HTTP

→ URL → Uniform Resource Locator

→ eg: http://food.com/recipes/broccoli-pasta

- http → URL scheme
- food.com → host name
- /recipes/broccoli-pasta → URL path

→ eg 2:

http://food.com:80/recipes/squash

- http → URL scheme
- food.com → Host
- 80 → Port no.
- /recipes/squash → URL path

↳ this port no. is used by host to listen to HTTP requests.

↳ default port no for http is 80.

→ eg 3:

http://bing.com/search?q=jdroticals

↳ everything after question mark is query or query string

↳ it is optional.

↳ most of the queries use "name: value" pairs.

→ Eg 4:

http://wikipedia.org/wiki/Tabletop#Description

Fragment.

↳ fragment is not processed by server. The fragment is used on the client side to identify and navigate to a particular section on a webpage or of a resource.

↳ In the above URL, on hitting 'enter', the client will be navigated to the Description section within the webpage.

⇒ you might have also seen that, when you search something in google, once you hit the enter, the URL is added with some special pair of sign. such as %20. %20 is used to represent space "___" in the URL. It is the US ASCII characters used for URL-encoding.
↳ and hexadecimal value.

⇒ MIME is used to label the content. It is to understand the type of the content, i.e, whether it is an image, a text, a video, audio etc.

→ MIME types are used to specify the representation of the resources.

→ HTTP → generic protocol for moving information around in an interoperable way

⇒ URL canonicalization :

⇒ HTTP request methods :

GET → to fetch or to get or to retrieve a resource.
POST → to update a resource.
PUT → to store a resource.
DELETE → to remove a resource.
HEAD → to retrieve the headers for a resource.

⇒ HTTP request message format :

start line

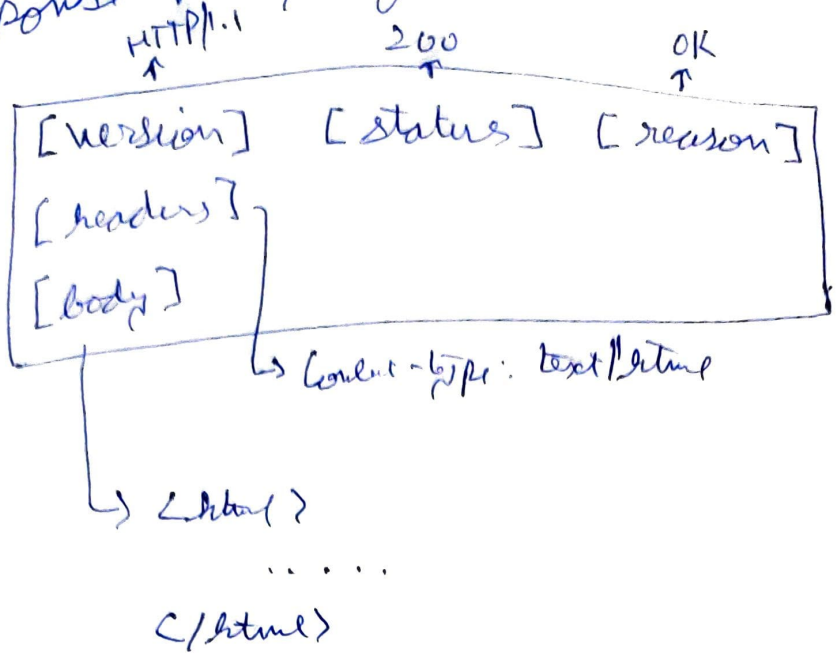
```
[method name] [URL] [version]
[headers]
[body]
```

eg: GET http://server.com/article HTTP/1.1

Headers
- Host : odetocode.com
- Accept-Language : fr-FR
- Date : ....

→ in GET method, there would be a body.

→ Response message format :

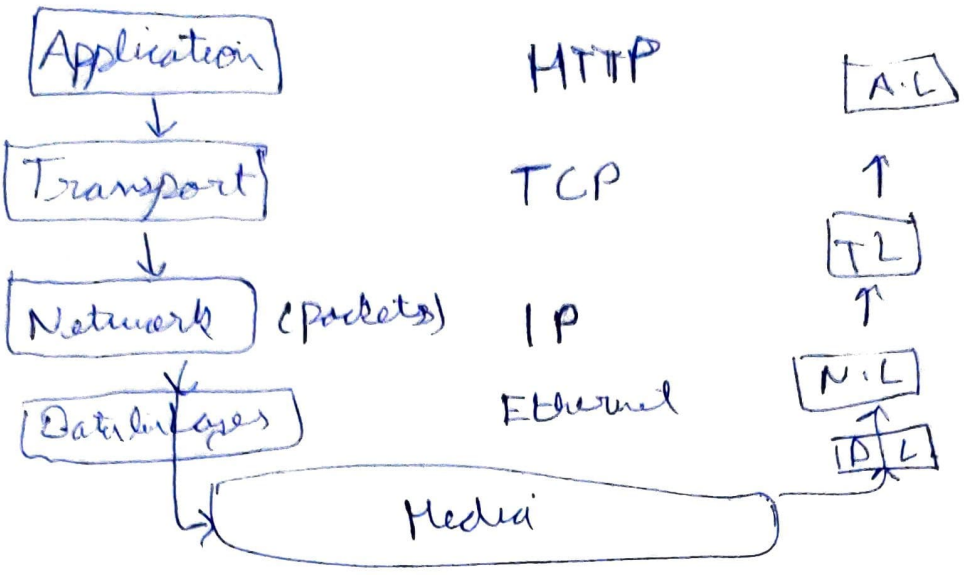


→ Status code categories :

100 - 199	Information
200 - 299	Successfull
300 - 309	Redirection (can be temporary or permanent)
400 - 499	Client error (eg : 404)
500 - 599	Server error (something is wrong with the server)

→ HTTP is an application layer protocol because it allows two applications to communicate with each other over the Network. In general, one of the application is web server and other is a web server, like Apache.

->



-> Most of the HTTP traffic travels over TCP.

-> Once, a user types a URL, the browser extracts the hostname and a port no, if provided in the URL. After this it opens a socket where the browser can write the HTTP messages. These messages are then taken by the TCP and now it is TCP's responsibility to make sure that the data reaches to the web server without getting ~~or~~ lost or duplicated while in transit. TCP, ~~also~~ resends any information or data (↳ automatically) that is lost in transit. Hence, the application need not be worry about the data. So, TCP is also called a

reliable protocol.

→ Additionally, TCP provides error detection as well as flow control.

→ Flow control means that TCP will ensure that the sender does not send the data too fast that the receiver is unable to process the data.

→ After transport layer, we have Network layer and it uses IP (Internet Protocol).

→ IP is responsible for taking the piece of information and moving them through all the switches and routers and N.W. devices so that the data reach to the destination.

→ IP does not guarantee the delivery of data, that is TCP's job.

→ IP takes the data from the TL and breaks it into packets.

→ IP uses the IP address of the computer in order to work.

→ Now, eventually, these packets need to reach to its destination. To do so, the packets need to travel through the wired wires or wireless in some cases. ~~Here~~, This all happens through the data link layer. Here the ~~packe~~ packets become frames and ~~Ethe~~ Ethernet protocol is used.

→ Cookies: (HTTP state management mechanism)

→ What is meaning of "HTTP is stateless protocol"?

→ It means that ~~HTTP~~ each ~~req~~ request response transaction is independent of previous or future transactions.

→ size limitation of cookie : 4KB.

→ cookie is used by the website or the host to keep ~~track~~ track of its user.

→ When we request for a particular resource, for example,

```
GET /search?q=login HTTP/1.1
Host : searchengine.com
...
```

~~The browser~~ now, the ~~server~~ searchengine.com wants to track the user, so, ~~it sent~~ in the HTTP response for the above request, it is going to send a Set cookie header.

```
HTTP/1.1 200 OK
Set-Cookie: fname=SantaBlue=Allen; ->(name : value pair)
            domain = .searchengine.com;
            path = /
....
```

→ Many websites also put a unique id in the cookie which is called GUID.

GUID = 00448671.....;

↳ instead of name: value pairs

→ GUID is used because there is a size limit in cookie, and secondly, a browser can't trust something that is stored in user's system unless it is cryptographic. It is really easy to store id.

→ Now, whenever the user requests for searchengine.com, the request message will contain a cookie header and it will pass the GUID with every subsequent request. When this id arrives at the server, the server can use this id and lookup for the associated data.

→ But, in order for all this to take place, the user must enable browsers to store cookies.

→ Cookies are browser specific, meaning that a cookie stored by Chrome, is not accessible by Firefox or IE.

→ Anatomy of IP address:

IPV4 address 192.168.114.102

Subnet mask 255.255.255.0

Default gateway 192.168.114.1

DNS servers 75.75.75.75
75.75.76.76

→ How to obtain IP address/configuration:

→ A system gets IP address from a DHCP server:

Dynamic Host Configuration ~~System~~ Protocol

→ ~~So~~ machine sends out broadcast message, ~~Says~~ requesting for IP address to DHCP server.

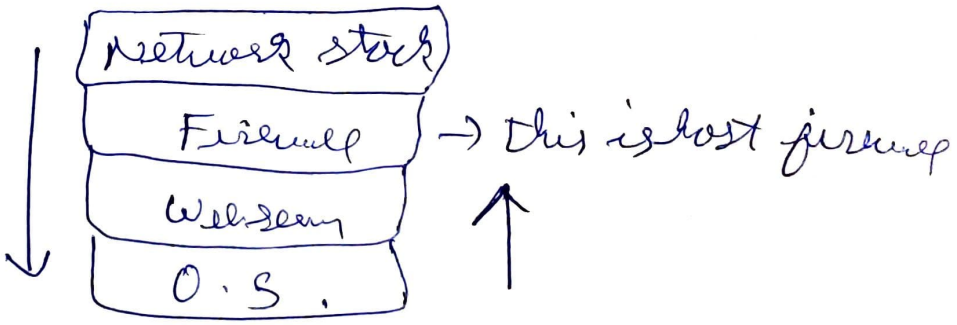
→ The DHCP server will receive the request and will provide an IP address, a ~~Subnet~~ subnet mask and a default gateway and DNS servers which the client machine can use.

→ The client configures itself with the provided config and sends ack to the DHCP server.

address. Now, once the client knows the IP address the website connect to, it will go ~~across~~ across the internet, (pass through
→ means request.

various N.W devices such as switch and routers and will eventually reach the web server. Before reaching to the web server it passes through a router, then a firewall and finally through a switch.

→ Once, the request reaches to the web server, it has to pass through various layers such as;



→ Now, once the ~~process is~~ request is processed, this process is removed. The data is sent back to the client and the web page renders on the client machine.

→ Now, in order to make the cookie, safe from being misused, we use or the session id

HTTPS, Secure HTTP. It also makes use of another method which is set the path property in headers as HttpOnly. Using the HttpOnly, this prevents any sort of malicious code or javascript code to read and write the session id.

⇒ Authentication protocols:

i) Basic authentication protocol:

→ uses base 64 authentication.
→ not safe over http:
→ Safe over secure http (https).

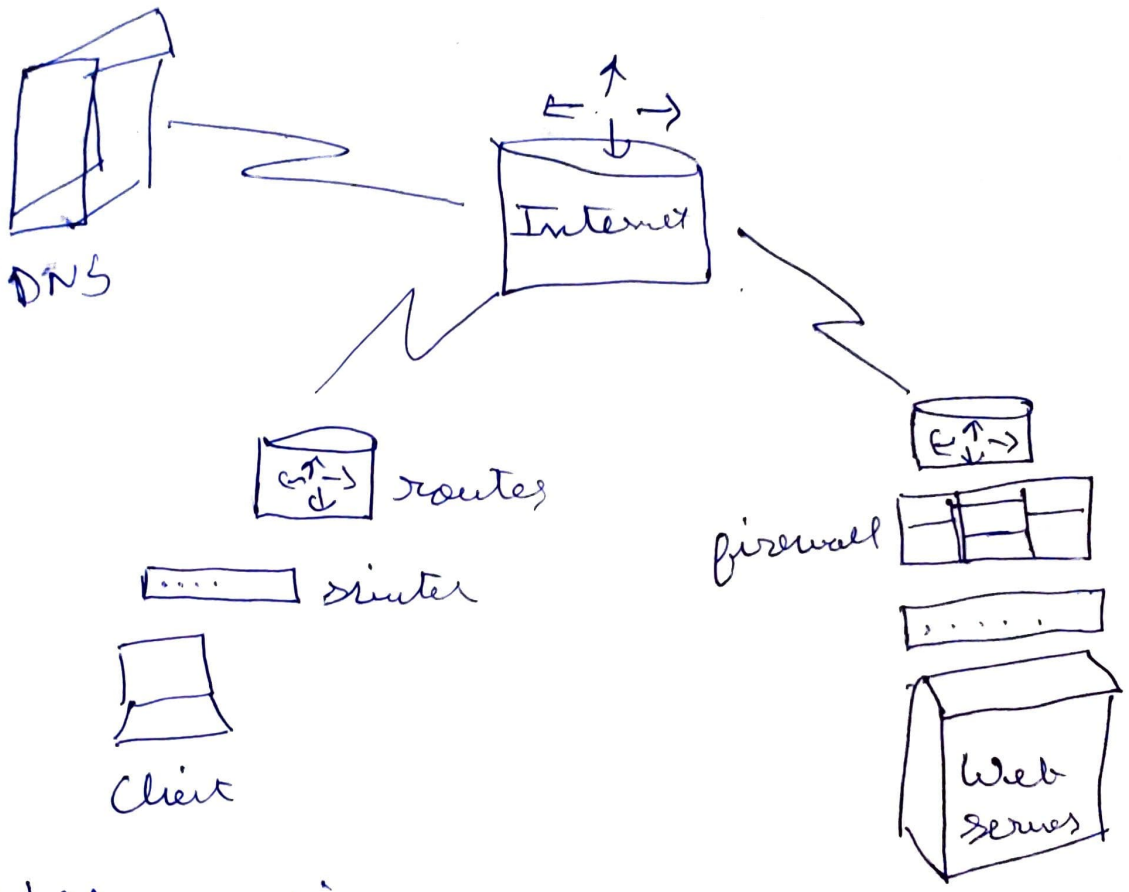
ii) Digest authentication:

→ used.

⇒ default port for HTTPS is 443
↳ for HTTP is 80.

→ TCP:

→ When we enter URL and hit enter, here's how that request is served.



→ When we hit enter, the client needs to know the IP address of the requested URL. For example if the requested resource is for pluralsight.com, then we need to know the IP address for this domain.

→ Hence, to do this the request goes to the DNS server and asks for it. The DNS server then replies back to the client by providing the IP

=> IPV4:

-> address space = 2^{32}

eg: 192.168.1.1

-> IPV6:

address space = 2^{128}

=> i.e equal to 100 IP's for every single atom on earth.