TCP vs QUIC ACKs

Gorry Fairhurst
Ana Custura
(University of Aberdeen)

Summary

This slide deck presents a short analysis of QUIC ACK packets.

The current QUIC specification can suffer performance penalties compared to TCP when used over asymmetric paths because of the larger volume of ACKs compared to TCP.

This impacts paths with limited return capacity and/or return link transmission is "expensive" compared to the forward path.

QUIC’s design to prevent in-network modification means that this is something that QUIC now needs to address.

Anatomy of TCP and QUIC ACKs – From Specifications

	TCP cummulative ACK	QUIC cumulative ACK	TCP with loss	QUIC with Loss
Min	40B/2*MSS	50B/2*Packet Size	52B/MSS	52B/Packet Size
Max	52B/2*MSS	72B/2*Packet Size	84B/MSS	Packet size*
Total**	1.3%-1.7%	1.8%-2.6%	3.46%-5.6%	3.8%-Unlimited

* Depends on implementation limit (transport, section 13.2.4.)

** Assuming 1460B TCP MSS and 1340 QUIC packet size

However, this isn't the full story wrt TCP…

Testbed used for these experiments

Endpoints:

- Linux TCP
- Quicly, draft revision 27
- Chromium, draft revision 26
- PicoQUIC, draft revision 26

FreeBSD router to emulate path delay of 600ms

When required, traffic shaping emulates a 1% packet loss for forward path

Experiments transferred 10MB of data on forward path, client to server

Network traces and logs collected and stored for analysis.

Experimental Scenarios

TCP No Loss data from a 8.5/1.5 Mbps Satellite Broadband Service (includes a PEP)

TCP Loss data from a 8.5/1.5 Mbps Emulated Satellite Network

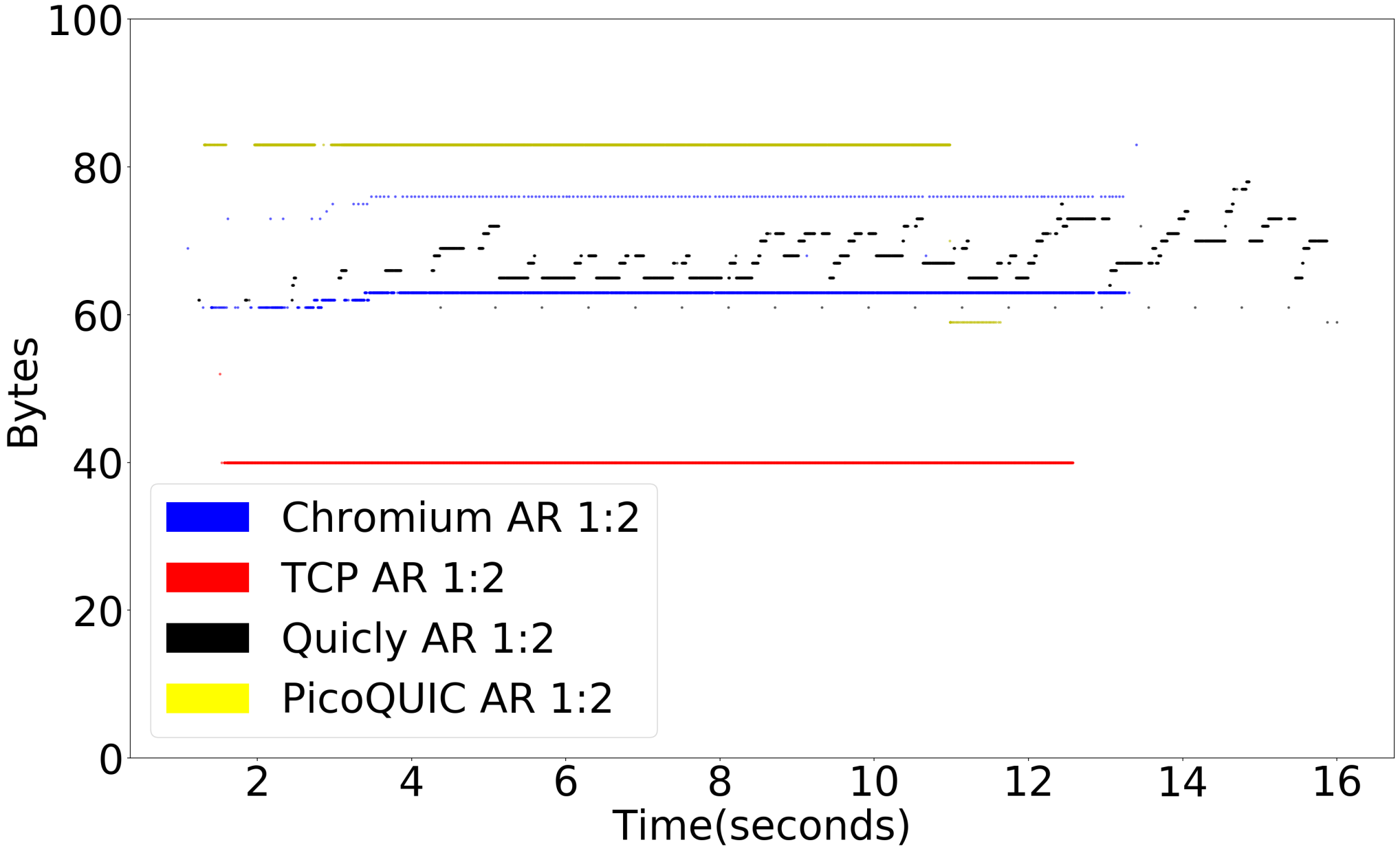
All Quicly and Chromium data from a 8.5/1.5 Mbps Emulated Satellite Network

PicoQUIC data from a 10/2 Mbps Emulated Satellite Network

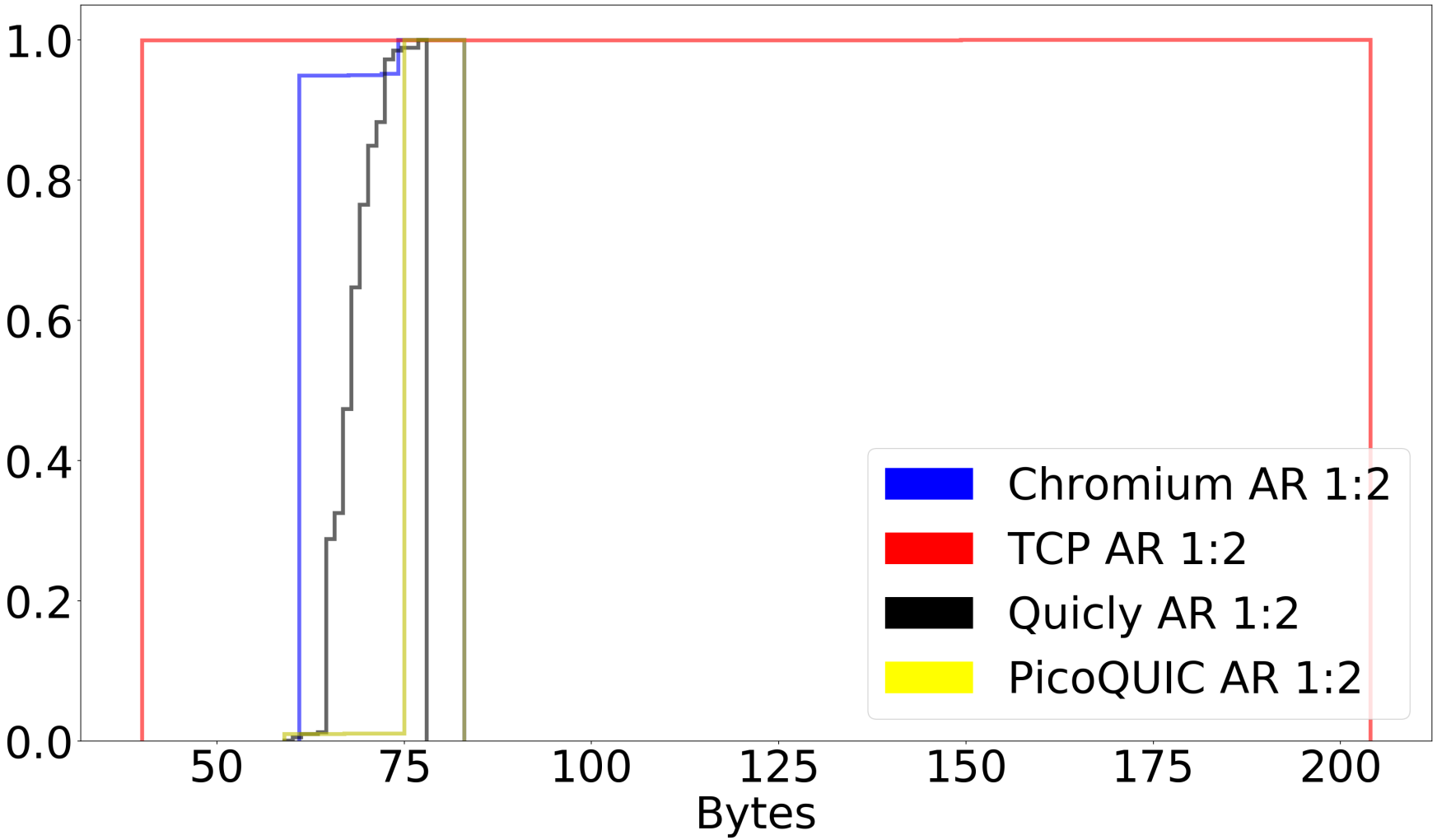
Chromium compiled with the decimation flag disabled

Quicly compiled for default ACK ratio 1:2

ACKs v Time (no loss)

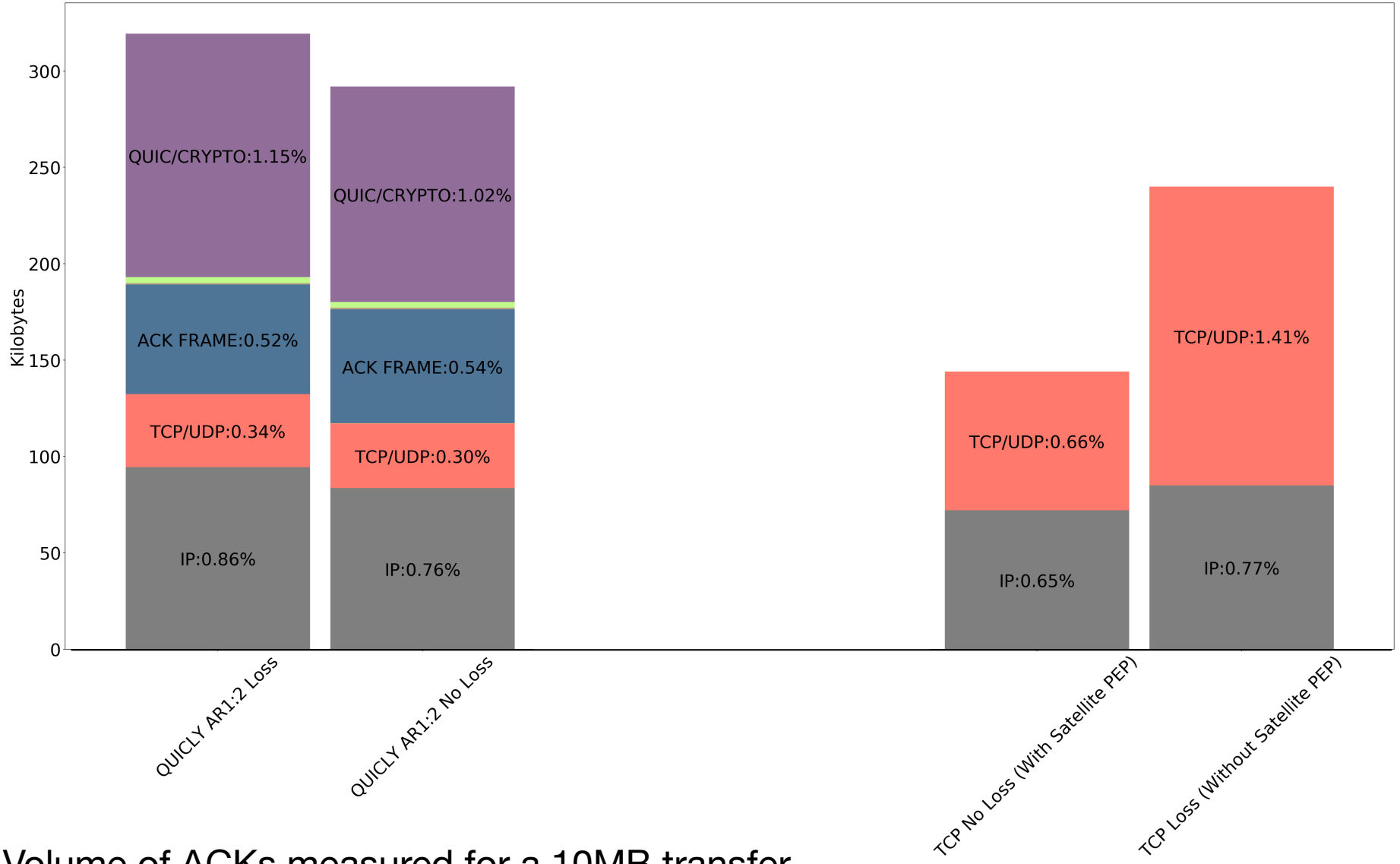


CDF of ACK sizes (no loss)



CDF of packet sizes on the return path measured after the first RTT for a 10MB transfer, with no link loss, emulated 600ms Path RTT.

Analysis of Return Traffic



Volume of ACKs measured for a 10MB transfer,
with no link loss and 1% link loss, emulated 600ms Path RTT.

Asymmetry and TCP

Some paths have limited return capacity or uses return links where transmission is ”expensive” compared to the forward path [RFC3449].

TCP ACK Filtering [RFC3449] observes ACKs and ”Thins”, redundant TCP ACKs.

- A simple method queues TCP ACKs for the same flow and removes all except except the last cumulative ACK.
- Even filtering of 2-4 can significantly reduce pressure on return path capacity, and queuing delay, benefitting any traffic sharing the bottleneck.
- These techniques are implemented (e.g., DOCSIS, Mobile, and WiFi).

Performance Enhancing Proxies [RFC3135] reduce ACK rate using a proxy to split end-to-end transport into a series of network segments, allowing a smaller ACK Ratio.

- These techniques are currently implemented (e.g., radio links, satellite broadband).

QUIC transport is ***designed*** to encrypt and authenticate their ACKs. This prevents in-network modification, and avoids this ossification by the network.

- QUIC transport needs an appropriate default ACK policy
- Copying TCP’s policy puts QUIC at a serious disadvantage for asymmetric paths

	10/2 Mbps	50/10 Mbps	250/3 Mbps
TCP - no loss	133 - 346 kbps	650 -1,730 kbps	3,250 - 8,650 kbps
TCP - loss	346 - 560 kbps	1,730 - 2,800 kbps	8,650 - 14,000 kbps
QUIC - 1:2 ACK ratio no loss	144 -438 kbps	720 - 2,190 kbps	3,600 - 10,950 kbps
QUIC - 1:2 ACK ratio with loss	290 - Unlimited	1450 - Unlimited	7,250 - Unlimited

Rate of ACK bytes required to fill a forward path
Cases where just ACKs would consume full return link capacity are highlighted in red

QUIC ACKs often bigger…

TCP is constrained by the SACK option size

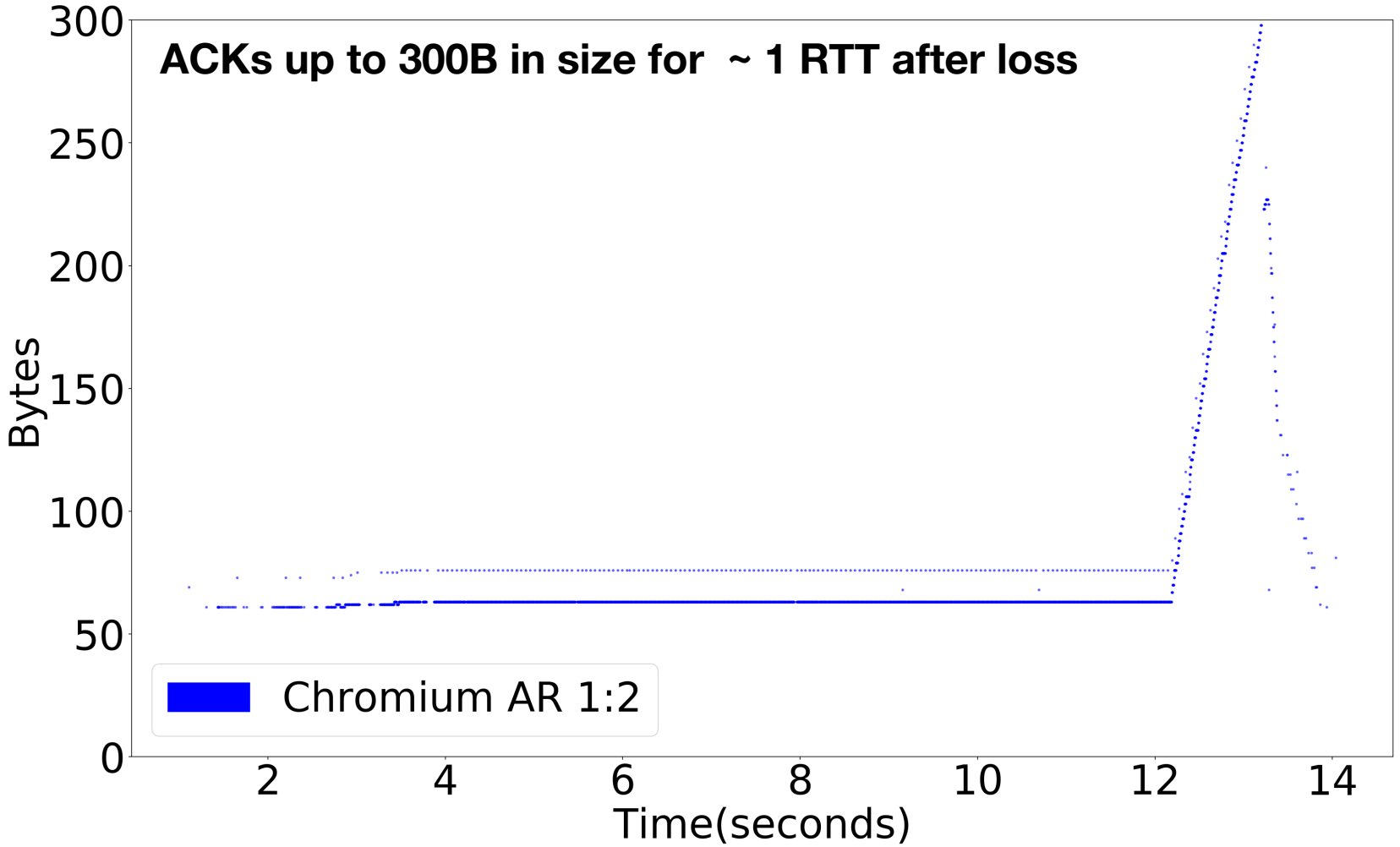
- QUIC transport's ACKs are not limited in this way
- QUIC provides better loss recovery than TCP

QUIC ACKs can carry many ACK Ranges

- QUIC return path packets can carry other (useful) information
- As far as we can tell, the implementations met the requirements of the current QUIC spec.
- QUIC ACKs sometimes are big!

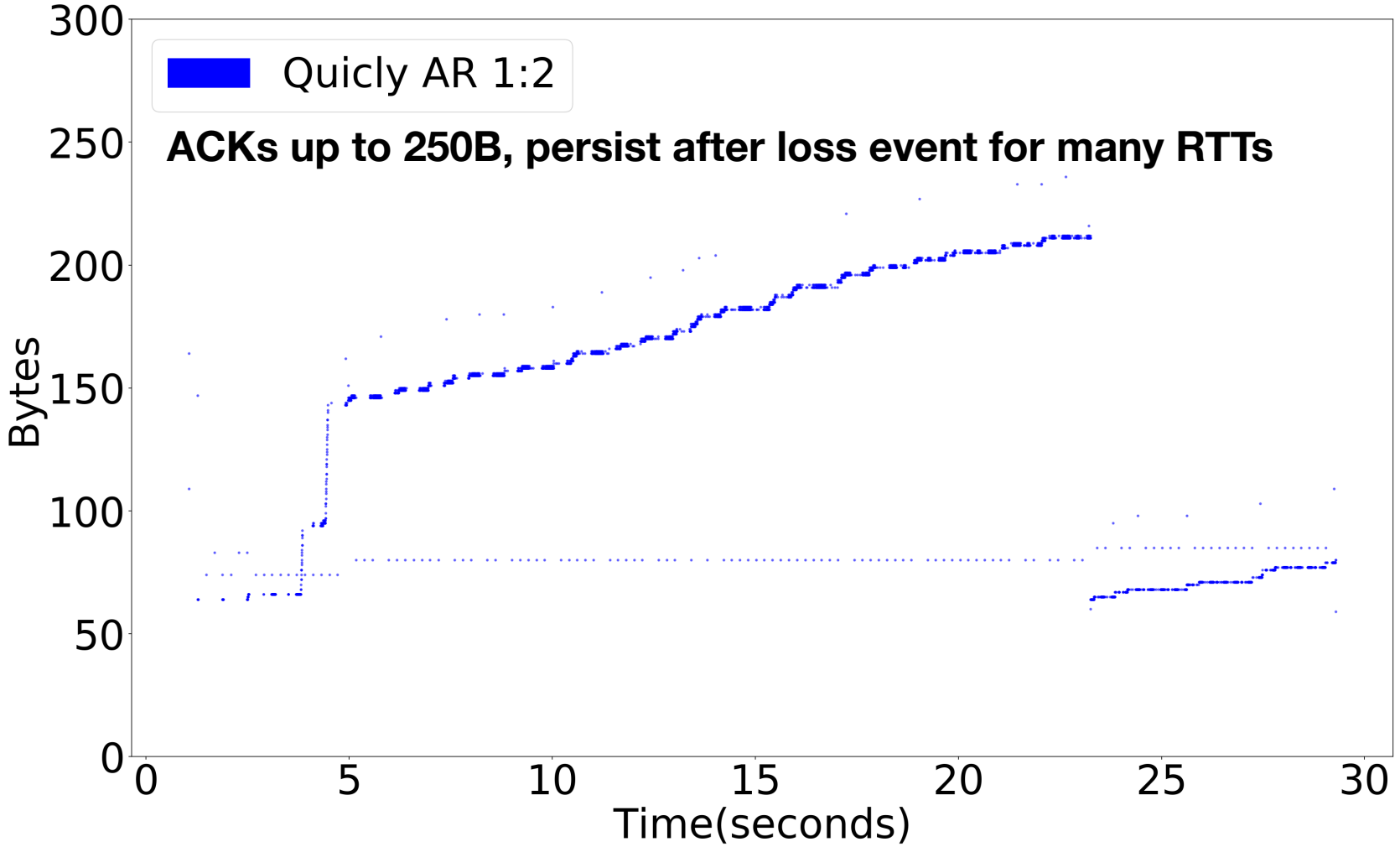
Note: The following are examples. The results are *NOT* criticism of a specific implementation and these implementations we tested continue to evolve.

QUIC Variability due to loss scenarios (Chromium)



Congestion control cases some loss
QUIC reports ACK ranges during loss/reordering
HOWEVER, we saw and expect similar results with other implementations of QUIC

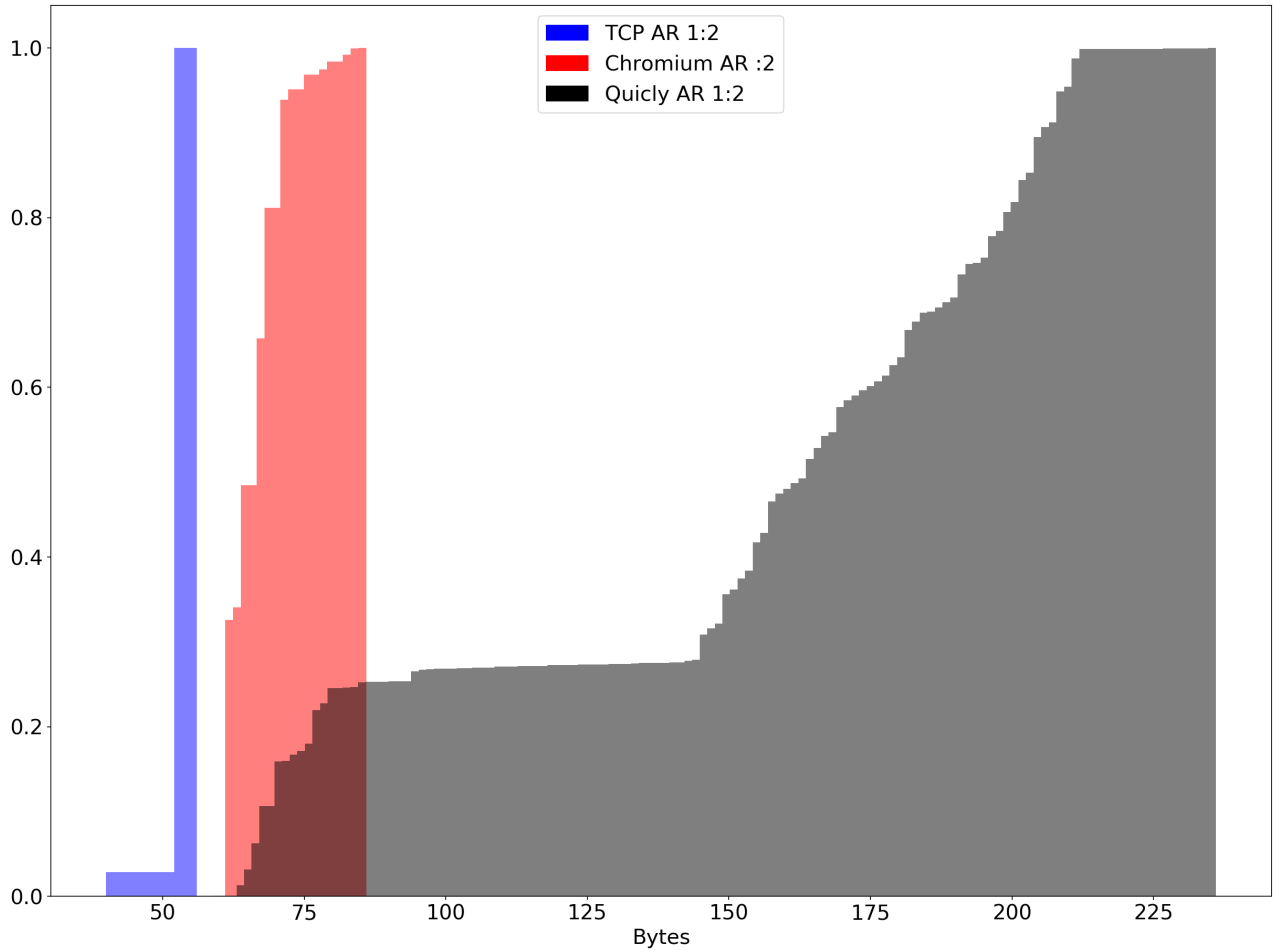
QUIC Variability due to loss scenarios



This result is when ACK ranges were not retired promptly (results actually using Quicly)
HOWEVER, we saw other results with periods of large ACKs using other implementations

Path RTT was 600mS (in this particular case, loss extended for ~25 RTTs)

Summary of QUIC ACK Variability



Size of ACKs sometimes were sometimes much larger than TCP.

Note: Different implementations have different outcomes (one is not necessarily consistently worse than the other !)

Conclusion

QUIC sends more types of frames on the return path

- TCP only sends ACKs

Each ACK is ***at least*** 1.5x-2x larger in a no loss scenario

- QUIC ACKs can be ***much, much*** larger

Current QUIC performance is compromised compared to TCP when used over asymmetric paths because of the larger volume of ACKs.

- In-network TCP ACK Thinning would not help QUIC. Total ACK traffic on an asymmetric link can ~x5 larger than for TCP with actual impact depending on the way TCP was enhanced on the link and the (radio) link properties.

QUIC transport needs a better ***default*** ACK Policy! (see separate slides)