The Application Layer:
The Web & HTTP

Overview
- Application protocols: HTTP vs. FTP
  - (use telnet)
  - Architecture: client-server
- Connection and handshaking
  - Control & data connections with FTP
- Message types and format
  - Request and response
  - Request types (get, head, post, ...)
  - Header format
- Caching

HTTP Overview
1) Connection request to port 80 (handshaking)
2) File/data request via dedicated connection/socket

HTTP Request Message
- ASCII (human-readable format)

Basic HTTP request message
- Request line
  - GET /directory/page.html HTTP/1.1
  - Host: www.amherst.edu
  - User-agent: Mozilla/6.0
  - Connection: close

- Header lines
  - Accept-language: en

- Carriage return, line feed indicates end of message
  - (extra carriage return, line feed)
HTTP request message: general format

Example: Uploading form input

the basic GET method:
- The 'entity body' in the request is empty

Post method:
- A web page often includes "form" input
- This input information is uploaded to server in the message's "entity body"
  (but also have the URL method: )
- Uses the GET method
- Input is uploaded in URL field of request line:
  www.somesite.com/animalsearch?monkeys&banana

HTTP response status codes

In the first line of a server->client response message.
A few sample codes:

200 OK
- request succeeded, requested object later in this message

301 Moved Permanently
- requested object moved, new location specified later in this message (Location:)

400 Bad Request
- request message not understood by server

404 Not Found
- requested document not found on this server

505 HTTP Version Not Supported
Trying out HTTP (client side) for yourself

1. Telnet to a Web server (from 'Terminal' window):

   telnet www.science.smith.edu 80

   Opens TCP connection to port 80 (default HTTP server port)
   Anything typed in sent to port 80 at science.smith.edu

2. Type in a GET HTTP request:

   GET /~jcardell/ HTTP/1.1
   Host: www.science.smith.edu

   End with 2 CR
   This is a minimal (but complete) GET request to an HTTP server

Trying out HTTP

What should you type to initiate an HTTP connection via telnet?

   telnet www.science.smith.edu
   Trying 131.229.72.74...

   telnet: connect to address 131.229.72.74: Operation timed out
   telnet: Unable to connect to remote host

Fun With Telnet

- Terminal emulation, for UNIX, to log on to remote computers
- A protocol and an application (using that protocol)
- Poke around on http://www.telnet.org/
http://www.telnet.org/

Miscellaneous fun places
- rainmaker.wunderground.com :: weather via telnet!
- india.colorado.edu 13 (Get the time) :: get the time
- telehack.com 23 :: Telehack
- telehack.com :: Telehack - web
- towel.blinkenlights.nl 23 :: Star Wars ascimation
- towel.blinkenlights.nl 666 :: The Bofh Excuse Server

Other systems
- thehatshop.mudhosting.net 3000 :: Hallowed Halls
- eclipse.cs.pdx.edu 7680 :: New Moon
- batmud.bat.org 23 :: BatMUD
- forgottenkingdoms.org 4000 :: Forgotten Kingdoms
- mush.shelteringcolorado.com 2601 :: Sheltering Sky: Colorado by Night
- igormud.org 1701 :: Igor MUD/
- zombiemud.org 23 :: Zombie MUD
- achaea.com 23 :: Achaea, Dreams of Divine Lands
- gcomm.com 23 :: Galacticomm BBS
- 1984.ws 23 :: 1984

User-server state: cookies

Many major Web sites use cookies

Four components:
1) A new header line of "Set-cookie: nnn" is included in the HTTP response message
2) A cookie header line is then included in the HTTP request message
3) This cookie file (Web server and cookie number) is kept on user's host, managed by user's browser
4) A back-end database at the Web site maintains browsing data
**Caching example**

**Problem**
- delay over access link

**Scenario**
- utilization on LAN = 15%
- utilization of net access link = 100%
- total delay = Internet delay + access delay + LAN delay
  
  = seconds + minutes + millisec

**Solution: install cache**
- Assume that ~40% of HTTP requests can be served by the cache

**Consequence?**
- 40% requests will be satisfied almost immediately
- 60% satisfied by origin server
- use of access link reduced to 60% → negligible delays

---

**Conditional GET – in Wireshark Lab**

- **Goal:** do not send the object if cache has up-to-date cached version
- **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

- object not modified

- HTTP request msg
  
  If-modified-since: <date>

- HTTP response HTTP/1.0 304 Not Modified

- object modified

---

**FTP elements**

- **File transfer protocol**

- **Two connections**
  - Control connection
  - Data connection

- **"Out of band"**

- The control connection maintains state information
**FTP: separate control & data connections**

The FTP Client:
- Contacts the FTP server at port 21
- Obtains authorization over control connection
  - with a username and password
- Transfer files by sending commands over control connection
  - List/Change directory
  - Request to send or receive files ..

![Diagram of FTP client and server connections]

The Server:
- Listens on port 21 for an incoming connection request
- When server receives a request, the server opens a data connection to client (through a different socket, with a different port number)

![Diagram of FTP client and server connections]

**Summary**

- New vocabulary
- HTTP vs. FTP
  - Characteristics
  - Message types
  - Caching
  - Control & data connections
- ASCII format - easy to watch and interpret → Wireshark