CS118 Discussion 1C, Week 2

Zengwen Yuan
Bunche Hall 3156, Friday 2:00—3:50 p.m.
Outline

• Lecture review: application layer
• Project 1 spec & demo
• More examples & homework questions
Application Layer: models

- Application Architectures
  - Client-server model: Web (TCP), FTP (TCP), E-mail (TCP), DNS (UDP/TCP), RTP
  - Peer-to-Peer (P2P): BitTorrent (TCP), Tor (aka Onion Routing, TCP)
  - Hybrid model: Skype (TCP&UDP), GTalk (TCP&UDP)
Application Layer: protocols

- HTTP: a stateless protocol on top of TCP
  - HTTP is based on pull model
  - Persistent HTTP V.S. Non-persistent HTTP
  - Method Types: GET, HEAD, POST, PUT, DELETE, Conditional GET
- What if we want stateful service (e.g. shopping cart)?
- Web Caches (proxy server)
Non-persistent v.s. Persistent v.s. Pipelining
Quiz

• How many RTTs do we need to get one HTML file with 5 embedded images?
HTTP Header: request

- Request message elements:
  - Method
  - URL
  - HTTP Version
  - Header lines
  - CRLF
HTTP Header: response

• Response message elements:
  • HTTP Version
  • Status line
  • Header lines
  • CRLF
  • Data requested
Try HTTP GET yourself

- `telnet google.com 80`
- `Get / HTTP/1.0`
- `Host: google.com`
- `<Enter>`
- `<Enter>`
Cookie

- Bring statefulness into HTTP

- Components
  - Cookie header line of HTTP response message
  - Cookie header line of HTTP request message
  - Cookie file on the browser
  - Back-end database at web-site

- Types of Cookies
  - Session Cookies (eg. shopping cart)
  - Persistent Cookies (eg. authentication id and password)
  - Third Party Cookies (eg. ad tracking agencies)
Cookie: make HTTP stateful

HTTP Client

Login
POST
username=david
password=davidh

HTTP is Stateless

Cookie: SESSIONID=66C530ACAF44D1605588619ECB0C737C

HTTP Server

Set-Cookie: SESSIONID=66C530ACAF44D1605588619ECB0C737C

Login successful?
1. create session id
2. return session id in cookie
3. store session id in database

Look up Session ID
1. session match a username?
2. session still valid?

Database

SESSION ID
SessionId
Username
createDate
expireDate
lastAccessDate
Cookie: operations

Client: has a cookie file

Web server

Amazon server creates ID 1678 for user

cookie-specific action

cookie-specific action
Application Layer: protocols

- FTP: separate control from data transmission ("out-of-band")
- SMTP: protocol for email exchange between email servers
  - SMTP is based on push model
  - Mail access protocol: POP, IMAP, HTTP-based
- P2P: no always-on server, peers are intermittently connected
  - BitTorrent: tracker and torrent. Files are divided into multiple chunks.
Application Layer: protocols

- DNS: convert hostname to IP address (and more)
- A distributed and hierarchical database
  - Root DNS servers (a—m)
  - Top-level domain (TLD) servers
  - Authoritative DNS servers
  - local DNS server
Application Layer: protocols

• **DNS:**
  • What is the transport layer protocol?
  • How the scalability is achieved?
  • Who will use iterative/recursive query?
  • Why is DNS resolver needed?
Web caching: Proxy v.s. CDN

• Proxy acts both as client and server
  • What if cache is stale?
    • HTTP conditional GET

• CDN: Content Distribution Network
  • Globally distributed network of web servers
  • Stores and replicates images, videos and other files
Project 1 Demo

- How to use Makefile
- See HTTP request received by our server program
- Serve content to a standard web browser
More examples & Homework questions
More about Web caching

- cache acts as both client and server
  - server for original requesting client
  - client to origin server
- typically cache is installed by ISP (university, company, residential ISP)

why Web caching?
- reduce response time for client request
- reduce traffic on an institution’s access link
- Internet dense with caches: enables “poor” content providers to effectively deliver content (so too does P2P file sharing)
Caching example:

**Assumptions:**
- avg object size: 100K bits
- avg request rate from browsers to origin servers: 15/sec
- avg data rate to browsers: 1.50 Mbps
- RTT from institutional router to any origin server: 2 sec
- access link rate: 1.54 Mbps

**Consequences:**
- LAN utilization: 0.15%
- access link utilization = 99% **problem!**
- total delay = Internet delay + access delay + LAN delay
- = 2 sec + minutes + usecs
Caching example: fatter access link

assumptions:
- avg object size: 100K bits
- avg request rate from browsers to origin servers: 15/sec
- avg data rate to browsers: 1.50 Mbps
- RTT from institutional router to any origin server: 2 sec
- access link rate: 1.54 Mbps

consequences:
- LAN utilization: 0.15%
- access link utilization = 99% 0.99%
- total delay = Internet delay + access delay + LAN delay = 2 sec + minutes + usecs msecs

Cost: increased access link speed (not cheap!)
Caching example: install local cache

assumptions:
- avg object size: 100K bits
- avg request rate from browsers to origin servers: 15/sec
- avg data rate to browsers: 1.50 Mbps
- RTT from institutional router to any origin server: 2 sec
- access link rate: 1.54 Mbps

consequences:
- LAN utilization: 0.15%
- access link utilization = ?
- total delay = ?

How to compute link utilization, delay?

Cost: web cache (cheap!)
Caching example: install local cache

Calculating access link utilization, delay with cache:

- Suppose cache hit rate is 0.4
  - 40% requests satisfied at cache, 60% requests satisfied at origin

- Access link utilization:
  - 60% of requests use access link

- Data rate to browsers over access link
  - \(0.6 \times 1.50 \text{ Mbps} = 0.9 \text{ Mbps}\)
  - Utilization = \(0.9 / 1.54 = 0.58\)

- Total delay
  - \(0.6 \times (\text{delay from origin servers}) + 0.4 \times (\text{delay when satisfied at cache})\)
  - \(0.6 \times 2.01 + 0.4 \times (\sim \text{msecs}) = \sim 1.2 \text{ secs}\)
  - Less than with 154 Mbps link (and cheaper too!)
Case study: Netflix

1. Bob manages Netflix account

2. Bob browses Netflix video

3. Manifest file returned for requested video

4. DASH streaming

Netflix registration, accounting servers

Amazon cloud

upload copies of multiple versions of video to CDN servers

CDN servers

Application Layer
Clarifications on Byte order conversion


• Make a byte-order conversion if you have the data type longer than a byte!