Motivation

• In some cases TCP/SCTP must use RTO for loss recovery
  – e.g., if a connection has 2 outstanding packets and 1 is lost

• Some solutions exist, but they are not always applicable
  – Limited Transmit (RFC 3042)
    • requires: unsent data, no ack loss
  – Early Retransmit (RFC 5827)
    • requires: 2 outstanding segments, no ack loss, no reordering
Motivation

• Thus, some flows have to use RTO for loss recovery

• However, the effective RTO often becomes $RTO = RTO + t$
  – Where $t \approx RTT + delACK$

• The reason is that the timer is restarted on each incoming ACK (RFC 6298, RFC 4960)
Impact

• The extra RTT could lead to performance problems for short-lived (e.g. web) and thin streams
  – Thin streams are flows that only use a fraction of the available bandwidth (e.g. signaling, online games, chat, VoIP, …)

• 80% of all web flows typically contain 7-8 segments or less [1], which is similar to general TCP flow lengths [2]
  – 2-3 RTTs in slow-start
  – RTO ≈ 4 RTTs (Linux and Windows) [3]

• It has previously been shown that web flows use RTOs frequently to recover lost packets [4]

Impact

• Standard approach no problem when congestion window is large
• Actually, it can be beneficial
  – lower risk for spurious RTOs
  – gives FR more time to detect loss
    • smaller congestion window reduction using FR
• This is not the case for short-lived/thin flows
  – congestion window low anyhow
TCP and SCTP RTO Restart

• To allow retransmissions after exactly RTO seconds, the timer is restarted as:
  – $RTO = RTO - t$

• The modified restart is only used when
  – the number of outstanding segments < 4;
  – and there is no unsent data ready for transmission.

• Thus, only flows incapable of FR can use the modified RTO restart
Costs vs. Benefits

• Benefits
  – reduces RTOs with approximately one RTT for flows incapable of FR
  – isn’t more aggressive than allowed by RFC 6298, RFC 4960

• Costs
  – more aggressive than the current algorithm
  – requires an extra variable per outstanding segment
Comments...

- We have implemented the algorithm in FreeBSD/Linux for TCP/SCTP
- Comments?