CMPE 150: Introduction to Computer Networks

Dr. Chane L. Fullmer
chane@cse.ucsc.edu
Homework Assignments

Homework assignment #4
Chapter three
Due May 29 – TODAY!!
Homework Assignments

Homework assignment #5

Chapter two

Due June 5th – only one short week from today
(Optional) Class Project

- Network programming project
  - In lieu of taking final examination
  - Or, just a wildcard – for mid-term or final

- Goal:
  - Build an FTP client/server from scratch
  - Using ‘C’ language

- Details on web page.. now...

Due by next Wednesday (June 4th)
Class Final Exam

- Final exam
  - Three hours
  - 40 – 50 questions -- comprehensive
  - Multiple choice as the midterm
  - Scantron – bring your sheet and pencils

- **Wednesday – June 11th** 8:00 – 11:00am
  - Thirteen days.. Tick, tick, tick...
CMPE 150: Introduction to Computer Networks
Set 16:

Application-level Protocols
Applications

- Name directory services (DNS)
- Remote terminal access (Telnet)
- File transfer (FTP)
- Web access (http)
- Electronic mail (SMTP)
- Internet telephony and streaming
Applications and application-layer protocols

Application: communicating, distributed processes
- e.g., e-mail, Web, P2P file sharing, instant messaging
- running in end systems (hosts)
- exchange messages to implement application

Application-layer protocols
- one “piece” of an app
- define messages exchanged by apps and actions taken
- use communication services provided by lower layer protocols (TCP, UDP)
Processes communicating across network

- process sends/receives messages to/from its socket
- socket analogous to door
  - sending process shoves message out door
  - sending process assumes transport infrastructure on other side of door which brings message to socket at receiving process
DNS

- Hierarchical name space.
- Distributed database.
- RFCs 1034 and 1035.
Domain Name System (DNS)

- Basic function: translation of names (ASCII strings) to network (IP) addresses and vice-versa.
- Example:
  - zephyr.isi.edu <-> 128.9.160.160
- Try the nslookup program (even in Windoze)
  - % nslookup zephyr.isi.edu (or any other name you wish to resolve)
DNS: Domain Name System

People: many identifiers:
- SSN, name, passport #

Internet hosts, routers:
- IP address (32 bit) - used for addressing datagrams
- "name", e.g., gaia.cs.umass.edu - used by humans

Q: map between IP addresses and name?

Domain Name System:
- distributed database implemented in hierarchy of many name servers
- application-layer protocol host, routers, name servers to communicate to resolve names (address/name translation)
  - note: core Internet function, implemented as application-layer protocol
  - complexity at network’s "edge"
History

- Original approach (ARPANET, 1970’s):
  - File *hosts.txt* listed all hosts and their IP addresses.
  - Every night every host fetches file from central repository.
  - OK for a few hundred hosts.
  - Scalability?
    - File size.
    - Centrally managed.
How is it used?

- Client-server model.
  - Client DNS (running on client hosts), or resolver.
  - Application calls resolver with name.
  - Resolver contacts local DNS server (using UDP) passing the name.
  - Server returns corresponding IP address.
DNS Name Space

A Tree-based Hierarchy:
Name Space Structure

- Top-level domains:
  - Generic.
  - Countries.

- In practice, all US organizations are under a generic domain, while everything outside the US is under the corresponding country domain.
DNS Names

- Domain names:
  - Concatenation of all domain names starting from its own all the way to the root separated by a dot “.” (Reverse order of IP addresses)
  - Refers to a tree node and all names under it.
  - Case insensitive.
  - Components up to 63 characters.
  - Full name less than 255 characters.
Name Space Management

- Domains are autonomous.
  - Organizational boundaries.
  - Each domain manages its own name space independently of other domains.

- Delegation:
  - When creating new domain: register with parent domain.
    - For name uniqueness.
    - For name resolution.
Resource Records

- Entry in the DNS database.
- Several types of entries or RRs.
- Example: RR “A” contains IP address.
- RR format: five-tuple.
  - Name.
  - TTL (in seconds).
  - Class (usually “IN” for Internet info).
  - Type: type of RR.
  - Value.
RR Types

- SOA: start of authority.
  - Marks beginning of zone’s database.
  - Provides general info about the zone: e-mail address of admin, default TTL, etc.

- A: address.
  - Contains 32-bit IP address.
  - Single name <-> several A RRs.

- MX: mail exchange.
  - Name of mail server for this domain.
RR Types

- **NS:** name server.
  - Name of name server for this domain.

- **CNAME:** canonical name.
  - Alias.

- **HINFO:** host description.
  - Provides information about host, e.g., CPU type, OS, etc.

- **TXT:** arbitrary string of characters.
  - Generic description of the domain, where it is located, etc.
Name Servers

- Entire database in a single name server.
  - Practical?
  - Why?

- DNS database is partitioned into zones.

- Each zone contains part of the DNS tree.

- Zone <-> name server.
  - Each zone may be served by more than one server.
  - A server may serve multiple zones.

- Primary and secondary name servers.
**DNS: Root name servers**

- Contacted by local name server that cannot resolve name
- Root name server:
  - contacts authoritative name server if name mapping not known
  - gets mapping
  - returns mapping to local name server

13 root name servers worldwide
Name Resolution

- Application wants to resolve name.
- Resolver sends query to local name server.
  - Resolver *configured* with list of local name servers.
  - Select servers in round-robin fashion.
- If name is local, local name server returns matching *authoritative* RRs.
  - *Authoritative* RR comes from authority managing the RR and is always correct.
  - *Cached* RRs may be out of date.
Name Resolution

- If information not available locally (not even cached), local NS will have to ask someone else.
  - It asks the server of the top-level domain of the name requested.
Recursive Resolution

- Recursive query:
  - Each server that does not have information forwards it to someone else.
  - Response finds its way back.

- Alternative is Iterative query:
  - Name server not able to resolve query, sends back the name of the next server to try.
  - Some servers use this method.
  - More control for clients.
Simple DNS example

host surf.eurecom.fr wants IP address of gaia.cs.umass.edu
1. contacts its local DNS server, dns.eurecom.fr
2. dns.eurecom.fr contacts root name server, if necessary
3. root name server contacts authoritative name server, dns.umass.edu, if necessary
DNS example

Root name server:
- may not know authoritative name server
- may know \textit{intermediate name server}: who to contact to find authoritative name server

Requests host: \texttt{surf.eurecom.fr}

Local name server: \texttt{dns.eurecom.fr}

Intermediate name server: \texttt{dns.umass.edu}

Authoritative name server: \texttt{dns.cs.umass.edu}

Gaia.cs.umass.edu
**DNS: Iterated Queries**

**recursive query:**
- puts burden of name resolution on contacted name server
- heavy load?

**iterated query:**
- contacted server replies with name of server to contact
- “I don’t know this name, but ask this server”
DNS: Caching and Updating Records

- Once (any) name server learns mapping, it caches mapping
  - cache entries timeout (disappear) after some time
- Update/notify mechanisms under design by IETF
  - RFC 2136
Electronic Mail
The First “Killer App”

- Non-interactive.
  - Deferred mail (e.g., destination temporarily unavailable).

- Spooling:
  - Message delivery as background activity.
  - Mail spool: temporary storage area for outgoing mail.
Mail system

User sends mail to User interface to Outgoing mail spool, which is connected to Client (send) via TCP connection (outgoing). Client (send) then sends mail to Server (receive) via TCP connection (incoming). Server (receive) delivers mail to Mailboxes incoming mail, which is then read by the User Interface to return mail to the User.
Observations

- When user sends mail, message stored in system spool area.
- Client transfer runs on background.
- Initiates transfer to remote machine.
- If transfer succeeds, local copy of message removed; otherwise, tries again later (30 min) for a maximum interval (3 days).
Mail alias expansion

- Mapping of e-mail identifiers to mail addresses.
  - Mail interface consults local alias database and performs mapping before passing message to outgoing mail spool.
  - One-to-many (e.g., mailing lists) and many-to-one (e.g., multiple ways to refer to a single user) mapping.
  - Incoming mail also goes through alias expansion before delivery.
SMTP

- Simple Mail Transfer Protocol
- How messages are transferred over a TCP/IP internet.
- Defines commands used to exchange mail between mail clients and servers.
- Problems reported to user by e-mail.
Example SMTP exchange

User smith@alpha.edu sends message to jones, green, and brown@beta.gov.

S: 220 beta.gov ready
C: Helo alpha.edu
S: 250 beta.gov
C: MAIL FROM smith@alpha.edu
S: 250 OK
C: RCPT TO: green@beta.gov
S: 250 OK
C: RCPT TO: brown@beta.gov
S: 550 no such user
C: DATA …
Mail retrieval

- SMTP implies server is always listening.
  - Port 25 (try it: telnet <server_name> 25)
- What about machines with intermittent Internet access?
  - 2-stage delivery: message delivered to user permanent mailbox; then user connects to retrieve messages.
  - User needs protocol to retrieve messages from “permanent” mailboxes.
POP3 & IMAP

**POP3** (Post Office Protocol v3) & **IMAP** (Internet Message Access Protocol)

- User invokes POP3/IMAP client; connects to server through TCP.
- Requires authentication (user id and passwd).
- Commands to retrieve and delete messages from permanent mailbox.
- POP3 downloads all messages, IMAP more sophisticated – read headers before selective download of mail.
- Mail server needs to run SMTP and POP3/IMAP.
More details...

- RFC 821 and 822 specify SMTP and its message formats.
- RFC 1939 specifies POP3
- RFC 3501 specifies IMAPv4