Network Reordering

Measurements from QUIC and TCP at Google

Presenter: Ian Swett

QUIC

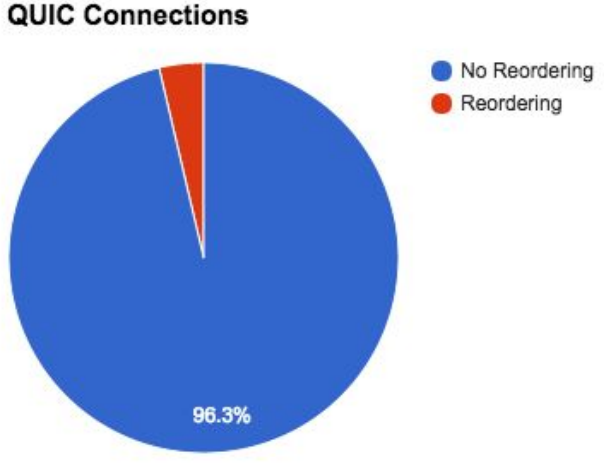
Quick **U**DP **I**nternet **C**onnections

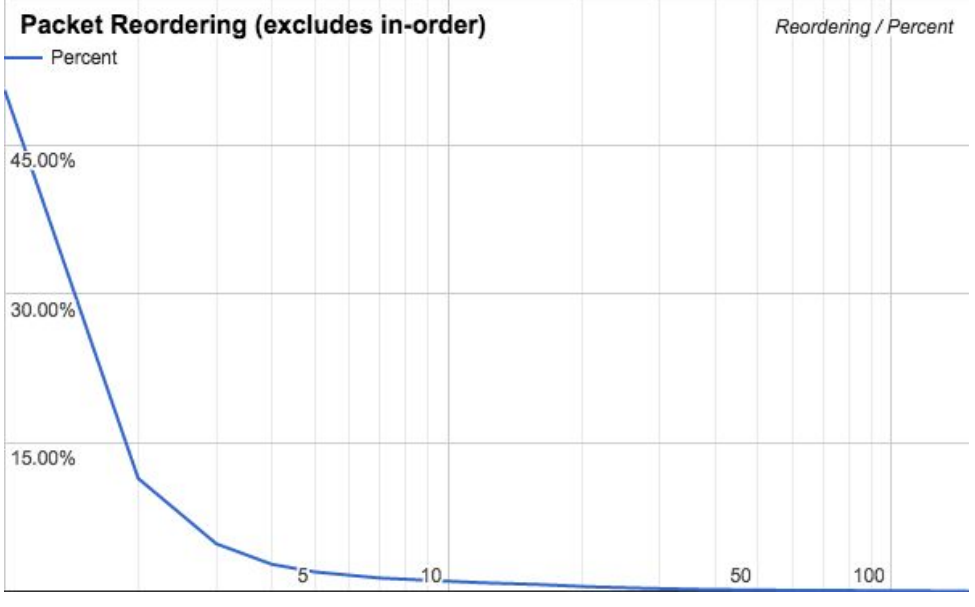
- A reliable, multiplexed transport over UDP
- Always encrypted and authenticated
- Reduces latency
- Runs in user-space
- Open sourced in Chromium

UDP Reordering - Client Side, All Services

Measuring the distribution of packet reordering

- Measured at the receiver
- Includes ack only packets, which have packet numbers in QUIC

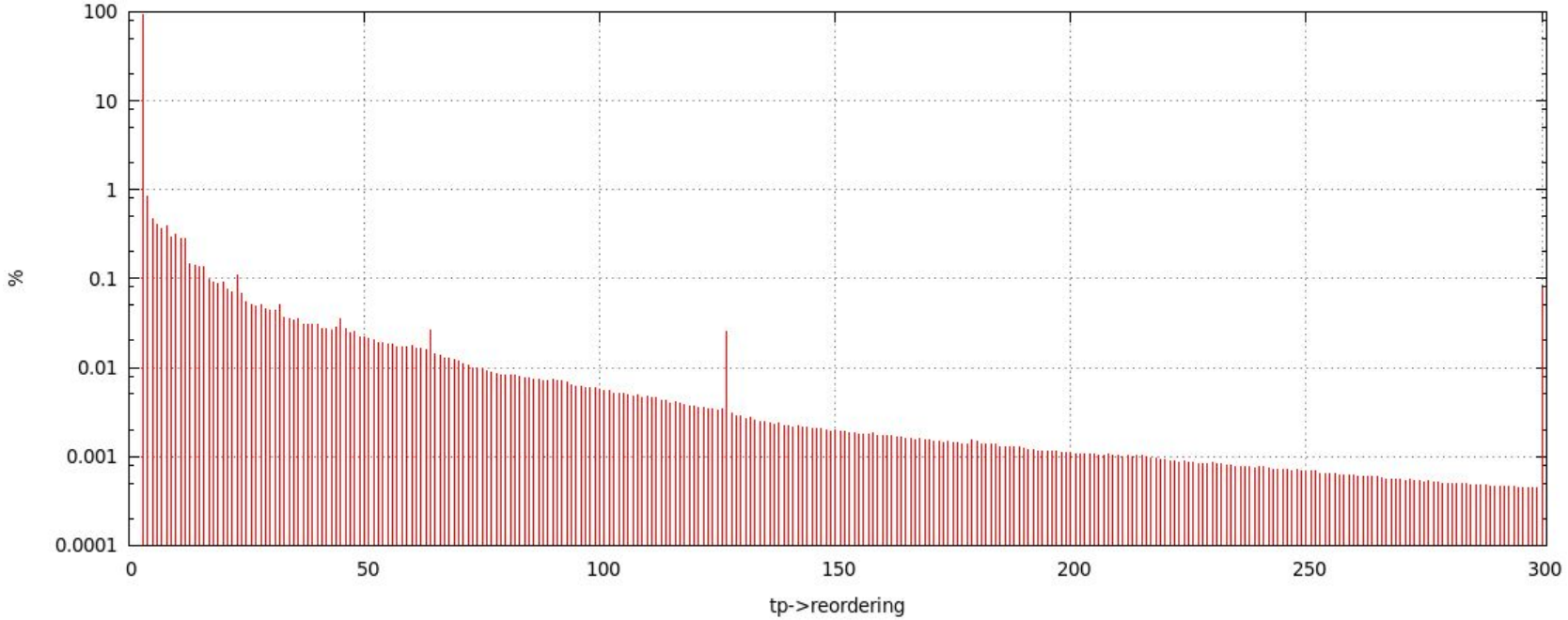




TCP Reordering - Server Side, Video Traffic

Measuring the largest packet reordering on a request

- Fewer than 7% of connections had reordering degree > 3 packets



Loss Recovery Comparison

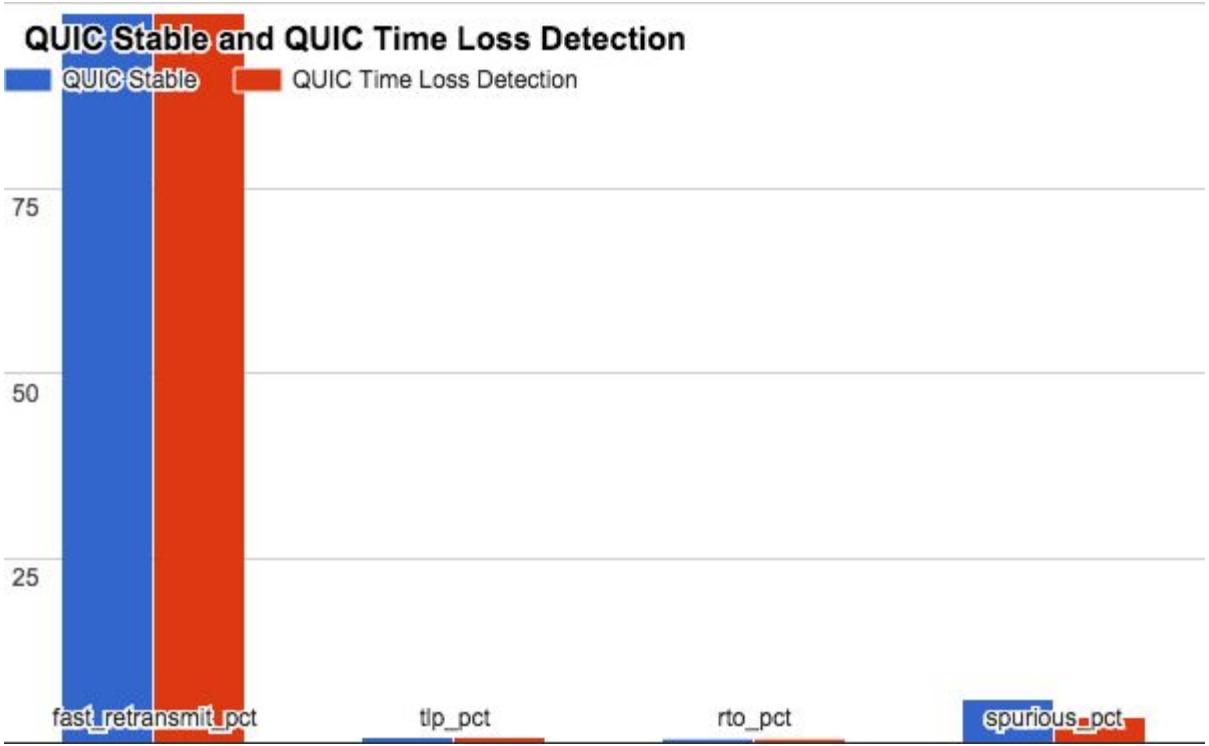
QUIC implements modern TCP features

- Fast Retransmit
- Early Retransmit with timer
- Tail Loss probe
- All QUIC loss detection is RACK-style
- Pacing is always enabled

Two Loss Recovery Algorithms

- Basis: A packet is lost if some packet sent sufficiently later is acked
  - FACK with a fixed reordering theshold of 3
  - Time based with a fixed reordering threshold of ¼ RTT

Loss Recovery Metrics - Server Side



Time based loss detection reduces spurious retransmits almost 50%

6.04% to 3.54%

Loss Recovery Metrics - Server Side

Time based loss detection reduces early slow start exit

41% to 34%

Loss Recovery Comparison - Summary

Time loss detection

- Reduces spurious retransmits by ~50%
- Reduces overall retransmit rate 1-2%
- Fewer connections prematurely exit slow start

User metrics

- Most user metrics are unchanged
- Reduces overall retransmit rate over 2% on long connections

Google

QUIC

Source: QUIC in Chromium
Page: www.chromium.org/quic
Public Mailing list: proto-quic@chromium.org
QUIC IETF draft: draft-tsvwg-quic-protocol-01
RACK IETF draft: draft-cheng-tcpm-rack-00

Thanks to Yuchung Cheng, Ryan Hamilton, and Jana Iyengar for helping me gather this data.