Protocol Layers and Their Service Model Part II

March 31, 2020

Inserting Records Into DNS

Example: new startup "Network Utopia"

- register name networkuptopia.com at DNS registrar (e.g. Network Solutions)
- create authoritative server locally with IP address

Web caches (proxy servers)

Goal: satisfy client request without involving origin server

- user configures browser to point to a Web cache
- browser sends all HTTP requests to cache
  - if objects in cache: cache returns object to client
  - else cache requests object from origin server, caches received object, then returns object to client
- Web cache acts as both client and server
  - server for original requesting client
  - client to origin server
- typically cache is installed by ISP

Why web caching?

- reduce response time for client request
  - cache is closer to client
- reduce traffic on an institution's access link
- internet is dense with caches
  - enables "poor" content providers to more effectively deliver content

Conditional GET

Goal: don't send object if cache has up-to-date cached version

  - no object transmission delay
  - lower link utilization

- cache: specify date of cached copy in HTTP request:

  if-modified-since: date

- server: response contains no object if cached copy is up-to-date:

  HTTP/1.0 304 Not Modified

HTTP/2

Key goal: decreased delay in multi-object HTTP requests

HTTP1.1: introduced multiple, pipelined GETs over single TCP connection

- server responds in-order (FCFS: first-come-first-served scheduling) to GET requests
- with FCFS, small object may have to wait for transmission (head-of-line (HOL) blocking) behind large objects
- loss recovery (retransmitting lost TCP segments) stalls object transmission

HTTP/2: increased flexibility at server in sending objects to client:

- methods, status codes, most header fields unchanged from HTTP 1.1

- transmission order of requested objects based on client-specified object priority (not necessarily FCFS)
- push unrequested objects to client
- divide objects into frames, schedule frames to mitigate HOL blocking

HTTP/2 to HTTP/3

Key goal: decreased delay in multi-object HTTP requests

HTTP/2 over single TCP connection means:

- recovery from packet loss still stalls all object transmissions
  - as in HTTP 1.1, browsers have incentive to open multiple parallel TCP connections to reduce stalling, increase overall throughput
- no security over vanilla TCP connection

HTTP/3: added security, per-object error, and congestion control (more pipelining) over UDP

Email

Three major components:

- user agents
- mail servers
- simple mail transfer protocol: SMTP

User Agent:

- a.k.a. "mail reader"
- composing, editing, reading mail messages
- e.g., Outlook, iPhone mail client
- outgoing, incoming messages stored on server

Mail Servers:

- mailbox contains incoming messages for user
- message queue of outgoing (to be sent) mail messages
- SMTP protocol between mail servers to send email messages
  - client: sending mail server
  - "server": receiving mail server

Email: RFC 5321

- uses TCP to reliably transfer email messages from client (mail server initiating connection) to server, port 25
- direct transfer: sending server (acting like client) to receiving server
- three phases of transfer
  - handshaking (greeting)
  - transfer of messages
  - closure
- command/response interaction (like HTTP)
  - commands: ASCII text
  - response: status code and phrase
- messages must be in 7 bit ASCII

SMTP: closing observations

comparison with HTTP:

- HTTP: pull
- SMTP: push
- both have ASCII command/response interaction, status codes
- HTTP: each object encapsulated in its own response message
- SMTP: multiple objects sent in multipart message

- SMTP uses persistent connections
- SMTP requires message (header body) to be in 7-bit ASCII
- SMTP server uses CRLF.CRLF to determine end of message

Mail message format

SMTP: protocol for exchanging email messages, defined in RFC 531 (like HTTP)

RFC 822 defines syntax for email messages itself (like HTML)

- header lines, e.g.,
  - To:
  - From:
  - Subject:

  these lines, within the body of the email message area different from SMTP MAIL FROM:, RCPT TO: commands
- Body: the "message", ASCII characters only

Mail access protocols

- SMTP: delivery/storage of email messages to receiver's server
- mail access protocol: retrieval from server
  - IMAP: Internet Mail Access Protocol [RFC 3501]: messages stored on server, IMAP provides retrieval, deletion, folders of stored messages on server.
- HTTP: gmail, Hotmail, Yahoo! Mail, etc. provides web-based interface on top of SMTP (to send), IMAP (or POP) to retrieve email messages

DNS: Domain Name System

people: many identifiers:

- SSN, name, passport

Internet hosts, routers:

- IP address (32 bit) - used for addressing datagrams
- "name", e.g., cs.virginia.edu - used by humans

Q: How to map between IP address and name, and vice versa?
Domain Name System:

- distributed database implemented in hierarchy of many name servers
- application-layer protocol: hosts, name servers communicate to resolve names (address/name translation)
  - note: core internet function, implemented as application layer-protocol
  - complexity at network's "edge"

DNS: Services, Structure

DNS Services

- hostname to IP address translation
- host aliasing
  - canonical, alias names
- mail server aliasing
- load distribution
  - replicated Web servers: many IP addresses correspond to one name

Q: Why not centralize DNS?

- single point of failure
- traffic volume
- distant centralized database
- maintenance