HTTP and the Web (but not HTML)

- Focus
  - How do Web transfers work?
- Topics
  - HTTP, HTTP/1.1
  - Performance Improvements
  - Protocol Latency
  - Caching

Web Protocol Stacks

- To view the URL http://server/page.html the client makes a TCP connection to port 80 of the server, by it’s IP address, sends the HTTP request, receives the HTML for page.html as the response, repeats the process for inline images, and displays it.

HTTP Request/Response

- HTTP headers, like “Content-Length:”, etc.

HTTP Request/Response in Action

- Problem is that:
  - Web pages are made up of many files. Most are very small (< 10k)
  - files are mapped to connections
  - For each file:
    - Setup/Teardown
    - Time Wait table blast
    - 2RTT “first byte” latency
    - Slow Start + AIMD Congestion
    - Avoidance

The goals of HTTP and TCP protocols are not aligned.

Simple HTTP 1.0

- HTTP is a tiny, text-based language
- The GET method requests an object
- There are HTTP headers, like “Content-Length:”, etc.
- Try “telnet server 80” then “GET index.html HTTP/1.0”
  - Other methods: POST, HEAD… google for details

TCP Behavior for Short Connections

- RTT=70ms

- Figure 12 shows that, in the actual test, using a TCP connection is, in most cases, not as fast as the HTTP/1.0 test. Even a 20 RTT timeout average only about 90% as fast as the throughput available with a reasonable timeout. These numbers should be considered in the context of the test method. This figure also shows that, for the 70 ms RTT, SSL is at low small window sizes limits the throughput to make low latency time is limited.
HTTP1.1: Persistent Connections

- Idea: Use one TCP connection for multiple page downloads (or just HTTP methods)
- Q: What are the advantages?
- Q: What are the disadvantages?
  - Application layer multiplexing

Caching

- It is faster and cheaper to get data that is closer to here than closer to there.
- “There” is the origin server. 2-5 RTT
- “Here” can be:
  - Local browser cache (file system) (1-10ms)
  - Client-side proxy (institutional proxy) (10-50)
  - Content-distribution network (CDN -- “cloud” proxies) (50-100)
  - Server-side proxy (reverse proxy @ origin server) (2-5RTT)

Browser Caches

- Bigger win: avoid repeated transfers of the same page
- Check local browser cache to see if we have the page
- GET with If-Modified-Since makes sure it’s up-to-date

Consistency and Caching Directives

- Browsers typically use heuristics
  - To reduce server connections and hence realize benefits
  - Check freshness once a “session” with GET If-Modified-Since and then assume it’s fresh the rest of the time
  - Possible to have inconsistent data.
- Key issue is knowing when cached data is fresh/stale
  - Otherwise many connections or the risk of staleness
- Caching directives provide hints
  - Expires: header is basically a time-to-live
  - Also indicate whether page is cacheable or not
Proxy Caches

- Insert further levels of caching for greater gain
- Share proxy caches between many users (not shown)
  - If I haven’t downloaded it recently, maybe you have
- Your browser has built-in support for this

Proxy Cache Effectiveness

Graph showing hit rate (%) vs. cache size (GB) for different cache sizes and configurations.