미니 알루미늄 USB 웹캠 카메라

Kinesis 비디오 스트림로 다음 IP 카메라를 테스트했습니다.

- Vivotek FD9371 – HTV/EHTV
- Vivotek IB9371 – HT
- Hikvision 3MP IP Camera DS-2CD2035FWD-I
- Sricam SP012 IP
- VStarcam 720P WiFi IP Camera(TCP)
- Ipccam Security Surveillance IP Camera 1080P
- AXIS P3354 고정 돔 네트워크 카메라

Kinesis 비디오 스트림로 테스트한 카메라 중에 Vivotek 카메라는 RTSP 스트림 일관성이 가장 높았습니다. 반면에 Sricam 카메라는 RTSP 스트림 일관성이 가장 낮았습니다.

테스트된 운영 체제

다음 디바이스와 운영 체제로 웹 카메라와 RTSP 카메라를 테스트했습니다.

- Mac mini
  - High Sierra
- MacBook Pro 랩톱
  - Sierra(10.12)
  - El Capitan(10.11)
- Ubuntu 16.04를 실행하는 HP 랩톱
- Ubuntu 17.10(도커 컨테이너)
- Raspberry Pi 3

SDK 스토리지 요구 사항

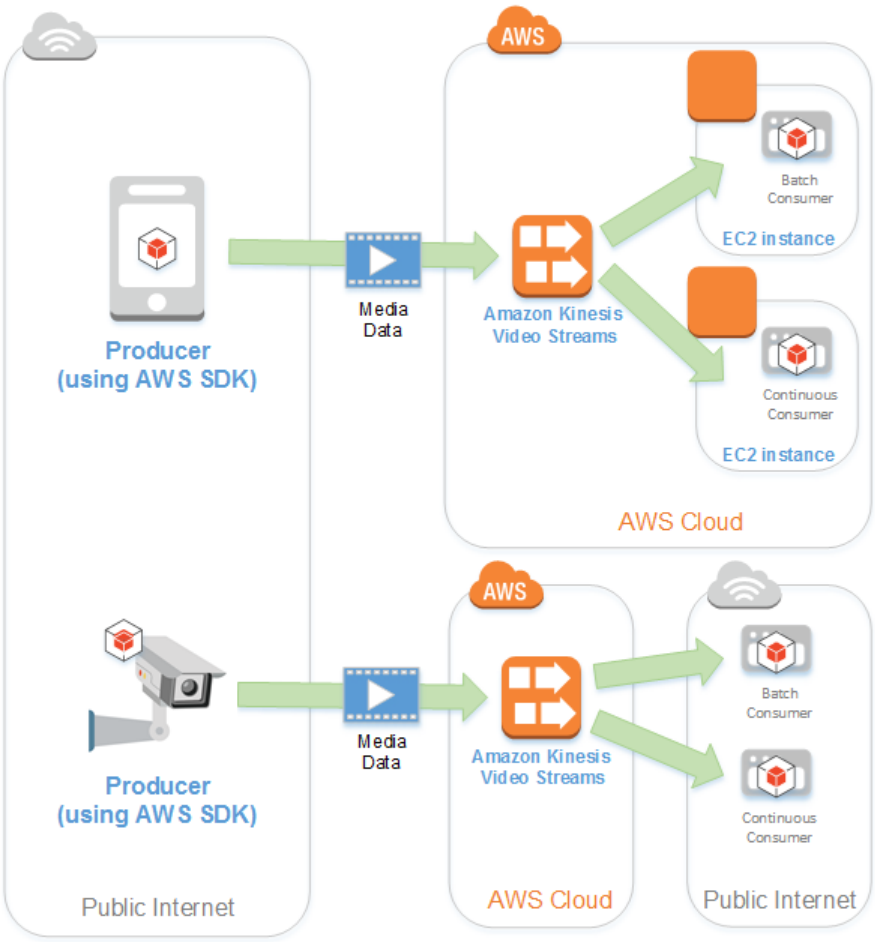
Kinesis 비디오 스트림 생산자 라이브러리 (p. 27) 설치 최소 요구 사항은 170MB이고 권장 스토리지 요구 사항은 512MB입니다.

Amazon Kinesis 비디오 스트림: 사용 방법

항목

Amazon Kinesis 비디오 스트림는 라이브 비디오를 디바이스에서 AWS 클라우드로 스트리밍하고 안정적으로 저장할 수 있는 완전 관리형 AWS 서비스입니다. 그리고 실시간 비디오 처리를 위한 자체 애플리케이션을 빌드하거나 배치 중심 비디오 분석을 수행할 수 있습니다.

다음 다이어그램은 Kinesis 비디오 스트림 사용 방법의 개요입니다.



다이어그램은 다음 구성 요소 간의 상호 작용을 보여 줍니다.

- 생산자 – 데이터를 Kinesis 비디오 스트림에 공급하는 임의의 소스. 생산자는 보안 카메라, 착용식 카메라, 스마트폰 카메라, 대시보드 카메라와 같은 임의의 영상 생성 디바이스일 수 있습니다. 생산자는 오디오 피드, 이미지, RADAR 데이터와 같은 비영상 데이터를 전송할 수도 있습니다.

  단일 생산자가 하나 이상의 비디오 스트림을 생성할 수 있습니다. 예를 들어, 비디오 스트림 하나가 비디오 데이터를 하나의 Kinesis 비디오 스트림에 푸시하고 오디오 데이터를 또 다른 스트림에 푸시할 수 있습니다.

  - Kinesis 비디오 스트림 생산자 라이브러리 - 디바이스에 설치하고 구성할 수 있는 사용하기 쉬운 소프트웨어 및 라이브러리 집합입니다. 이들 라이브러리를 사용하여 다양한 방식(예: 실시간, 몇 초 후 버퍼링, 사후 미디어 업로드)으로 비디오를 안전하게 연결하고 안정적으로 스트리밍할 수 있습니다.
- Kinesis 비디오 스트림 – 라이브 비디오 데이터를 전송하고, 선택적으로 저장하고, 실시간, 배치 혹은 애드혹 형식으로 데이터의 소비를 가능하게 할 수 있도록 해 주는 리소스입니다. 일반적인 구성에서는 Kinesis 비디오 스트림은 데이터를 푸시해 주는 생산자가 하나만 있습니다.

  스트림은 오디오, 비디오 및 타임 인코딩된 유사 데이터 스트림(예: 깊이 센서 피드, RADAR 피드 등)을 전송할 수 있습니다. AWS Management 콘솔을 사용하거나 AWS SDK를 사용하여 프로그램 방식으로 Kinesis 비디오 스트림을 생성합니다.

  복수의 비종속적 애플리케이션이 Kinesis 비디오 스트림을 병렬로 소비할 수 있습니다.
- 소비자 – Kinesis 비디오 스트림으로부터 조각 및 프레임과 같은 데이터를 가져와 열람하거나 처리하거나 분석합니다. 일반적으로 이런 소비자를 Kinesis 비디오 스트림 애플리케이션이라고 합니다. Kinesis 비디오 스트림에서 실시간으로 혹은 짧은 지연 시간이 요구되지 않는 경우 데이터가 안정적으로 저장되고 타임 인덱싱된 후에 데이터를 소비하고 처리하는 애플리케이션을 작성할 수 있습니다. 이와 같은 소비자 애플리케이션을 생성하여 Amazon EC2 인스턴스에서 실행되도록 할 수 있습니다.
  - Kinesis Video Stream Parser Library (p. 86) – Kinesis 비디오 스트림 애플리케이션이 짧은 지연 시간으로 Kinesis 비디오 스트림에서 미디어를 안정적으로 가져올 수 있도록 해 줍니다. 또한 애플리케이션이 프레임 자체의 처리 및 분석에 집중할 수 있도록 미디어에서 프레임 경계를 파싱합니다.

Kinesis 비디오 스트림 API 및 생산자 라이브러리 지원

Kinesis 비디오 스트림는 스트림을 생성하고 관리하며 비디오 스트림으로 혹은 스트림에서 미디어 데이터를 읽거나 쓸 수 있도록 API를 제공합니다. Kinesis 비디오 스트림 콘솔은 관리 기능 이외에 라이브 및 비디오 온디맨드 재생도 지원합니다. Kinesis 비디오 스트림는 애플리케이션 코드에서 사용하여 미디어 원본으로부터 데이터를 추출하고 Kinesis 비디오 스트림에 업로드할 수 있는 생산자 라이브러리 모음도 제공합니다.

항목

Kinesis 비디오 스트림 API

Kinesis 비디오 스트림는 Kinesis 비디오 스트림을 생성하고 관리할 수 있는 API를 제공합니다. 다음과 같이 미디어 데이터를 읽고 스트림에 쓸 수 있는 API도 제공합니다.

- 생산자 API – Kinesis 비디오 스트림는 미디어 데이터를 Kinesis 비디오 스트림에 쓸 수 있는 `PutMedia` API를 제공합니다. `PutMedia` 요청에서 생산자는 미디어 조각 스트림을 전송합니다. 조각은 독립적인 프레임 시퀀스입니다. 조각에 속한 프레임은 다른 조각의 어떤 프레임에 대한 종속성도 없어야 합니다. 자세한 내용은 PutMedia (p. 166) 단원을 참조하십시오.

  조각이 도착하면 Kinesis 비디오 스트림가 고유 조각 번호를 증가하는 순서대로 할당합니다. 또한 각 조각에 대해 생산자 측 및 서버 측 타임스탬프를 Kinesis 비디오 스트림 별 메타데이터로 저장합니다.

- 소비자 API –다음 API를 통해 소비자는 스트림으로부터 데이터를 가져올 수 있습니다.
  - `GetMedia` - 이런 API를 사용할 때 소비자는 시작 조각을 식별해야 합니다. 그런 다음 API가 스트림에 추가되는 순서대로 조각을 반환합니다(조각 번호가 증가하는 순서대로). 조각에 있는 미디어 데이터는 Matroska(MKV)와 같은 구조화 형식으로 압축됩니다. 자세한 내용은 GetMedia (p. 162) 단원을 참조하십시오.

    Note

    `GetMedia`는 조각이 어디에 있는지 압니다(데이터 스토어에 보관되거나 실시간으로 사용 가능). 예를 들어, `GetMedia`가 시작 조각이 보관되어 있다고 판단하는 경우 데이터 스토어로부터 조각을 반환하기 시작합니다. 아직 보관되지 않은 새 조각을 반환해야 하는 경우에는 `GetMedia`가 메모리 내 스트림 버퍼에서 조각 읽기로 전환합니다.

    이는 스트림에 의해 수집되는 순서대로 조각을 처리하는 지속적 소비자의 예입니다.

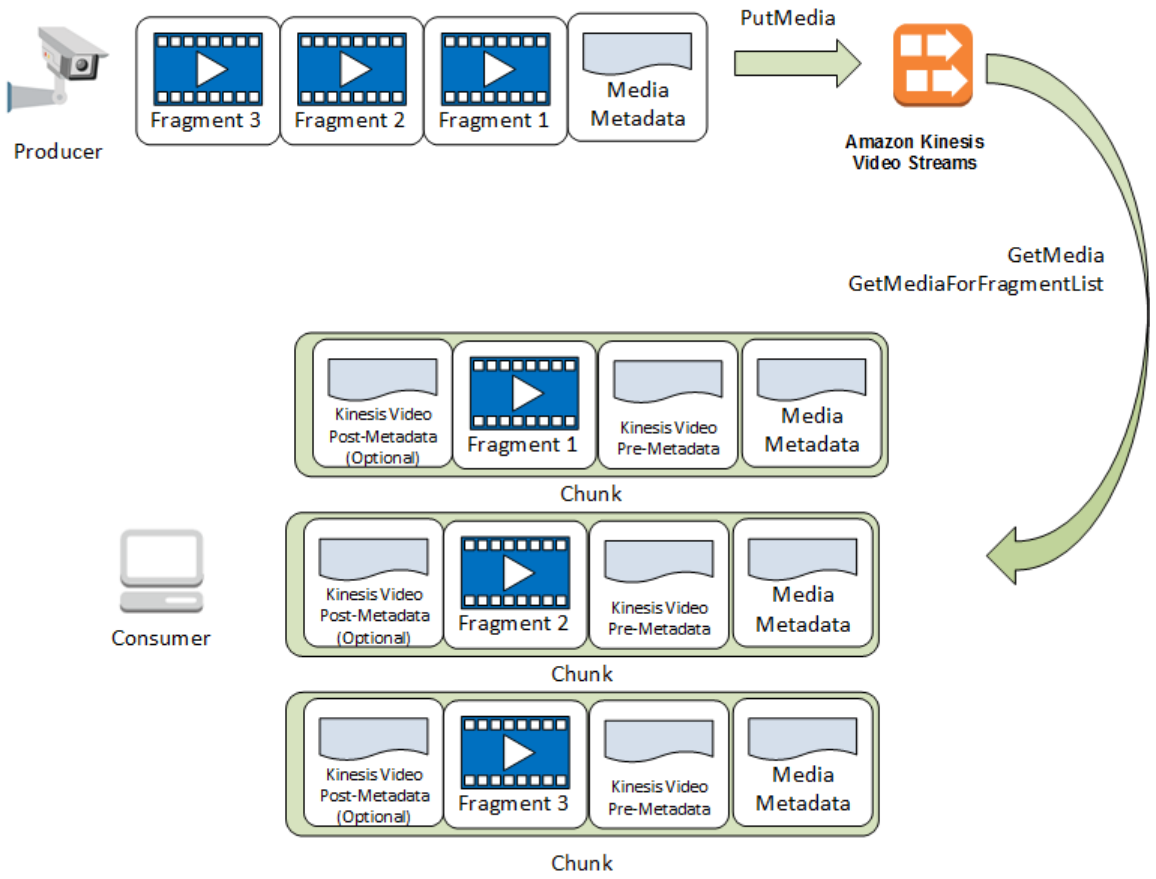
    `GetMedia`는 비디오 처리 애플리케이션이 실패하거나 뒤쳐지는 경우 추가적인 작업 없이 따라잡을 수 있도록 합니다. 애플리케이션은 `GetMedia`를 사용하여 데이터 스토어에 보관되어 있는 데이터를 처리하고, 애플리케이션이 따라잡으면서 `GetMedia`가 계속하여 미디어 데이터를 데이터가 도착하는 대로 실시간으로 공급합니다.
  - `GetMediaFromFragmentList` (및 `ListFragments`) - 배치 처리 애플리케이션은 오프라인 소비자로 간주됩니다. 오프라인 소비자는 `ListFragments` 및 `GetMediaFromFragmentList` API를 조합하여 특정 미디어 조각이나 비디오 범위를 명시적으로 가져올 수 있습니다. `ListFragments` 및 `GetMediaFromFragmentList`는 애플리케이션이 특정 시간 범위 혹은 조각 범위에 대해 비디오 조각을 식별한 다음 순차적으로 혹은 병렬로 가져와 처리할 수 있게 해 줍니다. 이와 같은 접근 방식은 대량의 데이터를 병렬로 신속하게 처리해야 하는 `MapReduce` 애플리케이션 제품군에 적합합니다.

    예를 들어, 소비자가 하루 동안의 비디오 조각을 처리해야 하는 경우를 가정해 보겠습니다. 소비자는 다음을 수행할 것입니다.

    1. `ListFragments` API를 호출하고 시간 범위를 지정하여 원하는 조각 모음을 선택함으로써 조각 목록을 얻습니다.

       API는 지정된 범위 안에 있는 모든 조각으로부터 메타데이터를 반환합니다. 메타데이터는 조각 번호, 생산자 측/서버 측 타임스탬프와 같은 정보를 제공합니다.
    2. 조각 메타데이터 목록을 취하고 임의의 순서대로 조각을 검색합니다. 예를 들어, 하루 동안의 모든 조각을 처리하기 위해서는, 소비자가 목록을 하위 목록으로 분할하고 작업자(예: 복수의 Amazon EC2 인스턴스)로 하여금 `GetMediaFromFragmentList`를 사용하여 조각으로 병렬로 가져와 병렬로 처리하는 것을 선택할 수 있습니다.

다음 다이어그램은 이들 API 호출 동안의 조각 및 청크의 데이터 흐름을 보여 줍니다.



생산자가 `PutMedia` 요청을 전송할 때 페이로드에 있는 미디어 메타데이터를 전송한 다음 미디어 데이터 조각의 시퀀스를 전송합니다. 데이터를 수신하면 Kinesis 비디오 스트림가 수신 미디어 데이터를 Kinesis 비디오 스트림 청크로 저장합니다. 각 청크는 다음과 같은 요소로 구성됩니다.

- 미디어 메타데이터의 사본
- 조각
- Kinesis 비디오 스트림 별 메타데이터(예: 조각 번호 및 서버 측 / 생산자 측 타임스탬프)

소비자가 미디어 메타데이터를 요청하는 경우 Kinesis 비디오 스트림는 요청에서 지정된 조각 번호로 시작하는 청크의 스트림을 반환합니다.

스트림에 대해 데이터 유지를 활성화한 경우 스트림에서 조각을 수신한 후에 Kinesis 비디오 스트림는 조각의 사본을 데이터 스토어에도 저장합니다.

생산자 라이브러리

Kinesis 비디오 스트림을 생성한 후에 스트림에 대한 데이터 전송을 시작할 수 있습니다. 애플리케이션 코드에서 이 라이브러리를 사용하여 미디어 원본으로부터 데이터를 추출하고 Kinesis 비디오 스트림에 업로드할 수 있습니다. 사용 가능한 생산자 라이브러리에 대한 자세한 내용은 Kinesis 비디오 스트림 생산자 라이브러리 (p. 27) 단원을 참조하십시오.

HLS를 사용한 Kinesis 비디오 스트림 재생

HTTP 라이브 스트리밍(HLS)은 산업 표준 HTTP 기반 미디어 스트리밍 통신 프로토콜입니다. HLS를 사용하여 실시간 재생 또는 아카이브된 비디오 시청을 위해 Amazon Kinesis 비디오 스트림를 볼 수 있습니다.

HLS 또는 GetMedia API를 사용하여 Kinesis 비디오 스트림를 볼 수 있습니다. 이러한 방법 사이의 차이는 다음과 같습니다.

- GetMedia: `GetMedia` API를 사용하여 Kinesis 비디오 스트림을 처리하는 자체 애플리케이션을 빌드합니다. `GetMedia`는 지연 시간이 낮은 실시간 API입니다. `GetMedia`를 사용하는 플레이어를 생성하려면 직접 빌드해야 합니다. `GetMedia`를 사용하여 Kinesis 비디오 스트림를 표시하는 애플리케이션을 개발하는 방법에 대한 자세한 내용은 스트림 구문 분석기 라이브러리 (p. 86) 단원을 참조하십시오.
- HLS: 실시간 재생을 위해 HLS를 사용할 수 있습니다. 지연 시간은 일반적으로 3~5초이지만, 사용 사례, 플레이어 및 네트워크 상태에 따라 1~10초일 수 있습니다. HLS 스트리밍 세션 URL을 프로그래밍 방식으로 또는 수동으로 제공하여 타사 플레이어(예: Video.js 또는 Google Shaka Player)를 사용해 비디오 스트림을 표시할 수 있습니다. 또한 Apple Safari 또는 Microsoft Edge 브라우저의 위치 표시줄에 HLS 스트리밍 세션 URL을 입력하여 비디오를 재생할 수도 있습니다.

HLS를 사용하여 Kinesis 비디오 스트림를 보려면 먼저 GetHLSStreamingSessionURL을 사용하여 스트리밍 세션을 생성합니다. 이 작업은 HLS 세션에 액세스하기 위한 URL(세션 토큰을 포함)을 반환합니다. 그런 다음 미디어 플레이어 또는 독립 실행형 애플리케이션에서 URL을 사용하여 스트림을 표시할 수 있습니다.

Amazon Kinesis 비디오 스트림가 HLS를 통해 비디오를 제공하기 위해서는 다음 요구 사항이 있습니다.

- 미디어 유형이 `video/h264`여야 합니다.
- 데이터 보존이 0보다 커야 합니다.
- 조각이 H.264 형식(MPEG-4 사양 ISO/IEC 14496-15)용 AVC(Advanced Video Coding)에 코덱 프라이빗 데이터를 포함해야 합니다. 스트림 데이터를 특정 형식에 맞게 조정하는 방법에 대한 자세한 내용은 NAL 적응 플래그 (p. 70) 단원을 참조하십시오.

예제: HTML 및 JavaScript에서 HLS 사용

다음 예제는 Kinesis 비디오 스트림용 HLS 스트리밍 세션을 검색하고 웹 페이지에서 재생하는 방법입니다. 예제에서는 다음 플레이어에서 비디오를 재생하는 방법을 보여줍니다.

- Video.js
- Google Shaka Player

항목

Kinesis 비디오 스트림 클라이언트 설정

HLS를 사용하여 스트리밍 비디오에 액세스하려면 먼저 Kinesis 비디오 스트림 클라이언트(서비스 엔드포인트를 검색하기 위해) 및 아카이브된 미디어 클라이언트(HLS 스트리밍 세션을 검색하기 위해)를 생성하고 구성합니다. 애플리케이션은 HTML 페이지의 입력 상자에서 필요한 값을 검색합니다.

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/aws-sdk/2.278.1/aws-sdk.min.js"></
script>
...

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

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

Kinesis 비디오 스트림 아카이브된 콘텐츠 엔드포인트 검색

클라이언트를 시작한 후 Kinesis 비디오 스트림 아카이브된 콘텐츠 엔드포인트를 검색하여 HLS 스트리밍 세션 URL을 검색합니다.

```
// Step 2: Get a data endpoint for the stream
kinesisVideo.getDataEndpoint({
    StreamName: streamName,
    APIName: "GET_HLS_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);
});
```

HLS 스트리밍 세션 URL 검색

아카이브된 콘텐츠 엔드포인트가 있으면 GetHLSStreamingSessionURL API를 호출하여 HLS 스트리밍 세션 URL을 검색합니다.

```
// Step 3: Get an HLS Streaming Session URL
console.log('Fetching HLS Streaming Session URL');
kinesisVideoArchivedContent.getHLSStreamingSessionURL({
    StreamName: streamName,
    PlaybackMode: $('#playbackMode').val(),
    HLSFragmentSelector: {
        FragmentSelectorType: $('#fragmentSelectorType').val(),
        TimestampRange: $('#playbackMode').val() === "LIVE" ? undefined : {
            StartTimestamp: new Date($('#startTimestamp').val()),
            EndTimestamp: new Date($('#endTimestamp').val())
        }
    },
    DiscontinuityMode: $('#discontinuityMode').val(),
    MaxMediaPlaylistFragmentResults: parseInt($('#maxMediaPlaylistFragmentResults').val()),
    Expires: parseInt($('#expires').val())
}, function(err, response) {
    if (err) { return console.error(err); }
    console.log('HLS Streaming Session URL: ' + response.HLSStreamingSessionURL);
```

스트리밍 비디오 표시

HLS 스트리밍 세션 URL이 있으면 비디오 플레이어에 제공합니다. 비디오 플레이어에 URL을 제공하는 방법은 사용되는 플레이어에 따라 다릅니다.

다음 코드 예제는 Video.js 플레이어에 스트리밍 세션 URL을 제공하는 방법입니다.

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

...

// Step 4: Give the URL to the video player.
var playerName = $('#player').val();
if (playerName === 'VideoJS') {
    var playerElement = $('#videojs');
    playerElement.show();
    var player = videojs('videojs');
    console.log('Created VideoJS Player');

    player.src({
        src: response.HLSStreamingSessionURL,
        type: 'application/x-mpegURL'
    });
    console.log('Set player source');

    player.play();
    console.log('Starting playback');
}
```

다음 코드 예제는 Google Shaka 플레이어에 스트리밍 세션 URL을 제공하는 방법입니다.

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

...

if (playerName === 'Shaka Player') {
    var playerElement = $('#shaka');
    playerElement.show();
    var player = new shaka.Player(playerElement[0]);
    console.log('Created Shaka Player');
    player.load(response.HLSStreamingSessionURL).then(function() {
        console.log('Starting playback');
});
console.log('Set player source');
```

문제 해결

비디오 스트림이 올바로 재생되지 않을 경우 HLS 문제 해결 (p. 121) 단원을 참조하십시오.

완성된 예제

여기서 완성된 예제 코드를 다운로드하거나 볼 수 있습니다.

IAM을 사용한 Kinesis 비디오 스트림 리소스 액세스 제어

AWS Identity and Access Management(IAM)과 Amazon Kinesis 비디오 스트림을 함께 사용하면 조직 내 사용자별로 특정 Kinesis 비디오 스트림 API 작업을 사용하는 작업 수행과 특정 AWS 리소스의 사용 권한을 제어할 수 있습니다.

IAM에 대한 자세한 내용은 다음을 참조하십시오.

- AWS Identity and Access Management(IAM)
- 시작하기
- IAM 사용 설명서

목차

정책 구문

IAM 정책은 하나 이상의 명령문으로 구성된 JSON 문서입니다. 각 명령문의 구조는 다음과 같습니다.

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

명령문을 이루는 요소는 다양합니다.

- Effect: effect는 `Allow` 또는 `Deny`일 수 있습니다. 기본적으로 IAM 사용자에게는 리소스 및 API 작업을 사용할 권한이 없으므로 모든 요청이 거부됩니다. 명시적 허용은 기본 설정을 무시합니다. 명시적 거부는 모든 허용을 무시합니다.
- Action: action은 권한을 부여하거나 거부할 특정 API 작업입니다.
- Resource: 작업의 영향을 받는 리소스입니다. 명령문에서 리소스를 지정하려면 Amazon 리소스 이름(ARN)을 사용해야 합니다.
- Condition: Condition은 선택 사항으로서 정책이 적용되는 시점을 제어하는 데 사용할 수 있습니다.

IAM 정책을 생성하고 관리할 때 IAM 정책 생성기 및 IAM 정책 시뮬레이터를 사용하려고 할 수 있습니다.

Kinesis 비디오 스트림 작업

IAM 정책 명령문에는 IAM을 지원하는 모든 서비스의 모든 API 작업을 지정할 수 있습니다. Kinesis 비디오 스트림의 경우 `kinesisvideo:` 접두사와 함께 API 작업 이름을 사용합니다. 예를 들어, `kinesisvideo:CreateStream`, `kinesisvideo:ListStreams` 및 `kinesisvideo:DescribeStream`입니다.

명령문 하나에 여러 작업을 지정하려면 다음과 같이 쉼표로 구분합니다.

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

와일드카드를 사용하여 여러 작업을 지정할 수도 있습니다. 예를 들어 다음과 같이 이름이 "Get"으로 시작되는 모든 작업을 지정할 수 있습니다.

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

모든 Kinesis 비디오 스트림 작업을 지정하려면 다음과 같이 별표(*) 와일드카드를 사용합니다.

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

Kinesis 비디오 스트림 API 작업의 전체 목록은 Kinesis 비디오 스트림 API 참조 단원을 참조하십시오.

Kinesis 비디오 스트림에 대한 Amazon 리소스 이름(ARN)

각 IAM 정책 명령문은 ARN을 사용하여 지정한 리소스에 적용됩니다.

Kinesis 비디오 스트림의 경우 다음 ARN 리소스 형식을 사용합니다.

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

예:

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

DescribeStream을 사용하여 스트림의 ARN을 가져올 수 있습니다.

다른 IAM 계정에 Kinesis 비디오 스트림 액세스 권한 부여하기

다른 IAM 계정에 Kinesis 비디오 스트림에서 작업을 수행할 권한을 부여해야 할 경우가 있습니다. 다음 개요에서는 계정 전체에 비디오 스트림 액세스 권한을 부여하는 일반적인 단계를 설명합니다.

1. 스트림에서 작업을 수행할 권한을 부여하고자 하는 계정의 12자리 계정 ID를 확보합니다(예: 111111111111).
2. 부여하고자 하는 액세스 수준을 허용하는 해당 스트림을 소유한 계정에 대한 관리형 정책을 생성합니다. Kinesis 비디오 스트림 리소스 정책의 예는 다음 단원의 정책 예제 (p. 14)를 참조하십시오.
3. 권한을 부여할 계정을 지정하여 역할을 생성하고 이전 단계에서 생성한 정책을 연결합니다.

4. 이전 단계에서 생성한 역할에 대한 `AssumeRole` 작업을 허용하는 관리형 정책을 생성합니다. 예를 들어, 역할의 모양은 다음과 같을 수 있습니다.

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

계정 교차 액세스에 대한 단계별 지침은 IAM 역할을 사용한 AWS 계정 간 액세스 권한 위임을 참조하십시오.

Kinesis 비디오 스트림 정책의 예

다음은 Kinesis 비디오 스트림에 대한 사용자 액세스를 제어하는 방법을 설명하는 예제 정책입니다.

Example 1: 사용자가 임의의 Kinesis 비디오 스트림에서 데이터를 가져오도록 허용

이 정책을 사용하면 사용자 또는 그룹이 임의의 Kinesis 비디오 스트림에 대해 `DescribeStream`, `GetDataEndpoint`, `GetMedia`, `ListStreams` 및 `ListTagsForStream` 작업을 수행할 수 있습니다. 이 정책은 임의의 비디오 스트림에서 데이터를 가져올 수 있는 사용자에게 해당됩니다.

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

Example 2: 사용자가 Kinesis 비디오 스트림를 생성하고 생성된 스트림에 데이터를 작성하도록 허용

이 정책을 사용하면 사용자 또는 그룹이 `CreateStream` 및 `PutMedia` 작업을 수행할 수 있습니다. 이 정책은 비디오 스트림을 생성하고 데이터를 생성된 스트림에 전송할 수 있는 보안 카메라에 해당됩니다.

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

Example 3. 사용자에게 모든 Kinesis 비디오 스트림 리소스에 대한 전면적인 액세스 권한 허용

이 정책을 사용하면 사용자 또는 그룹이 임의의 리소스에 대해 Kinesis 비디오 스트림 작업을 수행할 수 있습니다. 이 정책은 관리자에 해당됩니다.

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

Example 4: 사용자가 특정 Kinesis 비디오 스트림에 데이터를 작성하도록 허용

이 정책을 사용하면 사용자 또는 그룹이 특정 비디오 스트림에 데이터를 작성할 수 있습니다. 이 정책은 단일 스트림에 데이터를 전송할 수 있는 디바이스에 해당됩니다.

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

Kinesis 비디오 스트림를 통한 서버 측 암호화

AWS KMS(AWS Key Management Service) 키를 사용하는 서버 측 암호화를 통해 Amazon Kinesis 비디오 스트림에 저장된 데이터를 암호화하여 엄격한 데이터 관리 요구 사항을 더욱 쉽게 충족할 수 있습니다.

항목

Kinesis 비디오 스트림를 위한 서버 측 암호화

서버 측 암호화란 사용자가 지정하는 AWS KMS CMK(고객 마스터 키)를 사용하여 데이터가 저장되기 전에 자동으로 암호화하는 Kinesis 비디오 스트림의 기능입니다. 데이터는 Kinesis 비디오 스트림 스트림 스토리지 계층에 쓰여지기 전에 암호화되고 스토리지에서 검색된 후 해독됩니다. 결과적으로 데이터는 Kinesis 비디오 스트림 서비스 내에서 저장되는 동안 항상 암호화됩니다.

서버 측 암호화를 사용하면 Kinesis 비디오 스트림 생산자와 소비자가 마스터 키 또는 암호화 작업을 관리할 필요가 없습니다. 데이터 보존이 활성화되면 Kinesis 비디오 스트림로 들어오고 나갈 때 데이터가 자동으로

암호화되므로 데이터가 저장되는 동안에도 암호화됩니다. AWS KMS는 서버 측 암호화 기능에 사용되는 모든 마스터 키를 제공합니다. AWS KMS를 통해 AWS, 사용자 지정 AWS KMS CMK 또는 AWS KMS 서비스로 가져온 마스터 키로 관리되는 Kinesis 비디오 스트림용 CMK를 더욱 쉽게 사용할 수 있습니다.

비용, 리전 및 성능 고려 사항

서버 측 암호화를 적용하면 AWS KMS API 사용 및 주요 비용 대상이 됩니다. 사용자 지정 AWS KMS 마스터 키와는 달리 `(Default) aws/kinesis-video` 사용자 지정 마스터 키(CMK)는 무료로 제공됩니다. 하지만 Kinesis 비디오 스트림가 사용자를 대신하여 초래하는 API 사용 비용을 지불해야 합니다.

API 사용 비용은 사용자 지정을 포함한 모든 CMK에 적용됩니다. 각 사용자 자격 증명에 AWS KMS로의 고유한 API 호출이 필요하므로 KMS 비용은 사용자가 데이터 생산자 및 소비자에 사용하는 사용자 자격 증명의 수에 따라 조정됩니다.

다음은 리소스별 비용에 대한 설명입니다.

키

- AWS(별칭 = `aws/kinesis-video`)에서 관리하는 Kinesis 비디오 스트림용 CMK는 무료입니다.
- 사용자 생성 AWS KMS 키에는 AWS KMS 키 비용이 적용됩니다. 자세한 내용은 AWS Key Management Service 요금을 참조하십시오.

AWS KMS API 사용

트래픽이 증가함에 따라 새로운 데이터 암호화 키를 생성하거나, 기존 암호화 키를 검색하라는 API 요청이 증가합니다. 이 경우 AWS KMS 사용 비용이 적용됩니다. 자세한 내용은 AWS Key Management Service 요금: 사용량을 참조하십시오.

Kinesis 비디오 스트림는 보존이 0(보존 안 함)으로 설정되어 있어도 키 요청을 생성합니다.

리전별 서버 측 암호화 가용성

Kinesis 비디오 스트림를 사용할 수 있는 모든 AWS 리전에서 Kinesis 비디오 스트림의 서버 측 암호화를 사용할 수 있습니다.

서버 측 암호화를 시작하는 방법

Kinesis 비디오 스트림에는 서버 측 암호화가 항상 활성화되어 있습니다. 스트림 생성 시에 사용자 제공 키가 지정되지 않으면 기본 키(Kinesis 비디오 스트림가 제공)가 사용됩니다.

사용자 제공 AWS KMS 마스터 키는 생성될 때 Kinesis 비디오 스트림 스트림에 할당되어야 합니다. 나중에 UpdateStream API를 사용하여 스트림에 다른 키를 할당할 수 없습니다.

사용자 제공 AWS KMS 마스터 키는 다음과 같은 두 가지 방식으로 Kinesis 비디오 스트림 스트림에 할당할 수 있습니다.

- AWS Management 콘솔에서 Kinesis 비디오 스트림를 생성할 때 [Create new Kinesis Video stream] 페이지에 있는 [Encryption] 섹션에서 AWS KMS 마스터 키를 지정합니다.
- CreateStream API를 사용하여 Kinesis 비디오 스트림를 생성할 때 `KmsKeyId` 파라미터에서 키 ID를 지정합니다.

사용자 생성 AWS KMS 마스터 키 생성 및 사용

이 단원에서는 Amazon Kinesis 비디오 스트림에서 관리하는 마스터 키 대신 자체 AWS KMS 마스터 키를 생성하고 사용하는 방법을 설명합니다.

사용자 생성 AWS KMS 마스터 키 만들기

자체 마스터 키 생성 방법에 대한 정보는 AWS Key Management Service Developer Guide에 있는 Creating Keys를 참조하십시오. 계정 키를 생성하면 Kinesis 비디오 스트림 서비스가 [KMS master key] 목록에 이 키를 반환합니다.

사용자 생성 AWS KMS 마스터 키 사용

올바른 권한이 소비자, 생산자 및 관리자에게 적용되면 자체 AWS 계정 또는 다른 AWS 계정에서 사용자 지정 AWS KMS 마스터 키를 사용할 수 있습니다. 계정의 모든 AWS KMS 마스터 키는 콘솔의 [KMS master key] 목록에 표시됩니다.

다른 계정에 있는 사용자 지정 AWS KMS 마스터 키를 사용하려면 해당 키를 사용할 수 있는 권한이 필요합니다. 또한 `CreateStream` API를 사용하여 스트림을 생성해야 합니다. 콘솔에서 생성된 스트림에 있는 다른 계정의 AWS KMS 마스터 키는 사용할 수 없습니다.

> Note
>
> AWS KMS 키는 `PutMedia` 또는 `GetMedia` 작업이 실행되기 전까지 액세스되지 않습니다. 결과는 다음과 같습니다.
>
> - 여러분이 지정한 키가 존재하지 않는 경우, `CreateStream` 작업은 성공하지만, 스트림 상의 `PutMedia` 및 `GetMedia` 작업은 실패합니다.
> - 제공된 마스터 키(`aws/kinesis-video`)를 사용하는 경우, 해당 키는 첫 `PutMedia` 또는 `GetMedia` 작업이 수행되기 전까지 계정에 표시되지 않습니다.

사용자 생성 AWS KMS 마스터 키 사용 권한

사용자 생성 AWS KMS 마스터 키와 함께 서버 측 암호화를 사용하려면 먼저 스트림의 암호화와 스트림 레코드의 암호화 및 해독을 허용하도록 AWS KMS 키 정책을 구성해야 합니다. AWS KMS 권한에 대한 예제 및 자세한 내용은 AWS KMS API 권한: 작업 및 리소스 참조를 참조하십시오.

> Note
>
> 암호화에 기본 서비스 키를 사용할 때는 사용자 지정 IAM 권한을 적용할 필요가 없습니다.

사용자 생성 AWS KMS 마스터 키를 사용하려면 먼저 Kinesis 비디오 스트림 생산자 및 소비자(IAM 보안 주체)가 AWS KMS 마스터 키 정책에 속한 사용자여야 합니다. 그렇지 않으면 스트림에서 읽기 및 쓰기가 실패하여 궁극적으로 데이터 손실, 처리 지연 또는 애플리케이션 중단이 발생할 수 있습니다. IAM 정책을 사용하여 AWS KMS 키 권한을 관리할 수 있습니다. 자세한 내용은 Using IAM Policies with AWS KMS을 참조하십시오.

예제 생산자 권한

Kinesis 비디오 스트림 생산자에게 `kms:GenerateDataKey` 권한이 있어야 합니다.

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

예제 소비자 권한

Kinesis 비디오 스트림 소비자에게 `kms:Decrypt` 권한이 있어야 합니다.

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

Kinesis 비디오 스트림 데이터 모델

생산자 라이브러리 (p. 27) 및 스트림 구문 분석기 라이브러리 (p. 86)가 비디오 데이터와 함께 정보를 포함하는 것을 지원하는 형식으로 비디오 데이터를 전송하고 수신합니다. 이 형식은 Matroska(MKV) 사양을 기반으로 합니다.

MKV 형식은 미디어 데이터에 적용되는 공개 사양입니다. Amazon Kinesis 비디오 스트림 개발자 안내서에 있는 모든 라이브러리 및 코드 예제는 MKV 형식으로 데이터를 전송하거나 수신합니다.

Kinesis 비디오 스트림 생산자 라이브러리 (p. 27)는 `StreamDefinition` 및 `Frame` 유형을 사용하여 MKV 스트림 헤더, 프레임 헤더 및 프레임 데이터를 생성합니다.

전체 MKV 사양에 관한 내용은 Matroska 사양을 참조하십시오.

다음 단원에서는 C++ 생산자 라이브러리 (p. 36)에 의해 생성되는 MKV 형식 데이터의 구성 요소를 설명합니다.

항목

- 스트림 헤더 요소 (p. 19)
- 프레임 헤더 요소 (p. 21)
- MKV 프레임 데이터 (p. 22)

스트림 헤더 요소

다음과 같은 MKV 헤더 요소는 `StreamDefinition`에 의해 사용됩니다(`StreamDefinition.h`에 정의되어 있음).

요소	설명	일반적인 값
stream_name	Kinesis 비디오 스트림의 이름에 해당합니다.	my-stream
retention_period	Kinesis 비디오 스트림에 의해 스트림 데이터가 유지되는 기간입니다. 데이터를 유지하지 않는 스트림에 대해서는 `0`을 지정합니다.	24
tags	사용자 데이터의 키-값 모음입니다. 이 데이터는 AWS Management 콘솔에 표시되며, 클라이언트 애플리케이션이 읽어 필터링하거나 스트림에 관한 정보를 얻을 수 있습니다.	
kms_key_id	있는 경우, 스트림에서 데이터를 암호화하는 데 사용하는 사용자 정의 AWS KMS 마스터 키입니다. 없는 경우에는 데이터가 Kinesis가 제공하는 마스터 키에 의해 암호화됩니다(`aws/kinesis-video`).	01234567-89ab-cdef-0123-456789ab
streaming_type	현재 유효한 유일한 스트리밍은 `STREAMING_TYPE_REALTIME`입니다.	STREAMING_TYPE_REALTIME
content_type	사용자 정의 콘텐츠 유형입니다. 콘솔에서 재생할 비디오 데이터 스트리밍의 경우 콘텐츠 유형이 `video/h264`이어야 합니다.	video/h264
max_latency	이 값은 현재 사용되지 않으며, 0으로 설정해야 합니다.	0
fragment_duration	이는 조각 길이의 추정치로서 최적화에 사용됩니다. 실제 조각 지속 시간은 스트리밍 데이터에 의해 결정됩니다.	2
timecode_scale	프레임 타임스탬프에 사용되는 척도를 나타냅니다. 기본값은 1밀리초입니다. `0`을 지정하면 기본값 1밀리초도 할당됩니다. 이 값은 100나노초에서 1초 사이일 수 있습니다. 자세한 내용은 Matroska 설명서의 TimecodeScale을 참조하십시오.	

요소	설명	일반적인 값
key_frame_fragmentation	`true`인 경우, 키프레임 수신 시 스트림이 새 클러스터를 시작합니다.	true
frame_timecodes	`true`인 경우, 프레임이 수신될 때 Kinesis 비디오 스트림가 프레임에 스탬프를 지정합니다. `false`인 경우, Kinesis 비디오 스트림가 수신 프레임의 디코딩 타임을 사용합니다.	true
absolute_fragment_time	`true`인 경우 클러스터 타임코드가 절대 시간(예: 생산자 시스템 시계의 값)을 사용하는 것으로 해석됩니다. `false`인 경우, 클러스터 타임코드가 스트림 시작 시간에 상대적인 것으로 해석됩니다.	true
fragment_acks	`true`인 경우, Kinesis 비디오 스트림가 데이터를 수신할 때 ACK(acknowledgement)가 전송됩니다. ACK는 `KinesisVideoStreamFragmentAck` 또는 `KinesisVideoStreamParseFragmentAck` 콜백을 사용하여 수신할 수 있습니다.	true
restart_on_error	스트림 오류가 발생한 후 스트림이 전송을 다시 시작해야 하는지 여부를 나타냅니다.	true
nal_adaptation_flags	콘텐츠에 NAL(Network Abstraction Layer) 적응 또는 코덱 프라이빗 데이터가 존재하는지를 나타냅니다. 유효한 플래그에는 `NAL_ADAPTATION_ANNEXB_NALS` 및 `NAL_ADAPTATION_ANNEXB_CPD_NALS`가 있습니다.	NAL_ADAPTATION_ANNEXB_NALS
frame_rate	콘텐츠 프레임 속도의 추정치입니다. 이 값은 최적화에 사용되며, 실제 프레임 속도는 수신 데이터의 속도에 의해 결정됩니다. `0`을 지정하면 기본값 `24`가 할당됩니다.	24
avg_bandwith_bps	콘텐츠 대역폭의 추정치입니다. 이 값은 최적화에 사용되며, 실제 속도는 수신 데이터의 대역폭에 따라 결정됩니다. 예를 들어, 25FPS로 실행되는 720p 해상도 비디오 스트림의 경우 평균 대역폭을 5Mbps로 예상할 수 있습니다.	5

요소	설명	일반적인 값
buffer_duration	콘텐츠가 생산자에서 버퍼링되는 기간입니다. 네트워크 지연 시간이 짧은 경우 이 값이 감소될 수 있습니다. 네트워크 지연 시간이 긴 경우 이 값이 증가하므로, 할당을 통해 프레임을 더 작은 버퍼에 입력하지 못하기 때문에 프레임이 전송되기 전에 삭제되는 것을 방지할 수 있습니다.	
replay_duration	연결이 끊길 경우 비디오 데이터 스트림을 "뒤로 돌리는" 시간의 양을 가리킵니다. 연결 끊김으로 인한 프레임 손실이 문제가 되지 않을 경우 이 값은 0이 될 수 있습니다. 소비하는 애플리케이션이 과잉 프레임을 제거할 수 있는 경우 값을 증가시킬 수 있습니다. 이 값은 버퍼 기간보다 작아야 합니다. 그렇지 않으면 버퍼 기간이 사용됩니다.	
connection_staleness	수신 데이터가 없을 경우 연결이 유지되는 기간입니다.	
codec_id	콘텐츠에 사용되는 코덱입니다. 자세한 내용은 Matroska 사양의 CodecID를 참조하십시오.	V_MPEG2
track_name	사용자가 정의하는 트랙 이름입니다.	my_track
codecPrivateData	프레임 데이터를 디코딩하는 목적으로 사용되며 인코더에 의해 제공되는 데이터(예: 프레임 픽셀의 높이 및 폭)로서, 다수의 다운스트림 소비자가 필요로 합니다. C++ 생산자 라이브러리 (p. 36)에서는 `MkvStatics.cpp`의 `gMkvTrackVideoBits` 어레이에 프레임의 픽셀 폭과 높이가 포함되어 있습니다.	
codecPrivateDataSize	`codecPrivateData` 파라미터에 있는 데이터의 크기입니다.	

프레임 헤더 요소

다음의 MKV 헤더 요소는 `Frame`에 의해 사용됩니다(`mkvgen/Include.h`의 `KinesisVideoPic` 패키지에 정의되어 있음).

- 프레임 인덱스: 단순 증가 값.
- 플래그: 프레임의 유형 다음이 유효 값에 포함됩니다.
  - `FRAME_FLAGS_NONE`

  - `FRAME_FLAG_KEY_FRAME`: `key_frame_fragmentation`가 스트림에 설정되어 있는 경우, 키 프레임이 새 조각을 시작합니다.
  - `FRAME_FLAG_DISCARDABLE_FRAME`: 디코딩이 느린 경우 이 프레임을 무시할 수 있음을 디코더에 알립니다.
  - `FRAME_FLAG_INVISIBLE_FRAME`: 이 블록의 지속 시간은 0입니다.
- 디코딩 타임스탬프: 이 프레임이 디코딩되었을 때의 타임스탬프입니다. 이전 프레임이 이 프레임에 의존하여 디코딩하는 경우, 이 타임스탬프는 이전 프레임의 타임스탬프보다 앞설 수 있습니다. 이 값은 조각의 시작을 기준으로 합니다.
- 프레젠테이션 타임스탬프: 이 프레임이 표시될 때의 타임스탬프입니다. 이 값은 조각의 시작을 기준으로 합니다.
- 지속 시간: 프레임의 재생 시간입니다.
- 크기 프레임 데이터의 바이트 단위 크기입니다.

MKV 프레임 데이터

`frame.frameData`에 있는 데이터는 사용되는 인코딩 스키마에 따라 프레임에 대한 미디어 데이터만 포함할 수도 있고, 더 중첩된 헤더 정보를 포함할 수도 있습니다. AWS Management 콘솔에 표시되도록 하려면 데이터가 H.264 코덱으로 인코딩되어야 하지만, Kinesis 비디오 스트림은 형식을 불문하고 시계열화된 데이터 스트림을 수신할 수 있습니다.

Kinesis 비디오 스트림 시작하기

이 단원에서는 Amazon Kinesis 비디오 스트림에서 다음 작업을 수행하는 방법을 설명합니다.

- 아직 하지 않았다면 AWS 계정을 설정하고 관리자를 생성합니다.
- Kinesis 비디오 스트림 생성

이 안내서의 다른 단원에서는 데이터를 스트림에 전송하고 스트림에서 데이터를 보는 방법을 설명합니다.

Amazon Kinesis 비디오 스트림이 처음이라면 먼저 Amazon Kinesis 비디오 스트림: 사용 방법 (p. 5)를 읽어 보는 것이 좋습니다.

> **Note**
>
> 시작하기 샘플을 따르면 AWS 계정에 요금이 부과되지 않습니다. 해당 지역의 데이터 비용은 Amazon Kinesis 비디오 스트림 요금을 참조하십시오.

항목

1단계: AWS 계정 설정 및 관리자 생성하기

Kinesis 비디오 스트림를 처음 사용하는 경우, 먼저 다음 작업을 완료해야 합니다.

1. AWS에 가입 (p. 23) (이미 계정이 있는 경우는 제외)
2. 관리자 IAM 사용자 생성하기 (p. 24)

AWS에 가입

AWS 계정이 이미 있는 경우에는 이 단계를 건너뛸 수 있습니다.

Amazon Web Services(AWS)에 가입하면 Kinesis 비디오 스트림를 포함한 AWS의 모든 서비스에 AWS 계정이 자동으로 등록됩니다. Kinesis 비디오 스트림를 사용하는 경우 서비스에 유입 및 저장되고 서비스에서 사용되는 데이터의 양을 기준으로 과금됩니다. AWS를 처음 사용하는 고객인 경우 Kinesis 비디오 스트림를 무료로 시작할 수 있습니다. 자세한 내용은 AWS 프리 티어를 참조하십시오.

AWS 계정을 생성하려면

1. https://aws.amazon.com/을 열고 [Create an AWS Account]를 선택합니다.

   > **Note**
   >
   > 이전에 AWS Management 콘솔에 로그인한 적이 있을 경우 이를 브라우저에서 사용하지 못할 수 있습니다. 그와 같은 경우 [Sign in to a different account]를 선택한 다음, [Create a new AWS account]를 선택합니다.

2. 온라인 지시 사항을 따릅니다.

   등록 절차 중 전화를 받고 전화 키패드를 사용하여 PIN을 입력하는 과정이 있습니다.

이어지는 작업에 필요하므로 AWS 계정 ID를 기록해 두십시오.

관리자 IAM 사용자 생성하기

AWS에 가입할 때 AWS 계정과 연결된 이메일 주소 및 암호를 입력해야 합니다. 이는 AWS 계정 루트 사용자입니다. 루트 사용자의 자격 증명을 사용하면 모든 AWS 리소스에 전면적으로 액세스할 수 있습니다.

> Note
>
> 보안상 이유로 관리자, 즉 AWS 계정에 대한 전면적 권한이 있는 IAM 사용자를 생성할 때에만 루트 사용자를 사용하는 것이 좋습니다. 그런 다음 이 관리자를 사용하여 제한된 권한을 지닌 다른 IAM 사용자 및 역할을 생성할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 IAM 모범 사례 및 관리자 및 그룹 만들기를 참조하십시오.

관리자를 생성하고 콘솔에 로그인하는 방법

1. AWS 계정에서 관리자를 생성합니다. 이에 대한 지침은 IAM 사용 설명서의 IAM 사용자와 관리자 그룹 처음 만들기를 참조하십시오.
2. 관리자 자격으로 특별 URL을 사용하여 콘솔에 로그인합니다. 자세한 내용은 IAM 사용 설명서의 사용자 계정 로그인 방법 단원을 참조하십시오.

관리자는 계정에서 추가로 사용자를 생성할 수 있습니다. IAM 사용자는 기본적으로 어떤 권한도 없습니다. 관리자는 사용자를 생성하고 그 권한을 관리할 수 있습니다. 자세한 내용은 첫 번째 IAM 사용자 및 관리자 그룹 생성하기를 참조하십시오.

IAM에 대한 자세한 내용은 다음을 참조하십시오.

- AWS Identity and Access Management(IAM)
- 시작하기
- IAM 사용 설명서

다음 단계

2 단계: Kinesis 비디오 스트림 생성 (p. 24)

2 단계: Kinesis 비디오 스트림 생성

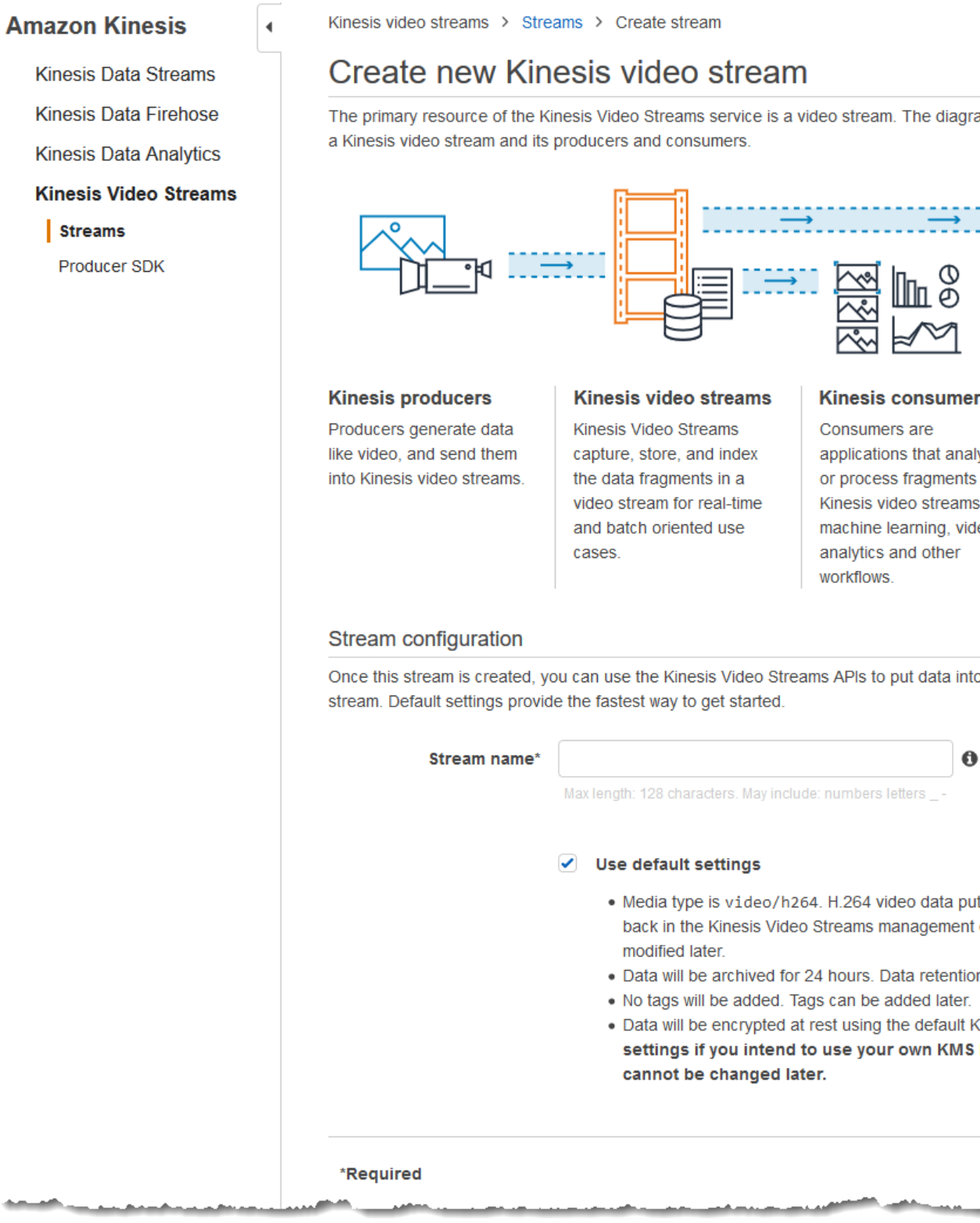
이 단원에서는 Kinesis 비디오 스트림을 생성하는 방법을 설명합니다.

이 단원에는 다음 절차가 포함됩니다.

- the section called “콘솔을 사용하여 비디오 스트림 생성하기” (p. 24)
- the section called “AWS CLI를 사용하여 비디오 스트림 생성” (p. 26)

콘솔을 사용하여 비디오 스트림 생성하기

1. AWS Management 콘솔에 로그인한 다음 https://console.aws.amazon.com/kinesis에서 Kinesis 콘솔을 엽니다.
2. [Manage streams] 페이지에서 [Create]을 선택합니다.
3. [Create new KinesisVideo Stream] 페이지에서 스트림 이름에 **`ExampleStream`**을 입력합니다. [Use default settings] 확인란을 선택된 상태로 둡니다.



4. [Create stream]을 선택합니다.
5. Kinesis 비디오 스트림가 스트림을 생성한 후에 [ExampleStream] 페이지에서 세부 정보를 검토합니다.

AWS CLI를 사용하여 비디오 스트림 생성

1. AWS CLI가 설치되고 구성되어 있는지 확인합니다. 자세한 내용은 AWS 명령줄 인터페이스 설명서를 참조하십시오.
2. AWS CLI에서 다음과 같은 `Create-Stream` 명령을 실행합니다.

```
$ aws kinesisvideo create-stream --stream-name "MyKinesisVideoStream" --data-retention-
in-hours "24"
```

다음 단계

다음 단계 (p. 26)

다음 단계

비디오 스트림이 생성된 후에는 데이터를 Java 애플리케이션으로부터 스트림에 전송할 수 있습니다. 코드에서 Kinesis 비디오 스트림 옵션을 사용하여 애플리케이션을 구성하고 미디어 원본으로부터 데이터를 추출하고 스트림에 업로드합니다. 자세한 내용은 Java 생산자 라이브러리 사용하기 (p. 28) 단원을 참조하십시오.

Kinesis 비디오 스트림 생산자 라이브러리

Amazon Kinesis 비디오 스트림 생산자 라이브러리는 Kinesis 비디오 스트림 생산자 SDK의 일부로서 사용하기 쉬운 라이브러리 집합입니다. 클라이언트는 라이브러리와 SDK를 사용하여 Kinesis 비디오 스트림에 안전하게 연결하고 비디오 및 기타 미디어 데이터를 스트리밍하기 위한 디바이스 탑재 애플리케이션을 빌드합니다. 그러면 콘솔이나 클라이언트 애플리케이션에서 실시간으로 이것을 볼 수 있습니다.

다음과 같은 방식으로 미디어 데이터를 스트리밍할 수 있습니다.

- 실시간 미디어 데이터 스트리밍
- 몇 초 간 버퍼링 후 미디어 데이터 스트리밍
- 미디어 업로드 후 스트리밍

Kinesis 비디오 스트림 스트림을 생성한 후에 스트림에 대한 데이터 전송을 시작할 수 있습니다. SDK를 사용하여 미디어 원본에서 비디오 데이터(프레임)를 추출하여 Kinesis 비디오 스트림에 업로드하는 애플리케이션 코드를 만들 수 있습니다. 이러한 애플리케이션을 생산자 애플리케이션이라고도 합니다.

생산자 라이브러리에는 다음의 구성 요소가 포함됩니다.

- Kinesis 비디오 스트림 생산자 클라이언트 (p. 27)
- Kinesis 비디오 스트림 생산자 라이브러리 (p. 28)

Kinesis 비디오 스트림 생산자 클라이언트

Kinesis 비디오 스트림 생산자 클라이언트는 단일 `KinesisVideoClient` 클래스를 포함합니다. 이 클래스는 미디어 원본을 관리하고, 원본으로부터 데이터를 수신하고, 미디어 원본에서 Kinesis 비디오 스트림로 향하는 데이터 흐름을 스트림 수명 주기로 관리합니다. 또한 Kinesis 비디오 스트림와 사용자의 사유 하드웨어 및 소프트웨어 간의 상호 작용을 정의하는 데 사용되는 `MediaSource` 인터페이스를 제공합니다.

미디어 원본은 거의 모든 것이 될 수 있습니다. 예를 들어, 카메라 미디어 원본 또는 마이크 미디어 원본을 사용할 수 있습니다. 미디어 원본은 오디오 및 비디오 원본만으로 국한되지 않습니다. 예를 들어, 데이터 로그는 텍스트 파일이어야 하지만, 데이터 스트림으로도 전송할 수 있습니다. 데이터를 동시에 스트리밍하는 여러 대의 카메라를 스마트폰에 탑재할 수도 있습니다.

임의의 원본으로부터 데이터를 얻기 위해서는 `MediaSource` 인터페이스를 구현해야 합니다. 이 인터페이스는 기본적으로는 지원하지 않는 추가적인 시나리오도 구현할 수 있습니다. 예를 들어, 다음을 Kinesis 비디오 스트림로 전송할 수 있습니다.

- 진단 데이터 스트림(예: 애플리케이션 로그 및 이벤트)
- 적외선 카메라, RADAR 또는 깊이 카메라의 데이터

Kinesis 비디오 스트림에는 카메라와 같은 미디어 생산 디바이스를 구현하기 위한 기능이 탑재되어 있지 않습니다. 이러한 디바이스로부터 데이터를 추출하려면 코드를 구현해야 하기 때문에 자체적인 미디어 원본 구현을 생성해야 합니다. 그런 다음 사용자 지정 미디어 원본을 Kinesis 비디오 스트림에 업로드하는 `KinesisVideoClient`에 명시적으로 등록합니다.

Kinesis 비디오 스트림 생산자 클라이언트는 Java 및 Android 애플리케이션에 사용할 수 있습니다. 자세한 내용은 Java 생산자 라이브러리 사용하기 (p. 28) 및 Android 생산자 라이브러리 사용 (p. 32) 단원을 참조하십시오.

Kinesis 비디오 스트림 생산자 라이브러리

Kinesis 비디오 스트림 생산자 라이브러리는 Kinesis 비디오 스트림 생산자 클라이언트에 포함됩니다. Kinesis 비디오 스트림와의 심층 통합을 원하는 사용자도 이 라이브러리를 직접 사용할 수 있습니다. 전용 운영 체제, 네트워크 스택 또는 제한적 내장 리소스를 갖춘 디바이스와의 통합도 가능합니다.

Kinesis 비디오 스트림 생산자 라이브러리는 Kinesis 비디오 스트림로의 스트리밍을 위한 상태 시스템을 구현합니다. 사용자 자체의 전송 구현을 요구하고 서비스를 왕래하는 각 메시지를 명시적으로 처리하는 콜백 후크를 제공합니다.

Kinesis 비디오 스트림 생산자 라이브러리를 사용하기로 선택하는 경우는 다음과 같습니다.

- 애플리케이션을 실행하고자 하는 디바이스에 Java 가상 머신이 없는 경우.
- Java 이외의 언어로 애플리케이션 코드를 작성하고자 하는 경우.
- 디바이스에 Java가 있지만, 메모리 및 처리 용량 제한과 같은 문제로 인해 코드의 양을 줄여 최소한의 추상화 수준으로 제한하려는 경우.

현재 Kinesis 비디오 스트림 생산자 클라이언트를 C++ 애플리케이션에 대해 이용할 수 있습니다. 자세한 내용은 C++ 생산자 라이브러리 사용하기 (p. 36) 단원을 참조하십시오.

관련 주제

Java 생산자 라이브러리 사용하기 (p. 28)

Android 생산자 라이브러리 사용 (p. 32)

C++ 생산자 라이브러리 사용하기 (p. 36)

Java 생산자 라이브러리 사용하기

Amazon Kinesis 비디오 스트림는 Java 생산자 라이브러리를 제공하며, 이를 사용하여 미디어 데이터를 디바이스에서 Kinesis 비디오 스트림로 전송하는 애플리케이션 코드를 최소한의 구성으로 작성할 수 있습니다.

애플리케이션이 Kinesis 비디오 스트림에 대한 데이터 스트리밍을 시작할 수 있으려면 다음 단계를 수행하여 코드를 Kinesis 비디오 스트림와 통합해야 합니다.

1. `KinesisVideoClient` 객체의 인스턴스를 생성합니다.
2. 미디어 원본 정보를 제공하여 `MediaSource` 객체를 생성합니다. 예를 들어 카메라 미디어 원본을 생성할 때 카메라를 식별하고 카메라가 사용하는 인코딩을 지정하는 등 정보를 제공합니다.

   스트리밍을 시작하고자 할 때 사용자 지정 미디어 원본을 생성해야 합니다.
3. 미디어 원본을 `KinesisVideoClient`에 등록합니다.

   `KinesisVideoClient`에 미디어 원본을 등록한 후에 해당 미디어 원본으로 데이터를 사용할 수 있을 때 해당 데이터로 `KinesisVideoClient`를 호출합니다.

절차: Java 생산자 SDK 사용

이 절차는 Java 애플리케이션에서 Kinesis 비디오 스트림 Java 생산자 클라이언트를 사용하여 데이터를 Kinesis 비디오 스트림에 전송하는 방법을 보여 줍니다.

이 단계에서는 카메라 또는 마이크와 같은 미디어 소스를 필요로 하지 않습니다. 대신 테스트 목적으로 코드가 일련의 바이트로 구성된 샘플 프레임을 생성합니다. 카메라 및 마이크와 같은 실제 소스로부터 미디어 데이터를 전송할 때 동일한 코딩 패턴을 사용할 수 있습니다.

이 절차에는 다음 단계가 포함됩니다.

- 코드 다운로드 및 구성
- 코드 쓰기 및 검사
- 코드 실행 및 확인

사전 조건

- 샘플 코드에서는 AWS 자격 증명 프로필 파일에서 설정한 프로필을 지정하여 자격 증명을 제공합니다. 아직 설정하지 않았다면 먼저 자격 증명 프로필을 설정합니다. 자세한 내용은 AWS SDK for Java의 개발을 위한 AWS 자격 증명 및 리전 설정 단원을 참조하십시오.

  Note

  Java 예제에서는 `SystemPropertiesCredentialsProvider` 객체를 사용하여 AWS 자격 증명을 취득합니다. 제공자가 `aws.accessKeyId` 및 `aws.secretKey` Java 시스템 속성으로부터 이들 자격 증명을 검색합니다. Java 개발 환경에서 이 시스템 속성을 설정합니다. Java 시스템 속성 설정 방법에 관한 정보는 특정 통합 개발 환경(IDE)에 관한 문서를 참조하십시오.
- `NativeLibraryPath`에 `KinesisVideoProducerJNI` 파일(https://github.com/awslabs/amazon-kinesis-video-streams-producer-sdk-cpp)이 있어야 합니다. 이 파일의 파일 이름 확장자는 운영 체제에 따라 다릅니다.
  - KinesisVideoProducerJNI.so - Linux
  - KinesisVideoProducerJNI.dylib - macOS
  - KinesisVideoProducerJNI.dll - Windows (현재 제공되지 않음)

    Note

    macOS, Ubuntu 및 Raspbian용 사전 구축 라이브러리는 `src/main/resources/lib`에서 제공됩니다. 다른 환경에서는 C++ 생산자 라이브러리 (p. 36)를 컴파일합니다.

1 단계: Java 생산자 라이브러리 코드 다운로드 및 구성

Java 생산자 라이브러리 절차의 이 단원에서는 Java 예제 코드를 다운로드하고, 프로젝트를 Java IDE에 가져온 다음, 라이브러리 위치를 구성합니다.

사전 요건과 이 예제에 관한 기타 세부 정보는 Java 생산자 라이브러리 사용을 참조하십시오.

1. 디렉터리를 생성한 다음 GitHub 리포지토리에서 예제 소스 코드를 복제합니다.

   ```
   $ git clone https://github.com/awslabs/amazon-kinesis-video-streams-producer-sdk-java
   ```
2. 사용하는 Java 통합 개발 환경(IDE)(예: Eclipse 또는 JetBrains IntelliJ IDEA)를 열고 다운로드한 Apache Maven 프로젝트를 가져옵니다.

   - IntelliJ IDEA에서 [Import]를 선택합니다. 다운로드한 패키지의 루트에 있는 `pom.xml` 파일을 찾습니다.

- Eclipse: [File], [Import], [Maven], [Existing Maven Projects]를 차례로 선택합니다. 그런 다음 `kinesis-video-java-demo` 디렉터리로 이동합니다.

자세한 내용은 IDE 관련 문서를 참조하십시오.

3. Java 예제 코드는 현재 AWS 자격 증명을 사용합니다. 다른 자격 증명 프로필을 사용하려면 `DemoAppMain.java`에 있는 다음 코드를 찾습니다.

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

코드를 다음으로 변경합니다.

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

자세한 내용은 AWS SDK for Java 참조에 있는 ProfileCredentialsProvider를 참조하십시오.

다음 단계

the section called “2 단계: 코드 쓰기 및 검사” (p. 30)

2 단계: 코드 쓰기 및 검사

Java 생산자 라이브러리 절차의 이 단원에서는 이전 단원에서 다운로드한 Java 예제 코드를 작성하고 검사합니다.

Java 테스트 애플리케이션(`DemoAppMain`)은 다음과 같은 코딩 패턴을 보입니다.

- `KinesisVideoClient`의 인스턴스를 만듭니다.
- `MediaSource`의 인스턴스를 만듭니다.
- `MediaSource`를 클라이언트에 등록합니다.
- 스트리밍을 시작합니다. 즉, `MediaSource`를 시작하면 클라이언트로의 데이터 전송을 시작합니다.

다음 단원들에서 세부 정보가 제공됩니다.

KinesisVideoClient의 인스턴스 생성

`createKinesisVideoClient` 작업을 호출하여 `KinesisVideoClient` 객체를 생성합니다.

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

`KinesisVideoClient`가 네트워크 호출을 하려면 인증을 위한 자격 증명이 필요합니다. `SystemPropertiesCredentialsProvider` 인스턴스를 전달하면, 인스턴스가 자격 증명 파일에 있는 기본 프로필의 `AWSCredentials`을 읽습니다.

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

MediaSource 인스턴스 생성

바이트를 Kinesis 비디오 스트림로 전송하려면 해당 데이터를 생성해야 합니다. Amazon Kinesis 비디오 스트림는 데이터 원본을 나타내는 `MediaSource` 인터페이스를 제공합니다.

예를 들어, Kinesis 비디오 스트림 Java 라이브러리가 `MediaSource` 인터페이스의 `ImageFileMediaSource` 구현을 제공합니다. 이 클래스는 Kinesis 비디오 스트림 이외의 미디어 파일 데이터만 읽지만 코드를 테스트할 때 이것을 사용할 수 있습니다.

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

클라이언트에 MediaSource 등록하기

클라이언트에 관해 알도록 생성한 미디어 원본을 `KinesisVideoClient`에 등록합니다(그런 다음 데이터를 클라이언트에 전송할 수 있습니다).

```
kinesisVideoClient.registerMediaSource(STREAM_NAME, bytesMediaSource);
```

미디어 원본 시작하기

미디어 원본이 데이터 생성 및 클라이언트로의 전송을 시작할 수 있도록 미디어 원본을 시작합니다.

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

다음 단계

the section called “3 단계: 코드 실행 및 확인” (p. 31)

3 단계: 코드 실행 및 확인

다음을 수행하여 Java 생산자 라이브러리를 위한 Java 테스트 도구를 실행합니다.

1. [DemoAppMain]을 선택합니다.
2. [Run]과 [Run 'DemoAppMain']을 선택합니다.
3. 애플리케이션용 JVM 인수에 자격 증명을 추가합니다.

   - 비-임시 AWS 자격 증명: `"-Daws.accessKeyId={YourAwsAccessKey} -Daws.secretKey={YourAwsSecretKey} -Djava.library.path={NativeLibraryPath}"`
   - 임시 AWS 자격 증명: `"-Daws.accessKeyId={YourAwsAccessKey} -Daws.secretKey={YourAwsSecretKey} -Daws.sessionToken={YourAwsSessionToken} -Djava.library.path={NativeLibraryPath}"`
4. AWS Management 콘솔에 로그인하고 Kinesis 비디오 스트림 콘솔을 엽니다.

   [Manage streams] 페이지에서 스트림을 선택합니다.
5. 내장 플레이어에서 샘플 비디오가 재생됩니다. 프레임이 쌓여 비디오가 재생될 때까지 잠깐 기다려야 할 수 있습니다(일반적인 대역폭과 프로세서 상태에서 최대 10초).

코드 예제는 스트림을 생성합니다. 코드에 있는 `MediaSource`가 시작되면 샘플 프레임을 `KinesisVideoClient`로 전송하기 시작합니다. 그런 다음 클라이언트가 데이터를 Kinesis 비디오 스트림에 전송합니다.

Android 생산자 라이브러리 사용

Amazon Kinesis 비디오 스트림는 Android 생산자 라이브러리를 제공하며, 이를 사용하여 미디어 데이터를 Android 디바이스에서 Kinesis 비디오 스트림로 전송하는 애플리케이션 코드를 최소한의 구성으로 작성할 수 있습니다.

애플리케이션이 Kinesis 비디오 스트림에 대한 데이터 스트리밍을 시작할 수 있으려면 다음 단계를 수행하여 코드를 Kinesis 비디오 스트림와 통합해야 합니다.

1. `KinesisVideoClient` 객체의 인스턴스를 생성합니다.
2. 미디어 원본 정보를 제공하여 `MediaSource` 객체를 생성합니다. 예를 들어 카메라 미디어 원본을 생성할 때 카메라를 식별하고 카메라가 사용하는 인코딩을 지정하는 등 정보를 제공합니다.

   스트리밍을 시작하고자 할 때 사용자 지정 미디어 원본을 생성해야 합니다.

절차: Android 생산자 SDK 사용

이 절차는 Android 애플리케이션에서 Kinesis 비디오 스트림 Android 생산자 클라이언트를 사용하여 데이터를 Kinesis 비디오 스트림에 전송하는 방법을 보여 줍니다.

이 절차에는 다음 단계가 포함됩니다.

- 코드 다운로드 및 구성
- 코드 검사
- 코드 실행 및 확인

사전 조건

- 애플리케이션 코드 검사, 편집, 실행에는 Android Studio를 권장합니다. 2017년 10월에 릴리스된 최신 버전 3.0.0 이상을 권장합니다.
- 샘플 코드에서 Amazon Cognito 자격 증명을 제공합니다. 다음 절차에 따라 Amazon Cognito 사용자 풀과 자격 증명 풀을 설정합니다.

  사용자 풀을 설정하려면

  1. Amazon Cognito 콘솔에 로그인합니다.
  2. [Manage your User Pools]를 선택합니다.
  3. [Create a user pool]을 선택합니다.
  4. [Pool name]의 값(예: **`<username>_android_user_pool`**)을 입력합니다.
  5. [Review defaults]를 선택합니다.
  6. [Create pool]을 선택합니다.
  7. [Pool Id] 값을 복사하고 저장합니다. 예제 애플리케이션을 구성할 때 이 값이 필요합니다.
  8. 풀 페이지에서 [App clients]를 선택합니다.
  9. [Add an app client]를 선택합니다.
  10. [App client name]의 값(예: **`<username>_android_app_client`**)을 입력합니다.
  11. [Create app client]를 선택합니다.

12. [Show Details]를 선택한 다음 [App client ID]와 [App client secret]을 복사하고 저장합니다. 예제 애플리케이션을 구성할 때 이 값들이 필요합니다.

자격 증명 풀을 설정하려면

1. Amazon Cognito 콘솔을 엽니다.
2. Manage Identity Pools(자격 증명 풀 관리)를 선택합니다.
3. [Create new identity pool]을 선택합니다.
4. [Identity pool name]의 값(예: **`<username>_android_identity_pool`**)을 입력합니다.
5. [Authentication providers] 섹션을 확장합니다. [Cognito] 탭에서 이전 절차의 [User Pool ID] 및 [App client ID] 값을 추가합니다.
6. [Create pool]을 선택합니다.
7. 다음 페이지에서 [Show Details] 섹션을 확장합니다.
8. [Auth_Role]로 끝나는 [Role name] 값이 있는 섹션에서 [View Policy Document]를 선택합니다.
9. [Edit]를 클릭하고 나타나는 [Edit Policy] 대화 상자를 확인합니다. 그러고 나서 다음 JSON을 복사하여 편집기에 붙여넣습니다.

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

10. [Allow]를 선택합니다.
11. 다음 페이지에서 [Get AWS Credentials] 코드 조각의 [Identity pool ID] 값을 복사하고 저장합니다. 예제 애플리케이션을 구성할 때 이 값이 필요합니다.

1단계: Android 생산자 라이브러리 코드 다운로드 및 구성

Android 생산자 라이브러리 절차의 이 단원에서는 Android 예제 코드를 다운로드하고 Android Studio에서 프로젝트를 엽니다.

사전 요건과 이 예제에 관한 기타 세부 정보는 Android 생산자 라이브러리 사용을 참조하십시오.

1. 디렉터리를 생성한 다음 GitHub 리포지토리에서 AWS Android SDK를 복제합니다.

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

2. Android Studio를 엽니다.
3. 열리는 화면에서 [Open an existing Android Studio project]를 선택합니다.
4. `aws-sdk-android-samples/AmazonKinesisVideoDemoApp` 디렉터리로 이동하여 [OK]를 선택합니다.

5. `AmazonKinesisVideoDemoApp/src/main/res/raw/awsconfiguration.json` 파일을 엽니다.

   `CredentialsProvider` 노드에서 사전 요건 단원의 [To set up an identity pool] 절차의 자격 증명 풀 ID를 제공하고, AWS 리전(예: **`us-west-2`**)을 제공합니다.

   `CognitoUserPool` 노드에서 사전 요건 단원의 [To set up a user pool] 절차의 App client secret, App client ID, Pool ID를 제공하고, AWS 리전(예: **`us-west-2`**)을 제공합니다.

6. `awsconfiguration.json` 파일은 다음과 비슷합니다.

```
{
  "Version": "1.0",
  "CredentialsProvider": {
    "CognitoIdentity": {
      "Default": {
        "PoolId": "us-west-2:01234567-89ab-cdef-0123-456789abcdef",
        "Region": "us-west-2"
      }
    }
  },
  "IdentityManager": {
    "Default": {}
  },
  "CognitoUserPool": {
    "Default": {
      "AppClientSecret": "abcdefghijklmnopqrstuvwxyz0123456789abcdefghijklmno",
      "AppClientId": "0123456789abcdefghijklmnop",
      "PoolId": "us-west-2_qRsTuVwXy",
      "Region": "us-west-2"
    }
  }
}
```

7. 해당 리전에서 `StreamingFragment.java` 파일을 업데이트합니다.

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

   AWS 리전 상수의 경우 리전을 참고하십시오.

다음 단계

the section called “2단계: 코드 검사” (p. 34)

2단계: 코드 검사

Android 생산자 라이브러리 절차의 이 단원에서는 예제 코드를 검사합니다.

Android 테스트 애플리케이션(`AmazonKinesisVideoDemoApp`)은 다음과 같은 코딩 패턴을 보입니다.

- `KinesisVideoClient`의 인스턴스를 만듭니다.
- `MediaSource`의 인스턴스를 만듭니다.
- 스트리밍을 시작합니다. 즉, `MediaSource`를 시작하면 클라이언트로의 데이터 전송을 시작합니다.

다음 단원들에서 세부 정보가 제공됩니다.

KinesisVideoClient의 인스턴스 생성

`createKinesisVideoClient` 작업을 호출하여 `KinesisVideoClient` 객체를 생성합니다.

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

`KinesisVideoClient`가 네트워크 호출을 하려면 인증을 위한 자격 증명이 필요합니다. `AWSCredentialsProvider`의 인스턴스를 전달하면 이전 단원에서 수정한 `awsconfiguration.json` 파일의 Amazon Cognito 자격 증명을 이 인스턴스가 읽습니다.

MediaSource 인스턴스 생성

바이트를 Kinesis 비디오 스트림로 전송하려면 해당 데이터를 생성해야 합니다. Amazon Kinesis 비디오 스트림는 데이터 원본을 나타내는 `MediaSource` 인터페이스를 제공합니다.

예를 들어 Kinesis 비디오 스트림 Android 라이브러리가 `MediaSource` 인터페이스의 `AndroidCameraMediaSource` 구현을 제공합니다. 이 클래스는 디바이스의 카메라 중 하나에서 데이터를 읽습니다.

다음 코드 예제(`fragment/StreamConfigurationFragment.java` 파일의 예제)에서 미디어 원본의 구성이 생성됩니다.

```
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().isEndcoderHardwareAccelerated())
                .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));
}
```

다음 코드 예제(`fragment/StreamingFragment.java` 파일의 예제)에서 미디어 원본이 생성됩니다.

```
mCameraMediaSource = (AndroidCameraMediaSource) mKinesisVideoClient
    .createMediaSource(mStreamName, mConfiguration);
```

미디어 원본 시작하기

미디어 원본이 데이터 생성 및 클라이언트로의 전송을 시작할 수 있도록 미디어 원본을 시작합니다. 다음 코드 예제는 `fragment/StreamingFragment.java` 파일의 예제입니다.

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

다음 단계

the section called “3 단계: 코드 실행 및 확인” (p. 36)

3 단계: 코드 실행 및 확인

Android 생산자 라이브러리용 Android 예제 애플리케이션을 실행하려면 다음을 수행합니다.

1. Android 디바이스에 연결합니다.
2. [Run], [Run...], [Edit configurations...]를 차례로 선택합니다.
3. [+]와 [Android App]을 차례로 선택합니다. [Name] 필드에 **`AmazonKinesisVideoDemoApp`**을 입력합니다. [Module] 풀다운에서 [AmazonKinesisVideoDemoApp]을 선택합니다. [OK]를 선택합니다.
4. [Run]과 [Run]을 차례로 선택합니다.
5. [Select a Deployment Target] 화면에서 연결된 디바이스를 선택하고 [OK]를 선택합니다.
6. 디바이스의 [AWSKinesisVideoDemoApp] 애플리케이션에서 [Create new account]를 선택합니다.
7. [USERNAME], [Password], [Given name], [Email address] 및 [Phone number]의 값을 입력한 다음 [Sign up]을 선택합니다.

   Note

   이러한 값의 제약 조건은 다음과 같습니다.

   - Password: 대문자, 소문자, 숫자, 특수 문자를 포함해야 합니다. Amazon Cognito 콘솔의 사용자 풀 페이지에서 이러한 제약 조건을 변경할 수 있습니다.
   - Email address: 확인 코드를 수신할 수 있는 유효한 주소여야 합니다.
   - Phone number: **`+<## ##><###>`** 형식이어야 합니다(예: **`+12065551212`**).
8. 이메일로 받은 코드를 입력하고 [Confirm]을 선택합니다. [OK]를 선택합니다.
9. 다음 페이지에서 기본값을 그대로 두고 [Stream]을 선택합니다.
10. AWS Management 콘솔에 로그인한 다음 미국 서부(오레곤) 지역의 https://console.aws.amazon.com/kinesisvideo/에서 Kinesis Video Streams 콘솔을 엽니다.

    [Manage Streams] 페이지에서 [demo-stream]을 선택합니다.
11. 내장 플레이어에서 스트리밍 비디오가 재생됩니다. 프레임이 쌓여 비디오가 재생될 때까지 잠깐 기다려야 할 수 있습니다(일반적인 대역폭과 프로세서 상태에서 최대 10초).

    Note

    디바이스의 화면이 회전하면 (예: 세로에서 가로로) 애플리케이션이 비디오 스트리밍을 중지합니다.

코드 예제는 스트림을 생성합니다. 코드에 있는 `MediaSource`가 시작되면 카메라에서 `KinesisVideoClient`로 프레임을 전송하기 시작합니다. 그러면 클라이언트가 [demo-stream]이라는 데이터를 Kinesis 비디오 스트림로 전송합니다.

C++ 생산자 라이브러리 사용하기

Amazon Kinesis 비디오 스트림는 C++ 생산자 라이브러리를 제공하며, 이를 사용하여 미디어 데이터를 디바이스에서 Kinesis 비디오 스트림로 전송하는 애플리케이션 코드를 쓸 수 있습니다.

객체 모델

C++ 라이브러리는 Kinesis 비디오 스트림에 대한 데이터 전송을 관리하는 다음과 같은 객체를 제공합니다.

- KinesisVideoProducer: 미디어 소스 및 AWS 자격 증명에 관한 정보를 포함하며 Kinesis 비디오 스트림 이벤트에 대해 보고하는 콜백을 유지합니다.
- KinesisVideoStream: Kinesis 비디오 스트림를 나타냅니다. 이름, 데이터 보존 기간, 미디어 콘텐츠 유형 등과 같은 비디오 스트림의 파라미터에 관한 정보를 포함합니다.

스트림에 대한 미디어 배포

C++ 라이브러리는 `KinesisVideoStream` 객체에 데이터를 배포하는 데 사용할 수 있는 메서드(예: `PutFrame`)를 제공합니다. 그런 다음 라이브러리는 내부 데이터 상태를 관리하는데, 다음과 같은 작업을 포함할 수 있습니다.

- 인증 수행.
- 네트워크 지연 시간 감시. 지연 시간이 너무 길면 라이브러리가 프레임 드롭을 선택할 수 있습니다.
- 진행 중인 스트리밍의 상태 추적.

콜백 인터페이스

이 계층은 일단의 콜백 인터페이스를 노출시키는데, 애플리케이션 계층에 알리는 데 사용됩니다. 이들 콜백 인터페이스에는 다음이 포함됩니다.

- 서비스 콜백 인터페이스(`CallbackProvider`): 이 라이브러리는 스트림을 생성하거나, 스트림 설명을 획득하거나, 스트림을 삭제할 때 이 인터페이스를 통해 획득되는 이벤트를 호출합니다.
- 클라이언트 준비 상태 또는 스토리지 부족 이벤트 인터페이스(`ClientCallbackProvider`): 라이브러리는 클라이언트가 준비 상태에 있거나 가용한 스토리지 혹은 메모리가 부족할 수 있음을 감지하면 이 인터페이스에서 이벤트를 호출합니다.
- 스트림 이벤트 콜백 인터페이스(`StreamCallbackProvider`): 라이브러리는 스트림이 준비 상태에 돌입하거나, 프레임이 끊기거나 스트림 오류가 발생하는 이벤트가 발생할 때 이 인터페이스에 이벤트를 호출합니다.

Kinesis 비디오 스트림는 이러한 인터페이스에 대한 기본적인 구현을 제공합니다. 예를 들어, 사용자 지정 네트워킹 로직을 필요로 하거나 낮은 스토리지 조건을 사용자 인터페이스에 노출시키고자 하는 경우 사용자 지정 구현을 제공할 수도 있습니다.

생산자 라이브러리의 콜백에 대한 자세한 내용은 생산자 SDK 콜백 (p. 81) 단원을 참조하십시오.

절차: C++ 생산자 SDK 사용

이 절차에서는 C++ 애플리케이션에서 Kinesis 비디오 스트림 클라이언트 및 미디어 소스를 사용하여 데이터를 Kinesis 비디오 스트림에 전송하는 방법을 보여 줍니다.

> **Note**
>
> C++ 라이브러리에는 macOS에 대한 샘플 빌드 스크립트가 포함됩니다. 현재 Windows에서는 C++ 생산자 라이브러리를 사용할 수 없습니다.
> Raspberry Pi 디바이스에서 C++ 생산자 라이브러리를 사용하려면 부록: Raspberry Pi에서 C++ 생산자 SDK 사용 (p. 44) 단원을 참조하십시오.

이 절차에는 다음 단계가 포함됩니다.

- 1 단계: 코드 다운로드 및 구성
- 2 단계: 코드 쓰기 및 검사
- 3 단계: 코드 실행 및 확인

사전 조건

- 자격 증명: 샘플 코드에서는 AWS 자격 증명 프로필 파일에서 설정한 프로필을 지정하여 자격 증명을 제공합니다. 아직 설정하지 않았다면 먼저 자격 증명 프로필을 설정합니다.

  자세한 내용은 개발을 위한 AWS 자격 증명 및 리전 설정을 참조하십시오.
- 인증서 스토어 통합: Kinesis 비디오 스트림 생산자 라이브러리는 호출하는 서비스의 신뢰성을 확인해야 합니다. 공용 인증서 스토어에서 인증 기관(CA)을 검증하여 확인합니다. Linux 기반 모델은 이 스토어가 `/etc/ssl/` 디렉터리에 위치합니다.

  다음 위치에서 인증 스토어로 인증서를 다운로드합니다.

  https://www.amazontrust.com/repository/SFSRootCAG2.pem
- 다음의 macOS용 빌드 종속성을 설치합니다.
  - Autoconf 2.69 (라이선스 GPLv3+/Autoconf: GNU GPL 버전 3 이상)
  - CMake 3.7 또는 3.8
  - Pkg-Config
  - Flex 2.5.35 Apple(flex-31) 이상
  - Bison 2.4 (GNU 라이선스)
  - Automake 1.15.1 (GNU 라이선스)
  - GNU Libtool(Apple Inc. version cctools-898)
  - xCode (macOS) / clang / gcc (xcode-select version 2347)
  - Java Development Kit (JDK) (Java JNI 컴파일용)
  - Lib-Pkg
- 다음의 Ubuntu용 빌드 종속성을 설치합니다(버전 명령에 대한 응답은 잘림).
  - Git를 설치합니다: `sudo apt-get install git`

    ```
    $ git --version
    git version 2.14.1
    ```
  - CMake를 설치합니다. `sudo apt-get install cmake`

    ```
    $ cmake --version
    cmake version 3.9.1
    ```
  - Libtool을 설치합니다. `sudo apt-get install libtool`

    ```
    2.4.6-2
    ```
  - libtool-bin을 설치합니다. `sudo apt-get install libtool-bin`

    ```
    $ libtool --version
    libtool (GNU libtool) 2.4.6
    Written by Gordon Matzigkeit, 1996
    ```
  - GNU Automake를 설치합니다. `sudo apt-get install automake`

    ```
    $ automake --version
    automake (GNU automake) 1.15
    ```
  - GNU Bison을 설치합니다. `sudo apt-get install bison`

    ```
    $ bison -V
    ```

```
bison (GNU Bison) 3.0.4
```

- G++를 설치합니다. `sudo apt-get install g++`

```
g++ --version
g++ (Ubuntu 7.2.0-8ubuntu3) 7.2.0
```

- curl을 설치합니다. `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
```

- pkg-config를 설치합니다. `sudo apt-get install pkg-config`

```
$ pkg-config --version
0.29.1
```

- Flex를 설치합니다. `sudo apt-get install flex`

```
$ flex --version
flex 2.6.1
```

- OpenJDK를 설치합니다. `sudo apt-get install openjdk-8-jdk`

```
$ java -version
openjdk version "1.8.0_171"
```

- `JAVA_HOME` 환경 변수를 설정합니다. `export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/`
- 빌드 스크립트를 실행합니다. `./install-script`

다음 단계

1 단계: C++ 생산자 라이브러리 코드 다운로드 및 구성

1 단계: C++ 생산자 라이브러리 코드 다운로드 및 구성

이 단원에서는 저수준 라이브러리를 다운로드하고 AWS 자격 증명을 사용하도록 애플리케이션을 구성합니다.

사전 요건과 이 예제에 관한 기타 세부 정보는 C++ 생산자 라이브러리 사용을 참조하십시오.

1. 디렉터리를 생성한 다음 GitHub 리포지토리에서 예제 소스 코드를 복제합니다.

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

2. 여러분이 선택한 통합 개발 환경(IDE)(예: Eclipse)에서 코드를 엽니다.
3. 명령줄에서 `ACCESS_KEY_ENV_VAR` 및 `SECRET_KEY_ENV_VAR` 환경 변수를 AWS 자격 증명에 설정합니다. 혹은 `ProducerTestFixture.h`의 다음 명령줄에서 AWS 자격 증명을 하드코딩할 수 있습니다.

```
        if (nullptr == (accessKey = getenv(ACCESS_KEY_ENV_VAR))) {
            accessKey = "AccessKey";
        }

        if (nullptr == (secretKey = getenv(SECRET_KEY_ENV_VAR))) {
            secretKey = "SecretKey";
```

```
        }
```

4. `tst/ProducerTestFixture.h`에서 `CreateStream`에 대한 호출을 찾습니다. 스트림 정의의 이름을 `ScaryTestStream2`에서 고유 이름으로 변경합니다.

```
shared_ptr<KinesisVideoStream> CreateTestStream(int index) {
        char stream_name[MAX_STREAM_NAME_LEN];
        sprintf(stream_name, "ScaryTestStream_%d", index);
```

다음 단계

2 단계: 코드 쓰기 및 검사 (p. 40)

2 단계: 코드 쓰기 및 검사

C++ 생산자 라이브러리 절차의 이 단원에서는 C++ 테스트 도구(`tst/ProducerTestFixture.h` 및 기타 다른 파일)에서 코드를 검사합니다. 이전 단원에서 이 코드를 다운로드했습니다.

비종속 플랫폼 C++ 예제는 다음과 같은 코딩 패턴을 보여 줍니다.

- Kinesis 비디오 스트림에 액세스할 `KinesisVideoProducer`의 인스턴스를 생성합니다.
- `KinesisVideoStream`의 인스턴스를 만듭니다. 이렇게 하면 동일한 이름의 스트림이 이미 존재하지 않는 경우, AWS 계정에 Kinesis 비디오 스트림가 생성됩니다.
- 가용한 경우 모든 데이터 프레임에 대해 `putFrame`을 `KinesisVideoStream`에 호출하여 스트림에 전송합니다.

다음 단원들에서 세부 정보가 제공됩니다:

KinesisVideoProducer의 인스턴스 생성

`KinesisVideoProducer::createSync` 메서드를 호출하여 `KinesisVideoProducer` 객체를 생성합니다. 다음 예제에서는 `ProducerTestFixture.h` 파일에서 `KinesisVideoProducer`를 생성합니다.

```
kinesis_video_producer_ = KinesisVideoProducer::createSync(move(device_provider_),
    move(client_callback_provider_),
    move(stream_callback_provider_),
    move(credential_provider_),
    defaultRegion_);
```

`createSync` 방법으로 다음 파라미터를 취합니다.

- 디바이스 또는 스토리지 구성에 관한 정보를 포함한 `DeviceInfo` 객체를 반환하는 `DeviceInfoProvider` 객체.

  Note

  `deviceInfo.storageInfo.storageSize` 파라미터를 사용하여 콘텐츠 스토어 크기를 구성합니다. 콘텐츠 스트림은 콘텐츠 스토어를 공유합니다. 스토리지 크기 요구 사항을 확인하려면 평균 프레임 크기에 모든 스트림의 최대 지속 시간 동안 저장된 프레임 수를 곱합니다. 그런 다음 조각 모음을 고려하여 1.2를 곱합니다. 예를 들어 애플리케이션에 다음 구성이 있는 것으로 가정하겠습니다.

  - 스트림 세 개
  - 최대 지속 시간 3분
  - 각 스트림은 초당 30프레임(FPS)
  - 프레임마다 10,000KB

이 애플리케이션의 콘텐츠 스토어 요구 사항은 3(스트림 수) * 3(분) * 60(초(1분)) * 10000(kb) * 1.2(조각 모음 허용량) = 194.4Mb ~ 200Mb입니다.

- 특정 클라이언트의 이벤트를 보고하는 함수 별칭을 반환하는 `ClientCallbackProvider` 객체.
- 특정 스트림의 이벤트가 발생할 때 콜백되는 함수 별칭을 반환하는 `StreamCallbackProvider` 객체.
- AWS 자격 증명 환경 변수에 대한 액세스를 제공하는 `CredentialProvider` 객체.
- AWS 리전("us-west-2"). 서비스 엔드포인트는 리전으로부터 결정됩니다.

KinesisVideoStream의 인스턴스 생성

`StreamDefinition` 파라미터로 `KinesisVideoProducer::CreateStream` 메서드를 호출하여 `KinesisVideoStream` 객체를 생성합니다. 예제에서는 `ProducerTestFixture.h` 파일에서 `KinesisVideoStream`를 생성합니다.

```
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` 객체에는 다음의 필드가 있습니다.

- 스트림 이름.
- 데이터 보존 기간.
- 스트림 태그. 이들 태그는 소비자 애플리케이션이 정확한 스트림을 찾거나 스트림에 관한 상세를 정보를 얻는데 사용할 수 있습니다. AWS Management 콘솔에서도 태그를 볼 수 있습니다.
- 스트림의 AWS KMS 암호화 키. 자세한 내용은 Kinesis 비디오 스트림와 함께 서버 측 암호화 사용을 참조하십시오.
- 스트리밍 유형. 현재 유일한 유효 값은 `STREAMING_TYPE_REALTIME`입니다.
- 미디어 콘텐츠 유형. 콘솔 뷰어에서 스트림을 보려면 이 값을 `"video/h264"`로 설정합니다.
- 미디어 지연 시간. 이 값은 현재 사용되지 않으며, 0으로 설정해야 합니다.
- 각 조각의 재생 시간.
- 미디어 타임코드 척도.
- 미디어가 키 프레임 조각을 사용하는지 여부.
- 미디어가 타임코드를 사용하는지 여부.
- 미디어가 절대 조각 시간을 사용하는지 여부.

Kinesis 비디오 스트림으로 프레임 배포

`KinesisVideoStream::putFrame`을 사용하여 미디어를 Kinesis 비디오 스트림에 배포하여 헤더 및 미디어 데이터를 포함하는 `Frame` 객체에 전달합니다. 예제에서는 `ProducerApiTest.cpp` 파일에서 `putFrame`를 호출합니다.

```
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

앞선 C++ 생산자 예제에서는 테스트 데이터의 버퍼를 전송합니다. 실제 애플리케이션에서는 미디어 소스(에: 카메라)의 프레임 데이터로부터 프레임 버퍼 및 크기를 알아야 합니다.

`Frame` 객체에는 다음의 필드가 있습니다.

- 프레임 인덱스. 이는 단순 점증 값이어야 합니다.
- 프레임과 관련된 플래그. 예를 들어, 인코더가 키 프레임을 생산하도록 구성되어 있다면 이 프레임에는 `FRAME_FLAG_KEY_FRAME` 플래그가 할당될 것입니다.
- 타임스탬프 디코딩.
- 프레젠테이션 타임스탬프.
- 프레임 지속 시간(최대 100 ns 단위).
- 프레임 크기(바이트).
- 프레임 데이터.

프레임의 형식에 대한 자세한 내용은 Kinesis 비디오 스트림 데이터 모델을 참조하십시오.

메트릭 및 메트릭 로깅

C++ 생산자 SDK에는 메트릭 및 메트릭 로깅 기능이 포함되어 있습니다.

`getKinesisVideoMetrics` 및 `getKinesisVideoStreamMetrics` API를 사용하여 Kinesis 비디오 스트림와 활성 스트림에 관한 정보를 검색할 수 있습니다.

다음은 `kinesis-video-pic/src/client/include/com/amazonaws/kinesis/video/client/Include.h` 파일의 코드입니다.

```
/**
* 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`로 채워지는 `PClientMetrics` 객체에는 다음 정보가 포함되어 있습니다.

- contentStoreSize: 콘텐츠 스토어의 전체 크기(바이트)(스트리밍 데이터 저장에 사용되는 메모리)입니다.
- contentStoreAvailableSize: 콘텐츠 스토어에서 사용 가능한 메모리(바이트)입니다.
- contentStoreAllocatedSize: 콘텐츠 스토어에 할당된 메모리입니다.
- totalContentViewsSize: 콘텐츠 보기에 사용되는 전체 메모리입니다. (콘텐츠 보기는 콘텐츠 스토어에 있는 일련의 정보 인덱스입니다.)
- totalFrameRate: 전체 활성 스트림에 누적되어 있는 초당 프레임 수입니다.
- totalTransferRate: 모든 스트림에서 전송 중인 전체 초당 비트(bps)입니다.

`getKinesisVideoStreamMetrics`로 채워지는 `PStreamMetrics` 객체에는 다음 정보가 포함되어 있습니다.

- currentViewDuration: 콘텐츠 보기의 헤드(프레임이 인코딩되어 있을 때)와 현재 위치(프레임 데이터가 Kinesis 비디오 스트림로 전송 중일 때) 사이의 100ns 단위 차이입니다.
- overallViewDuration: 콘텐츠 보기의 헤드(프레임이 인코딩되어 있을 때)와 테일(콘텐츠 보기에 할당된 전체 공간을 초과했거나 Kinesis 비디오 스트림에서 `PersistedAck` 메시지가 수신되고 지속되는 것으로 확인된 프레임이 플러싱되었기 때문에 프레임이 플러싱되었을 때) 사이의 100ns 단위 차이입니다.
- currentViewSize: 헤드(프레임이 인코딩되어 있을 때)부터 현재 위치(프레임이 Kinesis 비디오 스트림로 전송되었을 때)까지 콘텐츠 뷰의 크기(바이트)입니다.
- overallViewSize: 콘텐츠 보기의 총 크기(바이트)입니다.
- currentFrameRate: 스트림 레이트의 최종 측정값입니다(초당 프레임 단위).
- currentTransferRate: 스트림 레이트의 최종 측정값입니다(초당 바이트 단위).

다음 단계

the section called “3 단계: 코드 실행 및 확인” (p. 43)

3 단계: 코드 실행 및 확인

C++ 생산자 라이브러리 절차의 코드를 실행하고 확인하려면 다음을 수행합니다.

1. 자격 증명, 인증서 및 빌드 요구사항에 관한 내용은 필수 조건을 참조하십시오.
2. `/kinesis-video-native-build/install-script` 스크립트를 사용하여 프로젝트를 구축합니다. 설치 스크립트를 실행하면 다음 공개 소스 종속성이 설치됩니다.

   - curl lib
   - openssl (crypto 및 ssl)
   - log4cplus
   - jsoncpp

   Note

   log4cplus를 구성하려면 로깅 기능을 가리키도록 `PlatformUtils.h`에서 다음 값을 설정합니다.

```
#define __LOG(p1, p2, p3, ...)     printf(p3, ##__VA_ARGS__)
```

3. `kinesis-video-native-build/start`에 실행 파일이 구축됩니다. 이 파일을 시작하면 단위 테스트가 실행되고 더미 프레임 스트리밍이 시작됩니다.
4. 세부 정보 표시 로그를 활성화하려면 `CMakeList.txt`에서 해당 라인의 주석을 제거하여 `HEAP_DEBUG` 및 `LOG_STREAMING` C-정의를 정의합니다.

IDE의 디버그 출력에서 테스트 제품군의 진행 상황을 모니터링할 수 있습니다. 또한 Amazon CloudWatch 콘솔에서 스트림과 관련이 있는 지표(예: `PutMedia.IncomingBytes`)를 주시함으로써 스트림 상에서의 트래픽을 모니터링할 수 있습니다.

> **Note**
>
> 테스트 도구가 빈 프레임만을 전송하기 때문에 콘솔이 데이터를 비디오 스트림으로 표시하지 않습니다.

GStreamer 플러그인으로 C++ 생산자 SDK 사용

GStreamer는 수많은 카메라 및 비디오 소스에서 모듈식 플러그인을 결합하여 사용자 지정 미디어 파이프라인을 생성하는 데 사용하는 인기 있는 미디어 프레임워크입니다. Kinesis 비디오 스트림 GStreamer 플러그인은 기존 GStreamer 미디어 파이프라인과 Kinesis 비디오 스트림의 통합을 대폭 간소화합니다.

C++ 생산자 SDK를 GStreamer 플러그인으로 사용하는 방법에 대한 자세한 내용은 예제: Kinesis 비디오 스트림 생산자 SDK GStreamer 플러그인 (p. 93) 단원을 참조하십시오.

도커 컨테이너에서 GStreamer 플러그인으로 C++ 생산자 SDK 사용

GStreamer는 수많은 카메라 및 비디오 소스에서 모듈식 플러그인을 결합하여 사용자 지정 미디어 파이프라인을 생성하는 데 사용하는 인기 있는 미디어 프레임워크입니다. Kinesis 비디오 스트림 GStreamer 플러그인은 기존 GStreamer 미디어 파이프라인과 Kinesis 비디오 스트림의 통합을 대폭 간소화합니다.

또한 도커를 사용하여 GStreamer 파이프라인을 생성하면 Kinesis 비디오 스트림에 대한 운영 환경이 표준화되어 애플리케이션 구축 및 실행이 대폭 간소화됩니다.

Docker 컨테이너에서 C++ 생산자 SDK를 GStreamer 플러그인으로 사용하는 방법에 대한 자세한 내용은 도커 컨테이너에서 GStreamer 요소 실행 (p. 96) 단원을 참조하십시오.

부록: Raspberry Pi에서 C++ 생산자 SDK 사용

Raspberry Pi는 기본 컴퓨터 프로그래밍 기술을 가르치고 학습하는 데 사용할 수 있는 저렴한 소형 컴퓨터입니다. 이 자습서에서는 Raspberry Pi 디바이스에서 Amazon Kinesis 비디오 스트림 C++ 생산자 SDK를 설치 및 사용하는 방법에 대해 설명합니다. 또한 이러한 단계에는 GStreamer 데모 애플리케이션을 사용하여 설치를 확인하는 방법도 포함되어 있습니다.

항목

- Kinesis 비디오 스트림 C++ 생산자 SDK 다운로드 및 빌드 (p. 48)
- Kinesis 비디오 스트림으로 비디오 스트리밍 및 라이브 스트림 보기 (p. 48)

사전 조건

Raspberry Pi에서 C++ 생산자 SDK를 설치하기 전에 다음 사전 조건을 갖춰야 합니다.

- 다음과 같이 구성된 Raspberry Pi 디바이스:
  - 보드 버전: 3 모델 B 이상
  - 연결된 카메라 모듈
  - 8GB 이상의 용량이 남아 있는 SD 카드
  - 설치된 Raspbian 운영 체제(커널 버전 4.9 이상) 최신 Raspbian 이미지는 Raspberry Pi Foundation 웹 사이트에서 다운로드할 수 있습니다. Raspberry Pi 지침에 따라 다운로드한 이미지를 SD 카드에 설치합니다.
- Kinesis 비디오 스트림를 사용하는 AWS 계정. 자세한 내용은 Kinesis 비디오 스트림 시작하기를 참조하십시오.

Kinesis 비디오 스트림에 쓰기 권한으로 IAM 사용자 생성

아직 수행하지 않은 경우 Kinesis 비디오 스트림에 쓰기 권한으로 AWS Identity and Access Management(IAM) 사용자를 설정합니다.

1. AWS Management 콘솔에 로그인한 다음 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 왼쪽 탐색 메뉴에서 사용자를 클릭합니다.
3. 새 사용자를 생성하려면 사용자 추가를 선택합니다.
4. 해당 사용자에 대해 설명이 포함된 사용자 이름을 지정합니다(예: **`kinesis-video-raspberry-pi-producer`**).
5. 액세스 유형에서 프로그래밍 방식 액세스를 선택합니다.
6. Next: Permissions(다음: 권한)를 선택합니다.
7. kinesis-video-raspberry-pi-producer에 대한 권한 설정에서 기존 정책 직접 연결을 선택합니다.
8. [Create policy]를 선택합니다. 새 웹 브라우저 탭에서 정책 생성 페이지가 열립니다.
9. [JSON] 탭을 선택합니다.
10. 다음 JSON 정책을 복사해 텍스트 영역에 붙여 넣습니다. 이 정책은 데이터를 생성하여 Kinesis 비디오 스트림s에 쓸 수 있는 사용자 권한을 제공합니다.

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

11. [Review policy]를 선택합니다.

12. 정책에 이름을 지정합니다(예: **kinesis-video-stream-write-policy**).
13. [Create policy]를 선택합니다.
14. 브라우저에서사용자 추가 탭으로 돌아가서 새로 고침을 선택합니다.
15. 검색 상자에 생성한 정책 이름을 입력합니다.
16. 목록에서 새 정책 옆의 확인란을 선택합니다.
17. [Next: Review]를 선택합니다.
18. Create user를 선택합니다.
19. 콘솔에 새 사용자의 액세스 키 ID가 표시됩니다. 표시를 선택해 보안 액세스 키를 표시합니다. 애플리케이션을 구성할 때 필요하므로 이러한 값을 기록해 둡니다.

Raspberry Pi를 Wi-Fi 네트워크에 연결

헤드리스 모드에서 즉, 연결된 키보드, 모니터 또는 네트워크 케이블 없이 Raspberry Pi를 사용할 수 있습니다. 연결된 모니터와 키보드를 사용하는 경우에는 Raspberry Pi 카메라 구성 (p. 47).

1. 컴퓨터에서 `wpa_supplicant.conf`라는 이름의 파일을 하나 만듭니다.
2. 다음 텍스트를 복사해 `wpa_supplicant.conf` 파일에 붙여넣습니다(또는 sample wpa_supplicant.conf 파일 다운로드).

```
country=US
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

network={
ssid="<YOUR_WIFI_SSID>"
scan_ssid=1
key_mgmt=WPA-PSK
psk="<YOUR_WIFI_PASSWORD>"
}
```

   Wi-Fi 네트워크에 대한 정보로 `ssid` 및 `psk` 값을 바꿉니다.
3. `wpa_supplicant.conf` 파일을 SD 카드로 복사합니다. `boot` 볼륨의 루트에 복사해야 합니다.
4. SD 카드를 Raspberry Pi에 삽입하고 디바이스의 전원을 켭니다. 그러면 디바이스가 Wi-Fi 네트워크에 연결되고 SSH가 활성화됩니다.

Raspberry Pi에 원격으로 연결

헤드리스 모드에서 Raspberry Pi에 원격으로 연결할 수 있습니다. 모니터와 키보드를 연결한 상태에서 Raspberry Pi를 사용하는 경우 Raspberry Pi 카메라 구성 (p. 47).

1. Raspberry Pi 디바이스에 원격으로 연결하기 전에 다음 중 하나를 수행하여 이 디바이스의 IP 주소를 확인합니다.

   - 네트워크의 Wi-Fi 라우터에 액세스할 수 있는 경우 연결된 Wi-Fi 디바이스를 찾습니다. `Raspberry Pi`라는 디바이스를 찾아 디바이스의 IP 주소를 확인합니다.
   - 네트워크의 Wi-Fi 라우터에 액세스할 수 없는 경우에는 다른 소프트웨어를 사용하여 네트워크의 디바이스를 찾을 수 있습니다. Fing은 Android 및 iOS 디바이스에서 사용할 수 있는 널리 사용되는 애플리케이션입니다. 이 애플리케이션의 무료 버전을 사용하여 네트워크에 있는 디바이스의 IP 주소를 찾을 수 있습니다.
2. Raspberry Pi 디바이스의 IP 주소를 알고 있는 경우에는 임의의 터미널 애플리케이션을 사용해 연결할 수 있습니다.

   - macOS 또는 Linux에서는 `ssh`를 사용합니다.

```
$ ssh pi@<IP address>
```

- Windows에서는 Windows용 무료 SSH 클라이언트인 PuTTY를 사용합니다.

Raspbian을 새로 설치할 경우 사용자 이름은 **pi**이고 암호는 **raspberry**입니다. 기본 암호를 변경하는 것이 좋습니다.

Raspberry Pi 카메라 구성

아래 단계에 따라 디바이스에서 Kinesis 비디오 스트림로 비디오를 전송하도록 Raspberry Pi 카메라를 구성합니다.

1. 편집기를 열고 다음 명령을 사용해 `modules` 파일을 업데이트합니다.

```
$  sudo nano /etc/modules
```

2. 없는 경우 이 파일의 끝에 다음 행을 추가합니다.

```
bcm2835-v4l2
```

3. 파일을 저장하고 편집기를 종료합니다(Ctrl-X).
4. Raspberry Pi를 재부팅합니다.

```
$  sudo reboot
```

5. 원격으로 연결되어 있는 경우 디바이스가 재부팅되면 터미널 애플리케이션을 통해 다시 연결합니다.
6. Open `raspi-config`:

```
$  sudo raspi-config
```

7. Interfacing Options(인터페이스 옵션), 카메라를 선택합니다. 아직 활성화되지 않은 경우 카메라를 활성화하고 메시지가 표시되면 재부팅합니다.
8. 다음 명령을 입력하여 카메라가 작동하는지 확인합니다.

```
$  raspistill -v -o test.jpg
```

디스플레이에 카메라의 5초짜리 미리 보기가 표시되고 그림을 촬영하고(`test.jpg`로 저장), 정보 메시지를 표시합니다.

소프트웨어 설치 사전 조건

C++ 생산자 SDK를 사용하려면 Raspberry Pi에 다음 소프트웨어 사전 조건을 설치해야 합니다.

1. Git를 설치합니다:

```
$  sudo apt-get update
$  sudo apt-get install git
```

2. Yacc, Lex 및 OpenJDK(Open Java Development Kit)를 설치합니다.

```
$  sudo apt-get install byacc flex
```

```
$  sudo apt-get install openjdk-8-jdk
```

3. JAVA_HOME 환경 변수를 설정합니다.

```
$  export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-armhf/
```

Note

SDK를 빌드하기 전에 디바이스를 재부팅한 경우 이 단계를 반복해야 합니다. 또한 ~/.profile 파일에서 환경 변수를 설정할 수도 있습니다.

4. CMake는 SDK를 빌드하는 데 사용됩니다. 다음 명령으로 CMake를 설치합니다.

```
$  sudo apt-get install cmake
```

5. 다음 PEM 파일을 /etc/ssl/cert.pem에 복사합니다.

https://www.amazontrust.com/repository/SFSRootCAG2.pem

Kinesis 비디오 스트림 C++ 생산자 SDK 다운로드 및 빌드

1. C++ 생산자 SDK를 설치합니다.

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

2. 현재 작업 디렉터리를 설치 디렉터리로 변경합니다.

```
$  cd amazon-kinesis-video-stream-producer-sdk-cpp/kinesis-video-native-build
```

3. 다음과 같이 설치 스크립트 실행 파일을 만듭니다.

```
$  chmod +x install-script
```

4. 설치 스크립트를 실행합니다. 설치 스크립트가 소스를 다운로드하고 여러 오픈 소스 프로젝트를 빌드합니다. 처음에 실행하면 마칠 때까지 몇 시간 정도 소요될 수 있습니다.

```
$  ./install-script
```

5. Y를 입력해 확인합니다. 그런 다음 빌드 스크립트를 실행합니다.

Kinesis 비디오 스트림으로 비디오 스트리밍 및 라이브 스트림 보기

1. 샘플 애플리케이션을 실행하려면 다음 정보가 필요합니다.
   - 사전 조건 (p. 45) 단원에서 생성한 스트림 이름입니다.
   - Kinesis 비디오 스트림에 쓰기 권한으로 IAM 사용자 생성 (p. 45) 단원에서 생성한 계정 자격 증명(액세스 키 ID 및 보안 액세스 키)입니다.
2. 다음 명령을 사용하여 샘플 애플리케이션을 실행합니다.

```
$ export AWS_ACCESS_KEY_ID=<Access Key ID>
export AWS_SECRET_ACCESS_KEY=<Secret Access Key>
./kinesis_video_gstreamer_sample_app Stream Name
```

3. 다음과 같이 이미지 크기, 프레임 속도 및 비트 전송률을 지정할 수 있습니다.

```
$ export AWS_ACCESS_KEY_ID=<Access Key ID>
export AWS_SECRET_ACCESS_KEY=<Secret Access Key>
./kinesis_video_gstreamer_sample_app -w <width> -h <height> -f <framerate>
                    -b <bitrateInKBPS> Stream Name
```

4. 샘플 애플리케이션에서 `library not found` 오류가 발생하면 다음 명령을 입력하여 프로젝트가 오픈 소스 종속성에 올바르게 연결되어 있는지 확인합니다.

```
$  rm -rf ./kinesis-video-native-build/CMakeCache.txt ./kinesis-video-native-build/
CMakeFiles
$  ./kinesis-video-native-build/install-script
```

5. https://console.aws.amazon.com/kinesisvideo/에서 Kinesis 비디오 스트림 콘솔을 엽니다.
6. 생성한 스트림의 스트림 이름을 선택합니다.

Raspberry Pi에서 전송된 비디오 스트림이 콘솔에 나타납니다.

스트림이 재생 중인 경우 Kinesis 비디오 스트림 콘솔의 다음 기능을 시험할 수 있습니다.

- Video preview(비디오 미리 보기) 섹션에서 탐색 컨트롤을 사용하여 스트림을 되감거나 앞으로 빨리 재생합니다.
- Stream info(스트림 정보) 섹션에서 스트림의 코덱, 해상도 및 비트 전송률을 확인합니다. Raspberry Pi에서는 이 자습서를 진행하기 위한 대역폭 사용량을 최소화하기 위해 해상도와 비트 전송률 값을 의도적으로 낮게 설정합니다. 스트림에 대해 생성 중인 Amazon CloudWatch 지표를 보려면 View stream metrics in CloudWatch(CloudWatch에서 스트림 지표 보기)를 선택합니다.
- 데이터 보존 기간에서는 비디오 스트림이 1일 동안 보존된다고 표시되어 있습니다. 이 값을 편집해 No data retention(데이터 보존 안 함)으로 설정하거나 이 값을 1일~며칠로 설정할 수 있습니다.

  서버 측 암호화에는 AWS Key Management Service(AWS KMS)에서 유지 관리하는 키를 사용하여 데이터가 저장된 상태에서 암호화 중이라고 표시됩니다.

C++ 생산자 SDK 로깅 사용

`kinesis-video-native-build` 폴더의 `kvs_log_configuration` 파일에서 C++ 생산자 SDK 애플리케이션에 대한 로깅을 구성합니다.

다음 예제는 `DEBUG` 수준 로그 항목을 AWS Management 콘솔에 기록하는 애플리케이션을 구성하는 기본 구성 파일의 첫 번째 줄을 보여줍니다.

```
log4cplus.rootLogger=DEBUG, KvsConsoleAppender
```

로깅 수준을 `INFO`로 설정하여 로깅 상세도를 낮출 수 있습니다.

또한 애플리케이션이 로그 항목을 로그 파일에 기록하도록 구성하려면 파일의 첫 번째 줄을 다음과 같이 수정합니다.

```
log4cplus.rootLogger=DEBUG, KvsConsoleAppender, KvsFileAppender
```

그러면 애플리케이션이 `kinesis-video-native-build/log` 폴더의 `kvs.log`에 로그 항목을 기록하도록 구성됩니다.

로그 파일 위치를 변경하려면 다음 줄을 새 경로로 수정합니다.

```
log4cplus.appender.KvsFileAppender.File=./log/kvs.log
```

Note

DEBUG 수준 로깅이 파일에 기록될 경우 로그 파일이 디바이스의 가용 저장 공간을 빠르게 소모할 수 있습니다.

생산자 SDK 참조

이 단원에는 Kinesis 비디오 스트림 생산자 라이브러리 (p. 27)에 대한 제한 사항, 오류 코드 및 기타 참조 정보가 들어 있습니다.

항목

생산자 SDK 제한 사항

다음 표에는 생산자 라이브러리 (p. 27)의 값에 대한 최신 제한 사항이 나와 있습니다.

Note

이러한 값을 설정하기 전에 입력을 확인해야 합니다. SDK는 이러한 제한을 확인하지 않으며 제한이 초과되면 실행 시간 오류가 발생합니다.

값	한도	참고
최대 스트림 개수	128	생산자 객체에서 생성할 수 있는 최대 스트림 개수입니다. 이 제한은 소프트 제한으로, 제한을 늘리도록 요청할 수 있습니다. 생산자가 실수로 스트림을 재귀적으로 생성하면 안 됩니다.
디바이스의 최대 이름 길이	128자	
최대 태그 개수	스트림당 50개	
최대 스트림 이름 길이	256자	
최소 스토리지 크기	10MiB = 10 * 1024 * 1024바이트	
최대 스토리지 크기	10 GiB = 10 * 1024 * 1024 * 1024 바이트	
최대 루트 디렉터리 경로 길이	4,096자	
최대 인증 정보 길이	10,000 bytes	
최대 URI 문자열 길이	10,000자	
최대 태그 이름 길이	128자	
최대 태그 값 길이	1,024자	

값	한도	참고
최소 보안 토큰 기간	30초	
보안 토큰 유예 기간	40 minutes	지정된 기간이 더 길면 이 값으로 제한됩니다.
보존 기간	0 또는 한 시간 이상	0은 보관하지 않음을 나타냅니다.
최소 클러스터 기간	1초	이 값은 SDK 표준인 100ns 단위로 지정됩니다.
최대 클러스터 기간	30초	이 값은 SDK 표준인 100ns 단위로 지정됩니다. 백엔드 API가 더 짧은 클러스터 기간을 강제로 적용할 수 있습니다.
최대 조각 크기	50MB	자세한 내용은 Kinesis 비디오 스트림 제한 (p. 116) 단원을 참조하십시오.
최대 조각 기간	10초	자세한 내용은 Kinesis 비디오 스트림 제한 (p. 116) 단원을 참조하십시오.
최대 연결 기간	45 minutes	이 시간이 경과하면 백엔드가 연결을 닫습니다. SDK는 이 시간 이내에 토큰을 교체하고 새 연결을 설정합니다.
최대 ACK 세그먼트 길이	1,024자	ACK 구문 분석기 함수로 전송된 승인의 최대 세그먼트 길이입니다.
최대 콘텐츠 유형 문자열 길이	128자	
최대 코덱 ID 문자열 길이	32자	
최대 추적 이름 문자열 길이	32자	
최대 코덱 프라이빗 데이터 길이	1MiB = 1 * 1024 * 1024바이트	
최소 타임코드 스케일 값 길이	100ns	결과 MKV 클러스터에서 프레임 타임스탬프를 나타내는 최소 타임코드 스케일 값입니다. 이 값은 SDK 표준인 100ns 단위씩 증분되어 지정됩니다.
최대 타임코드 스케일 값 길이	1초	결과 MKV 클러스터에서 프레임 타임스탬프를 나타내는 최대 타임코드 스케일 값입니다. 이 값은 SDK 표준인 100ns 단위씩 증분되어 지정됩니다.
최소 콘텐츠 보기 항목 수	10	
최소 버퍼 기간	20초	이 값은 SDK 표준인 100ns 단위씩 증분되어 지정됩니다.
최대 업데이트 버전 길이	128자	

값	한도	참고
최대 ARN 길이	1024자	
최대 조각 시퀀스 길이	128자	
최대 보존 기간	10년	

오류 코드 참조

이 섹션에는 생산자 라이브러리 (p. 27)에 대한 오류 및 상태 코드 정보가 포함되어 있습니다.

공통 문제 해결 방법에 대한 자세한 내용은 Kinesis 비디오 스트림 문제 해결 (p. 119) 단원을 참조하십시오.

PutFrame 콜백에 의해 반환되는 오류 및 상태 코드

다음 단원에서는 `PutFrame` 연산에 대한 콜백을 통해 반환되는 오류와 상태 정보를 설명합니다.

항목

- 클라이언트 라이브러리에 의해 반환되는 오류 및 상태 코드 (p. 52)
- 기간 라이브러리에 의해 반환되는 오류 및 상태 코드 (p. 63)
- 공통 라이브러리에 의해 반환되는 오류 및 상태 코드 (p. 63)
- 힙 라이브러리에 의해 반환되는 오류 및 상태 코드 (p. 64)
- MKVGen 라이브러리에 의해 반환되는 오류 및 상태 코드 (p. 65)
- Trace 라이브러리에 의해 반환되는 오류 및 상태 코드 (p. 68)
- Utils 라이브러리에 의해 반환되는 오류 및 상태 코드 (p. 68)
- 뷰 라이브러리에 의해 반환되는 오류 및 상태 코드 (p. 69)

클라이언트 라이브러리에 의해 반환되는 오류 및 상태 코드

다음 표에는 Kinesis 비디오 스트림 `Client` 라이브러리의 메서드에서 반환하는 오류 및 상태 정보가 나와 있습니다.

코드	메시지	설명	권장 조치
`0x52000001`	`STATUS_MAX_STREAM_COUNT`	최대 스트림 수에 도달함.	생산자 SDK 제한 사항 (p. 50)에 지정된 대로 `DeviceInfo`에서 최대 스트림 개수를 늘려 지정합니다.
`0x52000002`	`STATUS_MIN_STREAM_COUNT`	최소 스트림 수 오류.	`DeviceInfo`에서 최대 스트림 수를 0보다 크게 지정합니다.
`0x52000003`	`STATUS_INVALID_DEVICE_NAME_LENGTH`	잘못된 디바이스 이름 길이.	생산자 SDK 제한 사항 (p. 50)에 문자 수 단위로 지정된 최대 디바이스 이름 길이를 참조하십시오.
`0x52000004`	`STATUS_INVALID_DEVICE_INFO_VERSION`	잘못된 `DeviceInfo` 구조 버전.	구조의 올바른 현재 버전을 지정합니다.

코드	메시지	설명	권장 조치
0x52000005	STATUS_MAX_TAG_COUNT	최대 태그 개수에 도달함.	생산자 SDK 제한 사항 (p. 50)에 지정된 현재 최대 태그 개수를 참조하십시오.
0x52000006	STATUS_DEVICE_FINGERPRINT_LENGTH		
0x52000007	STATUS_INVALID_CALLBACKS_VERSION	잘못된 Callbacks 구조 버전.	구조의 올바른 현재 버전을 지정합니다.
0x52000008	STATUS_INVALID_STREAM_INFO_VERSION	잘못된 StreamInfo 구조 버전.	구조의 올바른 현재 버전을 지정합니다.
0x52000009	STATUS_INVALID_STREAM_NAME_LENGTH	잘못된 스트림 이름 길이.	생산자 SDK 제한 사항 (p. 50)에 문자 수 단위로 지정된 최대 스트림 이름 길이를 참조하십시오.
0x5200000a	STATUS_INVALID_STORAGE_SIZE	잘못된 스토리지 크기가 지정됨.	바이트 단위의 스토리지 크기는 생산자 SDK 제한 사항 (p. 50)에 지정된 제한을 벗어나면 안 됩니다.
0x5200000b	STATUS_INVALID_ROOT_DIRECTORY_LENGTH	잘못된 루트 디렉터리 문자열 길이.	생산자 SDK 제한 사항 (p. 50)에 지정된 최대 루트 디렉터리 경로 길이를 참조하십시오.
0x5200000c	STATUS_INVALID_SPILL_RATIO	잘못된 유출 비율.	유출 비율을 0~100의 백분율로 표시합니다.
0x5200000d	STATUS_INVALID_STORAGE_INFO_VERSION	잘못된 StorageInfo 구조 버전.	구조의 올바른 현재 버전을 지정합니다.
0x5200000e	STATUS_INVALID_STREAM_STATE	스트림이 현재 작업을 허용하지 않는 상태입니다.	일반적으로 이 오류는 SDK가 요청된 작업을 수행해야 하는 상태에 도달하지 못한 경우 발생합니다. 예를 들어 `GetStreamingEndpoint` API 호출이 실패했는데 클라이언트 애플리케이션이 이를 무시하고 스트림에 프레임을 계속 넣으려고 할 경우 이 오류가 발생합니다.
0x5200000f	STATUS_SERVICE_CALL_CALLBACKS_MISSING	콜백 구조에 일부 필수 함수에 대한 함수 진입점이 누락됨.	필수 콜백이 클라이언트 애플리케이션 내에서 구현되는지 확인하십시오. 이 오류는 PIC(Platform Independent Code) 클라이언트에만 표시됩니다. C++ 및 기타 더 높은 수준의 래퍼는 이러한 호출을 충족합니다.

코드	메시지	설명	권장 조치
0x52000010	STATUS_SERVICE_CALL_NOT_AUTHORIZED_ERROR	인증되지 않음.	보안 토큰/인증서/보안 토큰 통합/만료를 확인합니다. 토큰에 토큰과 연결된 올바른 권한이 있는지 확인하십시오. Kinesis 비디오 스트림 샘플 애플리케이션의 경우 환경 변수가 제대로 설정되었는지 확인하십시오.
0x52000011	STATUS_DESCRIBE_STREAM_CALL_FAILED	DescribeStream API 실패.	이 오류는 DescribeStream API 재시도 실패 후 반환됩니다. PIC 클라이언트에서는 재시도를 포기한 후 이 오류를 반환합니다.
0x52000012	STATUS_INVALID_DESCRIBE_STREAM_RESPONSE	잘못된 DescribeStreamResponse 구조.	DescribeStreamResultEvent에 전달된 구조가 null이거나 null인 Amazon 리소스 이름(ARN)과 같이 잘못된 항목을 포함하고 있습니다.
0x52000013	STATUS_STREAM_IS_BEING_DELETED	스트림 삭제 중.	삭제 중인 스트림으로 인해 API 실패가 발생했습니다. 스트림이 사용 중인데 스트림을 삭제하려고 하는 다른 프로세스가 있는지 확인하십시오.
0x52000014	STATUS_SERVICE_CALL_INVALID_ARG_ERROR	서비스 호출에 대해 잘못된 인수가 지정됨.	서비스 호출 인수가 잘못되었거나 SDK에서 해석할 수 없는 오류가 발생한 경우 백엔드에서 이 오류를 반환합니다.
0x52000015	STATUS_SERVICE_CALL_DEVICE_NOT_FOUND_ERROR	디바이스를 찾을 수 없음.	사용 중인 디바이스가 삭제되지 않도록 합니다.
0x52000016	STATUS_SERVICE_CALL_DEVICE_NOT_PROVISIONED_ERROR	디바이스가 프로비저닝되지 않음.	디바이스가 프로비저닝되었는지 확인합니다.
0x52000017	STATUS_SERVICE_CALL_RESOURCE_NOT_FOUND_ERROR	서비스에서 반환된 일반 리소스가 없음.	이 오류는 서비스에서 리소스(예: 스트림)를 찾을 수 없는 경우 발생합니다. 이 오류는 상황마다 의미하는 바가 다를 수 있지만 스트림을 생성하기 전에 API를 사용한 것이 원인일 수 있습니다. SDK를 사용하면 스트림이 먼저 생성됩니다.
0x52000018	STATUS_INVALID_AUTH_LEN	잘못된 인증 정보 길이.	생산자 SDK 제한 사항 (p. 50)에 지정된 현재 값을 참조하십시오.
0x52000019	STATUS_CREATE_STREAM_CALL_FAILED	CreateStream API 호출 실패.	작업 실패 이유에 대한 자세한 정보는 오류 문자열을 참조하십시오.

코드	메시지	설명	권장 조치
0x5200002a	STATUS_GET_STREAMING_TOKEN_CALL_FAILED	GetStreamingToken 호출 실패.	작업 실패 이유에 대한 자세한 정보는 오류 문자열을 참조하십시오.
0x5200002b	STATUS_GET_STREAMING_ENDPOINT_CALL_FAILED	GetStreamingEndpoint API 호출 실패.	작업 실패 이유에 대한 자세한 정보는 오류 문자열을 참조하십시오.
0x5200002c	STATUS_INVALID_URI_LEN	GetStreamingEndpoint API에서 잘못된 URI 문자열 길이가 반환됨.	생산자 SDK 제한 사항 (p. 50)에 지정된 현재 최댓값을 참조하십시오.
0x5200002d	STATUS_PUT_STREAM_CALL_FAILED	PutMedia API 호출 실패.	작업 실패 이유에 대한 자세한 정보는 오류 문자열을 참조하십시오.
0x5200002e	STATUS_STORE_OUT_OF_MEMORY	콘텐츠 저장소에 메모리가 부족함.	콘텐츠 스토어는 스트림 간에 공유되기 때문에 모든 스트림에 대한 최대 기간을 저장할 수 있는 충분한 공간에 (조각 모음을 처리하기 위한) ~20%의 충분한 용량이 필요합니다. 스토리지를 오버플로우를 방지해야 합니다. 누적된 스토리지 크기 및 지연 시간 허용 오차에 해당하는 스트림당 최대 기간 값을 선택합니다. 프레임을 계속 넣기만 하는 것보다는 프레임이 콘텐츠 뷰 창에서 벗어날 때 프레임을 삭제하는 것이 좋습니다(콘텐츠 저장소 메모리 부족 문제). 이는 프레임을 삭제하면 스트림 부족 알림 콜백이 실행되기 때문입니다. 그러면 애플리케이션이 업스트림 미디어 구성 요소(예: 인코더)를 조정하여 비트 전송률 또는 프레임 수를 줄이거나 적절히 작동합니다.
0x5200002f	STATUS_NO_MORE_DATA_AVAILABLE	현재 스트림에 사용 가능한 데이터가 없음.	미디어 파이프라인이 서비스로 전송하기 위해 네트워킹 스레드에서 프레임을 소비하는 것보다 훨씬 더 느리게 생성하는 경우 이는 잠재적으로 유효한 결과입니다. 높은 수준의 클라이언트(예: C++, Java 또는 Android)에서는 내부적으로 처리되기 때문에 이 경고가 표시되지 않습니다.
0x52000030	STATUS_INVALID_TAG_VERSION	잘못된 Tag 구조 버전.	구조의 올바른 현재 버전을 지정합니다.

코드	메시지	설명	권장 조치
0x52000031	STATUS_SERVICE_CALL_UNKNOWN_ERROR	서비스 호출에서 알 수 없는 오류 또는 일반 오류가 반환됨.	자세한 내용은 로그를 참조하십시오.
0x52000032	STATUS_SERVICE_CALL_RESOURCE_NOT_FOUND_ERROR	리소스를 찾을 수 없음.	서비스에서 반환됩니다. 자세한 내용은 Kinesis 비디오 스트림 API 참조를 참조하십시오.
0x52000033	STATUS_SERVICE_CALL_CLIENT_LIMIT_ERROR	클라이언트 제한	서비스에서 반환됩니다. 자세한 내용은 Kinesis 비디오 스트림 API 참조를 참조하십시오.
0x52000034	STATUS_SERVICE_CALL_DEVICE_LIMIT_ERROR	디바이스 제한	서비스에서 반환됩니다. 자세한 내용은 Kinesis 비디오 스트림 API 참조를 참조하십시오.
0x52000035	STATUS_SERVICE_CALL_STREAM_LIMIT_ERROR	스트림 제한	서비스에서 반환됩니다. 자세한 내용은 Kinesis 비디오 스트림 API 참조를 참조하십시오.
0x52000036	STATUS_SERVICE_CALL_RESOURCE_DELETED_ERROR	리소스가 생성되었거나 삭제 중임.	서비스에서 반환됩니다. 자세한 내용은 Kinesis 비디오 스트림 API 참조를 참조하십시오.
0x52000037	STATUS_SERVICE_CALL_TIMEOUT_ERROR	서비스 호출 시간 초과.	특정 서비스 API 호출로 인해 시간 초과가 발생했습니다. 유효한 네트워크 연결이 있는지 확인하십시오. PIC에서는 이 작업을 자동으로 다시 시도합니다.
0x52000038	STATUS_STREAM_READY_CALLBACK_HANDLED	스트림 준비 상태 알림.	이 알림은 PIC에서 클라이언트로 전송되어 비동기 스트림이 생성되었는지를 나타냅니다.
0x52000039	STATUS_DEVICE_TAGS_COUNT_NON_ZERO_TAGS_NULL	잘못된 디바이스 태그가 지정됨.	태그 개수가 0개가 아닌데 태그가 비어 있습니다. 태그가 지정되어 있거나 태그 개수가 0인지 확인하십시오.
0x5200003a	STATUS_INVALID_STREAM_DESCRIPTION_VERSION	잘못된 StreamDescription 구조 버전.	구조의 올바른 현재 버전을 지정합니다.
0x5200003b	STATUS_INVALID_TAG_NAME_LENGTH	잘못된 태그 이름 길이.	생산자 SDK 제한 사항 (p. 50)에 지정된 태그 이름에 대한 제한 사항을 참조하십시오.

코드	메시지	설명	권장 조치
0x5200003c	STATUS_INVALID_TAG_VALUE_LEN	잘못된 태그 값 길이.	생산자 SDK 제한 사항 (p. 50)에 지정된 태그 값에 대한 제한 사항을 참조하십시오.
0x5200003d	STATUS_TAG_STREAM_CALL_FAILED	TagResource API 실패.	TagResource API 호출에 실패했습니다. 유효한 네트워크 연결이 있는지 확인하십시오. 실패에 대한 자세한 내용은 로그를 참조하십시오.
0x5200003e	STATUS_INVALID_CUSTOM_DATA	PIC API를 호출하는 잘못된 사용자 지정 데이터.	PIC API에 대한 호출에 잘못된 사용자 지정 데이터가 지정되었습니다. 이 오류는 PIC를 직접 사용하는 클라이언트에서만 발생할 수 있습니다.
0x5200003f	STATUS_INVALID_CREATE_STREAM_RESPONSE	잘못된 CreateStreamResponse 구조.	구조 또는 멤버 필드가 잘못되었습니다(즉, ARN이 null이거나 생산자 SDK 제한 사항 (p. 50)에 지정된 값보다 큼).
0x52000040	STATUS_CLIENT_AUTH_CALL_FAILED	클라이언트 인증 실패.	PIC에서 여러 번의 재시도 이후에 적절한 인증 정보(즉, AccessKeyId 또는 SecretAccessKey)를 가져오지 못했습니다. 인증 통합을 확인하십시오. 샘플 애플리케이션에서는 환경 변수를 사용하여 C++ 생산자 라이브러리로 자격 증명 정보를 전달합니다.
0x52000041	STATUS_GET_CLIENT_TOKEN_CALL_FAILED	보안 토큰 호출 실패.	이러한 상황은 PIC를 직접 사용하는 클라이언트에서 발생할 수 있습니다. 여러 번의 재시도 후 호출에 실패하고 이 오류가 발생합니다.
0x52000042	STATUS_CLIENT_PROVISIONING_CALL_FAILED	프로비저닝 오류	프로비저닝이 구현되지 않았습니다.
0x52000043	STATUS_CREATE_CLIENT_CALL_FAILED	생산자 클라이언트 생성 실패.	클라이언트 생성에 실패한 경우 여러 번의 재시도 후 PIC에서 일반 오류를 반환합니다.
0x52000044	STATUS_CLIENT_READY_CALLBACK_FAILED	생산자 클라이언트 준비 상태 실패.	PIC가 준비 상태로 되지 못한 경우 PIC 상태 시스템에서 반환합니다. 근본 원인에 대한 자세한 내용은 로그를 참조하십시오.

코드	메시지	설명	권장 조치
0x52000045	STATUS_TAG_CLIENT_CALL_FAILED	생산자 클라이언트에 대한 TagResource에 실패.	생산자 클라이언트에 대한 TagResource API 호출에 실패했습니다. 근본 원인에 대한 자세한 내용은 로그를 참조하십시오.
0x52000046	STATUS_INVALID_CREATE_DEVICE_RESPONSE	디바이스 생산자 생성 실패.	더 높은 수준의 SDK(예: C++ 또는 Java)는 아직 디바이스/생산자 생성 API를 구현하지 않습니다. PIC를 사용하는 클라이언트는 결과 알림을 사용해 직접 실패를 나타낼 수 있습니다.
0x52000047	STATUS_ACK_TIMESTAMP_NOT_IN_VIEW_WINDOW	수신된 ACK의 타임스탬프가 보기에 없음.	수신된 ACK에 해당하는 프레임이 콘텐츠 보기 창을 벗어나는 경우 이 오류가 발생합니다. 일반적으로 이 오류는 ACK 전송이 느린 경우 발생합니다. 이 오류는 다운링크가 느리다는 경고 및 표시로 해석할 수 있습니다.
0x52000048	STATUS_INVALID_FRAGMENT_ACK_VERSION	잘못된 FragmentAck 구조 버전.	FragmentAck 구조의 올바른 현재 버전을 지정합니다.
0x52000049	STATUS_INVALID_TOKEN_EXPIRATION	잘못된 보안 토큰 만료.	보안 토큰 만료는 향후 유예 기간을 포함해 현재 타임스탬프보다 긴 절대 타임스탬프를 가져야 합니다. 유예 기간에 대한 제한 사항은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.
0x5200004a	STATUS_END_OF_STREAM	EOS(스트림 끝) 표시기.	GetStreamData API 호출에서 현재 업로드 처리 세션이 종료되었음을 나타냅니다. 세션이 종료되었으나 오류가 발생한 경우 또는 세션 토큰이 만료되어 세션이 교체 중인 경우 이 오류가 발생합니다.
0x5200004b	STATUS_DUPLICATE_STREAM_NAME	중복 스트림 이름.	여러 스트림이 동일한 스트림 이름을 가질 수 없습니다. 스트림에 대해 고유한 이름을 선택하십시오.
0x5200004c	STATUS_INVALID_RETENTION_PERIOD	잘못된 보존 기간.	StreamInfo 구조에 잘못된 보존 기간이 지정되어 있습니다. 보존 기간에 유효한 값의 범위에 대한 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.

코드	메시지	설명	권장 조치
0x5200004d	STATUS_INVALID_ACK_KEY_START	잘못된 FragmentAck.	조각 ACK 문자열을 구문 분석하지 못했습니다. 잘못된 키 시작 표시기입니다. 조각 ACK 문자열이 손상되었을 수 있습니다. 자동 수정이 가능하므로 이 오류는 경고로 처리될 수 있습니다.
0x5200004e	STATUS_INVALID_ACK_DUPLICATE_KEY_NAME	잘못된 FragmentAck.	조각 ACK 문자열을 구문 분석하지 못했습니다. 여러 키의 이름이 동일합니다. 조각 ACK 문자열이 손상되었을 수 있습니다. 자동 수정이 가능하므로 이 오류는 경고로 처리될 수 있습니다.
0x5200004f	STATUS_INVALID_ACK_INVALID_VALUE_START	잘못된 FragmentAck.	키 값 시작 표시기가 잘못되어 조각 ACK 문자열을 구문 분석하지 못했습니다. 조각 ACK 문자열이 손상되었을 수 있습니다. 자동 수정이 가능하므로 이 오류는 경고로 처리될 수 있습니다.
0x52000050	STATUS_INVALID_ACK_INVALID_VALUE_END	잘못된 FragmentAck.	키 값 종료 표시기가 잘못되어 조각 ACK 문자열을 구문 분석하지 못했습니다. 조각 ACK 문자열이 손상되었을 수 있습니다. 자동 수정이 가능하므로 이 오류는 경고로 처리될 수 있습니다.
0x52000051	STATUS_INVALID_PARSED_ACK_TYPE	잘못된 FragmentAck.	ACK 유형이 잘못 지정되어 조각 ACK 문자열을 구문 분석하지 못했습니다.
0x52000052	STATUS_STREAM_HAS_BEEN_STOPPED	스트림이 중지됨.	스트림이 중지되었으나 프레임이 스트림에 계속 들어가고 있습니다.
0x52000053	STATUS_INVALID_STREAM_METRICS_VERSION	잘못된 StreamMetrics 구조 버전.	StreamMetrics 구조의 올바른 현재 버전을 지정합니다.
0x52000054	STATUS_INVALID_CLIENT_METRICS_VERSION	잘못된 ClientMetrics 구조 버전.	ClientMetrics 구조의 올바른 현재 버전을 지정합니다.
0x52000055	STATUS_INVALID_CLIENT_READY_STATE	생산자 초기화가 준비 상태에 도달하지 못함.	생산자 클라이언트 초기화 중 준비 상태에 도달하지 못했습니다. 자세한 내용은 로그를 참조하십시오.
0x52000056	STATUS_STATE_MACHINE_STATE_NOT_FOUND	내부 상태 시스템 오류.	공개적으로 표시되는 오류가 아닙니다.

코드	메시지	설명	권장 조치
0x52000057	STATUS_INVALID_FRAGMENT_ACK_TYPE	잘못된 ACK 유형이 FragmentAck 구조에 지정됨.	FragmentAck 구조에는 퍼블릭 헤더에 정의된 ACK 유형이 포함되어 있어야 합니다.
0x52000058	STATUS_INVALID_STREAM_READY_STATE	내부 상태 시스템 전환 오류.	공개적으로 표시되는 오류가 아닙니다.
0x52000059	STATUS_CLIENT_FREED_BEFORE_STREAM	생산자를 비운 후 스트림 객체를 비움.	생산자 객체를 비운 후 스트림 객체를 비우려는 시도가 있었습니다. 이 오류는 PIC를 직접 사용하는 클라이언트에서만 발생할 수 있습니다.
0x5200005a	STATUS_ALLOCATION_SIZE_SMALLER_THAN_REQUESTED	내부 스토리지 오류	콘텐츠 스토어의 실제 할당 크기가 패키징된 프레임/조각보다 작음을 나타내는 내부 오류입니다.
0x5200005b	STATUS_VIEW_ITEM_SIZE_GREATER_THAN_ALLOCATION	내부 스토리지 오류	콘텐츠 보기에서 할당에 저장된 크기가 콘텐츠 스토어의 할당 크기보다 큽니다.
0x5200005c	STATUS_ACK_ERR_STREAM_READ_ERROR	스트림 읽기 오류 ACK.	스트림 읽기/구문 분석 오류를 나타내는 ACK가 백엔드에서 반환되었음을 나타내는 오류입니다. 이 오류는 일반적으로 백엔드에서 스트림을 검색하지 못한 경우 발생합니다. 일반적으로 자동 다시 스트리밍이 수행되면 이 오류가 해결될 수 있습니다.
0x5200005d	STATUS_ACK_ERR_FRAGMENT_SIZE_REACHED	최대 조각 크기에 도달함.	바이트 단위의 최대 조각 크기는 생산자 SDK 제한 사항 (p. 50)에 정의되어 있습니다. 이 오류는 매우 큰 프레임이 있거나 관리 가능한 크기 조각을 생성하는 키 프레임이 없음을 나타냅니다. 인코더 설정을 확인하고 키 프레임이 적절하게 생성되었는지 확인하십시오. 밀도가 매우 높은 스트림의 경우 최대 크기를 관리하려면 보다 짧은 기간에 조각을 생성하도록 인코더를 구성합니다.

코드	메시지	설명	권장 조치
0x5200005e	STATUS_ACK_ERR_FRAGMENT_DURATION_MISMATCH	최대 조각 기간에 도달함.	최대 조각 기간은 생산자 SDK 제한 사항 (p. 50)에 정의되어 있습니다. 이 오류는 초당 프레임 속도가 매우 낮거나 관리 가능한 기간 조각을 생성하는 키 프레임이 없음을 나타냅니다. 인코더 설정을 확인하고 키 프레임이 적절하게 생성되었는지 정기적으로 확인하십시오.
0x5200005f	STATUS_ACK_ERR_CONNECTION_DURATION_EXCEEDED	최대 연결 기간에 도달함.	Kinesis 비디오 스트림에서는 생산자 SDK 제한 사항 (p. 50)에 지정된 최대 연결 기간을 강제로 적용합니다. 생산자 SDK는 최댓값에 도달하기 전에 스트림/토큰을 자동으로 교체하여 SDK를 사용하는 클라이언트에 이러한 오류가 발생하지 않도록 합니다.
0x52000060	STATUS_ACK_ERR_FRAGMENT_TIMECODE_NOT_MONOTONIC	타임코드가 단조 증가하지 않음.	생산자 SDK는 타임스탬프를 강제로 적용하여 SDK를 사용하는 클라이언트에 이러한 오류가 발생하지 않도록 합니다.
0x52000061	STATUS_ACK_ERR_MULTI_TRACK_MKV	MKV에 추적이 여러 개 있음.	생산자 SDK는 단일 추적 스트림을 강제로 적용하여 SDK를 사용하는 클라이언트에 이러한 오류가 발생하지 않도록 합니다.
0x52000062	STATUS_ACK_ERR_INVALID_MKV_DATA	MKV 데이터가 잘못됨.	백엔드 MKV 구문 분석기에서 스트림을 구문 분석하는 중 오류가 발생했습니다. 전환 중 스트림이 손상되었거나 버퍼 부족으로 SDK에서 부분적으로 전송된 테일 프레임을 강제로 삭제해야 하는 경우 SDK를 사용하는 클라이언트에 이러한 오류가 발생할 수 있습니다. 후자의 경우 FPS/해상도를 줄이거나, 압축비를 늘리거나, ("사용량이 급증하는" 네트워크의 경우) 일시적인 부족 현상을 해소하기 위해 더 큰 콘텐츠 스토어 및 버퍼 기간을 허용하는 것이 좋습니다.

코드	메시지	설명	권장 조치
0x52000063	STATUS_ACK_ERR_INVALID_PRODUCER_TIMESTAMP	생산자 타임스탬프가 잘못됨.	생산자 클록에 미래에 대해 큰 드리프트가 발생하면 서비스에서 이러한 오류 ACK를 반환합니다. 더 높은 수준의 SDK(예: Java 또는 C++)에서는 몇 가지 시스템 클록 버전을 사용하여 PIC의 현재 시간 콜백을 충족합니다. 시스템 클록이 제대로 설정되어 있는지 확인하십시오. PIC를 직접 사용하는 클라이언트는 콜백 함수가 올바른 타임스탬프를 반환하는지 확인해야 합니다.
0x52000064	STATUS_ACK_ERR_STREAM_NOT_ACTIVE	비활성 스트림.	스트림이 "활성" 상태가 아닌데 백엔드 API가 호출되었습니다. 클라이언트가 스트림을 생성한 후 즉시 해당 스트림에 프레임을 집어 넣으려고 하면 이 오류가 발생합니다. SDK에서는 상태 시스템 및 복구 시스템을 통해 이러한 시나리오를 처리합니다.
0x52000065	STATUS_ACK_ERR_KMS_KEY_ACCESS_DENIED	AWS KMS 액세스 거부 오류.	계정에 지정된 키에 대한 액세스 권한이 없는 경우 반환됩니다.
0x52000066	STATUS_ACK_ERR_KMS_KEY_DISABLED	AWS KMS 키가 비활성화됨.	지정된 키가 비활성화되었습니다.
0x52000067	STATUS_ACK_ERR_KMS_KEY_VALIDATION_ERROR	AWS KMS 키 확인 오류.	일반적인 확인 오류입니다. 자세한 내용은 AWS Key Management Service API Reference를 참조하십시오.
0x52000068	STATUS_ACK_ERR_KMS_KEY_UNAVAILABLE	AWS KMS 키를 사용할 수 없음.	이 키를 사용할 수 없습니다. 자세한 내용은 AWS Key Management Service API Reference를 참조하십시오.
0x52000069	STATUS_ACK_ERR_KMS_KEY_INVALID_USAGE	AWS KMS 키를 잘못 사용함.	AWS KMS 키가 이러한 상황에서 사용되도록 구성되지 않았습니다. 자세한 내용은 AWS Key Management Service API Reference를 참조하십시오.
0x5200006a	STATUS_ACK_ERR_KMS_KEY_INVALID_STATE	잘못된 AWS KMS 상태.	자세한 내용은 AWS Key Management Service API Reference를 참조하십시오.
0x5200006b	STATUS_ACK_ERR_KMS_KEY_NOT_FOUND	AWS KMS 키를 찾을 수 없음.	이 키를 찾을 수 없습니다. 자세한 내용은 AWS Key Management Service API Reference를 참조하십시오.

코드	메시지	설명	권장 조치
0x5200006c	STATUS_ACK_ERR_STREAM_DELETED	스트림이 삭제되었거나 삭제 중임.	다른 애플리케이션 또는 AWS Management 콘솔을 통해 스트림이 삭제 중입니다.
0x5200006d	STATUS_ACK_ERR_ACK_INTERNAL_ERROR	내부 오류.	일반적인 서비스 내부 오류
0x5200006e	STATUS_ACK_ERR_FRAGMENT_ARCHIVAL_ERROR	조각 보관 오류.	서비스에서 조각을 지속적으로 유지하고 인덱싱하지 못한 경우 반환됩니다. 드문 경우이긴 하지만 이 오류는 다양한 이유로 발생할 수 있습니다. 기본적으로 SDK에서는 조각 전송을 다시 시도합니다.
0x5200006f	STATUS_ACK_ERR_UNKNOWN_ACK_ERROR	알 수 없는 오류.	서비스에서 알 수 없는 오류를 반환했습니다.
0x52000070	STATUS_MISSING_ERR_ACK_ID	ACK 정보 누락.	ACK 구문 분석기가 구문 분석을 완료했지만 FragmentAck 정보가 누락되었습니다.
0x52000071	STATUS_INVALID_ACK_SEGMENT_LEN	잘못된 ACK 세그먼트 길이.	ACK 구문 분석기에 대해 길이가 잘못된 ACK 세그먼트 문자열이 지정되었습니다. 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.

기간 라이브러리에 의해 반환되는 오류 및 상태 코드

다음 표에는 Duration 라이브러리에 있는 메서드에 의해 반환되는 오류 및 상태 정보가 나와 있습니다.

코드	Message
0xFFFFFFFFFFFFFFFF	INVALID_DURATION_VALUE

공통 라이브러리에 의해 반환되는 오류 및 상태 코드

다음 표에는 Common 라이브러리에 있는 메서드에 의해 반환되는 오류 및 상태 정보가 나와 있습니다.

> **Note**
>
> 이러한 오류 및 상태 정보 코드는 다수의 API에 대해 공통적으로 적용됩니다.

코드	메시지	설명
0x00000001	STATUS_NULL_ARG	필수 인수에 NULL이 전달되었습니다.
0x00000002	STATUS_INVALID_ARG	인수에 대해 잘못된 값이 지정되었습니다.

코드	메시지	설명
0x00000003	STATUS_INVALID_ARG_LEN	잘못된 인수 길이가 지정되었습니다.
0x00000004	STATUS_NOT_ENOUGH_MEMORY	충분한 메모리를 할당할 수 없습니다.
0x00000005	STATUS_BUFFER_TOO_SMALL	지정된 버퍼 크기가 너무 작습니다.
0x00000006	STATUS_UNEXPECTED_EOF	예기치 않게 파일 끝에 도달했습니다.
0x00000007	STATUS_FORMAT_ERROR	잘못된 형식이 발생했습니다.
0x00000008	STATUS_INVALID_HANDLE_ERROR	잘못된 핸들 값입니다.
0x00000009	STATUS_OPEN_FILE_FAILED	파일을 열지 못했습니다.
0x0000000a	STATUS_READ_FILE_FAILED	파일에서 읽지 못했습니다.
0x0000000b	STATUS_WRITE_TO_FILE_FAILED	파일에 쓰지 못했습니다.
0x0000000c	STATUS_INTERNAL_ERROR	일반적으로 발생하지 않지만 SDK 또는 서비스 API 버그를 나타낼 수 있는 내부 오류입니다.
0x0000000d	STATUS_INVALID_OPERATION	잘못된 작업이 있거나 작업이 허용되지 않습니다.
0x0000000e	STATUS_NOT_IMPLEMENTED	기능이 구현되지 않았습니다.
0x0000000f	STATUS_OPERATION_TIMED_OUT	작업 시간이 초과되었습니다.
0x00000010	STATUS_NOT_FOUND	필요한 리소스가 없습니다.

힙 라이브러리에 의해 반환되는 오류 및 상태 코드

다음 표에는 Heap 라이브러리에 있는 메서드에 의해 반환되는 오류 및 상태 정보가 나와 있습니다.

코드	메시지	설명
0x01000001	STATUS_HEAP_FLAGS_ERROR	플래그의 잘못된 조합이 지정되었습니다.
0x01000002	STATUS_HEAP_NOT_INITIALIZED	힙이 초기화되기 전에 작업이 시도되었습니다.
0x01000003	STATUS_HEAP_CORRUPTED	힙이 손상되었거나 (디버그 모드의) 보호 대역을 덮어 썼습니다. 클라이언트 코드의 버퍼 오버플로우가 힙 손상으로 이어질 수 있습니다.
0x01000004	STATUS_HEAP_VRAM_LIB_MISSING	VRAM(비디오 RAM) 사용자 또는 커널 모드 라이브러리를 로드할 수 없거나 이 라이브러리가 없습

코드	메시지	설명
		니다. 기본 플랫폼에서 VRAM 할당을 지원하는지 확인하십시오.
0x01000005	STATUS_HEAP_VRAM_LIB_REOPEN	VRAM 라이브러리를 열지 못했습니다.
0x01000006	STATUS_HEAP_VRAM_INIT_FUNC_SYMBOL	INIT 함수 내보내기를 로드하지 못했습니다.
0x01000007	STATUS_HEAP_VRAM_ALLOC_FUNC_SYMBOL	ALLOC 함수 내보내기를 로드하지 못했습니다.
0x01000008	STATUS_HEAP_VRAM_FREE_FUNC_SYMBOL	FREE 함수 내보내기를 로드하지 못했습니다.
0x01000009	STATUS_HEAP_VRAM_LOCK_FUNC_SYMBOL	LOCK 함수 내보내기를 로드하지 못했습니다.
0x0100000a	STATUS_HEAP_VRAM_UNLOCK_FUNC_SYMBOL	UNLOCK 함수 내보내기를 로드하지 못했습니다.
0x0100000b	STATUS_HEAP_VRAM_UNINIT_FUNC_SYMBOL	UNINIT 함수 내보내기를 로드하지 못했습니다.
0x0100000c	STATUS_HEAP_VRAM_GETMAX_FUNC_SYMBOL	GETMAX 함수 내보내기를 로드하지 못했습니다.
0x0100000d	STATUS_HEAP_DIRECT_MEM_INIT	하이브리드 힙에서 기본 힙 풀을 초기화하지 못했습니다.
0x0100000e	STATUS_HEAP_VRAM_INIT_FAILED	VRAM 동적 초기화에 실패했습니다.
0x0100000f	STATUS_HEAP_LIBRARY_FREE_FAILED	VRAM 라이브러리의 할당을 취소해 해당 라이브러리를 비우지 못했습니다.
0x01000010	STATUS_HEAP_VRAM_ALLOC_FAILED	VRAM 할당에 실패했습니다.
0x01000011	STATUS_HEAP_VRAM_FREE_FAILED	VRAM 비우기에 실패했습니다.
0x01000012	STATUS_HEAP_VRAM_MAP_FAILED	VRAM 매핑에 실패했습니다.
0x01000013	STATUS_HEAP_VRAM_UNMAP_FAILED	VRAM 매핑 해제에 실패했습니다.
0x01000014	STATUS_HEAP_VRAM_UNINIT_FAILED	VRAM 초기화 취소에 실패했습니다.

MKVGen 라이브러리에 의해 반환되는 오류 및 상태 코드

다음 표에는 MKVGen 라이브러리에 있는 메서드에 의해 반환되는 오류 및 상태 정보가 나와 있습니다.

코드	메시지	설명/권장 조치
0x32000001	STATUS_MKV_INVALID_FRAME_DATA	Frame 데이터 구조의 멤버가 잘못되었습니다. 기간, 크기 및 프레임 데이터가 유효하고 생산자

코드	메시지	설명/권장 조치
		SDK 제한 사항 (p. 50)에 지정된 제한 내에 있는지 확인하십시오.
0x32000002	STATUS_MKV_INVALID_FRAME_TIMESTAMP	프레임 타임스탬프가 잘못되었습니다. 계산된 PTS(프레젠테이션 타임스탬프) 및 DTS(디코딩 타임스탬프)가 조각의 시작 프레임에 대한 타임스탬프보다 크거나 같습니다. 이는 미디어 파이프라인 다시 시작 가능성 또는 인코더 안정성 문제를 나타냅니다. 문제 해결 정보는 오류: "Kinesis 비디오 클라이언트에 프레임을 제출하지 못함" (p. 124)를 참조하십시오.
0x32000003	STATUS_MKV_INVALID_CLUSTER_DURATION	잘못된 조각 기간이 지정되었습니다. 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.
0x32000004	STATUS_MKV_INVALID_CONTENT_TYPE_LENGTH	잘못된 콘텐츠 유형 문자열 길이. 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.
0x32000005	STATUS_MKV_NUMBER_TOO_BIG	EBML(Extensible Binary Meta Language) 형식으로 표시하기에는 너무 큰 숫자를 인코딩하려고 했습니다. 이 숫자가 SDK 클라이언트에 노출되면 안 됩니다.
0x32000006	STATUS_MKV_INVALID_CODEC_ID_LENGTH	코덱 ID 문자열 길이가 잘못되었습니다. 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.
0x32000007	STATUS_MKV_INVALID_TRACK_NAME_LENGTH	추적 이름 문자열 길이가 잘못되었습니다. 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.
0x32000008	STATUS_MKV_INVALID_CODEC_PRIVATE_LENGTH	코덱 프라이빗 데이터 길이가 잘못되었습니다. 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.
0x32000009	STATUS_MKV_CODEC_PRIVATE_NULL	CPD(코덱 프라이빗 데이터)가 NULL인데 CPD 크기가 0보다 큽니다.
0x3200000a	STATUS_MKV_INVALID_TIMECODE_SCALE	타임코드 스케일 값이 잘못되었습니다. 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.

코드	메시지	설명/권장 조치
0x3200000b	STATUS_MKV_MAX_FRAME_TIMECODE	프레임 타임코드가 최댓값보다 큽니다. 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.
0x3200000c	STATUS_MKV_LARGE_FRAME_TIMECODE	최대 프레임 타임코드에 도달했습니다. MKV 형식에서는 서명된 16비트를 사용하여 클러스터 시작에 대한 프레임의 상대 타임코드를 나타냅니다. 이 오류는 프레임 타임코드를 나타낼 수 없을 때 생성됩니다. 이 오류는 타임코드 스케일 선택이 잘못되었거나 클러스터 기간이 너무 길어 프레임 타임코드가 서명된 16비트 공간에서 넘침을 나타냅니다.
0x3200000d	STATUS_MKV_INVALID_ANNEXB_NALU_IN_FRAME_DATA	잘못된 Annex-B 시작 코드가 발생했습니다. 예를 들어 Annex-B 적응 플래그가 지정되었으나 코드에서 0이 3개보다 많은 잘못된 시작 시퀀스가 발생했습니다. 잘못된 Annex-B 형식에는 바이트 스트림에서 0이 3개 이상인 시퀀스를 이스케이프하기 위한 "에뮬레이션 방지" 시퀀스가 있어야 합니다. 자세한 내용은 MPEG 사양을 참조하십시오.
0x3200000e	STATUS_MKV_INVALID_AVCC_NALU_IN_FRAME_DATA	AVCC 적응 플래그 지정 시 AVCC NALu 패키징이 잘못되었습니다. 바이트 스트림이 잘못된 AVCC 형식인지 확인하십시오. 자세한 내용은 MPEG 사양을 참조하십시오.
0x3200000f	STATUS_MKV_BOTH_ANNEXB_AND_AVCC_SPECIFIED	AVCC 및 Annex-B NAL 적응이 둘 다 지정되었습니다. 둘 중 하나만 지정하거나 아무 것도 지정하지 마십시오.
0x32000010	STATUS_MKV_INVALID_ANNEXB_NALU_IN_CPD	Annex-B 적응 플래그 지정 시 CPD의 Annex-B 형식이 잘못되었습니다. CPD가 잘못된 Annex-B 형식인지 확인하십시오. 형식이 잘못되지 않은 경우 CPD Annex-B 적응 플래그를 제거하십시오.
0x32000011	STATUS_MKV_PTS_DTS_ARE_NOT_EQUAL	Kinesis 비디오 스트림에서는 조각 시작 프레임에 대해 PTS(프레젠테이션 타임스탬프) 및 DTS(디코딩 타임스탬프)가 동일하도록 강제로 적용합니다. 둘 다 조각을 시작하는 키 프레임입니다.

코드	메시지	설명/권장 조치
0x32000012	STATUS_MKV_INVALID_H264_H265_CPD	H264/H265 코덱 프라이빗 데이터를 구문 분석하지 못했습니다.
0x32000013	STATUS_MKV_INVALID_H264_H265_SPS_WIDTH	코덱 프라이빗 데이터에서 너비를 추출하지 못했습니다.
0x32000014	STATUS_MKV_INVALID_H264_H265_SPS_HEIGHT	코덱 프라이빗 데이터에서 높이를 추출하지 못했습니다.
0x32000015	STATUS_MKV_INVALID_H264_H265_SPS_NALU	잘못된 H264/H265 SPS NALu입니다.
0x32000016	STATUS_MKV_INVALID_BIH_CPD	코덱 프라이빗 데이터의 비트맵 정도 헤더 형식이 잘못되었습니다.

Trace 라이브러리에 의해 반환되는 오류 및 상태 코드

다음 표에는 `Trace` 라이브러리에 있는 메서드에 의해 반환되는 오류 및 상태 정보가 나와 있습니다.

코드	Message
0x10100001	STATUS_MIN_PROFILER_BUFFER

Utils 라이브러리에 의해 반환되는 오류 및 상태 코드

다음 표에는 `Utils` 라이브러리에 있는 메서드에 의해 반환되는 오류 및 상태 정보가 나와 있습니다.

코드	Message
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
0x4000000a	STATUS_REMOVE_FILE_FAILED
0x4000000b	STATUS_REMOVE_LINK_FAILED
0x4000000c	STATUS_DIRECTORY_ACCESS_DENIED
0x4000000d	STATUS_DIRECTORY_MISSING_PATH

코드	Message
0x4000000e	STATUS_DIRECTORY_ENTRY_STAT_ERROR

뷰 라이브러리에 의해 반환되는 오류 및 상태 코드

다음 표에는 View 라이브러리에 있는 메서드에 의해 반환되는 오류 및 상태 정보가 나와 있습니다.

코드	메시지	설명
0x30000001	STATUS_MIN_CONTENT_VIEW_ITEMS	잘못된 콘텐츠 보기 항목 수가 지정되었습니다. 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.
0x30000002	STATUS_INVALID_CONTENT_VIEW_DURATION	잘못된 콘텐츠 보기 지속 시간이 지정되었습니다. 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.
0x30000003	STATUS_CONTENT_VIEW_NO_MORE_ITEMS	헤더 위치를 지나치려는 시도가 있었습니다.
0x30000004	STATUS_CONTENT_VIEW_INVALID_INDEX	잘못된 인덱스가 지정되어 있습니다.
0x30000005	STATUS_CONTENT_VIEW_INVALID_START_TIME	타임스탬프 또는 타임스탬프 중첩이 잘못되었습니다. 프레임 디코딩 타임스탬프는 이전 프레임 타임스탬프에 이전 프레임 기간을 더한 값(`DTS(n) >= DTS(n-1) + Duration(n-1)`)보다 크거나 같아야 합니다. 일반적으로 이 오류는 "불안정한" 인코더를 나타냅니다. 인코더는 인코딩된 프레임의 버스트를 생성하는데 이러한 프레임의 타임스탬프는 인트라 프레임 지속 시간보다 작아야 합니다. 또는 스트림이 SDK 타임스탬프를 사용하도록 구성되어 있고 프레임이 프레임 기간보다 빠르게 전송됩니다. StreamInfo.StreamCaps 구조에서 프레임 기간을 더 짧게 지정하면 인코더의 일부 "지터"와 관련해 도움이 됩니다. 예를 들어, 스트림이 25FPS인 경우 각 프레임의 기간은 40ms입니다. 그러나 인코더 지터를 처리하려면 해당 프레임 기간의 절반(20ms)을 사용하는 것이 좋습니다. 일부 스트림의 경우 오류를 감지하기 위해 보다 정확하게 타이밍을 제어해야 합니다.

코드	메시지	설명
0x30000006	STATUS_INVALID_CONTENT_VIEW_LENGTH	잘못된 콘텐츠 보기 항목 데이터 길이가 지정되었습니다.

NAL(Network Abstraction Layer) 적응 플래그 참조

이 단원에는 `StreamInfo.NalAdaptationFlags` 열거에 사용할 수 있는 플래그에 대한 정보가 포함되어 있습니다.

애플리케이션의 기본 스트림은 Annex-B 또는 AVCC 형식일 수 있습니다.

- Annex-B 형식은 2바이트의 0 뒤에 1 또는 3바이트의 0, 그 뒤에 숫자 1(시작 코드라고 함, 예: 00000001)이 오는 NALU(Network Abstraction Layer Unit)로 구분됩니다.
- AVCC 형식도 NALU를 줄 바꿈하지만 각각의 NALU 앞에 NALU의 크기를 나타내는 값이 옵니다(일반적으로 4바이트).

많은 인코더가 Annex-B 비트스트림 형식을 생성합니다. 일부 상위 수준 비트스트림 프로세서(예를 들어 AWS Management 콘솔의 재생 엔진 또는 MSE(Media Source Extensions) 플레이어)는 프레임에 AVCC 형식을 사용합니다.

H.264용 SPS/PPS(Sequence Parameter Set/Picture Parameter Set)인 CPD(Codec Private Data)도 Annex-B 또는 AVCC 형식일 수 있습니다. 하지만 CPD의 경우, 앞서 설명한 것과 형식이 다릅니다.

플래그는 다음과 같이 NALU를 프레임 데이터와 CPD를 위한 AVCC 또는 Annex-B에 적응시키라고 SDK에게 지시합니다.

플래그	적응	
`NAL_ADAPTATION_FLAG_NONE`	적응 없음	
`NAL_ADAPTATION_ANNEXB_NALS`	Annex-B NALU를 AVCC NALU에 적응	
`NAL_ADAPTATION_AVCC_NALS`	AVCC NALU를 Annex-B NALU에 적응	
`NAL_ADAPTATION_ANNEXB_CPD_NALS`	Codec Private Data용 Annex-B NALU를 AVCC 형식 NALU에 적응	
`NAL_ADAPTATION_ANNEXB_CPD_AND_FRAME_NALS`	Codec Private Data용 Annex-B NALU를 AVCC 형식 NALU에 적응	

NALU 유형에 대한 자세한 내용은 RFC 3984의 Section 1.3: Network Abstraction Layer Unit Types를 참조하십시오.

프로듀서 SDK 구조

이 단원에는 Kinesis 비디오 스트림 생산자 객체에 데이터를 제공하는 데 사용할 수 있는 구조에 대한 정보가 포함되어 있습니다.

항목

DeviceInfo/ DefaultDeviceInfoProvider

DeviceInfo와 DevaultDeviceInfoProvider 객체는 Kinesis 비디오 스트림 Producer 객체의 행태를 관리합니다.

멤버 필드

- 버전: 올바른 버전의 구조가 현재 버전의 코드 베이스와 함께 사용되는지 확인하는 데 사용되는 정수 값입니다. 현재 버전은 `DEVICE_INFO_CURRENT_VERSION` 매크로를 사용하여 지정됩니다.
- name: 디바이스의 인간이 읽을 수 있는 이름입니다.
- tagCount/ tags: 현재 사용되지 않습니다.
- streamCount: 디바이스가 처리할 수 있는 최대 스트림 수입니다. 이 값은 초기에 포인터를 위한 스토리지를 스트림 객체에 미리 할당하지만 실제 스트림 객체는 나중에 생성됩니다. 기본값은 16스트림이지만 `DefaultDeviceInfoProvider.cpp` 파일에서 이 수를 변경할 수 있습니다.
- storageInfo: 기본 스토리지 구성을 설명하는 객체입니다. 자세한 내용은 StorageInfo (p. 71) 단원을 참조하십시오.

StorageInfo

Kinesis 비디오 스트림를 위한 기본 스토리지의 구성을 지정합니다.

기본 구현은 스트리밍에 최적화된 빠른 힙의 낮은 조각화를 기반으로 합니다. 주어진 플랫폼에 중복 기재가 가능한 `MEMALLOC` 할당자를 사용합니다. 어떤 플랫폼은 피지컬 페이지로 할당을 지원하지 않아도 가상 메모리가 할당됩니다. 메모리가 사용될 때 가상 페이지는 피지컬 페이지로 지원됩니다. 결과적으로 스토리지가 충분히 사용되지 않을 때에도 전체 시스템에 메모리 부족 압력이 발생합니다.

다음 수식을 기반으로 기본 스토리지 크기를 계산합니다. `DefragmentationFactor`를 1.2(20퍼센트)로 설정해야 합니다.

```
Size = NumberOfStreams * AverageFrameSize * FramesPerSecond * BufferDurationInSeconds *
 DefragmentationFactor
```

다음 예제에서는 디바이스에 오디오 및 비디오 스트림이 있습니다. 오디오 스트림의 초당 샘플 수는 512이며 샘플은 평균 100바이트입니다. 비디오 스트림의 초당 프레임 수는 25이며 평균 10,000바이트입니다. 각 스트림의 버퍼 지속 시간은 3분입니다.

```
Size = (512 * 100 * (3 * 60) + 25 * 10000 * (3 * 60)) * 1.2 = (9216000 + 45000000) * 1.2 =
 65059200 = ~ 66MB.
```

디바이스에 추가로 사용 가능한 메모리가 있는 경우 심각한 조각화를 방지하기 위해 메모리를 스토리지에 추가하는 것이 좋습니다.

인코딩이 매우 복잡하거나(하이 모션으로 인해 프레임 크기가 더 큰 경우) 대역폭이 부족할 때 스토리지 크기가 모든 스트림에 대한 전체 버퍼를 수용하기에 적절한지 확인하십시오. 생산자가 메모리 압력에 도달하면 스토리지 오버플로우 압력 콜백(`StorageOverflowPressureFunc`)을 내보냅니다. 하지만 콘텐츠 저장소에 사용 가능한 메모리가 없으면 Kinesis 비디오 스트림로 푸시되는 프레임이 드롭되고 오류(`STATUS_STORE_OUT_OF_MEMORY = 0x5200002e`)가 발생합니다. 자세한 내용은 클라이언트 라이브러리에 의해 반환되는 오류 및 상태 코드 (p. 52) 단원을 참조하십시오. 애플리케이션 승인(ACK)을 사용할 수 없거나 지속적인 ACK가 표시되는 경우에도 이 결과가 발생할 수 있습니다. 이 경우 기존 프레임이 드롭되기 전에 "버퍼 기간" 용량까지 버퍼가 채워집니다.

멤버 필드

- 버전: 올바른 버전의 구조가 현재 버전의 코드 베이스와 함께 사용되는지 확인하는 데 사용되는 정수 값입니다.
- storageType: `DEVICE_STORAGE_TYPE` 열거를 통해 스토리지 지원 및 실행을 지정합니다. 현재 지원되는 값은 `DEVICE_STORAGE_TYPE_IN_MEM`입니다. 향후 구현에서는 `DEVICE_STORAGE_TYPE_HYBRID_FILE`이 지원되어, 스토리지가 파일 지원 콘텐츠 저장소로 다시 돌아갈 예정입니다.
- storageSize: 미리 할당할 스토리지 크기(바이트)입니다. 최소 할당은 10MB이고, 최대 할당은 10GB입니다. (이 값은 파일 지원 콘텐츠 스토어의 향후 구현에서 변경됩니다.)
- spillRatio: 보조 오버플로우 스토리지(파일 스토리지)와 반대로 직접 메모리 스토리지 유형(RAM)에서 할당될 스토리지 비율을 나타내는 정수 값입니다. 현재 사용 중이지 않습니다.
- rootDirectory: 파일 지원 콘텐츠 저장소가 있는 디렉터리의 경로입니다. 현재 사용 중이지 않습니다.

Kinesis 비디오 스트림 구조

다음 구조를 사용하여 Kinesis 비디오 스트림의 인스턴스에 데이터를 제공할 수 있습니다.

항목

StreamDefinition/ StreamInfo

C++ 계층의 `StreamDefinition` 객체는 PIC(Platform Independent Code)의 `StreamInfo` 객체를 래핑하고 생성자의 몇 가지 기본값을 제공합니다.

멤버 필드

필드	데이터 형식	설명	Default Value
stream_name	`string`	선택적 스트림 이름. 스트림 이름 길이에 대한 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오. 각 스트림은 고유한 이름을 지녀야 합니다.	이름을 지정하지 않으면 임의로 이름이 생성됩니다.
retention_period	`duration<uint64_t, ratio<3600>>`	스트림 보존 기간(초)입니다. **0**을 지정하면 보존 안 함을 나타냅니다.	3600(1시간)
tags	`const map<string, string>*`	사용자 정보가 포함된 키/값 페어의 맵입니다. 스트림에 태그 세트가 이미 있는 경우 새 태그는 기존 태그 세트에 추가됩니다.	태그 없음

필드	데이터 형식	설명	Default Value
kms_key_id	`string`	스트림을 암호화하는 데 사용할 AWS KMS 키 ID입니다. 자세한 내용은 Kinesis 비디오 스트림를 통한 서버 측 암호화 (p. 15) 단원을 참조하십시오.	기본 KMS 키(`aws/kinesis-video`)입니다.
streaming_type	`STREAMING_TYPE` 열거	지원되는 유일한 값은 `STREAMING_TYPE_REALTIME`입니다.	
content_type	`string`	스트림의 콘텐츠 형식. Kinesis 비디오 스트림 콘솔은 `video/h264` 형식으로 콘텐츠를 재생할 수 있습니다.	`video/h264`
max_latency	`duration<uint64_t, milli>`	천분의 1초 단위의 스트림 최대 지연 시간. 버퍼 지속 시간이 이 시간을 초과하면 스트림 지연 시간 압력 콜백(지정된 경우)이 호출됩니다. **0**을 지정하면 스트림 지연 시간 압력 콜백이 호출되지 않습니다.	`milliseconds::zero()`
fragment_duration	`duration<uint64_t>`	원하는 조각 지속 시간(초)입니다. 이 값은 `key_frame_fragmentation` 값과 함께 사용됩니다. 이 값이 `false`인 경우 이 시간이 경과된 후 Kinesis 비디오 스트림가 키 프레임에 조각을 생성합니다. 예를 들어, AAC(Advanced Audio Coding) 오디오 스트림의 각 프레임은 키 프레임입니다. `key_frame_fragmentation = false`를 지정하면 이 시간이 만료된 후 키 프레임에서 조각화가 수행되어 2초 조각이 생성됩니다.	2

필드	데이터 형식	설명	Default Value
timecode_scale	`duration<uint64_t, milli>`	MKV 클러스터 내 프레임에 대한 타임코드 세부 수준을 지정하는 MKV 타임코드 척도(밀리초) 입니다. MKV 프레임 타임코드는 언제나 클러스터의 시작과 연관됩니다. MKV는 서명된 16비트 값(0-32767)을 사용하여 클러스터(조각) 내 타임코드를 표시합니다. 따라서 지정된 타임코드 척도를 사용하여 프레임 타임 코드를 표시할 수 있는지 확인해야 합니다. 기본 타임코드 척도 값인 1ms를 지정할 경우 표시할 수 있는 최대 프레임은 32767ms ~= 32초입니다. 이 값은 Kinesis 비디오 스트림 제한 (p. 116)에 지정된 최대 조각 지속 시간인 10초를 초과합니다.	1
key_frame_fragmentation	`bool`	키 프레임에서 조각을 생성할지 여부입니다. `true`이면 SDK는 키 프레임이 있을 때마다 조각의 시작을 생성합니다. `false`이면 Kinesis 비디오 스트림는 최소 `fragment_duration` 이상 대기한 다음 키 프레임에 새로운 조각을 생성합니다.	`true`
frame_timecodes	`bool`	프레임 타임코드를 사용할지 또는 현재 시간 콜백을 사용하여 타임스탬프를 생성할지를 지정합니다. 대부분의 인코더는 프레임과 함께 타임스탬프를 생성하지 않습니다. 따라서 이 파라미터에 `false`를 지정하면 프레임이 Kinesis 비디오 스트림에 입력될 때 프레임에 타임스탬프가 생성됩니다.	`true`

필드	데이터 형식	설명	Default Value
absolute_fragment_times	`bool`	Kinesis 비디오 스트림는 MKV를 기본 패키징 메커니즘으로 사용합니다. MKV 사양에서는 엄격히 프레임 타임코드가 클러스터(조각)의 시작에 상대적입니다. 하지만 클러스터 타임코드는 스트림 시작 시간에 절대적이거나 상대적일 수 있습니다. 타임스탬프가 상대적이면 `PutMedia` 서비스 API 호출은 선택적 스트림 시작 타임스탬프를 사용하고 클러스터 타임스탬프를 조정합니다. 서비스는 항상 절대적 타임스탬프와 함께 조각을 저장합니다.	`true`
fragment_acks	`bool`	애플리케이션 수준의 조각 ACK(acknowledgements)를 수신할지 여부.	`true`는 SDK가 ACK를 수신하고 이에 따라 실행된다는 의미입니다.
restart_on_error	`bool`	특정 오류에 따라 재시작할지 여부.	`true`를 지정하면 SDK는 오류가 발생할 경우 스트리밍을 다시 시작하려고 합니다.
recalculate_metrics	`bool`	지표를 다시 계산할지 여부입니다. 각 호출에 따라 지표를 검색하고 최신 "실행"값을 다시 계산하여 CPU에 영향을 적게 줍니다. 매우 낮은 전력/풋프린트 디바이스에서는 CPU 주기를 절약하기 위해 이 값을 `false`로 설정해야 할 수 있습니다. 그렇지 않으면 이 값에 `false`를 사용하지 않는 것이 좋습니다.	`true`

필드	데이터 형식	설명	Default Value
nal_adaptation_flags	`uint32_t`	NALU(Network Abstraction Layer Unit) 적응 플래그를 지정합니다. 비트스트림이 H.264로 암호화되면 NALU에서 원시 또는 패키징된 상태로 처리될 수 있습니다. 이러한 비트스트림은 Annex-B 또는 AVCC 형식일 수 있습니다. 대부분의 초기 스트림 생산자/소비자(읽기 인코더/디코더)는 오류 복구와 같은 이점이 있기 때문에 Annex-B 형식을 사용합니다. 더 높은 수준의 시스템은 MPEG, HLS, DASH 등의 기본 형식인 AVCC 형식을 사용합니다. 콘솔 재생은 브라우저의 MSE(미디어 소스 확장)를 사용하여 AVCC 형식을 사용하는 스트림을 디코딩하고 재생합니다. H.264의 경우(그리고 M-JPEG 및 H.265의 경우) SDK는 적응 기능을 제공합니다. 대부분의 초기 스트림은 다음 형식입니다. 이 예제에서 `Ab`는 Annex-B 시작 코드(001 또는 0001)입니다. `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)` H.264의 경우 CPD(코덱 프라이빗 데이터)는 SPS(Sequence Parameter Set) 및 PPS(Picture Parameter Set) 파라미터에 있으며 AVCC 형식에 맞게 적응할 수 있습니다. 미디어 파이프라인이 CPD를 별도로 제공하지 않는 한, 애플리케이션은 프레임에서 CPD를 추출할 수 있습니다. 첫 번째	기본값은 프레임 데이터와 CPD(코덱 프라이빗 데이터) 모두에 대해 Annex-B 형식을 AVCC 형식으로 조정하는 것입니다.

필드	데이터 형식	설명	Default Value
		IDR 프레임(SPS/PPS가 포함됨)을 찾고, 두 개의 NALU(`Ab(Sps)Ab(Pps)`)를 추출한 다음, `StreamDefinition`의 CPD에서 설정하여 이렇게 할 수 있습니다. 자세한 내용은 NAL 적응 플래그 (p. 70) 단원을 참조하십시오.	
frame_rate	`uint32_t`	예상 프레임 속도입니다. 이 값은 버퍼링 니즈를 계산하는 데 사용됩니다.	25
avg_bandwidth_bps	`uint32_t`	스트림용 예상 평균 대역폭. 이 값은 버퍼링 니즈를 계산하는 데 사용됩니다.	4 * 1024 * 1024
buffer_duration	`duration<uint64_t>`	스트림 버퍼 지속 시간(초)입니다. SDK는 최대 `buffer_duration` 동안 프레임을 콘텐츠 저장소에 보관하며, 이 시간 이후에는 창이 앞으로 이동함에 따라 기존 프레임이 드롭됩니다. 드롭된 프레임이 백엔드에 전송되지 않은 경우 드롭된 프레임 콜백이 호출됩니다. 현재 버퍼 지속 시간이 `max_latency`보다 크면 스트림 지연 시간 압력 콜백이 호출됩니다. 조각 지속 ACK가 수신되면 버퍼가 다음 조각 시작에 맞게 잘립니다. 따라서 콘텐츠가 클라우드에 내구성 있게 지속되므로 더 이상 로컬 디바이스에 콘텐츠를 저장할 필요가 없습니다.	120

필드	데이터 형식	설명	Default Value
replay_duration	`duration<uint64_t>`	다시 시작이 활성화된 경우 오류가 발생한 동안 현재 리더를 뒤로 롤하여 다시 재생하는 기간(초) 입니다. 버퍼가 시작되면 롤백이 중지됩니다(스트리밍이 막 시작되었거나 지속적인 ACK가 나타나는 경우). 롤백은 조각화 시작을 나타내는 키 프레임에 랜딩하려고 합니다. 다시 시작을 유발하는 오류가 데드 호스트를 나타내지 않는 경우(즉, 호스트가 여전히 실행 중이고 내부 버퍼에 프레임 데이터가 들어 있는 경우) 마지막으로 수신된 ACK 프레임에서 롤백이 중단됩니다. 그런 다음 전체 조각이 호스트 메모리에 이미 저장되어 있으므로 다음 키 프레임으로 앞으로 롤합니다.	40
connection_staleness	`duration<uint64_t>`	SDK가 버퍼 ACK를 수신하지 않는 경우 스트림 기한 경과 콜백이 호출될 때까지의 시간(초)입니다. 이 값은 프레임이 디바이스에서 전송되고 있지만 백엔드가 프레임을 승인하고 있지 않음을 나타냅니다. 이 조건은 중간 홉 또는 로드 밸런서에서 연결이 끊어졌음을 나타냅니다.	30
codec_id	`string`	MKV 트랙의 Codec ID	`V_MPEG4/ISO/AVC`
track_name	`string`	MKV 트랙 이름	`kinesis_video`

필드	데이터 형식	설명	Default Value
codecPrivateData	`unsigned char*`	CPD(코덱 프라이빗 데이터) 버퍼입니다. 미디어 파이프라인이 스트림이 시작되기 전 CPD에 대한 정보를 가지고 있다면 미디어 파이프라인은 `StreamDefinition.codecPrivateData`에 설정됩니다. 비트가 복사되며, 스트림을 생성하기 위한 호출 후에 버퍼를 재사용하거나 비울 수 있습니다. 하지만 스트림이 생성될 때 데이터가 사용 가능하지 않은 경우 `KinesisVideoStream.start(cpd)` 함수의 오버로드 중 하나에서 설정할 수 있습니다.	`null`
codecPrivateDataSize	`uint32_t`	코덱 프라이빗 데이터 버퍼 크기입니다.	0

ClientMetrics

ClientMetrics 객체는 `getKinesisVideoMetrics`를 호출하여 채워지게 됩니다.

멤버 필드

필드	데이터 형식	설명
version	`UINT32`	구조의 버전은 `CLIENT_METRICS_CURRENT_VERSION` 매크로에 정의되어 있습니다.
contentStoreSize	`UINT64`	바이트 단위의 전체 콘텐츠 스토어 크기. `DeviceInfo.StorageInfo.storageSize`에서 이 값을 지정합니다.
contentStoreAvailableSize	`UINT64`	바이트 단위의 현재 사용 가능한 스토리지 크기.
contentStoreAllocatedSize	`UINT64`	현재 할당된 크기. 내부 기록 및 콘텐츠 저장소 구현으로 인해 할당된 크기 더하기 사용 가능한 크기는 전체 스토리지 크기보다 약간 작아야 합니다.
totalContentViewsSize	`UINT64`	모든 스트림에 대한 모든 콘텐츠에 할당된 메모리 크기. 이 크기는 스토리지 크기에 대해 계산되지 않습니다. 이 메모리는 `MEMALLOC` 매크로를 사용하여 할당되고 덮어

필드	데이터 형식	설명
		쓰여 사용자 지정 할당기를 제공합니다.
totalFrameRate	`UINT64`	모든 스트림의 관측된 총 프레임 속도.
totalTransferRate	`UINT64`	초당 바이트 단위의 모든 스트림의 관측된 총 스트림 속도.

StreamMetrics

StreamMetrics 객체는 `getKinesisVideoMetrics`를 호출하여 채워지게 됩니다.

멤버 필드

필드	데이터 형식	설명
version	`UINT32`	구조의 버전은 `STREAM_METRICS_CURRENT_VERSION` 매크로에 정의되어 있습니다.
currentViewDuration	`UINT64`	측척된 프레임 지속 시간. 고속 네트워킹 사례에서 이 지속 시간은 0 또는 프레임 지속 시간(프레임이 전송 중일 때)입니다. 현재 지속 시간이 `StreamDefinition`에 지정된 `max_latency`를 초과하고 스트림 지연 시간이 지정되면 스트림 지연 시간 콜백이 호출됩니다. 지속 시간은 PIC 계층의 기본 시간 단위인 100ns 단위로 지정됩니다.
overallViewDuration	`UINT64`	총 보기 지속 시간. ACK 또는 지속성 없음으로 스트림이 구성된 경우 이 값은 프레임이 Kinesis 비디오 스트림에 입력됨에 따라 증가하고 `StreamDefinition`에서 `buffer_duration`과 같게 됩니다. ACK 타임스탬프는 전체 조각의 시작을 나타내기 때문에 ACK가 활성화되고 지속적 ACK가 수신되면 버퍼는 다음 키 프레임에 맞게 잘립니다. 지속 시간은 PIC 계층의 기본 시간 단위인 100-ns 단위로 지정됩니다.
currentViewSize	`UINT64`	바이트 단위의 현재 버퍼 크기.
overallViewSize	`UINT64`	바이트 단위의 전체 보기 크기.
currentFrameRate	`UINT64`	현재 스트림의 관측된 총 프레임 속도.

필드	데이터 형식	설명
currentTransferRate	`UINT64`	초당 바이트 단위의 현재 스트림의 관측된 전송 속도.

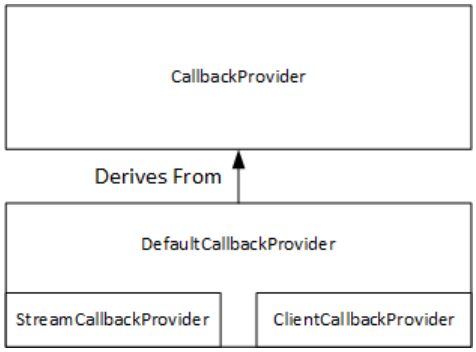
생산자 SDK 콜백

Amazon Kinesis 비디오 스트림 생산자 SDK의 클래스와 메서드는 고유의 프로세스를 유지하지 않습니다. 그 대신, 수신 함수 호출 및 이벤트를 사용하여 애플리케이션과 통신할 콜백을 예약합니다.

애플리케이션이 SDK와 상호 작용하는 데 사용할 수 있는 두 개의 콜백 패턴이 있습니다.

- `CallbackProvider`: 이 객체는 PIC(Platform Independent Code) 구성 요소에서 애플리케이션으로 이동하는 모든 콜백을 공개합니다. 이 패턴은 전체 기능을 허용하지만, 이 경우 구현은 C++ 계층에 있는 모든 퍼블릭 API 메서드 및 서명을 처리해야 합니다.
- StreamCallbackProvider (p. 82) 및 ClientCallbackProvider (p. 81): 이러한 객체는 스트림별 및 클라이언트별 콜백을 공개하며 SDK의 C++ 계층은 나머지 콜백을 공개합니다. 이것은 생산자 SDK와 상호 작용하기 위한 기본 콜백 패턴입니다.

다음 다이어그램은 콜백 객체의 객체 모델을 보여줍니다.



위의 다이어그램에서 `DefaultCallbackProvider`는 `CallbackProvider`(PIC에 있는 모든 콜백을 공개)에서 파생되며 `StreamCallbackProvider` 및 `ClientCallbackProvider`를 포함합니다.

이 주제는 다음 단원을 포함하고 있습니다.

- ClientCallbackProvider (p. 81)
- StreamCallbackProvider (p. 82)
- ClientCallbacks 구조 (p. 82)

ClientCallbackProvider

`ClientCallbackProvider` 객체는 클라이언트 수준 콜백 함수를 공개합니다. 이 함수의 세부 정보는 ClientCallbacks (p. 82) 단원에서 설명합니다.

콜백 메서드:

- `getClientReadyCallback`: 클라이언트의 준비 상태를 보고합니다.
- `getStorageOverflowPressureCallback`: 스토리지 오버플로우 또는 압력을 보고합니다. 스토리지 활용도가 전체 스토리지 크기의 5퍼센트인 `STORAGE_PRESSURE_NOTIFICATION_THRESHOLD` 값을 초과하면 이 콜백이 호출됩니다. 자세한 내용은 StorageInfo (p. 71) 단원을 참조하십시오.

`ClientCallbackProvider`의 소스 코드는 Include.h를 참조하십시오.

StreamCallbackProvider

`StreamCallbackProvider` 객체는 스트림 수준 콜백 함수를 공개합니다.

콜백 메서드:

- `getDroppedFragmentReportCallback`: 드롭된 조각을 보고합니다.
- `getDroppedFrameReportCallback`: 드롭된 프레임을 보고합니다.
- `getFragmentAckReceivedCallback`: 스트림을 위해 조각 ACK가 수신됨을 보고합니다.
- `getStreamClosedCallback`: 스트림 닫힘 조건을 보고합니다.
- `getStreamConnectionStaleCallback`: 기한 경과 연결 조건을 보고합니다. 이 조건에서 생산자는 서비스에 데이터를 전송하지만 승인을 받지 않습니다.
- `getStreamDataAvailableCallback`: 스트림에서 데이터가 사용 가능함을 보고합니다.
- `getStreamErrorReportCallback`: 스트림 오류 조건을 보고합니다.
- `getStreamLatencyPressureCallback`: 누적 버퍼 크기가 `max_latency` 값보다 큰 경우 스트림 지연 시간 조건을 보고합니다. 자세한 내용은 StreamDefinition/ StreamInfo (p. 72) 단원을 참조하십시오.
- `getStreamReadyCallback`: 스트림 준비 조건을 보고합니다.
- `getStreamUnderflowReportCallback`: 스트림 오퍼플로우 조건을 보고합니다. 이 함수는 현재 사용되지 않으며 향후 사용을 위해 예약됩니다.

`StreamCallbackProvider`의 소스 코드는 StreamCallbackProvider.h를 참조하십시오.

ClientCallbacks 구조

`ClientCallbacks` 구조에는 특정 이벤트가 발생할 때 PIC가 호출하는 콜백 함수 엔트리 포인트가 포함됩니다. 구조에는 `CALLBACKS_CURRENT_VERSION` 필드에 있는 버전 정보, 및 개별 콜백 함수를 통해 반환되는 사용자 정의 데이터에 대한 `customData` 필드도 포함됩니다.

다음 코드 예와 같이, 클라이언트 애플리케이션은 `this` 포인터를 `custom_data` 필드에 사용하여 멤버 함수를 정적 `ClientCallback` 함수에 실행 시간에 매핑합니다.

```
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(...);
```

이벤트

함수	설명	Type
`CreateDeviceFunc`	백엔드에서 현재 구현되지 않습니다. Java 또는 C++에서 호출되면 이 호출이 실패합니다. 다른 클라이언트는 플랫폼별 초기화를 수행합니다.	백엔드 API
`CreateStreamFunc`	스트림이 생성될 때 호출됩니다.	백엔드 API
`DescribeStreamFunc`	`DescribeStream`이 호출될 때 호출됩니다.	백엔드 API

함수	설명	Type
`GetStreamingEndpointFunc`	`GetStreamingEndpoint`이 호출될 때 호출됩니다.	백엔드 API
`GetStreamingTokenFunc`	`GetStreamingToken`이 호출될 때 호출됩니다.	백엔드 API
`PutStreamFunc`	`PutStream`이 호출될 때 호출됩니다.	백엔드 API
`TagResourceFunc`	`TagResource`이 호출될 때 호출됩니다.	백엔드 API
`CreateMutexFunc`	동기화 뮤텍스를 생성합니다.	동기화
`FreeMutexFunc`	뮤텍스를 비웁니다.	동기화
`LockMutexFunc`	동기화 뮤텍스를 잠급니다.	동기화
`TryLockMutexFunc`	뮤텍스를 잠그려고 시도합니다. 현재 구현되지 않습니다.	동기화
`UnlockMutexFunc`	뮤텍스 잠금을 해제합니다.	동기화
`ClientReadyFunc`	클라이언트가 준비 상태에 들어갈 때 호출됩니다.	알림
`DroppedFrameReportFunc`	프레임이 드롭될 때 보고합니다.	알림
`DroppedFragmentReportFunc`	조각이 드롭될 때 보고합니다. 이 함수는 현재 사용되지 않으며 향후 사용을 위해 예약됩니다.	알림
`FragmentAckReceivedFunc`	조각 ACK(버퍼링, 수신, 지속 및 오류)가 수신될 때 호출됩니다.	알림
`StorageOverflowPressureFunc`	스토리지 활용도가 전체 스토리지 크기의 5퍼센트로 정의되는 `STORAGE_PRESSURE_NOTIFICATION_THRESHOLD` 값을 초과하는 경우 호출됩니다.	알림
`StreamClosedFunc`	나머지 프레임의 마지막 비트가 스트리밍될 때 호출됩니다.	알림
`StreamConnectionStaleFunc`	스트림이 기한 경과 연결 상태에 들어갈 때 호출됩니다. 이 조건에서 생산자는 서비스에 데이터를 전송하지만 승인을 받지 않습니다.	알림
`StreamDataAvailableFunc`	스트림 데이터를 사용할 수 있을 때 호출됩니다.	알림

함수	설명	Type
`StreamErrorReportFunc`	스트림 오류가 발생할 경우 호출됩니다. 스트림이 이 조건에 있으면 PIC는 스트림을 자동으로 닫습니다.	알림
`StreamLatencyPressureFunc`	스트림이 지연 시간 조건에 들어갈 때 호출됩니다. 이 조건은 누적된 버퍼 크기가 `max_latency` 값보다 큰 경우입니다. 자세한 내용은 StreamDefinition/StreamInfo (p. 72) 단원을 참조하십시오.	알림
`StreamReadyFunc`	스트림이 준비 상태에 들어갈 때 호출됩니다.	알림
`StreamUnderflowReportFunc`	이 함수는 현재 사용되지 않으며 향후 사용을 위해 예약됩니다.	알림
`DeviceCertToTokenFunc`	연결 인증서를 토큰으로 반환합니다.	플랫폼 통합
`GetCurrentTimeFunc`	현재 시간을 반환합니다.	플랫폼 통합
`GetDeviceCertificateFunc`	디바이스 인증서를 반환합니다. 이 함수는 현재 사용되지 않으며 향후 사용을 위해 예약됩니다.	플랫폼 통합
`GetDeviceFingerprintFunc`	디바이스 지문을 반환합니다. 이 함수는 현재 사용되지 않으며 향후 사용을 위해 예약됩니다.	플랫폼 통합
`GetRandomNumberFunc`	0과 `RAND_MAX` 사이의 임의 숫자를 반환합니다.	플랫폼 통합
`GetSecurityTokenFunc`	백엔드 API와 통신하는 함수에 전달되는 보안 토큰을 반환합니다. 구현은 직렬화된 `AccessKeyId`, `SecretKeyId` 및 세션 토큰을 지정할 수 있습니다.	플랫폼 통합
`LogPrintFunc`	태그와 로그 수준을 사용하여 텍스트 줄을 기록합니다. 자세한 내용은 `PlatformUtils.h` 단원을 참조하십시오.	플랫폼 통합

위의 표에 있는 플랫폼 통합 함수의 경우 마지막 파라미터는 `ServiceCallContext` 구조이며, 이 구조에는 다음 필드가 있습니다.

- `version`: 구조의 버전입니다.
- `callAfter`: 함수를 호출할 때까지의 절대 시간입니다.
- `timeout`: 작업 제한 시간입니다(100나노초 단위).
- `customData`: 다시 클라이언트에게 전달할 사용자 정의 값입니다.

- `pAuthInfo`: 호출에 대한 자격 증명입니다. 자세한 내용은 다음(`__AuthInfo`) 구조를 참조하십시오.

권한 부여 정보는 `__AuthInfo` 구조를 사용하여 제공되며, 이 구조는 직렬화된 자격 증명 또는 공급자별 인증 토큰일 수 있습니다. 이 구조에는 다음 필드가 있습니다.

- `version`: `__AuthInfo` 구조의 버전입니다.
- `type`: 자격 증명의 유형(인증서 또는 보안 토큰)을 정의하는 `AUTH_INFO_TYPE` 값입니다.
- `data`: 인증 정보가 포함된 바이트 배열입니다.
- `size`: `data` 파라미터의 크기입니다.
- `expiration`: 100 나노초 단위로 표시되는 자격 증명의 만료입니다.

Kinesis Video Stream Parser Library

Kinesis Video Stream Parser Library는 Kinesis 비디오 스트림에서 MKV 데이터를 소비하기 위해 Java 애플리케이션에서 사용할 수 있는 사용하기 쉬운 도구 모음입니다.

이 라이브러리에는 다음 도구가 포함됩니다.

- StreamingMkvReader (p. 87): 이 클래스는 비디오 스트림에서 지정된 MKV 요소를 읽습니다.
- FragmentMetadataVisitor (p. 87): 이 클래스는 조각의 메타데이터를 검색하고(미디어 조각) 추적합니다(오디오 또는 자막과 같은 미디어 정보를 포함하는 개별 데이터 스트림).
- OutputSegmentMerger (p. 89): 이 클래스는 연속 조각 또는 청크를 한 비디오 스트림에 병합합니다.
- KinesisVideoExample (p. 89): Kinesis Video Stream Parser Library를 사용하는 방법을 보여 주는 샘플 애플리케이션입니다.

이 라이브러리에는 도구가 사용되는 방법을 보여 주는 테스트도 포함됩니다.

절차: Kinesis Video Stream Parser Library 사용

이 절차에는 다음 단계가 포함됩니다.

- the section called “1 단계: 코드 다운로드 및 구성” (p. 86)
- the section called “2 단계: 코드 쓰기 및 검사” (p. 87)
- the section called “3 단계: 코드 실행 및 확인” (p. 92)

사전 조건

Kinesis Video Stream Parser Library를 검사하고 사용하려면 다음이 필요합니다.

- Amazon Web Services (AWS) 계정. 아직 AWS 계정이 없는 경우에는 아래 단계를 수행합니다.
  - https://aws.amazon.com/을 열고 [Create an AWS Account]를 선택합니다.

    Note

    이전에 AWS Management 콘솔에 로그인한 적이 있을 경우 이를 브라우저에서 사용하지 못할 수 있습니다. 이 경우 [Sign In to the Console]을 선택한 다음, [Create a new AWS account]를 선택합니다.
  - 온라인 지시 사항을 따릅니다.

    등록 절차 중 전화를 받고 전화 키패드를 사용하여 PIN을 입력하는 과정이 있습니다.

  Kinesis 비디오 스트림에 대한 프로그래밍 가능 액세스를 구성할 때 필요하므로 AWS 계정 ID를 적어 두십시오.
- Eclipse Java Neon 또는 JetBrains IntelliJ Idea 같은 Java 통합 개발 환경(IDE).

1 단계: 코드 다운로드 및 구성

이 단원에서는 Java 라이브러리 및 테스트 코드를 다운로드하고 프로젝트를 Java IDE로 가져옵니다.

사전 조건 및 이 절차에 관한 기타 세부 정보는 스트림 구문 분석기 라이브러리 (p. 86)를 참조하십시오.

1. 디렉터리를 생성하고 GitHub 리포지토리에서 라이브러리 소스 코드를 복제합니다.

```
$ git clone https://github.com/aws/amazon-kinesis-video-streams-parser-library
```

2. 사용 중인 Java IDE(예: Eclipse 또는 IntelliJ IDEA)를 열고 다운로드한 Apache Maven 프로젝트를 가져옵니다.

   - Eclipse에서 [File], [Import], [Maven], [Existing Maven Projects]를 차례로 선택하여 `kinesis-video-streams-parser-lib` 폴더로 이동합니다.
   - IntelliJ Idea에서 [Import]를 선택합니다. 다운로드한 패키지의 루트에 있는 pom.xml 파일을 찾습니다.

   자세한 내용은 관련 IDE 문서를 참조하십시오.

다음 단계

the section called “2 단계: 코드 쓰기 및 검사” (p. 87)

2 단계: 코드 쓰기 및 검사

이 단원에서는 Java 라이브러리 및 테스트 코드를 검사하고 자체 코드에 있는 라이브러리에서 도구를 사용하는 방법을 배웁니다.

Kinesis Video Stream Parser Library에는 다음 도구가 포함됩니다.

- StreamingMkvReader (p. 87)
- FragmentMetadataVisitor (p. 87)
- OutputSegmentMerger (p. 89)
- KinesisVideoExample (p. 89)

StreamingMkvReader

이 클래스는 비차단 방식으로 스트림에서 지정된 MKV 요소를 읽습니다.

다음 코드 예제(`FragmentMetadataVisitorTest`)는 `Streaming MkvReader`를 생성하고 사용하여 입력 스트림(`inputStream`)에서 `MkvElement` 객체를 검색하는 방법을 보여 줍니다.

```
StreamingMkvReader mkvStreamReader =
                StreamingMkvReader.createDefault(new
 InputStreamParserByteSource(inputStream));
        while (mkvStreamReader.mightHaveNext()) {
            Optional<MkvElement> mkvElement = mkvStreamReader.nextIfAvailable();
            if (mkvElement.isPresent()) {
                mkvElement.get().accept(fragmentVisitor);
                ...
                }
            }
        }
```

FragmentMetadataVisitor

이 클래스는 조각(미디어 조각) 및 트랙(코덱 프라이빗 데이터, 픽셀 너비 또는 픽셀 높이와 같은 미디어 정보가 포함된 개별 데이터 스트림)의 메타데이터를 가져옵니다.

다음 코드 예제(FragmentMetadataVisitorTest 파일)는 FragmentMetadataVisitor를 사용하여 MkvElement 객체로부터 데이터를 검색하는 방법을 보여 줍니다.

```
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()) {
                            Optional<String> continuationToken =
 fragmentVisitor.getContinuationToken();
                            Assert.assertTrue(continuationToken.isPresent());
                            Assert.assertEquals(continuationTokens.get(segmentCount),
 continuationToken.get());
                        }
                        segmentCount++;
                    }
                }
            }

        }
```

앞선 예제는 다음과 같은 코딩 패턴을 보입니다.

- 데이터 구문 분석을 위한 FragmentMetadataVisitor와 데이터 제공을 위한 StreamingMkvReader (p. 87)를 생성합니다.
- 스트림에 있는 각 MkvElement에 대해 메타데이터가 유형 SIMPLEBLOCK인지 테스트합니다.
- 이 경우 MkvElement에서 MkvDataElement를 검색합니다.
- MkvDataElement에서 Frame(미디어 데이터)을 검색합니다.
- FragmentMetadataVisitor에서 Frame의 MkvTrackMetadata를 검색합니다.
- Frame 및 MkvTrackMetadata 객체로부터 다음 데이터를 검색하고 확인합니다.
  - 트랙 번호.
  - 프레임의 픽셀 높이.
  - 프레임의 픽셀 넓이.
  - 프레임 인코딩에 사용되는 코덱의 코덱 ID.
  - 이 프레임이 순서대로 도착했는지 여부. 즉, 이전 프레임의 트랙 번호(존재하는 경우)가 현재 프레임의 트랙 번호보다 작은지 여부를 확인합니다.

프로젝트에서 FragmentMetadataVisitor를 사용하려면, MkvElement 객체를 accept 메서드를 사용하여 방문자에 전달합니다

```
mkvElement.get().accept(fragmentVisitor);
```

OutputSegmentMerger

이 클래스는 스트림의 여러 트랙에서 취합되는 메타데이터를 단일 세그먼트로 병합합니다.

다음 코드 예제(`FragmentMetadataVisitorTest` 파일)는 `OutputSegmentMerger`를 사용하여 `inputBytes` 바이트 어레이의 트랙 메타데이터를 병합하는 방법을 보여 줍니다.

```
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);
    }
}
```

앞선 예제는 다음과 같은 코딩 패턴을 보입니다.

- 스트림에서 메타데이터를 검색하기 위해 FragmentMetadataVisitor (p. 87)를 생성합니다.
- 출력 스트림을 생성하여 병합된 메타데이터를 수신합니다.
- `OutputSegmentMerger`에 통과하는 `ByteArrayOutputStream`를 생성합니다.
- 두 방문자를 포함한 `CompositeMkvElementVisitor`를 생성합니다.
- 지정 파일을 가리키는 `InputStream`을 생성합니다.
- 입력 데이터에 있는 각 요소를 출력 스트림에 병합합니다.

KinesisVideoExample

Kinesis Video Stream Parser Library를 사용하는 방법을 보여 주는 샘플 애플리케이션입니다.

이 클래스는 다음 작업을 수행합니다.

- Kinesis 비디오 스트림를 생성합니다. 주어진 이름의 스트림이 이미 존재하는 경우, 해당 스트림은 삭제되고 다시 생성됩니다.
- PutMedia를 호출하여 비디오 조각을 Kinesis 비디오 스트림로 스트리밍합니다.
- GetMedia를 호출하여 비디오 조각을 Kinesis 비디오 스트림에서 스트리밍합니다.

- StreamingMkvReader (p. 87)을 사용하여 스트림에서 반환되는 조각을 구문 분석하고, FragmentMetadataVisitor (p. 87)를 사용하여 조각을 로깅합니다.

스트림 삭제 및 재생성

다음 코드 예제(`StreamOps.java` 파일의 예제)는 주어진 Kinesis 비디오 스트림를 삭제합니다.

```
//Delete the stream
amazonKinesisVideo.deleteStream(new
 DeleteStreamRequest().withStreamARN(streamInfo.get().getStreamARN()));
```

다음 코드 예제(`StreamOps.java` 파일의 예제)는 지정된 이름으로 Kinesis 비디오 스트림를 생성합니다.

```
amazonKinesisVideo.createStream(new CreateStreamRequest().withStreamName(streamName)
.withDataRetentionInHours(DATA_RETENTION_IN_HOURS)
.withMediaType("video/h264"));
```

PutMedia 호출

다음 코드 예제(`PutMediaWorker.java` 파일의 예제)는 스트림에서 PutMedia를 호출합니다.

```
 putMedia.putMedia(new PutMediaRequest().withStreamName(streamName)
.withFragmentTimecodeType(FragmentTimecodeType.RELATIVE)
.withProducerStartTimestamp(new Date())
.withPayload(inputStream), new PutMediaAckResponseHandler() {
...
});
```

GetMedia 호출

다음 코드 예제(`GetMediaWorker.java` 파일의 예제)는 스트림에서 GetMedia를 호출합니다.

```
GetMediaResult result = videoMedia.getMedia(new
 GetMediaRequest().withStreamName(streamName).withStartSelector(startSelector));
```

GetMedia 결과 구문 분석

이 단원에서는 StreamingMkvReader (p. 87), FragmentMetadataVisitor (p. 87)와 `CompositeMkvElementVisitor`를 사용하여 `GetMedia`에서 반환되는 데이터를 구문 분석하고, 파일에 저장하고, 로깅하는 방법을 설명합니다.

StreamingMkvReader를 사용하여 GetMedia 출력 읽기

다음 코드 예제(`GetMediaWorker.java` 파일의 예제)는 StreamingMkvReader (p. 87)를 생성해 GetMedia 작업의 결과를 구문 분석하는 데 사용합니다.

```
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);
```

```
}
```

앞의 코드 예제에서 StreamingMkvReader (p. 87)는 `GetMedia` 결과의 페이로드에서 `MKVElement` 객체를 검색합니다. 다음 단원에서는 이 요소들이 FragmentMetadataVisitor (p. 87)로 전달됩니다.

FragmentMetadataVisitor로 조각 검색

다음 코드 예제(`KinesisVideoExample.java` 및 `StreamingMkvReader.java` 파일의 예제)는 FragmentMetadataVisitor (p. 87)를 생성합니다. 그런 다음 StreamingMkvReader (p. 87)에 의해 반복되는 `MkvElement` 객체가 `accept` 메서드를 사용하여 방문자에게 전달됩니다

*`KinesisVideoExample.java`*로부터:

```
FragmentMetadataVisitor fragmentMetadataVisitor = FragmentMetadataVisitor.create();
```

*`StreamingMkvReader.java`*로부터:

```
if (mkvElementOptional.isPresent()) {
    //Apply the MkvElement to the visitor
    mkvElementOptional.get().accept(elementVisitor);
        }
```

요소를 로깅하고 파일에 쓰기

다음 코드 예제(`KinesisVideoExample.java` 파일의 예제)는 다음 객체를 생성하여 `GetMediaProcessingArguments` 함수의 반환 값의 일부로 반환합니다.

- 시스템 로그에 쓰는 `LogVisitor`(`MkvElementVisitor`의 확장).
- 수신 데이터를 MKV 파일에 쓰는 `OutputStream`.
- `OutputStream`으로 가는 데이터를 버퍼링하는 `BufferedOutputStream`.
- `GetMedia` 결과의 연속적 요소들을 동일 트랙 및 EBML 데이터와 병합하는 the section called “OutputSegmentMerger” (p. 89).
- FragmentMetadataVisitor (p. 87), the section called “OutputSegmentMerger” (p. 89) 및 `LogVisitor`를 단일 요소 방문자로 구성하는 `CompositeMkvElementVisitor`.

```
//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);
```

그런 다음 미디어 처리 인수가 전달되는 `GetMediaWorker`는 별도의 스레드에서 작업자를 실행하는 `ExecutorService`로 전달됩니다.

```
GetMediaWorker getMediaWorker = GetMediaWorker.create(getRegion(),
        getCredentialsProvider(),
        getStreamName(),
        new StartSelector().withStartSelectorType(StartSelectorType.EARLIEST),
        amazonKinesisVideo,
        getMediaProcessingArgumentsLocal.getMkvElementVisitor());
executorService.submit(getMediaWorker);
```

다음 단계

the section called “3 단계: 코드 실행 및 확인” (p. 92)

3 단계: 코드 실행 및 확인

Kinesis Video Stream Parser Library에는 사용자가 자신의 프로젝트에서 사용할 도구가 포함됩니다. 프로젝트에는 사용자가 설치를 확인할 때 실행할 수 있는 도구의 단위 테스트가 포함됩니다.

다음 유닛 테스트가 라이브러리에 포함됩니다.

- mkv
  - `ElementSizeAndOffsetVisitorTest`
  - `MkvValueTest`
  - `StreamingMkvReaderTest`
- 유틸리티
  - `FragmentMetadataVisitorTest`
  - `OutputSegmentMergerTest`

Amazon Kinesis 비디오 스트림 예제

다음 코드 예제는 Kinesis 비디오 스트림 API로 작업하는 방법을 보여 줍니다.

- 예제: HTML 및 JavaScript에서 HLS 사용 (p. 9): Kinesis 비디오 스트림용 HLS 스트리밍 세션을 검색하고 웹 페이지에서 재생하는 방법입니다.
- GStreamer 플러그인 (p. 93): Kinesis 비디오 스트림 생산자 SDK를 구축하여 GStreamer 대상으로 사용하는 방법을 보여 줍니다.
- 예: PutMedia API를 사용하여 Kinesis 비디오 스트림에 데이터 전송 (p. 101): PutMedia API를 사용하여 이미 컨테이너 형식(MKV)인 데이터를 전송합니다.
- 예: RTSP 소스에서 스트리밍 (p. 104): 도커 컨테이너에서 실행되고 RTSP 소스의 비디오를 스트리밍하는 샘플 애플리케이션입니다.
- 예: Kinesis 비디오 스트림와 함께 GStreamer 사용 (p. 105): GStreamer 오픈 소스 멀티미디어 프레임워크를 사용하여 Kinesis 비디오 스트림에 비디오 데이터를 전송합니다.
- 예: Kinesis 비디오 스트림 조각 구문 분석 및 렌더링 (p. 106): JCodec 및 JFrame을 사용하여 Kinesis 비디오 스트림 조각을 구문 분석하고 렌더링합니다.
- KinesisVideoExample (p. 89): Kinesis 비디오 스트림 구문 분석 라이브러리를 사용하여 비디오 조각을 구문 분석하고 로깅합니다.

사전 조건

- 샘플 코드에서는 AWS 자격 증명 프로필 파일에서 설정한 프로필을 지정하여 자격 증명을 제공하거나 IDE(integrated development environment: 통합 개발 환경)의 Java 시스템 속성에서 자격 증명을 제공합니다. 따라서 아직 설정하지 않은 경우 먼저 자격 증명을 설정합니다. 자세한 내용은 개발을 위한 AWS 자격 증명 및 리전 설정을 참조하십시오.
- Java IDE를 사용하여 코드를 보고 실행하는 것이 좋습니다(예: 다음 중 하나).
  - Eclipse Java Neon
  - JetBrains IntelliJ IDEA

예제: Kinesis 비디오 스트림 생산자 SDK GStreamer 플러그인

이 주제에서는 Amazon Kinesis 비디오 스트림 생산자 SDK를 구축하여 GStreamer 플러그인으로 사용하는 방법을 보여 줍니다.

항목

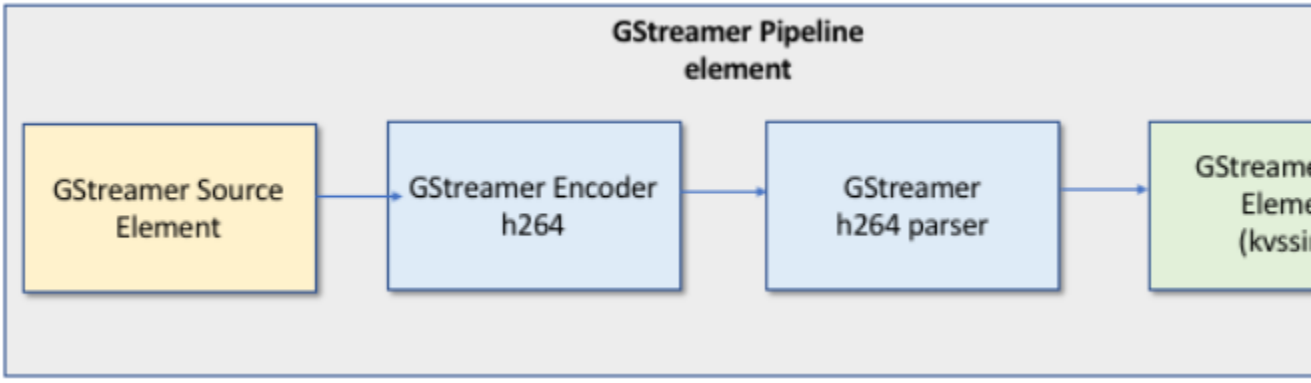
- GStreamer 요소 다운로드, 구축 및 구성 (p. 94)
- GStreamer 요소 실행 (p. 95)
- 예제 GStreamer 시작 명령 (p. 95)
- 도커 컨테이너에서 GStreamer 요소 실행 (p. 96)
- GStreamer 요소 파라미터 참조 (p. 98)

GStreamer는 수많은 카메라 및 비디오 소스에서 모듈식 플러그인을 결합하여 사용자 지정 미디어 파이프라인을 생성하는 데 사용하는 인기 있는 미디어 프레임워크입니다. Kinesis 비디오 스트림 GStreamer 플러그인은 기존 GStreamer 미디어 파이프라인과 Kinesis 비디오 스트림의 통합을 대폭 간소화합니다. GStreamer를 통합한 후에는 실시간 또는 향후 재생, 저장 및 추가 분석을 위해 웹캠 또는 RTSP(실시간 스트리밍 프로토콜) 카메라에서 Kinesis 비디오 스트림로 비디오 스트리밍을 시작할 수 있습니다.

GStreamer 플러그인은 Kinesis 비디오 스트림 생산자 SDK에서 제공된 기능을 GStreamer 싱크 요소인 `kvssink` 안에 캡슐화하여 Kinesis 비디오 스트림로 보내는 비디오 스트림 전송을 자동으로 관리합니다. GStreamer 프레임워크는 추가 처리, 렌더링 또는 저장을 위해 카메라와 같은 디바이스 또는 기타 비디오 소스의 미디어 흐름을 구성하기 위한 표준 관리형 환경을 제공합니다.

GStreamer 파이프라인은 일반적으로 소스(비디오 카메라)와 싱크 요소(비디오를 렌더링하기 위한 플레이어 또는 오프라인 검색을 위한 스토리지) 간의 링크로 구성됩니다. 이 예제에서는 생산자 SDK 요소를 싱크 또는 미디어 대상으로 비디오 소스(웹캠 또는 IP 카메라)에 사용합니다. 그런 다음 SDK를 캡슐화하는 플러그인 요소가 Kinesis 비디오 스트림로 보내는 비디오 스트림 전송을 관리합니다.

이 주제에서는 일반적으로 중간 인코딩 단계(H.264 인코딩 사용)를 통해 연결된 웹 카메라 또는 RTSP 스트림과 같은 비디오 소스에서 Kinesis 비디오 스트림로 비디오를 스트리밍할 수 있는 GStreamer 미디어 파이프라인을 구성하는 방법을 보여 줍니다. 비디오 스트림을 Kinesis 비디오 스트림로 사용할 수 있는 경우 비디오 스트림의 추가 처리, 재생, 저장 또는 분석을 위해 Kinesis Video Stream Parser Library를 사용할 수 있습니다.



GStreamer 요소 다운로드, 구축 및 구성

GStreamer 플러그인 예제는 Kinesis 비디오 스트림 C++ 생산자 SDK와 함께 포함됩니다. SDK 사전 조건 및 다운로드에 대한 자세한 내용은 1 단계: C++ 생산자 라이브러리 코드 다운로드 및 구성 (p. 39) 단원을 참조하십시오.

macOS, Ubuntu 또는 Raspberry Pi에서 생산자 SDK GStreamer 싱크를 동적 라이브러리로 구축하려면 `kinesis-video-native-build` 디렉터리에서 다음 명령을 실행합니다.

```
./gstreamer-plugin-install-script
```

싱크가 구축된 후에는 다음 디렉터리에서 `gst-launch-1.0`을 실행할 수 있습니다.

```
<YourSdkFolderPath>/kinesis-video-native-build/downloads/local/bin
```

`gst-launch-1.0`을 이 디렉터리에서 실행하거나 `PATH` 환경 변수에 추가할 수 있습니다.

```
$ export PATH=<YourSdkFolderPath>/kinesis-video-native-build/downloads/local/bin:$PATH
```

GStreamer 플러그인을 찾을 수 있도록 라이브러리 디렉터리를 경로에 추가합니다.

```
export GST_PLUGIN_PATH=<YourSdkFolderPath>/kinesis-video-native-build/downloads/local/lib:
$GST_PLUGIN_PATH
```

SDK의 라이브러리 경로를 설정합니다.

```
export LD_LIBRARY_PATH=<YourSdkFolderPath>/kinesis-video-native-build/downloads/local/lib
```

GStreamer 요소 실행

Kinesis 비디오 스트림 생산자 SDK 요소를 싱크로 사용하여 GStreamer를 실행하려면 `gst-launch-1.0` 명령을 실행합니다. 사용할 GStreamer 플러그인에 적합한 설정을 사용합니다. 예를 들어, Linux 시스템의 v412 디바이스에 v412src를 사용하거나 RTSP 디바이스에 rtspsrc를 사용할 수 있습니다. 생산자 SDK에 비디오를 전송하기 위한 싱크(파이프라인의 최종 대상)로 `kvssink`를 지정합니다.

`kvssink` 요소에는 다음과 같은 필수 파라미터가 있습니다.

- `stream-name`: 대상 Kinesis 비디오 스트림의 이름입니다.
- `storage-size`: 디바이스의 스토리지 크기(KB)입니다. 디바이스 스토리지 구성에 대한 자세한 내용은 StorageInfo (p. 71) 단원을 참조하십시오.
- `access-key`: Kinesis 비디오 스트림에 액세스하는 데 사용되는 AWS 액세스 키입니다. 이 파라미터 또는 `credential-path`를 제공해야 합니다.
- `secret-key`: Kinesis 비디오 스트림에 액세스하는 데 사용되는 AWS 비밀 키입니다. 이 파라미터 또는 `credential-path`를 제공해야 합니다.
- `credential-path`: Kinesis 비디오 스트림에 액세스하기 위한 자격 증명이 포함된 파일의 경로입니다. 예제 자격 증명 파일은 샘플 정적 자격 증명 및 샘플 교체 자격 증명을 참조하십시오. 교체 자격 증명에 대한 자세한 내용은 IAM 사용자를 위한 액세스 키 관리를 참조하십시오. 이 파라미터 또는 `access-key` 및 `secret-key`를 제공해야 합니다.

`kvssink` 선택적 파라미터에 대한 자세한 내용은 GStreamer 요소 파라미터 참조 (p. 98) 단원을 참조하십시오.

GStreamer 플러그인 및 파라미터에 대한 최신 정보는 GStreamer 플러그인을 참조하거나 다음 명령을 실행하여 옵션을 나열하십시오.

```
gst-inspect-1.0 kvssink
```

예제 GStreamer 시작 명령

이러한 예제는 GStreamer 플러그인을 사용하여 다양한 유형의 디바이스에서 비디오를 스트리밍하는 방법을 보여 줍니다.

예제 1: Ubuntu의 RTSP 카메라에서 비디오 스트리밍

다음 명령은 rtspsrc GStreamer 플러그인을 사용하여 네트워크 RTSP 카메라에서 스트리밍하는 GStreamer 파이프라인을 Ubuntu에서 생성합니다.

```
$ 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"
```

예제 2: Ubuntu의 USB 카메라에서 비디오 인코딩 및 스트리밍

다음 명령은 USB 카메라의 스트림을 H.264 형식으로 인코딩하고 Kinesis 비디오 스트림로 스트리밍하는 GStreamer 파이프라인을 Ubuntu에서 생성합니다. 이 예제에서는 v412src GStreamer 플러그인을 사용합니다.

```
$ 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"
```

예제 3: Ubuntu의 USB 카메라에서 사전 인코딩된 비디오 스트리밍

다음 명령은 카메라가 H.264 형식으로 이미 인코딩한 비디오를 Kinesis 비디오 스트림로 스트리밍하는 GStreamer 파이프라인을 Ubuntu에서 생성합니다. 이 예제에서는 v412src GStreamer 플러그인을 사용합니다.

```
$ 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"
```

예제 4: macOS의 네트워크 카메라에서 비디오 스트리밍

다음 명령은 네트워크 카메라에서 Kinesis 비디오 스트림로 비디오를 스트리밍하는 GStreamer 파이프라인을 macOS에서 생성합니다. 이 예제에서는 rtspsrc GStreamer 플러그인을 사용합니다.

```
$ 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
```

예제 5: Raspberry Pi의 카메라에서 비디오 스트리밍

다음 명령은 Kinesis 비디오 스트림로 비디오를 스트리밍하는 GStreamer 파이프라인을 Raspberry Pi에서 생성합니다. 이 예제에서는 v412src GStreamer 플러그인을 사용합니다.

```
$ 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" frame-timestamp=dts-only access-key="YourAccessKey" secret-
key="YourSecretKey"
```

예제 6: Raspberry Pi의 카메라로부터 비디오 스트리밍 및 리전 지정

다음 명령은 미국 동부(버지니아 북부) 리전의 Kinesis 비디오 스트림로 비디오를 스트리밍하는 GStreamer 파이프라인을 Raspberry Pi에서 생성합니다. 이 예제에서는 v412src GStreamer 플러그인을 사용합니다.

```
$ 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" frame-timestamp=dts-only access-key="YourAccessKey" secret-
key="YourSecretKey" aws-region=us-east-1
```

도커 컨테이너에서 GStreamer 요소 실행

도커는 컨테이너를 사용하여 애플리케이션을 개발, 배포 및 실행하기 위한 플랫폼입니다. 도커를 사용하여 GStreamer 파이프라인을 생성하면 Kinesis 비디오 스트림에 대한 운영 환경이 표준화되어 애플리케이션 구축 및 실행이 대폭 간소화됩니다.

도커를 설치하고 구성하려면 다음을 참조하십시오.

- 도커 다운로드 지침
- 도커 시작하기

도커를 설치한 후에는 `docker pull` 명령을 사용하여 Amazon Elastic Container Registry에서 Kinesis 비디오 스트림 C++ 생산자 SDK(및 GStreamer 플러그인)를 다운로드할 수 있습니다.

도커 컨테이너에서 Kinesis 비디오 스트림 생산자 SDK 요소를 싱크로 사용하여 GStreamer를 실행하려면 다음을 수행합니다.

항목

도커 클라이언트 인증

이미지를 가져오려는 Amazon ECR 레지스트리에 대해 Docker 클라이언트를 인증합니다. 사용되는 각 레지스트리에 대한 인증 토큰을 가져와야 하며 토큰은 12시간 동안 유효합니다. 자세한 내용은 Amazon Elastic Container Registry 사용 설명서의 레지스트리 인증을 참조하십시오.

Example : Amazon ECR을 사용하여 인증

```
aws ecr get-login --no-include-email --region us-west-2 --registry-ids 546150905175
```

위의 명령은 다음과 비슷한 출력을 생성합니다.

```
docker login -u AWS -p <Password>   https://YourAccountId.dkr.ecr.us-west-2.amazonaws.com
```

결과 출력은 Amazon ECR 레지스트리에 대해 도커 클라이언트를 인증하는 데 사용하는 도커 로그인 명령입니다.

Ubuntu, macOS 또는 Raspberry Pi용 도커 이미지 다운로드

운영 체제에 따라 다음 명령 중 하나를 사용하여 도커 이미지를 도커 환경으로 다운로드합니다.

Ubuntu용 도커 이미지 다운로드

```
sudo docker pull 546150905175.dkr.ecr.us-west-2.amazonaws.com/kinesis-video-producer-sdk-cpp-amazon-linux:latest
```

macOS용 도커 이미지 다운로드

```
sudo docker pull 546150905175.dkr.ecr.us-west-2.amazonaws.com/kinesis-video-producer-sdk-cpp-amazon-linux:latest
```

Raspberry Pi용 도커 이미지 다운로드

```
sudo docker pull 546150905175.dkr.ecr.us-west-2.amazonaws.com/kinesis-video-producer-sdk-cpp-raspberry-pi:latest
```

이미지가 성공적으로 추가되었는지 확인하려면 다음 명령을 사용합니다.

```
docker images
```

도커 이미지 실행

운영 체제에 따라 다음 명령 중 하나를 사용하여 도커 이미지를 실행합니다.

Ubuntu에서 도커 이미지 실행

```
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
```

macOS에서 도커 이미지 실행

```
sudo docker run -it --network="host" 546150905175.dkr.ecr.us-west-2.amazonaws.com/kinesis-
video-producer-sdk-cpp-amazon-linux /bin/bash
```

Raspberry Pi에서 도커 이미지 실행

```
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
```

도커가 컨테이너를 시작하고, 컨테이너 내에서 명령을 실행하기 위한 명령 프롬프트를 표시합니다.

컨테이너에서 다음 명령을 사용하여 환경 변수를 설정합니다.

```
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
```

디바이스에 적절한 `gst-launch-1.0` 명령을 사용하여 카메라에서 스트리밍을 시작합니다.

> **Note**
>
> macOS에서는 도커 컨테이너에서 GStreamer를 실행할 때만 네트워크 카메라에서 비디오를 스트리밍할 수 있습니다. 도커 컨테이너에서 macOS의 USB 카메라에서 비디오를 스트리밍하는 작업은 지원되지 않습니다.

`gst-launch-1.0` 명령을 사용하여 로컬 웹 카메라 또는 네트워크 RTSP 카메라에 연결하는 작업의 예는 시작 명령 (p. 95) 단원을 참조하십시오.

GStreamer 요소 파라미터 참조

비디오를 Amazon Kinesis 비디오 스트림 생산자 SDK로 전송하려면 `kvssink`를 싱크 또는 파이프라인의 최종 대상으로 지정합니다. 이 참조는 `kvssink` 필수 및 선택적 파라미터에 대한 정보를 제공합니다. 자세한 내용은 the section called “GStreamer 플러그인” (p. 93) 단원을 참조하십시오.

`kvssink` 요소에는 다음과 같은 필수 파라미터가 있습니다.

- `stream-name`: 대상 Kinesis 비디오 스트림의 이름입니다.
- `storage-size`: 디바이스의 스토리지 크기(KB)입니다. 디바이스 스토리지 구성에 대한 자세한 내용은 StorageInfo (p. 71) 단원을 참조하십시오.
- `access-key`: Kinesis 비디오 스트림에 액세스하는 데 사용되는 AWS 액세스 키입니다. 이 파라미터 또는 `credential-path`를 제공해야 합니다.

- `secret-key`: Kinesis 비디오 스트림에 액세스하는 데 사용되는 AWS 비밀 키입니다. 이 파라미터 또는 `credential-path`를 제공해야 합니다.
- `credential-path`: Kinesis 비디오 스트림에 액세스하기 위한 자격 증명이 포함된 파일의 경로입니다. 예제 자격 증명 파일은 샘플 정적 자격 증명 및 샘플 교체 자격 증명을 참조하십시오. 이 파라미터 또는 `access-key` 및 `secret-key`를 제공해야 합니다.

`kvssink` 요소에는 다음과 같은 선택적 파라미터가 있습니다. 이런 파라미터에 대한 자세한 내용은 Kinesis 비디오 스트림 구조 (p. 72) 단원을 참조하십시오.

파라미터	설명	단위/유형	기본값
`absolute-fragment-times`	절대 조각 시간을 사용할지 여부입니다.	부울	true
`avg-bandwidth-bps`	스트림용 예상 평균 대역폭.	초당 바이트	4194304
`aws-region`	사용할 AWS 리전입니다.	문자열	`us-west-2`
`buffer-duration`	스트림 버퍼 지속 시간입니다.	초	180
`codec-id`	스트림의 코덱 ID입니다.	문자열	`"V_MPEG4/ISO/AVC"`
`connection-staleness`	스트림 기한 경과 콜백이 호출될 때까지의 시간입니다.	초	60
`content-type`	스트림의 콘텐츠 유형입니다.	문자열	`"video/h264"`
`fragment-acks`	조각 ACK를 사용할지 여부입니다.	부울	true
`fragment-duration`	원하는 조각 지속 시간입니다.	밀리초	2000
`framerate`	예상 프레임 속도입니다.	초당 프레임	25
`frame-timecodes`	프레임 타임코드를 사용할지 또는 현재 시간 콜백을 사용하여 타임스탬프를 생성할지를 지정합니다.	부울	true
`frame-timestamp`	• (0): pts만 해당: Kinesis 비디오 스트림에 전송되는 모든 프레임에 대해 디코딩 타임스탬프(DTS)를 프레젠테이션 타임스탬프(PTS)와 같게 설정합니다. • (1): dts만 해당: Kinesis 비디오 스트림에 전송되는 모든 프레	열거형 `GstKvsSinkFrameTimestampType`	`default-timestamp`

파라미터	설명	단위/유형	기본값
	임에 대해 PTS를 DTS와 같게 설정합니다. • (2): 기본 타임스탬프: PTS와 DTS를 모두 사용해 봅니다. 하나를 사용할 수 없으면 다른 하나를 사용합니다.		
`key-frame-fragmentation`	키 프레임에서 조각을 생성할지 여부입니다.	부울	true
`log-config`	로그 구성 경로입니다.	문자열	`"./kvs_log_configuration"`
`max-latency`	스트림의 최대 지연 시간입니다.	초	60
`recalculate-metrics`	지표를 다시 계산할지 여부입니다.	부울	true
`replay-duration`	다시 시작이 활성화된 경우 오류가 발생한 동안 현재 리더를 뒤로 롤하여 다시 재생하는 기간입니다.	초	40
`restart-on-error`	오류 발생 시 다시 시작할지 여부입니다.	부울	true
`retention-period`	스트림이 보존되는 시간의 길이입니다.	시간	2
`rotation-period`	키 교체 기간입니다. 자세한 내용은 고객 마스터 키 교체를 참조하십시오.	초	2400
`streaming-type`	스트리밍 유형입니다. 유효한 값으로는 다음이 포함됩니다. • 0: 실시간 • 1: 근 실시간(현재 지원되지 않은) • 2: 오프라인(현재 지원되지 않음)	열거형 `GstKvsSinkStreamingType`	0: 실시간
`timecode-scale`	MKV 타임코드 척도입니다.	밀리초	1
`track-name`	MKV 트랙 이름	문자열	`"kinesis_video"`

예: PutMedia API를 사용하여 Kinesis 비디오 스트림에 데이터 전송

이 예제는 PutMedia API를 사용하는 방법을 보여 줍니다. 이것은 컨테이너 형식(MKV)으로 되어 있는 데이터를 전송하는 방법을 설명합니다. 전송하기 전에 데이터를 컨테이너 형식으로 조합해야 하는 경우(예를 들어 카메라 비디오 데이터를 프레임으로 조합하는 경우)에는 Kinesis 비디오 스트림 생산자 라이브러리 (p. 27) 단원을 참조하십시오.

Note

`PutMedia` 작업의 경우 연결, 데이터 흐름 및 인정의 전이중 관리로 인해 C++ 및 Java SDK에서만 가용합니다. 다른 언어에서는 지원되지 않습니다.

이 예제에는 다음 단계가 포함됩니다.

- 1 단계: 코드 다운로드 및 구성 (p. 101)
- 2 단계: 코드 쓰기 및 검사 (p. 102)
- 3 단계: 코드 실행 및 확인 (p. 103)

1 단계: 코드 다운로드 및 구성

이 단원에서는 Java 예제 코드를 다운로드하고, 프로젝트를 Java IDE에 가져오고, 라이브러리 위치를 구성하고, AWS 자격 증명을 사용하도록 코드를 구성합니다.

1. 디렉터리를 생성하고 GitHub 리포지토리에서 예제 소스 코드를 복제합니다. `PutMedia` 예제는 Java 생산자 라이브러리 (p. 28)의 일부입니다.

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

2. 사용 중인 Java IDE(예: Eclipse 또는 IntelliJ IDEA)를 열고 다운로드한 Apache Maven 프로젝트를 가져옵니다.

   - Eclipse에서 파일, 가져오기, Maven, Existing Maven Projects(기존 Maven 프로젝트)를 차례로 선택한 후 다운로드한 패키지의 루트로 이동합니다. `pom.xml` 파일을 선택합니다.
   - IntelliJ Idea에서 [Import]를 선택합니다. 다운로드한 패키지의 루트에 있는 `pom.xml` 파일을 찾습니다.

   자세한 내용은 관련 IDE 문서를 참조하십시오.

3. 가져온 라이브러리를 IDE가 알 수 있도록 프로젝트를 업데이트합니다.
   - IntelliJ IDEA의 경우 다음을 수행합니다.
     a. 프로젝트의 [lib] 디렉터리의 컨텍스트 메뉴를 열고(오른쪽 버튼 클릭) [Add as library]를 선택합니다.
     b. [File]과 [Project Structure...]를 선택합니다.
     c. [Project Settings]에서 [Modules]를 선택합니다.
     d. 원본 탭에서 Language Level(언어 수준)을 `7` 이상으로 설정합니다.
   - Eclipse의 경우 다음을 수행합니다.
     a. 프로젝트의 컨텍스트 메뉴를 열고(오른쪽 버튼 클릭) [Properties], [Java Build Path], [Source]를 차례로 선택합니다. 뒤이어 다음과 같이 하십시오.
        1. [Source] 탭에서 [Native library location]을 두 번 클릭합니다.

2. [Native Library Folder Configuration] 마법사에서, [Workspace]를 선택합니다.
3. [Native Library Folder] 선택에서 프로젝트에 있는 [lib] 디렉터리를 선택합니다.

b. 프로젝트에 대한 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) [Properties]를 선택합니다. 뒤이어 다음과 같이 하십시오.

1. [Libraries] 탭에서 [Add Jars]를 선택합니다.
2. [JAR selection] 마법사에서 프로젝트의 `lib` 디렉터리에 있는 모든 .jar을 선택합니다.

2 단계: 코드 쓰기 및 검사

`PutMedia` API 예제()는 다음과 같은 코딩 패턴을 보입니다.`PutMediaDemo`

항목

- PutMediaClient 생성 (p. 102)
- 미디어 스트리밍 및 스레드 일시 중지 (p. 103)

이 단원의 코드 예제는 `PutMediaDemo` 클래스가 출처입니다.

PutMediaClient 생성

`PutMediaClient` 객체 생성 시에는 다음 파라미터를 취합니다.

- `PutMedia` 엔드포인트의 URI.
- 스트리밍할 MKV 파일로 향하는 `InputStream`
- 스트림 이름입니다. 이 예제에서는 Java 생산자 라이브러리 사용하기 (p. 28)에서 생성된 스트림을 사용합니다(`my-stream`). 다른 스트림을 사용하려면 다음 파라미터를 변경합니다.

```
private static final String STREAM_NAME="my-stream";
```

> **Note**
>
> `PutMedia` API 예제는 스트림을 생성하지 않으므로 사용자가 Java 생산자 라이브러리 사용하기 (p. 28)용 테스트 애플리케이션을 사용하거나, Kinesis 비디오 스트림 콘솔을 사용하거나, AWS CLI를 사용하여 스트림을 생성해야 합니다.

- 현재 타임스탬프.
- 타임 코드 유형. 예제에서는 `RELATIVE`를 사용하는데, 타임스탬프가 컨테이너 시작 기준이라는 것을 가리킵니다.
- 수신된 패킷이 인가된 발신자에 의해 전송되었음을 확인하는 `AWSKinesisVideoV4Signer` 객체.
- 최대 업스트림 대역폭(단위: Kbps)
- 패킷을 받았다는 인정을 수신하는 `AckConsumer` 객체.

다음 코드는 `PutMediaClient` 객체를 생성합니다.

```
/* 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();
```

미디어 스트리밍 및 스레드 일시 중지

클라이언트가 생성된 후에는 샘플이 `putMediaInBackground`로 비동기식 스트리밍을 시작합니다. 그러면 메인 스레드가 `AckConsumer`가 반환될 때 까지, 즉 클라이언트가 닫히는 시점까지 `latch.await`에 의해 일시 중지됩니다.

```
/* start streaming video in a background thread */
           client.putMediaInBackground();

           /* wait for request/response to complete */
           latch.await();

           /* close the client */
           client.close();
```

3 단계: 코드 실행 및 확인

`PutMedia` API 예제를 실행하려면 다음을 수행합니다.

1. Kinesis 비디오 스트림 콘솔에서 또는 AWS CLI를 사용하여 `my-stream`이라는 스트림을 생성합니다.
2. 작업 디렉터리를 Java 생산자 SDK 디렉터리로 변경합니다.

   ```
   $ cd /<YOUR_FOLDER_PATH_WHERE_SDK_IS_DOWNLOADED>/amazon-kinesis-video-streams-producer-sdk-java/
   ```

3. Java SDK와 데모 애플리케이션을 컴파일합니다.

   ```
   mvn package
   ```

4. `/tmp` 디렉터리에 임시 파일 이름을 생성합니다.

   ```
   $ jar_files=$(mktemp)
   ```

5. 로컬 리포지토리에서 파일로 종속성의 classpath 문자열을 생성합니다.

   ```
   $ mvn -Dmdep.outputFile=$jar_files dependency:build-classpath
   ```

6. 다음과 같이 `LD_LIBRARY_PATH`환경 변수의 값을 설정합니다.

```
$ 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. AWS 자격 증명을 제공하여 다음과 같이 명령줄에서 데모를 실행합니다.

```
$ 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. https://console.aws.amazon.com/kinesisvideo/에서 Kinesis 비디오 스트림 콘솔을 열고 [Manage Streams] 페이지에서 해당 스트림을 선택합니다. 비디오가 [Video Preview] 창에서 재생됩니다.

예: RTSP 소스에서 스트리밍

C++ 생산자 라이브러리 (p. 36)에는 RTSP(실시간 스트리밍 프로토콜) 네트워크 카메라에 연결하는 도커 컨테이너에 대한 정의가 포함되어 있습니다. 도커를 사용하면 Kinesis 비디오 스트림에 대한 운영 환경이 표준화되어 애플리케이션 구축 및 실행이 대폭 간소화됩니다.

RTSP 데모 애플리케이션을 사용하려면 먼저 C++ 생산자 라이브러리 (p. 36)를 설치하고 구축합니다.

다음 절차에서는 RTSP 데모 애플리케이션을 설정하고 사용하는 방법을 보여 줍니다.

항목

사전 조건

Kinesis 비디오 스트림 RTSP 예제 애플리케이션을 실행하려면 다음이 필요합니다.

- 도커: 도커 설치 및 사용에 대한 자세한 내용은 다음 링크를 참조하십시오.
  - 도커 다운로드 지침
  - 도커 시작하기
- RTSP 네트워크 카메라 소스: 권장 카메라에 대한 자세한 내용은 시스템 요구 사항 (p. 3) 단원을 참조하십시오.

도커 이미지 구축

먼저 내부에서 데모 애플리케이션이 실행되는 도커 이미지를 구축합니다.

1. 새 디렉터리를 생성하고 `docker_native_scripts` 디렉터리에서 새 디렉터리로 다음 파일을 복사합니다.
   - `Dockerfile`
   - `start_rtsp_in_docker.sh`
2. 이전 단계에서 생성한 디렉터리로 변경합니다.

3. 다음 명령을 사용하여 도커 이미지를 구축합니다. 이 명령은 이미지를 생성하고 `rtspdockertest`로 태그 지정합니다.

```
docker build -t rtspdockertest .
```

4. 이전 단계에서 반환된 이미지 ID를 기록합니다(예: *54f0d65f69b2*).

RTSP 예제 애플리케이션 실행

다음 명령을 사용하여 Kinesis 비디오 스트림 도커 컨테이너를 시작합니다. 이전 단계의 이미지 ID, AWS 자격 증명, RTSP 네트워크 카메라의 URL, 데이터를 전송할 Kinesis 비디오 스트림의 이름을 제공합니다.

```
$ docker run -
it <IMAGE_ID> <AWS_ACCESS_KEY_ID> <AWS_SECRET_ACCESS_KEY> <RTSP_URL> <STREAM_NAME>
```

애플리케이션을 사용자 지정하려면 `Dockerfile`에서 `ENTRYPOINT` 명령을 설명하거나 제거하고 다음 명령을 사용하여 컨테이너를 시작합니다.

```
docker run -it <IMAGE_ID> bash
```

그러면 샘플 애플리케이션을 사용자 지정하고 스트리밍을 시작하기 위한 메시지가 도커 컨테이너 내부에 나타납니다.

예: Kinesis 비디오 스트림와 함께 GStreamer 사용

C++ 생산자 라이브러리 (p. 36)에는 오픈 소스 멀티미디어 프레임워크인 GStreamer를 Amazon Kinesis 비디오 스트림와 함께 사용하기 위한 데모 애플리케이션이 포함되어 있습니다.

다음 절차에서는 GStreamer 데모를 설정하고 사용하는 방법을 소개합니다. 이 애플리케이션은 다음 플랫폼에서 Kinesis 비디오 스트림에 비디오 데이터를 전송합니다.

- macOS 컴퓨터의 내장 카메라
- Linux 또는 Raspberry Pi 디바이스의 USB 카메라

> **Note**
>
> 현재 Windows 시스템에서는 GStreamer 예제를 사용할 수 없습니다.

사전 조건

GStreamer 예제에는 일반적으로 다음 구성 요소가 필요하며, 다음 명령을 사용하면 이 구성 요소를 설치할 수 있습니다.

- Automake: `brew install automake`
- Autoconf: `brew install autoconf`
- CMake: `brew install cmake`
- GStreamer: `brew install gstreamer gst-plugins-base gst-plugins-bad gst-plugins-good gst-plugins-ugly`
- GNU Bison: `brew install bison && brew link bison --force`

보다 종합적인 요구 사항 목록은 C++ 생산자 SDK 설명서의 사전 조건 (p. 38) 또는 SDK 자체의 `README.md` 파일을 참조하십시오.

GStreamer 예제 실행

1. 디렉터리를 생성한 다음 GitHub 리포지토리에서 예제 소스 코드를 복제합니다.

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

2. `/kinesis-video-native-build` 디렉터리에서 다음 스크립트를 실행하여 C++ 생산자 SDK를 빌드합니다.

```
./install-script
```

   이 스크립트는 SDK의 사전 요건을 설치하고 GStreamer 예제를 포함하여 바이너리를 빌드합니다. Ubuntu에서의 예제 빌드에 대한 정보는 SDK `README.md` 파일의 "문제 해결" 단원을 참조하십시오.

3. C++ 생산자 SDK를 설치하고 구성했으면 다음 명령(`/kinesis-video-native-build` 디렉터리에 있음)을 사용하여 애플리케이션을 실행합니다.

```
export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
export AWS_DEFAULT_REGION=<AWS region>
./kinesis_video_gstreamer_sample_app stream_name
```

   애플리케이션 파라미터에 다음 정보를 제공합니다.

   - AWS_ACCESS_KEY_ID: 계정의 액세스 키 ID입니다. 사전 조건 (p. 93) 단원을 참조하십시오.
   - AWS_SECRET_ACCESS_KEY: 계정의 보안 액세스 키입니다. 사전 조건 (p. 93) 단원을 참조하십시오.
   - AWS_DEFAULT_REGION: AWS 리전입니다(예: `us-west-2`).
   - stream_name: 카메라 데이터를 전송할 Kinesis 비디오 스트림의 이름입니다. 스트림이 존재하지 않으면 생성됩니다.

4. 데모 애플리케이션이 시작됩니다. 몇 초 안에 스트림의 Kinesis 비디오 스트림 콘솔에서 카메라의 비디오 데이터를 볼 수 있습니다.

   Note

   애플리케이션이 카메라를 획득하지 못하면 `Failed to negotiate pipeline` 오류가 표시될 수 있습니다. 문제 해결 정보는 SDK의 `README.md` 파일을 참조하십시오.

예: Kinesis 비디오 스트림 조각 구문 분석 및 렌더링

스트림 구문 분석기 라이브러리 (p. 86)에는 Amazon Kinesis 비디오 스트림 조각 구문 분석 및 렌더링을 보여주는 `KinesisVideoRendererExample` 데모 애플리케이션이 들어 있습니다. 이 예제에서는 JCodec을 사용하여 예: Kinesis 비디오 스트림와 함께 GStreamer 사용 (p. 105) 애플리케이션을 사용해 수집된 H.264 인코딩 프레임을 디코딩합니다. JCodec을 사용하여 프레임을 디코딩한 후에는 JFrame을 사용하여 표시된 이미지가 렌더링됩니다.

이 예에서는 다음 작업을 수행하는 방법을 보여줍니다.

- `GetMedia` API를 사용하여 Kinesis 비디오 스트림에서 프레임을 검색하고 볼 수 있도록 스트림을 렌더링합니다.
- Kinesis 비디오 스트림 콘솔을 사용하는 대신 사용자 지정 애플리케이션에서 스트림의 비디오 콘텐츠를 봅니다.

또한 이 예의 클래스를 사용하여 H.264로 인코딩되지 않은 Kinesis 비디오 스트림 콘텐츠를 봅니다(예: 표시되기 전에 디코딩이 필요 없는 JPEG 파일의 스트림).

다음 절차에서는 렌더러 데모 애플리케이션을 설정하고 사용하는 방법을 소개합니다.

사전 조건

렌더러 예제 라이브러리를 검사하고 사용하려면 다음이 필요합니다.

- Amazon Web Services (AWS) 계정. AWS 계정이 아직 없는 경우 Kinesis 비디오 스트림 시작을 참조하십시오.
- Eclipse Java Neon 또는 JetBrains IntelliJ Idea 같은 Java 통합 개발 환경(IDE).

렌더러 예제 실행

1. 디렉터리를 생성한 다음 GitHub 리포지토리에서 예제 소스 코드를 복제합니다.

```
$ git clone https://github.com/aws/amazon-kinesis-video-streams-parser-library
```

2. 사용 중인 Java IDE(예: Eclipse 또는 IntelliJ IDEA)를 열고 다운로드한 Apache Maven 프로젝트를 가져옵니다.

   - Eclipse: [File], [Import], [Maven], [Existing Maven Projects]를 차례로 선택합니다. `kinesis-video-streams-parser-lib` 디렉터리로 이동합니다.
   - IntelliJ Idea에서 [Import]를 선택합니다. 다운로드한 패키지의 루트에 있는 `pom.xml` 파일을 찾습니다.

     Note

     IntelliJ가 종속성을 찾을 수 없는 경우 다음을 수행해야 할 수 있습니다.

     - 빌드 정리: File(파일), Settings(설정), Build, Execution, Deployment(구축, 실행, 배포), Compiler(컴파일러)를 선택합니다. Clear output directory on rebuild(재구축 시 출력 디렉터리 정리)가 선택되어 있는지 확인한 다음 Build(구축), Build Project(프로젝트 구축)를 선택합니다.
     - 프로젝트 다시 가져오기: 프로젝트를 마우스 오른쪽 버튼으로 클릭하고 Maven, Reimport(다시 가져오기)를 선택합니다.

   자세한 내용은 관련 IDE 문서를 참조하십시오.
3. Java IDE에서 `src/test/java/com.amazonaws.kinesisvideo.parser/examples/KinesisVideoRendererExampleTest`를 엽니다.
4. 파일에서 `@Ignore` 명령을 제거합니다.
5. Kinesis 비디오 스트림의 이름으로 `.stream` 파라미터를 업데이트합니다.
6. `KinesisVideoRendererExample` 테스트를 실행합니다.

사용 방법

이 예제 애플리케이션은 다음과 같은 방법을 보여줍니다.

- MKV 데이터 전송 (p. 108)
- MKV 조각을 프레임으로 구문 분석 (p. 108)
- 프레임 디코딩 및 표시 (p. 108)

MKV 데이터 전송

이 예제에서는 `PutMedia`를 사용하여 render-example-stream이라는 스트림으로 비디오 데이터를 전송하는 방식으로 `rendering_example_video.mkv` 파일에서 샘플 MKV 데이터를 전송합니다.

이 애플리케이션에서는 `PutMediaWorker`를 생성합니다.

```
PutMediaWorker putMediaWorker = PutMediaWorker.create(getRegion(),
    getCredentialsProvider(),
    getStreamName(),
    inputStream,
    streamOps.amazonKinesisVideo);
executorService.submit(putMediaWorker);
```

`PutMediaWorker` 클래스에 대한 자세한 내용은 스트림 구문 분석기 라이브러리 (p. 86) 설명서의 PutMedia 호출 (p. 90) 단원을 참조하십시오.

MKV 조각을 프레임으로 구문 분석

그런 다음 이 예에서는 `GetMediaWorker`를 사용하여 스트림에서 MKV 조각을 검색해 구문 분석합니다.

```
GetMediaWorker getMediaWorker = GetMediaWorker.create(getRegion(),
    getCredentialsProvider(),
    getStreamName(),
    new StartSelector().withStartSelectorType(StartSelectorType.EARLIEST),
    streamOps.amazonKinesisVideo,
    getMediaProcessingArgumentsLocal.getFrameVisitor());
executorService.submit(getMediaWorker);
```

`GetMediaWorker` 클래스에 대한 자세한 내용은 스트림 구문 분석기 라이브러리 (p. 86) 설명서의 GetMedia 호출 (p. 90) 단원을 참조하십시오.

프레임 디코딩 및 표시

이 예에서는 JFrame을 사용하여 프레임을 디코딩하여 표시합니다.

다음 코드 예제는 `KinesisVideoFrameViewer` 클래스에서 가져온 것으로, `JFrame`을 확장합니다.

```
 public void setImage(BufferedImage bufferedImage) {
    image = bufferedImage;
    repaint();
}
```

이미지는 java.awt.image.BufferedImage의 인스턴스로 표시됩니다. `BufferedImage` 작업 방법을 보여주는 예제는 이미지 읽기/로드를 참조하십시오.

Kinesis 비디오 스트림 모니터링

모니터링은 Kinesis 비디오 스트림와 AWS 솔루션의 안정성, 가용성 및 성능을 유지하는 데 있어서 중요한 부분입니다. 발생하는 다중 지점 실패를 보다 쉽게 디버깅할 수 있도록 AWS 솔루션의 모든 부분으로부터 모니터링 데이터를 수집해야 합니다. 하지만 Kinesis 비디오 스트림 모니터링을 시작하기 전에 다음 질문에 대한 답변을 포함하는 모니터링 계획을 작성해야 합니다.

- 모니터링의 목표
- 모니터링할 리소스
- 이러한 리소스를 모니터링하는 빈도
- 사용할 모니터링 도구
- 모니터링 작업을 수행할 사람
- 문제 발생 시 알려야 할 대상

모니터링 목표를 정의하고 모니터링 계획을 생성했으면, 다음 단계는 환경에서 Kinesis 비디오 스트림 성능의 기준선을 설정하는 것입니다. 다양한 시간과 다양한 부하 조건에서 Kinesis 비디오 스트림 성능을 측정해야 합니다. Kinesis 비디오 스트림를 모니터링할 때 수집한 모니터링 데이터의 기록을 저장해야 합니다. 현재 Kinesis 비디오 스트림 성능을 이 기록 데이터와 비교하면 일반적인 성능 패턴과 성능 이상을 식별하고, 발생할 수 있는 문제를 해결할 방법을 찾아낼 수 있습니다.

항목

CloudWatch로 Kinesis 비디오 스트림 측정치 모니터링

Kinesis 비디오 스트림에서 원시 데이터를 수집하여 읽기 가능하며 실시간에 가까운 지표로 처리하는 Amazon CloudWatch를 사용해 Kinesis 비디오 스트림를 모니터링할 수 있습니다. 이러한 통계는 15개월간 기록되므로 기록 정보를 보고 웹 애플리케이션이나 서비스가 어떻게 실행되고 있는지 전체적으로 더 잘 파악할 수 있습니다.

Kinesis 비디오 스트림에 대한 CloudWatch 대시보드에 액세스하려면 스트림 콘솔 페이지의 [Stream info] 섹션에서 [View stream metrics in CloudWatch]를 선택합니다.

Kinesis 비디오 스트림가 지원하는 사용 가능한 측정치 목록은 Kinesis 비디오 스트림 측정치 및 차원을 참조하십시오.

CloudWatch 측정치 지침

CloudWatch 측정치는 다음 질문에 대한 답을 찾는 데 유용할 수 있습니다.

항목

데이터가 Kinesis 비디오 스트림 서비스에 도달합니까?

관련 측정치:

- `PutMedia.IncomingBytes`
- `PutMedia.IncomingFragments`
- `PutMedia.IncomingFrames`

작업 항목:

- 이러한 측정치가 하락하는 경우, 애플리케이션이 계속 서비스로 데이터를 보내고 있는지 확인합니다.
- 네트워크 대역폭을 확인합니다. 네트워크 대역폭이 충분하지 않으면 서비스에서 데이터를 수신하는 속도가 느려질 수 있습니다.

Kinesis 비디오 스트림 서비스에서 데이터를 수집하지 못하는 이유는 무엇입니까?

관련 측정치:

- `PutMedia.Requests`
- `PutMedia.ConnectionErrors`
- `PutMedia.Success`
- `PutMedia.ErrorAckCount`

작업 항목:

- `PutMedia.ConnectionErrors`가 상승하는 경우 생산자 클라이언트가 수신한 HTTP 응답/오류 코드를 확인해 연결을 설정하는 동안 어떤 오류가 발생하고 있는지 파악합니다.
- `PutMedia.Success`가 하락하거나 `PutMedia.ErrorAckCount`가 상승하는 경우 서비스에서 보낸 ack 응답의 ack 오류 코드를 확인해 데이터 수집이 실패하고 있는 이유를 파악합니다. 자세한 내용은 AckErrorCode.Values 단원을 참조하십시오.

Kinesis 비디오 스트림 서비스에서 생산자로부터 전송되는 것과 동일한 속도로 데이터를 읽을 수 없는 이유는 무엇입니까?

관련 측정치:

- `PutMedia.FragmentIngestionLatency`
- `PutMedia.IncomingBytes`

작업 항목:

- 이러한 측정치가 하락하는 경우, 연결의 네트워크 대역폭을 확인합니다. 낮은 대역폭 연결로 인해 데이터가 낮은 속도로 서비스에 도달할 수 있습니다.

콘솔에 비디오가 없거나 비디오가 지연되어 재생되는 이유는 무엇입니까?

관련 측정치:

- `PutMedia.FragmentIngestionLatency`
- `PutMedia.FragmentPersistLatency`
- `PutMedia.Success`
- `ListFragments.Latency`
- `PutMedia.IncomingFragments`

작업 항목:

- `PutMedia.FragmentIngestionLatency`가 상승하거나 `PutMedia.IncomingFragments`가 하락하는 경우 네트워크 대역폭과 데이터 전송 여부를 확인합니다.
- `PutMedia.Success`가 하락하는 경우 ack 오류 코드를 확인합니다. 자세한 내용은 AckErrorCode.Values 단원을 참조하십시오.
- `PutMedia.FragmentPersistLatency`가 상승하거나 `ListFragments.Latency`가 하락하는 경우 서비스 문제일 가능성이 가장 높습니다. 이 상태가 오랜 기간 지속되면 고객 서비스 담당자에게 문의하여 서비스에 문제가 있는지 확인하십시오.

실시간 데이터 읽기가 지연되는 이유는 무엇이며 클라이언트가 스트림의 헤드보다 지연되는 이유는 무엇입니까?

관련 측정치:

- `GetMedia.MillisBehindNow`
- `GetMedia.ConnectionErrors`
- `GetMedia.Success`

작업 항목:

- `GetMedia.ConnectionErrors`가 상승하는 경우 스트림에 다시 연결하려고 빈번하게 시도함에 따라 소비자가 스트림을 읽을 때 지연될 수 있습니다. `GetMedia` 요청에 반환된 HTTP 응답/오류 코드를 확인합니다.
- `GetMedia.Success`가 하락하는 경우, 서비스가 소비자에게 데이터를 보낼 수 없기 때문일 가능성이 높습니다. 이로 인해 연결이 끊어지고 소비자가 다시 연결을 시도함에 따라, 소비자가 스트림의 헤드보다 지연될 가능성이 있습니다.
- `GetMedia.MillisBehindNow`가 상승하는 경우, 대역폭 제한을 확인하여 낮은 대역폭 때문에 느린 속도로 데이터를 수신하고 있는지 파악합니다.

클라이언트가 Kinesis 비디오 스트림로부터 데이터를 읽습니까? 속도는 어느 정도입니까?

관련 측정치:

- `GetMedia.OutgoingBytes`
- `GetMedia.OutgoingFragments`
- `GetMedia.OutgoingFrames`

- `GetMediaForFragmentList.OutgoingBytes`
- `GetMediaForFragmentList.OutgoingFragments`
- `GetMediaForFragmentList.OutgoingFrames`

작업 항목:

- 이 측정치는 실시간 및 보관된 데이터가 읽히는 속도를 나타냅니다.

클라이언트가 Kinesis 비디오 스트림에서 데이터를 읽을 수 없는 이유는 무엇입니까?

관련 측정치:

- `GetMedia.ConnectionErrors`
- `GetMedia.Success`
- `GetMediaForFragmentList.Success`
- `PutMedia.IncomingBytes`

작업 항목:

- `GetMedia.ConnectionErrors`가 상승하는 경우 `GetMedia` 요청에서 반환한 HTTP 응답/오류 코드를 확인합니다. 자세한 내용은 AckErrorCode.Values 단원을 참조하십시오.
- 최신/실시간 데이터를 읽으려는 경우 `PutMedia.IncomingBytes`를 확인하여 서비스가 소비자에게 보낼 스트림으로 들어오는 데이터가 있는지 파악합니다.
- `GetMedia.Success` 또는 `GetMediaForFragmentList.Success`가 하락하는 경우 서비스가 소비자에게 데이터를 보낼 수 없기 때문일 가능성이 높습니다. 이 상태가 오랜 기간 지속되면 고객 서비스 담당자에게 문의하여 서비스에 문제가 있는지 확인하십시오.

AWS CloudTrail을 사용하여 Kinesis 비디오 스트림 API 호출 로깅

Amazon Kinesis 비디오 스트림는 Amazon Kinesis 비디오 스트림에서 사용자, 역할 또는 AWS 서비스가 수행한 작업에 대한 레코드를 제공하는 서비스인 AWS CloudTrail과 통합됩니다. CloudTrail은 Amazon Kinesis 비디오 스트림에 대한 API 호출을 모두 이벤트로 캡처합니다. 캡처되는 호출에는 Amazon Kinesis 비디오 스트림 콘솔로부터의 호출과 Amazon Kinesis 비디오 스트림 API 작업에 대한 호출이 포함됩니다. 추적을 생성하면 Amazon Kinesis 비디오 스트림 이벤트를 비롯하여 CloudTrail 이벤트를 Amazon S3 버킷으로 지속적으로 전송할 수 있습니다. 추적을 구성하지 않은 경우 이벤트 기록에서 CloudTrail 콘솔의 최신 이벤트를 볼 수도 있습니다. CloudTrail에서 수집하는 정보를 사용하여 Amazon Kinesis 비디오 스트림에 어떤 요청이 이루어졌는지, 어떤 IP 주소에서 요청했는지, 누가 언제 요청했는지 및 추가 세부 정보를 확인할 수 있습니다.

그 구성 및 활성화 방법을 포함하여 CloudTrail에 대한 자세한 내용은 AWS CloudTrail User Guide를 참조하세요.

Kinesis 비디오 스트림 및 CloudTrail

CloudTrail은 계정 생성 시 AWS 계정에서 활성화됩니다. 지원되는 이벤트 활동이 Amazon Kinesis 비디오 스트림에서 발생하면, 해당 활동은 이벤트 기록의 다른 AWS 서비스 이벤트와 함께 CloudTrail 이벤트에 기록됩니다. 이를 통해 AWS 계정에서 최신 이벤트를 확인, 검색 및 다운로드할 수 있습니다. 자세한 내용은 CloudTrail 이벤트 기록을 사용하여 이벤트 보기를 참조하십시오.

Amazon Kinesis 비디오 스트림 이벤트를 비롯하여 AWS 계정의 이벤트 기록을 보유하려면 추적을 생성하십시오. 추적은 CloudTrail이 Amazon S3 버킷으로 로그 파일을 전송할 수 있게 해 줍니다. 콘솔에서 추적을 생성하면 기본적으로 모든 AWS 리전에 추적이 적용됩니다. 추적은 AWS 파티션에 있는 모든 리전의 이벤트를 로깅하고 지정된 Amazon S3 버킷으로 로그 파일을 전송합니다. 또는 기타 AWS 서비스를 사용하여 CloudTrail 로그에서 수집된 이벤트 데이터를 추가 분석 및 처리하도록 구성할 수 있습니다. 자세한 내용은 다음 자료를 참조하십시오.

- 추적 생성 개요
- CloudTrail 지원 서비스 및 통합
- CloudTrail에 대한 Amazon SNS 알림 구성
- 여러 리전에서 CloudTrail 로그 파일 받기 및 여러 계정에서 CloudTrail 로그 파일 받기

Amazon Kinesis 비디오 스트림 서비스는 CloudTrail 로그 파일의 이벤트로 다음 작업의 로깅을 지원합니다.

- CreateStream
- DeleteStream
- DescribeStream
- GetDataEndpoint
- ListStreams
- ListTagsForStream
- TagStream
- UntagStream
- UpdateDataRetention
- UpdateStream

모든 이벤트 및 로그 항목에는 요청을 생성한 사용자에 대한 정보가 들어 있습니다. 자격 증명 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 루트 또는 AWS Identity and Access Management(IAM) 사용자 자격 증명으로 사용하여 요청했는지 여부.
- 역할 또는 연합된 사용자에 대한 임시 보안 자격 증명을 사용하여 요청이 생성되었는지 여부.
- 요청을 다른 AWS 서비스로 했는지.

자세한 정보는 CloudTrail userIdentity 요소를 참조하십시오.

예: Amazon Kinesis 비디오 스트림 로그 파일 항목

추적은 지정한 Amazon S3 버킷에 이벤트를 로그 파일로 제공할 수 있도록 해주는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함됩니다. 이벤트는 어떤 소스로부터의 단일 요청을 나타내며 요청된 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보가 들어 있습니다. CloudTrail 로그 파일은 퍼블릭 API 호출의 주문 스택 트레이스가 아니므로 특정 순서로 표시되지 않습니다.

다음은 CreateStream 작업을 다루는 CloudTrail 로그 항목을 보여주는 예입니다.

```
{
    "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": "b7acfcd0-6ca9-4ee1-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 비디오 스트림 제한

Kinesis 비디오 스트림에는 다음과 같은 제한이 있습니다.

아래 제한은 지원 티켓을 제출하여 업그레이드할 수 있는 소프트[s] 또는 증가할 수 있는 하드[h]입니다.

제어 영역 API 제한

다음 단원에서는 제어 플레인 API에 대한 제한을 설명합니다.

계정 레벨 요청 제한에 도달하면 `ClientLimitExceededException`이 발생합니다.

계정 레벨 스트림 제한에 도달하거나 스트림 레벨 제한에 도달하면 `StreamLimitExceededException`이 발생합니다.

제어 영역 API 제한

API	계정 제한: 요청	계정 제한: 스트림	스트림 레벨 제한	관련 예외 및 메모
CreateStream	50TPS[s]	계정당 100개의 스트림[s]	5TPS[h]	디바이스, CLI, SDK 구동 액세스 및 콘솔은 모두 이 API를 호출할 수 있습니다. 스트림이 이미 존재하지 않는 경우 하나의 API 호출만 성공합니다.
DescribeStream	300TPS[h]	해당 사항 없음	5TPS[h]	
UpdateStream	50TPS[h]	해당 사항 없음	5TPS[h]	
ListStreams	300TPS[h]	해당 사항 없음	5TPS[h]	
DeleteStream	50TPS[h]	해당 사항 없음	5TPS[h]	
GetDataEndpoint	300TPS[h]	해당 사항 없음	5TPS[h]	대부분의 PutMedia/GetMedia 사용 사례에서 스트리밍 토큰을 새로 고치기 위해 45분마다 호출됩니다. ListFragments/GetMediaForFragmentList에 대해 조각 1,000개마다 호출됩니다. 애플리케이션이 실패 시 다시 로드하는 경우 데이터 엔드포인트를 캐시하는 것이 안전합니다.

미디어 및 보관된 미디어 API 한도

다음 단원에서는 미디어 및 보관된 미디어 API에 대한 제한을 설명합니다.

스트림 레벨 제한을 초과하면 `StreamLimitExceededException`이 발생합니다.

연결 레벨 제한에 도달하면 `ConnectionLimitExceededException`이 발생합니다.

조각 레벨 제한에 도달하면 다음 오류 또는 ack가 발생합니다.

- 최소 기간 미만의 조각에 대해서는 `MIN_FRAGMENT_DURATION_REACHED` ack가 반환됩니다.
- 최대 기간을 초과하는 조각에 대해서는 `MAX_FRAGMENT_DURATION_REACHED` ack가 반환됩니다.
- 최대 데이터 크기를 초과하는 조각에 대해서는 `MAX_FRAGMENT_SIZE` ack가 반환됩니다.
- `GetMediaForFragmentList` 작업에서 조각 제한에 도달하면 `FragmentLimitExceeded` 예외가 발생합니다.

데이터 영역 API 제한

API	스트림 레벨 제한	연결 레벨 제한	대역폭 제한	조각 레벨 제한	관련 예외 및 메모
PutMedia	5TPS[h]	1[s]	12.5MB/초 또는 100Mbps[s]	• 최소 조각 기간: 1초[h] • 최대 조각 기간: 10초[h] • 최대 조각 크기: 50MB[h]	일반적인 PutMedia 요청에는 여러 초 동안 데이터가 포함되므로 스트림당 TPS가 더 낮습니다. 제한을 초과하는 다중 동시 연결의 경우 마지막 연결이 수락됩니다.
GetHLSStreamingSessionURL	5TPS, 버스트, 1TPS 지속적[h]	해당 사항 없음	해당 사항 없음	해당 사항 없음	한 번에 스트림당 5개 세션만 활성화될 수 있습니다[s]. 제한에 도달하면 새 세션이 생성될 때 가장 오래된 세션이 취소됩니다.
GetMedia	5TPS[h]	3[s]	25MB/s 또는 200Mbps[s]	해당 사항 없음	어떤 시간에든 세 개의 클라이언트만 동시에 미디어 스트림에서 콘텐츠를 수신할 수 있습니다. 그 이상의 클라이언트 연결은 거부됩니다. 연결이 설정되면 애플리케이션이 연속으로 읽을 것으로 예상되므로, 고유의 소비 클라이언트에는 2 또는 3TPS만 필요합니다. 일반적인 조각이 약 5MB인 경우 이 제한은 Kinesis 비디오 스트림당 ~75MB/초를 의미합니다. 이러한 스트림의 송신 비트 속도는 최대 수신 비트 속도의 2배입니다.
ListFragments	5TPS[h]	해당 사항 없음	해당 사항 없음	해당 사항 없음	
GetMediaForFragmentList	5TPS[h]	5[s]	25MB/s 또는 200MbpsA[s]	최대 조각 수: 1000[h]	5개의 조각 기반 소비 애플리케이션이 동시에 미디어를 가져올 수 있습니다. 그 이상의 연결은 거부됩니다.

HLS API 제한

API	스트림 레벨 제한	대역폭 제한	조각 레벨 제한
GetHLSMasterPlaylist	5TPS[h]	해당 사항 없음	해당 사항 없음
GetHLSMediaPlaylist	5TPS[h]	해당 사항 없음	재생 목록당 최대 조각 수: 1000[h]
GetMP4InitFragment	5TPS[h]	해당 사항 없음	해당 사항 없음
GetMP4MediaFragment	10TPS[h]	해당 사항 없음	25MB/s 또는 200Mbps[s]

Kinesis 비디오 스트림 문제 해결

다음 정보를 사용하여 Amazon Kinesis 비디오 스트림에서 일반적으로 발생하는 문제를 해결하십시오.

항목

일반적인 문제 해결

이 단원에서는 Kinesis 비디오 스트림 작업 시 발생할 수 있는 일반적인 문제에 대해 설명합니다.

문제

너무 긴 지연 시간

Kinesis 비디오 스트림 서비스로 전송되는 조각의 지속 기간으로 인해 지연 시간이 발생할 수 있습니다. 생산자와 서비스 사이의 지연 시간을 줄이는 한 가지 방법은 조각 지속 시간을 단축하도록 미디어 파이프라인을 구성하는 것입니다.

조각 하나당 전송하는 프레임 수를 줄여 각 조각에 소요되는 시간을 줄이려면 `kinesis_video_gstreamer_sample_app.cpp`에서 다음 값을 줄입니다.

```
g_object_set(G_OBJECT (data.encoder), "bframes", 0, "key-int-max", 45, "bitrate", 512,
 NULL);
```

API 문제 해결

이 단원에서는 Kinesis 비디오 스트림 작업 시 발생할 수 있는 API 문제에 대해 설명합니다.

문제

오류: "Unable to determine service/operation name to be authorized"(권한을 부여할 서비스/작업 이름을 확인할 수 없음)

`GetMedia`가 다음 오류로 인해 실패할 수 있습니다.

```
Unable to determine service/operation name to be authorized
```

이 오류는 엔드포인트가 제대로 지정되지 않은 경우 발생할 수 있습니다. 엔드포인트를 가져올 때 호출할 API에 따라 `GetDataEndpoint` 호출에 다음 파라미터를 포함시켜야 합니다.

```
--api-name GET_MEDIA
--api-name PUT_MEDIA
--api-name GET_MEDIA_FOR_FRAGMENT_LIST
--api-name LIST_FRAGMENTS
```

오류: "Failed to put a frame in the stream"(스트림에 프레임을 넣지 못했음)

`PutMedia`가 다음 오류로 인해 실패할 수 있습니다.

```
Failed to put a frame in the stream
```

이 오류는 서비스에서 연결 또는 권한을 사용할 수 없는 경우 발생할 수 있습니다. AWS CLI에서 다음을 실행하고 스트림 정보를 검색할 수 있는지 확인합니다.

```
aws kinesisvideo describe-stream --stream-name StreamName --endpoint https://
ServiceEndpoint.kinesisvideo.region.amazonaws.com
```

호출에 실패할 경우 자세한 내용은 AWS CLI 오류 문제 해결을 참조하십시오.

오류: "Service closed connection before final AckEvent was received"(최종 AckEvent를 수신하기 전에 서비스가 연결을 종료함)

`PutMedia`가 다음 오류로 인해 실패할 수 있습니다.

```
com.amazonaws.SdkClientException: Service closed connection before final AckEvent was
 received
```

이 오류는 `PushbackInputStream`이 올바로 구현되지 않을 경우 발생할 수 있습니다. `unread()` 메서드가 올바르게 구현되는지 확인합니다.

오류: "STATUS_STORE_OUT_OF_MEMORY"

`PutMedia`가 다음 오류로 인해 실패할 수 있습니다.

```
The content store is out of memory.
```

이 오류는 콘텐츠 스토어가 충분한 크기로 할당되지 않을 때 나타납니다. 콘텐츠 스토어 크기를 늘리려면 `StorageInfo.storageSize` 값을 증가시킵니다. 자세한 내용은 StorageInfo (p. 71) 단원을 참조하십시오.

HLS 문제 해결

이 단원에서는 Kinesis 비디오 스트림에서 HLS를 사용할 때 발생할 수 있는 문제에 대해 설명합니다.

문제

HLS 스트리밍 세션 URL 검색이 성공하지만 비디오 플레이어에서 재생이 실패함

이 상황은 `GetHLSStreamingSessionURL`을 사용해 HLS 스트리밍 세션 URL이 성공적으로 검색되지만 비디오 플레이어에 이 URL이 제공하면 비디오가 재생되지 않는 경우입니다.

이 상황을 해결하려면 다음과 같이 시도해 보십시오.

- Kinesis 비디오 스트림 관리 콘솔에서 비디오 스트림이 재생되는지 확인합니다. 콘솔에 표시되는 모든 오류를 고려합니다.
- 조각 기간이 1초 미만이라면 1초로 늘립니다. 조각 기간이 너무 짧을 경우 서비스가 너무 빈번히 비디오 조각을 요청하므로 플레이어에서 스로틀링이 발생할 수 있습니다.
- 각 HLS 스트리밍 세션 URL을 한 플레이에서만 사용하는지 확인합니다. 2대 이상의 플레이어가 단일 HLS 스트리밍 세션 URL을 사용할 경우 서비스가 너무 많은 요청을 수신하여 스로틀링이 발생할 수 있습니다.
- 플레이어가 HLS 스트리밍 세션에 대해 지정한 모든 옵션을 지원하는지 확인합니다. 다음 파라미터 값을 다르게 조합하여 시도해 봅니다.
  - `PlaybackMode`
  - `FragmentSelectorType`
  - `DiscontinuityMode`
  - `MaxMediaPlaylistFragmentResults`

생산자와 플레이어 사이에 지연 시간이 너무 높음

이 상황은 비디오가 캡처되는 시점에서 비디오 플레이에서 재생되는 시점까지의 지연 시간이 너무 긴 경우입니다.

비디오는 HLS를 통해 조각 단위로 재생됩니다. 따라서 지연 시간이 조각 기간보다 작을 수 없습니다. 또한 지속 시간에는 데이터 버퍼링 및 전송에 필요한 시간도 포함됩니다. 솔루션에 1초 미만의 지연 시간이 필요할 경우 대신 `GetMedia` API를 사용할 것을 고려해 보십시오.

다음 파라미터를 조정하여 전체 지연 시간을 단축할 수 있습니다. 그러나 파라미터를 조정하면 비디오 품질이 저하되거나 재버퍼링 비율이 높아질 수 있습니다.

- 조각 기간: 조각 기간은 스트림이 분할된 비디오의 길이로서, 비디오 인코더가 생성하는 키 프레임의 빈도로 제어됩니다. 권장되는 값은 1초입니다. 조각 기간이 짧으면 서비스에 비디오 데이터를 전송하기 전에 조각이 완료될 때까지 대기해야 하는 시간이 작아집니다. 또한 조각이 짧을수록 서비스 처리 속도가 빨라

집니다. 그러나 조각 기간이 너무 짧을 경우 플레이어가 콘텐츠가 떨어져 재생을 멈추고 콘텐츠를 버퍼링할 가능성이 증가합니다. 조각 기간이 500밀리초 미만일 경우 생산자가 너무 많은 요청을 생성하여 서비스에서 스로틀링이 발생할 수 있습니다.

- 비트레이트: 비디오 스트림은 비트레이트가 낮을수록 읽고, 쓰고, 전송하는 데 걸리는 시간이 짧습니다. 그러나 일반적으로 비디오 스트림의 비트레이트가 낮으면 비디오 품질도 떨어집니다.
- 미디어 재생 목록 내 조각 수: 지연 시간에 민감한 플레이어는 미디어 재생 목록에서 최신 조각만 로드해야 합니다. 그러나 대부분의 플레이어는 가장 오래된 조각에서 시작합니다. 재생 목록의 조각 수를 줄이면 오래된 조각과 새로운 조각 사이의 시간 구분을 단축하는 것입니다. 재생 목록 크기가 작아지면 새 조각을 재생 목록에 추가할 때 지연이 있을 경우 또는 플레이어가 업데이트된 재생 목록 가져올 때 지연이 있을 경우 재생 도중 조각을 건너뛰게 될 수 있습니다. 3~5개 조각을 사용할 것을 권장하며, 재생 목록에서 최신 조각만 로드하도록 구성된 플레이어를 사용하는 것이 좋습니다.
- 플레이어 버퍼 크기: 대부분의 비디오 플레이어는 최소 버퍼 기간을 구성할 수 있으며, 일반적으로 기본값이 10초입니다. 지연 시간을 최소화하기 위해 이 값을 0초로 설정할 수 있지만, 그렇게 할 경우 조각을 생산하는 데 지연이 있을 경우 플레이어가 이 지연을 흡수할 버퍼가 없으므로 플레이어는 재버퍼링을 하게 됩니다.
- 플레이어 "캐치업": 일반적으로 비디오 플레이어는 예를 들어 지연된 조각이 재생할 조각의 백로그를 초래할 정도로 버퍼가 채워질 경우 자동으로 비디오 버퍼의 앞부분까지 재생을 캐치업하지 않습니다. 사용자 지정 플레이어는 프레임을 제거하거나 재생 속도를 높여(예: 1.1x) 버퍼의 앞부분까지 캐치업하여 이를 방지할 수 있습니다. 그러면 플레이어가 캐치업하는 과정에서 재생이 중간에 끊기거나 속도가 빨라질 수 있고, 버퍼 크기가 짧게 유지되면 재버퍼링이 더 빈번해질 수 있습니다.

Java 문제 해결

이 단원에서는 Kinesis 비디오 스트림 작업 시 발생할 수 있는 일반적인 Java 문제를 해결하는 방법을 설명합니다.

문제

Java 로그 활성화

Java 샘플 및 라이브러리와 관련된 문제를 해결하는 데는 디버그 로그를 활성화하고 검사하는 것이 도움이 됩니다. 디버그 로그를 활성화하려면 다음을 수행합니다.

1. `log4j`을 `pom.xml` 노드에 있는 `dependencies` 파일에 추가합니다.

```
<dependency>
    <groupId>log4j</groupId>
    <artifactId>log4j</artifactId>
    <version>1.2.17</version>
</dependency>
```

2. `target/classes` 디렉터리에 다음 콘텐츠로 `log4j.properties`라는 파일을 생성합니다.

```
# 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
```

그러면 디버그 로그가 IDE 콘솔로 인쇄됩니다.

생산자 라이브러리 문제 해결

이 단원에서는 생산자 라이브러리 (p. 27) 사용 시 발생할 수 있는 문제에 대해 설명합니다.

문제

- 생산자 SDK를 컴파일할 수 없음 (p. 123)
- 비디오 스트림이 콘솔에 나타나지 않음 (p. 123)
- 오류: "GStreamer 데모 애플리케이션을 사용하여 데이터를 스트리밍할 때 "요청에 포함된 보안 토큰이 잘못되었음" (p. 124)
- 오류: "Kinesis 비디오 클라이언트에 프레임을 제출하지 못 함" (p. 124)
- OS X에 "스트리밍 중지됨, 이유가 협상되지 않음" 메시지와 함께 GStreamer 애플리케이션이 중지됨 (p. 124)
- 오류: Raspberry Pi 기반으로 GStreamer 데모에서 Kinesis 비디오 클라이언트를 만들 때 "힙 할당에 실패했음" (p. 125)
- 오류: Raspberry Pi 기반으로 GStreamer 데모를 실행하는 동안 "잘못된 명령" (p. 125)
- 카메라가 Raspberry Pi에서 로드하지 못함 (p. 125)
- macOS High Sierra에서 카메라를 찾을 수 없음 (p. 126)
- macOS High Sierra에서 컴파일 시 jni.h 파일을 찾을 수 없음 (p. 126)
- GStreamer 데모 앱 실행 시 Curl 오류 발생 (p. 126)
- Raspberry Pi에서 런타임 시 타임스탬프/범위 어설션 (p. 126)
- Raspberry Pi에서 gst_value_set_fraction_range_full에 어설션 (p. 126)

생산자 SDK를 컴파일할 수 없음

필요한 라이브러리가 경로에 있는지 확인합니다. 이를 확인하려면 다음 명령을 사용합니다.

```
$ 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
```

비디오 스트림이 콘솔에 나타나지 않음

콘솔에 비디오 스트림을 표시하려면 AvCC 형식으로 H.264를 사용하여 인코딩해야 합니다. 스트림이 보이지 않으면 다음을 확인합니다.

- 기존 스트림이 Annex-B 형식에 있으면 NAL 적응 플래그 (p. 70)는 `NAL_ADAPTATION_ANNEXB_NALS | NAL_ADAPTATION_ANNEXB_CPD_NALS`로 설정됩니다. 이는 `StreamDefinition` 생성자의 기본값입니다.
- 코덱 프라이빗 데이터를 올바르게 제공하고 있습니다. H.264의 경우 SPS(Sequence Parameter Set) 및 PPS(Picture Parameter Set)입니다. 미디어 원본에 따라 이 데이터를 별도로 미디어 원본에서 검색하거나 프레임에 인코딩할 수 있습니다.

  초기 스트림은 다음 형식이며 `Ab`가 Annex-B 시작 코드(001 혹은 0001)입니다.

```
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)
```

H.264의 경우, 코덱 프라이빗 데이터(CPD)는 SPS와 PPS와 같은 스트림에 있고 AvCC 형식을 적용할 수 있습니다. 미디어 파이프라인이 CPD를 개별적으로 주지 않는 한, 애플리케이션은 (SPS/PPS를 포함해야 하는) 첫 번째 LDR 프레임을 검색하여 프레임으로부터 CPD를 추출하거나(Ab(Sps)Ab(Pps)인) 두 개의 NALU를 추출하고 `StreamDefinition`의 CPD에 설정합니다.

오류: "GStreamer 데모 애플리케이션을 사용하여 데이터를 스트리밍할 때 "요청에 포함된 보안 토큰이 잘못되었음"

이 오류가 발생하면 자격 증명에 문제가 있는 것입니다. 다음을 확인합니다.

- 임시 자격 증명을 사용하는 경우 세션 토큰을 지정해야 합니다.
- 임시 자격 증명이 만료되지 않았는지 확인합니다.
- 적절한 권한이 설정되어 있는지 확인합니다.
- macOS에서 키체인에 자격 증명이 캐시되어 있지 않은지 확인합니다.

오류: "Kinesis 비디오 클라이언트에 프레임을 제출하지 못 함"

이 오류가 발생하면 타임스탬프가 소스 스트림에 제대로 설정되지 않은 것입니다. 다음을 수행해 보십시오.

- 최신 SDK 샘플을 사용합니다. 문제를 해결하는 업데이트가 있을 수 있습니다.
- 고품질 스트림을 더 높은 비트 전송률로 설정하고 카메라가 해당 작업을 지원하는 경우 소스 스트림의 모든 지터를 수정합니다.

OS X에 "스트리밍 중지됨, 이유가 협상되지 않음" 메시지와 함께 GStreamer 애플리케이션이 중지됨

다음 메시지와 함께 OS X에서 스트리밍이 중단될 수 있음:

```
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)
```

이 문제에 대한 가능한 해결 방법은 `kinesis_video_gstreamer_sample_app.cpp`에서 `gst_caps_new_simple` 호출에서 프레임 속도 파라미터를 제거하는 것입니다.

```
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);
```

오류: Raspberry Pi 기반으로 GStreamer 데모에서 Kinesis 비디오 클라이언트를 만들 때 "힙 할당에 실패했음"

GStreamer 샘플 애플리케이션이 시스템에서 사용할 수 없는 512MB RAM을 할당하려고 합니다. `KinesisVideoProducer.cpp`에서 다음 값을 줄임으로써 이 할당을 줄일 수 있습니다.

```
device_info.storageInfo.storageSize = 512 * 1024 * 1024;
```

오류: Raspberry Pi 기반으로 GStreamer 데모를 실행하는 동안 "잘못된 명령"

GStreamer 데모를 실행할 때 다음 오류가 발생하면 올바른 버전의 디바이스용 애플리케이션을 컴파일했는지 확인하십시오. (예를 들어, Raspberry Pi 2를 실행할 때 Raspberry Pi 3용으로 컴파일하지 않도록 합니다.)

```
INFO - Initializing curl.
Illegal instruction
```

카메라가 Raspberry Pi에서 로드하지 못함

카메라가 로드되었는지 확인하려면 다음을 실행합니다.

```
$ ls /dev/video*
```

아무 것도 없으면 다음을 실행합니다.

```
$ vcgencmd get_camera
```

다음과 같이 출력됩니다

```
supported=1 detected=1
```

드라이버가 카메라를 감지하지 못하면 다음을 수행합니다.

1. 실제 카메라 설정을 확인하고 올바르게 연결되었는지 확인하십시오.
2. 다음을 실행하여 펌웨어를 업그레이드합니다.

   ```
   $ sudo rpi-update
   ```
3. 디바이스를 다시 시작합니다.
4. 다음을 실행하여 드라이버를 로드합니다.

   ```
   $ sudo modprobe bcm2835-v4l2
   ```
5. 카메라가 감지되었는지 확인합니다.

   ```
   $ ls /dev/video*
   ```

macOS High Sierra에서 카메라를 찾을 수 없음

macOS High Sierra에서 두 대 이상의 카메라를 사용할 수 있는 경우 데모 애플리케이션이 카메라를 찾을 수 없습니다.

macOS High Sierra에서 컴파일 시 jni.h 파일을 찾을 수 없음

이 오류를 해결하려면 설치한 Xcode를 최신 버전으로 업데이트합니다.

GStreamer 데모 앱 실행 시 Curl 오류 발생

GStreamer 데모 애플리케이션 실행 시 발생한 curl 오류를 실행하려면 이 인증서 파일을 `/etc/ssl/cert.pem`으로 복사합니다.

Raspberry Pi에서 런타임 시 타임스탬프/범위 어설션

런타임 시 타임스탬프 범위 어설션이 발생하면 펌웨어를 업데이트하고 디바이스를 다시 시작합니다.

```
$ sudo rpi-update
$ sudo reboot
```

Raspberry Pi에서 gst_value_set_fraction_range_full에 어설션

`uv4l` 서비스가 실행 중인 경우 다음 어설션이 나타납니다.

```
gst_util_fraction_compare (numerator_start, denominator_start, numerator_end,
 denominator_end) < 0' failed
```

이 경우 `uv4l` 서비스를 중지하고 애플리케이션을 다시 시작합니다.

스트림 구문 분석기 라이브러리 문제 해결

이 단원에서는 스트림 구문 분석기 라이브러리 (p. 86) 사용 시 발생할 수 있는 문제에 대해 설명합니다.

문제

- 스트림에서 단일 프레임에 액세스할 수 없음 (p. 126)
- 조각 디코딩 오류 (p. 127)

스트림에서 단일 프레임에 액세스할 수 없음

소비자 애플리케이션의 스트리밍 소스에서 단일 프레임에 액세스하려면 스트림에 올바른 코덱 프라이빗 데이터가 포함되어 있는지 확인합니다. 스트림의 데이터 형식에 대한 자세한 내용은 데이터 모델 (p. 18) 단원을 참조하십시오.

코덱 프라이빗 데이터를 사용하여 프레임에 액세스하는 방법을 알아보려면 GitHub 웹사이트에서 다음 테스트 파일을 참조하십시오. KinesisVideoRendererExampleTest.java

조각 디코딩 오류

조각이 브라우저에서 지원하는 H.264 형식 및 레벨로 올바르게 인코딩되지 않은 경우 콘솔에서 스트림을 재생할 때 다음 오류가 표시될 수 있습니다.

```
Fragment Decoding Error
There was an error decoding the video data. Verify that the stream contains valid H.264
 content
```

이 경우 다음을 확인합니다.

- 프레임의 해상도는 코덱 프라이빗 데이터에 지정된 해상도와 일치합니다.
- 인코딩 프레임의 H.264 프로필 및 레벨은 코덱 프라이빗 데이터에 지정된 프로필 및 레벨과 일치합니다.
- 브라우저는 프로필/레벨 조합을 지원합니다. 최신 브라우저는 모든 프로필 및 레벨 조합을 지원합니다.
- 타임스탬프는 정확하고 순서가 올바르며 중복 타임스탬프가 생성되지 않습니다.
- 애플리케이션이 H.264 형식을 사용하여 프레임 데이터를 인코딩하고 있습니다.

Amazon Kinesis Video Streams 문서 기록

다음 표에서는 Amazon Kinesis 비디오 스트림의 마지막 릴리스 이후 이 설명서에서 변경된 중요 사항에 대해 설명합니다.

- 최신 API 버전: 2017-11-29
- 최종 설명서 업데이트: 2018년 18월 7일

변경 사항	설명	날짜
C++ 생산자 SDK 로깅	C++ 생산자 SDK 애플리케이션에 대한 로깅을 구성하는 방법입니다. 자세한 내용은 C++ 생산자 SDK 로깅 사용 (p. 49) 단원을 참조하십시오.	2018년 7월 18일
HLS 비디오 스트리밍	이제 HTTP 라이브 스트리밍을 사용하여 Kinesis 비디오 스트림을 볼 수 있습니다. 자세한 내용은 HLS를 사용한 Kinesis 비디오 스트림 재생 (p. 9) 단원을 참조하십시오.	2018년 7월 13일
RTSP 소스에서 스트리밍	도커 컨테이너에서 실행되고 RTSP 소스의 비디오를 스트리밍하는 Kinesis 비디오 스트림용 샘플 애플리케이션입니다. 자세한 내용은 RTSP 및 도커 (p. 104) 단원을 참조하십시오.	2018년 20월 6일
C++ 생산자 SDK GStreamer 플러그인	C++ 생산자 라이브러리 (p. 36)를 구축하여 GStreamer 대상으로 사용하는 방법을 보여 줍니다. 자세한 내용은 GStreamer 플러그인 (p. 93) 단원을 참조하십시오.	2018년 15월 6일
생산자 SDK 콜백 참조 설명서	Kinesis 비디오 스트림 생산자 라이브러리 (p. 27)에 사용되는 콜백에 대한 참조 문서입니다. 자세한 내용은 생산자 SDK 콜백 (p. 81) 단원을 참조하십시오.	2018년 6월 12일
시스템 요구 사항	생산자 디바이스와 SDK의 메모리 및 스토리지 요구 사항 설명서. 자세한 내용은 Kinesis 비디오 스트림 시스템 요구 사항 (p. 3) 단원을 참조하십시오.	2018년 5월 30일
CloudTrail 지원	API 사용량을 모니터링하기 위한 CloudTrail 사용 설명서. 자세한 내용은 AWS CloudTrail을 사용하여	2018년 5월 24일

변경 사항	설명	날짜
	Kinesis 비디오 스트림 API 호출 로깅 (p. 112) 단원을 참조하십시오.	
생산자 SDK 구조 참조 설명서	Kinesis 비디오 스트림 생산자 라이브러리 (p. 27)에 사용된 구조 참조 문서 자세한 내용은 프로듀서 SDK 구조 (p. 70) 및 Kinesis 비디오 스트림 구조 (p. 72) 단원을 참조하십시오.	2018년 5월 7일
렌더러 예제 설명서	Kinesis 비디오 스트림에서 프레임을 디코딩 및 표시하는 방법을 보여주는 Renderer 예제 애플리케이션에 대한 설명서입니다. 자세한 내용은 예: Kinesis 비디오 스트림 조각 구문 분석 및 렌더링 (p. 106) 단원을 참조하십시오.	2018년 3월 15일
생산자 SDK 제한 사항 참조 설명서	Information about limits for operations in the C++ 생산자 라이브러리 (p. 36)에는 작업에 대한 제한 사항 정보가 수록되어 있습니다. 자세한 내용은 생산자 SDK 제한 사항 (p. 50) 단원을 참조하십시오.	2018년 3월 13일
Raspberry Pi용 C++ 생산자 SDK	Raspberry Pi 디바이스에서 C++ 생산자 라이브러리 (p. 36) 설치 및 실행에 대한 절차입니다. 자세한 내용은 Raspberry Pi에서 C++ 생산자 SDK 사용 (p. 44) 단원을 참조하십시오.	2018년 3월 13일
모니터링	Amazon CloudWatch 및 AWS CloudTrail을 사용한 Kinesis 비디오 스트림 지표 및 API 호출 모니터링에 대한 정보입니다. 자세한 내용은 Kinesis 비디오 스트림 모니터링 (p. 109) 단원을 참조하십시오.	2018년 2월 5일
NAL(Network Abstraction Layer) 적응 플래그 참조	스트리밍 비디오 이용 시 NAL 적응 플래그 설정에 대한 정보입니다. 자세한 내용은 NAL 적응 플래그 (p. 70) 단원을 참조하십시오.	2018년 1월 15일
스트리밍 비디오에 대한 Android 지원	Kinesis 비디오 스트림는 이제 Android 디바이스에서의 스트리밍 비디오를 지원합니다. 자세한 내용은 Android 생산자 라이브러리 (p. 32) 단원을 참조하십시오.	2018년 1월 12일

변경 사항	설명	날짜
Kinesis Video 예제 설명서	애플리케이션에서 Kinesis Video Stream Parser Library (p. 86) 사용 방법을 보여주는 Kinesis Video 예제 애플리케이션 설명서입니다. 자세한 내용은 KinesisVideoExample (p. 89) 단원을 참조하십시오.	2018년 1월 9일
GStreamer 예제 설명서	C++ 생산자 SDK에 포함된 GStreamer 예제 애플리케이션 설명서입니다. 자세한 내용은 예: Kinesis 비디오 스트림와 함께 GStreamer 사용 (p. 105) 단원을 참조하십시오.	2018년 1월 5일
Kinesis 비디오 스트림 설명서 릴리스	이것은 Amazon Kinesis Video Streams 개발자 안내서의 최초 릴리스입니다.	2017년 11월 29일

API Reference

이 단원은 API 참조 문서를 포함합니다.

Actions

The following actions are supported by Amazon Kinesis Video Streams:

- CreateStream (p. 133)
- DeleteStream (p. 137)
- DescribeStream (p. 139)
- GetDataEndpoint (p. 142)
- ListStreams (p. 145)
- ListTagsForStream (p. 148)
- TagStream (p. 151)
- UntagStream (p. 154)
- UpdateDataRetention (p. 156)
- UpdateStream (p. 159)

The following actions are supported by Amazon Kinesis Video Streams Media:

- GetMedia (p. 162)
- PutMedia (p. 166)

The following actions are supported by Amazon Kinesis Video Streams Archived Media:

- GetHLSStreamingSessionURL (p. 172)
- GetMediaForFragmentList (p. 179)
- ListFragments (p. 182)

Amazon Kinesis Video Streams

The following actions are supported by Amazon Kinesis Video Streams:

- CreateStream (p. 133)
- DeleteStream (p. 137)
- DescribeStream (p. 139)
- GetDataEndpoint (p. 142)
- ListStreams (p. 145)
- ListTagsForStream (p. 148)
- TagStream (p. 151)
- UntagStream (p. 154)
- UpdateDataRetention (p. 156)
- UpdateStream (p. 159)

CreateStream

Service: Amazon Kinesis Video Streams

Creates a new Kinesis video stream.

When you create a new stream, Kinesis Video Streams assigns it a version number. When you change the stream's metadata, Kinesis Video Streams updates the version.

`CreateStream` is an asynchronous operation.

For information about how the service works, see How it Works.

You must have permissions for the `KinesisVideo:CreateStream` action.

Request Syntax

```
POST /createStream HTTP/1.1
Content-type: application/json

{
   "DataRetentionInHours": number,
   "DeviceName": "string",
   "KmsKeyId": "string",
   "MediaType": "string",
   "StreamName": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

DataRetentionInHours (p. 133)

The number of hours that you want to retain the data in the stream. Kinesis Video Streams retains the data in a data store that is associated with the stream.

The default value is 0, indicating that the stream does not persist data.

When the `DataRetentionInHours` value is 0, consumers can still consume the fragments that remain in the service host buffer, which has a retention time limit of 5 minutes and a retention memory limit of 200 MB. Fragments are removed from the buffer when either limit is reached.

Type: Integer

Valid Range: Minimum value of 0.

Required: No

DeviceName (p. 133)

The name of the device that is writing to the stream.

Note

In the current implementation, Kinesis Video Streams does not use this name.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

KmsKeyId (p. 133)

The ID of the AWS Key Management Service (AWS KMS) key that you want Kinesis Video Streams to use to encrypt stream data.

If no key ID is specified, the default, Kinesis Video-managed key (`aws/kinesisvideo`) is used.

For more information, see DescribeKey.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2048.

Required: No

MediaType (p. 133)

The media type of the stream. Consumers of the stream can use this information when processing the stream. For more information about media types, see Media Types. If you choose to specify the `MediaType`, see Naming Requirements for guidelines.

To play video on the console, the media must be H.264 encoded, and you need to specify this video type in this parameter as `video/h264`.

This parameter is optional; the default value is `null` (or empty in JSON).

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `[\w\-\.\+]+/[\w\-\.\+]+`

Required: No

StreamName (p. 133)

A name for the stream that you are creating.

The stream name is an identifier for the stream, and must be unique for each account and region.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: Yes

Response Syntax

```
HTTP/1.1 200
Content-type: application/json

{
   "StreamARN": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

StreamARN (p. 134)

The Amazon Resource Name (ARN) of the stream.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

AccountStreamLimitExceededException

The number of streams created for the account is too high.

HTTP Status Code: 400

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

DeviceStreamLimitExceededException

Not implemented.

HTTP Status Code: 400

InvalidArgumentException

The value for this input parameter is invalid.

HTTP Status Code: 400

InvalidDeviceException

Not implemented.

HTTP Status Code: 400

ResourceInUseException

The stream is currently not available for this operation.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DeleteStream

Service: Amazon Kinesis Video Streams

Deletes a Kinesis video stream and the data contained in the stream.

This method marks the stream for deletion, and makes the data in the stream inaccessible immediately.

To ensure that you have the latest version of the stream before deleting it, you can specify the stream version. Kinesis Video Streams assigns a version to each stream. When you update a stream, Kinesis Video Streams assigns a new version number. To get the latest stream version, use the `DescribeStream` API.

This operation requires permission for the `KinesisVideo:DeleteStream` action.

Request Syntax

```
POST /deleteStream HTTP/1.1
Content-type: application/json

{
   "CurrentVersion": "string",
   "StreamARN": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

CurrentVersion (p. 137)

Optional: The version of the stream that you want to delete.

Specify the version as a safeguard to ensure that your are deleting the correct stream. To get the stream version, use the `DescribeStream` API.

If not specified, only the `CreationTime` is checked before deleting the stream.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 64.

Pattern: `[a-zA-Z0-9]+`

Required: No

StreamARN (p. 137)

The Amazon Resource Name (ARN) of the stream that you want to delete.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

Required: Yes

Response Syntax

```
HTTP/1.1 200
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

InvalidArgumentException

The value for this input parameter is invalid.

HTTP Status Code: 400

NotAuthorizedException

The caller is not authorized to perform this operation.

HTTP Status Code: 401

ResourceNotFoundException

Amazon Kinesis Video Streams can't find the stream that you specified.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeStream

Service: Amazon Kinesis Video Streams

Returns the most current information about the specified stream. You must specify either the `StreamName` or the `StreamARN`.

Request Syntax

```
POST /describeStream HTTP/1.1
Content-type: application/json

{
   "StreamARN": "string",
   "StreamName": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

StreamARN (p. 139)

The Amazon Resource Name (ARN) of the stream.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

Required: No

StreamName (p. 139)

The name of the stream.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

Response Syntax

```
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"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

StreamInfo (p. 139)

An object that describes the stream.

Type: StreamInfo (p. 186) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

InvalidArgumentException

The value for this input parameter is invalid.

HTTP Status Code: 400

NotAuthorizedException

The caller is not authorized to perform this operation.

HTTP Status Code: 401

ResourceNotFoundException

Amazon Kinesis Video Streams can't find the stream that you specified.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript

- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

GetDataEndpoint

Service: Amazon Kinesis Video Streams

Gets an endpoint for a specified stream for either reading or writing. Use this endpoint in your application to read from the specified stream (using the `GetMedia` or `GetMediaForFragmentList` operations) or write to it (using the `PutMedia` operation).

> Note
>
> The returned endpoint does not have the API name appended. The client needs to add the API name to the returned endpoint.

In the request, specify the stream either by `StreamName` or `StreamARN`.

Request Syntax

```
POST /getDataEndpoint HTTP/1.1
Content-type: application/json

{
   "APIName": "string",
   "StreamARN": "string",
   "StreamName": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

APIName (p. 142)

The name of the API action for which to get an endpoint.

Type: String

Valid Values: `PUT_MEDIA | GET_MEDIA | LIST_FRAGMENTS | GET_MEDIA_FOR_FRAGMENT_LIST`

Required: Yes

StreamARN (p. 142)

The Amazon Resource Name (ARN) of the stream that you want to get the endpoint for. You must specify either this parameter or a `StreamName` in the request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

Required: No

StreamName (p. 142)

The name of the stream that you want to get the endpoint for. You must specify either this parameter or a `StreamARN` in the request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

Response Syntax

```
HTTP/1.1 200
Content-type: application/json

{
   "DataEndpoint": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DataEndpoint (p. 143)

The endpoint value. To read data from the stream or to write data to it, specify this endpoint in your application.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

InvalidArgumentException

The value for this input parameter is invalid.

HTTP Status Code: 400

NotAuthorizedException

The caller is not authorized to perform this operation.

HTTP Status Code: 401

ResourceNotFoundException

Amazon Kinesis Video Streams can't find the stream that you specified.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

ListStreams

Service: Amazon Kinesis Video Streams

Returns an array of `StreamInfo` objects. Each object describes a stream. To retrieve only streams that satisfy a specific condition, you can specify a `StreamNameCondition`.

Request Syntax

```
POST /listStreams HTTP/1.1
Content-type: application/json

{
   "MaxResults": number,
   "NextToken": "string",
   "StreamNameCondition": {
      "ComparisonOperator": "string",
      "ComparisonValue": "string"
   }
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

MaxResults (p. 145)

The maximum number of streams to return in the response. The default is 10,000.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 10000.

Required: No

NextToken (p. 145)

If you specify this parameter, when the result of a `ListStreams` operation is truncated, the call returns the `NextToken` in the response. To get another batch of streams, provide this token in your next request.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 512.

Required: No

StreamNameCondition (p. 145)

Optional: Returns only streams that satisfy a specific condition. Currently, you can specify only the prefix of a stream name as a condition.

Type: StreamNameCondition (p. 188) object

Required: No

Response Syntax

```
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"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken (p. 145)

If the response is truncated, the call returns this element with a token. To get the next batch of streams, use this token in your next request.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 512.

StreamInfoList (p. 145)

An array of `StreamInfo` objects.

Type: Array of StreamInfo (p. 186) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

InvalidArgumentException

The value for this input parameter is invalid.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface

- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

ListTagsForStream

Service: Amazon Kinesis Video Streams

Returns a list of tags associated with the specified stream.

In the request, you must specify either the `StreamName` or the `StreamARN`.

Request Syntax

```
POST /listTagsForStream HTTP/1.1
Content-type: application/json

{
   "NextToken": "string",
   "StreamARN": "string",
   "StreamName": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

NextToken (p. 148)

If you specify this parameter and the result of a `ListTagsForStream` call is truncated, the response includes a token that you can use in the next request to fetch the next batch of tags.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 512.

Required: No

StreamARN (p. 148)

The Amazon Resource Name (ARN) of the stream that you want to list tags for.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

Required: No

StreamName (p. 148)

The name of the stream that you want to list tags for.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

Response Syntax

```
HTTP/1.1 200
Content-type: application/json

{
   "NextToken": "string",
   "Tags": {
      "string" : "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken (p. 149)

If you specify this parameter and the result of a `ListTags` call is truncated, the response includes a token that you can use in the next request to fetch the next set of tags.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 512.

Tags (p. 149)

A map of tag keys and values associated with the specified stream.

Type: String to string map

Key Length Constraints: Minimum length of 1. Maximum length of 128.

Value Length Constraints: Minimum length of 0. Maximum length of 256.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

InvalidArgumentException

The value for this input parameter is invalid.

HTTP Status Code: 400

InvalidResourceFormatException

The format of the `StreamARN` is invalid.

HTTP Status Code: 400

NotAuthorizedException

The caller is not authorized to perform this operation.

HTTP Status Code: 401

ResourceNotFoundException

Amazon Kinesis Video Streams can't find the stream that you specified.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

TagStream

Service: Amazon Kinesis Video Streams

Adds one or more tags to a stream. A tag is a key-value pair (the value is optional) that you can define and assign to AWS resources. If you specify a tag that already exists, the tag value is replaced with the value that you specify in the request. For more information, see Using Cost Allocation Tags in the AWS Billing and Cost Management User Guide.

You must provide either the `StreamName` or the `StreamARN`.

This operation requires permission for the `KinesisVideo:TagStream` action.

Kinesis video streams support up to 50 tags.

Request Syntax

```
POST /tagStream HTTP/1.1
Content-type: application/json

{
   "StreamARN": "string",
   "StreamName": "string",
   "Tags": {
      "string" : "string"
   }
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

StreamARN (p. 151)

The Amazon Resource Name (ARN) of the resource that you want to add the tag or tags to.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

Required: No

StreamName (p. 151)

The name of the stream that you want to add the tag or tags to.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

Tags (p. 151)

A list of tags to associate with the specified stream. Each tag is a key-value pair (the value is optional).

Type: String to string map

Key Length Constraints: Minimum length of 1. Maximum length of 128.

Value Length Constraints: Minimum length of 0. Maximum length of 256.

Required: Yes

Response Syntax

```
HTTP/1.1 200
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

InvalidArgumentException

The value for this input parameter is invalid.

HTTP Status Code: 400

InvalidResourceFormatException

The format of the `StreamARN` is invalid.

HTTP Status Code: 400

NotAuthorizedException

The caller is not authorized to perform this operation.

HTTP Status Code: 401

ResourceNotFoundException

Amazon Kinesis Video Streams can't find the stream that you specified.

HTTP Status Code: 404

TagsPerResourceExceededLimitException

You have exceeded the limit of tags that you can associate with the resource. Kinesis video streams support up to 50 tags.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

UntagStream

Service: Amazon Kinesis Video Streams

Removes one or more tags from a stream. In the request, specify only a tag key or keys; don't specify the value. If you specify a tag key that does not exist, it's ignored.

In the request, you must provide the `StreamName` or `StreamARN`.

Request Syntax

```
POST /untagStream HTTP/1.1
Content-type: application/json

{
   "StreamARN": "string",
   "StreamName": "string",
   "TagKeyList": [ "string" ]
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

StreamARN (p. 154)

The Amazon Resource Name (ARN) of the stream that you want to remove tags from.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

Required: No

StreamName (p. 154)

The name of the stream that you want to remove tags from.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

TagKeyList (p. 154)

A list of the keys of the tags that you want to remove.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Length Constraints: Minimum length of 1. Maximum length of 128.

Required: Yes

Response Syntax

```
HTTP/1.1 200
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

InvalidArgumentException

The value for this input parameter is invalid.

HTTP Status Code: 400

InvalidResourceFormatException

The format of the `StreamARN` is invalid.

HTTP Status Code: 400

NotAuthorizedException

The caller is not authorized to perform this operation.

HTTP Status Code: 401

ResourceNotFoundException

Amazon Kinesis Video Streams can't find the stream that you specified.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

UpdateDataRetention

Service: Amazon Kinesis Video Streams

Increases or decreases the stream's data retention period by the value that you specify. To indicate whether you want to increase or decrease the data retention period, specify the `Operation` parameter in the request body. In the request, you must specify either the `StreamName` or the `StreamARN`.

> Note
>
> The retention period that you specify replaces the current value.

This operation requires permission for the `KinesisVideo:UpdateDataRetention` action.

Changing the data retention period affects the data in the stream as follows:

- If the data retention period is increased, existing data is retained for the new retention period. For example, if the data retention period is increased from one hour to seven hours, all existing data is retained for seven hours.
- If the data retention period is decreased, existing data is retained for the new retention period. For example, if the data retention period is decreased from seven hours to one hour, all existing data is retained for one hour, and any data older than one hour is deleted immediately.

Request Syntax

```
POST /updateDataRetention HTTP/1.1
Content-type: application/json

{
   "CurrentVersion": "string",
   "DataRetentionChangeInHours": number,
   "Operation": "string",
   "StreamARN": "string",
   "StreamName": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

CurrentVersion (p. 156)

The version of the stream whose retention period you want to change. To get the version, call either the `DescribeStream` or the `ListStreams` API.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 64.

Pattern: `[a-zA-Z0-9]+`

Required: Yes

DataRetentionChangeInHours (p. 156)

The retention period, in hours. The value you specify replaces the current value.

Type: Integer

Valid Range: Minimum value of 1.

Required: Yes

Operation (p. 156)

Indicates whether you want to increase or decrease the retention period.

Type: String

Valid Values: `INCREASE_DATA_RETENTION | DECREASE_DATA_RETENTION`

Required: Yes

StreamARN (p. 156)

The Amazon Resource Name (ARN) of the stream whose retention period you want to change.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

Required: No

StreamName (p. 156)

The name of the stream whose retention period you want to change.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

Response Syntax

```
HTTP/1.1 200
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

InvalidArgumentException

The value for this input parameter is invalid.

HTTP Status Code: 400

NotAuthorizedException

The caller is not authorized to perform this operation.

HTTP Status Code: 401

ResourceInUseException

The stream is currently not available for this operation.

HTTP Status Code: 400

ResourceNotFoundException

Amazon Kinesis Video Streams can't find the stream that you specified.

HTTP Status Code: 404

VersionMismatchException

The stream version that you specified is not the latest version. To get the latest version, use the DescribeStream API.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

UpdateStream

Service: Amazon Kinesis Video Streams

Updates stream metadata, such as the device name and media type.

You must provide the stream name or the Amazon Resource Name (ARN) of the stream.

To make sure that you have the latest version of the stream before updating it, you can specify the stream version. Kinesis Video Streams assigns a version to each stream. When you update a stream, Kinesis Video Streams assigns a new version number. To get the latest stream version, use the `DescribeStream` API.

`UpdateStream` is an asynchronous operation, and takes time to complete.

Request Syntax

```
POST /updateStream HTTP/1.1
Content-type: application/json

{
   "CurrentVersion": "string",
   "DeviceName": "string",
   "MediaType": "string",
   "StreamARN": "string",
   "StreamName": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

CurrentVersion (p. 159)

The version of the stream whose metadata you want to update.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 64.

Pattern: `[a-zA-Z0-9]+`

Required: Yes

DeviceName (p. 159)

The name of the device that is writing to the stream.

> Note
>
> In the current implementation, Kinesis Video Streams does not use this name.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

MediaType (p. 159)

The stream's media type. Use `MediaType` to specify the type of content that the stream contains to the consumers of the stream. For more information about media types, see Media Types. If you choose to specify the `MediaType`, see Naming Requirements.

To play video on the console, you must specify the correct video type. For example, if the video in the stream is H.264, specify `video/h264` as the `MediaType`.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `[\w\-\.\+]+/[\w\-\.\+]+`

Required: No

StreamARN (p. 159)

The ARN of the stream whose metadata you want to update.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

Required: No

StreamName (p. 159)

The name of the stream whose metadata you want to update.

The stream name is an identifier for the stream, and must be unique for each account and region.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

Response Syntax

```
HTTP/1.1 200
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

InvalidArgumentException

The value for this input parameter is invalid.

HTTP Status Code: 400

NotAuthorizedException

The caller is not authorized to perform this operation.

HTTP Status Code: 401

ResourceInUseException

The stream is currently not available for this operation.

HTTP Status Code: 400

ResourceNotFoundException

Amazon Kinesis Video Streams can't find the stream that you specified.

HTTP Status Code: 404

VersionMismatchException

The stream version that you specified is not the latest version. To get the latest version, use the DescribeStream API.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

Amazon Kinesis Video Streams Media

The following actions are supported by Amazon Kinesis Video Streams Media:

- GetMedia (p. 162)
- PutMedia (p. 166)

GetMedia

Service: Amazon Kinesis Video Streams Media

Use this API to retrieve media content from a Kinesis video stream. In the request, you identify the stream name or stream Amazon Resource Name (ARN), and the starting chunk. Kinesis Video Streams then returns a stream of chunks in order by fragment number.

> Note
>
> You must first call the `GetDataEndpoint` API to get an endpoint. Then send the `GetMedia` requests to this endpoint using the --endpoint-url parameter.

When you put media data (fragments) on a stream, Kinesis Video Streams stores each incoming fragment and related metadata in what is called a "chunk." For more information, see PutMedia (p. 166). The `GetMedia` API returns a stream of these chunks starting from the chunk that you specify in the request.

The following limits apply when using the `GetMedia` API:

- A client can call `GetMedia` up to five times per second per stream.
- Kinesis Video Streams sends media data at a rate of up to 25 megabytes per second (or 200 megabits per second) during a `GetMedia` session.

Request Syntax

```
POST /getMedia HTTP/1.1
Content-type: application/json

{
   "StartSelector": {
      "AfterFragmentNumber": "string",
      "ContinuationToken": "string",
      "StartSelectorType": "string",
      "StartTimestamp": number
   },
   "StreamARN": "string",
   "StreamName": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

StartSelector (p. 162)

Identifies the starting chunk to get from the specified stream.

Type: StartSelector (p. 189) object

Required: Yes

StreamARN (p. 162)

The ARN of the stream from where you want to get the media content. If you don't specify the `streamARN`, you must specify the `streamName`.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

Required: No

StreamName (p. 162)

The Kinesis video stream name from where you want to get the media content. If you don't specify the `streamName`, you must specify the `streamARN`.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

Response Syntax

```
HTTP/1.1 200
Content-Type: ContentType

Payload
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The response returns the following HTTP headers.

ContentType (p. 163)

The content type of the requested media.

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `^[a-zA-Z0-9_\.\-]+$`

The response returns the following as the HTTP body.

Payload (p. 163)

The payload Kinesis Video Streams returns is a sequence of chunks from the specified stream. For information about the chunks, see PutMedia (p. 166). The chunks that Kinesis Video Streams returns in the `GetMedia` call also include the following additional Matroska (MKV) tags:

- AWS_KINESISVIDEO_CONTINUATION_TOKEN (UTF-8 string) - In the event your `GetMedia` call terminates, you can use this continuation token in your next request to get the next chunk where the last request terminated.
- AWS_KINESISVIDEO_MILLIS_BEHIND_NOW (UTF-8 string) - Client applications can use this tag value to determine how far behind the chunk returned in the response is from the latest chunk on the stream.
- AWS_KINESISVIDEO_FRAGMENT_NUMBER - Fragment number returned in the chunk.
- AWS_KINESISVIDEO_SERVER_TIMESTAMP - Server time stamp of the fragment.
- AWS_KINESISVIDEO_PRODUCER_TIMESTAMP - Producer time stamp of the fragment.

The following tags will be present if an error occurs:

- AWS_KINESISVIDEO_ERROR_CODE - String description of an error that caused GetMedia to stop.
- AWS_KINESISVIDEO_ERROR_ID: Integer code of the error.

The error codes are as follows:

- 3002 - Error writing to the stream
- 4000 - Requested fragment is not found
- 4500 - Access denied for the stream's KMS key
- 4501 - Stream's KMS key is disabled
- 4502 - Validation error on the Stream's KMS key
- 4503 - KMS key specified in the stream is unavailable
- 4504 - Invalid usage of the KMS key specified in the stream
- 4505 - Invalid state of the KMS key specified in the stream
- 4506 - Unable to find the KMS key specified in the stream
- 5000 - Internal error

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

ConnectionLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client connections.

HTTP Status Code: 400

InvalidArgumentException

The value for this input parameter is invalid.

HTTP Status Code: 400

InvalidEndpointException

Status Code: 400, Caller used wrong endpoint to write data to a stream. On receiving such an exception, the user must call `GetDataEndpoint` with `AccessMode` set to "READ" and use the endpoint Kinesis Video returns in the next `GetMedia` call.

HTTP Status Code: 400

NotAuthorizedException

Status Code: 403, The caller is not authorized to perform an operation on the given stream, or the token has expired.

HTTP Status Code: 401

ResourceNotFoundException

Status Code: 404, The stream with the given name does not exist.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

PutMedia

Service: Amazon Kinesis Video Streams Media

Use this API to send media data to a Kinesis video stream.

> **Note**
>
> Before using this API, you must call the `GetDataEndpoint` API to get an endpoint. You then specify the endpoint in your `PutMedia` request.

In the request, you use the HTTP headers to provide parameter information, for example, stream name, time stamp, and whether the time stamp value is absolute or relative to when the producer started recording. You use the request body to send the media data. Kinesis Video Streams supports only the Matroska (MKV) container format for sending media data using this API.

You have the following options for sending data using this API:

- Send media data in real time: For example, a security camera can send frames in real time as it generates them. This approach minimizes the latency between the video recording and data sent on the wire. This is referred to as a continuous producer. In this case, a consumer application can read the stream in real time or when needed.
- Send media data offline (in batches): For example, a body camera might record video for hours and store it on the device. Later, when you connect the camera to the docking port, the camera can start a `PutMedia` session to send data to a Kinesis video stream. In this scenario, latency is not an issue.

When using this API, note the following considerations:

- You must specify either `streamName` or `streamARN`, but not both.
- You might find it easier to use a single long-running `PutMedia` session and send a large number of media data fragments in the payload. Note that for each fragment received, Kinesis Video Streams sends one or more acknowledgements. Potential network considerations might cause you to not get all these acknowledgements as they are generated.
- You might choose multiple consecutive `PutMedia` sessions, each with fewer fragments to ensure that you get all acknowledgements from the service in real time.

> **Note**
>
> If you send data to the same stream on multiple simultaneous `PutMedia` sessions, the media fragments get interleaved on the stream. You should make sure that this is OK in your application scenario.

The following limits apply when using the `PutMedia` API:

- A client can call `PutMedia` up to five times per second per stream.
- A client can send up to five fragments per second per stream.
- Kinesis Video Streams reads media data at a rate of up to 12.5 MB/second, or 100 Mbps during a `PutMedia` session.

Note the following constraints. In these cases, Kinesis Video Streams sends the Error acknowledgement in the response.

- Fragments that have time codes spanning longer than 10 seconds and that contain more than 50 megabytes of data are not allowed.

- An MKV stream containing more than one MKV segment or containing disallowed MKV elements (like `track*`) also results in the Error acknowledgement.

Kinesis Video Streams stores each incoming fragment and related metadata in what is called a "chunk." The fragment metadata includes the following:

- The MKV headers provided at the start of the `PutMedia` request
- The following Kinesis Video Streams-specific metadata for the fragment:
  - `server_timestamp` - Time stamp when Kinesis Video Streams started receiving the fragment.
  - `producer_timestamp` - Time stamp, when the producer started recording the fragment. Kinesis Video Streams uses three pieces of information received in the request to calculate this value.
    - The fragment timecode value received in the request body along with the fragment.
    - Two request headers: `producerStartTimestamp` (when the producer started recording) and `fragmentTimeCodeType` (whether the fragment timecode in the payload is absolute or relative).

    Kinesis Video Streams then computes the `producer_timestamp` for the fragment as follows:

    If `fragmentTimeCodeType` is relative, then

    `producer_timestamp` = `producerStartTimeStamp` + fragment timecode

    If `fragmentTimeCodeType` is absolute, then

    `producer_timestamp` = fragment timecode (converted to milliseconds)
  - Unique fragment number assigned by Kinesis Video Streams.

  Note

  When you make the `GetMedia` request, Kinesis Video Streams returns a stream of these chunks. The client can process the metadata as needed.

  Note

  This operation is only available for the AWS SDK for Java. It is not supported in AWS SDKs for other languages.

Request Syntax

```
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 Request Parameters

The request requires the following URI parameters.

FragmentTimecodeType (p. 167)

You pass this value as the `x-amzn-fragment-timecode-type` HTTP header.

Indicates whether timecodes in the fragments (payload, HTTP request body) are absolute or relative to `producerStartTimestamp`. Kinesis Video Streams uses this information to compute the `producer_timestamp` for the fragment received in the request, as described in the API overview.

Valid Values: `ABSOLUTE | RELATIVE`

ProducerStartTimestamp (p. 167)

You pass this value as the `x-amzn-producer-start-timestamp` HTTP header.

This is the producer time stamp at which the producer started recording the media (not the time stamp of the specific fragments in the request).

StreamARN (p. 167)

You pass this value as the `x-amzn-stream-arn` HTTP header.

Amazon Resource Name (ARN) of the Kinesis video stream where you want to write the media content. If you don't specify the `streamARN`, you must specify the `streamName`.

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

StreamName (p. 167)

You pass this value as the `x-amzn-stream-name` HTTP header.

Name of the Kinesis video stream where you want to write the media content. If you don't specify the `streamName`, you must specify the `streamARN`.

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Request Body

The request accepts the following binary data.

Payload (p. 167)

The media content to write to the Kinesis video stream. In the current implementation, Kinesis Video Streams supports only the Matroska (MKV) container format with a single MKV segment. A segment can contain one or more clusters.

Note

Each MKV cluster maps to a Kinesis video stream fragment. Whatever cluster duration you choose becomes the fragment duration.

Response Syntax

```
HTTP/1.1 200

Payload
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The response returns the following as the HTTP body.

Payload (p. 168)

After Kinesis Video Streams successfully receives a `PutMedia` request, the service validates the request headers. The service then starts reading the payload and first sends an HTTP 200 response.

The service then returns a stream containing a series of JSON objects (`Acknowledgement` objects) separated by newlines. The acknowledgements are received on the same connection on which the media data is sent. There can be many acknowledgements for a `PutMedia` request. Each `Acknowledgement` consists of the following key-value pairs:

- `AckEventType` - Event type the acknowledgement represents.
  - Buffering: Kinesis Video Streams has started receiving the fragment. Kinesis Video Streams sends the first Buffering acknowledgement when the first byte of fragment data is received.
  - Received: Kinesis Video Streams received the entire fragment. If you did not configure the stream to persist the data, the producer can stop buffering the fragment upon receiving this acknowledgement.
  - Persisted: Kinesis Video Streams has persisted the fragment (for example, to Amazon S3). You get this acknowledgement if you configured the stream to persist the data. After you receive this acknowledgement, the producer can stop buffering the fragment.
  - Error: Kinesis Video Streams ran into an error while processing the fragment. You can review the error code and determine the next course of action.
  - Idle: The `PutMedia` session is in-progress. However, Kinesis Video Streams is currently not receiving data. Kinesis Video Streams sends this acknowledgement periodically for up to 30 seconds after the last received data. If no data is received within the 30 seconds, Kinesis Video Streams closes the request.

    Note

    This acknowledgement can help a producer determine if the `PutMedia` connection is alive, even if it is not sending any data.
- `FragmentTimeCode` - Fragment timecode for which acknowledgement is sent.

  The element can be missing if the `AckEventType` is Idle.
- `FragmentNumber` - Kinesis Video Streams-generated fragment number for which the acknowledgement is sent.
- `ErrorId` and `ErrorCode` - If the `AckEventType` is ErrorId, this field provides corresponding error code. The following is the list of error codes:
  - 4000 - Error reading the data stream.
  - 4001 - Fragment size is greater than maximum limit, 50 MB, allowed.
  - 4002 - Fragment duration is greater than maximum limit, 10 seconds, allowed.
  - 4003 - Connection duration is greater than maximum allowed threshold.
  - 4004 - Fragment timecode is less than the timecode previous time code (within a `PutMedia` call, you cannot send fragments out of order).
  - 4005 - More than one track is found in MKV.
  - 4006 - Failed to parse the input stream as valid MKV format.
  - 4007 - Invalid producer timestamp.
  - 4008 - Stream no longer exists (deleted).
  - 4500 - Access to the stream's specified KMS key is denied.
  - 4501 - The stream's specified KMS key is disabled.
  - 4502 - The stream's specified KMS key failed validation.
  - 4503 - The stream's specified KMS key is unavailable.
  - 4504 - Invalid usage of the stream's specified KMS key.
  - 4505 - The stream's specified KMS key is in an invalid state.

- 4506 - The stream's specified KMS key is not found.
- 5000 - Internal service error
- 5001 - Kinesis Video Streams failed to persist fragments to the data store.

  Note

  The producer, while sending the payload for a long running `PutMedia` request, should read the response for acknowledgements. A producer might receive chunks of acknowledgements at the same time, due to buffering on an intermediate proxy server. A producer that wants to receive timely acknowledgements can send fewer fragments in each `PutMedia` request.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

ConnectionLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client connections.

HTTP Status Code: 400

InvalidArgumentException

The value for this input parameter is invalid.

HTTP Status Code: 400

InvalidEndpointException

Status Code: 400, Caller used wrong endpoint to write data to a stream. On receiving such an exception, the user must call `GetDataEndpoint` with `AccessMode` set to "READ" and use the endpoint Kinesis Video returns in the next `GetMedia` call.

HTTP Status Code: 400

NotAuthorizedException

Status Code: 403, The caller is not authorized to perform an operation on the given stream, or the token has expired.

HTTP Status Code: 401

ResourceNotFoundException

Status Code: 404, The stream with the given name does not exist.

HTTP Status Code: 404

Example

Acknowledgement Format

The format of the acknowledgement is as follows:

```
{
       Acknowledgement : {
          “EventType”: enum
          "FragmentTimecode": Long,
          “FragmentNumber”: Long,
          “ErrorId” : String
      }
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

Amazon Kinesis Video Streams Archived Media

The following actions are supported by Amazon Kinesis Video Streams Archived Media:

- GetHLSStreamingSessionURL (p. 172)
- GetMediaForFragmentList (p. 179)
- ListFragments (p. 182)

GetHLSStreamingSessionURL

Service: Amazon Kinesis Video Streams Archived Media

Retrieves an HTTP Live Streaming (HLS) URL for the stream. You can then open the URL in a browser or media player to view the stream contents.

You must specify either the `StreamName` or the `StreamARN`.

An Amazon Kinesis video stream has the following requirements for providing data through HLS:

- The media type must be `video/h264`.
- Data retention must be greater than 0.
- The fragments must contain codec private data in the AVC (Advanced Video Coding) for H.264 format (MPEG-4 specification ISO/IEC 14496-15). For information about adapting stream data to a given format, see NAL Adaptation Flags.

Kinesis Video Streams HLS sessions contain fragments in the fragmented MPEG-4 form (also called fMP4 or CMAF), rather than the MPEG-2 form (also called TS chunks, which the HLS specification also supports). For more information about HLS fragment types, see the HLS specification.

The following procedure shows how to use HLS with Kinesis Video Streams:

1. Get an endpoint using GetDataEndpoint, specifying `GET_HLS_STREAMING_SESSION_URL` for the `APIName` parameter.
2. Retrieve the HLS URL using `GetHLSStreamingSessionURL`. Kinesis Video Streams creates an HLS streaming session to be used for accessing content in a stream using the HLS protocol. `GetHLSStreamingSessionURL` returns an authenticated URL (that includes an encrypted session token) for the session's HLS master playlist (the root resource needed for streaming with HLS).

   > **Note**
   >
   > Don't share or store this token where an unauthorized entity could access it. The token provides access to the content of the stream. Safeguard the token with the same measures that you would use with your AWS credentials.

   The media that is made available through the playlist consists only of the requested stream, time range, and format. No other media data (such as frames outside the requested window or alternate bit rates) is made available.
3. Provide the URL (containing the encrypted session token) for the HLS master playlist to a media player that supports the HLS protocol. Kinesis Video Streams makes the HLS media playlist, initialization fragment, and media fragments available through the master playlist URL. The initialization fragment contains the codec private data for the stream, and other data needed to set up the video decoder and renderer. The media fragments contain H.264-encoded video frames and time stamps.
4. The media player receives the authenticated URL and requests stream metadata and media data normally. When the media player requests data, it calls the following actions:
   - GetHLSMasterPlaylist: Retrieves an HLS master playlist, which contains a URL for the `GetHLSMediaPlaylist` action, and additional metadata for the media player, including estimated bit rate and resolution.
   - GetHLSMediaPlaylist: Retrieves an HLS media playlist, which contains a URL to access the MP4 initialization fragment with the `GetMP4InitFragment` action, and URLs to access the MP4 media fragments with the `GetMP4MediaFragment` actions. The HLS media playlist also contains metadata about the stream that the player needs to play it, such as whether the `PlaybackMode` is `LIVE` or `ON_DEMAND`. The HLS media playlist is typically static for sessions with a `PlaybackType` of `ON_DEMAND`. The HLS media playlist is continually updated with new fragments for sessions with a `PlaybackType` of `LIVE`.

- GetMP4InitFragment: Retrieves the MP4 initialization fragment. The media player typically loads the initialization fragment before loading any media fragments. This fragment contains the "`fytp`" and "`moov`" MP4 atoms, and the child atoms that are needed to initialize the media player decoder.

  The initialization fragment does not correspond to a fragment in a Kinesis video stream. It contains only the codec private data for the stream, which the media player needs to decode video frames.
- GetMP4MediaFragment: Retrieves MP4 media fragments. These fragments contain the "`moof`" and "`mdat`" MP4 atoms and their child atoms, containing the encoded fragment's video frames and their time stamps.

  Note

  After the first media fragment is made available in a streaming session, any fragments that don't contain the same codec private data are excluded in the HLS media playlist. Therefore, the codec private data does not change between fragments in a session.

  Data retrieved with this action is billable. See Pricing for details.

Note

The following restrictions apply to HLS sessions:

- A streaming session URL should not be shared between players. The service might throttle a session if multiple media players are sharing it. For connection limits, see Kinesis Video Streams Limits.
- A Kinesis video stream can have a maximum of five active HLS streaming sessions. If a new session is created when the maximum number of sessions is already active, the oldest (earliest created) session is closed. The number of active `GetMedia` connections on a Kinesis video stream does not count against this limit, and the number of active HLS sessions does not count against the active `GetMedia` connection limit.

You can monitor the amount of data that the media player consumes by monitoring the `GetMP4MediaFragment.OutgoingBytes` Amazon CloudWatch metric. For information about using CloudWatch to monitor Kinesis Video Streams, see Monitoring Kinesis Video Streams. For pricing information, see Amazon Kinesis Video Streams Pricing and AWS Pricing. Charges for both HLS sessions and outgoing AWS data apply.

For more information about HLS, see HTTP Live Streaming on the Apple Developer site.

Request Syntax

```
POST /getHLSStreamingSessionURL HTTP/1.1
Content-type: application/json

{
   "DiscontinuityMode": "string",
   "Expires": number,
   "HLSFragmentSelector": {
      "FragmentSelectorType": "string",
      "TimestampRange": {
         "EndTimestamp": number,
         "StartTimestamp": number
      }
   },
   "MaxMediaPlaylistFragmentResults": number,
   "PlaybackMode": "string",
   "StreamARN": "string",
   "StreamName": "string"
```

```
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

DiscontinuityMode (p. 173)

Specifies when flags marking discontinuities between fragments will be added to the media playlists. The default is `ALWAYS` when HLSFragmentSelector (p. 193) is `SERVER_TIMESTAMP`, and `NEVER` when it is `PRODUCER_TIMESTAMP`.

Media players typically build a timeline of media content to play, based on the time stamps of each fragment. This means that if there is any overlap between fragments (as is typical if HLSFragmentSelector (p. 193) is `SERVER_TIMESTAMP`), the media player timeline has small gaps between fragments in some places, and overwrites frames in other places. When there are discontinuity flags between fragments, the media player is expected to reset the timeline, resulting in the fragment being played immediately after the previous fragment. We recommend that you always have discontinuity flags between fragments if the fragment time stamps are not accurate or if fragments might be missing. You should not place discontinuity flags between fragments for the player timeline to accurately map to the producer time stamps.

Type: String

Valid Values: `ALWAYS | NEVER`

Required: No

Expires (p. 173)

The time in seconds until the requested session expires. This value can be between 300 (5 minutes) and 43200 (12 hours).

When a session expires, no new calls to `GetHLSMasterPlaylist`, `GetHLSMediaPlaylist`, `GetMP4InitFragment`, or `GetMP4MediaFragment` can be made for that session.

The default is 300 (5 minutes).

Type: Integer

Valid Range: Minimum value of 300. Maximum value of 43200.

Required: No

HLSFragmentSelector (p. 173)

The time range of the requested fragment, and the source of the time stamps.

This parameter is required if `PlaybackMode` is `ON_DEMAND`. This parameter is optional if `PlaybackMode` is `LIVE`. If `PlaybackMode` is `LIVE`, the `FragmentSelectorType` can be set, but the `TimestampRange` should not be set. If `PlaybackMode` is `ON_DEMAND`, both `FragmentSelectorType` and `TimestampRange` must be set.

Type: HLSFragmentSelector (p. 193) object

Required: No

MaxMediaPlaylistFragmentResults (p. 173)

The maximum number of fragments that are returned in the HLS media playlists.

When the `PlaybackMode` is `LIVE`, the most recent fragments are returned up to this value. When the `PlaybackMode` is `ON_DEMAND`, the oldest fragments are returned, up to this maximum number.

When there are a higher number of fragments available in a live HLS media playlist, video players often buffer content before starting playback. Increasing the buffer size increases the playback latency, but it decreases the likelihood that rebuffering will occur during playback. We recommend that a live HLS media playlist have a minimum of 3 fragments and a maximum of 10 fragments.

The default is 5 fragments if `PlaybackMode` is `LIVE`, and 1,000 if `PlaybackMode` is `ON_DEMAND`.

The maximum value of 1,000 fragments corresponds to more than 16 minutes of video on streams with 1-second fragments, and more than 2 1/2 hours of video on streams with 10-second fragments.

Type: Long

Valid Range: Minimum value of 1. Maximum value of 1000.

Required: No

PlaybackMode (p. 173)

Whether to retrieve live or archived, on-demand data.

Features of the two types of session include the following:

- **`LIVE`** : For sessions of this type, the HLS media playlist is continually updated with the latest fragments as they become available. We recommend that the media player retrieve a new playlist on a one-second interval. When this type of session is played in a media player, the user interface typically displays a "live" notification, with no scrubber control for choosing the position in the playback window to display.

  Note

  In `LIVE` mode, the newest available fragments are included in an HLS media playlist, even if there is a gap between fragments (that is, if a fragment is missing). A gap like this might cause a media player to halt or cause a jump in playback. In this mode, fragments are not added to the HLS media playlist if they are older than the newest fragment in the playlist. If the missing fragment becomes available after a subsequent fragment is added to the playlist, the older fragment is not added, and the gap is not filled.
- **`ON_DEMAND`** : For sessions of this type, the HLS media playlist contains all the fragments for the session, up to the number that is specified in `MaxMediaPlaylistFragmentResults`. The playlist must be retrieved only once for each session. When this type of session is played in a media player, the user interface typically displays a scrubber control for choosing the position in the playback window to display.

In both playback modes, if `FragmentSelectorType` is `PRODUCER_TIMESTAMP`, and if there are multiple fragments with the same start time stamp, the fragment that has the larger fragment number (that is, the newer fragment) is included in the HLS media playlist. The other fragments are not included. Fragments that have different time stamps but have overlapping durations are still included in the HLS media playlist. This can lead to unexpected behavior in the media player.

The default is `LIVE`.

Type: String

Valid Values: `LIVE | ON_DEMAND`

Required: No

StreamARN (p. 173)

The Amazon Resource Name (ARN) of the stream for which to retrieve the HLS master playlist URL.

You must specify either the `StreamName` or the `StreamARN`.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

Required: No

StreamName (p. 173)

The name of the stream for which to retrieve the HLS master playlist URL.

You must specify either the `StreamName` or the `StreamARN`.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

Response Syntax

```
HTTP/1.1 200
Content-type: application/json

{
   "HLSStreamingSessionURL": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

HLSStreamingSessionURL (p. 176)

The URL (containing the session token) that a media player can use to retrieve the HLS master playlist.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

InvalidArgumentException

A specified parameter exceeds its restrictions, is not supported, or can't be used.

HTTP Status Code: 400

InvalidCodecPrivateDataException

The Codec Private Data in the video stream is not valid for this operation.

HTTP Status Code: 400

MissingCodecPrivateDataException

No Codec Private Data was found in the video stream.

HTTP Status Code: 400

NoDataRetentionException

A `PlaybackMode` of `ON_DEMAND` was requested for a stream that does not retain data (that is, has a `DataRetentionInHours` of 0).

HTTP Status Code: 400

NotAuthorizedException

Status Code: 403, The caller is not authorized to perform an operation on the given stream, or the token has expired.

HTTP Status Code: 401

ResourceNotFoundException

`GetMedia` throws this error when Kinesis Video Streams can't find the stream that you specified.

`GetHLSStreamingSessionURL` throws this error if a session with a `PlaybackMode` of `ON_DEMAND` is requested for a stream that has no fragments within the requested time range, or if a session with a `PlaybackMode` of `LIVE` is requested for a stream that has no fragments within the last 30 seconds.

HTTP Status Code: 404

UnsupportedStreamMediaTypeException

An HLS streaming session was requested for a stream with a media type that is not `video/h264`.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

GetMediaForFragmentList

Service: Amazon Kinesis Video Streams Archived Media

Gets media for a list of fragments (specified by fragment number) from the archived data in an Amazon Kinesis video stream.

> Note
>
> You must first call the `GetDataEndpoint` API to get an endpoint. Then send the `GetMediaForFragmentList` requests to this endpoint using the --endpoint-url parameter.

The following limits apply when using the `GetMediaForFragmentList` API:

- A client can call `GetMediaForFragmentList` up to five times per second per stream.
- Kinesis Video Streams sends media data at a rate of up to 25 megabytes per second (or 200 megabits per second) during a `GetMediaForFragmentList` session.

Request Syntax

```
POST /getMediaForFragmentList HTTP/1.1
Content-type: application/json

{
   "Fragments": [ "string" ],
   "StreamName": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

Fragments (p. 179)

A list of the numbers of fragments for which to retrieve media. You retrieve these values with ListFragments (p. 182).

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 1000 items.

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `^[0-9]+$`

Required: Yes

StreamName (p. 179)

The name of the stream from which to retrieve fragment media.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: Yes

Response Syntax

```
HTTP/1.1 200
Content-Type: ContentType

Payload
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The response returns the following HTTP headers.

ContentType (p. 180)

The content type of the requested media.

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `^[a-zA-Z0-9_\.\-]+$`

The response returns the following as the HTTP body.

Payload (p. 180)

The payload that Kinesis Video Streams returns is a sequence of chunks from the specified stream. For information about the chunks, see PutMedia. The chunks that Kinesis Video Streams returns in the `GetMediaForFragmentList` call also include the following additional Matroska (MKV) tags:

- AWS_KINESISVIDEO_FRAGMENT_NUMBER - Fragment number returned in the chunk.
- AWS_KINESISVIDEO_SERVER_SIDE_TIMESTAMP - Server-side time stamp of the fragment.
- AWS_KINESISVIDEO_PRODUCER_SIDE_TIMESTAMP - Producer-side time stamp of the fragment.

The following tags will be included if an exception occurs:

- AWS_KINESISVIDEO_FRAGMENT_NUMBER - The number of the fragment that threw the exception
- AWS_KINESISVIDEO_EXCEPTION_ERROR_CODE - The integer code of the exception
- AWS_KINESISVIDEO_EXCEPTION_MESSAGE - A text description of the exception

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

InvalidArgumentException

A specified parameter exceeds its restrictions, is not supported, or can't be used.

HTTP Status Code: 400

NotAuthorizedException

Status Code: 403, The caller is not authorized to perform an operation on the given stream, or the token has expired.

HTTP Status Code: 401

ResourceNotFoundException

`GetMedia` throws this error when Kinesis Video Streams can't find the stream that you specified.

`GetHLSStreamingSessionURL` throws this error if a session with a `PlaybackMode` of `ON_DEMAND` is requested for a stream that has no fragments within the requested time range, or if a session with a `PlaybackMode` of `LIVE` is requested for a stream that has no fragments within the last 30 seconds.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

ListFragments

Service: Amazon Kinesis Video Streams Archived Media

Returns a list of Fragment (p. 191) objects from the specified stream and time stamp range within the archived data.

Listing fragments is eventually consistent. This means that even if the producer receives an acknowledgment that a fragment is persisted, the result might not be returned immediately from a request to `ListFragments`. However, results are typically available in less than one second.

> Note
>
> You must first call the `GetDataEndpoint` API to get an endpoint. Then send the `ListFragments` requests to this endpoint using the --endpoint-url parameter.

Request Syntax

```
POST /listFragments HTTP/1.1
Content-type: application/json

{
   "FragmentSelector": {
      "FragmentSelectorType": "string",
      "TimestampRange": {
         "EndTimestamp": number,
         "StartTimestamp": number
      }
   },
   "MaxResults": number,
   "NextToken": "string",
   "StreamName": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

FragmentSelector (p. 182)

Describes the time stamp range and time stamp origin for the range of fragments to return.

Type: FragmentSelector (p. 192) object

Required: No

MaxResults (p. 182)

The total number of fragments to return. If the total number of fragments available is more than the value specified in `max-results`, then a ListFragments:NextToken (p. 183) is provided in the output that you can use to resume pagination.

Type: Long

Valid Range: Minimum value of 1. Maximum value of 1000.

Required: No

NextToken (p. 182)

A token to specify where to start paginating. This is the ListFragments:NextToken (p. 183) from a previously truncated response.

Type: String

Length Constraints: Minimum length of 1.

Required: No

StreamName (p. 182)

The name of the stream from which to retrieve a fragment list.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: Yes

Response Syntax

```
HTTP/1.1 200
Content-type: application/json

{
   "Fragments": [
      {
         "FragmentLengthInMilliseconds": number,
         "FragmentNumber": "string",
         "FragmentSizeInBytes": number,
         "ProducerTimestamp": number,
         "ServerTimestamp": number
      }
   ],
   "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Fragments (p. 183)

A list of archived Fragment (p. 191) objects from the stream that meet the selector criteria. Results are in no specific order, even across pages.

Type: Array of Fragment (p. 191) objects

NextToken (p. 183)

If the returned list is truncated, the operation returns this token to use to retrieve the next page of results. This value is `null` when there are no more results to return.

Type: String

Length Constraints: Minimum length of 1.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 195).

ClientLimitExceededException

Kinesis Video Streams has throttled the request because you have exceeded the limit of allowed client calls. Try making the call later.

HTTP Status Code: 400

InvalidArgumentException

A specified parameter exceeds its restrictions, is not supported, or can't be used.

HTTP Status Code: 400

NotAuthorizedException

Status Code: 403, The caller is not authorized to perform an operation on the given stream, or the token has expired.

HTTP Status Code: 401

ResourceNotFoundException

`GetMedia` throws this error when Kinesis Video Streams can't find the stream that you specified.

`GetHLSStreamingSessionURL` throws this error if a session with a `PlaybackMode` of `ON_DEMAND` is requested for a stream that has no fragments within the requested time range, or if a session with a `PlaybackMode` of `LIVE` is requested for a stream that has no fragments within the last 30 seconds.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

Data Types

The following data types are supported by Amazon Kinesis Video Streams:

- StreamInfo (p. 186)
- StreamNameCondition (p. 188)

The following data types are supported by Amazon Kinesis Video Streams Media:

- StartSelector (p. 189)

The following data types are supported by Amazon Kinesis Video Streams Archived Media:

- Fragment (p. 191)
- FragmentSelector (p. 192)
- HLSFragmentSelector (p. 193)
- HLSTimestampRange (p. 194)
- TimestampRange (p. 195)

Amazon Kinesis Video Streams

The following data types are supported by Amazon Kinesis Video Streams:

- StreamInfo (p. 186)
- StreamNameCondition (p. 188)

StreamInfo

Service: Amazon Kinesis Video Streams

An object describing a Kinesis video stream.

Contents

CreationTime

A time stamp that indicates when the stream was created.

Type: Timestamp

Required: No

DataRetentionInHours

How long the stream retains data, in hours.

Type: Integer

Valid Range: Minimum value of 0.

Required: No

DeviceName

The name of the device that is associated with the stream.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

KmsKeyId

The ID of the AWS Key Management Service (AWS KMS) key that Kinesis Video Streams uses to encrypt data on the stream.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2048.

Required: No

MediaType

The `MediaType` of the stream.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `[\w\-\.\+]+/[\w\-\.\+]+`

Required: No

Status

The status of the stream.

Type: String

Valid Values: `CREATING | ACTIVE | UPDATING | DELETING`

Required: No

StreamARN

The Amazon Resource Name (ARN) of the stream.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 1024.

Pattern: `arn:aws:kinesisvideo:[a-z0-9-]+:[0-9]+:[a-z]+/[a-zA-Z0-9_.-]+/[0-9]+`

Required: No

StreamName

The name of the stream.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

Version

The version of the stream.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 64.

Pattern: `[a-zA-Z0-9]+`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

StreamNameCondition

Service: Amazon Kinesis Video Streams

Specifies the condition that streams must satisfy to be returned when you list streams (see the `ListStreams` API). A condition has a comparison operation and a value. Currently, you can specify only the `BEGINS_WITH` operator, which finds streams whose names start with a given prefix.

Contents

ComparisonOperator

A comparison operator. Currently, you can specify only the `BEGINS_WITH` operator, which finds streams whose names start with a given prefix.

Type: String

Valid Values: `BEGINS_WITH`

Required: No

ComparisonValue

A value to compare.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Pattern: `[a-zA-Z0-9_.-]+`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Amazon Kinesis Video Streams Media

The following data types are supported by Amazon Kinesis Video Streams Media:

- StartSelector (p. 189)

StartSelector

Service: Amazon Kinesis Video Streams Media

Identifies the chunk on the Kinesis video stream where you want the `GetMedia` API to start returning media data. You have the following options to identify the starting chunk:

- Choose the latest (or oldest) chunk.
- Identify a specific chunk. You can identify a specific chunk either by providing a fragment number or time stamp (server or producer).
- Each chunk's metadata includes a continuation token as a Matroska (MKV) tag (`AWS_KINESISVIDEO_CONTINUATION_TOKEN`). If your previous `GetMedia` request terminated, you can use this tag value in your next `GetMedia` request. The API then starts returning chunks starting where the last API ended.

Contents

AfterFragmentNumber

Specifies the fragment number from where you want the `GetMedia` API to start returning the fragments.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `^[0-9]+$`

Required: No

ContinuationToken

Continuation token that Kinesis Video Streams returned in the previous `GetMedia` response. The `GetMedia` API then starts with the chunk identified by the continuation token.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `^[a-zA-Z0-9_\.\-]+$`

Required: No

StartSelectorType

Identifies the fragment on the Kinesis video stream where you want to start getting the data from.

- NOW - Start with the latest chunk on the stream.
- EARLIEST - Start with earliest available chunk on the stream.
- FRAGMENT_NUMBER - Start with the chunk containing the specific fragment. You must also specify the `StartFragmentNumber`.
- PRODUCER_TIMESTAMP or SERVER_TIMESTAMP - Start with the chunk containing a fragment with the specified producer or server time stamp. You specify the time stamp by adding `StartTimestamp`.
- CONTINUATION_TOKEN - Read using the specified continuation token.

Note

If you choose the NOW, EARLIEST, or CONTINUATION_TOKEN as the `startSelectorType`, you don't provide any additional information in the `startSelector`.

Type: String

Valid Values: `FRAGMENT_NUMBER | SERVER_TIMESTAMP | PRODUCER_TIMESTAMP | NOW | EARLIEST | CONTINUATION_TOKEN`

Required: Yes

StartTimestamp

A time stamp value. This value is required if you choose the PRODUCER_TIMESTAMP or the SERVER_TIMESTAMP as the `startSelectorType`. The `GetMedia` API then starts with the chunk containing the fragment that has the specified time stamp.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Amazon Kinesis Video Streams Archived Media

The following data types are supported by Amazon Kinesis Video Streams Archived Media:

- Fragment (p. 191)
- FragmentSelector (p. 192)
- HLSFragmentSelector (p. 193)
- HLSTimestampRange (p. 194)
- TimestampRange (p. 195)

Fragment

Service: Amazon Kinesis Video Streams Archived Media

Represents a segment of video or other time-delimited data.

Contents

FragmentLengthInMilliseconds

The playback duration or other time value associated with the fragment.

Type: Long

Required: No

FragmentNumber

The index value of the fragment.

Type: String

Length Constraints: Minimum length of 1.

Required: No

FragmentSizeInBytes

The total fragment size, including information about the fragment and contained media data.

Type: Long

Required: No

ProducerTimestamp

The time stamp from the producer corresponding to the fragment.

Type: Timestamp

Required: No

ServerTimestamp

The time stamp from the AWS server corresponding to the fragment.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

FragmentSelector

Service: Amazon Kinesis Video Streams Archived Media

Describes the time stamp range and time stamp origin of a range of fragments.

Only fragments with a start time stamp greater than or equal to the given start time and less than or equal to the end time are returned. For example, if a stream contains fragments with the following start time stamps:

- 00:00:00
- 00:00:02
- 00:00:04
- 00:00:06

A fragment selector range with a start time of 00:00:01 and end time of 00:00:04 would return the fragments with start times of 00:00:02 and 00:00:04.

Contents

FragmentSelectorType

The origin of the time stamps to use (Server or Producer).

Type: String

Valid Values: `PRODUCER_TIMESTAMP | SERVER_TIMESTAMP`

Required: Yes

TimestampRange

The range of time stamps to return.

Type: TimestampRange (p. 195) object

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

HLSFragmentSelector

Service: Amazon Kinesis Video Streams Archived Media

Contains the range of time stamps for the requested media, and the source of the time stamps.

Contents

FragmentSelectorType

The source of the time stamps for the requested media.

When `FragmentSelectorType` is set to `PRODUCER_TIMESTAMP` and GetHLSStreamingSessionURL:PlaybackMode (p. 175) is `ON_DEMAND`, the first fragment ingested with a producer time stamp within the specified FragmentSelector:TimestampRange (p. 192) is included in the media playlist. In addition, the fragments with producer time stamps within the `TimestampRange` ingested immediately following the first fragment (up to the GetHLSStreamingSessionURL:MaxMediaPlaylistFragmentResults (p. 175) value) are included.

Fragments that have duplicate producer time stamps are deduplicated. This means that if producers are producing a stream of fragments with producer time stamps that are approximately equal to the true clock time, the HLS media playlists will contain all of the fragments within the requested time stamp range. If some fragments are ingested within the same time range and very different points in time, only the oldest ingested collection of fragments are returned.

When `FragmentSelectorType` is set to `PRODUCER_TIMESTAMP` and GetHLSStreamingSessionURL:PlaybackMode (p. 175) is `LIVE`, the producer time stamps are used in the MP4 fragments and for deduplication. But the most recently ingested fragments based on server time stamps are included in the HLS media playlist. This means that even if fragments ingested in the past have producer time stamps with values now, they are not included in the HLS media playlist.

The default is `SERVER_TIMESTAMP`.

Type: String

Valid Values: `PRODUCER_TIMESTAMP | SERVER_TIMESTAMP`

Required: No

TimestampRange

The start and end of the time stamp range for the requested media.

This value should not be present if `PlaybackType` is `LIVE`.

Type: HLSTimestampRange (p. 194) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

HLSTimestampRange

Service: Amazon Kinesis Video Streams Archived Media

The start and end of the time stamp range for the requested media.

This value should not be present if `PlaybackType` is `LIVE`.

> Note
>
> The values in the `HLSTimestampRange` are inclusive. Fragments that begin before the start time but continue past it, or fragments that begin before the end time but continue past it, are included in the session.

Contents

EndTimestamp

The end of the time stamp range for the requested media. This value must be within 3 hours of the specified `StartTimestamp`, and it must be later than the `StartTimestamp` value.

If `FragmentSelectorType` for the request is `SERVER_TIMESTAMP`, this value must be in the past.

If the `HLSTimestampRange` value is specified, the `EndTimestamp` value is required.

> Note
>
> This value is inclusive. The `EndTimestamp` is compared to the (starting) time stamp of the fragment. Fragments that start before the `EndTimestamp` value and continue past it are included in the session.

Type: Timestamp

Required: No

StartTimestamp

The start of the time stamp range for the requested media.

If the `HLSTimestampRange` value is specified, the `StartTimestamp` value is required.

> Note
>
> This value is inclusive. Fragments that start before the `StartTimestamp` and continue past it are included in the session. If `FragmentSelectorType` is `SERVER_TIMESTAMP`, the `StartTimestamp` must be later than the stream head.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

TimestampRange

Service: Amazon Kinesis Video Streams Archived Media

The range of time stamps for which to return fragments.

Contents

EndTimestamp

The ending time stamp in the range of time stamps for which to return fragments.

Type: Timestamp

Required: Yes

StartTimestamp

The starting time stamp in the range of time stamps for which to return fragments.

Type: Timestamp

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Common Errors

This section lists the errors common to the API actions of all AWS services. For errors specific to an API action for this service, see the topic for that API action.

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

IncompleteSignature

The request signature does not conform to AWS standards.

HTTP Status Code: 400

InternalFailure

The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

InvalidAction

The action or operation requested is invalid. Verify that the action is typed correctly.

HTTP Status Code: 400

InvalidClientTokenId

The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403

InvalidParameterCombination

Parameters that must not be used together were used together.

HTTP Status Code: 400

InvalidParameterValue

An invalid or out-of-range value was supplied for the input parameter.

HTTP Status Code: 400

InvalidQueryParameter

The AWS query string is malformed or does not adhere to AWS standards.

HTTP Status Code: 400

MalformedQueryString

The query string contains a syntax error.

HTTP Status Code: 404

MissingAction

The request is missing an action or a required parameter.

HTTP Status Code: 400

MissingAuthenticationToken

The request must contain either a valid (registered) AWS access key ID or X.509 certificate.

HTTP Status Code: 403

MissingParameter

A required parameter for the specified action is not supplied.

HTTP Status Code: 400

OptInRequired

The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403

RequestExpired

The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400

ServiceUnavailable

The request has failed due to a temporary failure of the server.

HTTP Status Code: 503

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationError

The input fails to satisfy the constraints specified by an AWS service.

HTTP Status Code: 400

Common Parameters

The following list contains the parameters that all actions use for signing Signature Version 4 requests with a query string. Any action-specific parameters are listed in the topic for that action. For more information about Signature Version 4, see Signature Version 4 Signing Process in the Amazon Web Services General Reference.

Action

The action to be performed.

Type: string

Required: Yes

Version

The API version that the request is written for, expressed in the format YYYY-MM-DD.

Type: string

Required: Yes

X-Amz-Algorithm

The hash algorithm that you used to create the request signature.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Valid Values: `AWS4-HMAC-SHA256`

Required: Conditional

X-Amz-Credential

The credential scope value, which is a string that includes your access key, the date, the region you are targeting, the service you are requesting, and a termination string ("aws4_request"). The value is expressed in the following format: access_key/YYYYMMDD/region/service/aws4_request.

For more information, see Task 2: Create a String to Sign for Signature Version 4 in the Amazon Web Services General Reference.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-Date

The date that is used to create the signature. The format must be ISO 8601 basic format (YYYYMMDD'T'HHMMSS'Z'). For example, the following date time is a valid X-Amz-Date value: `20120325T120000Z`.

Condition: X-Amz-Date is optional for all requests; it can be used to override the date used for signing requests. If the Date header is specified in the ISO 8601 basic format, X-Amz-Date is not required. When X-Amz-Date is used, it always overrides the value of the Date header. For more information, see Handling Dates in Signature Version 4 in the Amazon Web Services General Reference.

Type: string

Required: Conditional

X-Amz-Security-Token

The temporary security token that was obtained through a call to AWS Security Token Service (AWS STS). For a list of services that support temporary security credentials from AWS Security Token Service, go to AWS Services That Work with IAM in the IAM User Guide.

Condition: If you're using temporary security credentials from the AWS Security Token Service, you must include the security token.

Type: string

Required: Conditional

X-Amz-Signature

Specifies the hex-encoded signature that was calculated from the string to sign and the derived signing key.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-SignedHeaders

Specifies all the HTTP headers that were included as part of the canonical request. For more information about specifying signed headers, see Task 1: Create a Canonical Request For Signature Version 4 in the Amazon Web Services General Reference.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional