The Fundamentals

of Internet and WWW

[Comments: this lecture is not essential but necessary for the completeness of this course]

An Aside That is Relevant

The Current Trends of Computer Languages

TIOBE Programming Community Index for the current year

http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

A Brief Intro to the Internet

Origins of Internet

- ARPAnet - late 1960s and early 1970s
  - Network **reliability**
  - For ARPA-funded research organizations
- BITnet, CSnet - late 1970s & early 1980s
  - ***email*** and ***file transfer*** for other institutions
  - **NSFnet** - 1986
  - Originally for non-DOD funded places
  - Initially connected only five supercomputer centers
  - By 1990, it had replaced ARPAnet for many **non-military** uses
  - Soon became the **network for all** (by the early 1990s)
- NSFnet eventually became known as the **Internet**

What the Internet is?

- A world-wide *network* of *networks of* computers

- At the lower level, since 1982, all connections use TCP/IP

- TCP/IP hides the differences among devices connected to the Internet

The Internet Protocol Suite (4-L Model)

1. Application Layer
   - BGP · DHCP · DNS · FTP · GTP · HTTP · IMAP · IRC · Megaco · MGCP · NNTP · NTP · POP · RIP · RPC · RTP · RTSP · SDP · SIP · SMTP · SNMP · SOAP · SSH · Telnet · TLS/SSL · XMPP · (more)
2. Transport Layer
   - **TCP** · UDP · DCCP · SCTP · RSVP · ECN · (more)
3. Internet Layer
   - **IP** (IPv4, IPv6) · ICMP · ICMPv6 · IGMP · IPsec · (more)
4. Link Layer
   - ARP/InARP · NDP · OSPF · Tunnels (L2TP) · PPP · Media Access Control (Ethernet, DSL, ISDN, FDDI) · (more)

Internet Protocol (IP) Addresses

- Every node has a **unique numeric address**

- Form: 32-bit binary number
  - New standard, IPv6, has 128 bits (1998)

- Organizations are usually assigned groups of IPs for their computers

Domain names

- Example: www.cs.siu.edu

- Form: host-name.domain-names

    - First domain is the smallest; and the last is the largest

    - Last domain specifies the **type** of organization

- *Fully qualified domain name* - the host name plus all the domain names

- **DNS servers** - convert fully qualified domain names to IP addresses

Problem occurred:

By the mid-1980s, several (actually many) different protocols had been invented and were being used on the Internet, all with different user interfaces and conventions (Telnet, FTP, Usenet, mailto, etc.)

- Ordinary Users easily felt **overwhelmed**!!

The World-Wide Web

- A possible solution to the proliferation of different protocols being used on the Internet

- *Origins*

  -Tim Berners-Lee at CERN proposed the Web in 1989

  -Purpose: to allow scientists to have access to many databases of scientific work through their own computers

  - Document form: hypertext

  - Hypermedia – more than just text – images, sound, etc. as well

  - Pages? Documents? Resources?

    - We'll call them **documents** **(but remain open to other terms)**

Web or Internet?

- The **Web** mainly uses one of the protocols, **http**, that runs on the **Internet** -- there are many other protocols used as well (telnet, mailto, etc.)

Web Browsers

- Mosaic - NCSA (Univ. of Illinois), in early 1993
  - First used a GUI, led to the explosion of Web use
  - Initially for X-Windows under UNIX, but was ported to other platforms by late 1993

- Browsers are clients - always **initiate**, servers **react** (although sometimes servers require responses)

- Most requests are for existing documents, using HyperText Transfer Protocol (HTTP)

  -But some requests are for program execution, with the output being returned as a document and sent back to client

Web Servers

- Provide responses to browser requests, with either ***existing*** documents or ***dynamically*** built documents

- Browser-server connection is now maintained through more than one request-response cycle

- "**All**" communications between browsers and servers use Hypertext Transfer Protocol (HTTP)
  - Now allows other protocols, ftp, gopher, news, mailto, …

- Web servers run as background processes in the operating system
  - Monitor a communication port on the host, accepting HTTP messages when they appear

- All current Web servers came from either
  1. The original one from CERN
  2. The second one from NCSA

- Web servers have two **main directories**:
  1. Document root (servable documents)
  2. Server root (server system software)

- Document root is accessed ***indirectly*** by clients
  - Its actual location is set by the server configuration file
  - Requests are ***mapped*** to the actual location

- Virtual document trees – secondary areas for docs

- Virtual hosts -- secondary hosts (more than one sites)

- Proxy servers – serve docs in doc root of other machines

Two Popular Web Servers in Use

- **Apache** (open source, fast, reliable)
  - Started as the NCSA server, named ***httpd***
  - Maintained by editing its configuration file

- **IIS** (Internet Information Server) from Microsoft
  - Maintained through a program with a GUI interface

URL (Uniform Resource Locator)

- General form:

  scheme:object-address

  - The “scheme” is a communication protocol, such as *telnet* or *ftp*

- For the ***http*** protocol, the object-address is:
  fully-qualified-domain-name/doc-path

- For the ***file*** protocol, only the doc-path is needed

- Host name may include a **port number**, as in zeppo:80 (80 is the default, so this is **silly**)

- URLs cannot include spaces or any of a collection of other special characters (semicolons, colons, ...)

- The doc-path may be abbreviated as a *partial path*
  - The rest is then furnished by the server configuration
  - such *relative addressing* is advantageous!

- If the doc-path ends with a **slash**, it means it is a directory

Multipurpose Internet Mail Extensions (MIME)

- Originally developed for ***email***, now commonly for the Web
  - to specify to the browser the form of a file returned by the server (attached by the server to the beginning of the returned document)

- Type specifications
  - Form:

    type/subtype

  - Examples: text/plain, text/html, image/gif, image/jpeg

- Server gets the ***type*** from the requested file name's suffix for example, *html* and *htm* imply text/html

- Browser gets the type **explicitly** from the server

- *Experimental types*

  - Subtype begins with **x- e.g**., video/x-msvideo

  - Experimental types require the server to send a ***helper*** application or ***plug-in*** so the browser can deal with the file

The HyperText Transfer Protocol

The main protocol used by ALL Web communications

- *Request Phase*

- Form:

  HTTP method domain part of URL HTTP ver.

  Header fields

  blank line

  Message body

- An example of the first line of a request:

  GET /cs.uccp.edu/degrees.html HTTP/1.1

- *Most commonly used **HTTP methods***:

GET - Fetch a document
POST - Execute the document, using the data in body
HEAD - Fetch just the header of the document
PUT - Store a new document on the server
DELETE - Remove a document from the server

- Four categories of header fields:

  general (for general info such as date),
  request (used only in request message),
  response (only for response),
  entity (for both request and response)

- Common request fields:

  Accept: text/plain
  Accept: text/*
  If-Modified_since: date

- Common response fields:

  Content-length: 488

  Content-type: text/html

Note: You can communicate server with HTTP **without** using a browser, for example:

> ***telnet*** blanca.uccs.edu http

/* connects to the server, then can run http commands: */

GET /respond.html HTTP/1.1

Host: blanca.uccs.edu

- Response Phase

- Form:

  Status line

  Response header fields

  blank line

  Response body

- Status line format:

  HTTP version status code explanation

- Example:

  HTTP/1.1 200 OK

  (Current version is 1.1)

- Status code is a three-digit number; first digit specifies the general status
  - 1 => Informational
  - 2 => Success
  - 3 => Redirection
  - 4 => Client error (e.g., 400 URL Error and 404 Not Found)
  - 5 => Server error

- **The header field, content-type, is always required!**

- An example of a complete **response header**:

```
HTTP/1.1  200  OK
Date: Tues, 18 May 2004 16:45:13 GMT
Server: Apache (Red-Hat/Linux)
Last-modified: Tues, 18 May 2004 16:38:38 GMT
Etag: "841fb-4b-3d1a0179"
Accept-ranges: bytes
Content-length: 364
Connection: close
Content-type: text/html, charset=ISO-8859-1
[_____]
```

- Both request and response headers must be followed by a *blank line*

Security

- There are many kinds of security problems with the Internet and the Web
  - IE seems constantly having security holes

- One fundamental issue is to move data between a browser and a server without being intercepted during the process

- ***Web security is still a major issue, leads to nice jobs*** 
  - ***You may need first become a topnotch hacker***

- **Security issues** for communication between browser and server:

    1. Privacy – for confidential info, e.g., ssn, not exposed
    2. Integrity – e.g., credit card # not being modified
    3. Authentication -- to be certain with identity
    4. Nonrepudiation – can prove a message is being sent/received

- The basic tool for privacy and integrity is ***encryption***

Basics of Encryption

- If the sender and the receiver both use the same encryption key, the key must be transmitted from the sender to the receiver

  - *Solution*: (1976, Diffie and Hellman)
  - **Public-key encryption**
    - *Use a public/private key pair*
    - Everyone uses a public key to encrypt messages sent to you
    - You decrypt them with your matching private key
    - It works because it is *virtually impossible* to compute the private key from a given public key

  - RSA is the most widely used public-key algorithm

Other Security Issues

- ***Another*** *(indirect) security problem*: *destruction of data* on computers connected to the Internet
  - Viruses and worms
- **Yet another** common security problem:

  Denial-of-Service (DoS)
  -- Created by flooding a Web server with requests
- This course does not particularly address security issues!
  - But we do have dedicated courses on security:
  - CS 408 Applied Cryptography
  - CS 410 Computer Security

The Web Programmer's Toolbox

- ***HTML***

  - To describe the **general** form and layout of documents

  - An HTML document is a mix of content and controls

    - Controls are tags and their attributes

      - Tags often *delimit or mark up* content elements
        - While *presentation* is left to CSS and browser to explain

      - Attributes provide *additional* info about the content of a tag

- ***Tools for creating XHTML documents***

  - HTML editors - make document creation easier
    - Shortcuts to typing tag names, spell-checker,

  - WYSIWYG HTML editors
    - Need not know HTML to create HTML documents

- ***Plug ins***
  - **Integrated** into tools like word processors, effectively converting them to WYSIWYG XHTML editors, like the current MS Word

- **External *Filters/Converters***
  - Convert documents in other formats to HTML

- ***Advantages*** *of both Converters plug-ins*:
  - Can convert existing documents in other forms to XHTML
  - Use a familiar tool to produce XHTML

- ***Disadvantages*** *of both filters and plug-ins*:

-HTML output of both is not perfect - must be fine tuned – bad experience with MS Word, have you?

- HTML may be non-standard

-You will have two (or more) versions of the same document, which are difficult to synchronize

Note: This course requires you to work with a plain editor!

- e.g. NotePad, WordPad, etc.

- ***XML***

  - A **meta-markup** language

  - Used to create a new markup language for a particular purpose or application area

  - Because the tags are designed for a more *specific* area, they can be more meaningful →Semantic Web!

  - Not for presentation, but for contents

  - A simple and universal way of representing data of **any** kinds – not just *textual* data

- ***JavaScript***

  - A client-side XHTML-embedded **scripting** language

  - JavaScript is a dialect of the ECMAScript standard

  - Only related to Java through name & syntax – nothing else!

  - Dynamically typed and object-based (not object-oriented?)

  - Provides a way to access elements of HTML documents on the fly and dynamically change them

- ***PHP***

  - A server-side scripting language (very popular nowadays)

  - An alternative to CGI and Servlets, etc.

  - Similar to JavaScript (but works on the server side)

  - Great for **form processing** and **database access** through the Web

- **Ajax** (shorthand for Asynchronous JavaScript and XML) is a group of interrelated web development techniques mostly used on the client-side but with collaboration from the server-side in order to create interactive and efficient web applications.
  - Much **faster** for Web applications that have **extensive user/server interactions**
  - Uses **asynchronous requests** to the server
  - Requests and receives **small** parts of documents, resulting in much **faster** responses
  - Embodies “divide and conquer” philosophy

- *ASP.NET* (not covered by this course)

- **ASP** (Active Server Page) is a Web server technology from Microsoft, allows for the creation of dynamic, interactive sessions with the user, introduced with IIS as MS's alternative to CGI and JSP .
- **ASP.NET**, also known as ASP+, is an enhanced version of ASP for use on Microsoft's .NET platform
- The **.NET** environment of Microsoft is a platform/framework intended to compete with J2EE
- Allows many .NET languages to be used as server-side scripting language