Lecture 1b

WEB

Architecture Overview

All Rights Reserved
The Internet

• see http://www.ietf.org/, http://www.iab.org/ and http://www.w3.org/TR/

• A set of technology
  – Standards & Protocols
    • RFCs (Request For Comment)
      • IP, ICMP, UDP, TCP, SMTP, Telnet, FTP, NNP, NTP, …
  – Client Tools
    • FTP Client
    • Email Readers
  – Protocol servers
    • FTP
    • SMTP/POP3/IMAP (Email)
  – The physical network and interconnections between networks that the protocols run on

• A set of users connected to it
What does the Internet look like?

• A collection of interconnected IP networks
  – Over 250,000! Worldwide networks that route/forward packets in between each other around the world

• All are based on IP
  – not a requirement, actually

• Packet based network
  – not connection based (Mail vs. Telephone metaphor)

• UDP and TCP supported (and IP of course)
  – IP on top of Ethernet, ATM, SONET, FDDI, …
Some Backbones in North America

• AGIS
• ANS
• ATMnet
• BBNplanet
• Compuserve
• CRL
• CWIX
• DataXchange
• DIGEX

• Epoch
• GetNet
• GlobalCenter
• GoodNet
• GridNet
• IBM
• Interconnect
• InternetMCI

• iSTAR
• MCIWorldcom 2000
• NapNet
• Netrail
• NFS
• PsiNet
• Savvis
• Sprint
• UUNET
T1 = 1.54 megabits per second (Mbs)
- DSL / Cable connection can be 1-3 Mbs!
BBN Planet
45 Mbps DS-3 Backbone
internetMCI
45-622 Mbps
Backbone
MCI Global Backbones

MCI’s Global Presence

[Map of MCI Global Backbones with various connections and labels.]
Major Internet ISPs (Internet Service Providers) – they connect you to the internet – directly or indirectly

http://research.lumeta.com/ches/map/gallery/index.html#pookie
Line thickness & color shows relative bandwidth (USA)
Internet2

Shows bandwidth of backbones and saturation of bandwidth

http://loadrunner.uits.iu.edu/weathermaps/abilene/
SuperJANET (in the UK)

http://www.superjanet4.net
Pass your mouse over the Canadian cities to see a complete list of CA*net 4 connected institutions.
“Future Internet” Projects

• Abilene (Internet2)
  – http://www.internet2.org

• SuperJANET4 (Joint Academic NETwork)
  – http://www.superjanet4.net

• CANARIE's CA*net4 (Canada)
  – interconnects the provincial research networks, and through them universities, research centres, government research laboratories, schools, and other eligible sites, both with each other and with international peer networks.
  – Point-to-point optical, most are OC-192 (10 Gbps) speeds.

• DFN's G-WiN (Deutschen Forschungsnetz - German research net)
  – http://www.dfn.de/win/gwin/

• Surfnet's Gigaport (Nederlands)
  – An advanced national infrastructure that connects the networks of universities, polytechnics, research centres, academic hospitals and scientific libraries to one another and to other networks in Europe and the rest of the world.
  – http://www.surfnet.nl/en/#

• Nordunet2v
  – http://www.nordunet2.org
Chicago NAP Connections
(Network Access Point – how your ISP connects to a backbone)

- Adelphia (OC12c ATM)
- Akamai Technologies (OC3c ATM)
- Allegiance Internet (OC12c ATM)
- Ameritech Corp. (DS3 ATM)
- Argonne National Laboratory (Univ. of Chicago) (OC12c ATM)*
- AT&T Canada (OC12c ATM)
- AT&T Worldnet (OC3c ATM)
- BBC Internet (OC3c ATM)
- Bell Nexxia (OC-3c ATM)
- Beyond The network (OC-12c ATM)
- Big Pipe (OC12c ATM)
- Broadwing (OC12c ATM)
- BT Global Services (OC-12c ATM)
- Cogent (OC12c ATM)
- Comdisco (OC3c ATM)
- Electric Lightwave (OC3c ATM)
- EnterAct Corp. (OC-3c ATM)
- Epoch Networks (OC3c ATM)
- Genuity (OC12c ATM)
- Getronix Gov. Solutions (OC-3c ATM)
- Global Crossing (OC12c ATM)
- GT Group (OC3c ATM)
- Harvey Library (DS3 ATM)
- HiNet (AT&T Solutions) (OC3c ATM)
- MCIvBNS (OC12c ATM)
- MCI Telecom (2xDS3)
- McLeodUSA (OC12c ATM)
- Motorola (OC3c+DS3 ATM)
- NetNitco (DS3 ATM)
- One Call Communications (OC3c ATM)
- Qwest (OC12c ATM)
- State of Illinois ISBE (OC-12c ATM)
- Time Warner (OC12c ATM)
- University of Chicago (OC3c ATM)
- University of Notre Dame (DS3 ATM)
- UUNet, An MCI WorldCom Company (OC12c ATM)
- Verio (OC12C ATM)
• Real time how ATT is doing
• Current map of the ATT backbones
The World Wide Web

- “Invented” by Tim Berners – Lee at CERN 1989
- Protocols and Applications that run on “top” of the Internet – HTTP and HTML
- A set of applications that allowed getting documents from another scientists machine, linking to that document from you own documents
  - HTTP Protocol and application demon
  - HTML markup language
  - Primitive browser to read documents in HTML
- Utilized existing protocols like TCP, DNS, …

- WWW = HTML + HTTP + Browser + DNS + Internet
HTML – HyperText Markup Language

• Markup language data format for telling an application how to display the contents of the document

• Things like:
  – fonts, typefaces, bold, italics, underlining
  – lists (Bulleted, numbered)
  – background and foreground colors
  – hyperlinks
  – images and graphics
  – tables
  – metadata (information about the data in the page)
HTML uses tags

• Tags are the things between the “<“ and “>” are tags – they are instructions about how to display/interpret the data between the start tag (<tagname>) and end tag (</tagname>)

• Example HTML tags:
  <html> - first tag in an HTML doc
  <head> - heading (title bar info & metadata)
  <meta> - meta data about the page
  <title> - title for the title bar
  <body> - body of the document
  <p> - paragraph
  <ul> - list
  <li> - list item
  <font> - set the font
  <font size="5"> - size is an attribute of the tag/element
  <a> - anchor (hyperlinks)
  <img> - embeds an image in the text
Sample HTML page in a browser

This is a sample HTML text

This is a sample HTML text - BOLDED

This is a sample HTML text - larger size 5

A bulleted list

- Item 1
- Item 2
- Item 3

A link to the class page
<html>
<head>
<meta http-equiv="Content-Language" content="en-us">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<title>This is the Title of the HTML page</title>
</head>
<body>
<p>This is a sample HTML text</p>
<p><b>This is a sample HTML text - BOLDED</b></p>
<p><font size="5">This is a sample HTML text - larger size 5</font></p>
<p><font size="5">A bulleted list</font></p>
<ul>
<li><font size="5">Item 1</font></li>
<li><font size="5">Item 2</font></li>
<li><font size="5">Item 3</font></li>
</ul>
<p><font size="5">A link to the class page</font></p>
<p><img border="0" src="images/sub2asubway%20map..gif" width="293" height="191"></p>
</body>
</html>
HTML

- A human readable text format that describes what a page should look like when displayed
- Loosely based on SGML
  - Standardized General Markup Language based on GML
  - Around 1980
  - More Complex and Richer markup language than HTML [more ...]
- HTML is an “application” of SGML
- HTML has tags and values and elements and attributes – list of tags [http://www.w3schools.com/tags/default.asp]
HTML - Presentation Oriented

• Presentation of text oriented not data or object oriented
• No clue about the identity/meaning (semantics) of that text.
• An HTML document is just a bundle of strings, not an object or even a data structure.
• No “name” given to a “value” i.e. string of text.
• Only display attributes are supported, like bold, color, and font.
• Need XML to understand the structure of the text in a semantic way (understanding the meaning of the content). Need information about the data (Metadata)
• What you want is (name, value, data type) tuples.
  – See XML schemas/DTD for this.
• Maybe we need is really (name, value, type, presentation attributes, rules including time related behavior) sextuples in or documents?
XML – Extensible Markup Language

• Really XML is a language for defining markup languages (like SGML not an application of a markup language like HTML is)

• XML can be used to define an XML DTD/schema for HTML compliant documents. Thus, HTML could be an application written in XML, too.

• Defining your own markup language (also called an XML doctype or for short XML doc) is creating your own DTD or XSD (Schema)

• More **data/object oriented** not presentation
XML Terms

• **Well Formed Document**
  – has matching tags and follows the XML syntax rules

• **Validating**
  – passes the rules of a DTD or XSD (schema) – must have an associated DTD or schema

• **DTD (Doc Type Definition)**
  – a simple template that defines what a document must “look” like. What it a can contain, can’t contain and where it can occur. No typing of values, everything is string.(not in an XML doc!)
  – You can ignore DTDs, they are old technology!

• **XSD (XML Schema Definition)**
  – a more robust and flexible template for defining an XML document. Supports typing of the values in an XML document e.g. int, long, date, string, …
  – Like a data structure in a programming language like C or Java that the contents of the data structures fields are in a machine readable format

• **XSL (Extensible Stylesheet Language) - a better style sheet than CSS)**
  – defines a “set of rules”/template that tell how to take an XML doc type instance and transforms it to a display format like HTML or PDF or Text.
  – Supports conversions of XML (name, value, type) into other (n,v,t) via XSLT transforms.
  – XSL/XSLT are a Programming language –
    • OS, Hardware, Browser and Character set neutral language
XML Terms (2)

• **Namespaces**
  – Only in Schemas, not DTDs supports the same tag name in a single doc e.g. Name, Address, Phone, inside a document without collisions.
  – Two different schemas (data structure definitions) in a single XML doc without creating problems referencing the correct data structure's contents.
  – Ex:
    • a customer schema
    • a supplier schema
    • Both have name, address and phone elements.
    • You can see a Customer and Supplier namespace in the combined document.
    • Customer.address is different than Supplier.address
33

HTML document

<head>
  <title>this is just HTML not XML</title>
</head>
<body>
  <div>this is just HTML, not XML</div>
  Customer Address: 100 west street, NY, NY, 10006<br>
  Shipping Address: 243 Mercer Street, NY, NY 10008<br>
    Customer Address: 75 Wall Street, NY, NY, 10005<br>
  Customer Phone: 888-888-8888 or 888-888-8889
</body>
</html>

Tags  <head>,
      <title>,
      <body>
Well formed but non-validating XML

```xml
<?xml version="1.0"?>
<head>
    <title>hello</title>
</head>
<body>
    <div>this is valid but not well formed</div>
</body>
</html>
```

- That’s is a legal XML document
  - Its also a legal HTML doc BTW, HTML ignores the “<?xml ...”
- Its well formed XML
- Its not validating XML
  - no schema or DTD to validate against.
Well formed and Valid XML that is also valid HTML (XHTML)

```xml
<?xml version="1.0"?>
<xhtml:html xmlns:xhtml="http://www.w3.org/1999/xhtml">
  <xhtml:head>
    <xhtml:title>validating and well formed XML XHTML HTML</xhtml:title>
  </xhtml:head>
  <xhtml:body>
    <xhtml:div>this is valid but not well formed</xhtml:div>
  </xhtml:body>
</xhtml:html>
```

Namespace is `xhtml`
Schema is `http://www.w3.org/1999/xhtml`
XHTML in a browser

This is a sample of text.

This is a sample [hyperlink](#).

- This is a bullet
- Another bulleted item
- Yet a third bullet

<table>
<thead>
<tr>
<th>This is a cell in table</th>
<th>cell 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>cell 3</td>
<td>cell 4</td>
</tr>
</tbody>
</table>

this is a submit button
XML data doc referencing a non-existent schema

<?xml version="1.0"?>
<mynamespace:entryinschema xmlns:mynamespace="http://www.nyu.edu/poelman/someschemathatdoesn'texist">
    <mynamespace:sometag>
        <title>non-validating and well formed XML</title>
        <!--this is a comment - this line has a non-namespaced tag - default namespace -->
    </mynamespace:sometag>
</mynamespace:sometag>
Same XML with no schema in a browser

<?xml version="1.0"?>
<mynamespace:sometag>
  <title>non-validating and well formed XML</title><!--this is a comment - this line has a non-namespace tag - default namespace -->
</mynamespace:sometag>

This XML file does not appear to have any style information associated with it. The document tree is shown below.

- <mynamespace:sometag>
  <title>non-validating and well formed XML</title>
  <!--this is a comment - this line has a non-namespace tag - default namespace -->
</mynamespace:sometag>
IE and Firefox disagree whether no schema reference is an error or not.
An XML doc that is just a data structure, no schema and no presentation tags

```xml
<?xml version="1.0"?>
<book_information>
  <title>Non-validating and well formed XML</title>
  <author>Logan Poelman</author>
  <email_address>lpp@lpp.com</email_address>
  <publisher>O'Really</publisher>
  <address>
    <street>243 mercer</street>
    <city>New York</city>
    <state>NY</state>
    <zip>10008</zip>
  </address>
</book_information>
```

- Is it Well Formed? Validating? Valid?
- Most browsers will still display it (IE, FF, Opera)
- How could you make this into an HTML doc with presentation tags?
  - use XSL!
<?xml version="1.0"?>
<head>
<title>this a HTML version of the XML data structure</title>
</head>
<body>
<book_information>
  <h1>Book Info:</h1>
  Author: <author>Logan Poelman</author><br/>
  Email: <email_address>lpp@lpp.com</email_address><br/>
  Pub: <publisher>O'Really</publisher><br/>
  <bold>Address:<address>
    Street:<street>243 mercer</street>
    City:<city>New York</city>
    State:<state>NY</state>
    Xip: <zip>10008</zip>
  </address></bold>
</book_information>
</body>
XML editors

• Lots out there, most are not very good.
• For a trial of a java based XML editor go to http://www.xfytec.com
• NetBeans has some capabilities built in
• Fix the next slide!!!!!
Internet/WWW Protocol Layers

“roughly” OSI Layers

Application Protocol
SNMP  RTP  DNS  NFS  NFS  FTP  TelNET  POP3  SMTP  SIP  SSH  HTTP(S)  LDAP

Session & Transport
UDP (Unreliable)  TCP (Reliable)  SCTP

Network
IPv4 / IPv6

Data Link
SLIP/PPP  Frame Relay  X.25

Physical Medium
FDDI  ATM  Ethernet 802

   SDH / SONET  Fiber Optical
   ADSL  ISDN  T1  10baseT  1000baseT

Wire Based Cabling
Open Systems Interconnection (OSI) Reference Model

Upper Layers

- Application Layer (7)
  - E-mail
  - Newsgroups
  - Web Applications
  - File Transfer
  - Host Sessions
  - Directory Services
  - Network Mgt.
  - File Services

- Presentation Layer (6)
  - POP/SMTP
  - Usenet
  - HTTP
  - FTP
  - Telnet
  - DNS
  - SNMP
  - NFS

- Session Layer (5)
  - POP/25
  - 532
  - 20/21
  - 23
  - 53
  - 161/162

- Transport Layer (4)
  - Transmission Control Protocol (TCP)
  - User Datagram Protocol (UDP)

Lower Layers

- Network Layer (3)
  - Internet Protocol Version 6
  - Internet Protocol Version 4

- Data Link Layer (2)
  - SLIP, PPP
  - 802.2 SNAP

- Physical Layer (1)
  - RS-X, CAT 1
  - ISDN
  - ADSL
  - ATM
  - FDDI
  - CAT 1-5
  - Coaxial Cables
OSI Model

- Layered Model for Computer Communications
  - [http://www.answers.com/OSI%20model](http://www.answers.com/OSI%20model)

<table>
<thead>
<tr>
<th>Layer</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Type of communication: E-mail, file transfer, client/server.</td>
</tr>
<tr>
<td>Presentation</td>
<td>Encryption, data conversion: ASCII to EBCDIC, BCD to binary, etc.</td>
</tr>
<tr>
<td>Session</td>
<td>Starts, stops session. Maintains order.</td>
</tr>
<tr>
<td>Transport</td>
<td>Ensures delivery of entire file or message.</td>
</tr>
<tr>
<td>Network</td>
<td>Routes data to different LANs and WANs based on network address.</td>
</tr>
<tr>
<td>Data Link</td>
<td>Transmits packets from node to node based on station address.</td>
</tr>
<tr>
<td>Physical</td>
<td>Electrical signals and cabling.</td>
</tr>
</tbody>
</table>
IP

- Internetworking Protocol – developed around 1975
  - Replaced original internet network protocol NCP
- Runs on top of “transport” protocols:
  - Ethernet (IEEE 802.x) protocol and others (ATM, FDDI, X.25)
- Provides sending packets between addresses
- “Fire and forget” protocol – no guarantee of delivery/order/duplicates/lost packets
- IPv4 addresses are 4 numbers separated by dots
  - 10.255.0.128 (all numbers are between 0-255\textsubscript{d} or 0x00 - 0xFF hexadecimal or a 32 bit binary number. It takes 4 bytes to hold the address.)
  - Means \(2^{32} = 4,294,967,296 - 1 \approx 4.29 \times 10^9\) possible addresses!
  - Like “phone numbers” for internet hosts
IP (IPv4)

- Addressing subdivides “spaces” that may be granted to a group, corporation, university, individual, etc.
  - Class A – 1.a.c.d to 126.b.c.d (16+ million nodes)
  - Class B – 128.x.c.d 191.x.c.d (65,534 nodes)
  - Class C – 192.255.x.d 233.255.c.d (254 nodes)
  - Class D – 224.255.c.d 239.255.c.d (?? nodes)
  - Class E – 240.255.c.d 255.255.c.d – reserved
    *X is a number assigned to you!

- Because of class addressing, there are far fewer than $2^{32}=4$ billion addresses, in reality.
- CIDR and IPv6 will fix this.
IP

• Connectionless
  – like sending a letter not like a phone call
  – Handshake, Send Packet, Stop Communication
  – No state info preserved between packet sends

• Addressing and route selection

• Packet assembly and disassembly

• Header (these are the fields in the packet header)
  – IP Version (IPv4, IPv6)
  – Source Address – 4 bytes IPv4, 16 bytes IPv6
  – Destination Address – 4 bytes IPv4, 16 bytes IPv6
  – Checksum
  – TTL (Time To Live) – number of router hops allowed
  – Total Length in bytes
  – Payload – between 1 – 64K bytes long (max)
IP Addresses

- Assigned to a sender/receiver on the network
- Static –
  - Never changes
  - Assigned to the MAC address (Unique ID embedded in the Network Interface Card).
- Dynamic –
  - Assigned by a piece of software on the LAN that maintains a pool of addresses and hands them out as needed.
  - IP addresses are **leased** and **expire**
  - Using **DHCP** (Dynamic Host Configuration Protocol) your machine gets new IP Addresses.
  - Efficient and flexible on a network with machines moving around or attaching and unattaching.
  - Look at the info. windows use ipconfig.exe, unix ??
How do you find out a host’s IP address?

- They (the host’s system admin) tells you
- You look it up
  - **Hosts file** on your machine – maps names to ip addresses. (original way)
    - Names can be any form ex:
      - 121.6.7.8  www.thisthing.com
      - 121.6.7.9  foobardobar
    - Doesn’t scale (grow over time) well
  - use DNS
DNS

• Domain Name Server
• Converts from an web URL to an IP address (no ports that’s UDP/TCP)
  – www.nyu.edu is 128.122.108.9
  – on windows try ipconfig, winipconfig
  – on unix try ??
• Based on UDP on port 53
• Contains static IP addresses of hosts
• Periodically updated
DNS (cont)

• A set of servers that store all the top level domain names i.e. those that are:
  - www.xyz.com
  - .edu
  - .mil
  - .gov
  - .us / .uk / .tv / .de (or any other country code)
  - see www.alldomains.com/alltlds.html

• Your browser talks to a specified DNS server to look things up
• Returns IP addresses only (browser determines the presentation protocol (HTTP, SMTP, FTP, ...), transport protocol (TCP or UDP) and port address)
• When you add a new host you must register the domain name and IP address with a DNS somewhere, usually through your ISP.
• DNS Servers share changes between themselves, periodically
• Root servers are updated daily and managed by ICANN
TCP

• Transmission Control Protocol
• Connection oriented
  – Like a phone call – dial, connect, talk talk talk, hangup
• Not a client / server protocol, really more a peer to peer – either can send data to the other whenever they choose
  – Initiator and Acceptor (often wrongly called Client and Server)
  – Ex: you call mom, but she does all the talking! Not client/server.
• Guaranteed packet delivery
  – error free, received in, no duplicate packets
• Runs on top of IP
• Slower than UDP
• Each IP Address has 65536 TCP Ports
  – Like phone extensions on a phone number.
TCP Socket

- The connection between hosts is called a “socket”
- IP Address + TCP Port Numbers of Client and Server
- Always PTP – Point to Point – like a phone call
- Handshaking between Initiator (Client) TCP Port and Acceptor (Server) machine. Establishes a socket connection.
- Packets sent get acknowledged by the receiver.
- TCP handles congestion control – slow down sending packets if the receiver can’t keep up
- Sockets were originally OS dependant implementation
- BSD Unix Sockets was first OS independent version
TCP Packet

- Source TCP Port – 2 bytes
  - Address is in the IP Header
- Target TCP Port - 2 bytes
  - Address is in the IP Header
- Sequence Number – 4 bytes = 0 to \((2^{32}-1)\) - increments
- Acknowledgement Number
- Window – size of the senders buffer
- Checksum – 2 bytes
- Control Bits
- Packet Size is in the IP header
- TCP data is in the IP Payload
Well Known TCP Ports (0-1023)

- 80  HTTP
- 443  SSL
- 20 & 21  FTP (20 control, 21 data)
- 25  SMTP
- 110  POP3
- 23  TelNet
- 161  SNMP
- 7  Echo
- 53  DNS
- others
UDP

• User Datagram Protocol
• Not connection oriented
  – Like Sending a Letter – address it, drop in the mailbox and hope it gets there
• “Fire and forget” like IP
• No guarantee
  – lost packets, dups, and out of sequence packets can happen
• Faster than TCP – lower overhead
• UDP Port + IP Address
UDP Packet

• Inside the payload of an IP packet
• Source & Destination Port
  – Addresses are in the IP Header
• Checksum – 2 bytes
• Message Length is in IP Header
• The UDP data is in the IP Payload
FTP Example

FTP client  FTP Server

“roughly” OSI Layers

Application Protocol

Transport

Network

Data Link

Physical Medium

“Only on top of lower protocol”

FTP

TCP (Reliable)

IPv4 / IPv6

Ethernet 802

SDH / SONET

Fiber Optical

ADSL

Wire Based Cabling
Email (POP3 & SMTP) Example

- Email Client
- Email Server
- POP3
- SMTP
- TCP (Reliable)
- IPv4 / IPv6
- ATM
- Ethernet 802
- SDH / SONET
- 1000baseT
- Fiber Optical
- Wire Based Cabling

"roughly" OSI Layers
Application Protocol
Transport
Network
Data Link
Physical Medium

Only on top of lower protocol
NFS (Network File Sharing)

Sys A OS Sharing Files

Sys B OS Sharing Files

“roughly” OSI Layers

Application Protocol

Transport

Network

Data Link

Physical Medium

NFS

UDP (Unreliable)

IPv4 / IPv6

Ethernet 802

ADSL

T1

Wire Based Cabling
IPv6

- IPv6 has a 128 bit address field.
  - \(2^{128} \approx 3.4 \times 10^{38} \approx 3.4 \times 10^{27} \times 10^9 \approx 3.4 \times 10^{27}\) billion addresses!
  - Enough for every atom on the earth to have its own IP address.
- Addresses look like:
  - 9000:110:1601:211:3:50:60:1
  - 1:2:3:4:5:6:7:8
  - 8 - 16 bit numbers separated by colons (not dots)
- Larger headers than IPv4
- Host to host security
- Mobile Devices – Phones, PDAs
- Peer to peer and routable addresses
- Examples needing IPv6 Address Space
  - Content sharing (Napster, Kazaa, Morpheus, etc.
  - Distributed data processing (grid computing)
  - SETI@home
  - Serverless groupware
  - Multimedia conferencing
    - Home Appliances
      - Refrigerators
        - Electrolux Screenfridge, LG Internet Refrigerator, Samsung Digital Network Refrigerator
    - Security/Fire Alarm systems
    - Internet-enabled ATM machines
      - Fujitsu Series 8000, Infonox
    - Internet-enabled luxury cars
    - Playstation / XBox
IPv6

• Improvements over IPv4
  – Autoconfiguration
  – Security (authentication, encryption)
  – Address space
  – IPSEC built into headers of IPv6
  – Authentication
  – Supports new security schemes (“future proof”)

• Potential Problems
  – Existing devices can’t handle the new features
  – Devices will need redesign
  – Routing table size
  – NAT induced restrictions
  – Router overhead growing
  – IPv4 to/from IPv6 networks
  – Rewriting of other protocols to support it – OSPF, ICMP, BGP, …
URL — Universal Resource Locator

- **protocol :// address : port / path #anchor**
  - ftp://www.cs.nyu.edu (defaults to 20&21)
  - smtp://
  - http://www.yahoo.com (defaults to :80)
  - http://www.yahoo.com:80
  - http://64.58.76.178 (same as yahoo.com)
  - file://

- The protocol (and port) is used by the Browser (or other applications) to know how to communicate with a server or peer at the other end of the TCP/IP communication (unless file://).

- The address is used by DNS to resolve URL to a IP Address
HTTP

- HyperText Transfer Protocol
- Developed by Tim Berners-Lee at CERN in 1989-91
- Designed to share research information between users as HTML documents – Hyperlinks allowed navigation between parts of a document to other related documents

“The Hypertext Transfer Protocol (HTTP) is an application-level protocol with the lightness and speed necessary for distributed, collaborative, hypermedia information systems. It is a generic, stateless, object-oriented protocol which can be used for many tasks, such as name servers and distributed object management systems, through extension of its request methods (commands). A feature of HTTP is the typing of data representation, allowing systems to be built independently of the data being transferred.”

- Tim Berners-Lee, et al. IETF Working Draft 1991 HTTP/1.0
http://www.w3.org/Protocols/HTTP/1.0/spec.html
HTTP Browser

• Initial browsers were text only, no images in the text.
• What started the revolution was MOSAIC browser
  – Embedded graphic images in the text when displayed
  – Workstudy $6.75 / hour job. He left and created Netscape.
    http://www.ibiblio.org/pioneers/andreesen.html
  – By May, 1998, Andreesen had made $52 million from netscape!
  – Part of his code ended up in IE (not netscape, though)
• **HTTP Port 80** is the default but can be configured on a different port say 8080 or 7000
• Is a client/server based protocol.
  – Client (Usually a web browser) initiated the communication with a web server (HTTP server).
  – Server only responds to requests made by the client, **never initiates sending information to the client**.
• Designed to transfer **static documents** (HTML pages).
• Not designed for dynamic applications like getting data from a database
• See [http://www.w3.org/Protocols/HTTP/1.0/spec.htm](http://www.w3.org/Protocols/HTTP/1.0/spec.htm)
Browser Client – Web Server

- Browser is the client
- Server sends a reply
- Web Server is the server
WWW (HTTP) Example

- **Application Protocol**: DNS, HTTP(S)
- **Transport**: UDP (Unreliable), TCP (Reliable)
- **Network**: IPv4 / IPv6
- **Data Link**: Ethernet 802
- **Physical Medium**: Fiber Optical, ADSL, Wire Based Cabling

“roughly” OSI Layers

Only on top of lower protocol
HTTP Functions

• Invoked as:

- `http_URL = "http:\://" host [ ":" port ] [ abs_path [ "?" query ] ]`

• HTTP Functions:
  - **GET** returns the item defined by the URN. Header and body. Metadata.
  - **HEAD** returns the header of the get not the body. Metadata about the item
  - **POST** sends data to the server from an HTML form
  - **PUT** puts the item on the server. Allows putting a HTML doc to a server.
  - **OPTIONS** returns the options supported by the HTTP server
  - **DELETE** removes an item from the server like a file delete via HTTP
  - **TRACE** returns the request back to the client like an HTTP echo
  - **CONNECT** used with proxy server (advanced)
  - **extension-method** allows adding new functions to HTTP server (proprietary)
“Structure” for a HTTP request

Request =

Request-Line
*(
( general-header | request-header | entity-header )
CRLF)
CRLF
CRLF
[ message-body ]
“Structure” for a HTTP response

Response =

Status-Line

* ((general-header | response-header | entity-header)
  CRLF)
  CRLF
  [ message-body ]
Sessionless and stateless

- Browser connects, makes request, gets response and the server (or client) disconnects.
- No information (no state) about the browser is maintained by the server *between connections*  
  - no session.
- Efficient for operations where the server is waiting around a lot for browsers to make requests.
- Inefficient for long, session oriented things like shopping online
  - The “shopping cart” problem - you don’t want the system to forget the items you have placed in your “shopping cart” if you wait to long and the HTTP connection times out.
  - You (your code) must somehow create a session on top of HTTP
  - HTTP 1.1 actually has support for connections that last over multiple requests – scales better for long sessions
MIME

• Multimedia Internet Mail Extension (IETF 1992)
• A way to encode information send in email so that the receiver understands what the data is.
• Converts binary and non-US characters into 7 bit ASCII stream, then at the receiver can convert it back into the original data set
• mime header:
  – mime version
  – content type/sub type
  – content transfer encoding
  – content ID
MIME Header fragments

- MIME-Version: 1.0
  Content-type: text/plain; charset=us-ascii (Plain text)
- MIME-Version: 1.0
  Content-type: text/plain; charset="us-ascii"
- MIME-Version: 1.0
  Content-Type: text/plain; charset=ISO-8859-1
  Content-transfer-encoding: base64
S/MIME

• Secure MIME

• S/MIME was originally developed by RSA Data Security, Inc.

• Currently uses:
  – X509v3 Certificates
  – TRiple DES encryption
  – Diffie-Hellman with DSS Signatures
  – SHA-1 Hash Algorithm

• Encrypts and signs MIME messages
Internet/WWW Protocol Layers

OSI Layers

Application Protocol
- SNMP
- RTP
- DNS
- NFS
- NFS
- FTP
- TelNET
- POP3
- SMTP
- SIP
- SSH
- HTTP(S)
- LDAP

Session & Transport
- UDP (Unreliable)
- TCP (Reliable)
- SCTP

Network
- IPv4 / IPv6

Data Link
- SLIP/PPP
- Frame Relay
- X.25
- FDDI
- ATM
- Ethernet 802
- SDH / SONET
- Fiber Optical
- ADSL
- ISDN
- T1
- 10baseT
- 1000baseT

Physical Medium
- Wire Based Cabling

“roughly”

physical medium
Architecture and WWW

• Application architectures can be broken down into layers. Common layering is a 3 tier (layer) architecture
  – **Presentation Logic** - does interaction with a User. Code that handles displaying data in terms of the look e.g. colors, fonts, arrangement on the page. Also, handles the input from the user.
  – **Application Logic** – business logic of an application. Does calculations, manipulation and the essence of what the application does for you. This is the code that would be the same if you accessed it via a web browser, desktop UI or phone UI
  – **Data Logic** – data access/manipulation logic. Stores and retrieves data from disk. The code that would change if you changed databases or files.

• WWW
  – Presentation Layer = Browser + WebServer
  – Application Logic Layer = App Server (Servlets with no HTML, java classes, POJO[Plain Old Java Objects])
  – Data Logic = Database or files
Tiers

• **Presentation** – formatting information to display it. Checking input with simple checks.

• **Business (Biz)/Application** Logic
  – calculations, rules that are business level
    \[(\text{if age}<18 \text{ then don’t sell them beer})\]

• **Data Logic**
  – storage and manipulation of **persistent data**.
  – Things like name, address, phone, invoices, orders, grades,…
  – Database or file or dataset based storage
Why Tiers/Layers?

• Allow building a more flexible application
• Allows changing parts without changing the whole – changing the database from Oracle to MySQL, for instance
• “Separation of concerns” – design pattern
• Allows reusing functionality in a different application
  – Different Screens for different users
    • Ex: Employee **view** and HR Administrator views of data in the enterprise applications that store the employees address, for instance. (one is read only privileges maybe, the other Read/Write)
  – Could be two different applications that **share** an application component called “employee” and use a method of that component called “getEmployeeAddress()”
Tiers (Analogy)

• A Skyscraper has different layers (or zones):
  – Tenant Floors – Offices, Apartments, etc.
  – Roof – Air conditioning Equipment, Water Tank, Antennas, Heliport, …
  – Equipment Floor – often in the middle of a building’s total floors. It has various systems that support the floors above.
  – Basement – Electrical Power, Heating, Water, Sewer, Plumbing, Telecom, …

• Different types of floors (Layers) serve different functions and have different structure, look, access/security rules, etc.
• The layers work together by separating different concerns and interfacing with each other in well defined ways.
Summary

• The WWW is based on HTML + HTTP + DNS + Internet
• HTTP is based on TCP based on IP …
• TCP is connection oriented
• HTTP is sessionless and stateless
• HTTP uses MIME to encode information passed back and forth between the client and the server
• HTML is presentation oriented markup language
• XML language for creating markup languages
  – It is data oriented and can be used to develop markup languages including HTML or various XML schemas
• The general architecture of WWW applications is 2 or 3 tier – presentation, application and data tiers
For Next Week

• Do Homework #0
  – http://www.cs.nyu.edu/courses/fall05/G22.3033-008/homework_assignment_0.htm

• Learning Java - Chap 1-4
  – If you don’t read ahead you will get lost in lectures and homework, fast!
HTML page in a browser

This is a sample of text.

This is a sample hyperlink

- This is a bullet
- Another bulleted item
- Yet a third bullet

<table>
<thead>
<tr>
<th>This is a cell in table</th>
<th>cell 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>cell 3</td>
<td>cell 4</td>
</tr>
</tbody>
</table>

this is a submit button
Sample HTML Form

<html>
<head>
<meta http-equiv="Content-Language" content="en-us">
<meta name="GENERATOR" content="Microsoft FrontPage 5.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<title>Sample HTML page that has a form</title>
<meta name="Microsoft Theme" content="global 110, default">
<meta name="Microsoft Border" content="tlb, default">
</head>

<body>

<p>Sample HTML page that has a form</p>
<form method="POST">
<p>This Text is in the form</p>
<p>Enter Your Name <input type="text" name="UserNameTextBox" size="20"></p>
<p>Enter Your State <select size="1" name="UserState">
<option value="AL">AL - Alabama</option>
<option value="NY" selected>NY - New York</option>
<option value="WI">WI - Wisconsin</option>
</select></p>
<p><input type="submit" value="Submit" name="SubmitButton"><input type="reset" value="Reset" name="ResetButton"></p>
</form>
<p>&nbsp;</p>
</body>
</html>