Internet Technology

Internet Basics
- LANS and WANS
- Packets and Protocols
- Clients and Servers
- HTML and XML

References
- Comer, The Internet: Everything You Need to Know...
- Comer, Computer Networks and Internets (both © 1997, Prentice Hall)

History of the Internet
- Many ad hoc networks developed in 60’s and 70’s
- Local area networks (LANs) located in a single room or building
- Incompatibility between networks
  - physical (signal levels and types; connectors; ...)
  - message formats (size, content, error handling; ...)
- Mostly commercial networking systems
  - little or no support for heterogeneous networks
  - proprietary specifications

History, cont’d
- Wide Area Networks (WANs) were also developed
- Often connected two or more LANs
- Much more expensive to build and operate
  - long distance phone (e.g. modem)
  - dedicated leased lines (phone lines w/o the phone)
- Centrally administered
- Ad hoc interfaces
  - n LANs [] n² translators

ARPA’s Role
- US DoD Advanced Projects Research Agency
- Funded civilian research in network design
- Goals for a large scale military network:
  - reliable
  - redundant
  - decentralized administration
- Key concept: internetworking
  - “network of networks”
  - single inter-network protocol for traffic between LANs

Internet Organization
- Local administration of participating LANs
- Open specifications
  - defined by RFC (Request for Comment) documents
  - any system could connect to internet by implementing software according to RFCs
- Managed by IETF
  The Internet Engineering Task Force (IETF) is a large open international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet. (http://www.ietf.org)

Point to Point Communication
- To send information from one system to another:
  1. The originating system delivers a message to a LAN node that connects to a router
  2. Routers transfer the message to a node in the receiving LAN
  3. Local communications deliver the message to the final recipient
- Routers maintain local tables that find the best path to the recipient
- Routers can respond to network traffic, pick alternate routes
Protocols

- Application level communication usually requires a sequence of operations
- A network protocol defines
  - the type and format of messages exchanged
  - the meaning of individual message contents
  - the "handshaking" steps required to transfer messages
- Example: SMTP (simple mail transfer protocol)
  
  S: are you accepting messages?  
  R: yes, go ahead
  S: my name is X, I want to send to Y  
  R: Y's not here

Layers

- Complex protocols are defined in terms of layers
- An application relies on a transport protocol to deliver a message
- Transport protocol uses an internet protocol for packets, routing, ...
- Lower layers hide messy physical details

TCP/IP

- "The Internet":
  The set of computers that use TCP and IP for network communications
- TCP: Transmission Control Protocol
  - reliable, full duplex, ...
- IP: Internet Protocol
  - packet switched virtual network

IP Addresses

- Nodes (hosts and routers) have unique addresses
  - LANs translate IP addresses to local physical addresses
- 32-bit number
  - increasing to 128 bits with IPv6
- Usually written in "dotted decimal" notation, e.g.
  
  100000 11011111 00000110 10100010

  06_{16} = 6  
  A2_{16} = 162

  128.223.6.162

IP Addresses (cont'd)

- Addresses have two parts
  - network number (prefix) = 223 in the example
  - host number (suffix) = 1698 in the example
- LAN administrator
  - applies for unique network number
  - assigns host numbers

  128.223.6.162

  100000 11011111 00000110 10100010

  Class B address

  prefix = 223_{10}  
  suffix = 1698_{10}
Symbolic Names

- Users do not have to remember IP addresses
- The Domain Name System (DNS) maps symbolic names to IP addresses
  - Example: cs.uoregon.edu maps to 128.223.6.41
- Each (sub)domain administrator is responsible for assigning symbolic domains
  - A new computer in the CS department can be assigned a name (drosophila.cs.uoregon.edu) and an IP address by the CS network administrator

Dynamic Names

- Systems can acquire IP addresses dynamically
- A system can request an IP address from a DHCP (Dynamic Host Configuration Protocol) server
  - Example: "take your laptop on a trip, plug it in to give a presentation, read e-mail from the hotel, …"
- Home network has one IP address via DSL connection to EFN: router running DHCP assigns local IP addresses to home systems

Early Application Level Protocols

- Telnet
  - Defined for command line interactions (remote login)
- FTP
  - File transfer protocol
- NNTP
  - Network news transfer protocol

NOTE: Don’t confuse an application program (e.g., ftp) with the protocol of the same name
- fetch, ftp, etc all implement the FTP protocol

Clients and Servers

- Most internet communication is based on a client/server model
  - SMTP: mail server, client mail readers
  - DHCP: address server, client computers
  - DNS: name lookup server, TCP/IP program clients
- A server is a continuously running process
- The network interface passes incoming requests to the server process
  - Based on Unix socket abstraction
  - E.g., SMTP uses socket #25

Hypertext Transfer Protocol

- The Hypertext Transfer Protocol (http) was developed by T. Berners-Lee at CERN, 1989-91
- The "world wide web" included:
  - A set of document servers
  - WWW, a text-oriented client that displayed documents
  - HTTP, a protocol for client-server communication
  - HTML, a language used to format documents (more later)
- Innovation: system-independent document views
- Mosaic (NCSA/UIUC) was the first GUI browser
- "The Web": subset of "The Internet" running HTTP

HTTP Example

- Client:
  
  GET /index.html HTTP/1.1
  Host: localhost
  User-agent: Mozilla/4.0

  header (command, args) params

- Server:
  
  HTTP/1.1 200 OK
  Content-Type: text/html
  Content-Length: 226

  header (result type) params

  <HTML>...</HTML>

  blank line content

- NOTE: Stateless connection -- next request is independent of this one
URL

- Browsers typically implement several other protocols in addition to HTTP
- A uniform resource locator is a complete specification of a document on the web:
  - the protocol used to fetch it
  - the name of the system where the document can be found
  - the document ID (e.g. file name)
 Examples:
  - http://cs.uoregon.edu/classes/cis410bio/index.html

Dynamic Documents

- A server is not limited to simply fetching a document to return to a client
- A server can run a program that generates the document dynamically
  - Example: get your current bank account balance

CGI

- The standards that define how a program should generate output returned to a browser are called the Common Gateway Interface
- CGI programs can be written in any language
- Common languages (with libraries to implement CGI standards):
  - Perl
  - Python
  - PHP
  - Java

State

- CGI programs can be difficult
  - Bank example, again:
    - User connects, sees login/password screen
    - Enter info, click login button
    - Response is list of account numbers; click on one to get balance
    - HTTP is stateless -- where is the name/password saved so the second access is successful?
      - Local DB -- saved on bank’s server
      - “cookie” -- data uploaded to user’s machine
      - URL -- info appended to future connections

URL Parameters

- A URL can include a parameter list following the document name
- Parameters are name/value pairs separated by ampersands:
- Not a good idea for the bank: the user’s login and password are sent in plain text, maybe displayed at the top of the browser, saved in a bookmark list, ...

Security

- Most browsers and servers now implement a data encryption standard known as Pretty Good Privacy (PGP)
  - It’s actually very good -- just not as good as the best
- There are now secure versions of the common protocols:
  - SSH (replaces telnet)
  - SFTP
  - SHTTP
Scripting Languages

- Perl, Python, and other “scripting languages” are popular for CGI programs:
  - they were designed to launch other programs
  - extensive string processing facilities
  - libraries for database operations, network connections, ...
- They are also widely available (and free)

CGI Script Example

- This program is in ~conery/public_html/htbin/time.cgi:
  ```perl
  #!/local/bin/perl
  print "Content-type: text/html\n\n"
  print "<html>\n";
  print "<body>\n";
  $time = `date`;
  print "<h1>$time</h1>\n";
  print "</body>\n";
  print "</html>\n";
  ```
- The URL that runs this program is:
  [http://cs.uoregon.edu/~conery/htbin/time.cgi](http://cs.uoregon.edu/~conery/htbin/time.cgi)

Markup Languages

- A “markup language” is a meta-language
- Commands and symbols of the ML are embedded in a document, explain how to format and display the document
- HTML: hypertext markup language
- Commands are bracketed: `<cmd>..</cmd>`
  - `<H1>The time is</H1>`
  - `<A href="http://ncbi.nih.gov">NCBI</A>`

XML

- XML is a recent (1998) language used to describe structured documents
- Markup tags define the begin, end of document parts
- Unlike HTML, XML tags have no meaning — they are used only to organize the document
  ```xml
  <person>
    <firstname>Gregor</firstname> <lastname>Mendel</lastname>
  </person>
  ```

XML (cont’d)

- Why add all these tags?
  - self-documenting
  - text format for binary files
  - “serializer” translates binary to text string
  - string can be parsed, turned back into binary by the receiver
- Many public libraries for generating/parsing XML
- NCBI BLAST has option to generate XML format

Databases

- Most large database systems are implemented with a client-server architecture
- MySQL is an open source SQL relational database
  - `mysql` is a client program with a command line interface
  - users can get data via `mysql` or from a Perl program that uses the MySQL “API”
Bioinformatics Services

- There is a tremendous amount of bioinformatics data available on the Web, provided by the technology described here.
- PubMed at NCBI:
  - Database of scientific literature
  - Browser displays form, user enters author, keywords, etc
  - Database server searches for relevant documents, returns "hits" as an HTML document (with links to document itself, when possible)

Bioinformatics Services (cont’d)

- BLAST servers
  - User enters sequence (or gives name of local file to upload)
  - BLAST parameters entered into other parts of form
  - When form is submitted, the server runs the BLAST program, formats the results, returns as HTML document
  - Document may contain links, e.g. click on name of matching sequence to get info about it

Internet Technology in 410/510

- How will we use these ideas in this class?
- As users
  - connect to NCBI, other web services
- As programmers
  - write scripts that get info from the net
  - write CGI programs to fetch data from a local database, format it and return it