[AWS Black Belt Online Seminar]

Amazon Kinesis Video Streams

サービスカットシリーズ

Solutions Architect 三平 悠磨
2020/09/30

AWS 公式 Webinar
https://amzn.to/JPWebinar

過去資料
https://amzn.to/JPArchive

自己紹介

- 三平 悠磨
- IoT Specialist SA
- 経歴
    - 会話AI開発
    - 家庭用ロボット開発
- 好きなサービス
    - AWS IoT Greengrass
    - Amazon Kinesis Video Streams
- Twitter: @yu_ma_m

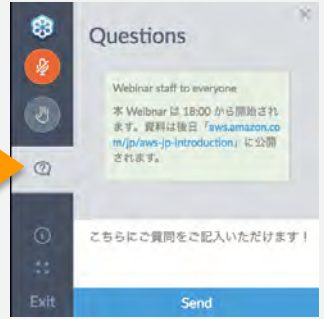
AWS Black Belt Online Seminar とは

「サービス別」「ソリューション別」「業種別」のそれぞれのテーマに分かれて、アマゾン ウェブ サービス ジャパン株式会社が主催するオンラインセミナーシリーズです。

質問を投げることができます！

- 書き込んだ質問は、主催者にしか見えません
- 今後のロードマップに関するご質問は
  お答えできませんのでご了承下さい

① 吹き出しをクリック
② 質問を入力
③ Sendをクリック



Twitter ハッシュタグは以下をご利用ください
#awsblackbelt

内容についての注意点

- 本資料では2020年09月30日時点のサービス内容および価格についてご説明しています。最新の情報はAWS公式ウェブサイト(http://aws.amazon.com)にてご確認ください。

- 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます。

- 価格は税抜表記となっています。日本居住者のお客様には別途消費税をご請求させていただきます。

- AWS does not offer binding price quotes. AWS pricing is publicly available and is subject to change in accordance with the AWS Customer Agreement available at http://aws.amazon.com/agreement/. Any pricing information included in this document is provided only as an estimate of usage charges for AWS services based on certain information that you have provided. Monthly charges will be based on your actual use of AWS services, and may vary from the estimates provided.

本日のアジェンダ

- Amazon Kinesis Video Streams の概要
- Amazon Kinesis Video Streams - メディア形式で収集
    1. メディアの取り込み
    2. メディアの保存とインデックス
    3. メディアの再生
    4. メディアの分析
- Amazon Kinesis Video Streams – WebRTC
- Amazon Kinesis Video Streams を活用した構成例
- サービス利用上のポイント
- まとめ

本日のアジェンダ

- Amazon Kinesis Video Streams の概要
- Amazon Kinesis Video Streams - メディア形式で収集
    1. メディアの取り込み
    2. メディアの保存とインデックス
    3. メディアの再生
    4. メディアの分析
- Amazon Kinesis Video Streams – WebRTC
- Amazon Kinesis Video Streams を活用した構成例
- サービス利用上のポイント
- まとめ

IoT における動画・音声のユースケースの例

デバイスからメディア(動画・音声)を収集して、再生・分析に利用する


ペットカメラ、見守りカメラ


インターホン


スマートスピーカー、文字起こしシステム


ドローン、ロボット


監視カメラ、ドライブレコーダー


スマートファクトリー

IoT メディアユースケースにおける技術領域

コネクテッドデバイス	映像伝送・保存	分析・機械学習	アプリケーション連携
• デバイスとの通信 • 遠隔制御 • デバイスの監視・管理	• クラウドへの送信 • メディアの長期保存 • ライブ再生・オンデマンド再生 • メディアのエクスポート	• フレームの取り出し • 画像認識 • 動画分析	• モバイル・ウェブアプリ • ユーザとデバイスの紐付け • ユーザや管理者への通知

aws

IoT メディアユースケースにおける課題

コネクテッドデバイス	映像伝送・保存	分析・機械学習	アプリケーション連携
• デバイスとの通信 • 遠隔制御 • デバイスの監視・管理	• クラウドへの送信 • ライブ再生・オンデマンド再生 • 動画のエクスポート • 動画の長期保存	• フレームの取り出し • 画像認識 • 動画分析	• モバイル・ウェブアプリ • ユーザとデバイスの紐付け • ユーザや管理者への通知

スケーラビリティ・セキュリティ

デバイス向けソフトウェア開発	動画ストリーミングの専門知識	機械学習や画像処理の専門知識	アプリケーション開発

aws

IoT メディアユースケースにおける課題

コネクテッドデバイス	映像伝送・保存	分析・機械学習	アプリケーション連携
• デバイスとの通信 • 遠隔制御 • デバイスの監視・管理	• クラウドへの送信 • ライブ再生・オンデマンド再生 • 動画のエクスポート • 動画の長期保存	• フレームの取り出し • 画像認識 • 動画分析	• モバイル・ウェブアプリ • ユーザとデバイスの紐付け • ユーザや管理者への通知

スケーラビリティ・セキュリティ

デバイス向けソフトウェア開発	**動画ストリーミングの専門知識**	機械学習や画像処理の専門知識	アプリケーション開発

aws

IoT メディアユースケースにおける課題

コネクテッドデバイス	映像伝送・保存	分析・機械学習	アプリケーション連携
• デバイスとの通信 • 遠隔制御 • デバイスの監視・管理	• クラウドへの送信 • ライブ再生・オンデマンド再生 • 動画のエクスポート • 動画の長期保存	• フレームの取り出し • 画像認識 • 動画分析	• モバイル・ウェブアプリ • ユーザとデバイスの紐付け • ユーザや管理者への通知

スケーラビリティ・セキュリティ

デバイス向けソフトウェア開発	動画ストリーミングの専門知識	**機械学習や画像処理の専門知識**	アプリケーション開発

aws

IoT メディアユースケースにおける課題

コネクテッドデバイス	映像伝送・保存	分析・機械学習	アプリケーション連携
• デバイスとの通信 • 遠隔制御 • デバイスの監視・管理	• クラウドへの送信 • ライブ再生・オンデマンド再生 • 動画のエクスポート • 動画の長期保存	• フレームの取り出し • 画像認識 • 動画分析	• モバイル・ウェブアプリ • ユーザとデバイスの紐付け • ユーザや管理者への通知

スケーラビリティ・セキュリティ

デバイス向けソフトウェア開発	動画ストリーミングの専門知識	機械学習や画像処理の専門知識	**アプリケーション開発**

aws

Amazon Kinesis Video Streams および AWS サービスでカバーできる技術領域

コネクテッドデバイス	映像伝送・保存	分析・機械学習	アプリケーション連携
• デバイスとの通信 • 遠隔制御 • デバイスの監視・管理	• クラウドへの送信 • ライブ再生・オンデマンド再生 • 動画のエクスポート • 動画の長期保存	• フレームの取り出し • 画像認識 • 動画分析	• モバイル・ウェブアプリ • ユーザとデバイスの紐付け • ユーザや管理者への通知


AWS SDK 各種


AWS IoT Core


Amazon Kinesis Video Streams


Amazon S3


Amazon Rekognition Video


Amazon SageMaker


Amazon Cognito


Amazon SNS

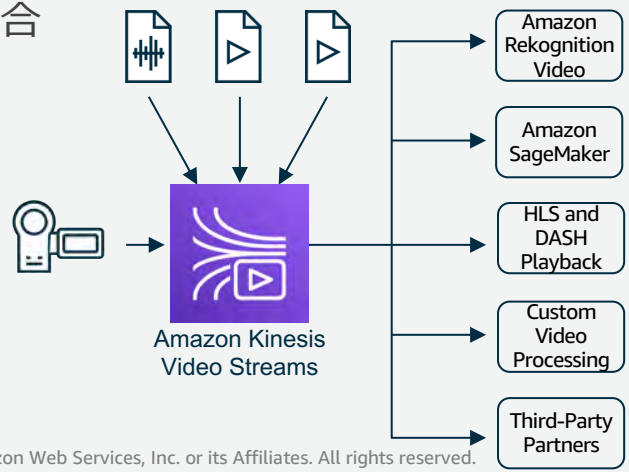
aws

Amazon Kinesis Video Streams - 2種類のストリーミング方法

メディア形式で収集

数百万台規模のデバイスからのセキュアなデータ取り込み

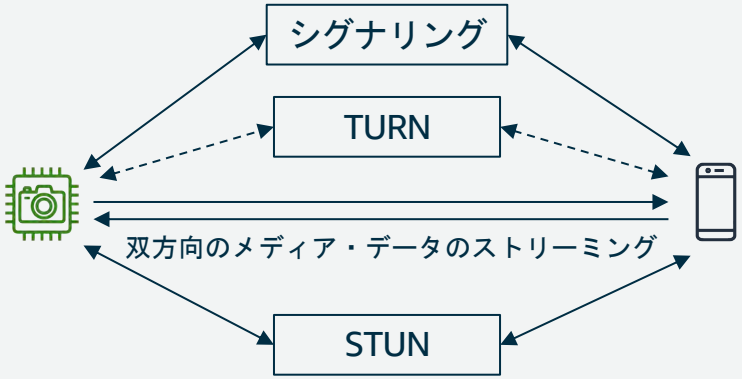
- 時系列でインデックスされたメディアデータの保存とAPI経由での検索
- カメラデバイス向けのSDK
- メディアの再生と機械学習サービスとの統合



WebRTC

WebRTC によるリアルタイムの双方向メディアストリーミング

- シグナリング、STUN、TURN のマネージドサービス
- 組み込みデバイス向けSDK、ウェブ・モバイルアプリ向けのSDK



Amazon Kinesis Video Streams の使い分け

	メディア形式で収集	WebRTC
動画データのクラウド保存 機械学習サービスでの分析	できる	できない
双方向ストリーミング	できない	できる
ライブ再生時のレイテンシ	2秒～	1秒未満
カメラデバイスからの ストリーミングに利用する SDK	Producer SDK	WebRTC SDK

※ レイテンシはデバイス性能、ネットワーク環境、映像のビットレート、フラグメントの設定などによって変化します

aws

本日のアジェンダ

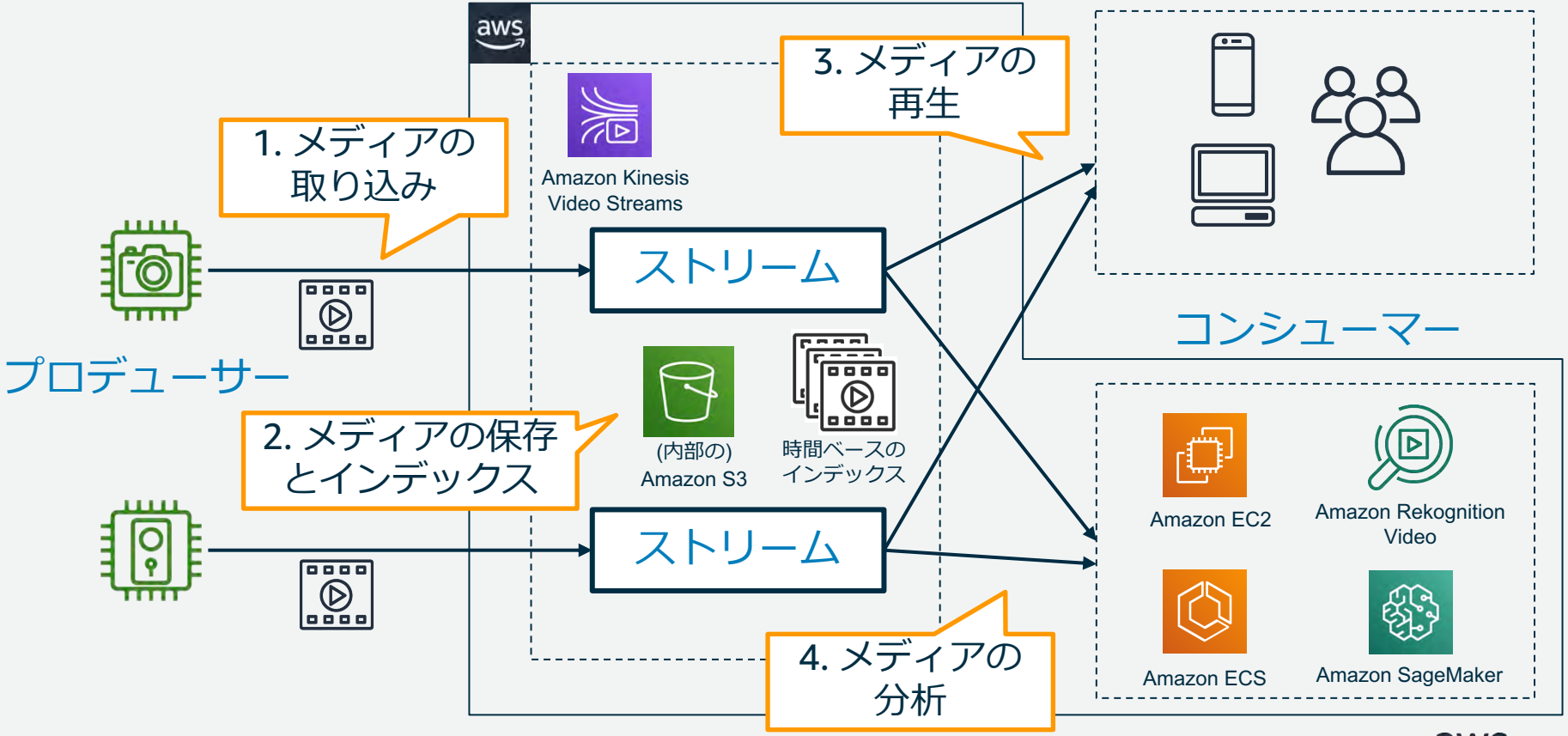
メディア形式で収集

- Amazon Kinesis Video Streams の概要
- Amazon Kinesis Video Streams - メディア形式で収集
    1. メディアの取り込み
    2. メディアの保存とインデックス
    3. メディアの再生
    4. メディアの分析
- Amazon Kinesis Video Streams – WebRTC
- Amazon Kinesis Video Streams を活用した構成例
- サービス利用上のポイント
- まとめ

aws

アーキテクチャの概要

メディア形式で収集



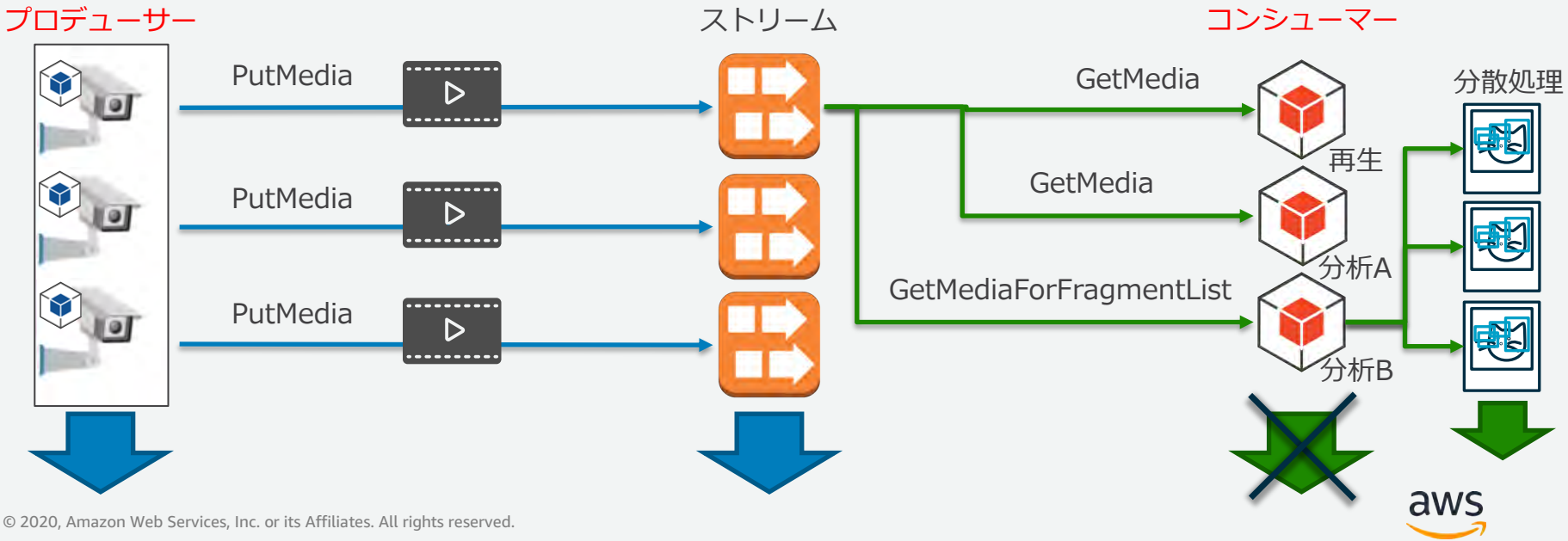
aws

スケールアウトの単位

メディア形式で収集

- **基本的にプロデューサーとストリームは1対1の関係**
- ストリームの単位で何百万ものデバイスにスケール
- PutMedia APIに呼出頻度や帯域幅の制限があるが、それ以外に**ストリームのキャパシティ管理は不要**
- デバイスのプロビジョニング時に、CreateStream APIによりストリームを動的に作成することを推奨

- **ストリームとコンシューマーは1対Nの関係だが、GetMedia API の同時接続数には制限がある**
- コンシューマーは処理内容の役割ごとに分ける
- コンシューマー単位での分散処理はせず、分散処理が必要であればフレーム単位などで考える
- 同じ理由で、**不特定多数への動画配信にも向かない**



動画データのフォーマット

メディア形式で収集

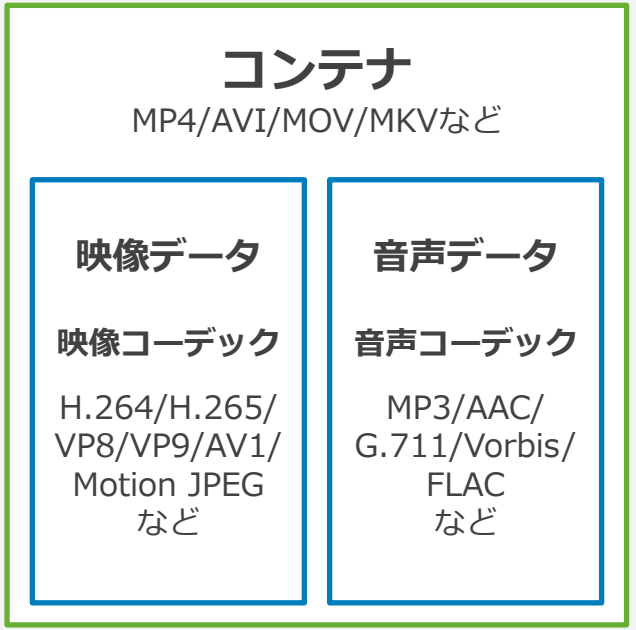
コンテナ と コーデック

■ コンテナ（コンテナフォーマット）

- 映像データと音声データなどをまとめて動画データにするためのファイルフォーマット（入れ物に相当）
- 入れ物なので、画質や音質には影響しない
- 代表的なコンテナとして、MP4、AVI、MOV、MKV などがある

■ コーデック

- コンテナに入れるデータを圧縮するためのアルゴリズム
- 映像データや音声データにそれぞれコーデックがある
- コンテナによって使用可能なコーデックが決まっている
- 代表的な映像コーデックとして H.264、H.265、VP8 などがある



本日のアジェンダ

- Amazon Kinesis Video Streams の概要
- Amazon Kinesis Video Streams - メディア形式で収集
    1. メディアの取り込み
    2. メディアの保存とインデックス
    3. メディアの再生
    4. メディアの分析
- Amazon Kinesis Video Streams – WebRTC
- Amazon Kinesis Video Streams を活用した構成例
- サービス利用上のポイント
- まとめ

メディアの取り込み

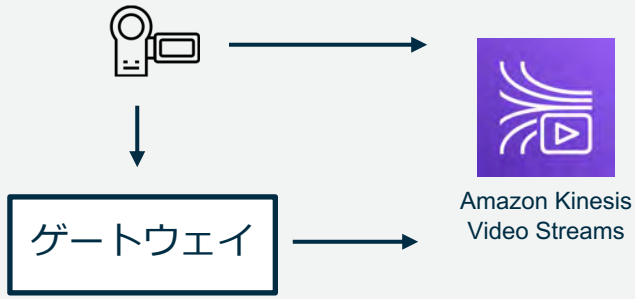
メディア形式で収集

メディアの取り込み方法

- PutMedia API による取り込み
- Amazon Kinesis Video Streams Producer SDK の利用 (C, C++, Java, Android)

メディアの取り込み経路

- カメラデバイスからの直接のストリーミング
- カメラと同一ネットワーク内のプロキシやゲートウェイ経由での取り込み



https://docs.aws.amazon.com/ja_jp/kinesisvideostreams/latest/dg/producer-sdk.html

PutMedia API

メディア形式で収集

MKVフォーマットのフラグメントをPayloadとして、HTTPS POSTする

- APIリクエストに必要なパラメータは HTTPヘッダに入れて送信する
- PutMedia API は Long Running Sessionのため、セッションを張って、そのセッションに対してフラグメントを送信する

Request

```
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
```

Response

```
HTTP/1.1 200

Payload
```

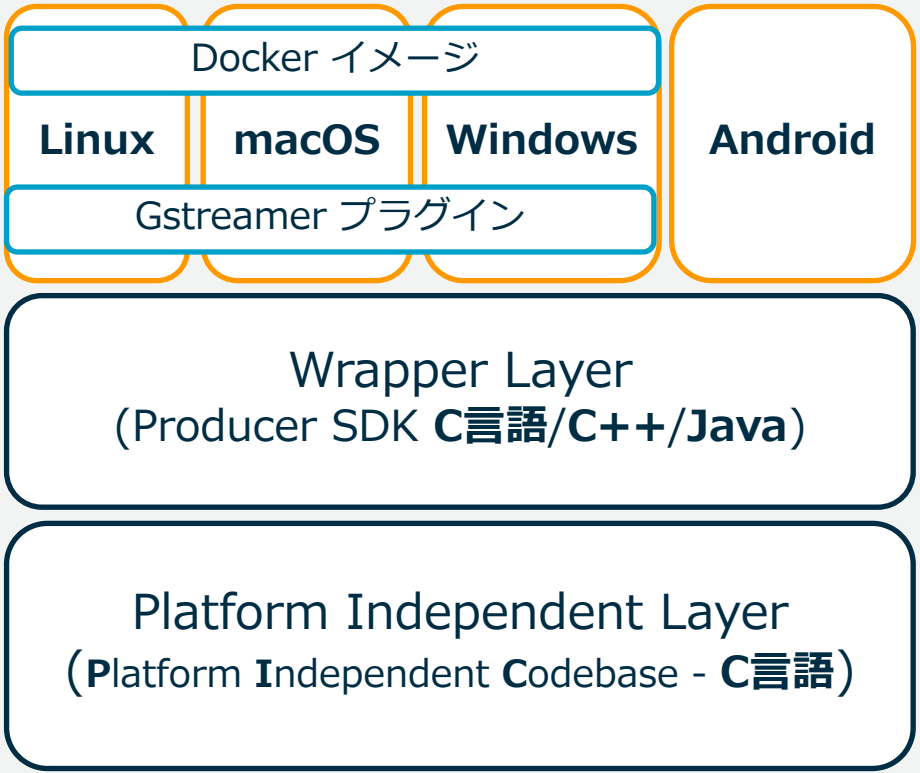
https://docs.aws.amazon.com/ja_jp/kinesisvideostreams/latest/dg/API_dataplane_PutMedia.html

aws

Amazon Kinesis Video Streams Producer SDK

メディア形式で収集

デバイス上のハードウェアメディアパイプラインと統合するためのSDK



デモアプリケーション

- アプリケーション開発者向け
- ターゲットOSにインストールしてそのまま利用
- ハードウェアやビデオソースを全てサポートするわけではないが、簡単に実行可能

Producer SDK

- カメラから取得した動画を統合したい開発者向け
- 柔軟にカスタマイズが可能な、オブジェクト指向の統合フレームワークを提供

コア機能を提供するライブラリ

- （カメラ）メーカー向け
- 様々なハードウェア環境に合わせて、ファームウェアレベルで動画ソースとの統合を実装可能
- 他のライブラリとは完全に独立

https://docs.aws.amazon.com/ja_jp/kinesisvideostreams/latest/dg/producer-sdk.html

aws

参考：GStreamer とは

オープンソースのマルチメディアアプリケーション開発用フレームワーク

- 動画プレイヤーのベースとなる部分などに使われており、動画のストリーミング配信、動画の変換・合成などの処理を実装できる
- 核となる部分以外は、プラグインのライブラリ群で構成されており、様々な通信プロトコルやコーデックに対応している
- Producer SDK C++ には、GStreamer で Amazon Kinesis Video Streams を使用するためのプラグイン・デモアプリケーションが含まれている

https://gstreamer.freedesktop.org/

aws

Producer SDK のデモアプリケーション

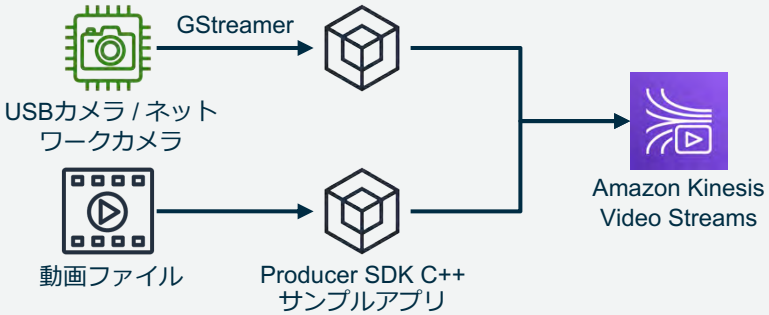
メディア形式で収集

Producer SDK C++

kvs_gstreamer_sample

下記の映像ソースから Amazon Kinesis Video Streams に動画を送信

- USB等で接続したWebカメラ
- RTSP 接続のネットワークカメラ
- 既存の動画ファイル (MKV, MP4, MPEG_TS)



https://github.com/awslabs/amazon-kinesis-video-streams-producer-sdk-cpp

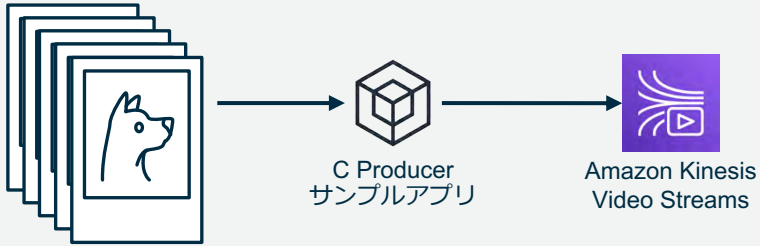
C Producer

KvsAacAudioVideoStreamingSample

H.264の映像フレーム、AAC のオーディオサンプルを読んで Amazon Kinesis Video Streams に送信

KvsVideoOnlyStreamingSample

上記の映像のみのサンプル



https://github.com/awslabs/amazon-kinesis-video-streams-producer-c

aws

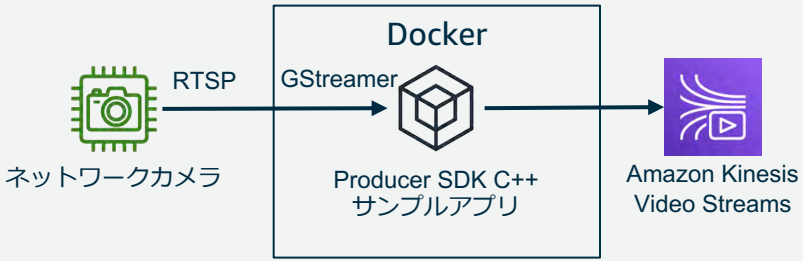
Amazon Kinesis Video Streams Demos リポジトリ

メディア形式で収集

各種 SDK をベースにしたユースケース別のサンプルやデモを掲載

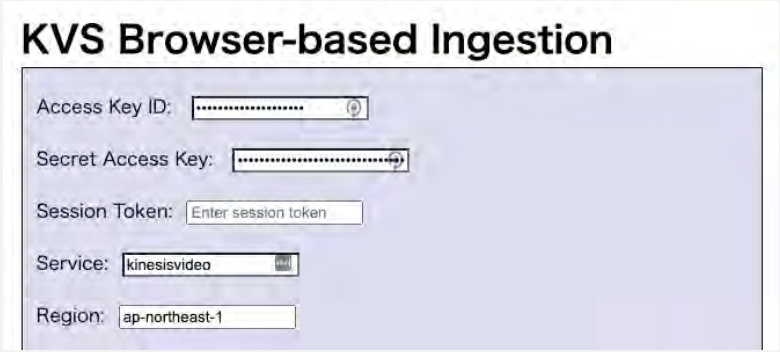
RTSP demo application

Producer SDK C++ を利用して、RTSP 接続可能なネットワークカメラから Amazon Kinesis Video Streams に動画を送信する Docker アプリケーション



Browser-based Ingestion

ウェブブラウザからWebカメラの映像を Amazon Kinesis Video Streams にアップロード (Chrome のみ)



https://github.com/aws-samples/amazon-kinesis-video-streams-demos

aws

本日のアジェンダ

メディア形式で収集

- Amazon Kinesis Video Streams の概要
- Amazon Kinesis Video Streams - メディア形式で収集
    1. メディアの取り込み
    2. メディアの保存とインデックス
    3. メディアの再生
    4. メディアの分析
- Amazon Kinesis Video Streams – WebRTC
- Amazon Kinesis Video Streams を活用した構成例
- サービス利用上のポイント
- まとめ

aws

メディアデータに関する主要なコンセプト

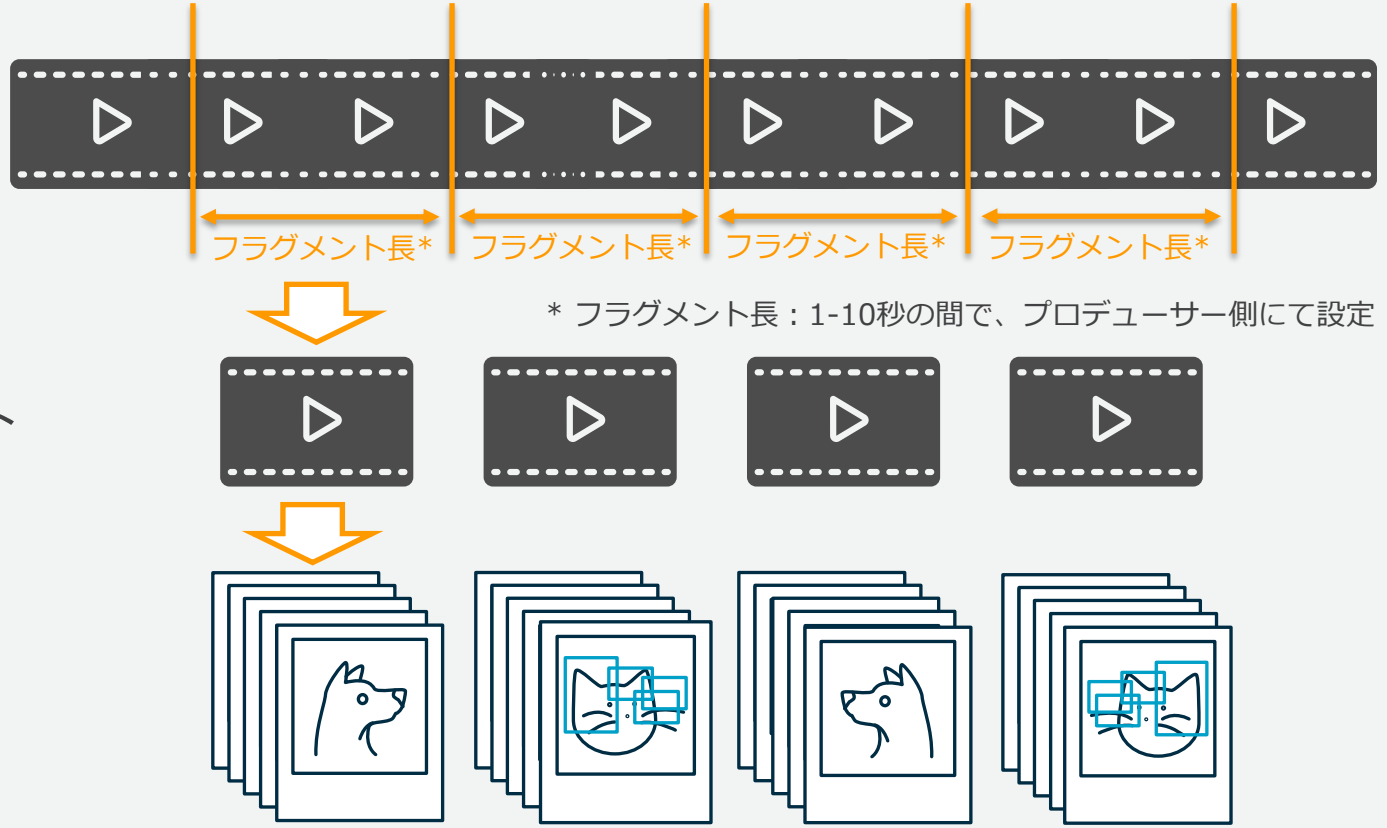
メディア形式で収集

コンセプト名	説明
フラグメント	短時間のフレームをまとめたシーケンスであり、固有の番号が割り当てられる。 フラグメントに属するフレームは、他のフラグメントのフレームに依存しない。
チャンク	ストリーム内でのデータの格納形式。 フラグメント、プロデューサーから送信されたメディアメタデータのコピー、 さらにはフラグメント番号、サーバー側とプロデューサー側のタイムスタンプ などの Amazon Kinesis Video Streams 固有のメタデータで構成される。
フレーム	動画のもとになる 1 コマの静止画像で、フラグメントに含まれる。 必要に応じてコンシューマー側でデコードして取り出し、画像解析等に利用する。

aws

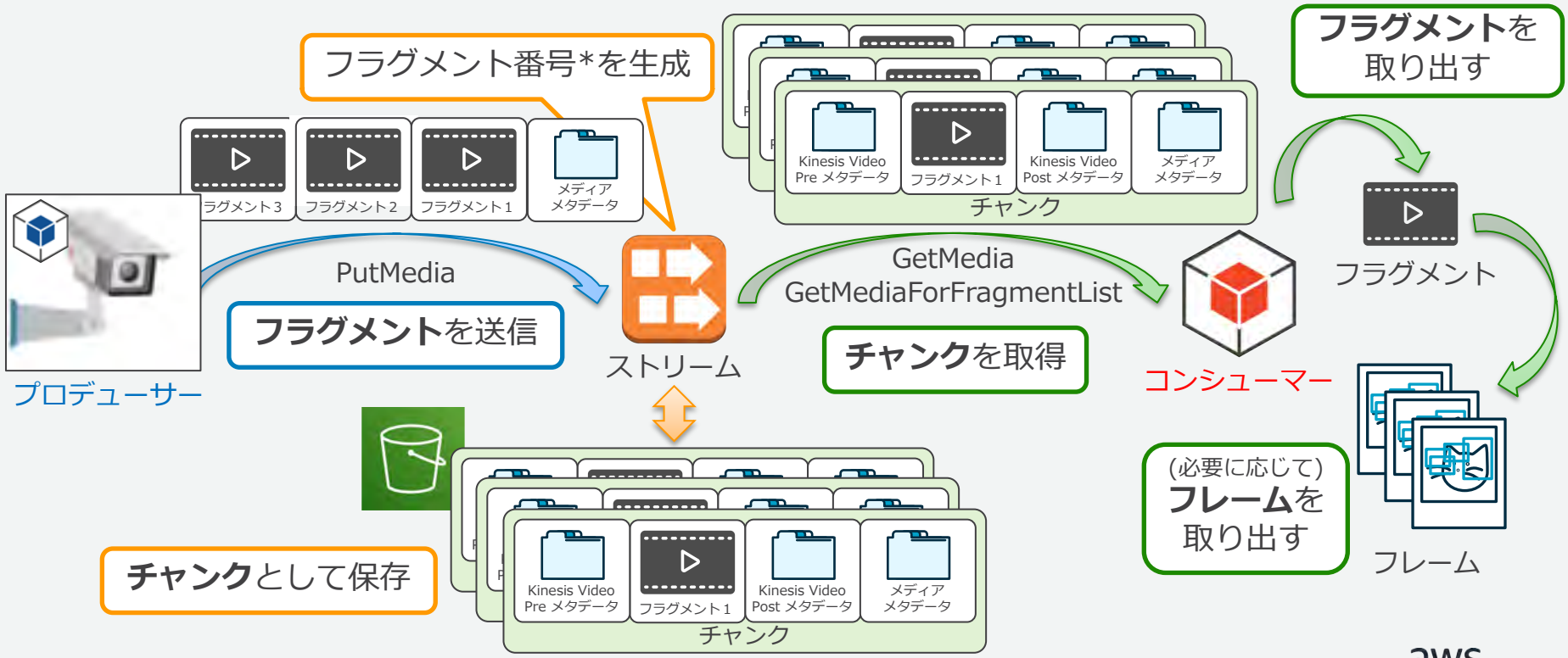
フラグメントとフレーム

メディア形式で収集



フラグメントとチャンク

メディア形式で収集

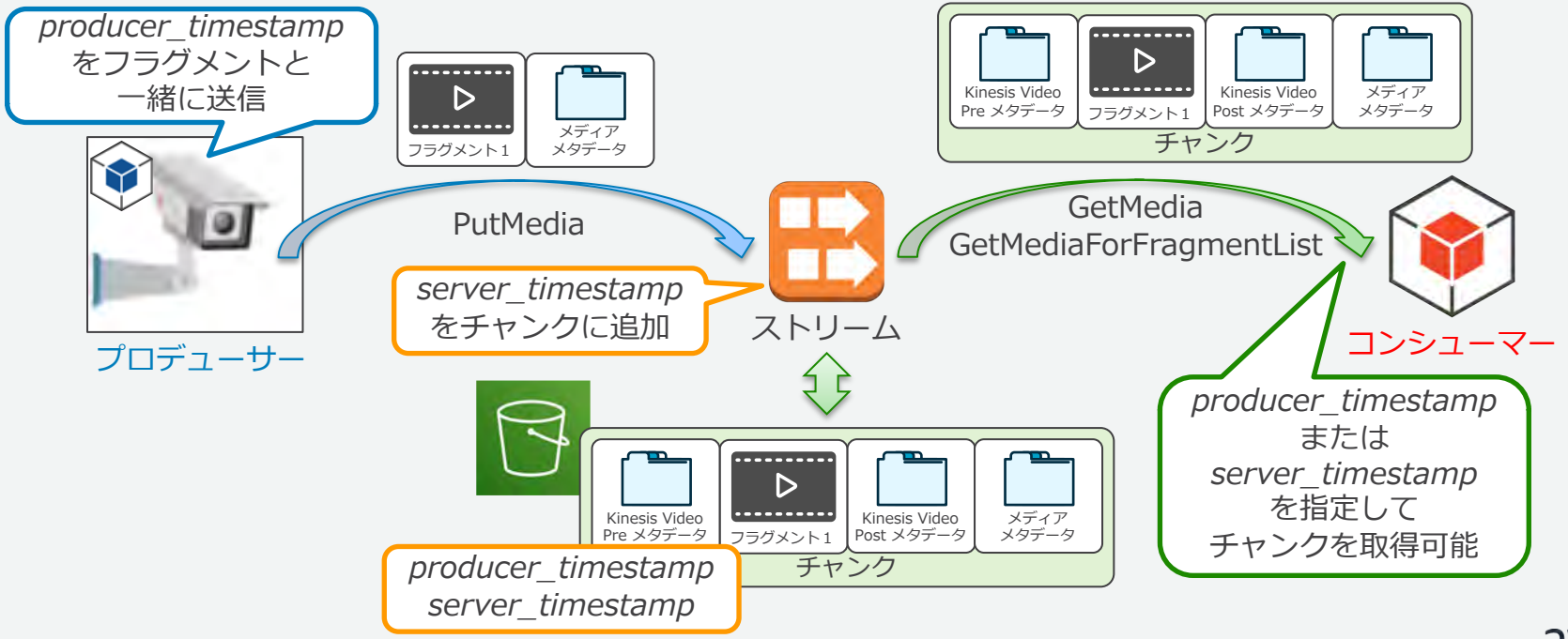


*フラグメント番号：64ビット整数で表される識別子で、Kinesis Video Streams が受信時に生成してフラグメントに割り当てる

タイムスタンプの扱い

メディア形式で収集

二種類のタイムスタンプでフラグメントが管理されている
producer_timestamp：プロデューサー側でそのフラグメントの記録を開始した時刻
server_timestamp　：Amazon Kinesis Video Streams がフラグメントの受信を開始した時刻



https://docs.aws.amazon.com/ja_jp/kinesisvideostreams/latest/dg/API_dataplane_PutMedia.html

通信のセキュリティとアクセス許可

メディア形式で収集

- SSLでの通信暗号化

  書き込みと読み込みのAPIを使用した通信は全てHTTPSで暗号化

- IAMでのアクセス管理

  AWSの認証メカニズムである Signature Version 4 を利用して、ストリーム単位、アクション単位でIAMでのアクセス制御が可能

  例：

```
"Resource": arn:aws:kinesisvideo:ap-northeast-1:111122223333:stream/my-stream-*
"Resource": arn:aws:kinesisvideo:*:111122223333:stream/my-stream-1/0123456789012

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

https://docs.aws.amazon.com/ja_jp/kinesisvideostreams/latest/dg/how-iam.html

aws

メディアの保存 - データの暗号化と保存期間

メディア形式で収集

■ データの暗号化

- 常にサーバーサイドの暗号化が有効になっている
- データはストレージレイヤーに書き込まれる前に暗号化され、ストレージから取得された後に復号される
- ストリームの作成時に、暗号化に使用する AWS KMS カスタマーマスターキー (CMK) を指定できる（あとから変更はできない）
- ストリームの作成時にユーザー指定のキーが指定されていない場合は、既定のキー (Kinesis Video Streams が提供) が使用される

■ データの保存期間

- ストリーム作成時にデータの保存期間を指定できる（あとから変更もできる）
- 設定可能な範囲は、最短は0（保存しない）で最長は10年（87600時間）
- データの保存量に応じてサービス利用料金が発生する

aws

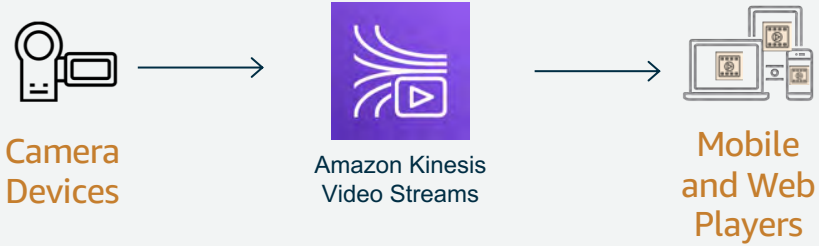
本日のアジェンダ

- Amazon Kinesis Video Streams の概要
- Amazon Kinesis Video Streams - メディア形式で収集
    1. メディアの取り込み
    2. メディアの保存とインデックス
    3. メディアの再生
    4. メディアの分析
- Amazon Kinesis Video Streams – WebRTC
- Amazon Kinesis Video Streams を活用した構成例
- サービス利用上のポイント
- まとめ

ライブ再生・オンデマンド再生

メディア形式で収集

- マネジメントコンソールのメディア再生ビューワー
- HTTP Live Streaming (HLS) や Dynamic Adaptive Streaming over HTTP (DASH) による、ライブもしくはオンデマンド再生



マネジメントコンソールの簡易ビューアー

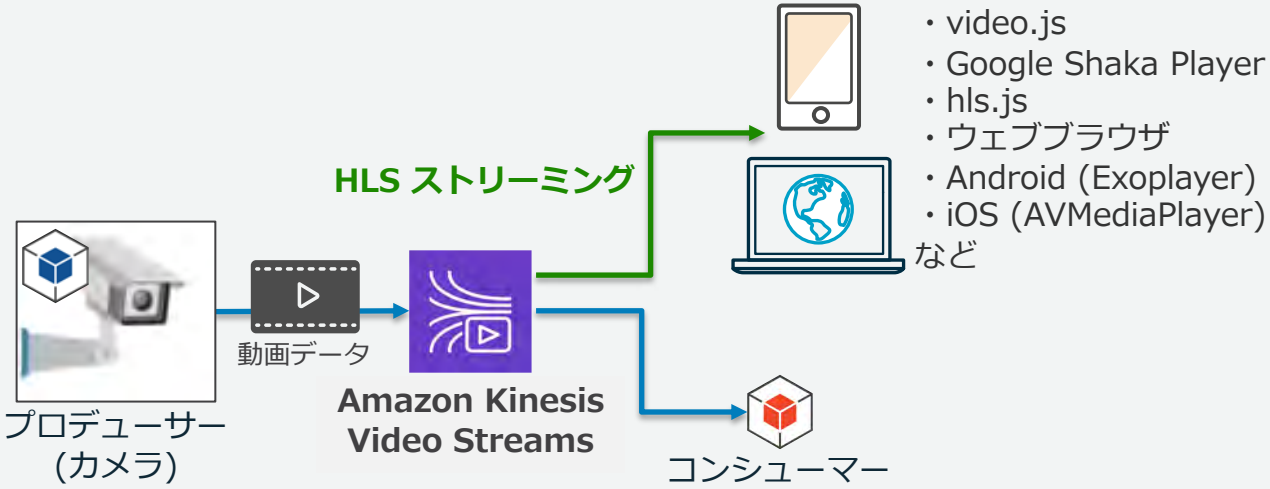
メディア形式で収集

- ストリームを選択して動画を再生可能
- 再生できるのはH.264のコーデックで圧縮された動画のみ
- 基本的には開発とテスト用途での使用を想定

HTTP ライブストリーミング (HLS) 再生機能

メディア形式で収集

ライブストリーミング や 動画ビューワー をコンシューマーの実装なしで実現



動作条件
- データの保持期間を0より大きく設定
- 映像は H.264 または H.265 でエンコード
- 音声がある場合には、AAC でエンコード
- コーデックIDなどのメタデータを正しく設定

HLS形式の再生機能を フルマネージドで提供

1. GetDataEndpoint API によってエンドポイントを取得
2. 得られたエンドポイントから GetHLSStreamingSession URL API によって HLS ストリーミング URL を取得
3. 得られたURLを指定して任意の再生ツールで動画を再生

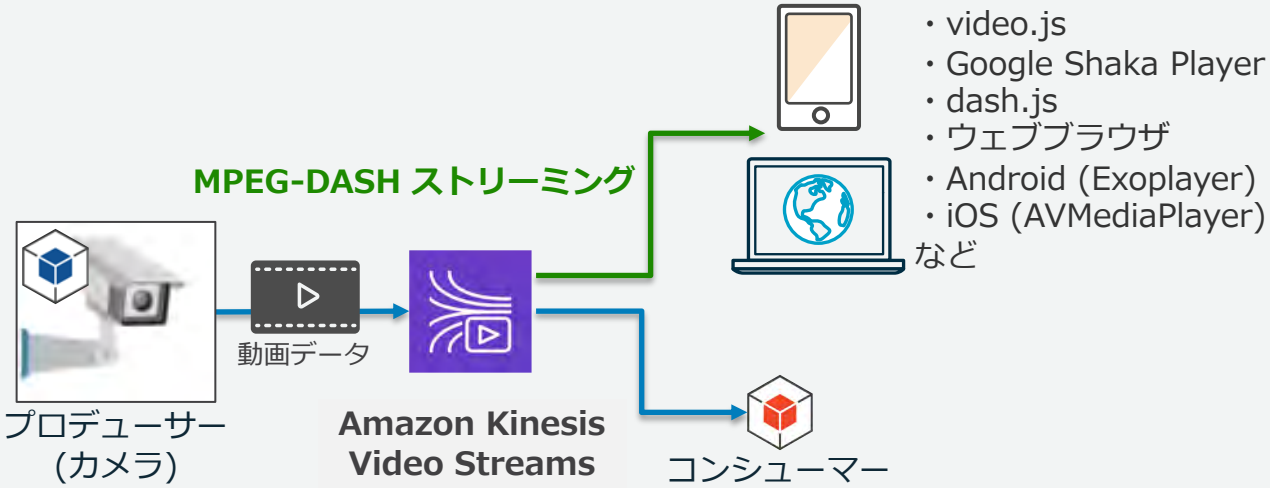
https://docs.aws.amazon.com/kinesisvideostreams/latest/dg/hls-playback.html

aws

MPEG-DASH 再生機能

メディア形式で収集

Dynamic Adaptive Streaming over HTTP (DASH) での再生にも対応



動作条件
- データの保持期間を0より大きく設定
- 映像は H.264 または H.265 でエンコード
- 音声がある場合には、AAC または G.711 でエンコード
- コーデックIDなどのメタデータを正しく設定

DASH形式の再生機能をフルマネージドで提供

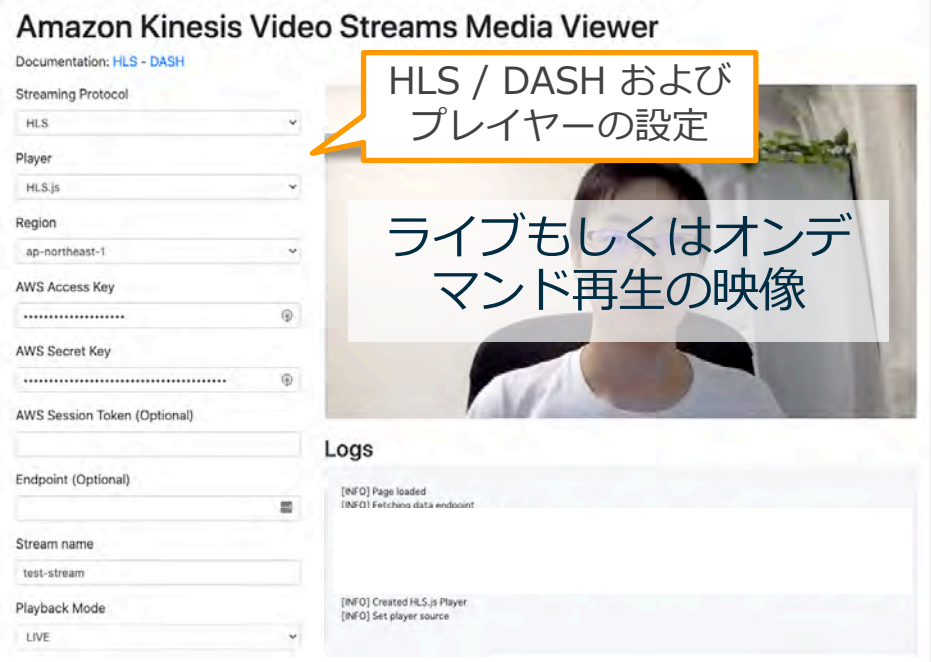
1. GetDataEndpoint API によってエンドポイントを取得
2. 得られたエンドポイントから GetDASHStreamingSession URL API によって MPEG-DASH ストリーミング URL を取得
3. 得られたURLを指定して任意の再生ツールで動画を再生

https://docs.aws.amazon.com/kinesisvideostreams/latest/dg/dash-playback.html

Amazon Kinesis Video Streams Media Viewer

メディア形式で収集

- HLS もしくは DASH を利用したメディア再生のサンプルページ
- HLS もしくは DASH のストリーミングセッションURLを取得して、HLS.js や Video.js などのライブラリを利用して再生できる



アクセスキーなどの認証情報

ストリーミングセッション取得時の設定

https://aws-samples.github.io/amazon-kinesis-video-streams-media-viewer/

HLS や DASH での再生に関する注意点

- 同時に再生できるセッション数の制限
  - ストリームあたりのアクティブな HLS および DASH のストリーミングセッション数: 10 (ソフトリミット)
  - 上限を超えた場合は、最も古いセッションが廃止される

- ライブ再生時のレイテンシー
  - 必ずテストを行って評価する
  - 動画フラグメントのサイズ、プロデューサの設定、プレイヤーのチューニング、ネットワークなどの条件に左右される

https://docs.aws.amazon.com/kinesisvideostreams/latest/dg/limits.html
https://aws.amazon.com/jp/kinesis/video-streams/faqs/

レイテンシーの考え方

メディア形式で収集

End-to-End のメディアデータフローにおける、レイテンシーの重要な要因

デバイスのメディアパイプライン	イメージセンサーからのデータ読み取り、(ハードウェア/ソフトウェア)エンコーダーでのメディア変換処理
インターネットのデータ伝送	デバイスからクラウドまでのネットワークスループットやレイテンシ
Amazon Kinesis Video Streams のデータ受信時のレイテンシー	データ保持設定時のデータの暗号化と時間ベースのインデックス生成
Amazon Kinesis Video Streams からコンシューマーの間のレイテンシー	HLS 再生時の動画変換の内部処理、クラウドとコンシューマの間のネットワーク環境、プレイヤーの設定(バッファなど)

タイムスタンプの計測や、プロデューサー・コンシューマーの設定、ネットワーク環境の変更などによってレイテンシーの要因を切り分けることで、レイテンシーを短縮するためのアクションが可能

https://aws.amazon.com/jp/kinesis/video-streams/faqs/

aws

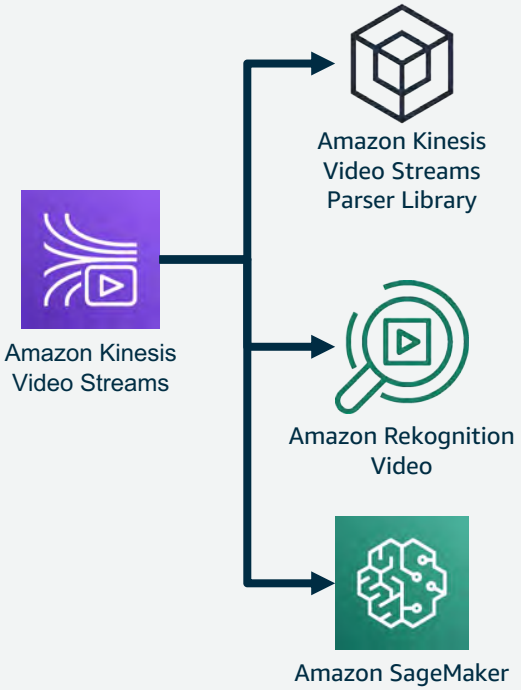
本日のアジェンダ

- Amazon Kinesis Video Streams の概要
- Amazon Kinesis Video Streams - メディア形式で収集
    1. メディアの取り込み
    2. メディアの保存とインデックス
    3. メディアの再生
    4. メディアの分析
- Amazon Kinesis Video Streams – WebRTC
- Amazon Kinesis Video Streams を活用した構成例
- サービス利用上のポイント
- まとめ

メディアの取得・分析

メディア形式で収集

- GetClip API での MP4クリップの出力
- GetMedia API でのチャンクの取得
- Amazon Kinesis Video Streams Parser Library によるフラグメントやフレームの取り扱い
- Amazon Rekognition Video を利用して、フルマネージドでMLベースの処理を実行
- Amazon SageMaker で Amazon Kinesis Video Inference Template を利用



https://docs.aws.amazon.com/kinesisvideostreams/latest/dg/examples-sagemaker.html

aws

GetClip API による MP4クリップの出力

メディア形式で収集

- マネジメントコンソールもしくは GetClip API を利用して、MP4 クリップを生成してダウンロード可能
- サポートするコーデック
  - Video: H.264 or H.265
  - Audio: AAC or G.711
- 1リクエストで出力できるのは、100MB もしくは 200 フラグメントまで

マネジメントコンソールからも利用可能

https://docs.aws.amazon.com/ja_jp/kinesisvideostreams/latest/dg/API_reader_GetClip.html

コンシューマー側のAPI

メディア形式で収集

リアルタイム
処理指向

GetMedia API

Kinesis Video Streams のストリームから、メディアデータを取得するためのAPI

- ストリーム名またはストリーム Amazon リソースネーム (ARN) と、開始チャンクをリクエストする
- Kinesis Video Streams はフラグメント番号順に**チャンク**のストリームを返す

バッチ
処理指向

GetMediaForFragmentList API

ストリームに保存されたアーカイブデータから指定したメディアデータを取得するAPI

- フラグメントのリスト (フラグメント番号で指定)を指定してリクエストする
- 通常は、この API を呼び出す前に、ListFragments API を呼び出す

ListFragments API

フラグメントのリストを取得するAPI

- フラグメント番号またはタイムスタンプを使用して、ストリームに対して開始位置を指定してリクエストする

aws

GetMedia API

メディア形式で収集

PutMediaで送信されたフラグメントを含むチャンクのストリームを Payloadとして受信する

- リクエストパラメータは JSON形式でBodyの中に入れて送信する
- GetMedia API は Long Running Sessionのため、セッションを張って、そのセッションの中でチャンクを受信する

Request

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

{
      "StartSelector": {
            "AfterFragmentNumber": "string",
            "ContinuationToken": "string",
            "StartSelectorType": "string",
            "StartTimestamp": number
      },
      "StreamARN": "string",
      "StreamName": "string"
}
```

Response

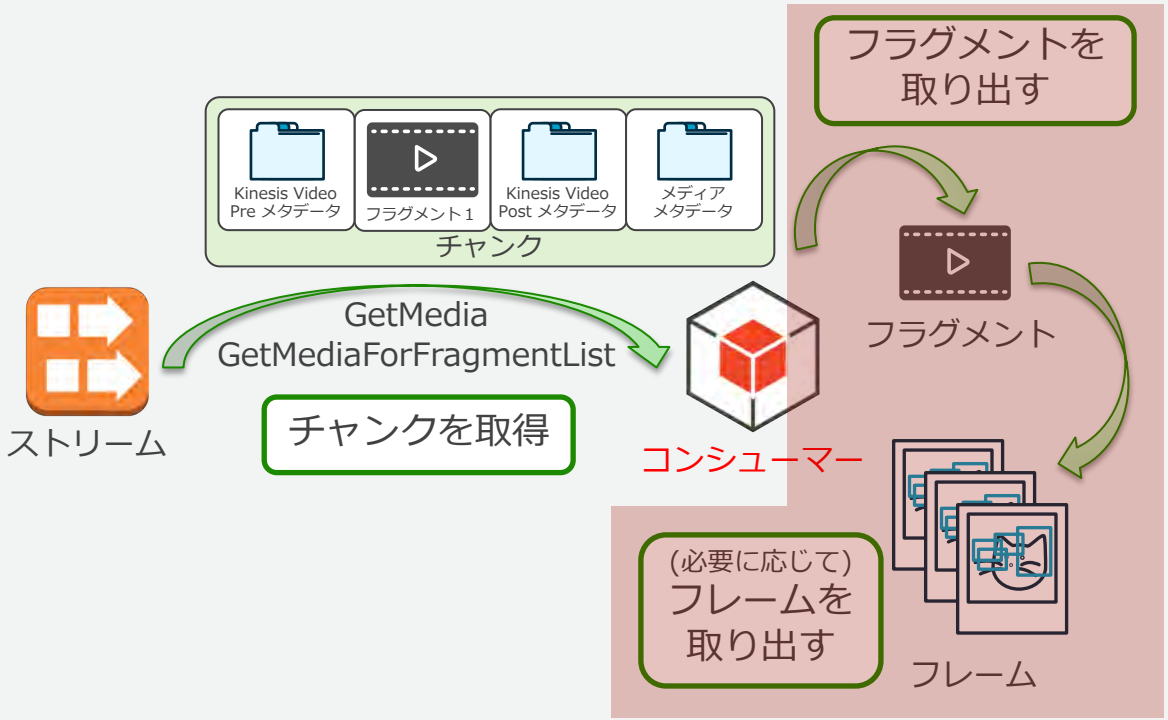
```
HTTP/1.1 200
Content-Type: ContentType

Payload
```


https://docs.aws.amazon.com/ja_jp/kinesisvideostreams/latest/dg/API_dataplane_GetMedia.html

GetMedia API で取得できるのは、あくまでMKV形式のチャンクでしかない

メディア形式で収集



Amazon Kinesis Video Streams Parser Library

メディア形式で収集

ストリームから取得したMKV形式のデータを使いやすい形に加工するためのライブラリ

- Javaのライブラリ（Jarファイル）として提供され、
  AWS SDK for Java と組み合わせてJavaアプリケーション内で使用する
- 現在の実装では、主なツールとして以下のものが含まれるほか、サンプルアプリケーションを提供

主なツール	内容
StreamingMkvReader	MKVデータをストリームから MKV Element として読み取る
FragmentMetadataVisitor	フラグメント（メディアデータ）およびトラック（コーデックの種類やピクセル幅・高さなどを含むデータ）からメタデータを取得する。 フラグメント番号やタイムスタンプの情報もここから取得出来る。
OutputSegmentMerger	異なるトラックのメタデータを単一のセグメントを持つストリームにマージして、連続したフラグメント（チャンク）を結合する
FrameVisitor	フラグメントからフレームを取り出してフレーム毎の処理に渡す。 インタフェースが提供されるので、デコードと処理内容は自分で実装する。

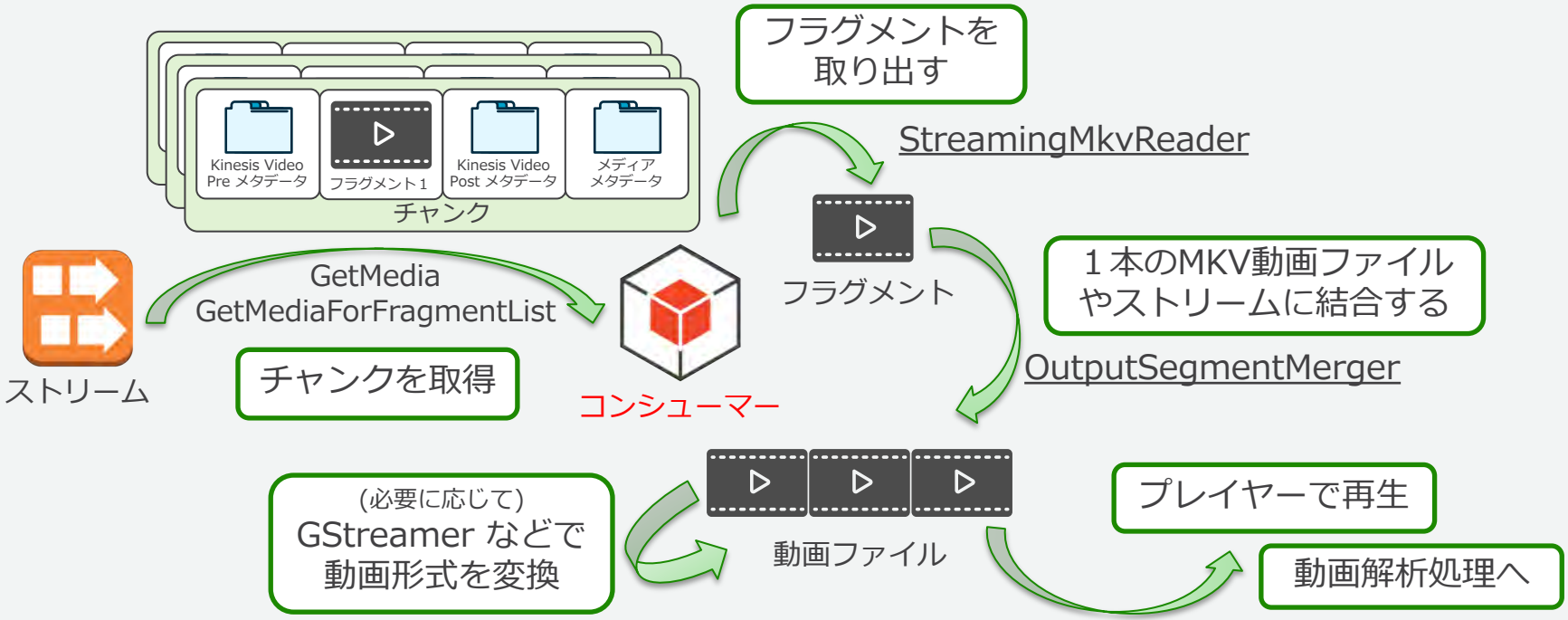
https://github.com/aws/amazon-kinesis-video-streams-parser-library

aws

動画ファイルを出力して解析する

メディア形式で収集

Stream Parser Library の OutputSegmentMerger を活用する

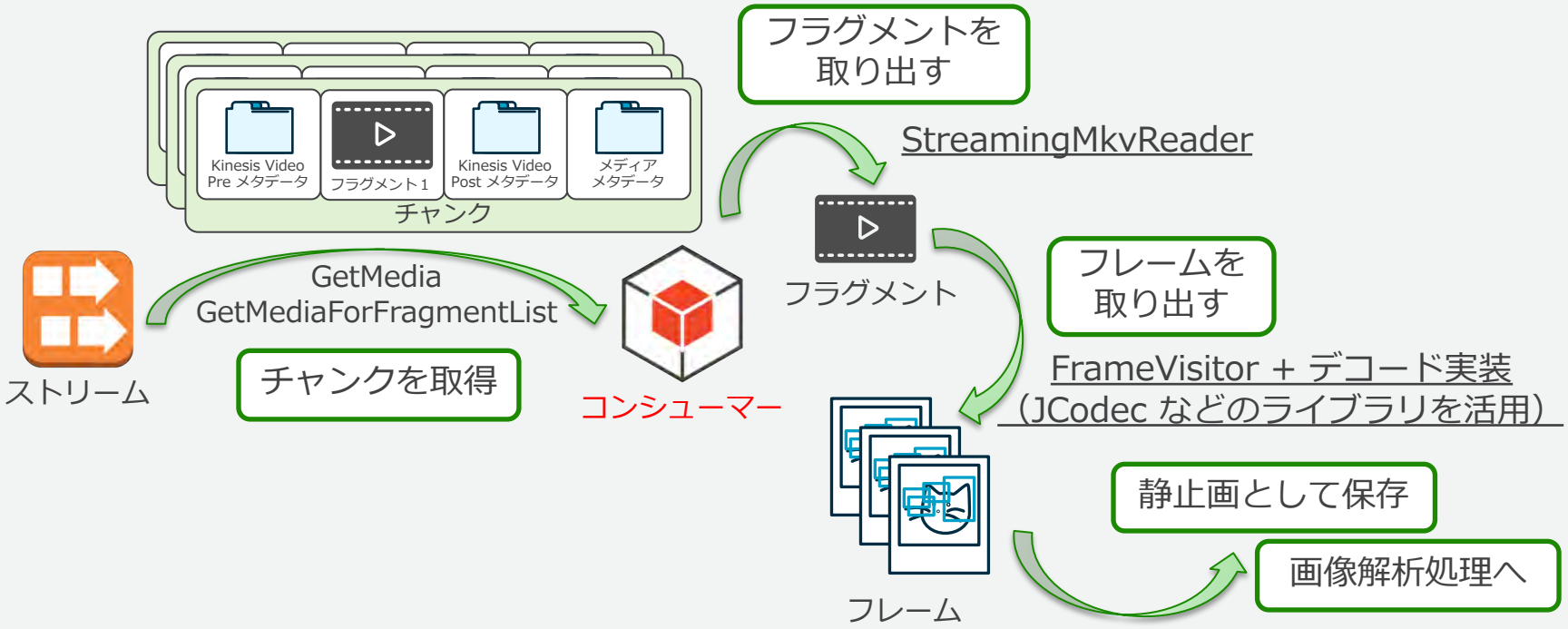


aws

画像フレームを取り出して解析する

メディア形式で収集

Stream Parser Library の FrameVisitor を活用する

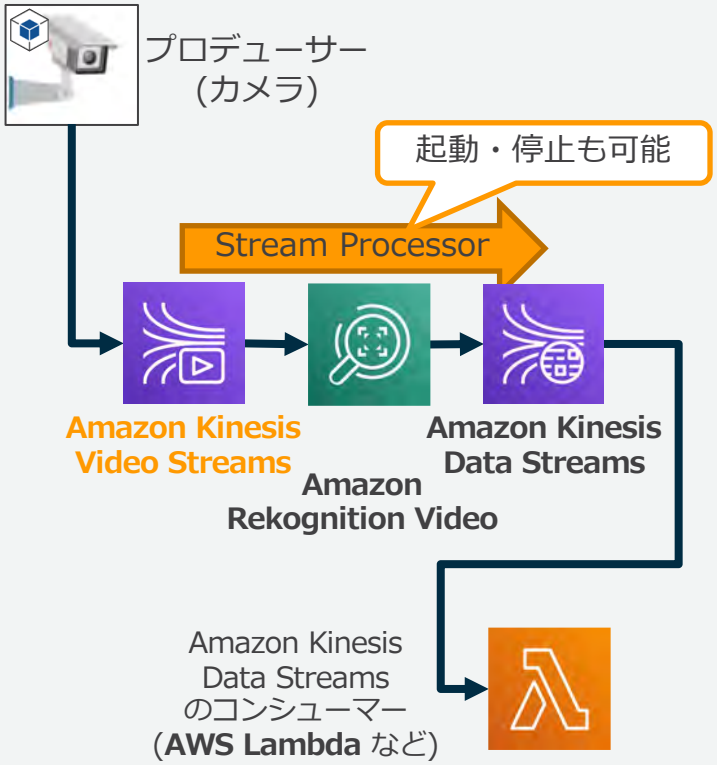


Amazon Rekognition Video と連携する

メディア形式で収集

- Kinesis Video Streams のコンシューマーとして Amazon Rekognition Video を使用して動画ストリーム内の顔を検出・認識できる
- 予め Rekognition Video に覚えさせておいた既知の顔を検出することができる

- Rekognition Video には、動画ストリームの分析を管理するための Stream Processor が用意されている
- 分析結果は、Rekognition Video から、Amazon Kinesis Data Streams のストリームに出力される

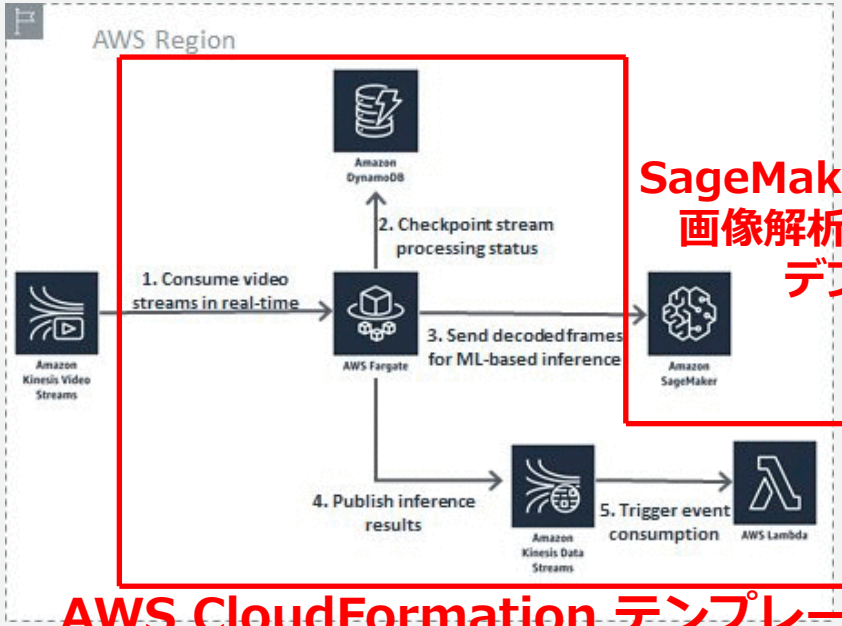


https://docs.aws.amazon.com/rekognition/latest/dg/streaming-video.html

Amazon SageMakerと連携するための Amazon Kinesis Video Streams Inference Template (KIT)

メディア形式で収集

コンシューマーを実装せずに Kinesis Video Streams のストリームを SageMaker の推論エンドポイントと接続することが可能



SageMakerで学習した画像解析のモデルをデプロイ

AWS CloudFormation テンプレート

SageMakerとの連携機能を完全なテンプレートで提供

1. SageMakerで学習した画像解析のモデルを使用して推論エンドポイントを作成
2. 作成した推論エンドポイントとストリームを指定して CloudFormation を起動
3. ストリームに入力された動画の認識結果が Kinesis Data Streams に出力される

https://aws.amazon.com/jp/blogs/news/analyze-live-video-at-scale-in-real-time-using-amazon-kinesis-video-streams-and-amazon-sagemaker/

料金構成

メディア形式で収集

2020/09/30 現在、東京リージョンの場合

料金構成	料金
（プロデューサーから） 取り込まれたデータ量	$0.01097 / 1GB
（コンシューマーに） 取り出されたデータ量 (※)	$0.01097 / 1GB
（HLS/DASH再生機能で） 取り出されたデータ量 (※)	$0.01536 / 1GB
保存されたデータ量	$0.02500 / 1GB・月

※ インターネット経由でビデオストリームから AWS 外の送信先にデータを送信する場合は、
AWS の標準のデータ転送料金がかかります。
https://aws.amazon.com/jp/kinesis/video-streams/pricing/

aws

本日のアジェンダ

- Amazon Kinesis Video Streams の概要
- Amazon Kinesis Video Streams - メディア形式で収集
    1. メディアの取り込み
    2. メディアの保存とインデックス
    3. メディアの再生
    4. メディアの分析
- Amazon Kinesis Video Streams – WebRTC
- Amazon Kinesis Video Streams を活用した構成例
- サービス利用上のポイント
- まとめ

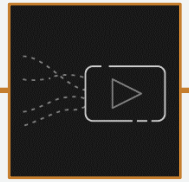
aws

Amazon Kinesis Video Streams WebRTC

WebRTC

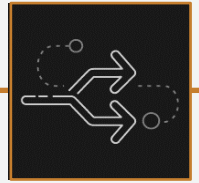
超低遅延でのメディアストリーミングと
100万台規模のデバイス間での双方向の通信をサポート

低遅延のライブメディア ストリーミング



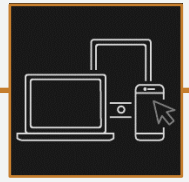
Peer-to-peer の音声・映像のライブストリーミングと、1秒未満のレイテンシでの再生

リアルタイム 双方向通信



デバイス・モバイル・ウェブアプリ間での、音声・映像・データのリアルタイム双方向通信

標準規格準拠



ウェブやモバイルのプラットフォームから容易に利用可能

フルマネージド

フルマネージドなWebRTCシグナリング・TURN・STUNのサービスとSDK

aws

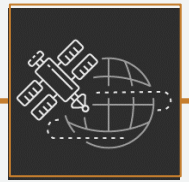
WebRTC とは？

WebRTC は単なるメディアストリーミングのプロトコルではなく、以下のような特徴を持ったリアルタイムコミュニケーションのためのオープンな標準規格

シグナリング

接続のメタデータの交換

相互通信



Peer-to-peer接続の確立

メディア配信

低遅延でのメディアと任意のデータの双方向通信

暗号化

End-to-endでの暗号化

Amazon Kinesis Video Streams WebRTC における主要な用語・概念

WebRTC

シグナリングチャネル	アプリケーションが、P2P接続を検出、開始、制御、終了するためにシグナリングメッセージをやりとりするためのチャンネル。セキュアな Websocket (WSS) が使用されている。
Peer	WebRTCの通信を行う任意のデバイス・アプリケーション 例：ウェブブラウザ、スマートフォンアプリ、ホームセキュリティカメラ
Master	接続の初期化を行い、シグナリングチャンネルで複数のViewerから接続、メディアの送受信を行うPeer 2020年9月現在、1シグナリングチャンネルには1 Masterまで
Viewer	1つのMasterのみとメディアの送受信を行えるPeer、他のViewerと接続することはできない 2020年9月現在、1シグナリングチャンネルには10 Viewerまで(ソフトリミット)

→ Master と Viewer は 1:N の関係

aws

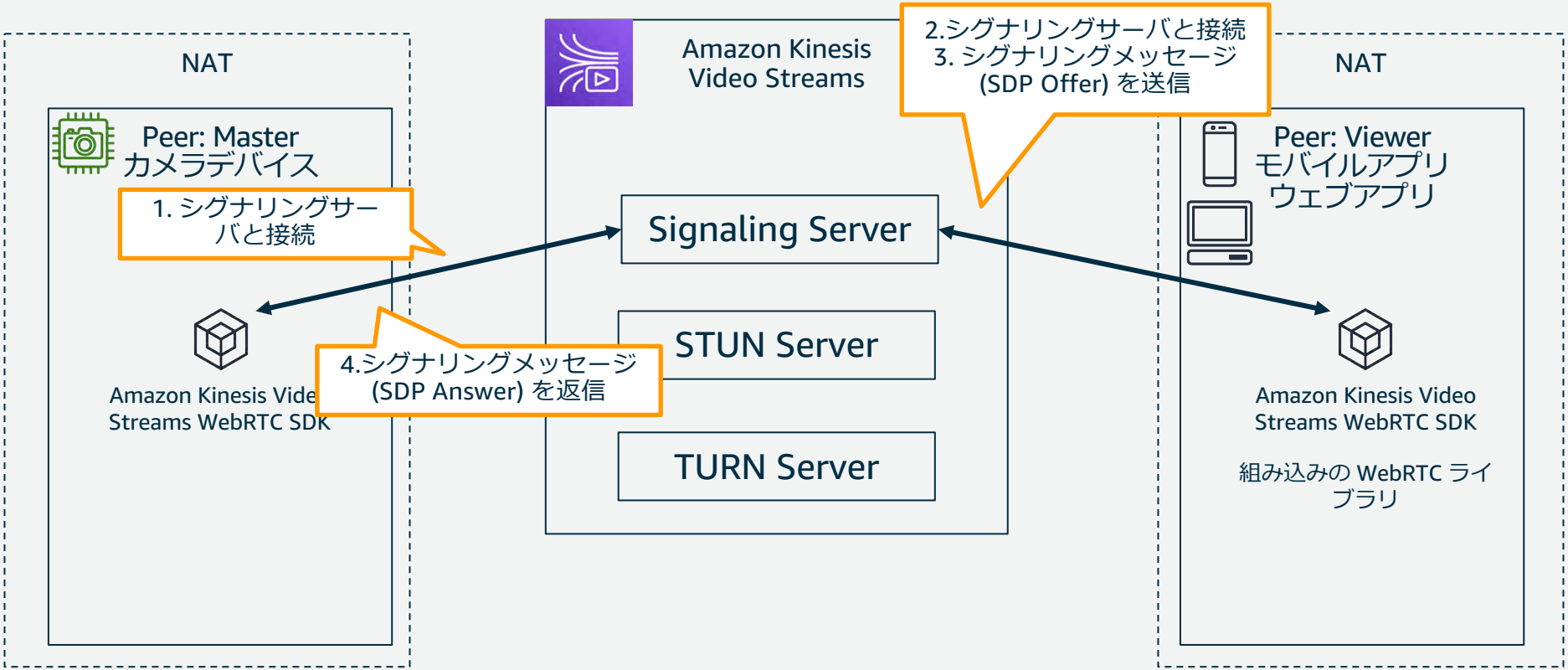
WebRTC の接続フロー

WebRTC

aws

WebRTC の接続フロー

WebRTC

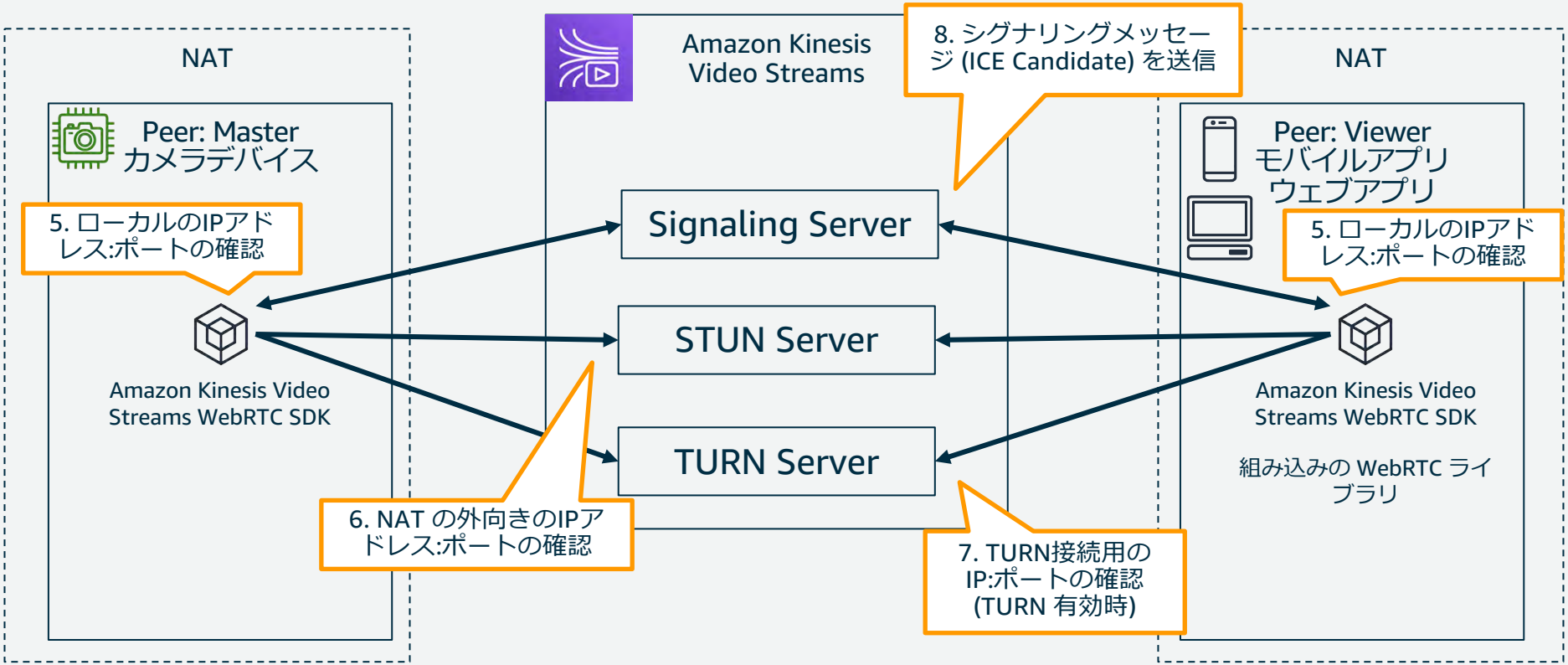


※ 接続フローのうち主要な通信のみを記載、順序は入れ替わる場合もあり

aws

WebRTC の接続フロー

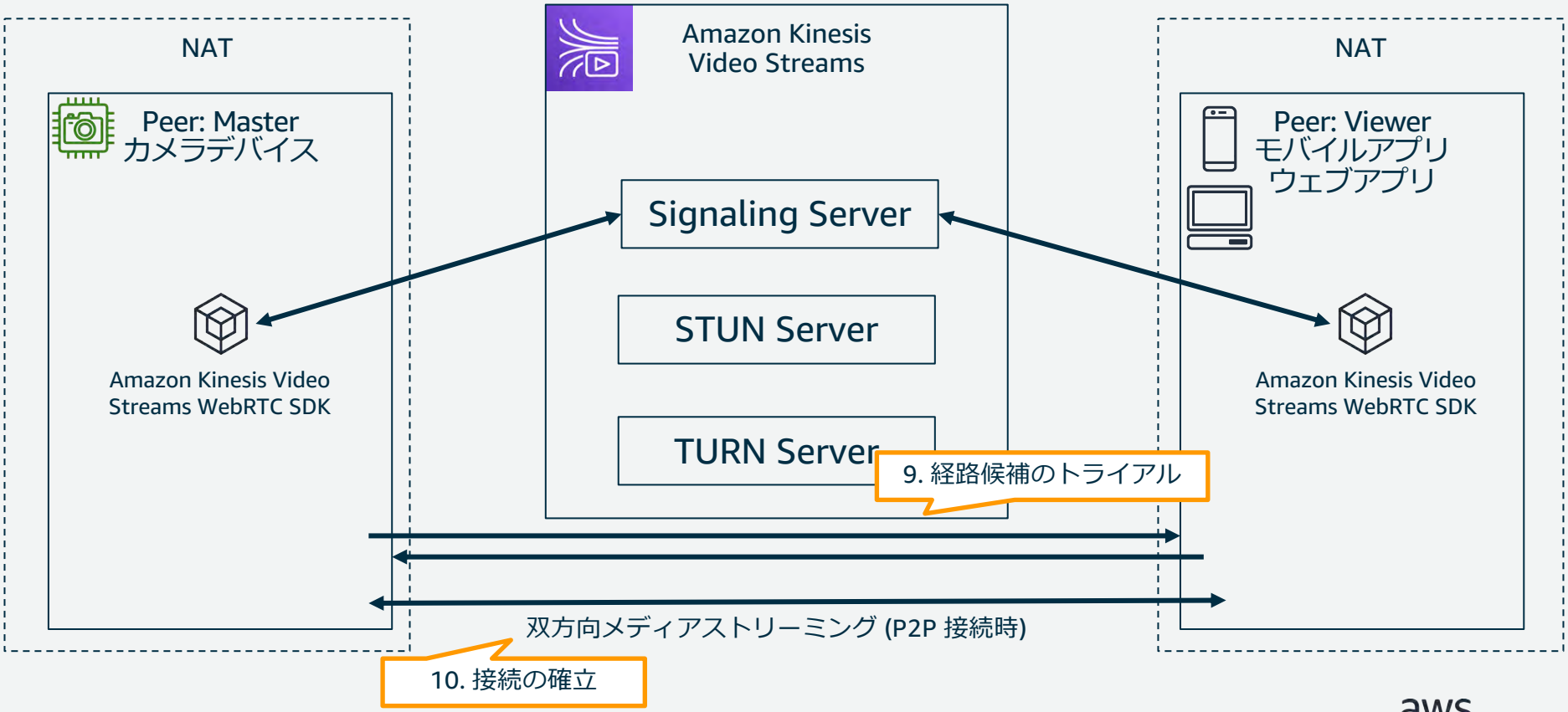
WebRTC



※ 接続フローのうち主要な通信のみを記載、順序は入れ替わる場合もあり

WebRTC の接続フロー

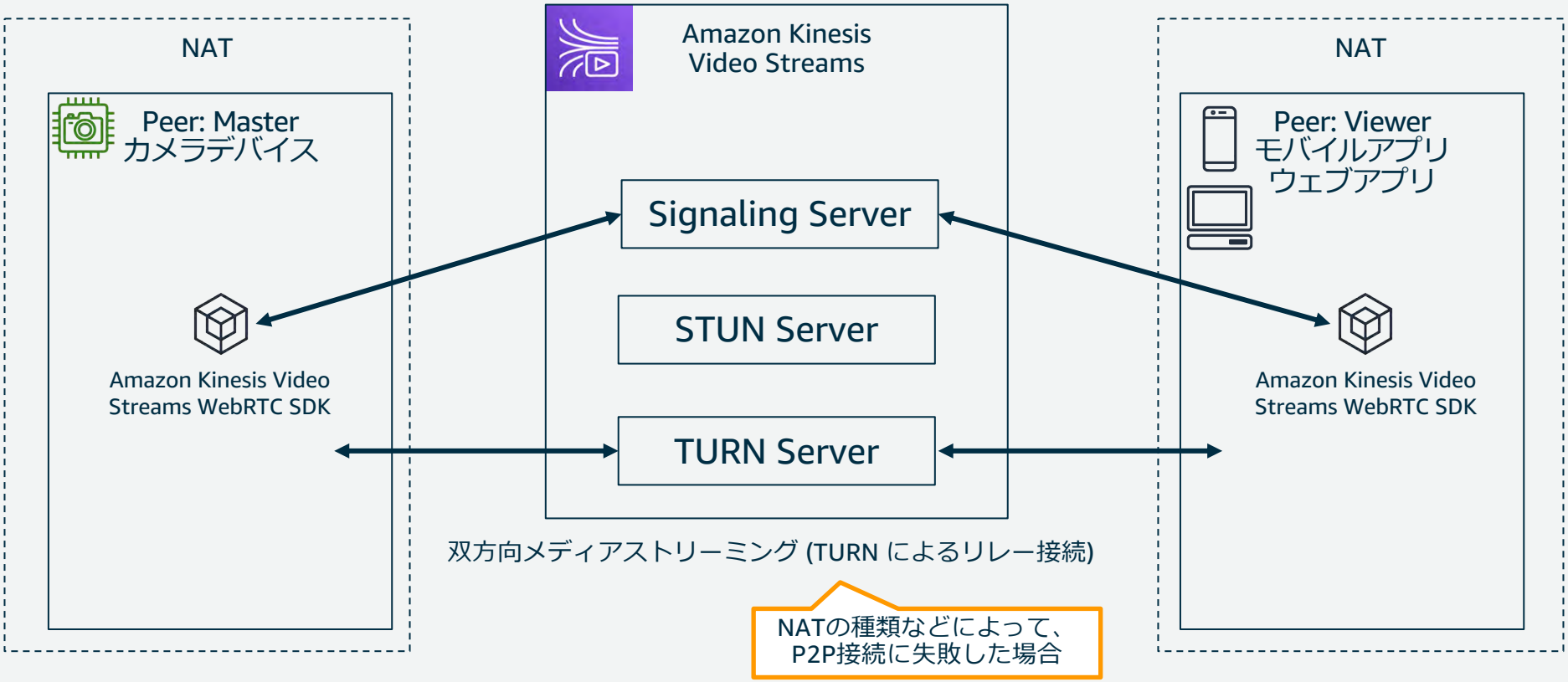
WebRTC



aws

WebRTC の接続フロー

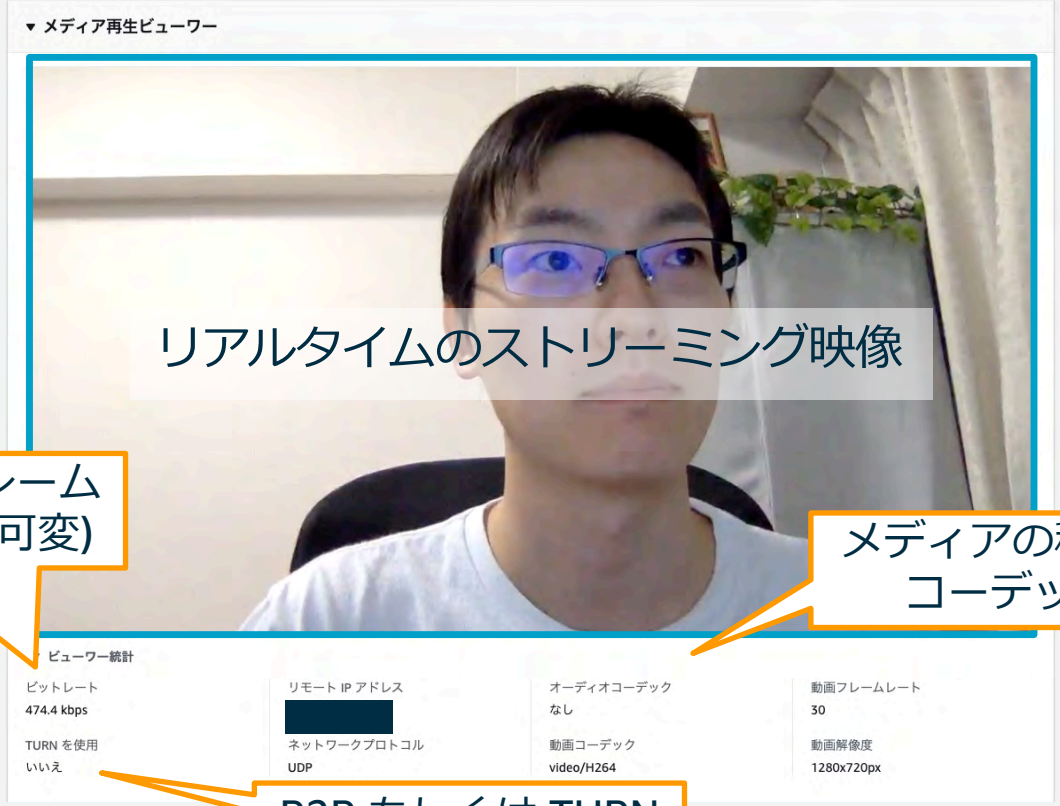
WebRTC



マネジメントコンソールのメディア再生ビューワー

WebRTC

開発・テスト用途に利用できる簡易ビューワー



ビットレートやフレームレートは現在の値(可変)

メディアの種類・コーデック

P2P もしくは TURN

aws

WebRTC SDK の構成

WebRTC

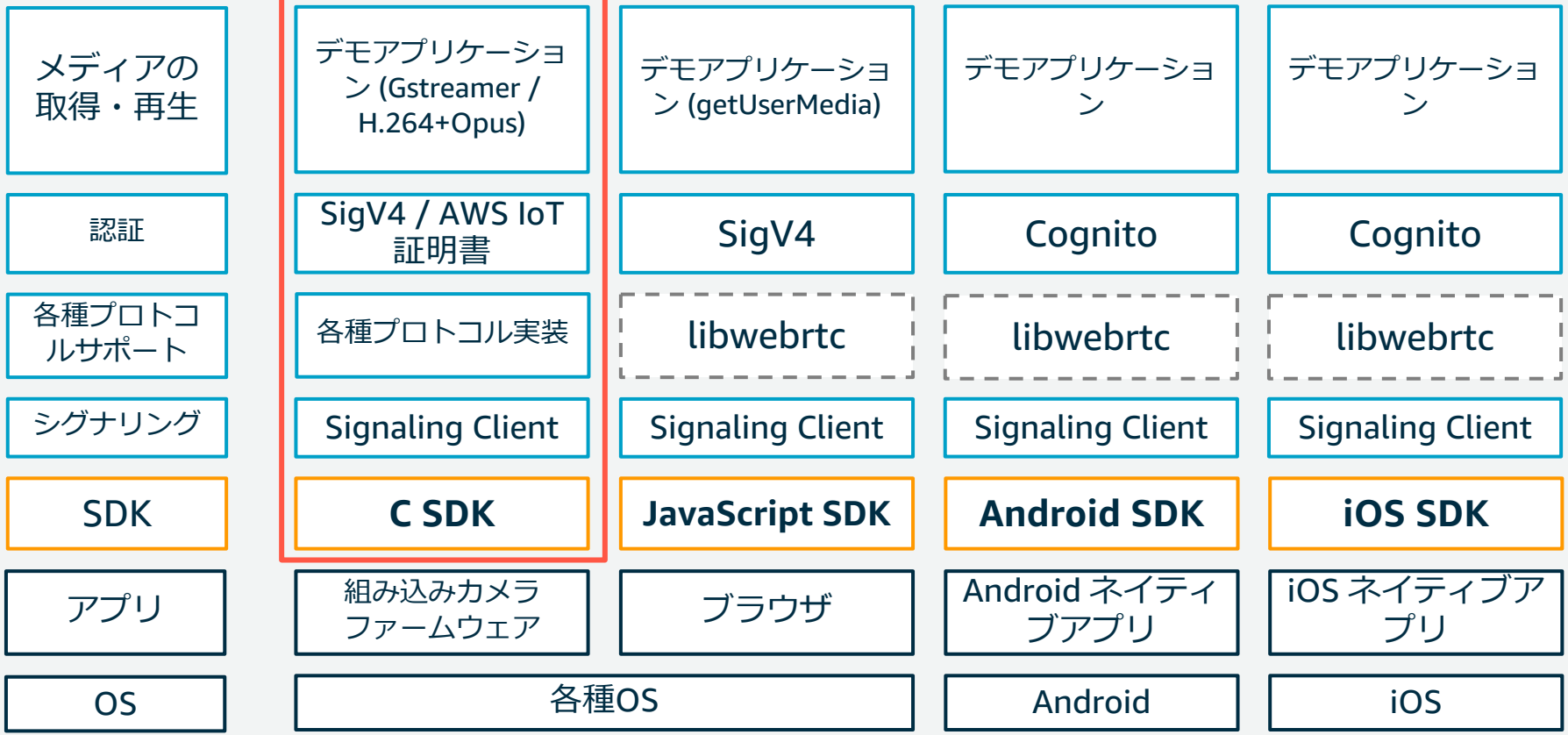
メディアの取得・再生	デモアプリケーション (Gstreamer / H.264+Opus)	デモアプリケーション (getUserMedia)	デモアプリケーション	デモアプリケーション
認証	SigV4 / AWS IoT 証明書	SigV4	Cognito	Cognito
各種プロトコルサポート	各種プロトコル実装	libwebrtc	libwebrtc	libwebrtc
シグナリング	Signaling Client	Signaling Client	Signaling Client	Signaling Client
SDK	**C SDK**	**JavaScript SDK**	**Android SDK**	**iOS SDK**
アプリ	組み込みカメラファームウェア	ブラウザ	Android ネイティブアプリ	iOS ネイティブアプリ
OS	各種OS		Android	iOS

aws

WebRTC SDK の構成

組み込みカメラデバイス向けの
C言語の軽量なSDK

WebRTC

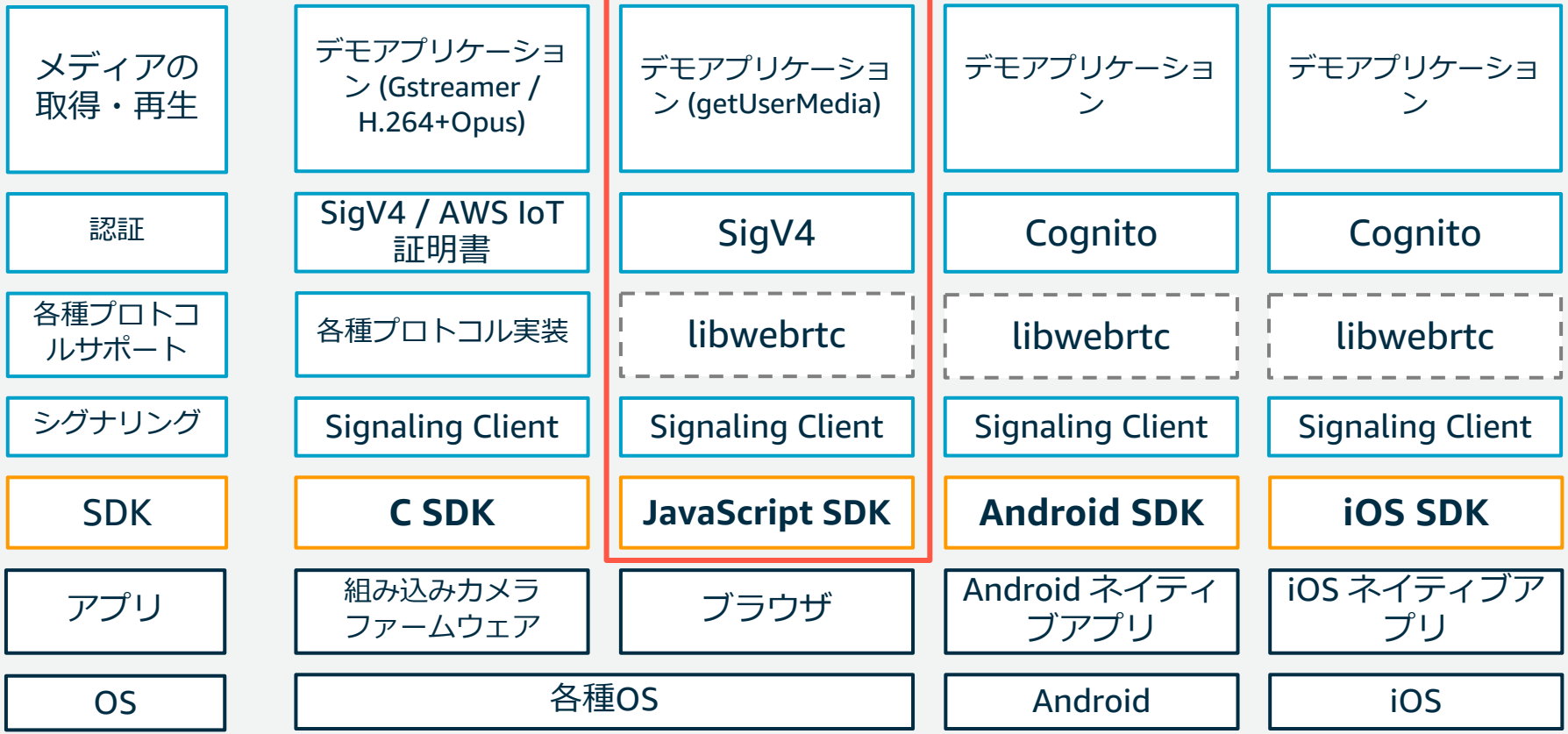


aws

WebRTC SDK の構成

ウェブアプリ開発用のSDK

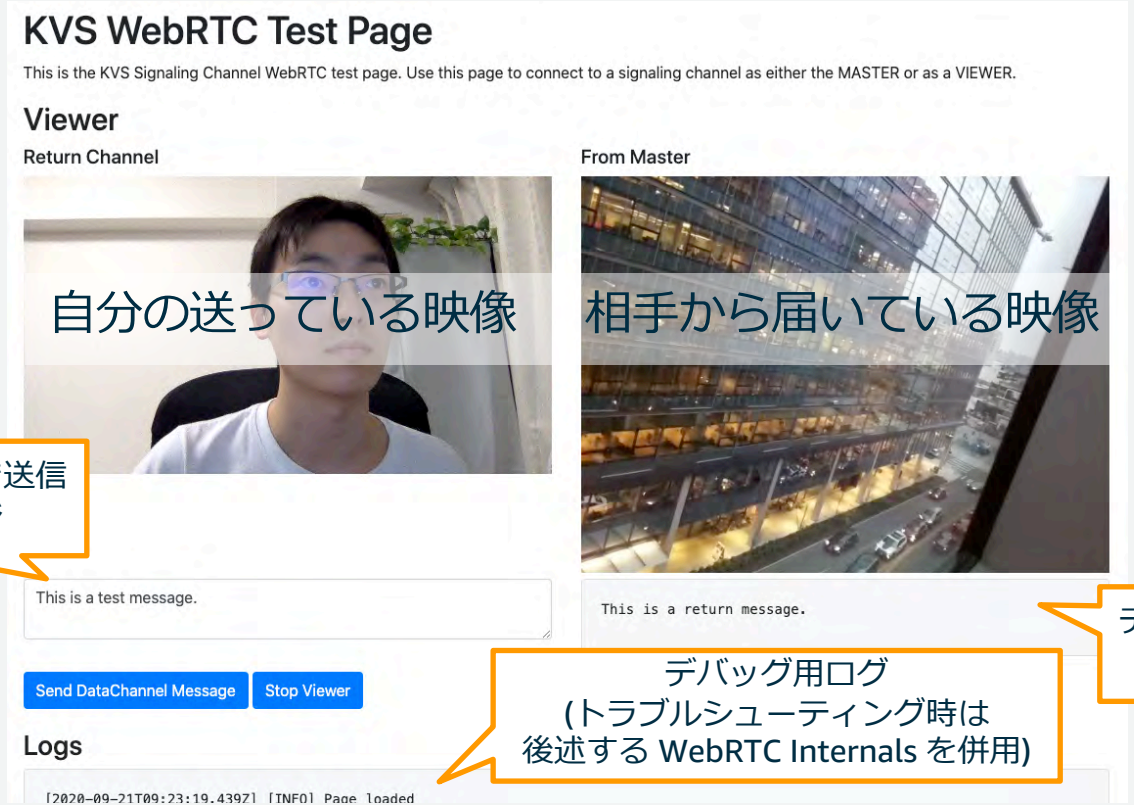
WebRTC



aws

WebRTC SDK JavaScript のサンプルページ (Viewer の場合)

WebRTC



データチャンネルで送信するメッセージ

データチャンネルで受信したメッセージ

デバッグ用ログ
(トラブルシューティング時は後述する WebRTC Internals を併用)

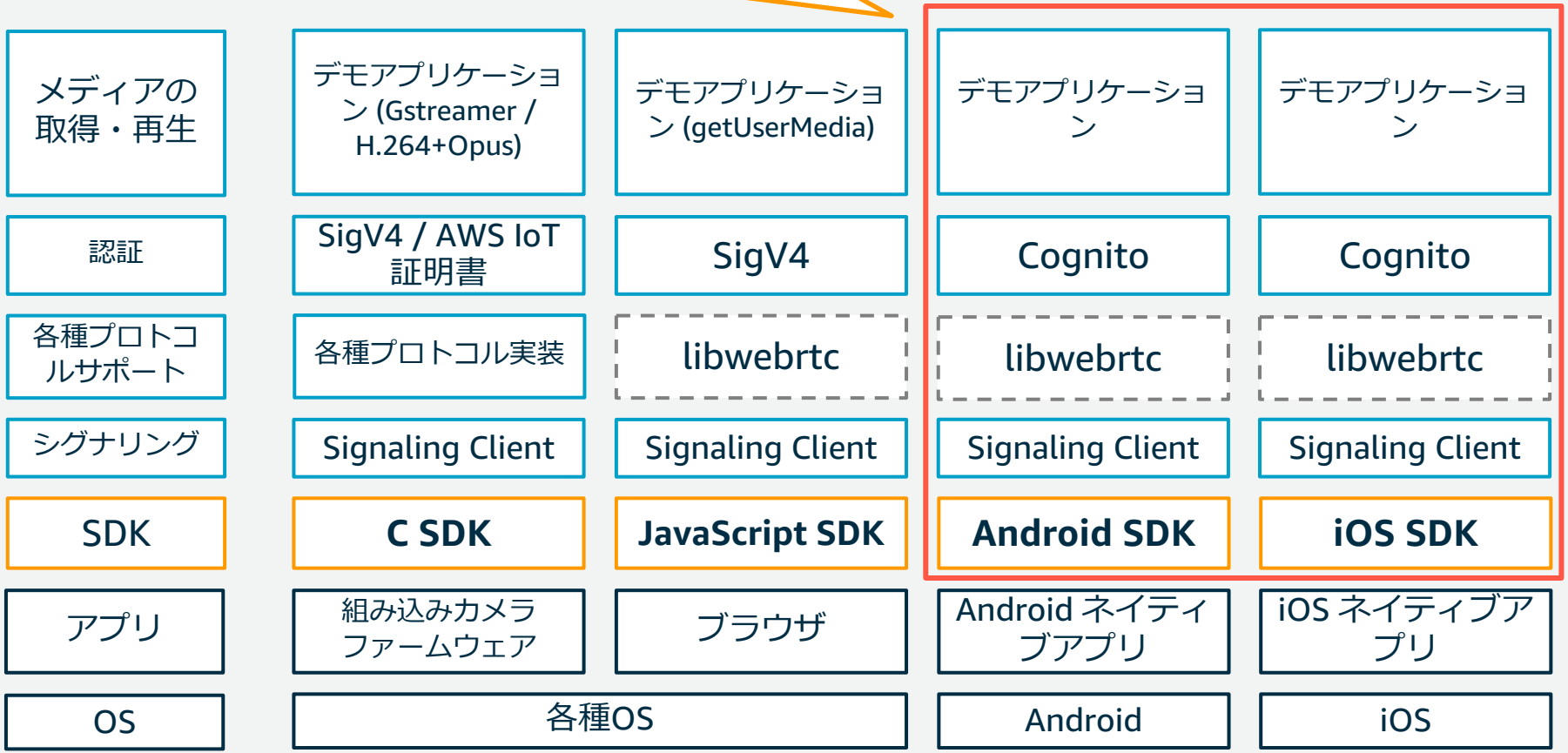
https://awslabs.github.io/amazon-kinesis-video-streams-webrtc-sdk-js/examples/index.html

aws

WebRTC SDK の構成

モバイルアプリ開発用のSDK

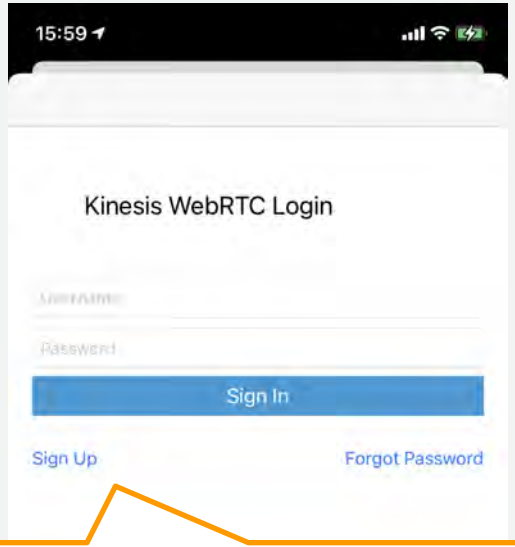
WebRTC



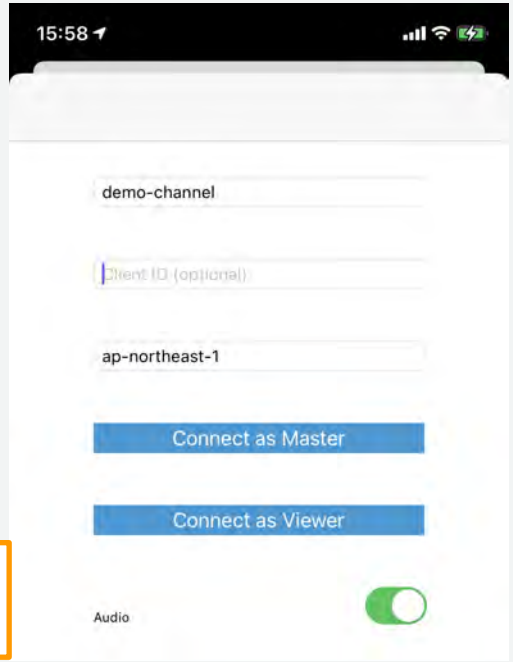
aws

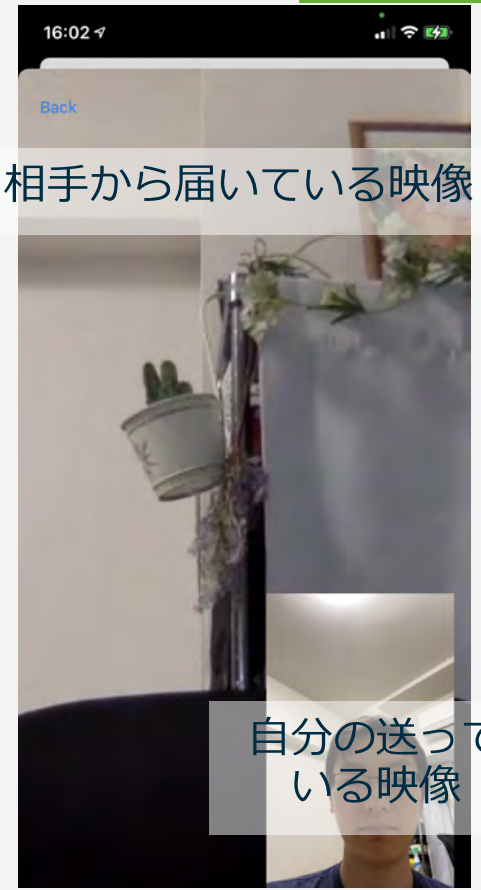
WebRTC SDK iOS のデモアプリ

WebRTC



Amazon Cognito によるログイン機能
(事前にクラウド側のセットアップが必要)





ログイン画面

接続先設定

https://github.com/awslabs/amazon-kinesis-video-streams-webrtc-sdk-ios

WebRTC のトラブルシューティング

WebRTC

接続できない、映像が表示されない

- シグナリングチャネルへ接続できているか、CloudWatch のメトリクスから確認
- 別のネットワークで試してみる
- TURN を有効にしているかを確認
- Master, Viewer でサポートしているコーデックを確認 (SDP Offer / SDP Answer)

シグナリング
メッセージ数

TURN 経由での
接続時間

Master/Viewer
の接続成功率

aws

WebRTC のトラブルシューティング

WebRTC

遅延が大きい、映像が滑らかでない

- 映像のビットレートを確認、ネットワーク帯域に対して大きすぎる場合は解像度を下げてみる
- デバイスのスペックが低い場合、映像のエンコーディング処理などでCPUを使い切っていないかを確認し、解像度を下げたりハードウェアエンコーダの利用を検討する

```
2020-09-29 10:07:55 DEBUG   iceAgentGatherCandidateTimerCallback(): Candidate gathering completed.
2020-09-29 10:07:55 DEBUG   onIceCandidateHandler(): ice candidate gathering finished
2020-09-29 10:07:55 VERBOSE signalingClientGetCurrentState(): Signaling Client Get Current State
2020-09-29 10:07:55 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:55 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:55 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:55 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:55 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 422542.000000
2020-09-29 10:07:55 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:55 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:56 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:56 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:56 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 422542.000000
2020-09-29 10:07:56 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:56 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:56 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:56 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:56 VERBOSE lwsWssCallbackRoutine(): WSS callback with reason 10
2020-09-29 10:07:56 DEBUG   lwsWssCallbackRoutine(): Client is writable
2020-09-29 10:07:56 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 423542.000000
2020-09-29 10:07:56 VERBOSE resendPacketOnNack(): Resent packet ssrc 2049021958 seq 1338 succeeded
2020-09-29 10:07:56 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:56 VERBOSE sampleBandwidthEstimationHandler(): received bitrate suggestion: 516872.000000
2020-09-29 10:07:56 VERBOSE lwsWssCallbackRoutine(): WSS callback with reason 9
```

WebRTC SDK C では、VERBOSE レベルのログを有効化すると、帯域推定結果やパケット再送などのログが表示される

`export AWS_KVS_LOG_LEVEL=1（VERBOSE レベル）`

aws

WebRTC のトラブルシューティング

WebRTC

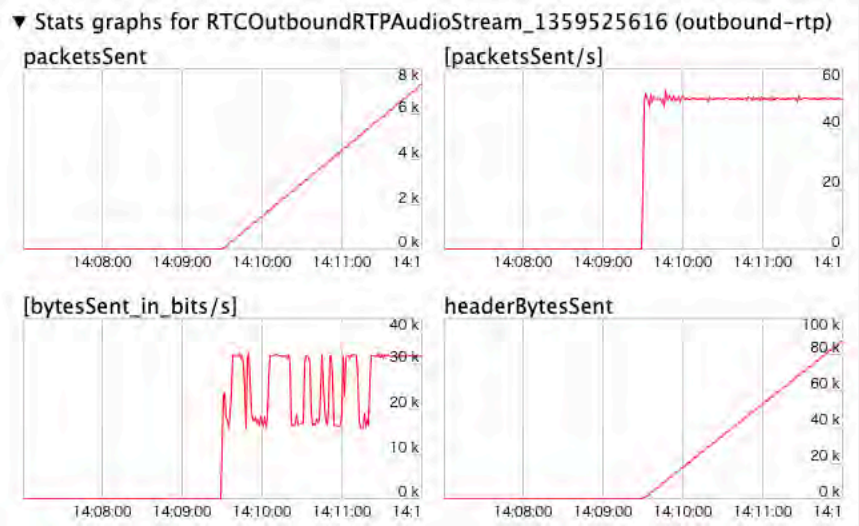
WebRTC Internals でイベントや統計情報から問題を切り分け

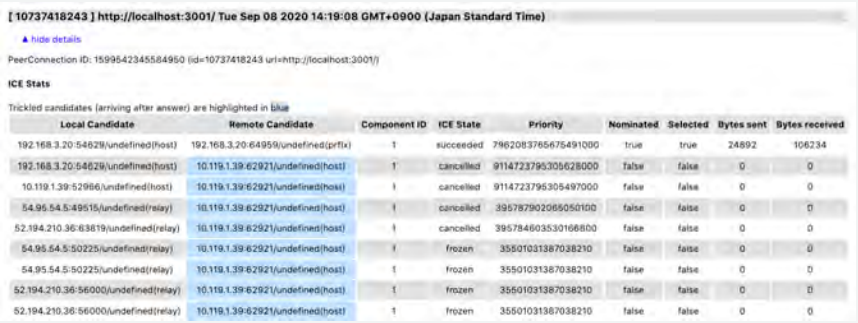
WebRTC Internals (Chrome)

chrome://webrtc-internals/

WebRTC Internals (Firefox)

about:webrtc





[10737418243] http://localhost:3001/ Tue Sep 08 2020 14:19:08 GMT+0900 (Japan Standard Time)

▲ hide details

PeerConnection ID: 1599542345584950 (id=10737418243 url=http://localhost:3001/)

ICE Stats

Trickled candidates (arriving after answer) are highlighted in blue

Local Candidate	Remote Candidate	Component ID	ICE State	Priority	Nominated	Selected	Bytes sent	Bytes received
192.168.3.20:54629/undefined(host)	192.168.3.20:64959/undefined(prflx)	1	succeeded	7962083765675491000	true	true	24892	106234
192.168.3.20:54629/undefined(host)	10.119.1.39:62921/undefined(host)	1	cancelled	9114723795305628000	false	false	0	0
10.119.1.39:52986/undefined(host)	10.119.1.39:62921/undefined(host)	1	cancelled	9114723795305497000	false	false	0	0
54.95.54.5:49515/undefined(relay)	10.119.1.39:62921/undefined(host)	1	cancelled	3957879020650501000	false	false	0	0
52.194.210.36:63819/undefined(relay)	10.119.1.39:62921/undefined(host)	1	cancelled	3957846035301666800	false	false	0	0
54.95.54.5:50225/undefined(relay)	10.119.1.39:62921/undefined(host)	1	frozen	35501031387038210	false	false	0	0
54.95.54.5:50225/undefined(relay)	10.119.1.39:62921/undefined(host)	1	frozen	35501031387038210	false	false	0	0
52.194.210.36:56000/undefined(relay)	10.119.1.39:62921/undefined(host)	1	frozen	35501031387038210	false	false	0	0
52.194.210.36:56000/undefined(relay)	10.119.1.39:62921/undefined(host)	1	frozen	35501031387038210	false	false	0	0

aws

料金構成 (WebRTC)

WebRTC

2020/09/30 現在、東京リージョンの場合

料金構成	料金
アクティブなシグナリングチャネル	$0.045 / チャネル・月
シグナリングメッセージ	$3.375 / 100万メッセージ
TURNでストリーミングした時間 (※)	$0.180 / 1,000分

※ インターネット経由でビデオストリームから AWS 外の送信先にデータを送信する場合は、
AWS の標準のデータ転送料金がかかります。
https://aws.amazon.com/jp/kinesis/video-streams/pricing/

aws

本日のアジェンダ

- Amazon Kinesis Video Streams の概要
- Amazon Kinesis Video Streams - メディア形式で収集
    1. メディアの取り込み
    2. メディアの保存とインデックス
    3. メディアの再生
    4. メディアの分析
- Amazon Kinesis Video Streams – WebRTC
- Amazon Kinesis Video Streams を活用した構成例
- サービス利用上のポイント
- まとめ

Amazon Kinesis Video Streams の使い分け (再掲)

	メディア形式で収集	WebRTC
動画データのクラウド保存 機械学習サービスでの分析	できる	できない
双方向ストリーミング	できない	できる
ライブ再生時のレイテンシ	2秒～	1秒未満
カメラデバイスからの ストリーミングに利用する SDK	Producer SDK	WebRTC SDK

※ レイテンシはデバイス性能、ネットワーク環境、映像のビットレート、フラグメントの設定などによって変化します

aws

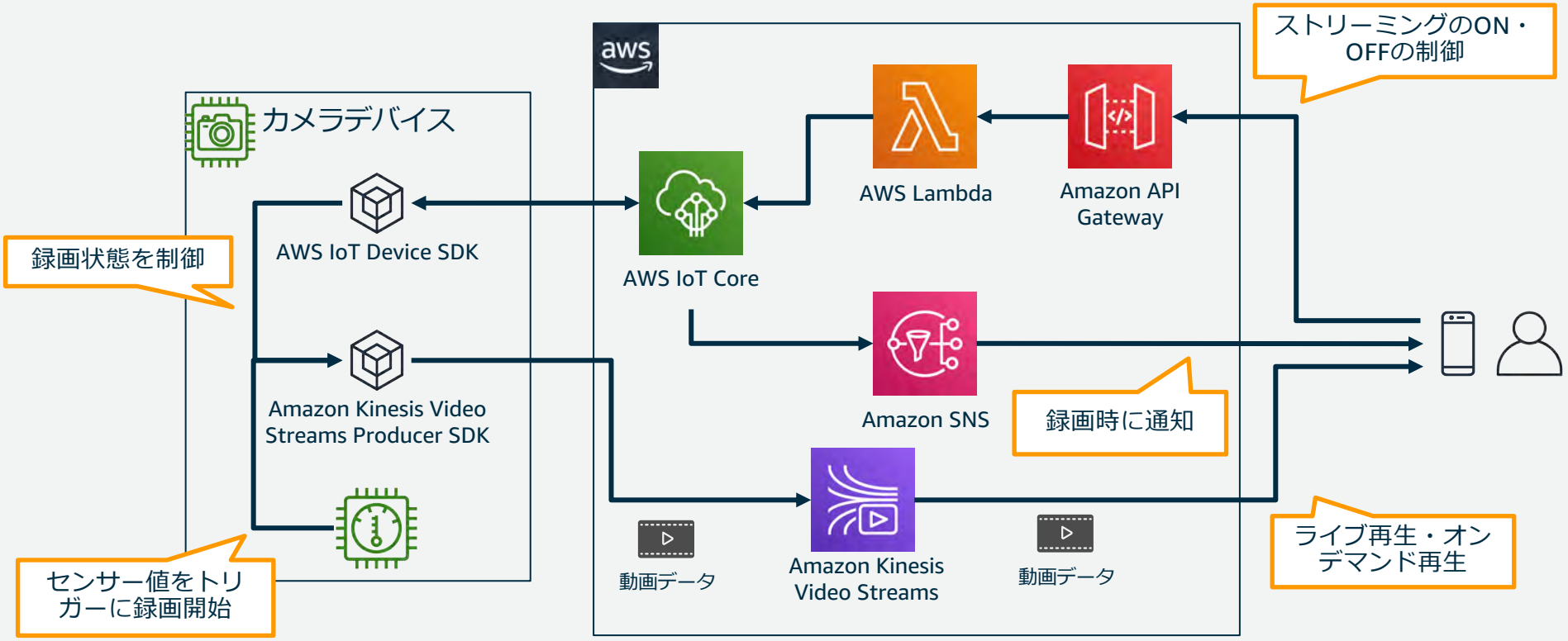
動画の常時保存と再生
監視カメラ・ドライブレコーダー

メディア形式で収集

https://docs.aws.amazon.com/kinesisvideostreams/latest/dg/getting-started.html

必要に応じた動画の収集
ホームセキュリティカメラ

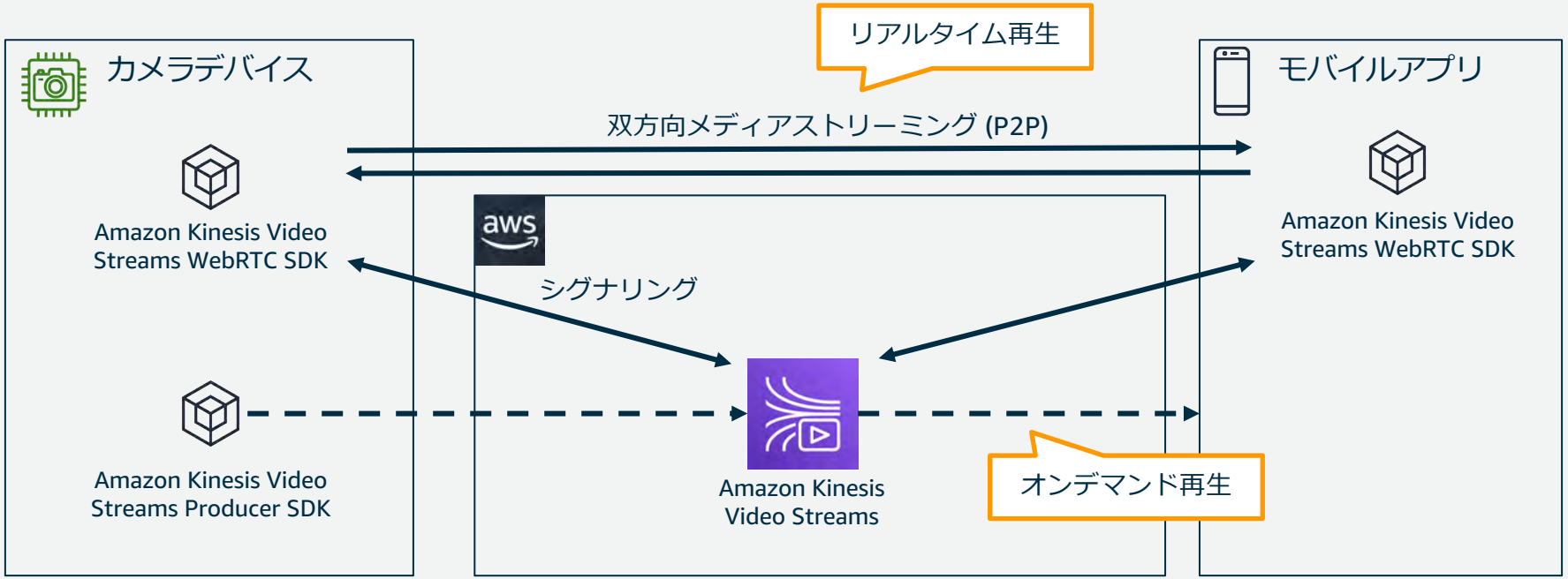
メディア形式で収集



双方向コミュニケーション、遠隔監視
インターホン・(スピーカー付き)監視カメラ

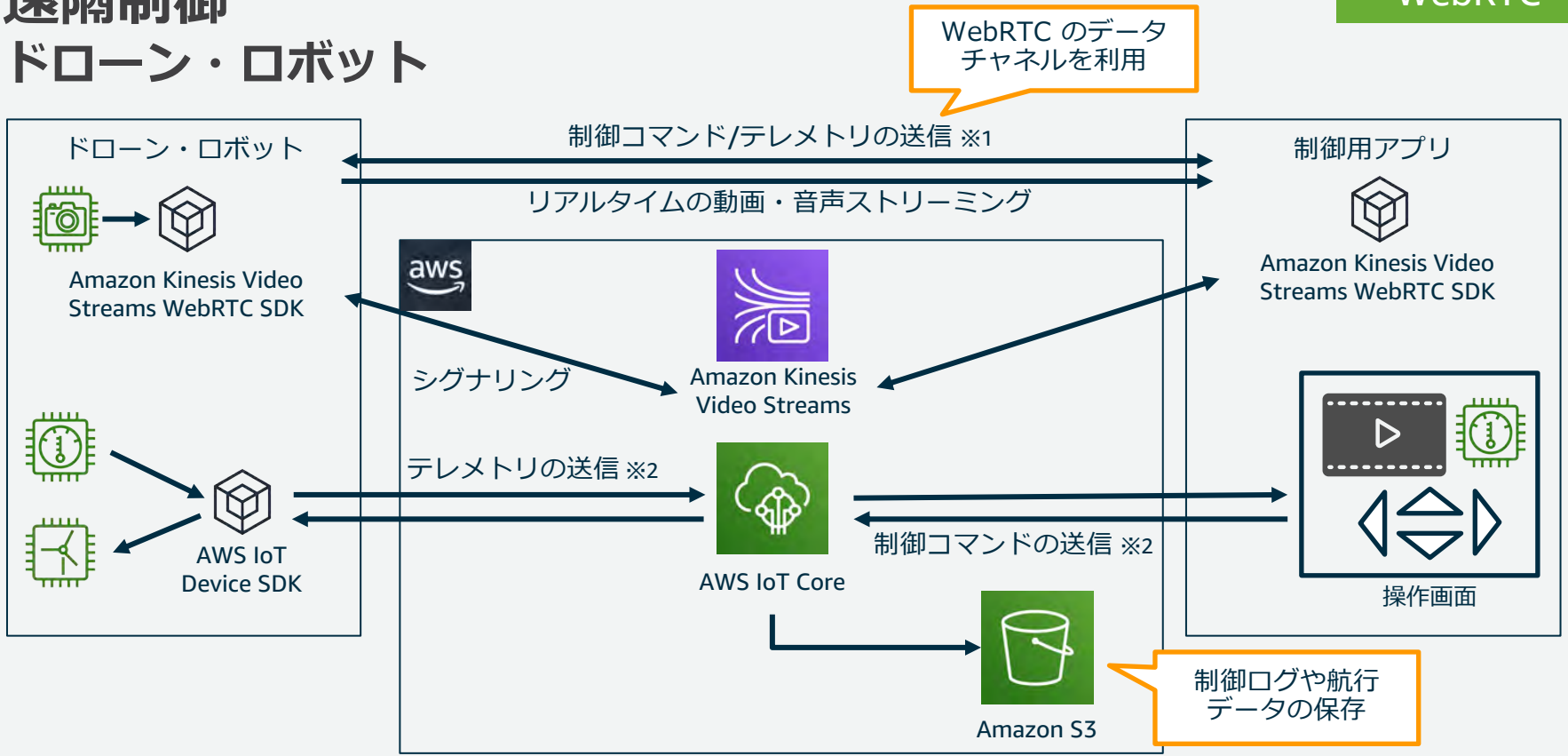
WebRTC

メディア形式で収集



遠隔制御
ドローン・ロボット

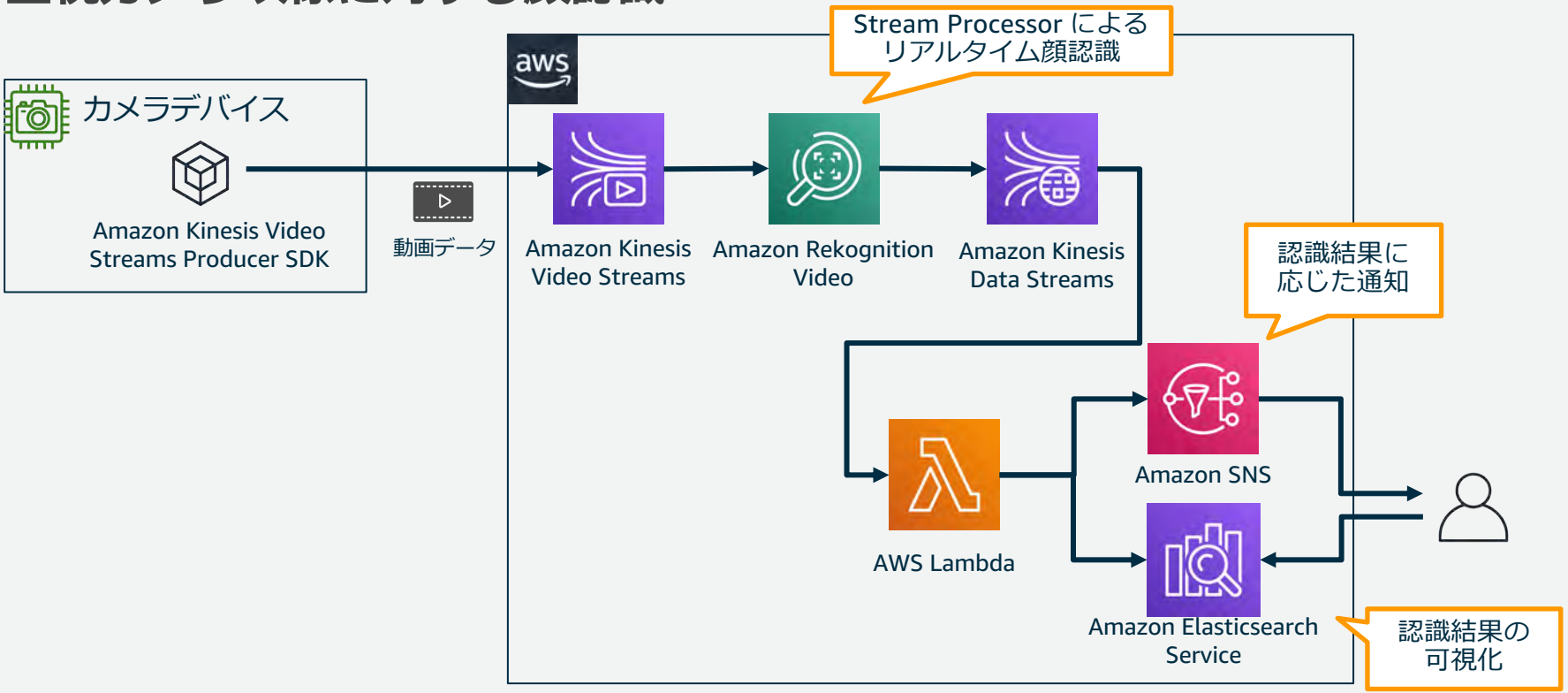
WebRTC



※1 接続中のみメッセージングが出来れば良い場合はWebRTCのデータチャンネルを利用
※2 ログを保存したい場合やオフライン対応をしたい場合は AWS IoT Core を利用

リアルタイム映像分析
監視カメラ映像に対する顔認識

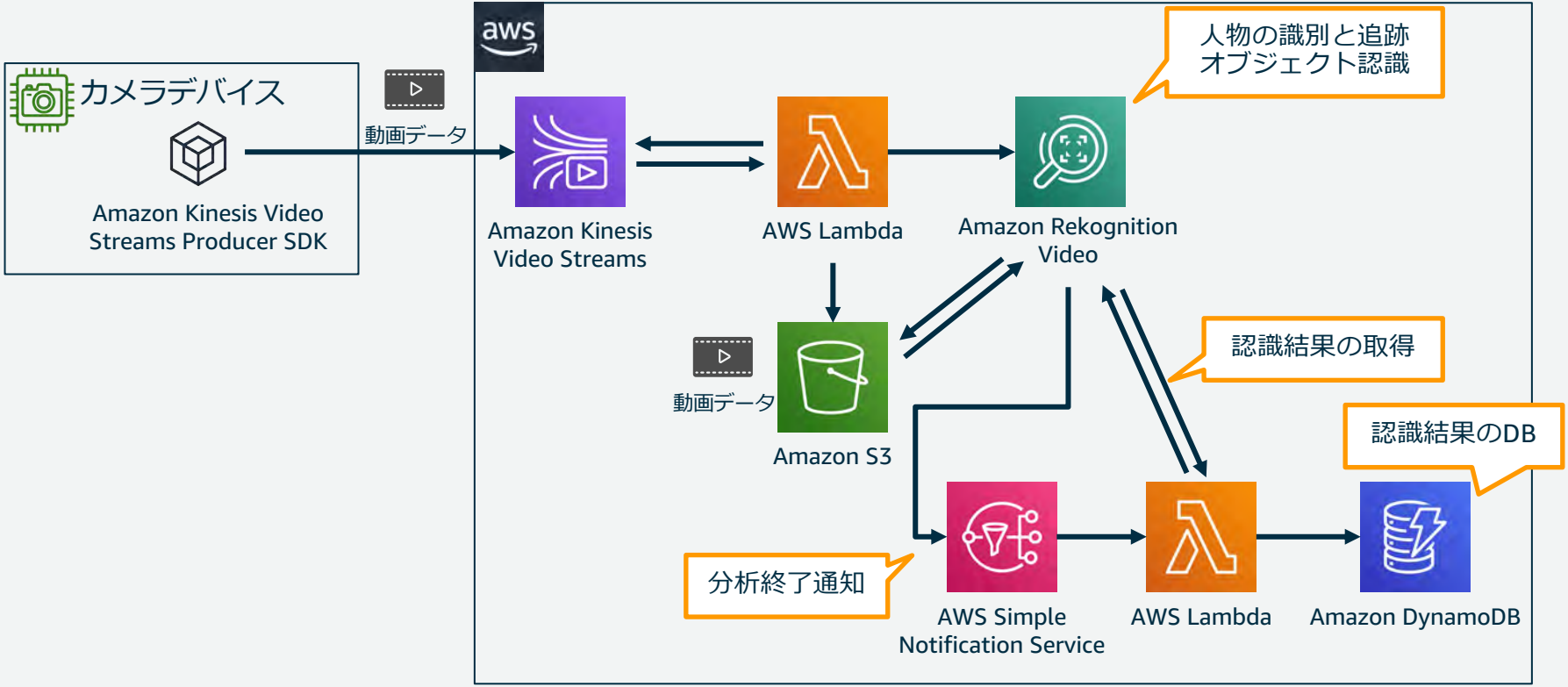
メディア形式で収集



https://aws.amazon.com/jp/blogs/news/easily-perform-facial-analysis-on-live-feeds-by-creating-a-serverless-video-analytics-environment-with-amazon-rekognition-video-and-amazon-kinesis-video-streams/

バッチ映像分析
監視カメラ映像に対するオブジェクト認識・人物トラッキング

メディア形式で収集



本日のアジェンダ

- Amazon Kinesis Video Streams の概要
- Amazon Kinesis Video Streams - メディア形式で収集
    1. メディアの取り込み
    2. メディアの保存とインデックス
    3. メディアの再生
    4. メディアの分析
- Amazon Kinesis Video Streams – WebRTC
- Amazon Kinesis Video Streams を活用した構成例
- サービス利用上のポイント
- まとめ

aws

動画ストリーミングサービスの使い分け

Amazon Kinesis Video Stream

- コネクテッドカメラデバイスからクラウドへ安全にストリーミング
- 機械学習、動画再生、分析、その他の処理をリアルタイムもしくはバッチ駆動で処理可能
- WebRTC による低遅延の双方向ストリーミング
- 例: スマートホーム、スマートシティ、遠隔監視、産業用の自動処理、コネクテッドカメラ製品

AWS Elemental Media Services & Amazon CloudFront

- 放送グレードの品質および柔軟な配信要件に対応可能
- ビルディングブロック、配信要件に応じて詳細な設定変更が可能
- 例: 放送グレードのメディア配信、パッケージング、広告挿入、 DRM、スケジューラー、 DVR/タイムシフト

Amazon Interactive Video Sercvice

- 素早く簡単にセットアップできるマネージドライブストリーミングサービス
- OTT の知識や経験がなくとも、数分でワールドワイドにライブ配信基盤を展開可能
- 超低遅延配信 、 Timed Metadata API を利用することで双方向コミュニケーションを容易に実現可能
- 例: インタラクティブなライブ配信

Amazon Chime / Amazon Chime SDK

- 複数の話者が登場する M : N のビデオ通話用途
- 双方向の円滑な会話のための秒未満配信遅延
- 例: ビデオ会議

カメラデバイスでの SDK への認証情報の渡し方

メディア形式で収集

WebRTC

AWS のアクセスキーを利用

- Access Key ID, Secret Access Key, (Session Token) を環境変数(もしくは引数)として設定
- 認証情報を動的に発行する仕組みもしくは、あらかじめ埋め込んでおく必要がある
- IAM のリソース数の上限に注意が必要

AWS IoT の証明書を利用

- AWS IoT Core で管理しているクライアント証明書を利用して、AWS IoT の認証情報プロバイダーから一時認証情報を取得
- 証明書をデバイスに埋め込む仕組みが必要

https://docs.aws.amazon.com/ja_jp/iot/latest/developerguide/authorizing-direct-aws.html

aws

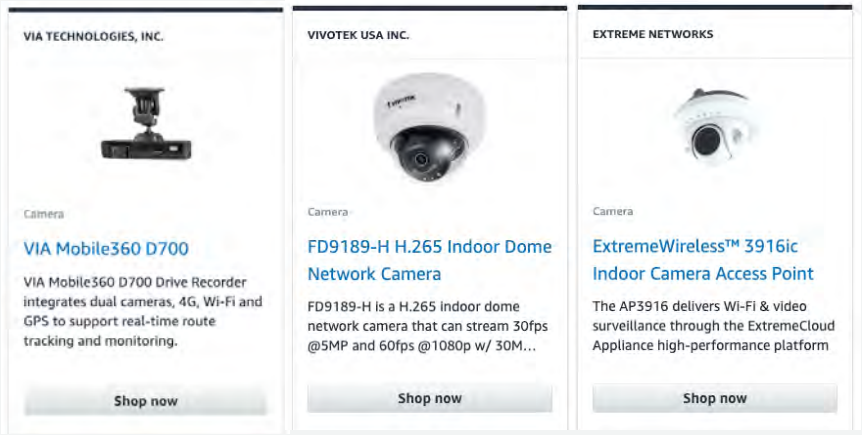
パートナーデバイスとクイックスタート

メディア形式で収集

設定のみでデバイスからクラウドへ動画をストリーミング

パートナーデバイスカタログ

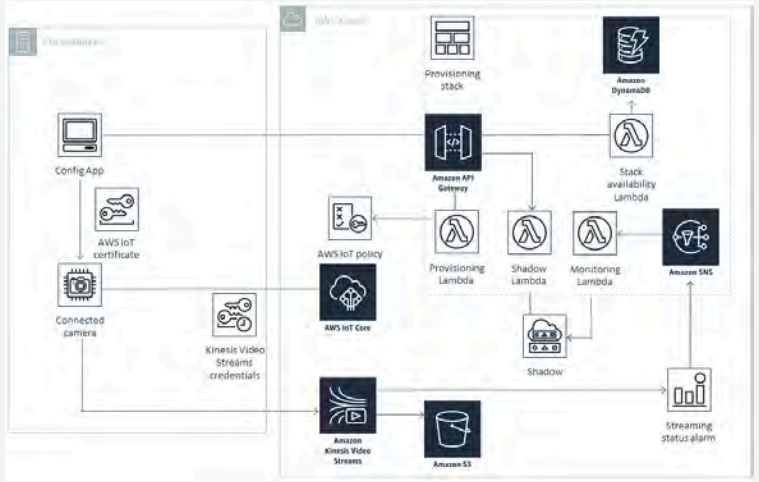
- Amazon Kinesis Video Streams 対応のカメラ製品などが掲載



https://devices.amazonaws.com/

AWS IoT Camera Connector

- IoTカメラ向けのサーバレス環境を１クリックでデプロイ



https://aws.amazon.com/jp/quickstart/architecture/camera-connector-onica/

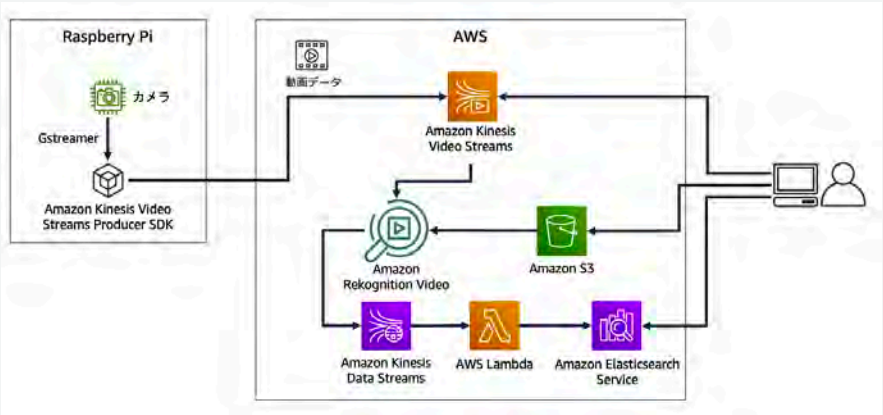
入門用コンテンツのご紹介

メディア形式で収集

WebRTC

Amazon Kinesis Video Streams ハンズオン

https://video-streaming-and-analysis.workshop.aws/



Amazon Kinesis Video Streams WebRTC Getting Started

https://docs.aws.amazon.com/ja_jp/kinesisvideostreams-webrtc-dg/latest/devguide/kvswebrtc-getting-started.html

aws

本日のアジェンダ

- Amazon Kinesis Video Streams の概要
- Amazon Kinesis Video Streams - メディア形式で収集
    1. メディアの取り込み
    2. メディアの保存とインデックス
    3. メディアの再生
    4. メディアの分析
- Amazon Kinesis Video Streams – WebRTC
- Amazon Kinesis Video Streams を活用した構成例
- サービス利用上のポイント
- まとめ

まとめ

- Amazon Kinesis Video Streams はサーバレスなフルマネージドサービスとして提供されており、何百万ものカメラデバイスと接続をして、メディアデータをストリーミングし保存、再生や分析に利用可能

- Amazon Kinesis Video Streams には2つの方式があり、ユースケースに応じて使い分ける

- Amazon Kinesis Video Streams と他の AWS サービスを組み合わせて、IoT における様々なメディアのユースケースを容易に実現できる

Amazon Kinesis Video Streams 関連リンク

- 公式ページ - https://aws.amazon.com/jp/kinesis/video-streams/
- ドキュメントやライブラリなどのリソース - https://aws.amazon.com/jp/kinesis/video-streams/resources/
- 各種 Producer SDK - https://docs.aws.amazon.com/ja_jp/kinesisvideostreams/latest/dg/producer-sdk.html
- Parser Library - https://github.com/aws/amazon-kinesis-video-streams-parser-library
- 各種 WebRTC SDK - https://docs.aws.amazon.com/kinesisvideostreams-webrtc-dg/latest/devguide/webrtc-sdks.html

Q&A

お答えできなかったご質問については
AWS Japan Blog 「https://aws.amazon.com/jp/blogs/news/」にて
後日掲載します。

AWS の日本語資料の場所「AWS 資料」で検索

https://amzn.to/JPArchive

AWS Well-Architected 個別技術相談会

毎週”W-A個別技術相談会”を実施中

- AWSのソリューションアーキテクト(SA)に対策などを相談することも可能
- 申込みはイベント告知サイトから

(https://aws.amazon.com/jp/about-aws/events/)

AWS イベント で[検索]

ご視聴ありがとうございました

AWS 公式 Webinar
https://amzn.to/JPWebinar

過去資料
https://amzn.to/JPArchive