Media over QUICとRTMP+HLSの比較

Kota Yatagai (@kota_yata)

自己紹介

八谷航太（ヤタガイ コウタ）

- 慶應義塾大学環境情報学部2年（村井研究室）
- QUICに興味がある
- WebRTCも高校の頃からいじっていたりする
- X: @kota_yata, GitHub: kota-yata

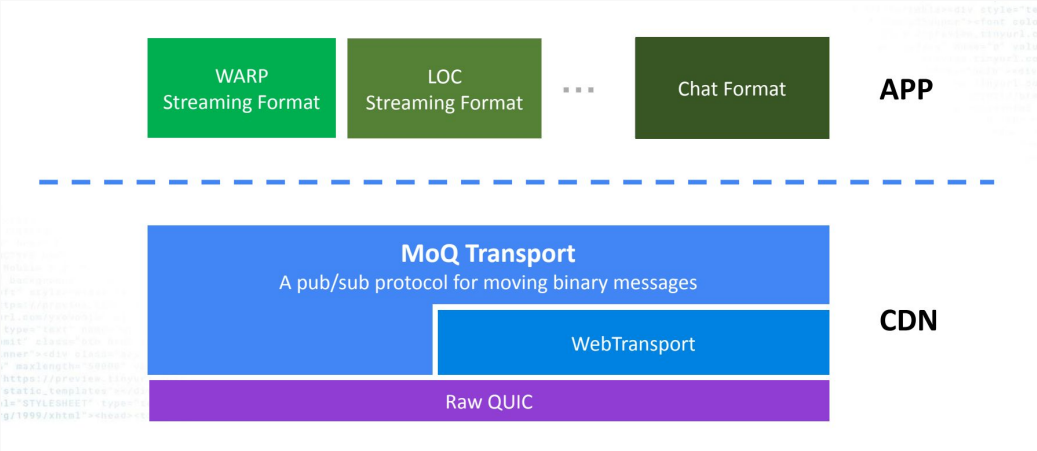
今回やったこと

新興技術Media over QUICと従来のライブ配信手法（RTMP+HLS）との比較

- Media over QUICについてはデコーダは自作、エンコーダとサーバーはMetaのOSSを利用
  - facebookexperimental/moq-go-server
  - facebookexperimental/moq-encoder-player
- RTMP+HLSについてはAmazon IVSのLow-latency Streamingを利用

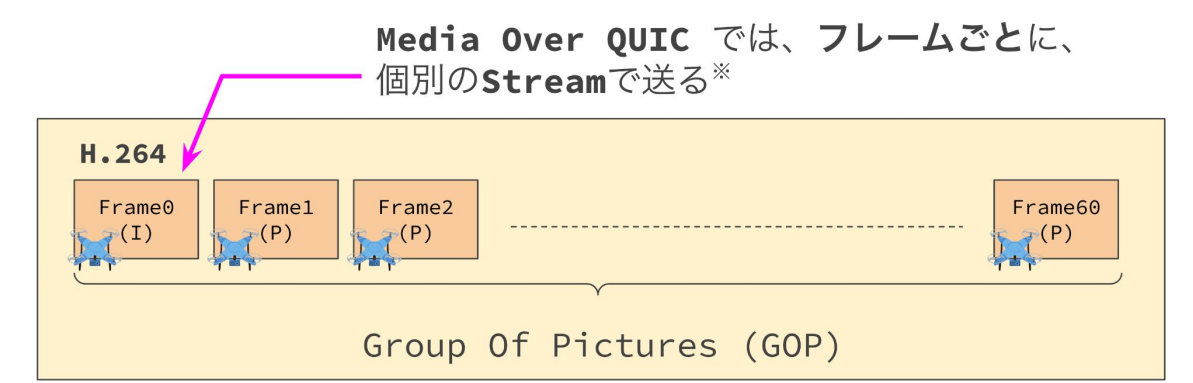
Media over QUICとは？？

- **QUIC上で動くメディアプロトコル**
- コアプロトコルとしてMedia over QUIC Transport (MOQT), ストリーミングフォーマットとしてWARPやLOCが存在する
  - QUICの多重化ストリームを利用することでHoLブロッキングを避ける
  - リレーサーバーの振る舞いをプロトコルに組み込んで大規模配信に対応



Media over QUICの特徴

1フレームごとにストリームを分けて送信できる



（小松さんの資料から引用）

- フレームごとにQUICのストリームを張って送信する
- フレーム単位での優先順位制御（ex.新しいフレームを優先）が可能になる
- パケットドロップしても欠落するのは1フレーム分のみ
- 複数トラックの時間同期が可能になる（音声・映像・MIDI・触覚データ等）

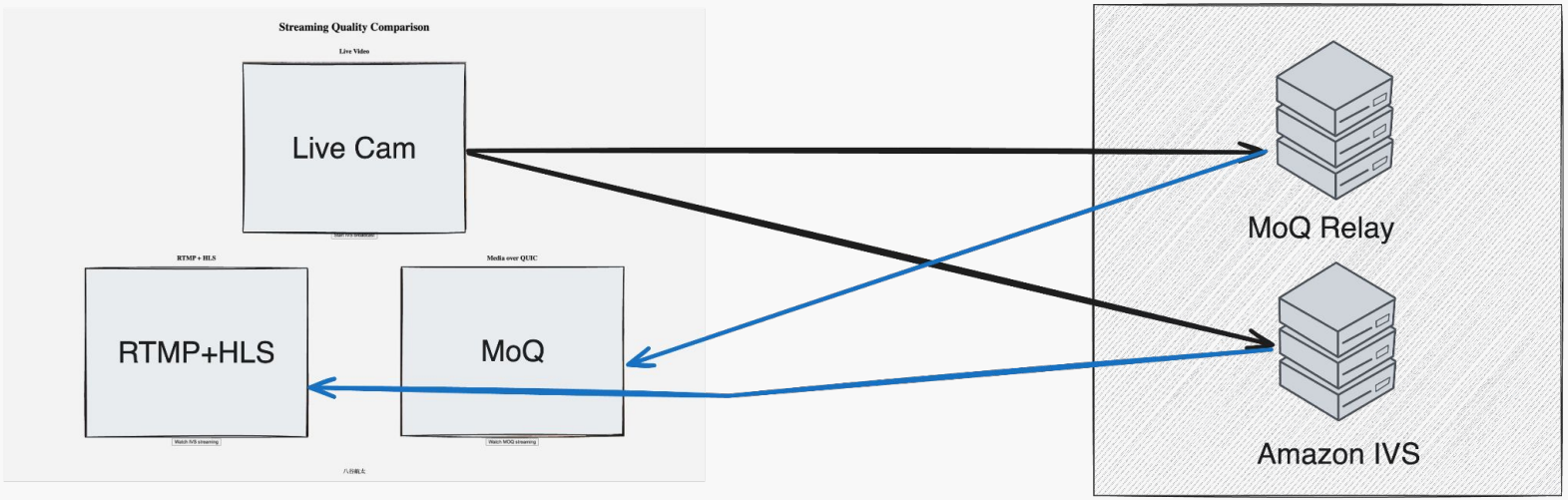
Media over QUICと他プロトコルの比較

- **HLSやDASHではダメなのか？**
  - TCPを使っている限りHoLブロッキングの問題がついて回る
  - Adaptive bitrate(ABR)やセグメント処理が遅延の原因になる（LL-*でもなお遅い）
  - HLSやMPEG-DASHをQUIC上で使っても多重化ストリームを利用できない

- **WebRTCじゃダメなのか？**
  - WebRTCはカスタマイズ性が著しく低い
    - libwebrtcをいじらないといけない
    - MoQはエンコード/デコードはWebCodecsに任せるなど、結構柔軟性がある
  - P2Pのユースケースに特化されている（特にビデオ会議）

デモ

デモの構成



白金高輪

オレゴン（US-West-2）

感想

- jitter buffer入れていないとはいえMoQとても早い
    - QUICが普及するにつれてもっと注目されそう
- 輻輳を意図的に発生させたりしてその時の挙動も比較したい