Announcements

- Project 2 due on 06.05 (by midnight).
  - Look at the instructions on Project 2’s Web page.
- Final exam on 06.09 from 7:30-10:30pm.
  - Comprehensive.
  - Location: BE 152 and BE 165 (overflow room).
- Course evaluation on 06.05.
  - Still need volunteer(s).
- Discussion sessions for final.
  - Chandra’s on Friday 06.06 at regular time.
  - Nacho’s: Friday pm or Monday?

Today

- DNS.
DNS

- IP addresses are not easy to remember.
- The Domain Name System (DNS) maps IP addresses to host names.
- Host name is formed by machine name followed by domain name.
  - $\text{Host\_name.\ domain\_name}$

DNS (cont’d)

- The domain\_name is formed by the institutional site name and the Top-Level Domain name (TLD).
  - So the host name is of the form $\text{machine\_name.1st\_site\_name.TLD\_name}$
- Examples:
  - `sundance.ucsc.edu`
  - `soe.ucsc.edu` (alias for `sundance.ucsc.edu`)
  - `italia.cse.ucsc.edu`
  - `helios.jpl.nasa.gov`
  - `www.cnn.com`

TLD

- TLD names identify organization types or country codes.
- Examples:
  
<table>
<thead>
<tr>
<th>TLD</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.com</td>
<td>Commercial org.</td>
</tr>
<tr>
<td>.edu</td>
<td>Educational site in US</td>
</tr>
<tr>
<td>.gov</td>
<td>Government site in US</td>
</tr>
<tr>
<td>.mil</td>
<td>Military organization in US</td>
</tr>
<tr>
<td>.net</td>
<td>Network site</td>
</tr>
<tr>
<td>.org</td>
<td>Nonprofit organization</td>
</tr>
<tr>
<td>.ac</td>
<td>United Kingdom</td>
</tr>
<tr>
<td>.au</td>
<td>Australia</td>
</tr>
<tr>
<td>.ca</td>
<td>Canada</td>
</tr>
<tr>
<td>.fr</td>
<td>France</td>
</tr>
<tr>
<td>.de</td>
<td>Germany</td>
</tr>
<tr>
<td>.uk</td>
<td>United Kingdom</td>
</tr>
<tr>
<td>.it</td>
<td>Italy</td>
</tr>
<tr>
<td>.es</td>
<td>Spain</td>
</tr>
</tbody>
</table>

Countries define their own internal hierarchy (e.g., .ac.uk, .edu.au)

DNS (cont’d)

- Organizations can create any internal DNS hierarchy.
- Authority for creating new subdomains within a domain name is delegated to each domain.
  - Administration of ucsc.edu has authority to create `cse.ucsc.edu` and need not contact any central naming authority.
Example of DNS Hierarchy

Announcements
- Final exam: Monday 06.09 7:30-1030pm
  - Cumulative.
  - Location: BE152 and BE 165.
- Project 2: due today by midnight.
  - Email to Nacho and Chandra.
- Discussion sessions for final:
  - Chandra: Tomorrow, 06.06, from 10-12.
  - Nacho: Monday, 06.09, from 2-4.
  - Kata: office hours tomorrow, Friday from 12:30-2pm.
- Course evaluation today!

Today
- DNS (cont’d).
- Application layer.
  - FTP,
  - E-mail,
  - WWW.

DNS Name Space
- DNS names are managed by a hierarchy of DNS servers.
  - Hierarchy is related to DNS domain hierarchy
- Root server at top of tree knows about next level servers.
- Next level servers, in turn, know about lower level servers.
Example of DNS Hierarchy

Example of DSN Hierarchy (cont’d)

Choosing DNS Server Architecture

- Small organizations can use a single server.
  - Easy to administer.
  - Inexpensive.
- Large organizations often use multiple servers.
  - Reliability through redundancy.
  - Improved response time through load sharing.

Name Resolution

- “Resolving a name” means mapping the host name to the IP address.
- Reverse mapping is also possible.
- A client computer calls a DNS server for name resolution
  - DNS request contains name to be resolved.
  - DNS reply contains IP address for name in request.
Using DNS Servers

- Each DNS server is the authoritative server for the names it manages.
  - If request contains name managed by receiving server, that server replies directly.
  - Otherwise, request is forwarded to the appropriate authoritative server.
- DNS request is originally sent to root server, which points at next server to use
  - Eventually, the authoritative server for the DNS name in the request is located and IP address is returned.

Internet Applications

Client-Server Architecture

Client

request

response

Server

Clients

Server

Clients

Server
**Client-Server Model**

**Remote File Access**

- In the old times: file transfer by removable medium.
- Network allows "seamless" remote access:
  - File transfer: equivalent of tape/floppy/CD transfer.
  - Remote file system: access to files on networked computers as though files are local.

**File Transfer Protocol (FTP)**

- File transfer is one of the Internet’s "original" applications.
- FTP is a general purpose protocol
  - Transfer arbitrary files.
  - Accommodates file ownership and access restrictions.
- Usually requires a password. Some FTP servers provide anonymous ftp (does not require password).

**FTP Client-Server Model**

- Remote server accepts control connection from local client
  - Client sends commands to server.
  - Persists through entire session.
- Server creates data connection for data transfer
  - One data connection for each transferred file.
FTP Operation

Electronic Mail

- Another "original" Internet application.
- Enables new forms of interaction.
- E-mail also uses client-server architecture.
  - E-mail client accepts mail from user and sends it to server on destination computer.

Electronic Mailboxes

- Users have an electronic mailbox into which incoming mail is deposited.
- User then accesses mail with a mail reader program.
- Usually associated with a computer account.

E-mail Addresses

- Electronic mailbox is identified by an e-mail address
  - Typically user's ID account.
- To deliver e-mail, need computer name together with mailbox
  - Typically: user@host
  - Sender uses the host part to select a destination; receiving server uses user part to select a mailbox.
Mail Client-Server

- Source mail client:
  - Resolves destination name using DNS.
  - Contacts mail delivery server at destination.
  - Copies mail to server.
- Destination mail server:
  - Interprets user name as mailbox id.
  - Places mail in appropriate mailbox.
  - What's a mailbox?

E-mail Message Format

- Simple two-part format:
  - Header includes delivery information
  - Body carries text of message
- Header:
  - Lines of text in format
    keyword:information
  - Some keyword are required; some optional.

Header Fields

- To: Addr. of primary recipient
- Cc: Addr. of secondary recipients
- Bcc: Addr. for blind carbon copies
- From: Person who "created" the message
- Sender: Person who actually "sent" the message
- Receiver: Line added by each transfer agent along the route
- Date: Date and time message was sent
- Reply-To: Email address to which replies should be sent
- Message-Id: Unique number for referencing this message
- In-Reply-To: Message-Id of message to which this is a reply
- Subject: Short summary of message for 1-line display

Header - Example

Date: Wed, 24 Mar 95 21:46:26 PST
From: Ram Nevatsa <nevatsa@hsc.usc.edu>
To: kate@kula
Subject: talk
Message-Id: <CMN.6.95.2.756545286 nevatsa@hsc.usc.edu>
Data in E-mail

- Original Internet system: RFC 822.
- Perfectly suited for transmitting text.
- Does not work with:
  - Messages in languages with accents (e.g. French).
  - Messages in non-Latin alphabets (e.g. Russian) or languages without alphabets (e.g. Chinese).
  - Messages not containing text at all (e.g., audio and video).
- Solution: Multipurpose Internet Mail Extension (MIME)

MIME

- MIME extends and automates encoding mechanisms.
- Allows inclusion of separate components (programs, pictures, audio/video clips) in a single mail message.
- Sending program identifies the components so receiving program can automatically extract and inform mail recipient.

MIME (cont’d)

- MIME uses RFC 822 but adds structure to message body and defines encoding rules for non-ASCII messages.
  - Uses same delivery mechanism.
- MIME adds a Content-Type field, where a sender can specify the nature of the message body.
- E.g., Content-Type = video/mpeg
  - Then receiver knows that the message body contains a video encoded with the MPEG standard.

E-mail Transfer

- E-mail communication is really a two-part process:
  - User composes mail with an e-mail interface program.
  - Mail transfer program delivers mail to destination.
    - Waits for mail to be placed in outgoing message queues.
    - Picks up message and determines recipient(s).
    - Contacts server on recipient’s computer.
    - Passes message to server for delivery.
**SMTP**

- **Simple Mail Transfer Protocol (SMTP)** is the standard application protocol for delivery of email from source to destination.
- Provides **reliable delivery** of messages.
- Uses TCP.
- Other functions:
  - E-mail address lookup.
  - E-mail address verification.
- Allows for multiple recipients.

**Mail Gateways**

- Mail processing may take significant resources in large organizations.
- May be segregated to a dedicated server computer: **mail gateway.**
  - Provides single mail destination point for all incoming mail (e.g.: `ucsc.edu`).
**Mailbox Access**

- Where should mailbox be located?
- Users want to access mail from most commonly used computer.
- Solution: ssh to remote computer with mail server.
- Or, Post Office Protocol (POP).

**POP**

- Usually, mailboxes are kept on a “dedicated” machine.
- Computer with mailboxes runs POP server.
- User runs POP client on local computer.
- POP client can access and retrieve messages from mailbox.
- Requires authentication (password).
- Local computer uses SMTP for outgoing mail.

**POP and Dialup Access**

- POP useful for dialup connection.
  - User’s computer not always connected.
  - Can download all mail at once and read off-line.

- An alternative to POP: IMAP (Interactive Mail Access Protocol):
  - Does not copy email to user’s PC - email always resides in the mail server.
  - Useful when user uses several computers.
The Web

- WWW, or the world-wide web is a resource discovery service.
  - Resource space is organized hierarchically, and resources are linked to one another according to some relation.
  - Hypertext organization: link “granularity”; allows links within documents.
  - Graphical user interface.

Hypertext/Hypermedia

- Hypermedia system allows interactive access to collections of documents

- Document can hold:
  - Text (hypertext)
  - Graphics
  - Sound
  - Animations
  - Video

- Documents linked together
  - Non-distributed (all stored locally - like CD-ROM)
  - Distributed (stored on remote servers)

Some History 1

- Started in 1989 at CERN, European center for nuclear research, in Switzerland.
- Original motivation: need for scientists around the world to collaborate and share multimedia information.
- Tim Berners-Lee came up with initial proposal of a web of linked documents

Some History 2

- First text-based prototype demo in 12.91.
- Release of first graphical interface, Mosaic, in 02.93 at NCSA by M. Andreessen.
- In 1994, Andreessen creates Netscape.
- In 1994, CERM and MIT set up the WWW Consortium to further develop the Web.
  - www.w3.org for more information.
**The Client Side**

- Users perceive the Web as a vast collection of information.
  - Page is the Web’s information transfer unit.
  - Each page may contain links to other pages.
  - Users follow links by clicking on them which takes them to the corresponding page.
  - This process can go on indefinitely, traversing several pages located in different places.

**The Browser**

- Program running on client that retrieves and displays pages.
  - Interacts with server of page.
  - Interprets commands and displays page.
- Examples: Mosaic, Netscape’s Navigator and Communicator, Microsoft Internet Explorer.
- Other features: back, forward, bookmark, caching, handle multimedia objects.

**The Server Side**

- Web site has Web server running that answers requests for pages locally served.
  - Web server listens to port 80 for requests.
  - When request from client arrives, connection is set up.
  - Server replies.
  - Connection released.

**Example**

- User clicked on www.w3.org/hypertext/WWW/TheProject.html.
  - Browser asks DNS to resolve www.w3.org.
  - DNS replies with 18.23.0.23.
  - Browser sets up connection to 18.23.0.23 port 80.
  - Browser sends GET /hypertext/WWW/TheProject.html.
  - www.w3.org server sends TheProject.html file.
  - Connection released.
  - Browser displays TheProject.html, fetching and displaying all embedded objects (images, etc).
Observations

- Many browsers display status information at bottom of the screen.
- For each embedded object (in-line image like icon, picture, etc), browser establishes new connection.
  - Performance hit.
  - Revisions to protocol (HTTP) address this.
- Since HTTP is ASCII, easy for user to talk to Web servers directly (e.g., telnet to port 80).

More Observations

- Server’s response specifies object type (using MIME) followed by object body.
- For example:
  - Content-Type: Image/GIF
  - Content-Type: Text/html

HTTP

- HyperText Transfer Protocol.
- Each interaction: client’s ASCII request followed by MIME-like response.
- Use TCP as underlying transport protocol (although not required by standard).
- Several co-existing versions of HTTP.

HTTP Operations

- Commands (method) to be executed on object (Web page).

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>Request to read Web page</td>
</tr>
<tr>
<td>HEAD</td>
<td>Request to read Web page’s header</td>
</tr>
<tr>
<td>PUT</td>
<td>Request to store Web page</td>
</tr>
<tr>
<td>POST</td>
<td>Append to specified object (e.g., Web page)</td>
</tr>
<tr>
<td>DELETE</td>
<td>Deletes Web page</td>
</tr>
<tr>
<td>LINK</td>
<td>Connects 2 objects</td>
</tr>
<tr>
<td>UNLINK</td>
<td>Disconnects 2 objects</td>
</tr>
</tbody>
</table>
**HTTP Responses**

- Every request gets response with status information.
  - Status code 200: OK.
  - Status code 400: bad request.
  - Status code 304: not modified.

**Persistent connections**

- Up to HTTP 1.0, separate connection for each data transfer.
- HTTP 1.1: persistent connections.
  - Same connection for multiple transfers.
  - Less overhead (connection management, slow start), less machine resources (buffers, connection id’s).
  - But, need to recognize beginning and end of an item (use length information).

**Proxy servers**

- What are proxy servers?
  - Close to client.
  - Close to server.

**Client-side proxies**

Diagram showing the flow of data between clients, proxies, and servers.
**Server-side proxies**

**Caching**

- What’s caching?
- What are its benefits?
- What are its problems?

**HTTP support for caching**

- Servers can control caching.
  - Cachable/non-cacheable object.
  - Cacheable at proxy.
  - Cached object expiration time.
  - Operations performed on cached copy.
- Browsers can force request to go to server.
  - Specifies maximum age not to be > 0.

**Internet caches**

- Peers.
- Hierarchical caches.
Uniform Resource Locator (URL)

- Way to identify objects (pages).
  - What is page called?
  - Where is it located?
  - How to access page?
- URL has 3 parts:
  - Protocol (or scheme).
  - Machine’s name/address.
  - Local name (file name).

URL

- Ability to handle other protocols.
  - HTTP, FTP, news, gopher, mail, telnet.
- Universal Resource Identifier (URI).
  - Location transparency.
  - Replication.

HyperText Markup Language (HTML)

- Allows users to produce Web pages including text, graphics, pointers, etc.
- Markup language: describe how objects are to be formatted.
  - Contains explicit commands for formatting.
  - Example: `<B>` and `</B>`.
  - Advantages: easy to parse.

Types of Documents

- Documents on the web can be of three types:
  - Static (those we have seen so far)
    - Defined in text file by page author
    - Remains unchanged unless edited by author
  - Dynamic
    - Generated on demand by HTTP server
  - Active
    - Execute code on the WWW browser in the host computer
**Dynamic Pages**

- A dynamic document is generated by the server at each new connection.
  - That’s why sometimes, when downloading from the same URL, we obtain different pages.
- **Common Gateway Interface (CGI) standard** defines server-application interaction.
  - CGI programs can be as simple as adding the time or date to the page.
- Browser may supply parameters to CGI program.
  - Browser extends URL with additional parameters separated by ?

**“Personalizing” Web Content**

- If the server has personal information about the user, CGI can be used to “personalize” the page content.
  - Based on a current set of preferences.
    - Stock quotes.
    - Advertising based on customer personal info, or past preferences.

**Forms**

- Forms permit a web page to have blank areas in which the user must enter information.
  - Makes it possible to enter data directly.
  - Name, address, credit card info...
  - Allows information to be sent to the server directly.

**What is a Cookie?**

- A server invokes a CGI program each time a request arrives for the associated URL.
  - The server does not maintain any history of requests.
  - But a history is useful to allow CGI program to participate in dialog (e.g., to avoid having a user answer questions repeatedly).
  - Information saved between invocations is called state information.
  - State information is kept at the client’s side!
What is a Cookie (cont’d)
- State information is passed by browser in the form of a cookie.
  - A cookie is just a piece of state information that can be a few hundreds bytes long.
  - Cookies mostly contain values assigned by the server; additional information is stored on the server itself.
  - The cookie is kept in the client’s computer.
  - When it contacts the Web site again, the browser inserts the cookie in the request.
  - From the server’s perspective, it appears that the browser can store and return state information!

Plugins
- A plugin is a small program that knows how to interpret one specific data format.
  - Extends browser
  - E.g.: Adobe Acrobat plugin (to read pdf files), Quicktime plugin, Real plugin...