Different TCP Flavors

CSCI 634, Fall 2010

TCP Congestion Control

- Slow-start
- Congestion Avoidance
- Congestion Recovery
  - Tahoe, Reno, New-Reno
  - SACK
How to Detect a Packet Loss

- Timeout
  - All behave as Tahoe after timeout
  - Set ssthresh to cwnd/2
  - Set cwnd to 1
  - Entering slow-start
- Fast Retransmit (3 dup ACKs)
  - Different behaviors tell them apart

Fast Retransmit

- When can duplicate ACKs occur
  - loss
  - Packet re-ordering
- Assume packet re-ordering is infrequent
  - Use receipt of 3 dup acks as indication of loss
  - Retransmit that segment before timeout
Tahoe’s behavior

[Diagram of Tahoe's behavior with congestion window and time axis]

Tahoe TCP

[Fall96a] figure 3

nothing else is happening here

fast retransmit after 3 dup ACKs
TCP Reno

- Fast Retransmit+Fast Recovery
  - When 3 dup ACKs are received, retransmit the packet and reduce cwnd to half (fast retransmit)
  - Each dup ACK indicates a packet has left the network, increase cwnd by one for each dup ACK (fast recovery)

Fast retransmit and recovery

- If we get 3 duplicate acks for segment N
  - Retransmit segment N
  - Set ssthresh to 0.5×cwnd
  - Set cwnd to ssthresh+3

- For every subsequent duplicate ACK
  - Increase cwnd by 1 segment

- When new ACK received
  - Reset cwnd to ssthresh (resume congestion avoidance)
Problem with Reno

- Multiple packet losses within a window of data
  - terminates recovery prematurely
  - deflates cwnd back to ssthresh
  - detection of second loss relies on another fast retransmission
    - but with much less incoming dup ACKs
    - much less new data packets being sending out
    - lose self-clocking
TCP New-Reno

- Remember the outstanding packets at the start of fast recovery
- If a new ACK is partial ACK, immediately retransmit the following packet, and does not exit the recovery phase
- Upon two dup ACKs, send out one new data packet, keep the “flywheel” running
Partial ACK

- A partial ACK is an ACK that acknowledges some but not all of the packets that were outstanding at the start of the fast recovery period.

- New-Reno idea: use partial ACKs to stay in fast recovery and fix more lost segments.

![New-Reno TCP Diagram]

- Fast recovery due to add’l dup ACKs.
- Additional fast retx and recovery from New Reno.
- Fast retransmit after 3 dup ACKs.
**Drawback with TCP New-Reno**

- Recover a single loss per RTT
- There are diminishing returns

**TCP SACK**

- Implements selective ACK option in TCP header
  - Contains two/three SACK blocks
  - Each block reports non-contiguous set of data
- Can recover more than one packet losses per RTT since sender now knows which packets are dropped
SACK vs New-Reno

Sack TCP

New-Reno TCP