Chair of Connected Mobility
Department of Computer Science
Technische Universität München

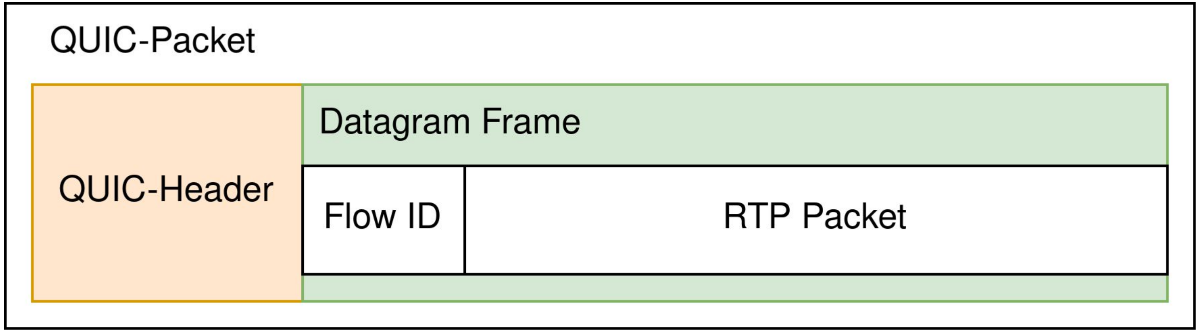
Congestion Control for RTP over QUIC

M. Engelbart and J. Ott

Chair of Connected Mobility
Department of Computer Science
Technische Universität München

RTP over QUIC

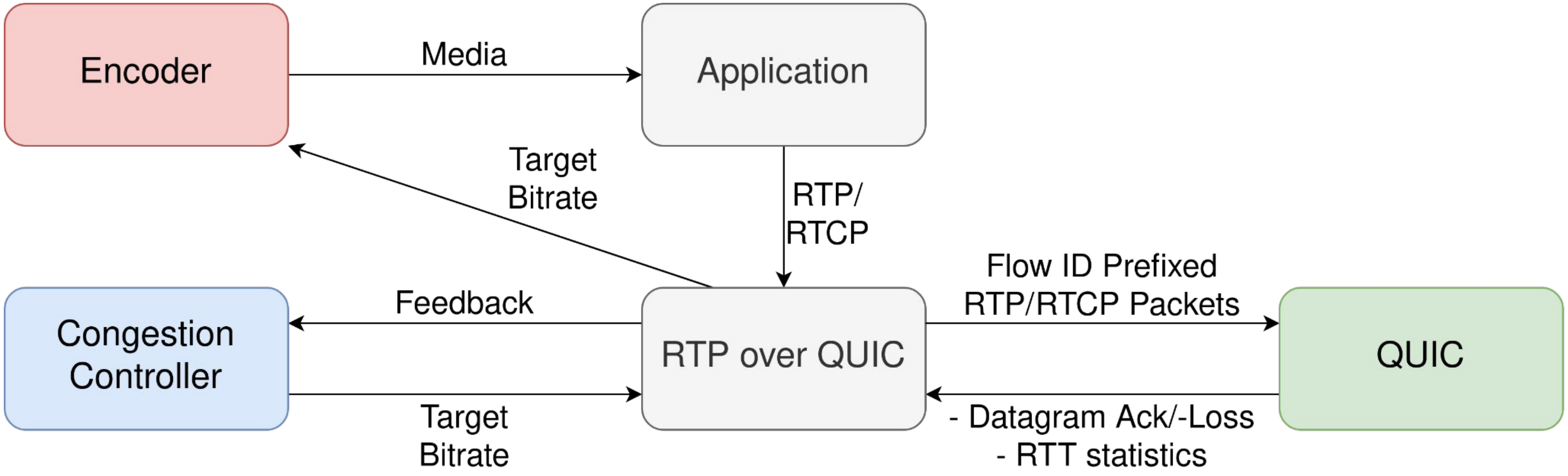
- Encapsulation for carrying RTP/RTCP over QUIC
- Using unreliable datagram extension
- Flow IDs to demultiplex datagram flows



Internet Drafts: draft-engelbart-rtp-over-quic-01, draft-hurst-quic-rtp-tunnelling-01

Chair of Connected Mobility
Department of Computer Science
Technische Universität München

RTP over QUIC



- QUIC Interface Requirements (Expose Acks/Loss and RTT statistics)
- Congestion Controller Requirements
  (Feedback input / Target bandwidth output)

Congestion Control in QUIC

- QUIC specifies congestion controller similar to TCP NewReno
- QUIC provides connection statistics to implement congestion control
    - RTT (latest, min, smoothed, mean-deviation)
    - acknowledgments
- Senders may choose a different algorithm, several under investigation
    - Cubic, BBR(v2)

Congestion Control in RTP

- Requires low-delay while still providing useful bandwidth
- Loss-based algorithms not applicable due to filled network queues and thus large delay (-variations)
- RMCAT proposed **SCReAM** and NADA (and GCC) to use for congestion control for real-time applications
- RTP uses RTCP to signal congestion (e.g. RFC 8888)
  - Acknowledgments
  - Arrival Timestamps

Combining Congestion Control in RTP and QUIC

- How to do proper congestion control for interactive real-time media over QUIC given RTP realizes its own congestion control?

- Use only QUIC congestion control (NewReno) combined with trivial encoder rate control

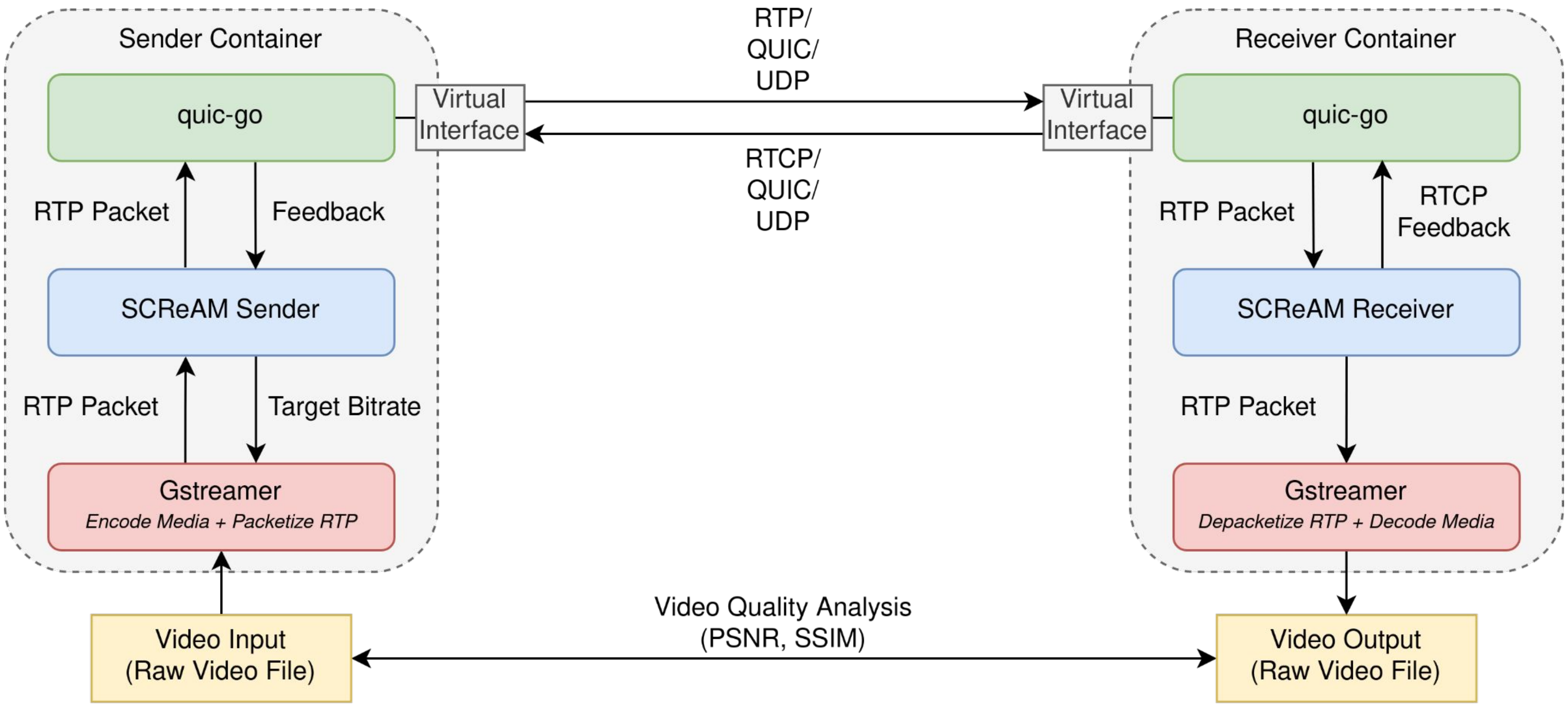
- Disable QUIC's internal congestion control and only use real-time congestion control (SCReAM)

- Use real-time and QUIC congestion control simultaneously (SCReAM+NewReno)

Congestion Signaling

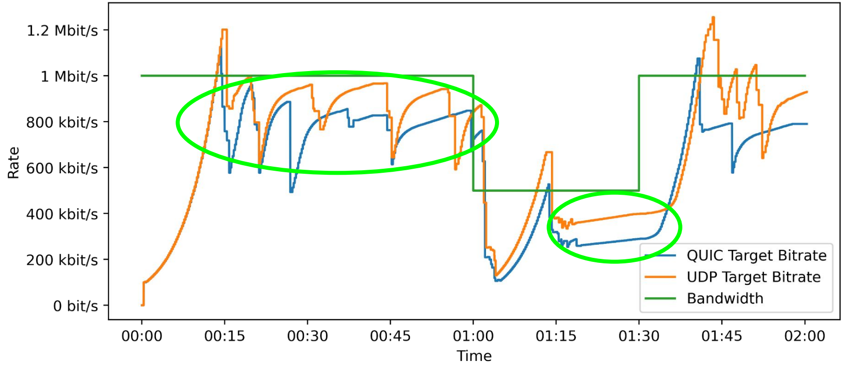
- How to avoid duplicate signaling in RTCP?
- Instead of using RTCP, feedback can be created at the sender using QUIC connection stats
- QUIC Datagram acknowledgements signal reception of RTP packets in the ACK’ed DATAGRAM frames
- Use QUIC `latest_rtt` to infer the receive time of an RTP packet:
  - `receive-ts = send-ts + latest_rtt / 2`

Implementation and Testbed

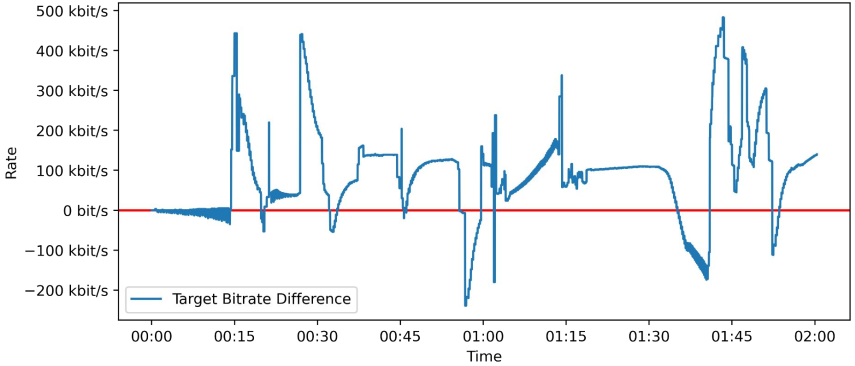


Chair of Connected Mobility
Department of Computer Science
Technische Universität München

RTP over QUIC/UDP using SCReAM



SCReAM target bitrates in kbit/s

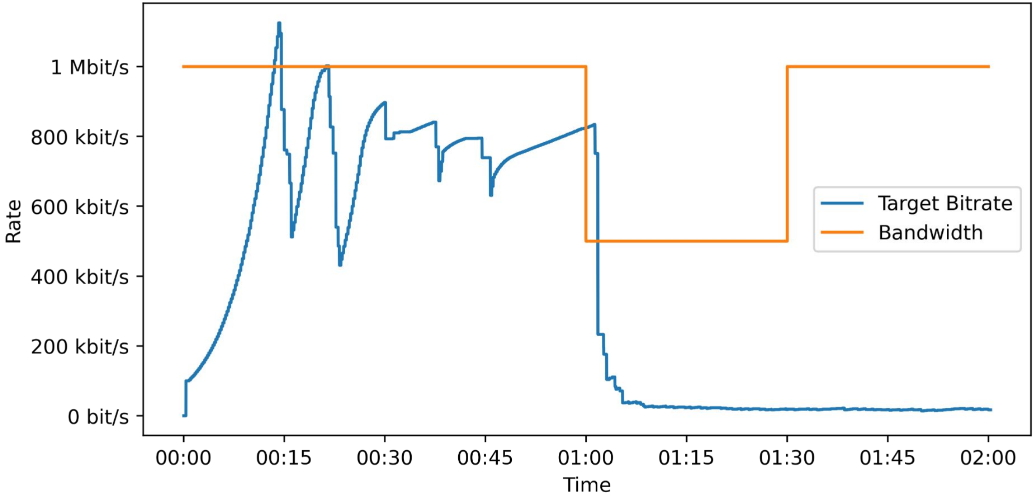


Difference of SCReAM target bitrates

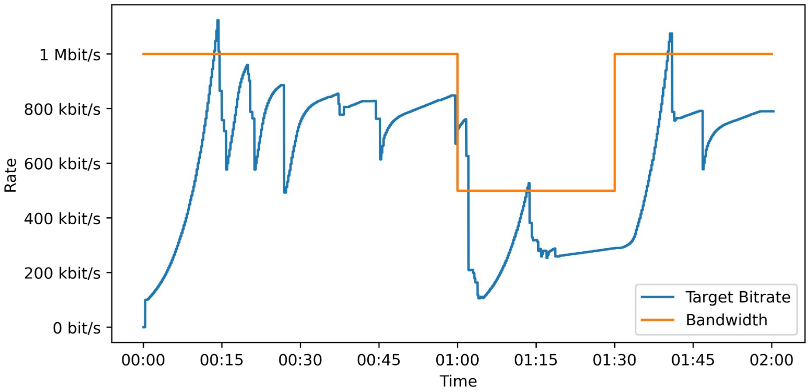
Chair of Connected Mobility
Department of Computer Science
Technische Universität München

Running SCReAM over NewReno

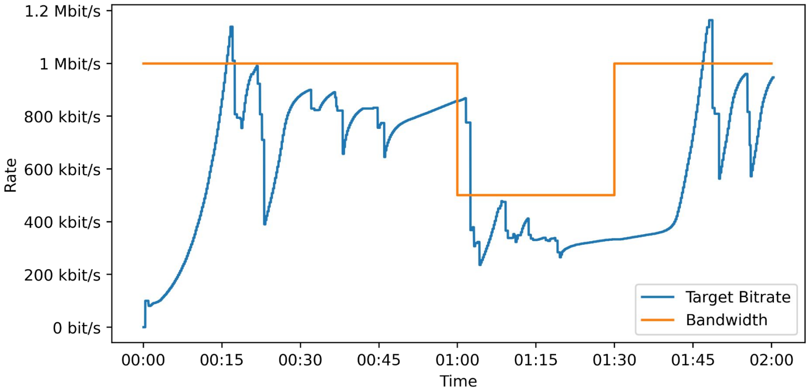
- SCReAM target bitrate similar to RTP over UDP, when application limited
- Problematic, when both congestion controllers try to adapt to bandwidth reduction



Only SCReAM and Reducing RTCP (50ms OWD)

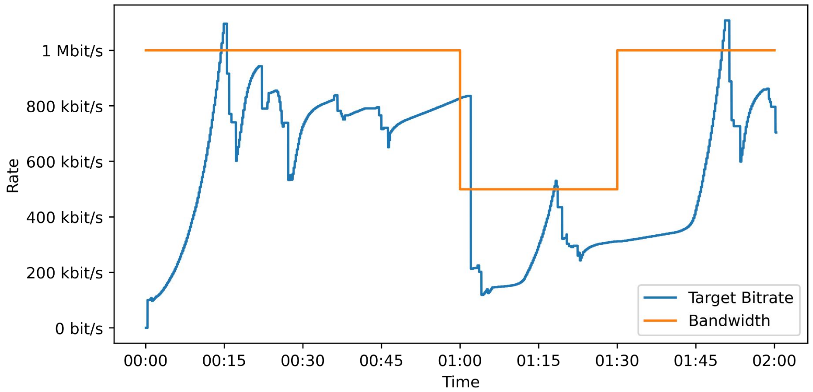


RTCP Feedback

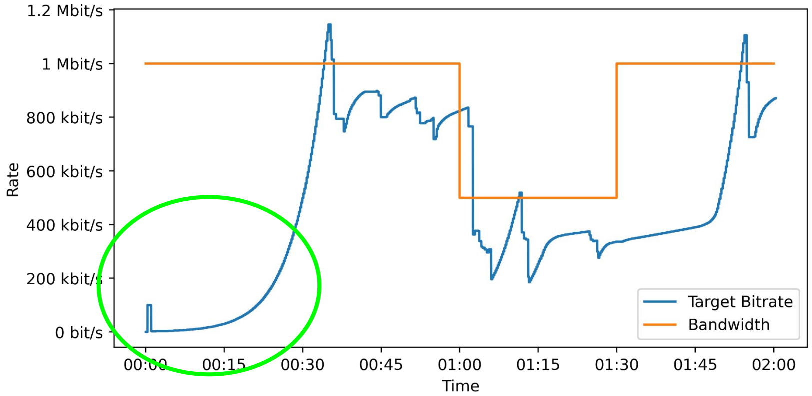


Feedback from QUIC connection statistics

Only SCReAM and Reducing RTCP (150ms OWD)



RTCP Feedback



Feedback from QUIC connection statistics

Prioritization

- We ran all experiments again while streaming data on a QUIC stream parallel to RTP in QUIC Datagrams
- Results show, that some form of prioritization is necessary
- Without prioritization, real-time streams may degrade or even starve as a function of the internal operation of the QUIC implementation
- Test cases were rather artificial, more investigation with a more natural form of background traffic needs to be done

Conclusion

Main takeaways:

1. Two separate CC loops at transport and media level (expectedly) problematic
2. We can reuse QUIC state to reduce RTCP feedback
3. Prioritization is necessary

Next steps:

- Extend testbed with new test scenarios (RFC 8867) and more CC schemes
- Come up with some prioritization scheme that gives a reasonable share of bandwidth to each real-time/non-real-time stream