TCP for Large Congestion Windows

Sally Floyd
May 13-14, 2002
End-to-End Research Group

Joint work with Deb Agarwal, Sylvia Ratnasamy, Scott Shenker, Evandro de Souza, and others.
The problem:

- TCP’s average congestion window is roughly $1.2/\sqrt{p}$ packets.

- Maintaining an average cwnd of at least $1.2 \times 10^k$ packets requires a packet loss/corruption rate of at most $10^{-2k}$.  
  - E.g., a *bit* coorruption rate of at most $1.5 \times 10^{-2k-3}$.

- Given 1500-byte packets and a 100 ms RTT, filling a 10 Gbps pipe would correspond to a congestion window of $W = 83,333$ packets.  
  - At least 1.6 hours between packet drops.

- How much better can we do, given only the current feedback from routers?
Is this a pressing problem?

• Nope. In practice, users do one of the following:
  – Open up $N$ parallel TCP connections; or
  – Use MulTCP (roughly like an aggregate of $N$ virtual TCP connections).

• However, we think it is possible to do much better, with:
  – Better flexibility (no $N$ to configure);
  – Better scaling;
  – Better slow-start behavior;
  – Competing more fairly with current TCP (for environments where TCP is able to use the available bandwidth).
The approach: use a modified response function.
Simulations in NS:

- ./test-all-tcpHighspeed in tcl/test.

- The parameters specifying the response function:
  - Agent/TCP set low_window_ 31
  - Agent/TCP set high_window_ 83000
  - Agent/TCP set high_p_ 0.0000001

- The parameter specifying the decrease function at high_p_:
  - Agent/TCP set high_decrease_ 0.1
Relative fairness:

Relative Fairness \( \frac{0.11}{p^{0.32}} \)

- Highspeed TCP / Regular TCP, Sending Rates
- Loss Rate \( P \)
Another issue: modifying slow-start:

- Slow-starting up to a window of 83,000 packets doesn’t work well.
  - Tens of thousands of packets dropped from one window of data.
  - Slow recovery for the TCP connection.

- The answer:
  - Agent/TCP set max_ssthresh\_N
  - During the initial slow-start, increase the congestion window by at most N packets in one RTT.