Media over QUIC

Максим Шарабайко

2023

Обо Мне

- 7+ лет опыта разработки видеокодеков (MPEG-2, H.264/AVC, H.265/HEVC),
- 5+ лет опыта разработки сетевых протоколов.
- Кандидат технических наук, диссертация по алгоритмам сжатия в рамках стандарта H.265/HEVC.
- Интернет-драфты
  - draft-sharabayko-moq-metrics
  - draft-sharabayko-srt
  - draft-sharabayko-srt-over-quic

GitHub

@maxsharabayko

LinkedIn

maxim-sharabayko

maxim.sharabayko@gmail.com

О Чём Доклад?

- IETF и Группа Media Over QUIC.
- Live Media Streaming. Протоколы.
- Протокол Media Over QUIC.
- Как принять участие в IETF?

Internet Engineering Task Force (IETF)

- Открытое международное сообщество проектировщиков, учёных, сетевых операторов и провайдеров.
- Создано в 1986 году.
- Создатели/стандартизаторы:
  - RFC 9293: Transmission Control Protocol (TCP).
  - RFC 768: User Datagram Protocol (UDP).
  - RFC 3550: Real-Time Transport Protocol (RTP).
  - RFC 8831: WebRTC Data Channels.
  - RFC 9000: Quick UDP Internet Connections (QUIC).
  - …
- Рабочая группа Media Over QUIC (февраль 2022 г.).

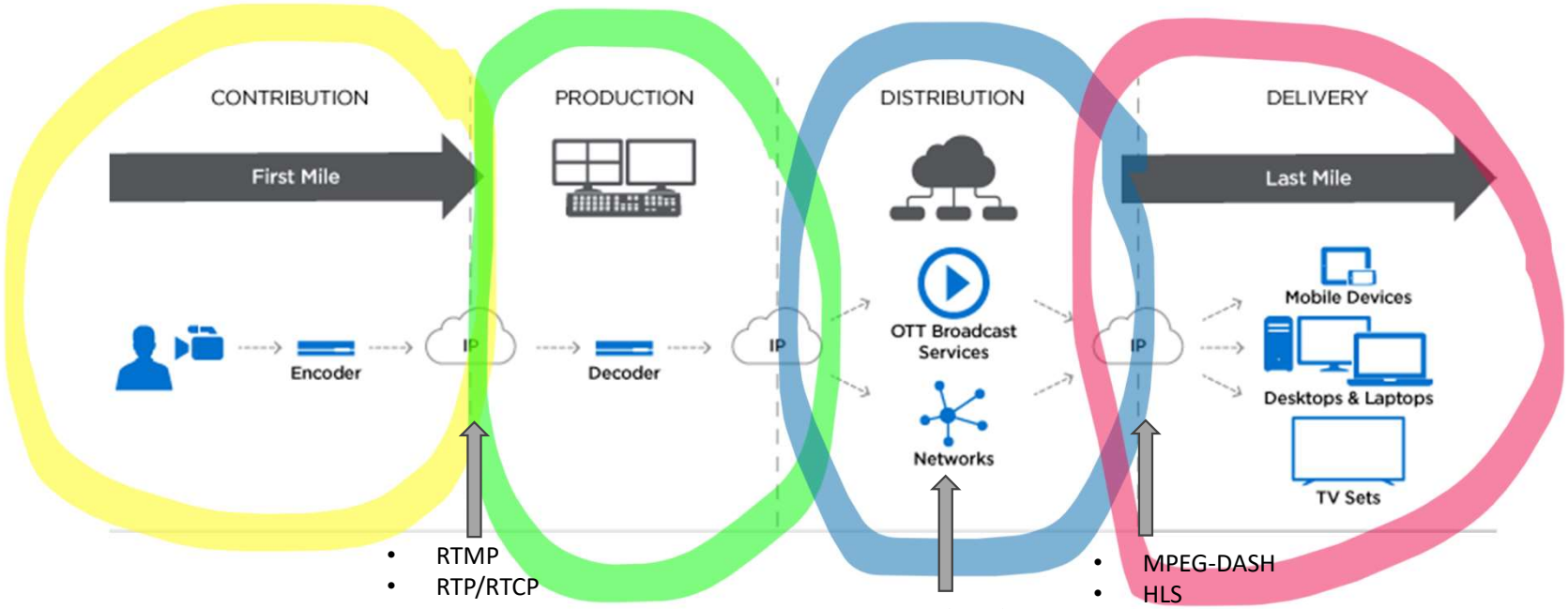
Задачи с Медиа в Фокусе MOQ

- Interactive media.
  - Gaming.
  - Remote desktop.
  - Video Conferencing/Telephony.
- Live media
  - Live Media Ingest.
  - Live Media Syndication.
  - Live Media Streaming.
- Hybrid Interactive and Live Media.

*draft-ietf-moq-requirements-02 - MOQ Use Cases and Requirements

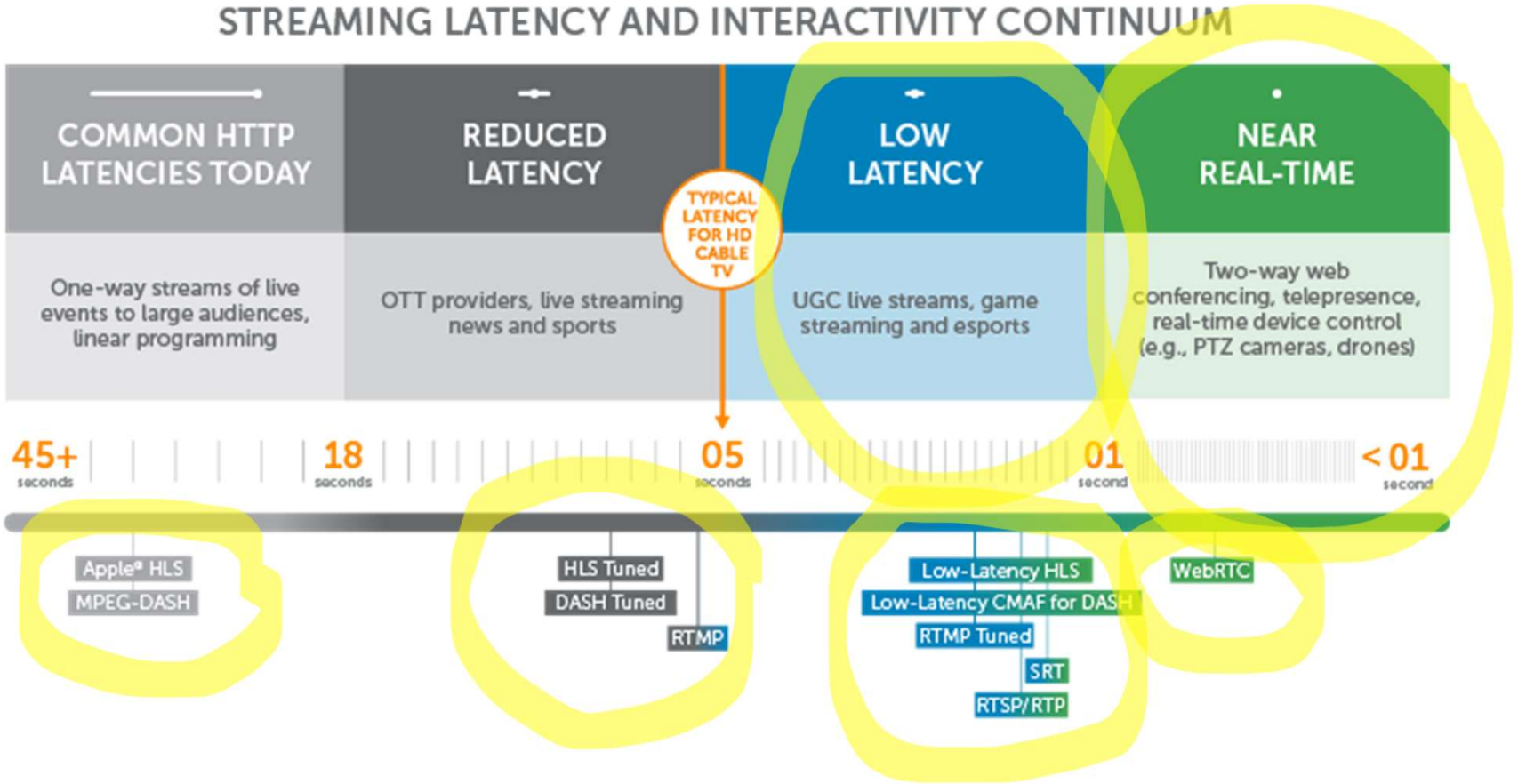
Live Media Streaming

- Обычно односторонняя передача медиа.
- Более высокая задержка лучше, чем потери пакетов.



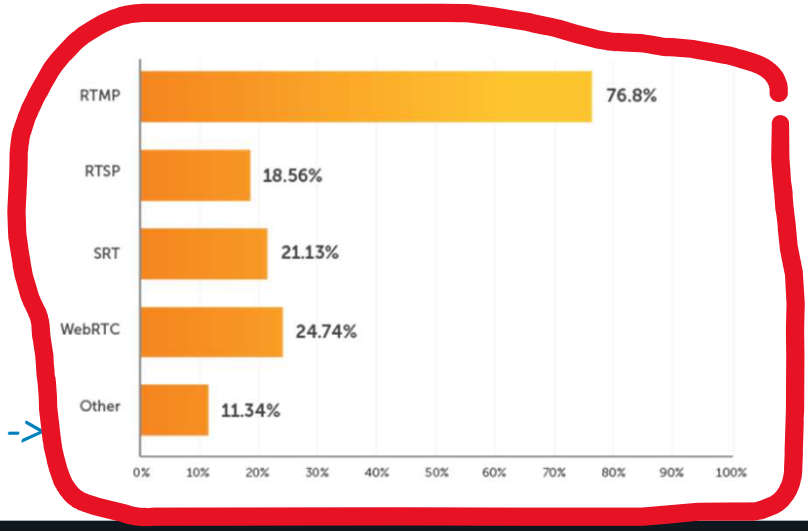
Video Encoding Basics: Live Video Streaming Protocols for Broadcast Contribution and Distribution - Haivision

Протоколы Ingestion

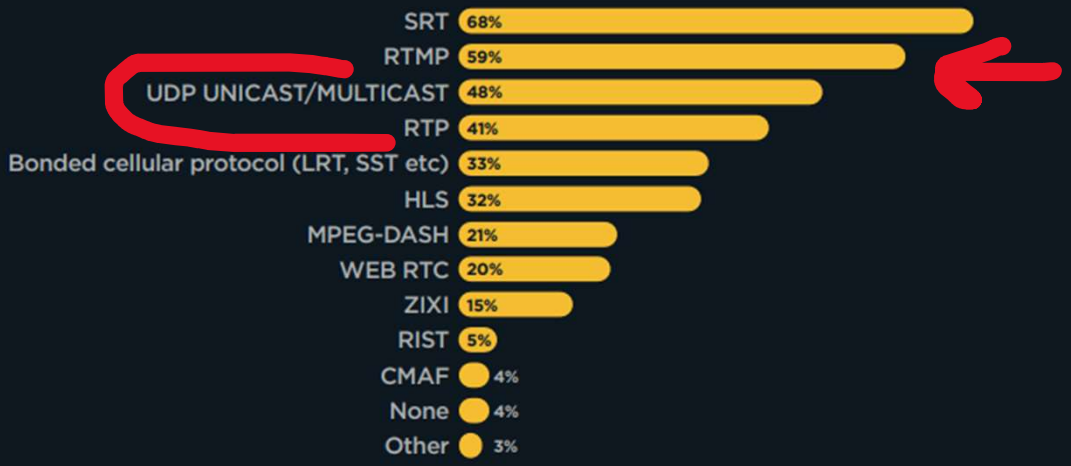


[Streaming Protocols: Everything You Need to Know | Wowza]

Использование в Broadcast



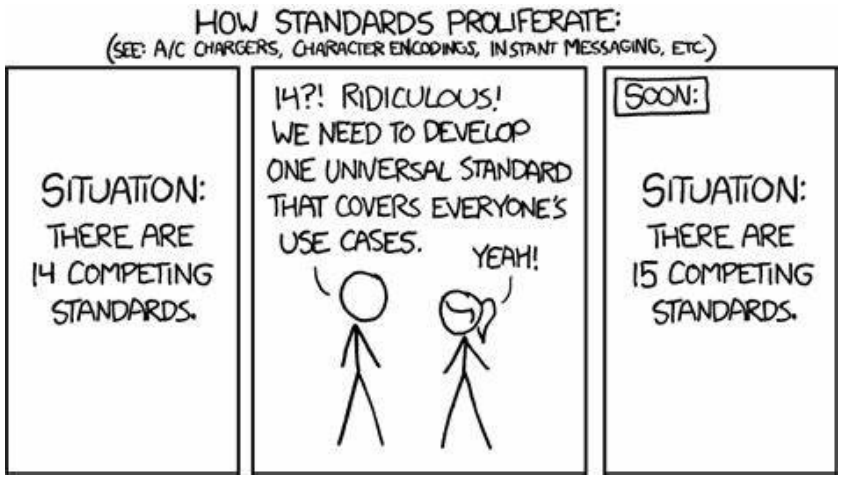
[2021 Video Streaming Latency Report | Wowza ->]



MOQ - Ещё Один Стандарт?

Задачи/цели Media over QUIC (moq):

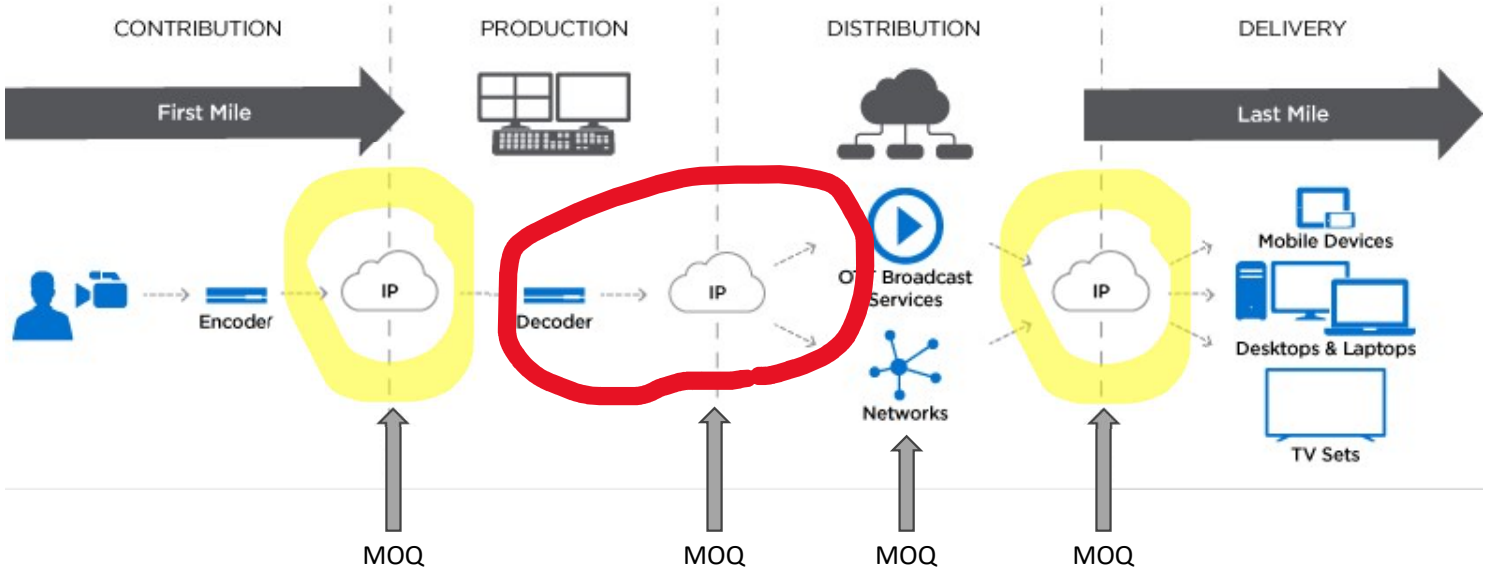
- простое решение для ingestion и distribution медиа;
- низкая задержка;
- эффективная масштабируемость;
- доступность в и вне браузеров.



https://xkcd.com/927

Источники Задержек

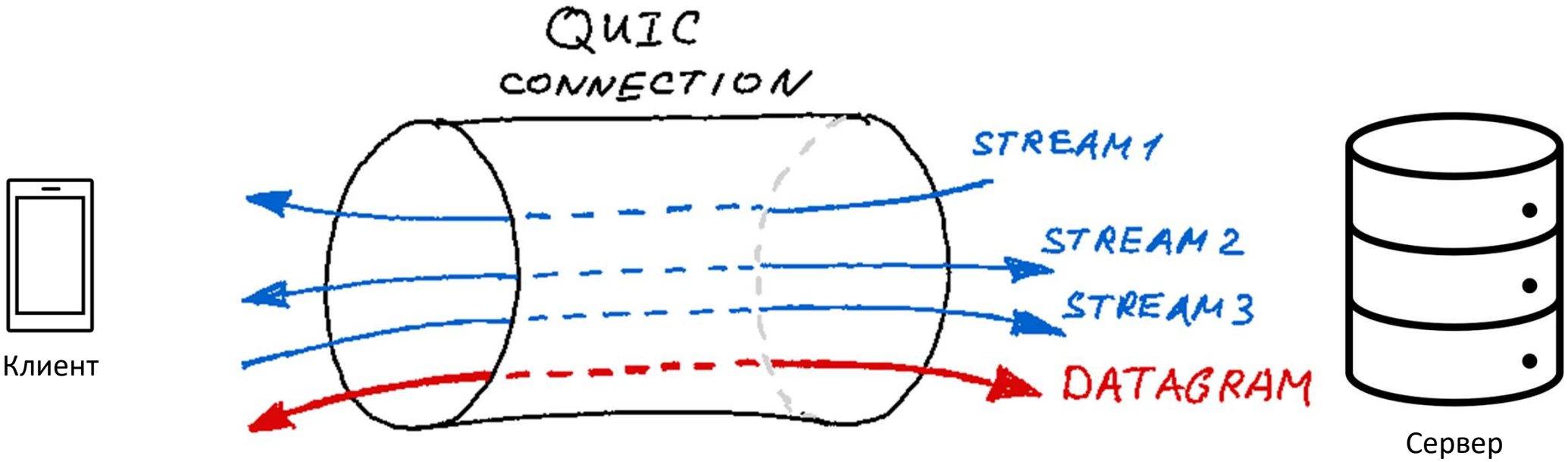
Подготовка контента (нарезка HLS) существенный источник задержек, особенно для крупных игроков рынка live streaming под Web (Twitch, YouTube Live,..).



Новый Протокол QUIC

- Основанный на UDP **транспортный протокол** с мультиплексированием стримов, шифрованием, ~~блэкджеком и ...~~
- Начало работы в 2010 г., опубликован в мае 2021.
  - RFC 8999 (version-independent properties),
  - RFC 9000 (QUIC),
  - RFC 9001 (TLS/QUIC),
  - RFC 9002 (loss detection, congestion control).
  - RFC 9221 (**Unreliable** Datagram Extension).
- Задача: адресовать недостатки TCP, сохранив его достоинства.

Стримы и Датаграммы



QUIC vs TCP vs UDP

	TCP	UDP	QUIC Streams	QUIC Datagrams
Шифрование	(TLS)	(DTLS)	TLS 1.3	TLS 1.3
Наличие CC, ARQ, …				!
Минимальная задержка				(CC)
Отсутствие Head-of-line blocking			! (внутри стримов)	
Легко менять				
Мультиплексирование				
Распозн. congestion				
Быстрое переподключение		!		
Доступность протокола в сети		!	!	!
Доступ в браузере			(WebTransport)	(WebTransport)

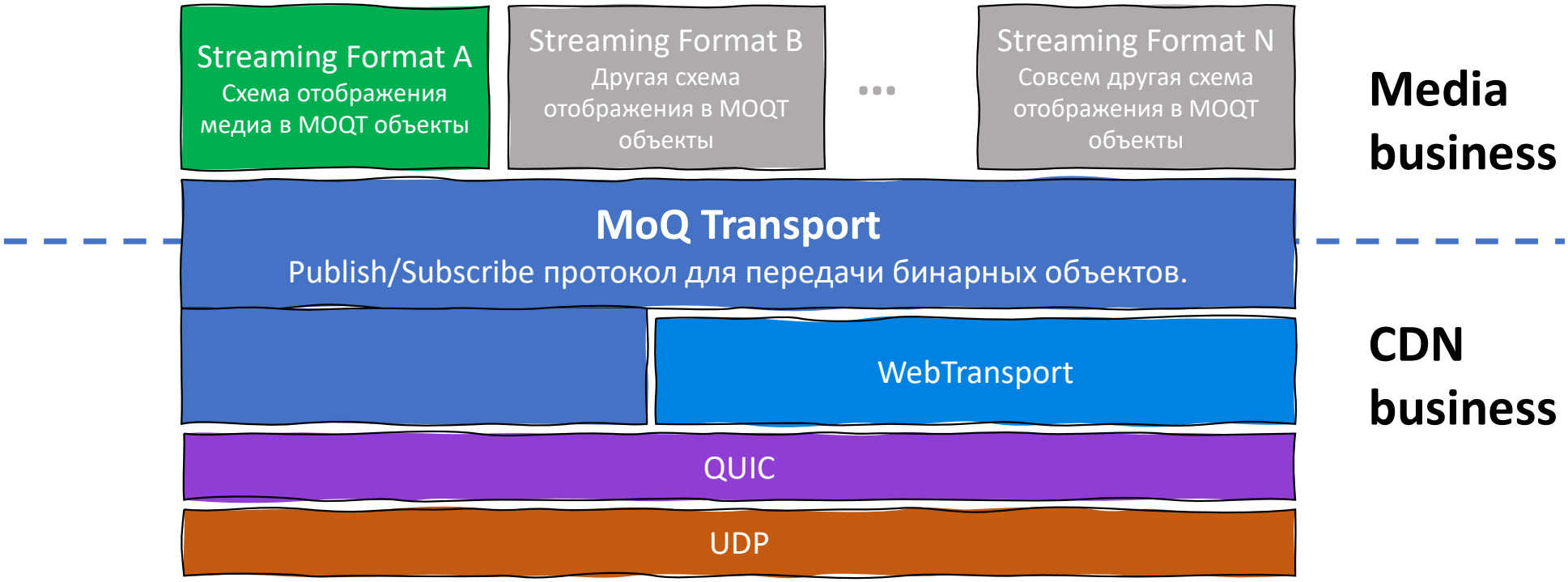
Так Зачем MoQ?

	HLS/DASH	WebRTC	RTMP	SRT	MoQ
Задержки	👎	🤘	👎	🤘	👍
Масштабируемость	🤘	👎	👎	👎	🤘
Шифрование	👍 (TLS)	👍 (TLS)	👍 (TLS)	👍	🤘
ABR	🤘	👎	👎	👎	🤘
Rate Control	👎	👌	👎	👉	?
Доступность в сети	🤘	👍	👍	👍	👍
Доступ в браузере	🤘	🤘	👎	👎	🤘
Доступность вне браузера	👍	👍	👍	👍	🤘
Простота использования	🤘	👌	👍	🤘	🤘
Ingestion	👎	👍	👌	🤘	👍
Distribution/Delivery	🤘	👎	👎	👎	🤘

Media Over QUIC

- Создана в феврале 2022 г.
- Используя достоинства QUIC создать гибкий протокол с низкими задержками и быстрой реакцией на congestion.
- Воспользоваться параллельной природой QUIC стримов.
- Предоставить гибкое восстановление потерь.
- **Устранить переупакопку контента. Единый протокол от точки входа до конечного потребителя**.
- Поддержка существующих кодеков и контейнеров (обратная совместимость и интероперабельность с существующими экосистемами).
- Поддержка relay, масштабируемость. Relay должны иметь доступ к информации, необходимой для доставки контента, но не к самому контенту.

Архитектура MoQ



MoQ Transport

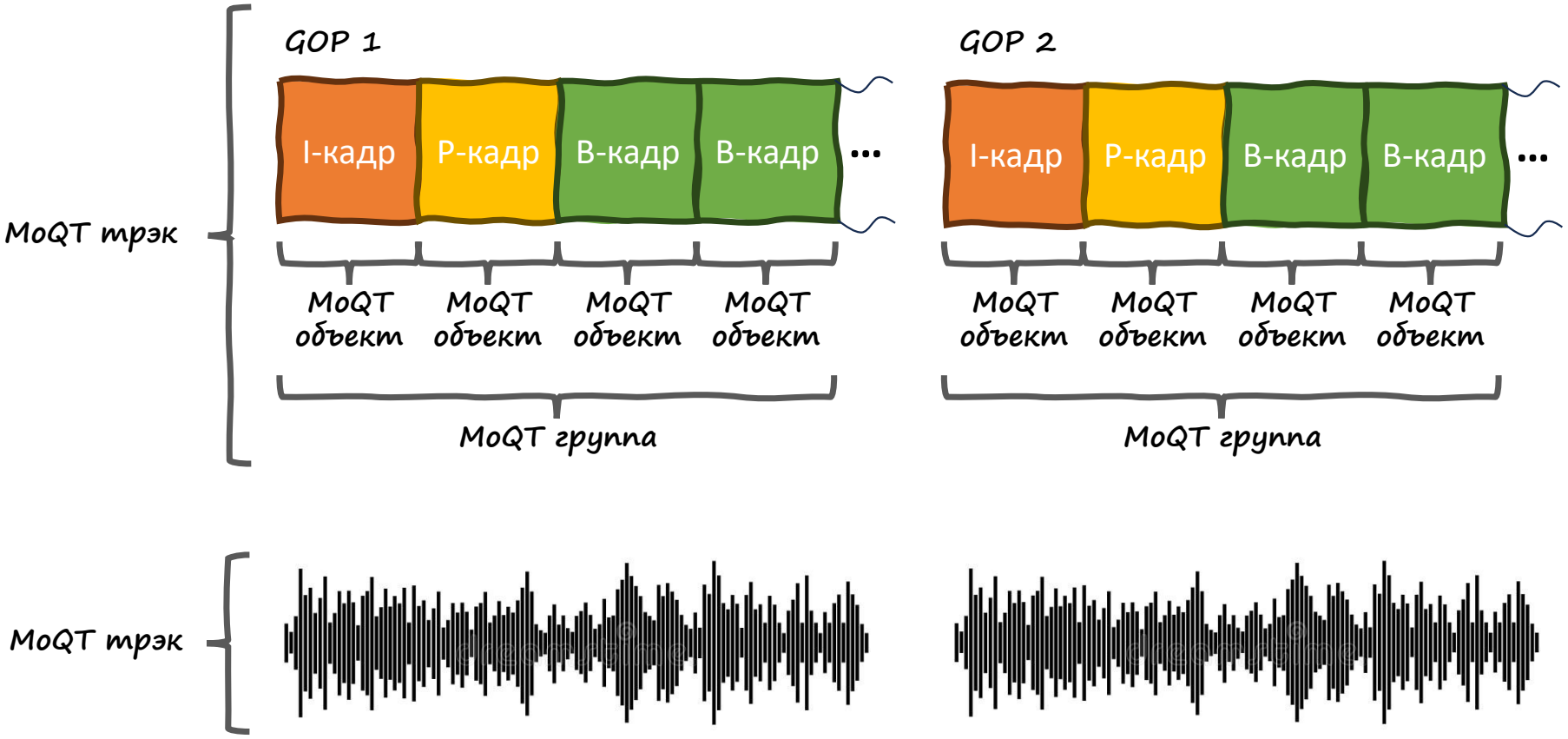
- Media over QUIC Transport (MOQT) - это транспортный протокол, который использует быстрый сетевой протокол QUIC, либо напрямую, либо через WebTransport, для передачи медиа.
- Модель publish/subscribe:
  - Производители публикуют (publish) медиа;
  - Множество потребителей отправляют subscribe запросы.
  - Relay-серверы агрегируют и перенаправляют запросы и медиа.
- MOQT – обобщенный протокол, поэтому предполагается использование одного или нескольких MOQ Streaming Formats

Базовые Единицы MoQT

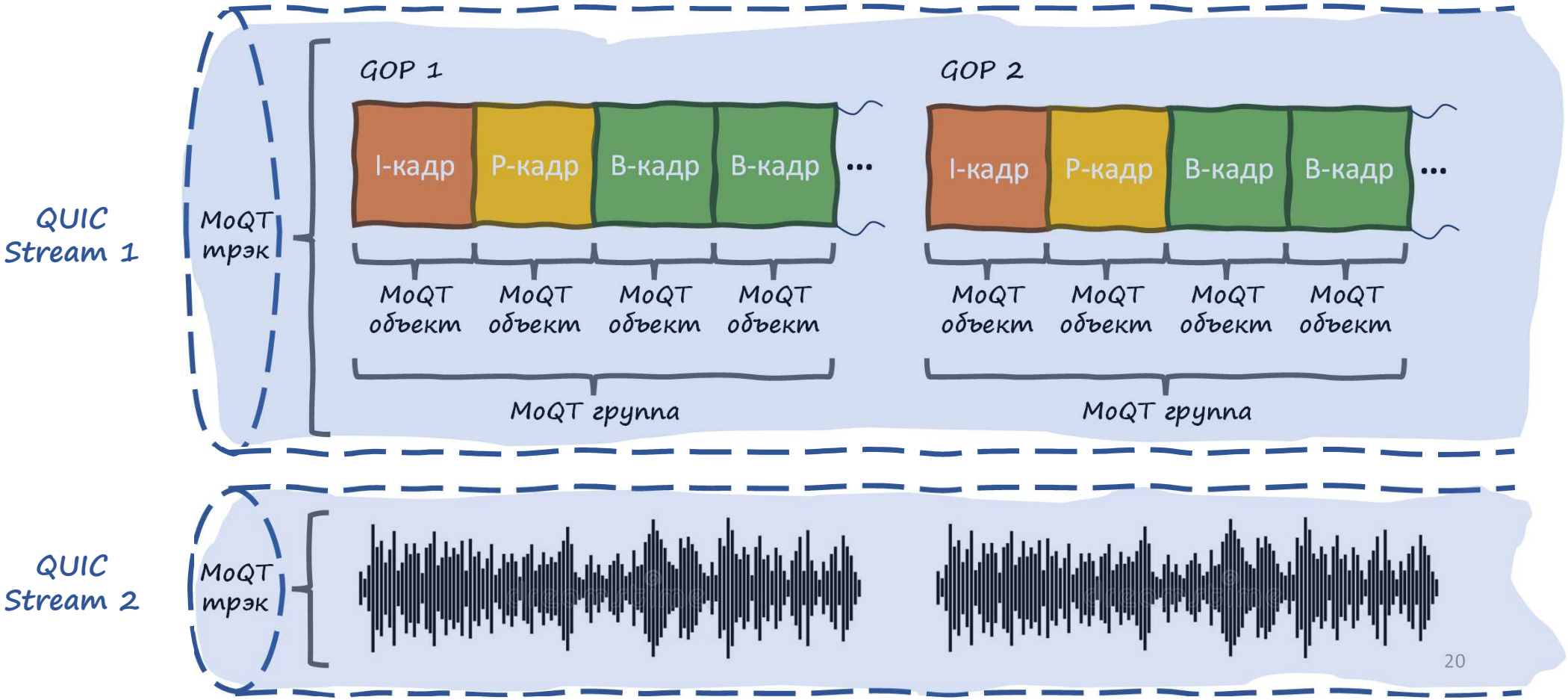
- **Объект** – базовая адресуемая единица с payload.
- **Группа** – последовательность объектов и базовая единица трэка.
  - Объекты одной группы не должны зависеть от объектов другой группы.
  - Представляет собой точку, с которой можно начать/продолжить проигрывание трэка (join point).
- **Трэк** – закодированный битстрим, последовательность групп.
  - Можно присвоить имя и пространство имён (namespace).
  - Содержат последовательности из одной или нескольких групп.
  - Являются объектами подписки начиная с границы группы, включая новые объекты, публикуемые в будущем пока трэк активен.

Отображение Медиа → MoQT

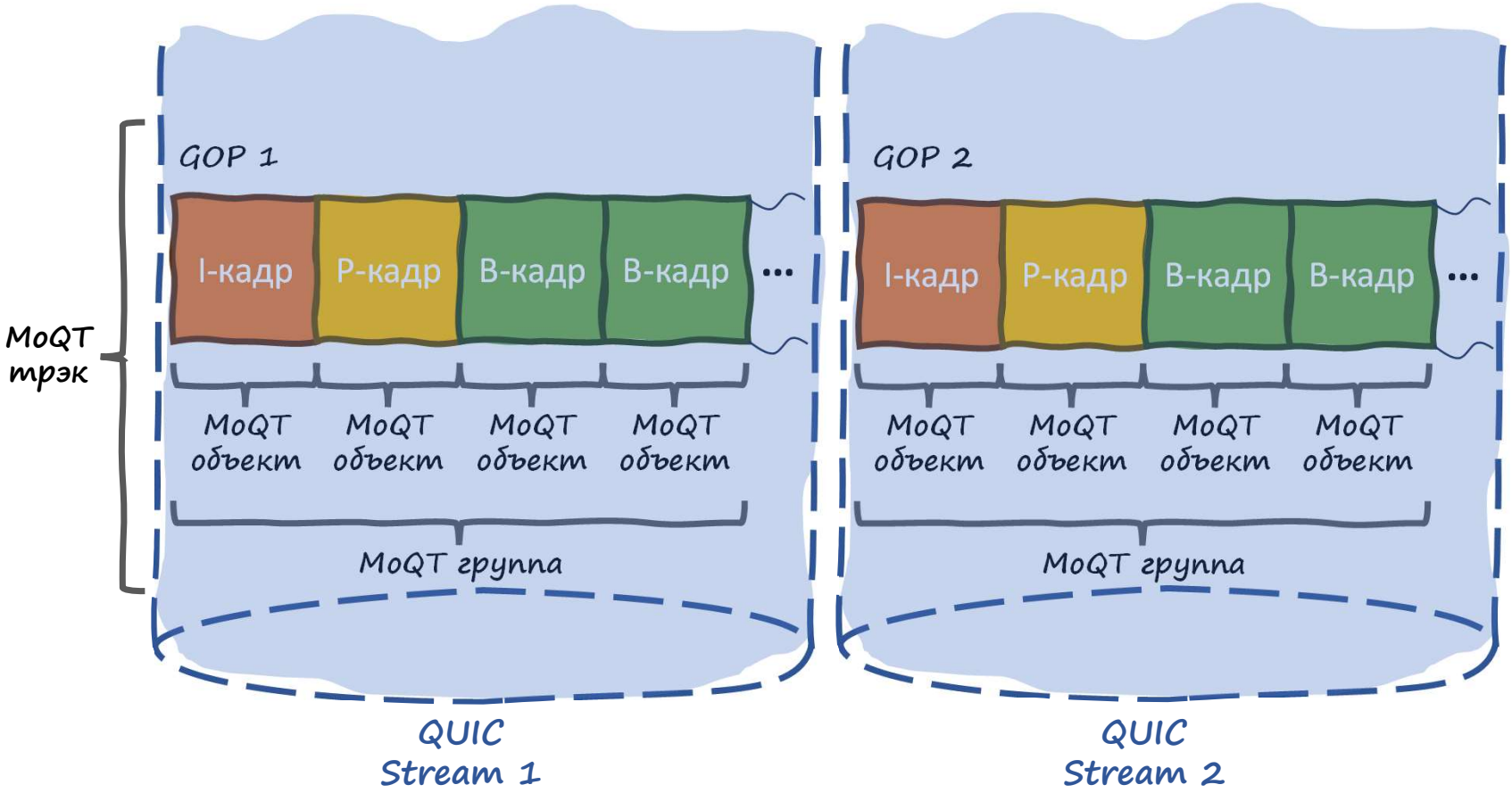
MoQ Streaming Format определяет отображение элементарного видеопотока в MoQT сущности!



Медиа → MoQT → QUIC Streams



Медиа → MoQT → QUIC Streams Вариант 2



Отображение MOQT в QUIC

- Релэям позволяется применять и изменять отображение в транспортный протокол QUIC по своему усмотрению. Но нельзя объединять, нарезать и как-либо ещё изменять payload объектов.
- Один QUIC стрим на MOQT группу.
  - Удобное, логично и упорядочено для принимающей стороны.
  - Head-of-line blocking и потенциальные задержки.
- Один QUIC стрим на MOQT трэк.
  - Самая простая схема, но HoL blocking.
- Один QUIC стрим на MOQT объект.
  - Можно максимально снизить задержки.
  - Много стримов, особенно если объекты короткие.
  - Приемное приложение должно уметь переупорядочивать объекты.
- Один QUIC стрим на несколько MOQT трэков.
- Один QUIC стрим на MOQT объекты одного приоритета.
  - Потенциально для релэев

Сообщения MoQT

- OBJECT с payload;
- OBJECT без payload;
- SUBSCRIBE;
- SUBSCRIBE_OK;
- SUBSCRIBE_ERROR;
- ANNOUNCE;
- ANNOUNCE OK;
- ANNOUNCE ERROR;
- UNANNOUNCE;
- UNSUBSCIBE;
- SUBSCRIBE_FIN;
- SUBSCRIBE_RST;
- GOAWAY;
- CLIENT_SETUP;
- SERVER_SETUP.

```
MOQT Message {
  Message Type (i),
  Message Payload (..),
}
```

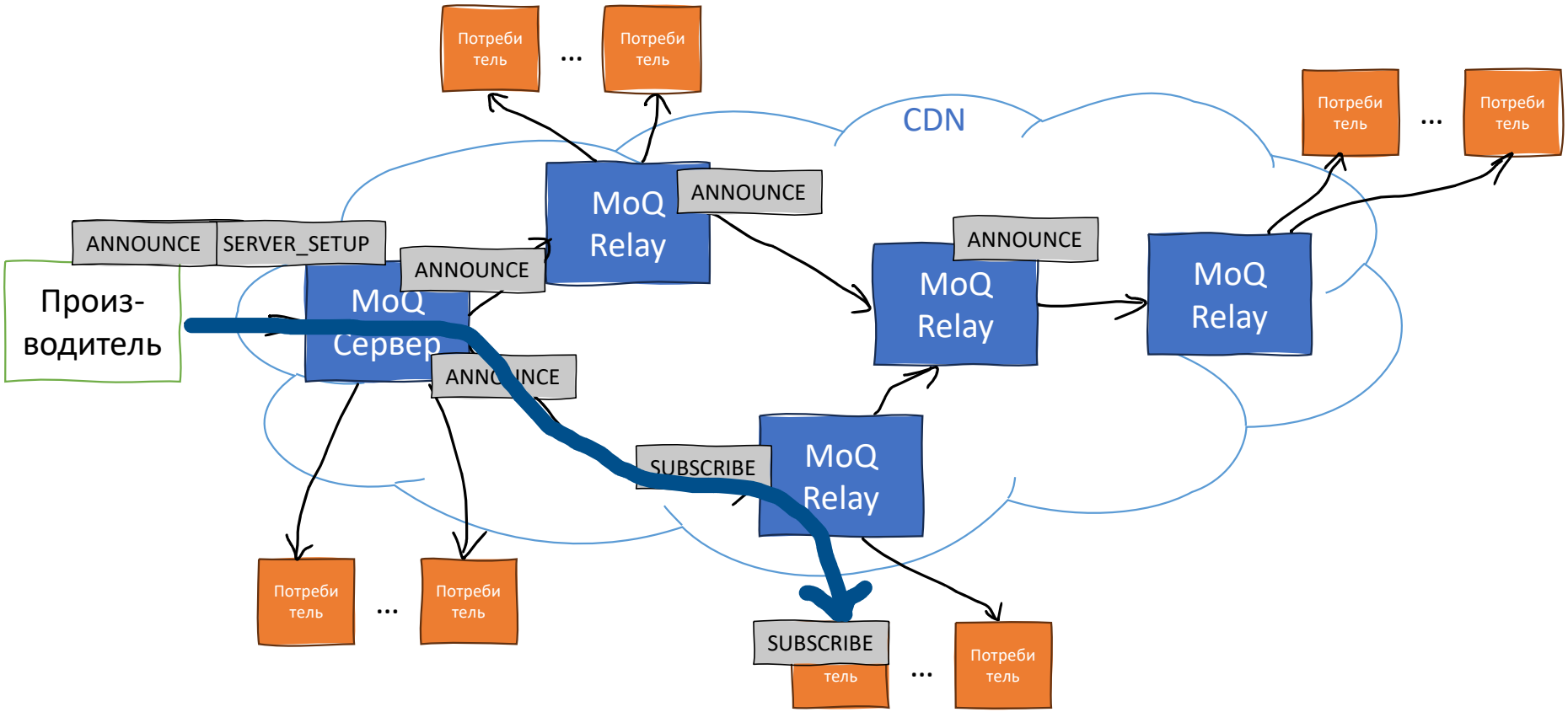
Объекты MoQT

- Объект состоит из двух частей:
  - метаданные (не закодированы и видны релэям);
  - Payload (может быть закодирован, тогда доступен только производителю и потребителям).
- Любой объект должен принадлежать какой-либо группе, тем самым обозначая очередность и потенциальные зависимости.

```
OBJECT Message {
  Track ID (i),
  Group Sequence (i),
  Object Sequence (i),
  Object Send Order (i),
  [Object Payload Length (i),]
  Object Payload (b),
}
```

```
MOQT Message {
  Message Type (i) = OBJECT,
  Message Payload (..) = OBJECT Message,
}
```

Архитектура Системы

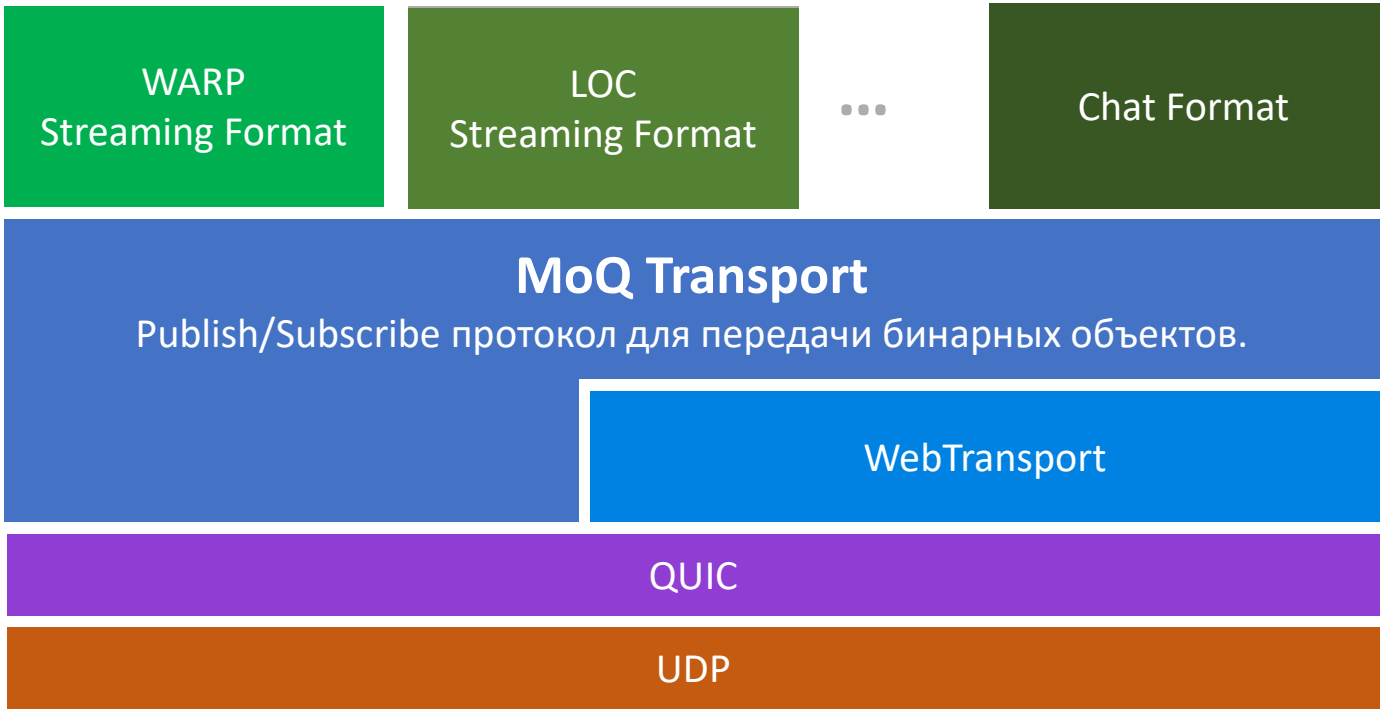


URL Схема

- **moq-URI** = moq:// authority path-abempty ["?" query]
- **WebTransport** = https:// authority path-abempty ["?" query]

Что такое MoQ Streaming Format?

MoQ Streaming форматы определяют как содержимое закодировано, запаковано и отображено на объекты MOQT, а также правила обнаружения (discovery) и подписки (subscription).



WARP Streaming Format

- Цель: интерактивные уровни задержки.
- Битстрим каждого кодека должен быть помещен в отдельный трэк и свою последовательность объектов.
- Либо весь GOP, либо каждый кадр в отдельности отображается в объект.
- Трэки должны быть синхронизированы по времени (CMAF Aligned Switching Sets удовлетворяют данному условию).
- Каждая группа должна быть независимо декодируема. Например, новый ID группы под каждый CMAF фрагмент.
- Объект «Каталог» со списком трэков.
- Более высокий приоритет у более свежих объектов.
- Содержимое может быть зашифровано, например, AES CBC. Обмен ключами пока не описан.

WARP: Объект «Каталог»

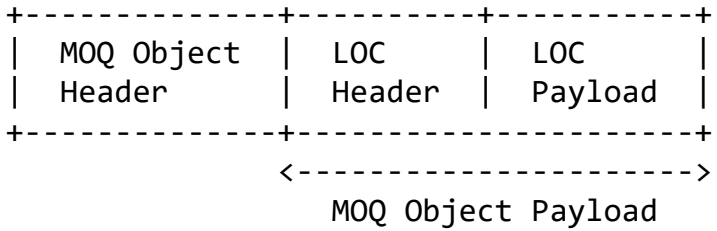
Имя трэка: «catalog».

```
CATALOG payload {
   media format type (i),
   version (i),
   parent object sequence (i),
   track change count (i),
   track change descriptors (..)
 }
```

```
Track Change Descriptor {
   full track name length (i),
   full track name (..),
   operation (1),
   change payload(..)
 }
```

Low Overhead Container (LOC)

- Цель: минимальный оверхэд при работе с WebCodecs.
- Поддержка кодеков определенных в WebCodecs Codec Registry.



- LOC Header – метаданные (таймстэмп, порядковый номер кадра, частота аудиодорожки, пр.) для функционирования релэев в случае, если LOC Payload зашифрован.
- LOC Payload <- EncodedAudioChunk или EncodedVideoChunk.
- Объект «Каталог» со списком трэков. Может быть end-to-end зашифрован.

Как Контрибьютить?

- Рабочая группа MoQ: https://datatracker.ietf.org/group/moq/about
- Mail-to: moq@ietf.org
- Почтовый архив https://mailarchive.ietf.org/arch/browse/moq/
- https://quicdev.slack.com
- https://quic.video
- moq-rs: реализация на Rust
  - relay server
  - ffmpeg plugin
  - MoqTransport library
  - WebTransport library
- moq-js: веб-реализация на Typescript
  - web playback
  - web contribution
  - library

Предстоящие Встречи

IETF | Upcoming meetings

- IETF 119: 16-22 марта 2024. Брисбен, Австралия
- IETF 120: 20-26 июля 2024. Ванкувер, Канада.
- IETF 121: 2-8 ноября 2024. Дублин, Ирландия

Спасибо за внимание!

@maxsharabayko

maxim-sharabayko

maxim.sharabayko@gmail.com