Internet Protocols
Fall 2004

Lecture 19

HTTP/TCP Interactions

Andreas Terzis

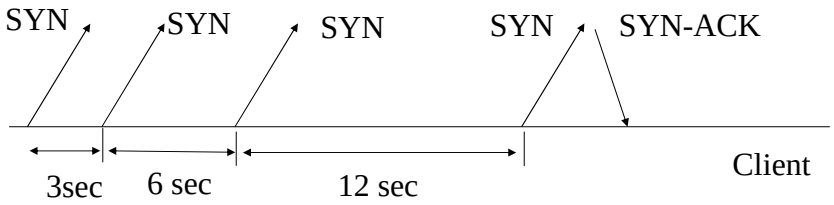
Background

- TCP standardized in 1980
- Early TCP applications very different from HTTP
  - Telnet uses a single long lasting connection
  - FTP uses separate control and data connections
  - FTP connections longer than HTTP connections

TCP Timers

- TCP uses timers to trigger protocol operations
  - Retransmission timers
  - Repeating slow-start phase
  - Reclaiming state from terminated connections
- HTTP is an interactive application
  - Timers translate to additional delay perceptible by end-user
  - Contrast with email, FTP, telnet

The effect of retx timer



- Initial RTO is 3 sec
  - RTO doubles for each lost packet
  - Losing two SYNs means additional delay of 9 sec
- Congestion causes (still) causes lost packets
  - Edge links, inter-domain links
  - SYNs lost on busy web server

The effect of retx timer (II)

- Frustrated user stops and reloads page
  - Reduced end-user perceived delay
  - At the expense of higher load on web-server and network

The effect of retx timer (III)

- After connection is established negative effect of RTO is reduced
  - RTO converges to "correct" values
  - Duplicate ACKs
- However!
  - Most HTTP responses are 8-12KB
  - Connection may not exit from slow-start phase
  - Window is not big enough to generate duplicate ACKs
  - Connection stalls, window shrinks back to 1

Slow-Start restart

- Persistent connections reduce the impact of short HTTP transfers
  - TCP window opens to "correct" value
- Connection might be idle for some time
  - During this interval congestion levels might have changed
- Slow Start restart mechanism
  - After connection is idle for some time (a few RTTs) reduce cwnd to initial value

Reducing penalty of slow start restart

- Different solutions
  - Disable slow start restart at the web server-> increased network congestion
  - Gradually decrease the cwnd
  - Pace the transmission of packets

TIME_WAIT state

- Busy web servers have *lots* of TCP connections -> would like to reclaim them ASAP
- Server has to wait for 2*MSL (1-4min) before releasing connection
  - Same scenario occurs for busy proxies

Solutions

- Reduce the overhead of keeping connections in TIME_WAIT state
  - Memory and timers
- Change TCP
  - Move the burden of TIME_WAIT to receiver of FIN
- Change HTTP
  - Allow client to close connection (after server suggestion)

Aborting HTTP connections

- Users abort HTTP transfers by pressing "stop" button
- No "Abort" method in HTTP
  - Browser needs to terminate TCP connection
  - Have to go through 3-way handshake and slow-start again…
  - What happens if request changes state on HTTP server???
  - Extra complications at proxies using pipelining

Nagle's algorithm

- TCP sender transmits at most one small packet per RTT.
  - Small means smaller than the MSS (1460 bytes)
  - Avoid high overhead from interactive applications (e.g. telnet)

Effect on HTTP

- Web server sends response using two write()'s
  - One for header one for body
  - Assume response < MSS
  - TCP sends first segment
  - Second segment has to wait for RTT
    - Unless HTTP server closes the connection
  - Second segment is sent only after client sends ACK

Effect on HTTP (II)

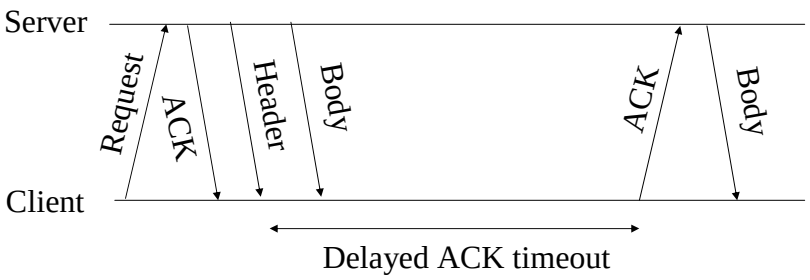
- Disable Nagle's algorithm
  - setsockopt(TPC_NODELAY)
- Negative effect
  - Increases load on web server

Delayed ACKs

- Motivation
  - Sending ACKs increases protocol overhead
- Solution
  - Don't send ACK for each packet
  - Piggyback
    - Delay transmission of ACK in hope that data will be sent (200ms)

Interaction with HTTP traffic

- It is *unlikely* that both hosts would be transmitting data



Multiplexing TCP connections

- Motivations for parallel connections
  - Browser downloading embedded images
  - Proxy acting on behalf of multiple clients
  - Higher throughput by transmitting aggressively

Multiple connections effects

- Problems
  - Unfairness to other clients
  - Higher network and server load
  - Higher user-perceived latency
- Remedies
  - Removing performance incentives for parallel connections (network and end-host solutions)
  - Provide alternatives to parallel connections