Goals

Discuss QUIC loss recovery mechanisms

No slides on congestion control, but can discuss
   (it’s just NewReno)

Learn about egregious errors and blind spots
   TCPM has the right experts

Increase engagement with TCPM
   can do an update again at the next IETF
Non-Goals (for the next hour)

Re-design mechanisms

Re-litigate constants

Re-litigate QUIC’s use of TCP standards

6298 and 5681 are non-normative references

... these things can be done, just don’t do them right now
Overview

Some relevant QUIC details

Recovery mechanisms

Potential improvements / Open questions
QUIC Packet Numbers

Monotonically increasing 62-bit *packet numbers*
  (caveat: multiple PN spaces during connection setup)

Packet number DOES NOT indicate delivery ordering
QUIC Acknowledgements

ACK frame is encrypted and carried within QUIC packets

ACK frame contains:
  largest acked
  one or more ack ranges
  "ack delay": T(ack send) - T(largest acked packet received)
  3 ECN counts: #ECT(0), #ECT(1), #CE
Generating ACKs

SHOULD ACK every other packet
  subject to 25ms delayed ack timer

SHOULD ACK immediately if:
  Received packet number != largest received + 1
  CE codepoint received

MAY process more packets before ACK
  allows less frequent acking
Notation

Packet: PN X  
packet with Packet Number X

Ack Frame: A X(K-L)(M-N)  
largest acked of X
ack ranges K-L and M-N
(Note: X > K, L, M, N)
Same, but different

Loss Detection
- fast retransmit, early retransmit
- tail loss probe, RTO
- spurious RTO detection

Congestion Control
- NewReno, but largest_acked ends recovery period
Recovery Mechanisms
Fast Retransmit (Packet threshold)

S

PN: 1
PN: 2
PN: 3
PN: 4
PN: 5
PN: 6 (rtx frames from PN 2)

R

A: 3 (1-1)
A: 4 (3-3) (1-1)
A: 5 (4-3) (1-1)
A: 6 (5-3) (1-1)
Fast Retransmit (FACK)

PN: 1
PN: 2
PN: 3
PN: 4
PN: 5
PN: 6
(rt x frames from PN 2)

A: 3 (1-1)
A: 4 (3-3) (1-1)
A: 5 (4-3) (1-1)
A: 6 (5-3) (1-1)
Fast Retransmit (Time threshold)

\[ \frac{9}{8} \times \text{max}(sRTT, \text{latest RTT}) \]

PN: 1
PN: 2
PN: 3
PN: 4
PN: 5
PN: 6
PN: 7

A: 3 (1-1)
A: 4 (3-3) (1-1)
A: 5 (4-3) (1-1)
A: 6 (5-3) (1-1)
A: 7 (6-3) (1-1)

(rtx frames from PN 2)
Fast Retransmit (Time threshold)

Time threshold allows reordering tolerance in packet space
Early Retransmit

Small delay allows for some reordering
RTT and Timeouts

RTT is RFC 6298, except for RTT sample:

\[
\text{rtt} = \text{now} - \text{largest_acked.sent_time} - \text{ack.ack_delay}
\]

max_ack_delay

declared by both endpoints during handshake

Timeouts:

\[
\text{RTO} = \text{srtt} + 4 \times \text{rttvar} + \text{max_ack_delay} \quad (\text{min: 200ms})
\]

\[
\text{TLP} = 1.5 \times \text{srtt} + \text{max_ack_delay} \quad (\text{min: 10ms})
\]
TLP

3/2 * SRTT + max_ack_delay

PN: 1
PN: 2
PN: 3
PN: 4

(TLP: send new data or retx)

A: 2
A: 4 (2-1)

TLP always includes max_ack_delay
2 TLPs

3/2 * SRTT + max_ack_delay

3/2 * SRTT + max_ack_delay

PN: 1
PN: 2
PN: 3

PN: 4
(TLP)

PN: 5
(TLP)

A: 2

A: 5 (2-1)
max(SRTT + 4*RTTVar + MaxAckDelay, MinRTO)
Spurious RTO Detection

RTO verified!
Spurious RTO Detection

RTO spurious!
Spurious RTO

No congestion control actions on RTO

If any packet sent prior to RTO is newly acked
  declare RTO as spurious
  nothing more to be done

If all packets acked are ones sent after RTO
  declare RTO as verified
  congestion control actions

(open issue: #1966)
Crypto Timeout

Set aggressively
before RTT sample: 200 ms
after RTT sample: 2 x smoothed RTT
set to max(timeout, kMinTLPTimeout)

Exponential backoff on consequent timeouts

Retransmit all outstanding crypto packets on timeout
Potential Improvements
(NOT IN DRAFT!)
Generating fewer ACKs

SHOULD be sent immediately upon receipt of a second packet. Wireless drivers, middleboxes compress TCP acks. Should QUIC generate acks less frequently by default?
Removing MinRTO ([1017])

MaxAckDelay is explicitly communicated in the handshake

TCP's minRTO was to avoid spurious RTOs (RFC 6298)
- primary cost is bandwidth collapse when timer fires
- spurious RTO detection eliminates this cost

QUIC could remove the MinRTO
- since spurious RTOs have substantially lower cost
Potential Timeout Simplification

Combine TLP and RTO

both are similar, but no practical difference in QUIC

Issue

TLP is commonly spurious

Why different than TCP

cost of spurious RTO and TLP is low in QUIC
Fast retransmit

Should we do adaptive time thresholding?

How do we best use both packet and time thresholds?

working on this now