INTERNET and World Wide Web

Internet - the massive global network of network (www.internetsociety.org)

- Global Information System

\*Internet and World Wide Web are logically linked together by a globally unique spaced address

INTERNETWORK:"

Internet - Sandra Bullock: 'net'

Intra: within the domain

Inter: with different domains

Inter: different networks + Network: nodes = Internet/ Internetwork

ISP- Internet Service Provider (PLDT, Bayantel)

Internet - global connection of network

IP Address- used to uniquely describe a node

IPv4 - 32 bit (4Billion connections)

IPv6 - 64 bit

- is able to support communication using the Transmission Control Protocol/ Internet Protocol (TCP/IP)

- it provides, uses or makes accessible, either public or private, high level services layered on the communication and related infrastructure described herein.

**HISTORY DEVELOPMENT OF THE WEB**

**1969** -date year of Internet (48 years old)

-before, they are all in packet-switched store and forward: here, do not need a direct connection.

-originally created for military bases to communicate from one country to another

**1972** – Electronic Mail (e-mail) was introduced: first killer of the internet

**1989** –The second killer of the internet was created which was the “World Wide Web’

**World Wide Web** – created by Tim Berners – Lee (CERN)

**www (vocabulary)**: HTTP, HTML, URL, Web Server, Web Client

**WAIS** - Wide Area Information Servers

**WWW**- information system on the internet that lets documents to be connected to other documents by hypertext links, which enables the user to search by moving one document to another, using the Hypertext Transfer Protocol (HTTP)

**Gopher (protocol)** : pronounced as ‘gofer’

* It has hierarchy of information

**Search Engines before**: Archie, Verson, Juglen

Sir Time Berners-Lee focuses mainly on the semantics.

**Usenet** - Worldwide distributed discussion system available on computers

* Something like you can comment

**WWW WAS NOT INTENDED FOR PUBLIC CONSUMPTION**

* It was for scientists
* Full of text before
* And passive only

**HTML-** Everything is made out of HTML (WebServer).

**URL -** is used to identify which HTML file.

**FTP** - File Transfer Protocol

**Web Communication -** HTTP

**Sir Tim Berners Lee is the FATHER OF MODERN WEB**

HTTP was created by Berners-Lee and then jointly developed by the IETF

SPDY: protocol by Google (chromium browser)

HTTP runs on top of TCP/IP using TCP port 80 by default, or TCP port 443 for HTTP

HTTP- based on a client-server architecture (clients a.k.a. User Agents)

Socket = IP Address + Port#

**HTTPS -** secured but will put anything to terminal (Encryption Layer)

**Origin Server**

**Gateway**

**Proxy Server**

**Tunnel**

**CLIENT**

**Tunnel** – Blind traffic where packets can travel

**USER AGENT (UA) -** Web Browser

-Web Crawlers/ Spiders: used by google to look for keywords from websites

**SERVERS –** Origin Servers,Proxy servers, gateways, tunnels

**CLIENTS –** initiates the request

**HTTP2 –** “push”protocol (server will automatically sends you the page if there’s new)

HTTP is a stateless communications protocol – servers do not keep information about clients in browser requests.

HTTP PROVIDES SUPPORT FOR OTHER FUNCTIONALIRIES, SUCH AS: CACHE CONTROL, CONTENT MEDIA TYPE (MIME), SPECIFICATION …. ETC.

**Cache Control**

* Low Reference Policy : faster execution because all data will be saved first

**Content Media Type (MIME)**

* MIME : Multipurpose Mail extension)

**Language & Character Set Specifications**

* ASCII : American Standard Code in Information Interchange **(8 bit code)**
* 1 letter – 1 byte
* UNICODE – more than 1 byte

**Content/ Transfer Coding**

* Way of transferring data and strings
* 80 – 90% of data transferring 1000 byte – 100 byte (compressed)

**Content Negotiation**

* All first four above must have a client-server agreement first (negotiation)
* Request before sending

**Client-Server Protocol Negotiaion**

* HTTPS Version 1.0, 1.1, 2
* They will both ( client and server) whether they will upgrade or downgrade for the version of the protocol

**Persistent Connection**

* **Stages:**

Request: client will open a socket connection and send the request to the server

Response: server will respond and therefore a connection will be established

Disconnection: The session will terminate

* 1.0 : Non Persistent
* 1.1: Connection can be reuse in multiples connections

**Request Pipelining**

* Multiple Requests
* Pipelining/ Persistent/ Pushing
* MOTIVATION: To make everything/ anything faster

**Authentication / Authorization**

* HTTP not in FB, Google, and etc.
* Authentication to view a certain page / website

**HTTP RESOURCE ADDRESSING**

HTTP resources are identifies using URLs (RFC 3986), or, node specifically, HTTP URLs

**Scheme (http or https)**

* Authority
* URI : URN and URL

URN: Uniform Resource Name – Name without location

URL: Uniform Resource Locator- You can see the location where you can find it

**Authority**

* User information/ authentication/ credentials (deprecated)
* Host: (name of the machine)

Domain name: (resolved to an IP address using DNS of the server where the resource resides.

* Port Number

80 – Http

443 and 440 – Https

Eg:

http://usr:pwd@server.org:81/info/profile.php

Scheme : http

User Info : usr:pwd

Host : server.org

Port No : 81

Path : /info/profile.php

**Path to the Resource**

* (resolved relative to true document root on the server)
* May be a static (served as-is)
* May be dynamic (scripts/programs that run on the server side)

**Query**

* Usually starts with ‘?’
* Typically provided as key=value
* Pairs with ‘&’ to separates parameters

e.g. products.php?id+1234&color=red&size=xL

**Fragment Identifiers**

* Usually starts with ‘#’

**REQUEST HEADERS**

**Get**

* to request

**Post**

* to submit

**Put**

* store the closest entity
* put and delete is not allowed at the same time (it depends upon the configuration of the server)
* WebDAV: propFind, proppatch, mkCol, copy, move, lock, unlock

**Delete**

**Option**

* Ask the serve “What can I do?”
* List of services

**Trace**

* Used for diagnostics of the request/ response chain
* Request a loop-back of the request message
* Testing your response content length
* CHUNKED: requrst the establishment of a tunnel- blind relay
* HTTPS TRAFFIC: if you can configure and decrypt you will not see it but Fiddler can see it (man-in the middle)

**Type of Methods**

**Safe Methods**

* Methods that do not cause any modification
* E.g. GET, HEAD, OPTIONS, TRACE

**Idempotent Methods**

* we can use it in several times but there would be no difference in using it once, twice, thrice, etc.
* e.g. GET, HEAD, OPTIONS, TRACE, PUT,DELETE

**HTTP MESSAGE HEADERS**

**General Header Fields**

**Cache Control**

* to control your cache configuration
* local storage

**Connection**

* to indicate that the connection is non-persistent
* e.g. connection: keep-alive**,** connection: closed

**Pragma**

* generic message we used to HTTP 1.0
* General Directive (older)

**Transfer Encoding**

* Gzip, compression related

**Trailer**

**-**instead of having header, it will be a trailer located at the end

**Upgrade**

* Transferring to another version
* Protocol upgrade
* Protocol negotiation
* Switching protocols

**Via**

* Forward your message to a proxy
* STAMP – when a certain website comes across, the server will stamp a mark on it then pass it eventually to the requester or the client.
* Not that necessary

**Warning**

* Identifier/ warning that something is wrong

**Accept**

* \*/\* (anything will do)
* Anyt type from html/text, css/text….etc
* Application/xml
* q=0.9 (the lower value the lower the priority)

**Accept-Language**

* change of setting regarding the language
* bs, English, en-us, fil… etc

**Accept-Encoding**

* compression
* e.g. gzip, deflate

**Authorization**

* something with a protected content – www (authenticated)
* proxy authentication

**Expect**

* client: Content request
* what the client will be expecting from the server
* e.g. 417 client failure, 100 Continue

**From**

* the e-mail address of the user where you will be contacted
* typically use when you’re creating an user agent

**Post**

* Html 1.1 idea: to serve the virtual host
* You can host multiple site using one physical address

**If-Match**

**If-Modified-Since**

**If-unmodified since**

**If-None-Match-Entity**

* Node: location in actual file system

**If-Ranges & Accept Ranges**

**Max-Forwards**

* Will simply tell you the number of the maximum number of proxy

**Referrer**

**Trailer**

**User Agent**

* Provide an information about itself
* Applicable also in mobile

**Age**

* Will tell you the age of the resource

**ETag**

**Accept-Ranges**

**Location**

* Redirection
* 301 Redirect Command

**Proxy-Authenticate**

**Retry-After**

* Conjunction with 503: service unavailable

**Server**

* Apache & wamp
* Not required but it depends upon the server

**Vary**

* Whether it is accepted, etc.

**ENTITY HEADER FIELDS**

Entity: represents the body

**Allow:** what operations can we do in certain

**Content Encoding**: chunk, zip..etc.

What type of encoding?

**Content Language:** important in case of accessibility

**Content-Length:** how long it is in bytes?

**STATUS CODES**

## 1xx Informational

100 Continue

101 Switching Protocols

102 Processing (WebDAV)

## 2xx Success

 200 OK

201 Created

202 Accepted

203 Non-Authoritative Information

 204 No Content

205 Reset Content

206 Partial Content

207 Multi-Status (WebDAV)

208 Already Reported (WebDAV)

226 IM Used

## 3xx Redirection

300 Multiple Choices

301 Moved Permanently

302 Found

303 See Other

 304 Not Modified

305 Use Proxy

306 (Unused)

307 Temporary Redirect

308 Permanent Redirect (experiemental)

## 4xx Client Error

 400 Bad Request

 401 Unauthorized

402 Payment Required

 403 Forbidden

 404 Not Found

405 Method Not Allowed

406 Not Acceptable

407 Proxy Authentication Required

408 Request Timeout

 409 Conflict

410 Gone

411 Length Required

412 Precondition Failed

413 Request Entity Too Large

414 Request-URI Too Long

415 Unsupported Media Type

416 Requested Range Not Satisfiable

417 Expectation Failed

418 I'm a teapot (RFC 2324)

420 Enhance Your Calm (Twitter)

422 Unprocessable Entity (WebDAV)

423 Locked (WebDAV)

424 Failed Dependency (WebDAV)

425 Reserved for WebDAV

426 Upgrade Required

428 Precondition Required

429 Too Many Requests

431 Request Header Fields Too Large

444 No Response (Nginx)

449 Retry With (Microsoft)

450 Blocked by Windows Parental Controls (Microsoft)

451 Unavailable For Legal Reasons

499 Client Closed Request (Nginx)

## 5xx Server Error

 500 Internal Server Error

501 Not Implemented

502 Bad Gateway

503 Service Unavailable

504 Gateway Timeout

505 HTTP Version Not Supported

506 Variant Also Negotiates (Experimental)

507 Insufficient Storage (WebDAV)

508 Loop Detected (WebDAV)

509 Bandwidth Limit Exceeded (Apache)

510 Not Extended

511 Network Authentication Required

598 Network read timeout error

599 Network connect timeout error