TCP and SCTP RTO Restart

draft-hurtig-tcpm-rtorestart-03

Michael Welzl
michawe@ifi.uio.no

TCPM, 85th IETF Meeting
6. 11. 2012
The problem

- Sometimes, RTO must be used for loss recovery
  - e.g., if a connection has 2 outstanding packets and 1 is lost
- However, the effective RTO often becomes
  \[ RTO = RTO + t \]
  - Where \( t \approx RTT + \text{delACK} \)
- Because the timer is restarted on each incoming ACK
  (RFC 6298, RFC 4960)
TCP and SCTP RTO Restart

• To allow retransmissions after exactly RTO seconds, the timer is restarted as $RTO = RTO - t$
  – Requires storing the 4 most recent timestamps

• 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 modified RTO restart

• Risk of spurious timeout seemed low in initial experiments

• Note: this algorithm is still the same
  – Variations were proposed and discussed, but dismissed
RTO restart vs. TLP

• RTO restart
  – focus on thin streams, limited to max. 4 outstanding segments
  – makes the RTO timer expire after exactly RTO seconds more often

• TLP
  – send up to two "probe segments" when a different (Probe Timeout, PTO) timer fires
  – PTO timer is set to 2 RTTs, which is sometimes smaller than the RTO
    • Spurious PTO is not a big deal compared to spurious RTO
Example: thin stream

• App-limited case
  – TLP: a "probe segment" is a new segment if new data is available, *else a retransmission*
  – PTO applies to max. 2 packets, RTO restart applies to max. 3 packets
  – PTO fires after 2 RTTs, RTO restart fires after ("correct") RTO
Example: repeated web downloads

• End of a larger window
  – TLP: PTO timer fires after 2 RTTs
  – Spurious PTO: send up to 2 unnecessary packets
  – Spurious RTO from RTO restart: impact worse, but:
    • Unlikely because of RTO variation “safety net”
    • Spurious RTO’s effect limited if application’s sending pause > RTO (cwnd should become RW anyway)
Backup slides
What if we use both?

- TLP could kick in in situations where RTO restart does not
- TLP could overrule (= retransmit packets earlier) RTO restart in cases where (# outstanding segments < 4) and no new segments are available for transmission.
- RTO restart reduces the probability that TLP is activated because PTO might be farther than RTO
Faster Recovery Needed?

• One 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, …)

• Example: Anarchy Online [1]
  - Approx. 1% packet loss
  - Most loss recovered using RTOs
  - Maximum tolerable latency about 500 msec [2]

Performance

• Initial simulations
  – Ns-3 (with real Linux TCP)
  – Short-lived flows
  – Multiple clients served by one host
  – Large set of bw’s and delays
• Results show that
  – Loss recovery times are reduced with approximately 1 RTT on average
  – The amount of spurious RTOs is slightly higher than for regular TCP (<1% more)
• New experiments underway
  – Congestion losses
  – New RTO management alg.
  – To investigate burst situations more thoroughly

Results from 200 concurrent flows with 100 ms RTT