1.264 Lecture 11

Web introduction, HTTP
Internet and Web Context

• We have covered databases, which are the “nodes” in our information network where data is stored
• We next turn to the Internet and Web, which will be the links that connect the nodes, transferring data from one node to another
• The Web, based on the Internet, is a communications channel or protocol converter to transfer data from one database to another
  – Communications channel has many layers
  – Protocol conversion includes converting meaning (semantics) as well as technical conversions in how data are represented among the cooperating computers
• We’ll start with Internet and Web basics, including its historic roots as a way for people to interact
  – We’ll then transition to the Web as a way for systems to interact, which is the current focus
Web browsers and servers

- Internet is “just” a set of loosely interconnected networks
  - A set of local area networks connected via wide area networks
  - No centralized control or management
  - Network segments interconnect via routers:
    - Dedicated computers that manage packets of data
  - TCP/IP is the universal data protocol on the network
    - Data format, content is left to higher-level protocols, like the Web
- TCP/IP connections
  - Client is typically a data consumer that sends short requests
    - On Web, client is a browser or an application program
  - Server is typically a data provider that sends long responses
    - On Web, server is typically an application server or database server
    - Server listens for requests and transmits desired data
    - Can send static or dynamic data
  - Web connection is active only long enough to exchange data
    - Avoid overhead of many communication channels, but lose state
Transmission Control Protocol/Internet Protocol TCP/IP

Server computer

Layer 7: Application
Layer 4: Transport
Layer 3: Network
Layer 2: Data Link
Layer 1: Physical

Browser or client computer

Layer 7: Application
Layer 4: Transport
Layer 3: Network
Layer 2: Data Link
Layer 1: Physical

HTTP/SOAP
TCP
IP
Ethernet
10BaseT

Web pgs
Connections
Packets
Frames
Bits
TCP/IP layering

TCP header: source port, destination port, seq nbr, checksum, time to live, out of band signal

IP header: protocol (TCP), IP source addr, IP dest addr

Ethernet header: Eth source addr, Eth dest addr, protocol (IP)

<table>
<thead>
<tr>
<th>Layer</th>
<th>Protocol/Protocol Type</th>
<th>Source Addr</th>
<th>Destination Addr</th>
<th>Header Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10BaseT</td>
<td></td>
<td></td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>Ethernet</td>
<td></td>
<td></td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>IP</td>
<td></td>
<td></td>
<td>20</td>
</tr>
<tr>
<td>4</td>
<td>TCP</td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>HTTP, SOAP</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Data

Checksum
Web organization: Web pages

- Web page or document is unit of organization
  - Simple pages retrieved in single operation
  - Compound pages (often graphics) take multiple passes (multiple TCP/IP connections unless ‘Keep-Alive’ is set)
  - Pages have hypertext
    - Display text and images
    - Links to other pages, implemented via embedded URLs
  - Pages are described using Hypertext Markup Language (html)
    - High level document description language
    - Specifies layout but not exact appearance of document
    - Defines sections, such as “level 1 header”, “list”, “emphasized text”
    - Browser handles display of page on client machine
  - HTML may be replaced by XHTML in the future
    - Subset of XML, more structured, verifiable than HTML
Web organization: MIME types

• Every document on Web has a type
  – Multipurpose Internet Mail Extensions (MIME) types, in http standard
    • Some types, such as html, text and some graphics, are displayed directly by browser
    • Others need helper or plug-in external programs to display type
  – Web type system is extensible. New types are easy to accommodate.
    • Define new MIME type in server
    • Browser will prompt for helper application if MIME type unrecognized
    • Browser can suggest Web site to download helper app
    • Download and install helper app, and use the new MIME type
  – Web server extensibility: programs to generate documents
    • URLs can point at programs as well as pages
    • Programs are based on Java 2 Enterprise Edition (J2EE), Microsoft .NET frameworks, or LAMP (Linux-Apache-MySQL-php)
MIME type examples

- application/msword: Word
- application/pdf: Acrobat
- application/vnd.ms-excel: Excel
- application/zip: Zip file
- audio/basic: .au, .snd
- image/gif: GIF
- image/jpeg: JPEG
- text/plain: Plain text
- text/html: HTML
- text/xml: XML
- video/mpeg: Video

See http://www.rfc-editor.org for current list of HTTP, MIME, other Internet specifications
HTTP protocol

- HTTP 1.1 is the current protocol
- Four phases:
  - **Open connection**: Based on URL
  - **Request**: Client opens connection to server and sends:
    - Request method, (and request data at bottom if POST or PUT request)
    - URL,
    - HTTP version number
    - Header information (informational, optional), terminated with blank line
  - **Response**: Server processes request and sends:
    - HTTP protocol version and status code
    - Header information, terminated by blank line
    - Text (data)
  - **Close connection**
These transactions are **stateless**. The connection is closed after each page and re-established: Server can’t connect successive requests from the client.
HTTP request examples

Example browser request:
telnet web.mit.edu 80 HTTP/1.1 (telnet opens connection, http is port 80)
GET /newsoffice/ (request)
Host: web.mit.edu (header)

Example server response:
HTTP/1.1 200 OK (response)
Server: Apache/1.3.3 Ben-SSL/1.28 (Unix) (header)
Content-Type: text/html
Content-Length: 8300

<MIME (html) document>
<HTML>
<HEAD><TITLE>MIT news...</TITLE></HEAD>
<BODY>MIT news...</BODY>
</HTML>
Connection closed (close connection)
## HTTP request phase (browser to server)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>Requests specified document (used to ‘post’ too)</td>
</tr>
<tr>
<td>HEAD</td>
<td>Requests only header of specified document</td>
</tr>
<tr>
<td>POST</td>
<td>Requests that server accept data from browser and generate dynamic content</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>Get server and access options (OPTIONS *)</td>
</tr>
<tr>
<td>TRACE</td>
<td>Used in debugging</td>
</tr>
<tr>
<td>PUT</td>
<td>Replace server document with data from browser</td>
</tr>
<tr>
<td>DELETE</td>
<td>Delete specified document on server</td>
</tr>
</tbody>
</table>

- GET is the most frequent request from browsers
- POST used for processing fill-out forms
- HEAD is used by search engines to check for live pages
- OPTIONS used sometimes for machine-machine connections
- TRACE for debugging
- PUT, DELETE used for blogs, wikis; never for commerce
# Common HTTP request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>MIME types accepted by client (multiple allowed)</td>
</tr>
<tr>
<td>Connection</td>
<td>Connection type for client (keep-alive, close)</td>
</tr>
<tr>
<td>Cookie</td>
<td>Return previous cookie to server (session data)</td>
</tr>
<tr>
<td>From</td>
<td>Email address of user (sent only by crawlers)</td>
</tr>
<tr>
<td>Host</td>
<td>Original host requested (forward, multiple names)</td>
</tr>
<tr>
<td>If-Modified-Since</td>
<td>Used to reduce fetching of docs client has</td>
</tr>
<tr>
<td>Referer</td>
<td>URL from which this link was obtained</td>
</tr>
<tr>
<td>User-Agent</td>
<td>Name and version of client software (browser)</td>
</tr>
</tbody>
</table>
## HTTP response status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>100-101</td>
<td>Informational response; client should respond with some other action (continue, new protocol)</td>
</tr>
<tr>
<td>200-206</td>
<td>Request was successful</td>
</tr>
<tr>
<td>300-307</td>
<td>Document has moved; indicate new address</td>
</tr>
<tr>
<td>400-417</td>
<td>Client error, such as unauthorized request</td>
</tr>
<tr>
<td>500-505</td>
<td>Server error</td>
</tr>
</tbody>
</table>

### Examples:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
</tr>
<tr>
<td>404</td>
<td>Not found</td>
</tr>
<tr>
<td>505</td>
<td>HTTP version not supported</td>
</tr>
</tbody>
</table>
## HTTP response header examples

<table>
<thead>
<tr>
<th>Header</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow</td>
<td>Requests allowed, such as GET</td>
</tr>
<tr>
<td>Content-Length</td>
<td>Length in bytes of data to be returned</td>
</tr>
<tr>
<td>Content-Type</td>
<td>MIME type of returned data</td>
</tr>
<tr>
<td>Expires</td>
<td>Date at which document expires</td>
</tr>
<tr>
<td>Last-Modified</td>
<td>Date at which document was last modified</td>
</tr>
<tr>
<td>Location</td>
<td>New document address (with 300 status)</td>
</tr>
<tr>
<td>Set-Cookie</td>
<td>Gives browser a ‘cookie’</td>
</tr>
</tbody>
</table>
Web pages

• Hypertext markup language (HTML) is being replaced by XHTML (extensible HTML)
  – XHTML is based on XML (extensible markup language)
  – Grammar is more structured
  – Validation and machine-machine communications are better
• XHTML (and HTML) have tags (markup). Examples:
  – <HEAD> and </HEAD>, <BODY> and </BODY>
  – <P> and </P>, <STRONG> and </STRONG>
• You can write XHTML by hand or with a tool
  – We’ll use Dreamweaver to make it faster and more correct
  – You can write Web pages by hand in Notepad
    • Not MS Word, which has its own markup language, RTF, that gets in the way