Chapter one

HTTP – the Hyper-Text Transfer Protocol (2 week)

Introduction

The history of the Internet dates back to the Cold War days of the late 1950s. It evolved when the US Defense Force began to investigate a method of geographically dispersing their centralized computer system. It was believed that reducing reliance on one single route for transmission of data and using decentralized system, would provide a safer option for controlling their missiles. This idea was a safe guard to protect the flow of communications in the event of a major interruption. A 'fear' that in the event of a nuclear war, an enemy may destroy a link in the US chain of communications, was the precursor to the technology revolution!

Today, the internet is an international forum for exchange of information and ideas between millions of people world wide, a rapidly growing information super-highway. In contrast, two decades prior to this, it was known mainly to those involved in the military or academia.

The Department of Defense (USA) began an initiative which later contributed towards the physical network of the first four computers (nodes) right across America in 1969. This project was named Advanced Research Project Agency ARPA. ARPA net was a response to the launching of the first artificial satellite Sputnik by the USSR in 1957.

The World Wide Web and the use of hypertext, dates back to the early 1990s, when a physicist decided to invent a method of sharing information with his colleagues. Dr. Tim Berners-Lee was working at the laboratory for particle physics in Geneva, Switzerland, when he thought his work would be easier if he and his colleagues could simply link to each other's computers. The links were believed to provide enormous efficiencies as they would freely facilitate sharing of information and ideas. The name of the World Wide Web originated because the hyperlinks to and from these computers were imagined to be like a spider's web.

The CERN site is the name of the particle physics laboratory where Dr. Berners-Lee worked. It is regarded the birthplace of the World Wide Web.

Marc Andreessen was a university student in 1993 at the University of Illinois. It was during this time that he led a team that invented the first graphical user interface browser. Prior to this date, most people had to rely on text based browsers such as Lynx. Access to the World Wide Web during this period, was mostly for those working in academia and of course the military.

The software, designed to assist people in accessing the Internet was named Mosaic. As well as making it possible to explore the Internet in a simple and intuitive manner, it provided an extremely user friendly medium for home page design using multimedia.

The Client/Server Architecture of the World Wide Web

What is the World Wide Web?

- The World Wide Web (WWW) is most often called **the Web**.
- The Web is a network of computers **all over the world**.
- All the computers in the Web can **communicate with each other**.
- All the computers use a **communication standard called HTTP**.

How does the WWW work?

- Web information is stored in documents called **Web pages**.
- Web pages are files stored on computers called **Web servers**.
- Computers reading the Web pages are called **Web clients**.
- Web clients view the pages with a program called a **Web browser**.
- Popular browsers are **Internet Explorer and Netscape Navigator**.

How does the browser fetch the pages?

- A browser fetches a Web page from a server **by a request**.
- A request is a standard HTTP request containing **a page address**.
- A page address looks like this: **http://www.someone.com/page.htm.**

How does the browser display the pages?

- All Web pages contain **instructions for display**
- The browser displays the page by **reading these instructions**.
- The most common display instructions are called **HTML tags**.
- HTML tags look like this **<p>This is a Paragraph</p>.**

Who is making the Web standards?

- The Web standards are **not made up** by Netscape or Microsoft.
- The rule-making body of the Web is the **W3C**.
- W3C stands for the **World Wide Web Consortium**.
- W3C puts together specifications for **Web standards**.
- The most essential Web standards are **HTML, CSS and XML**.
- The latest HTML standard is **XHTML 1.0**.

The Domain Name System (DNS)

What is DNS?

- DNS is the method by which Internet addresses in mnemonic form such as sunc.scit.wlv.ac.uk. are converted into the equivalent numeric IP address such as 134.220.4.1.
- To the user and application process this translation is a service provided either by the local host or from a remote host via the Internet.
- The DNS server (or resolver) may communicate with other Internet DNS servers if it cannot translate the address itself.

What does DNS name structure look like?

- DNS names are constructed hierarchically.
- The highest level of the hierarchy being the last component or label of the DNS address.

- Labels can be up to 63 characters long and are case insensitive.
- A maximum length of 255 characters is allowed.
- Labels must start with a letter and can only consist of letters, digits and hyphens.
- DNS addresses can be relative or fully qualified.
- The final most significant label of a fully qualified name can fall into one of three classes

1 ***arpa***

This is a special facility used for reverse translation

2 ***Three letter codes***

Indicate the type of organization hosting the computer.

code	**meaning**
com	Commercial. Now international.
edu	Educational.
gov	Government.
Int	International Organization.
mil	Military.
net	Network related.
org	Miscellaneous Organization.

3 ***Two letter codes***

Indicate the country of origin

URL, URI, URN

What is URL?

- A Uniform Resource Locator is the exact address or location of the files or web pages be retrieved on the internet.
- URL is a syntax that mandated the following format:
- <protocol>:// <host> [:<port>] [<path>] [?<query>]
- Example:- http://mail5srv1.tech.aau.edu.et/en/mail.html?sid=test&lang=en

What is the difference among URL, URI and URN?

- URI(Uniform Resource Identifier) is a formatted string that univocally and uniquely identifies a resource.
- There are two types of URIs: URLs and Uniform Resource Names (URNs)
- A URL takes you straight to the resource and the data.
- A URN is only a unique name you can use to identify any resource you want.
- URNs are related to namespaces.
- Both URLs and URNs can uniquely identify resources over the Web.
- Use URLs when you need to know or specify location information.
- Use URNs if the resource is location-independent.
- address-specific (URL) and name-based (URN)

HTTP Header

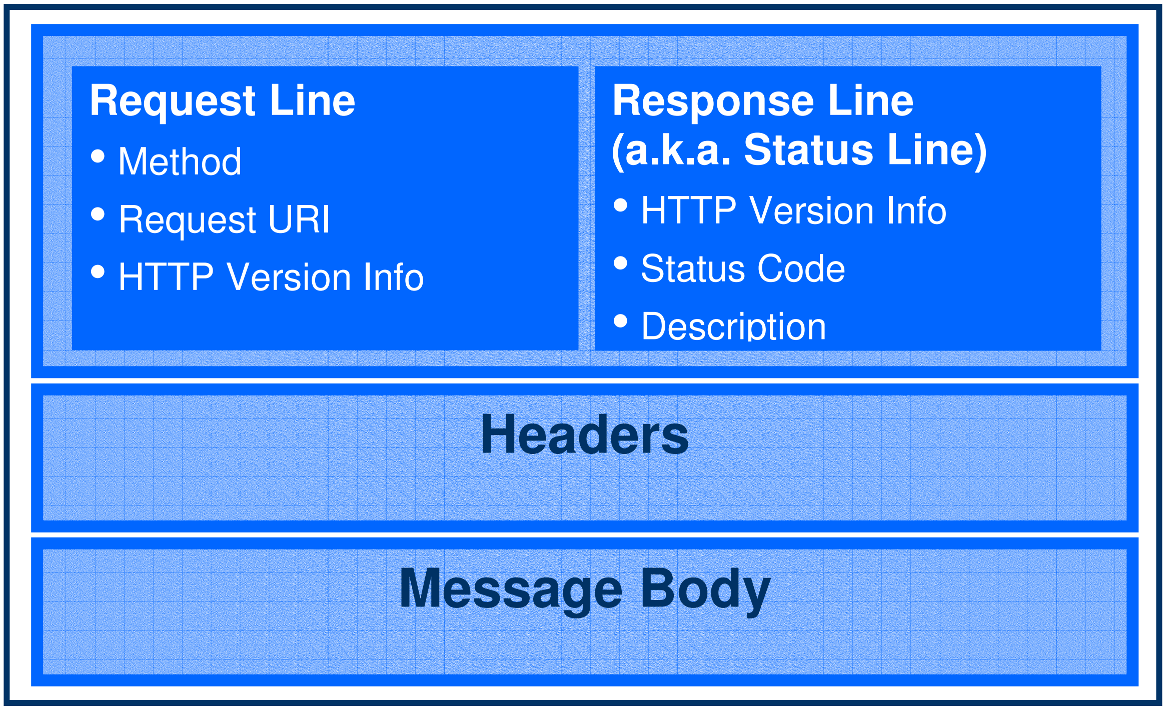
HTTP Message

- HTTP messages consist of requests from client to server and responses from server to client.
- Request and Response messages use the generic message format for transferring entities
- Both types of message consist of a start-line, one or more header fields (also known as "headers"), an empty line (i.e., a line with nothing preceding the CRLF) indicating the end of the header fields, and an optional message-body.
- Message example

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Tue, 27 Mar 2001 10:35:30 GMT
Content-Type: text/html
Accept-Ranges: bytes
Last-Modified: Tue, 27 Mar 2001 10:34:52 GMT
ETag: "8c70de8ea9b6c01:d0d“
Content-Length: 488
<html>
 <head>
 <title> Test Page For HTTP </title>
 </head>
 <body>
  <p>
   <img src="IN00483_.gif" width="36"
```

```
height="35">
   Test Page!
  </p>
 </body>
</html>
```

- Message dissected by diagram



Message body

- Used to carry an entity body. Entity differs from message body when "encoding" exist. Example: the entity body is compressed
- It is an Octet – an 8-bit sequence of data
- May be divided into pieces and sent in chunk. When size cannot be predetermined and reassembled during reception of the messages
- Messages do not have to have a message body. Some messages cannot have a message body
- Example of message body
  - A Web page! The text to render as the page is the body
  - Login information or other form data
  - Shopping information – item you wish to buy

HTTP Header

- HTTP header fields, which include general-header, request-header, response-header, and entity-header fields, follow the same generic format .
- Each header field consists of a name followed by a colon (":") and the field value.
- The order in which header fields with differing field names are received is not significant. However, it is "good practice" to send general-header fields first, followed by request-header or response- header fields, and ending with the entity-header fields.

General HTTP Header

- There are a few header fields which have general applicability for both request and response messages, but which do not apply to the entity being transferred.
- These header fields apply only to the message being transmitted.

```
general-header = Cache-Control
                 | Connection
                 | Date
                 | Pragma
                 | Transfer-Encoding
                 | Upgrade
                 | Via
```

Entity Headers

- It gives meta-information (meta means information about) about the entity-body being transferred Or, if no entity-body exists, about the resource of the request
- It apply only if a message body exists
- Examples of entity headers
  - Allow: List of methods supported by the resource
  - Content-Encoding: Indicates types of content codings applied
  - Content-Language: Language of the intended audience
  - Content-Length: Size of entity-body
  - Expires: Date/time after which response is considered stale
  - etc

Requests Headers

- Additional information about the request
- May include information about the client (or sender) itself
- Examples of request headers
- Accept: Specifies media types acceptable for response
- Accept-Charset: Indicates acceptable character sets
- Accept-Encoding: Similar to Accept; specific to encodings
- Accept-Language: Limits response to preferred languages
- Host: Specifies the host & (optional) port of the resource
- etc

Responses Headers

- More information than available from just the status line
- May be information about the server or the resource
- Examples of response headers
- Age: Estimate of time since response was generated
- ETag: Current value of the entity tag
- Location: Used to redirect to a different location (URI)
- Proxy-Authenticate: Proxy authentication challenge
- Retry-After: Expected time that a service will be unavailable
- Server: Information about the server software used
- WWW-Authenticate: Authentication challenge

HTTP Request/Response, the Stateless nature of HTTP

The stateless nature of HTTP

- A fundamental characteristic of the Web is the stateless interaction between browsers and web servers.
- HTTP is a stateless protocol.
- Each HTTP request a browser sends to a web server is independent of any other request.
- The stateless nature of HTTP allows users to browse the Web by following hypertext links and visiting pages in any order.
- HTTP also allows applications to distribute or even replicate content across multiple servers to balance the load generated by a high number of requests.

HTTP Request

- Three Parts of a Request Line are Request Method, Request URI and HTTP version information

Request Methods

- GET (or retrieve) information from the resource server
- POST “the information” back to the resource server
- DELETE “the information” from the resource server
- PUT “the information” at the resource location
- HEAD: Like GET but only returns meta-information
- OPTIONS: Gets the communication available

HTTP Version

- Used by sender to notify receiver of its abilities
- Version information is included in first line of message
- Uses <major> . <minor> numeric notation
- Examples: 1.0 or 1.1
- <major> number indicates the message format
- <minor> number indicates extensions to major format
- HTTP-Version = "HTTP" "/" 1*DIGIT "." 1*DIGIT
- Examples: HTTP/1.0 or HTTP/1.1

Response Line Dissected

- HTTP response include HTTP Version Information, Status Code and Status Description
- The response from the server contains 3 digit status code and a text phrase which describe about the status.

Status Codes – 5 Categories

- 1xx: Informational request received and processing is continuing
- 2xx: Success The action was successfully received, understood, & accepted
- 3xx: Redirection Further action must be taken to complete the request
- 4xx: Client Error A client error occurred
- 5xx: Server Error A server error occurred

- Example of Status Codes
    - 100: Continue → Tells the client to continue with a request
    - 200: OK → The request has succeeded
    - 202: Accepted → The request has been accepted but not processed
    - 302: Found → Resource requested found but temporarily moved
    - 400: Bad Request → The request could not be understood
    - 401: Unauthorized → The request requires proper authorization
    - 403: Forbidden → The client may not access the resource
    - 500: Internal Server Error → The server encountered an unexpected error. The request was not fulfilled
    - 505: HTTP Version Not Supported → The server does not or will not support the HTTP version

Web browser configuration

- A browser is an application program that provides a way to look at and interact with all the information on the World Wide Web.
- The word "browser" seems to have originated prior to the Web as a generic term for user interfaces that let you browse (navigate through and read) text files online.
- the first Web browser with a graphical user interface is Mosaic (in 1993)
- Technically, a Web browser is a client program that uses the Hypertext Transfer Protocol (HTTP) to make requests of Web servers throughout the Internet on behalf of the browser user.
- the first widely-used browser, Netscape Navigator. Microsoft followed with its Microsoft Internet Explorer.
- Today, these two browsers are the only two browsers that the vast majority of Internet users are aware of.
- Lynx is a text-only browser for UNIX shell and VMS users.
- Another recently offered and well-regarded browser is Opera.
- Use of browser is
    - Font mapping, e.g. Unicode
    - Compression, decompression
    - Handles multimedia, manages plug-ins

    - Interprets scripts
    - Executes Java applets
    - Maintains cache, history
    - Manipulates cookies

HTTP Authentication

- The HTTP protocol (RFC 2616) defines a simple framework for access authentication schemes.
- The assumption is that a certain group of pages (usually referred to as a protected realm or just a realm) should only be accessible to certain people who are able to provide credentials if challenged by the server.
- If an HTTP client, e.g. a web browser, requests a page that is part of a protected realm, the server responds with a 401 Unauthorized status code and includes a WWW-Authenticate header field in his response. This header field must contain at least one authentication challenge applicable to the requested page.
- Next, the client makes another request, this time including an Authentication header field which contains the client's credentials applicable to the server's authentication challenge.
- If the server accepts the credentials, it returns the requested page. Otherwise, it returns another 401 unauthorized response to inform the client the authentication has failed.

Basic Access Authentication

- The basic authentication scheme assumes that your (the client's) credentials consist of a username and a password where the latter is a secret known only to you and the server.
- The server's 401 response contains an authentication challenge consisting of the token "Basic" and a name-value pair specifying the name of the protected realm. Example:
    - WWW-Authenticate: Basic realm="Control Panel"
- Upon receipt of the server's 401 response, your web browser prompts you for the username and password associated with that realm. The Authentication header of your browser's follow-up request again contains the token "Basic" and the base64-encoded concatenation of the username, a colon, and the password.
    - Authorization: Basic QWRtaW46Zm9vYmFy