RTP over QUIC

draft-engelbart-rtp-over-quic-00

IETF 111

Jörg Ott - ott@in.tum.de
Mathis Engelbart - mathis.engelbart@gmail.com

Motivation

- Provide an encapsulation for carrying RTP/RTCP over QUIC
- Leverage information already available in QUIC
- Adjust how to use RTCP

- Discuss how to interact with QUIC congestion control

Related Work

- Similar draft: QRT: QUIC RTP Tunneling (draft-hurst-quic-rtp-tunnelling-01) by Sam Hurst (BBC Research & Development)
  - Leverage QUIC connection state to reduce exchange of RTCP packets
- Both drafts use QUICs Unreliable Datagram Extension (draft-ietf-quic-datagram-03)
- Currently using Flow IDs in Datagrams to demultiplex RTP Sessions
- Use SDP for signalling

- Recent related work: RUSH: A/V over non-RTP over QUIC
  - draft-kpugin-rush-00.html

A note on RTP (RFC 3550, 3551, ...)

- Real-time Transport Protocol: Rich toolbox for real-time media over IP
- RTP: Packet framing for real-time media
  - Payload identification
  - Payload formats for payload-specific encapsulation (plentiful!)
  - Playout timing, jitter computation, loss detection, stream synchronization
  - Support for retransmissions, FEC, codec-specific repair schemes
- RTCP: feedback channel for real-time media sessions
  - Reception + QoE statistics
  - Loss signaling + codec-specific recovery mechanisms
  - Congestion control signaling
- RTP+RTCP may be carried in UDP, TCP (DTLS, TLS)
  - Can be multiplexed on the same port, so can be multiple sessions
  - Works for groups: with multicasting (by design) and for different (p2p) topologies
  - Now: mapping onto QUIC
  - New: opportunity for more reliability
- Basis for WebRTC, SIP, and other conferencing systems
  - Integrated with MIME types, signaling frameworks, NAT traversal, ...
  - Earlier: also for media streaming but lost to HAS / DASH

RTP/RTCP

- Early background: draft-rtpfolks-quic-rtp-over-quic-01

- RTCP traditionally provides feedback information
  - Round-trip time, jitter, packet losses – at varying granularity in resolution and time
  - RR, Generic NACK, XR, ...
- RFC 8888 defines detailed congestion control feedback
  - Per-packet receipt / loss reports + detailed timing
  - To feed into NADA, SCReAM, GCC, ...
- QUIC naturally gathers partly similar packet statistics
  - Sometimes limited resolution: e.g., only aggregate timing per ACK, though
  - Exact loss information (also for QUIC DATAGRAM frames)
  - RTT measurements every few packets
  - Packet “timestamp” for every packet that triggers an ACK
    - Which may or may not carry a DATAGRAM frame
- Where possible, use QUIC’s information instead of redundant RTCP packets
  - May require interpolation at the sender

Local Interface Requirements

QUIC Implementation

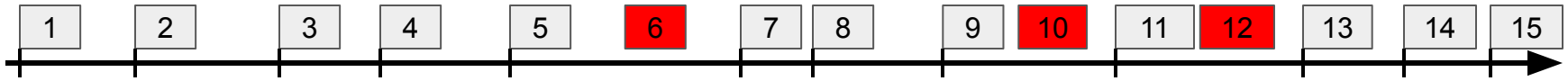
- Unreliable QUIC DATAGRAM frames MUST be supported
- ACKs/Loss of DATAGRAM frames MUST be signalled to the application
- RTT statistics MUST be exposed to application

Congestion Controller

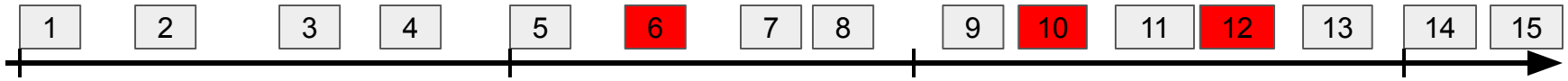
- Assuming one of the algorithms proposed by RMCAT for real-time media
- Input: ACKed packets, delay, RTT estimations, optionally ECN
- Output: bandwidth estimation for media encoder
- QUIC internal vs. external CC: What about QUIC streams and non-RTP DATAGRAM frames?

Some details (see I-D)

RTP + RFC 8888



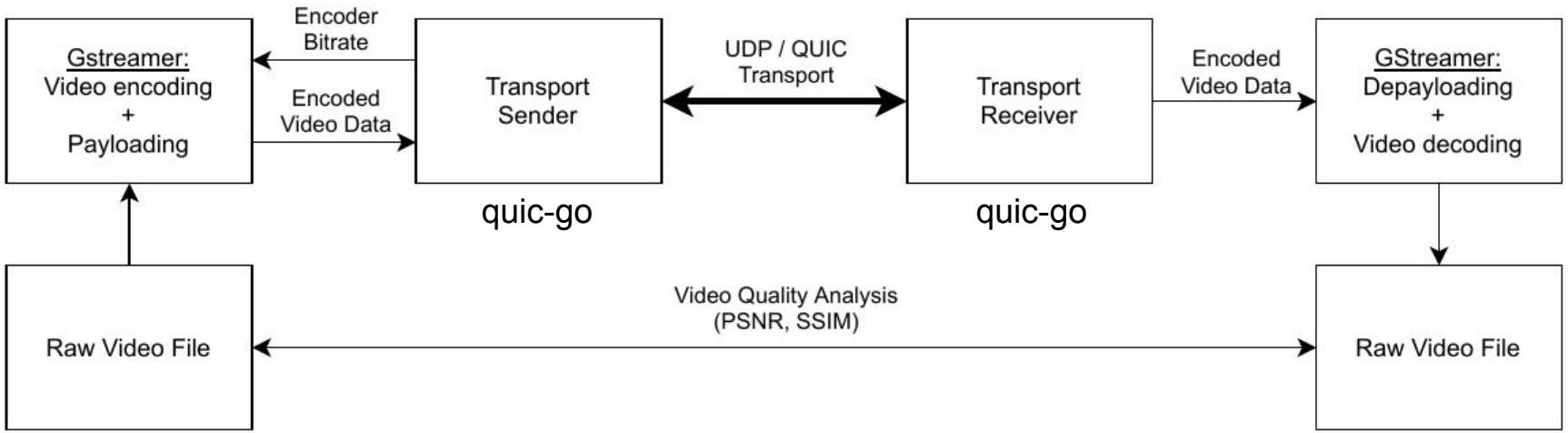
QUIC DATAGRAM frames



- QUIC indicates which DATAGRAM frames map to which packets
  - and when they were actually sent
- QUIC indicates which packets were received / lost
  - along with the reception time of the QUIC packet that triggered the ACK
- QUIC indicates RTT stats (min_rtt, smoothed_rtt, rtt_var)
- Use the above to interpolate the reception times of the other RTP packets
- Option: use QUIC TS extension for one-way delay estimation

Experiments

- Exploring different approaches to interpolation
  - for different CC algorithms, different reporting, ...
- Currently: gstreamer (w/ SCReAM | plain) over (quic-go | plain UDP)
  - quic-go with congestion control disabled for DATAGRAM frames



Quick note on signaling

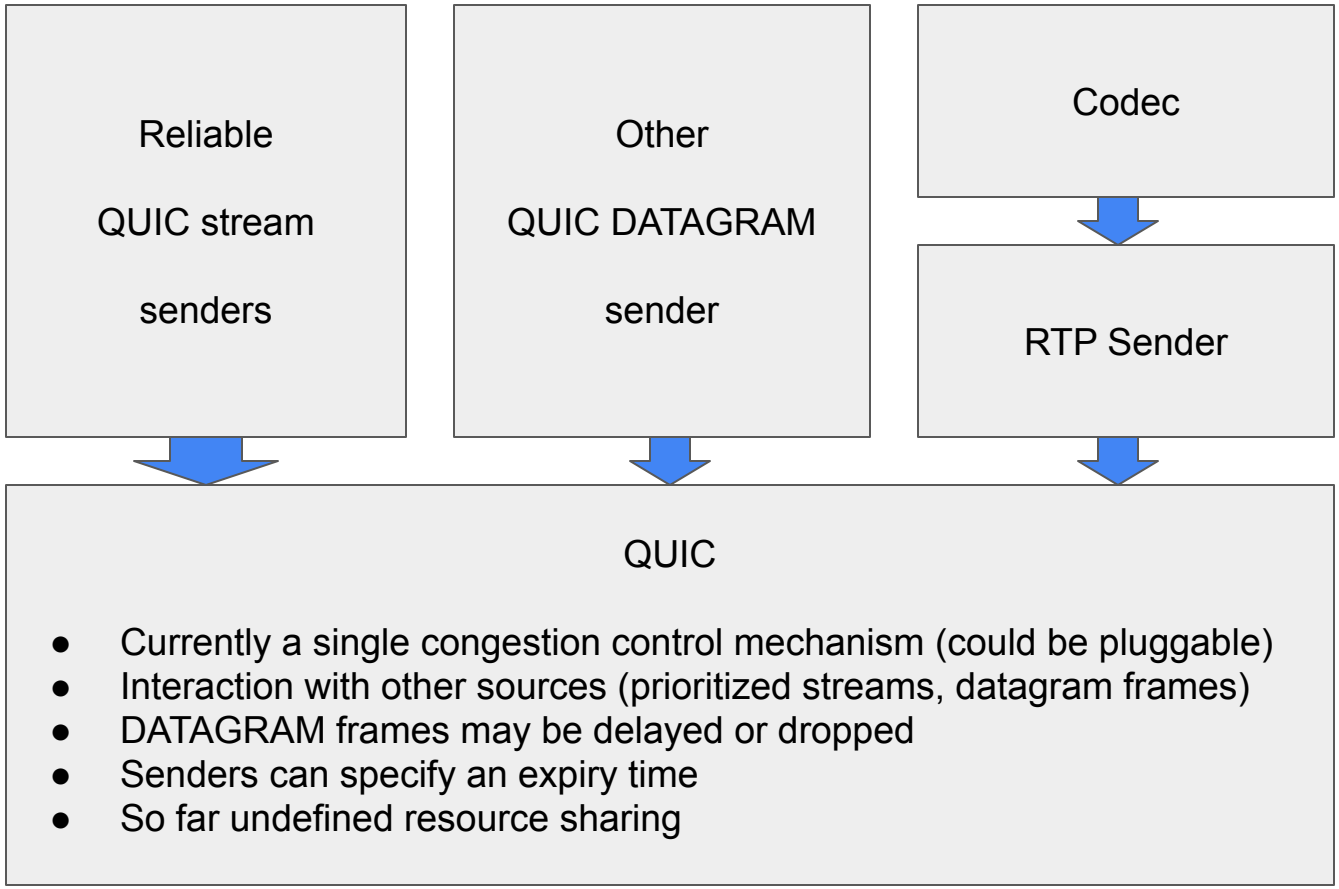
- Connection-oriented operation [RFC 8122]
- Bundling of RTP sessions into a single transport connection [RFC 8843]
- Multiplexing RTP and RTCP [RFC 5761]
  - Possibly rtcp-mux-only [RFC 8858]
- Example

```
a=group:BUNDLE abc xyz

m=audio 10000 QUIC/RTP/AVP 0 8
a=setup:actpass
a=connection:new
a=fingerprint:SHA-256 \
12:DF:3E:5D:49:6B:19:E5:7C:AB: … :DF:3E:5D:49:6B:19:E5:7C:AB:4A:AD
b=AS:200
a=mid:abc
a=rtcp-mux
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=extmap:1 urn:ietf:params:<tbd>

m=video 0 QUIC/RTP/AVP 97
b=AS:1000
a=bundle-only
a=mid:xyz
a=rtcp-mux
a=rtpmap:97 H264/90000
a=extmap:2 urn:ietf:params:<tbd>
```

Meta question 1: congestion control interaction?



Meta question 2: hybrid reliability for video ingest?

Real-time forwarding vs. media storage

- DATAGRAM frames with optional RTP / application layer reliability
- STREAM frames (1 stream per media “frame”)
- Likely more to explore

Implications of congestion control interaction on real-time properties?

Design orthogonal to header+payload formats

Wrapping up...

Rich resources available for RTP

Designed for UDP, but works on well-provisioned networks even on reliable connections

Different mappings conceivable for QUIC (some explored before)

May (or may not) need (marginally) different solutions for different purposes

Common basis useful for interop, to ease coding, ...