Introduction to LAN/WAN

Application Layer (Part II)
Application Layer Topics

- Domain Name System (DNS) (7.1)
- Electronic Mail (Email) (7.2) ←
- World Wide Web (WWW) (7.3)
Electronic Mail (Email)

- Mostly used in academia before 1990
- 1990s:
  - Became widely used by public
  - Grew exponentially, now exceeds snail mail
- 1997 quote:
  - Amount of information on Internet per day in terabytes
    (more than library of Congress)
- Email relatively informal with conventions
  - BTW (By The Way), ROFL, IMHO, etc
Email: Smileys

- Smileys popular (Sanderson and Dougherty, 1993)
- Usually read by rotating 90 degrees clockwise

<table>
<thead>
<tr>
<th>Smiley</th>
<th>Meaning</th>
<th>Smiley</th>
<th>Meaning</th>
<th>Smiley</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>:-)</td>
<td>I’m happy</td>
<td>=</td>
<td>l</td>
<td>:-)</td>
<td>Abe Lincoln</td>
</tr>
<tr>
<td>:-(</td>
<td>I’m sad/angry</td>
<td>=)::-)</td>
<td>Uncle Sam</td>
<td>:-))</td>
<td>Double chin</td>
</tr>
<tr>
<td>:-l</td>
<td>I’m apathetic</td>
<td>*&lt;:-)</td>
<td>Santa Claus</td>
<td>:-{}</td>
<td>Mustache</td>
</tr>
<tr>
<td>;:-)</td>
<td>I’m winking</td>
<td>&lt;:-(</td>
<td>Dunce</td>
<td>#::-)</td>
<td>Matted hair</td>
</tr>
<tr>
<td>:-)(O)</td>
<td>I’m yelling</td>
<td>(-:-)</td>
<td>Australian</td>
<td>8-)</td>
<td>Wears glasses</td>
</tr>
<tr>
<td>:-(*</td>
<td>I’m vomiting</td>
<td>:-)X</td>
<td>Man with bowtie</td>
<td>C:-)</td>
<td>Large brain</td>
</tr>
</tbody>
</table>
Electronic Mail (Email)

Email systems initially simple
- Used FTP with recipient address as first line

Major limitations
- Inconvenient to send to group
- Messages: no structure, processing difficult. E.g. extracting forwarded portion
- Sender not sure of delivery
- Secretary couldn’t handle absent boss’ mail
- Poor interface: edit first, exit editor, send
- Impossible to send mixed media mail (text, images, voice, video, fax, etc)
Email Standards

- 1982: ARPANET proposals published
  - RFC 821: transmission protocol
  - RFC 822: message format
- Later: RFC 2821 and 2822 became standards
- Most people still refer to RFC 821 and 822
- Note: Internet standardization process
  - RFC are Request For Proposals (www.ietf.org/rfc)
  - Tech report, describes a protocol or idea (email, TCP, IP, etc)
  - Internet culture informal, everyone critiques proposal
  - If RFC has merit, working implementations for 4 months, becomes standard
Email Standards

- Email system parts:
  - *User agent*: (mail reader) is program used for composing, editing and manipulating email. E.g. *pine* (RFC 822 or 2822)
  - *Message transfer agent*: concerned with relaying email message from originator to recipient (RFC 821 or 2821)

- Note RFC 822 deals with ASCII text
- MIME deals with multimedia extensions
RFC 822

- To: field gives DNS address of primary recipient
- RFC 822 header fields related to message transport:

<table>
<thead>
<tr>
<th>Header</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>To:</td>
<td>E-mail address(es) of primary recipient(s)</td>
</tr>
<tr>
<td>Cc:</td>
<td>E-mail address(es) of secondary recipient(s)</td>
</tr>
<tr>
<td>Bcc:</td>
<td>E-mail address(es) for blind carbon copies</td>
</tr>
<tr>
<td>From:</td>
<td>Person or people who created the message</td>
</tr>
<tr>
<td>Sender:</td>
<td>E-mail address of the actual sender</td>
</tr>
<tr>
<td>Received:</td>
<td>Line added by each transfer agent along the route</td>
</tr>
<tr>
<td>Return-Path:</td>
<td>Can be used to identify a path back to the sender</td>
</tr>
</tbody>
</table>
Multipurpose Internet Mail Extensions (MIME)

- RFC 822 had problems with international languages with:
  - accents (French, German), non-Latin alphabets (Hebrew, Russian) and without alphabets (Chinese, Japanese)
- Also with messages containing NO text (audio, images)
- Basic idea: continue RFC 822 format, but add structure to message body and define encoding rules for non-ASCII
- Five new (RFC 822) headers added:

<table>
<thead>
<tr>
<th>Header</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIME-Version:</td>
<td>Identifies the MIME version</td>
</tr>
<tr>
<td>Content-Description:</td>
<td>Human-readable string telling what is in the message</td>
</tr>
<tr>
<td>Content-Id:</td>
<td>Unique identifier</td>
</tr>
<tr>
<td>Content-Transfer-Encoding:</td>
<td>How the body is wrapped for transmission</td>
</tr>
<tr>
<td>Content-Type:</td>
<td>Type and format of the content</td>
</tr>
</tbody>
</table>
MIME

Most interesting MIME header is *Content-Type*

7 types in RFC 2045. E.g: *Content-Type: video/mpeg*

MIME types and subtypes defined in RFC 2045

Properly designed user agents must interpret these types

<table>
<thead>
<tr>
<th>Type</th>
<th>Subtype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text</td>
<td>Plain</td>
<td>Unformatted text</td>
</tr>
<tr>
<td></td>
<td>Enriched</td>
<td>Text including simple formatting commands</td>
</tr>
<tr>
<td>Image</td>
<td>Gif</td>
<td>Still picture in GIF format</td>
</tr>
<tr>
<td></td>
<td>Jpeg</td>
<td>Still picture in JPEG format</td>
</tr>
<tr>
<td>Audio</td>
<td>Basic</td>
<td>Audible sound</td>
</tr>
<tr>
<td>Video</td>
<td>Mpeg</td>
<td>Movie in MPEG format</td>
</tr>
<tr>
<td>Application</td>
<td>Octet-stream</td>
<td>An uninterpreted byte sequence</td>
</tr>
<tr>
<td></td>
<td>Postscript</td>
<td>A printable document in PostScript</td>
</tr>
<tr>
<td>Message</td>
<td>Rfc822</td>
<td>A MIME RFC 822 message</td>
</tr>
<tr>
<td></td>
<td>Partial</td>
<td>Message has been split for transmission</td>
</tr>
<tr>
<td></td>
<td>External-body</td>
<td>Message itself must be fetched over the net</td>
</tr>
<tr>
<td>Multipart</td>
<td>Mixed</td>
<td>Independent parts in the specified order</td>
</tr>
<tr>
<td></td>
<td>Alternative</td>
<td>Same message in different formats</td>
</tr>
<tr>
<td></td>
<td>Parallel</td>
<td>Parts must be viewed simultaneously</td>
</tr>
<tr>
<td></td>
<td>Digest</td>
<td>Each part is a complete RFC 822 message</td>
</tr>
</tbody>
</table>
SMTP

Internet email delivery:
- Source machine establishes TCP connection to port 25 of destination machine
- E-mail daemon (process) listening at this port speaks SMTP (Simple Mail Transfer Protocol), RFC 821
- Daemon accepts incoming connections, copies messages to appropriate mailboxes
- Error generated for undeliverable message

SMTP is simple ASCII protocol
Message Transfer

Example: Transferring a message from elinore@abc.com to carolyn@xyz.com.

Try telnet garden 25
Message Transfer with Temporary Connections

- **Case a**: Sending and reading mail when the receiver has a permanent Internet connection and the user agent runs on the same machine as the message transfer agent.

- **Case b**: Reading e-mail when the receiver has a dial-up connection to an ISP.
Disconnected Users

- **Post Office Protocol (Pop3) (RFC 1939)**
  - ISP accepts mail for disconnected subscribers (USPS??)
  - Disconnected user calls up ISP, transfers using port 110
  - Pop3 protocol goes through 3 states: *authorization*, *transaction* and *update*

- **Internet Message Access Protocol (IMAP) (RFC 2060)**
  - One user can have multiple IMAP accounts (WPI, AOL, etc)
  - Pop3 assumes use will download all mail once connected
  - IMAP allows manipulation of message parts on server

- **Webmail (hotmail, yahoo, etc)**
  - Uses SMTP on port 25
  - Web interface, similar process as before
World Wide Web

- Architectural framework for accessing linked documents
- Linked pages spread all over the Internet, on many machines
- Innovator: Tim Berners-Lee, *CERN* (European Nuclear Physics Center), March 1989
- In 10 years, went from means to distributing high-energy physics to pervasive application
- Most people (meatheads??) think of the web as the Internet
- Milestones:
  - First public demo’ of WWW at *Hypertext ’91* conference
  - Mosaic, first web browser, free, Marc Andreessen, UIUC, 1993
  - Netscape, first commercial browser, 1995, IPO, mania!!!
  - WWW consortium formed 1994 (MIT, CERN), www.w3.org
Web Architecture

- User perceives web of worldwide collection of documents or web pages
- Original hypertext vision: Vanavar Bush, MIT prof, US presidential advisor, 1945
- Web pages viewed using browsers (Netscape navigator, Internet Explorer)
- Browser fetches pages, interpretes and displays them
- User can access linked pages by clicking on them
- Linked pages or pieces (images, video, text, etc) can be on same host or in Australia
Web Architecture

Parts of the web model:
Client Side

Client side (browser) steps:

- User enters or click on URL (e.g. www.itu.org)
- Browser asks DNS server for IP address of www.itu.org
- DNS replies with IP address (e.g. 156.106.192.32)
- Browser makes TCP connection (port 80) with 156.106.192.32
- Asks for file (web page) on server’s directory /home/index.html
- www.itu.org server sends file /home/index.html
- TCP connection is released
- Browser displays all text in web page
- Browser fetches and displays all images in this file
Client Side

- Web pages written in HTML,
- HTML: interpreted markup language derived from SGML
- Browsers support HTML as well as MIME types
- Ever increasing number of file formats
- Instead of building larger browsers, use:
  - Plug-ins: software extension (module) that browser loads to run new file. Usually runs MIME types
  - Helper applications: complete program, separate process. E.g. application/pdf specified in browser option will load Adobe Acrobat reader. Can be non-MIME type
Server Side

- Web server steps:
  - Waits for connections coming in on port 80
  - Accepts a TCP connection from client (browser)
  - Get name of requested file
  - Fetch requested file from disk
  - Return file to client (similar to FTP)
  - Release TCP connection

- Above is basic web server, modern ones do more

- Server could be inundated with requests (caching)
Web Architecture

Universal Resource Locator (URL)
- A web page’s unique name (worldwide)
- Three parts (protocol, DNS name of host, local file name)
  `protocol` (http), `www.cs.vu.nl` (DNS name) and `video/index-en.html` (file name)
- Other web protocols: `ftp:`, `file:`, `news`, `gopher`, `mailto:`, `telnet:

Statelessness and cookies
- Stateless server, doesn’t remember previous requests or clients
- Sometimes want to track. E.g e-Commerce shopping cart
- Can’t use IP address, many people share computers, IP addresses
- Netscape devised **cookies** (No!! can’t eat ‘em), RFC 2109
- Cookies: server returns requested file, plants cookie (small info) on client
- Next time client logs in, sends cookies back
Two types: static and dynamic

**Static:**
- Simple, same exact page (file) every time
- Examples: HTML, forms, XML

**Dynamic:**
- Web page or parts of it are generated on demand
- Content generation can either take place on server or client
- Server side usually involves scripting (CGI, PHP, JSP) and maybe database access
- Client side is for interactive (and kewl) websites, with features like rollover buttons. E.g. Javascript
HTTP

- Web transfer protocol, *Hypertext Transfer Protocol (HTTP)*
- RFC 2616, specifies what messages a client sends and responses server returns
- TCP used so that servers don’t worry about lost messages
- Initially, one TCP connection per web object (part)
- Too much eye candy these days, try *persistent connections*
- HTTP methods:

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>Request to read a Web page</td>
</tr>
<tr>
<td>HEAD</td>
<td>Request to read a Web page’s header</td>
</tr>
<tr>
<td>PUT</td>
<td>Request to store a Web page</td>
</tr>
<tr>
<td>POST</td>
<td>Append to a named resource (e.g., a Web page)</td>
</tr>
<tr>
<td>DELETE</td>
<td>Remove the Web page</td>
</tr>
<tr>
<td>TRACE</td>
<td>Echo the incoming request</td>
</tr>
<tr>
<td>CONNECT</td>
<td>Reserved for future use</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>Query certain options</td>
</tr>
</tbody>
</table>