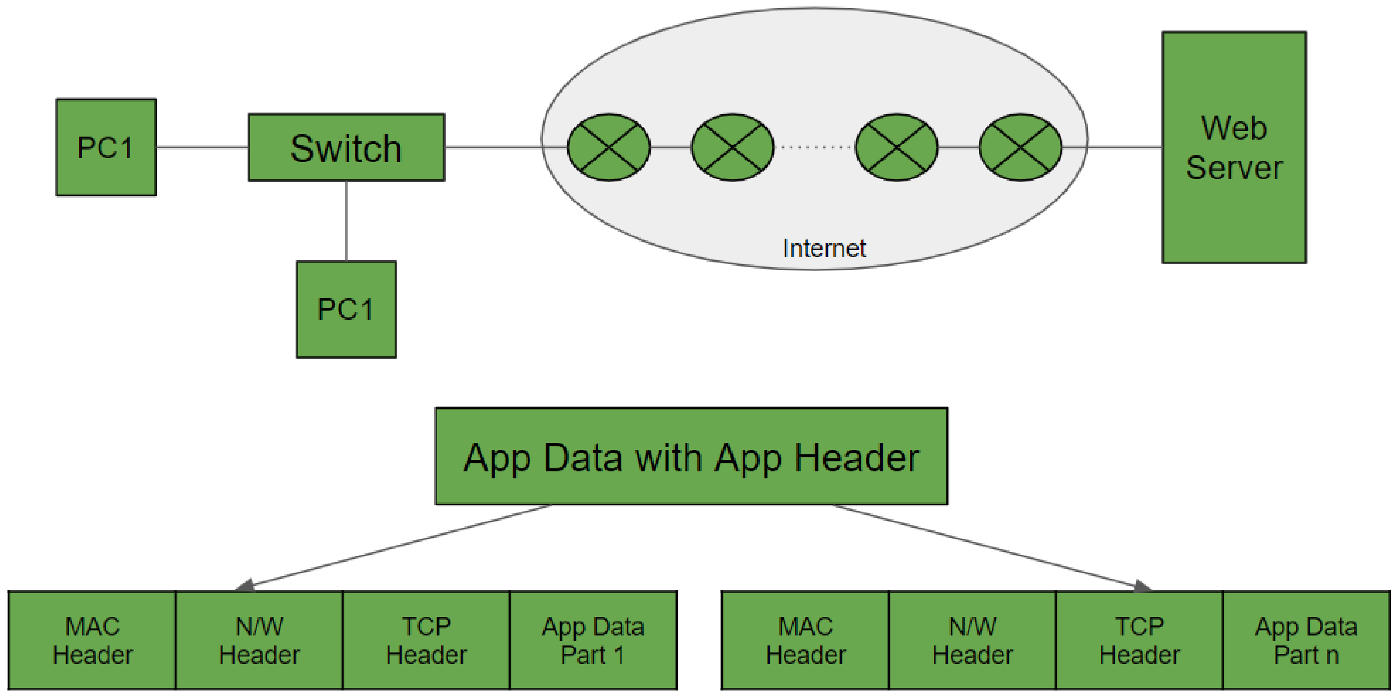
We have combined Classroom and Theory tab and created a new Learn tab for easy access. You can access Classroom and Theory from the left panel.

## Application Layer

The application layer is the topmost layer of the OSI Model. It is the layer in which user applications run. Various protocols run at this layer serving different requirements. Let's understand the working using the below diagram:



When someone browses the internet, then the browser generates Application Data with specific Header files. Following this, the transport layer breaks the Application data into various parts. To this TCP header is added to each part of the Application Data. Next comes the network layer, which adds the destination address in the network header. Then the link is made between the computer and the ISP routers. All the traffic are being sent to the routers. To send the data over the data link layer to any other routers, the computer uses MAC address where the routers MAC address is used to set the link. This MAC address is known using the ARP or Address Resolution Protocol. The switch reads the MAC header of the destination router. The routers laying on the Internet implements three layers namely network layer, DLL, and physical layer. Now finally when the data reached the webserver then using the TCP header, the webserver combines all the data.

We shall go over in brief over each of the protocols:

- **HTTP**: Stands for **H**yper **T**ext **T**ransfer **P**rotocol. It is a request-response protocol that is used to receive web-pages on a client-server architecture. The client requests for a resource (HTML page, javascript file, images or any other file) to the server. The server returns a response accordingly. It uses TCP as the underlying Transport Layer protocol. HTTP supports the following methods (modes) of requests:
    1. *GET* - Retrieve information from the server (GET Requests is intended only for data fetching and should not have any side-effect).
    2. *HEAD* - Retrieve only header (meta-information) and no response-body.
    3. *POST* - Post/Send data back to the server. e.g. User-data, form-data.
    4. *DELETE* - Delete the specified resource.
    5. *OPTIONS* - Returns the list of HTTP methods supported by the server.

  Port no. for HTTP: 80 (8080 occasionally)
- **HTTPS**: It is a secured version of HTTP made possible by encrypting the data transferred using TLS (Transport-Layer-Security). Earlier, its predecessor ~ SSL was used. The use of HTTPs over HTTP increases security by preventing eavesdropping, tampering and man-in-the-middle attacks. Port used in HTTPs is the same as that of HTTP.
- **TELNET**: Stands for **TEL**ecommunications **NET**work. It is used in terminal emulation, which one can use to access a remote system. It is used for file-access and for the initial setup of switches. One may draw it's similarities to SSH (Secure-Shell), but as the name suggests SSH is secure as it uses encryption in addition to normal terminal emulation. Telnet is thus no longer used thanks to SSH.
  Port no. for Telnet: 23
  Port no. for SSH: 22
- **FTP**: Stands for **F**ile **T**ransfer **P**rotocol. It provides reliable and efficient file-transfer between two remote machines.
  Port no. for FTP Data: 20
  Port no. for FTP Control: 21s

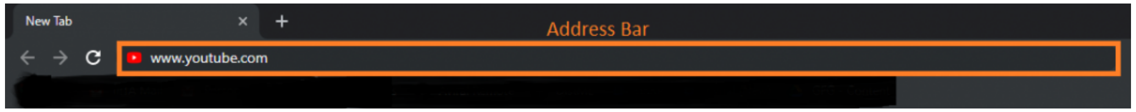
- **SMTP**: Stands for **S**imple **M**ail **T**ransfer **P**rotocol. Uses TCP under the hood. Using a process called “store and forward,” SMTP moves your email on and across networks. It works closely with something called the Mail Transfer Agent (MTA) to send your communication to the right computer and email inbox.
  Port no. for SMTP: 25.
- **DNS**: Stands for **D**omain **N**ame **S**ervice. DNS maps human-addressable english domain names to IP addresses.e.g. www.abc.com might translate to 198.105.232.4. We as humans are comfortable in dealing with named addresses of websites (facebook.com, google.com etc.). However, to uniquely identify the server hosting the application, numeric IP addresses are required by the machine. DNS servers contain this mapping of named addresses to IP addresses. Whenever we us a named address is used, client machine services a request to the DNS server to fetch the IP address.
  Port no. for DNS: 53
- **DHCP**: Stands for **D**ynamic **H**ost **C**onfiguration **P**rotocol. Used for dynamic addressing of devices in a network. DHCP server keeps a pool of available IP addresses. Whenever a new device joins the network, it provides it with an IP from the available pool with an expiration time. DHCP is required in place of static addresses because current requirements involve managing devices which are continuously leaving/joining a network. Thus, a pool of available addresses are required which can be leased to devices currently residing in the network.
  Port no. for DHCP: 67, 68

## − Domain Name System

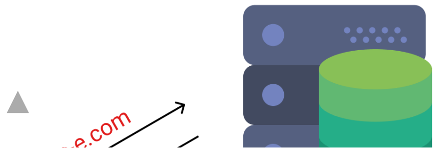
We are fortunate to have been living in the age of the internet, whose power and utility we often take for granted. We open our browser, type an URL or a search query of Google, and Boom!!, we are presented with thousands of results which we can browse, follow links within each page and visit multiple websites, the process wh mmonly known as web-surfing. But, have you ever wondered what happens under the hood of this process which seems so simple and fast to the end-user. We shall cover the whole walkthrough of what happens step-by-step starting from entering an URL on the address bar of the browser till the loading of the actual webpage. So, let's hop on this exciting journey!!

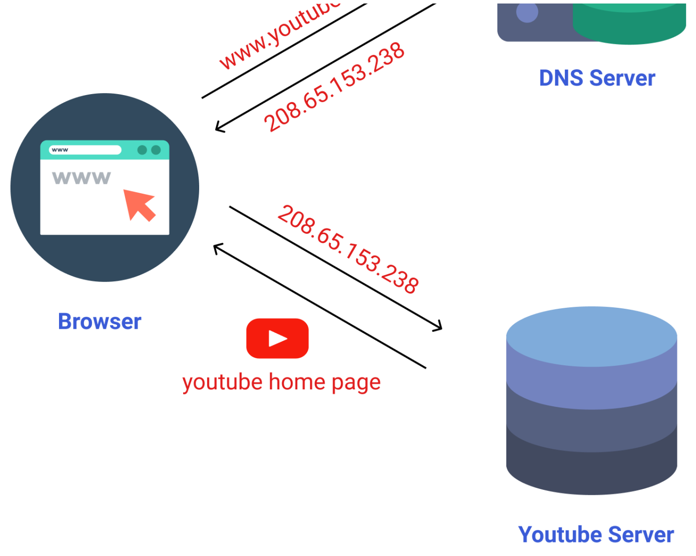
## Entering the URL

We use browsers to surf the internet (Chrome, Firefox, Edge, etc.). Each of them has an address-bar at the top where we provide the URL of the website we want to visit -



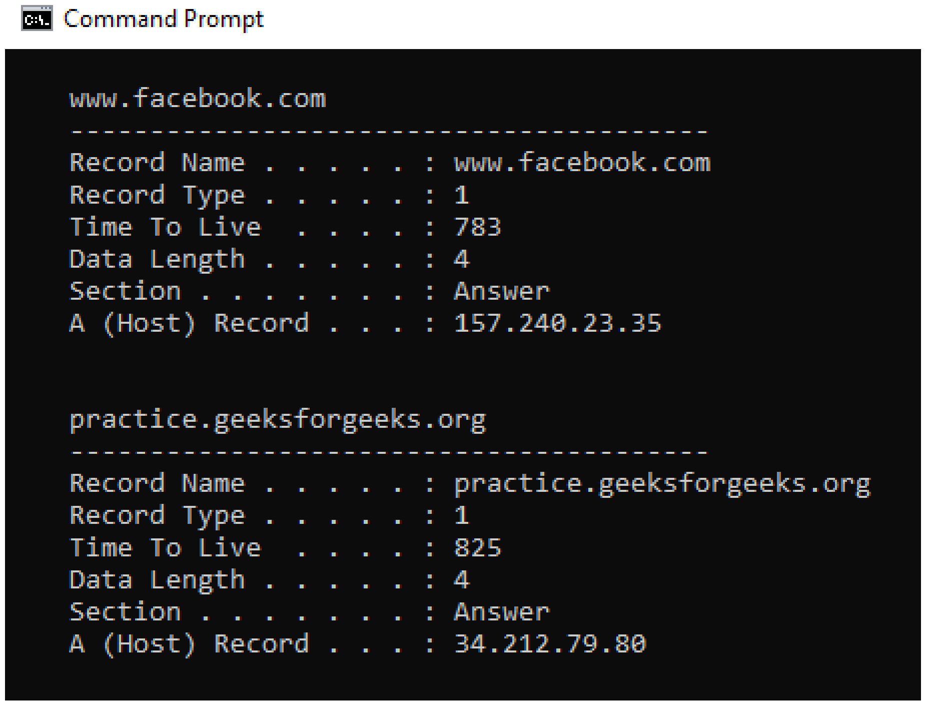
We enter some URL, say *www.google.com* or *youtube.com* as an example and press Enter. After waiting for a while, we are presented with the landing pages of the website. But, we all know Computer Systems can't understand human-language addresses. Also, human-language addresses have many issues such as (uppercase/lowercase, etc.). We have IP addresses (IPv4/IPv6) as the numerical equivalent for addressing in the realm of computer networks. An IP address is unique to a particular system at a time. i.e. A system running currently can't have multiple IP addresses ~ providing us the best scheme for addressing systems present in the network.
However, humans are not good with memorizing numbers (IPs) for websites, so the usage of English-language addresses can't be eliminated. Thus, we require some mechanism to map the English-language addresses to numeric IP addresses. Here, comes the role of DNS, which we cover in the next section. As of know, all we need to know is whenever we type an English-language address, the system calls the DNS server with the URL to get the corresponding IP address. Only, after the browser receives the IP, it can request the actual server for the webpage. The process looks as -





DNS Lookup

DNS calls are an extra overhead which serves us no good in loading the actual website. Thus, it would be very beneficial if we can cache DNS IP values for frequently visited websites in the user-system itself. Thus, comes the concept of DNS caching. Before making a call to the actual DNS server, the browser looks up the DNS cache of the system. The DNS cache looks as -



```
www.facebook.com
----------------------------------------
Record Name . . . . . : www.facebook.com
Record Type . . . . . : 1
Time To Live  . . . . : 783
Data Length . . . . . : 4
Section . . . . . . . : Answer
A (Host) Record . . . : 157.240.23.35


practice.geeksforgeeks.org
----------------------------------------
Record Name . . . . . : practice.geeksforgeeks.org
Record Type . . . . . : 1
Time To Live  . . . . : 825
Data Length . . . . . : 4
Section . . . . . . . : Answer
A (Host) Record . . . : 34.212.79.80
```

We can display the DNS cache information in Windows CMD as -

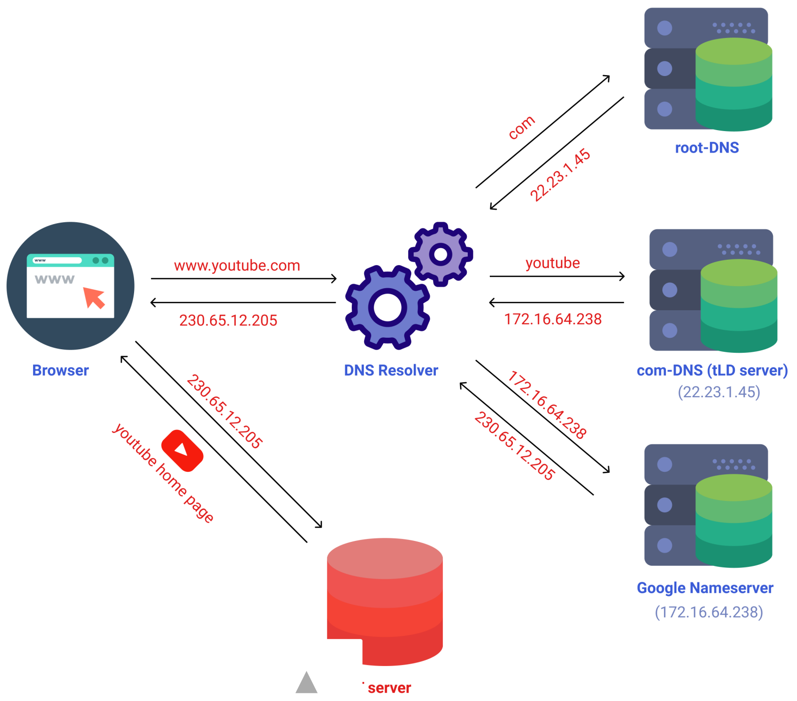
```

ipconfig /displaydns

```

Run

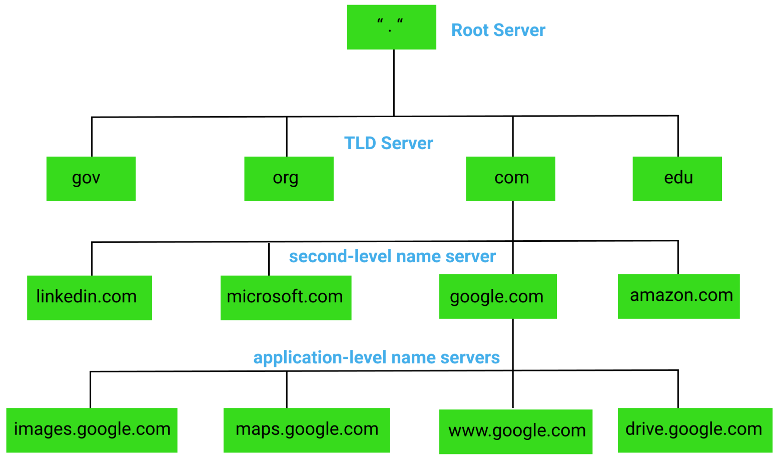
If no entry in Cache is found, then we call the DNS server. DNS server IP is either provided by the ISP, or there are public DNS servers provided by Google (8.8.8.8/8.8.4.4) or OpenDNS (208.67.222.222/208.67.220.220). These settings can be set/adjusted in Network Settings of the system. The system performs a DNS call with the address provided. The type of packets used is generally **UDP** because we require a lot of DNS calls and UDP packets are smaller in size (max. 512 bytes) as compared to TCP. Also, DNS requests are done on a separate port no. ~ 53.

## DNS Resolution

We shall understand this with an example. Say we search ***www.youtube.com***. DNS resolution occurs from end to start of the address. i.e. for our query, ***com -> youtube -> www*** . THe DNS resolver first requests the **root-DNS** with *com* as the search query.
*What is the root-DNS Server?* Root-DNS is the topmost level DNS server which contains the addresses of tLD (top-level-domain) name-servers such as *com, org, gov*. We have queried *.com*, so it returns the address of *.com tLD server.*
We thereafter query the *.com-tLD* server with the request for *youtube*. This name server looks up into its database and other similar tLDs and returns back the list of name servers matching youtube. Youtube is owned by Google, so we are returned name server list ***ns1.google.com-ns4.google.com***.
We then query one of these Google name servers for youtube, and we get back the IP address for the website which is geographically closest to our location. (Nowadays, the same website is hosted in a distributed fashion over multiple regions). Diagrammatically -



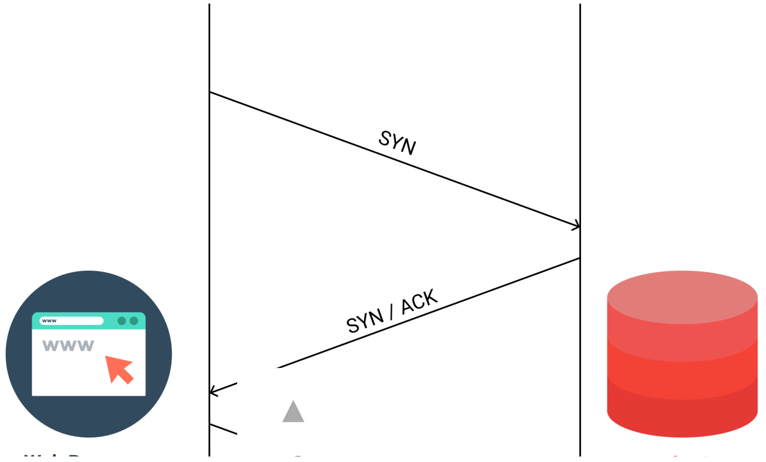
*NOTE* - All the IP addresses shown in the image is just for illustration. They are by no means accurate to the example discussed.

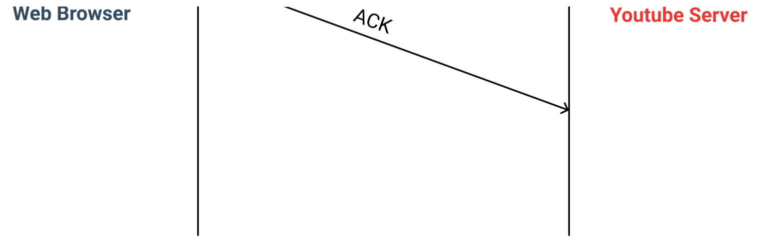


Above diagram shows the hierarchy of DNS servers and the various levels of resolution. After the browser receives the correct IP address for the requested website, it establishes a TCP connection which we describe below -

## TCP Connection & HTTP Request

Establishing a TCP connection is a 3-way handshake process which is described in the figure shown below -

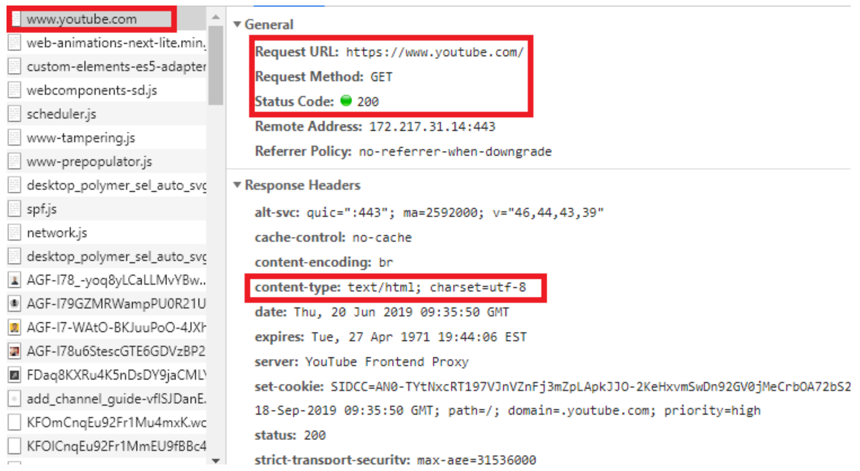




The client sends a **SYN** packet to the Youtube server to check whether it can accommodate any new connections or not. The server replies with a SYN/ACK (acknowledgment to the SYN request) back to the client. The client completes the 3-way handshake process by replying with its own acknowledgment (ACK).

## Website & Resource Delivery

After client and server are successfully connected via a secure TCP connection, browser issues an HTTP Request to the server demanding it to serve the page requested by the user. The server responds with an HTTP Response (the HTML page containing images, links to videos and other relevant data ~ JSON data perhaps) back to the client. The browser application then renders the source files received onto the user screen as a webpage.



```
www.youtube.com
web-animations-next-lite.min.
custom-elements-es5-adapter
webcomponents-sd.js
scheduler.js
www-tampering.js
www-prepopulator.js
desktop_polymer_sel_auto_svg
spf.js
network.js
desktop_polymer_sel_auto_svg
AGF-I78_-yoq8yLCaLLMvYBw..
AGF-I79GZMRWampPU0R21U
AGF-I7-WAtO-BKJuuPoO-4JXh
AGF-I78u6StescGTE6GDVzBP2
FDaq8KXRu4K5nDsDY9jaCMLY
add_channel_guide-vflSJDanE.
KFOmCnqEu92Fr1Mu4mxK.wc
KFOlCnqEu92Fr1MmEU9fBBc4
```

```
▼ General
    Request URL: https://www.youtube.com/
    Request Method: GET
    Status Code: ● 200
    Remote Address: 172.217.31.14:443
    Referrer Policy: no-referrer-when-downgrade

▼ Response Headers
    alt-svc: quic=":443"; ma=2592000; v="46,44,43,39"
    cache-control: no-cache
    content-encoding: br
    content-type: text/html; charset=utf-8
    date: Thu, 20 Jun 2019 09:35:50 GMT
    expires: Tue, 27 Apr 1971 19:44:06 EST
    server: YouTube Frontend Proxy
    set-cookie: SIDCC=AN0-TYtNxcRT197VJnVZnFj3mZpLApkJJO-2KeHxvmSwDn92GV0jMeCrbOA72bS2
    18-Sep-2019 09:35:50 GMT; path=/; domain=.youtube.com; priority=high
    status: 200
    strict-transport-security: max-age=31536000
```

## Sub-URL Resolution

What happens when we access, say *geeksforgeeks.org/data-strucure-and-algorithms/* or *geeksforgeeks.org/users/*. i.e. The issue we are trying to tackle here is how the part after the main URL gets resolved once we hit the main server for the website. This issue has been handled in 2 different ways, the 1st one of which has gone obsolete (the reason we discuss why) -

1. ***Seperate HTML files*** - In both the implementations, we are required to have one root file named **index.html**. We should have to keep the name as same, as this is the 1st webpage which is required to be served once HTTP request hits the server. It is the root/landing page in case a sub-URL is not specified. i.e. say we access *http://abc.com/*, then index.html present in the *abc server* will get served. Instead, if we access say http://abc.com/feed.html, then the server looks for feed.html file present in the directory and serves that to the client browser. Similarly, http://abc.com/profile.html asks the server to look for profile.html and serve it back. This kind of system relies on the presence of separate HTML files along with other resources (images, js files, CSS files, etc.). Each of them gets served based on the page requested. It is obvious that this kind of system won't scale. Imagine having a profile of millions of users. Then we would have to keep separate files like abc.com/profile/levi.html, abc.com/profile/eren.html, abc.com/profile/erwin.html

   .
2. ***Single Bundle File with API end-points*** - Modern systems have a single JS bundle file which contains the basic HTML structure to load in-case each URL is requested. Whenever we request a website, the whole bundle is downloaded into the user system. Accordingly, when we browse to different web-pages, API calls are made and the response data is injected into the HTML template (served by the bundle). Understanding this requires good knowledge of modern front-end frameworks (such as React, Angular, etc.) and REST-API backend frameworks (Nodejs, Django, etc.). Hence, it is out of the scope of this article to explain the overall mechanism in detail.

LIVE BATCHES

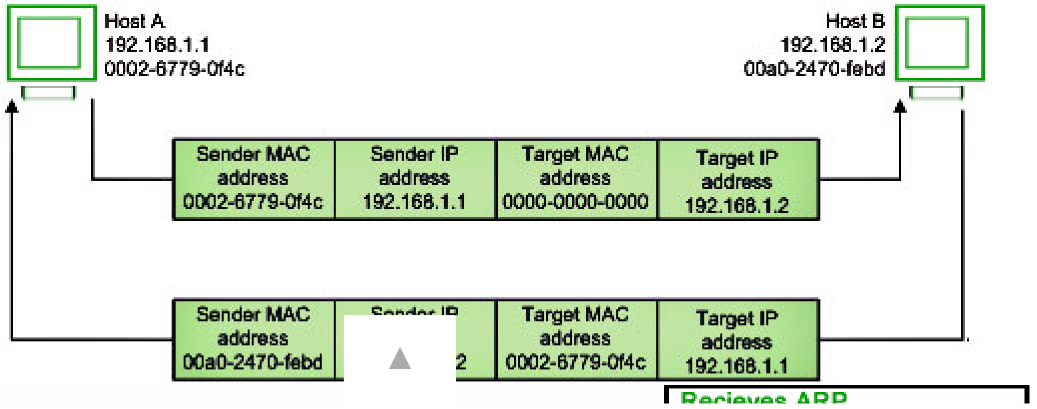
- ARP (Address Resolution Protocol) & Reverse-ARP

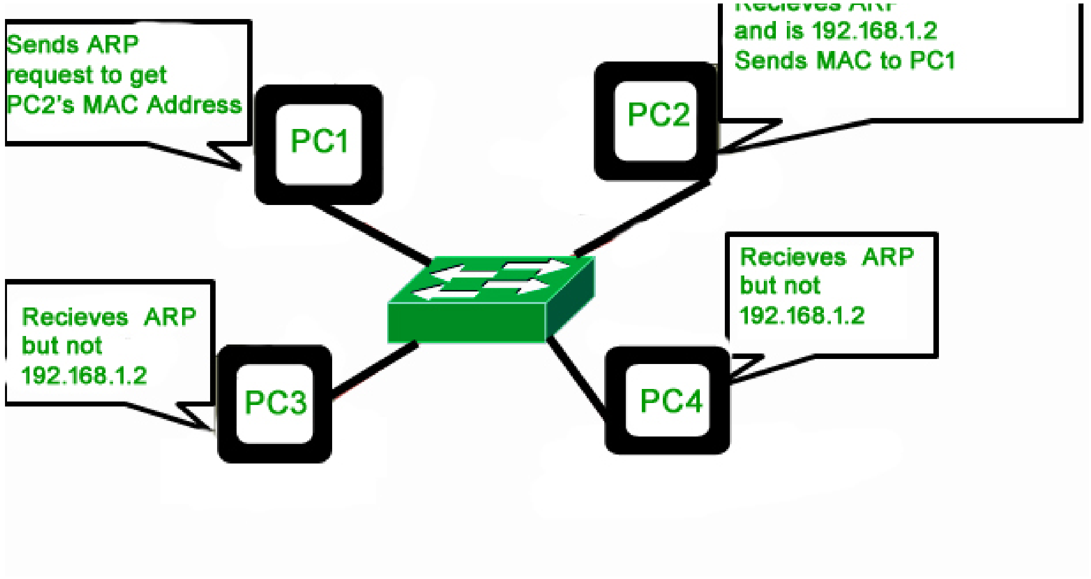
In a computer network, we have 2 addresses associated with a device (physical & logical). Physical Address is permanent and fixed (although can be changed, but shouldn't be done ~ MAC spoofing) for a device and doesn't change if the device changes network. Logical Addresses (IP) is transient and changes once device leaves current network and joins another. **To finally transmit data from one device to another, however physical/MAC address is required** (at Data-Link-layer). But, all a Network Layer knows is the logical address/IP of the next-hop-device. **ARP** (Address-Resolution-Protocol) is the de-facto method of acquiring the **physical address of next-hop from it's logical address**. Similarly, Reverse-ARP is the process of getting the IP address from the device's physical address.

**ARP** To get the MAC address of the target machine, the sender broadcasts a special ARP-message over it's immediate neighbors, requesting the MAC address. The contents of this message are:

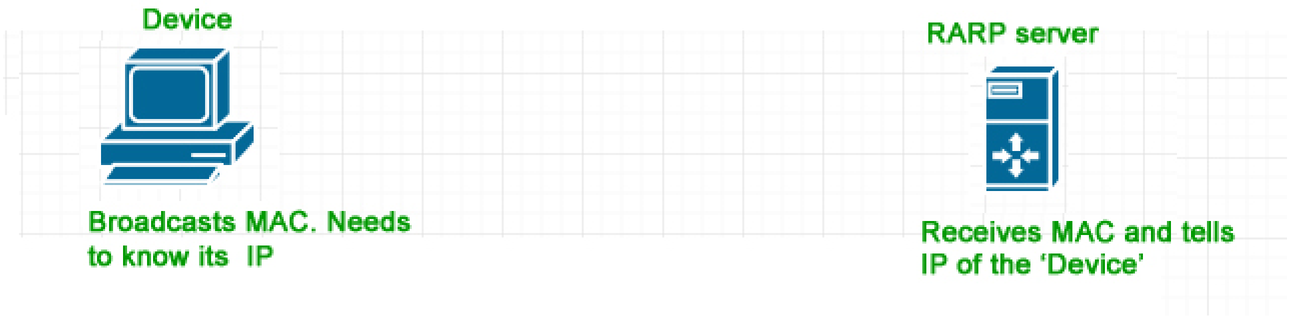
- Sender IP address
- Sender MAC address
- Destinaton MAC address (filled as all 0s initially)
- Destination IP addresss

Upon reception of this ARP message, the device associated with the destination IP, fills it's MAC address into the destination MAC space (filled with 0s), and unicasts it to the sender (Sender MAC is provided for this purpose). All other machines simply ignore the request. Finally, the sender recieves the reply and gets to know the destination MAC address.


Host A
192.168.1.1
0002-6779-0f4c
Host B
192.168.1.2
00a0-2470-febd
Sender MAC address 0002-6779-0f4c
Sender IP address 192.168.1.1
Target MAC address 0000-0000-0000
Target IP address 192.168.1.2
Sender MAC address 00a0-2470-febd
Sender IP
2
Target MAC address 0002-6779-0f4c
Target IP address 192.168.1.1
Recieves ARP


Sends ARP request to get PC2's MAC Address
and is 192.168.1.2 Sends MAC to PC1
PC1
PC2
Recieves ARP but not 192.168.1.2
PC3
PC4
Recieves ARP but not 192.168.1.2

**Reverse ARP** Reverse ARP is a networking protocol used by a client machine in a local area network to request its Internet Protocol address (IPv4) from the gateway-router's ARP table. The network administrator creates a table in gateway-router, which is used to map the MAC address to corresponding IP address.
When a new machine is setup or any machine which don't have memory to store IP address, needs an IP address for its own use. So the machine sends a RARP broadcast packet which contains its own MAC address in both sender and receiver hardware address field.


Device
RARP server
Broadcasts MAC. Needs to know its IP
Receives MAC and tells IP of the 'Device'

A special host configured inside the local area network, called as RARP-server is responsible to reply for these kind of broadcast packet[illegible] the RARP server attempt to find out the entry in IP to MAC address mapping table. If any entry matches in table, RARP server send the response packet to the requesting device along with IP address.

## DHCP(Dynamic Host Configuration Protocol)

**Dynamic Host Configuration Protocol(DHCP)** is an application layer protocol which is used to provide:

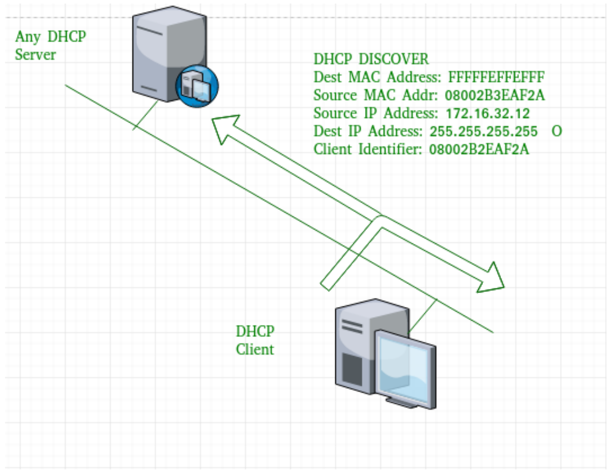
1. Subnet Mask (Option 1 - e.g., 255.255.255.0)
2. Router Address (Option 3 - e.g., 192.168.1.1)
3. DNS Address (Option 6 - e.g., 8.8.8.8)
4. Vendor Class Identifier (Option 43 - e.g., 'unifi' = 192.168.1.9 ##where unifi = controller)

DHCP is based on a client-server model and based on discovery, offer, request, and ACK.

DHCP **port number** for server is 67 and for the client is 68. It is a Client-server protocol which uses UDP services. IP address is assigned from a pool of addresses. In DHCP, the client and the server exchange mainly 4 DHCP messages in order to make a connection, also called DORA process, but there are 8 DHCP messages in the process.

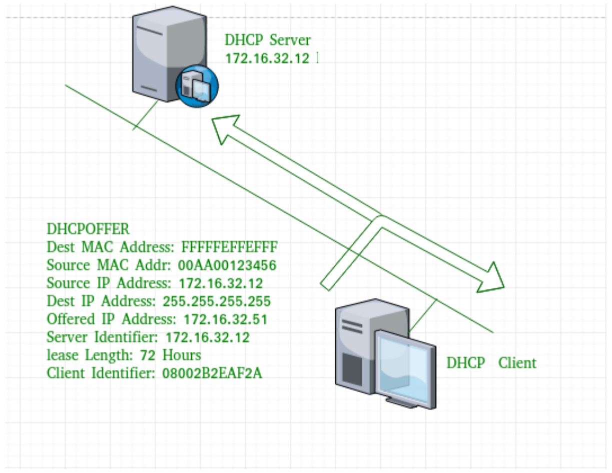
These messages are given as below:

1. **DHCP discover message -** This is a first message generated in the communication process between server and client. This message is generated by Client host in order to discover if there is any DHCP server/servers are present in a network or not. This message is broadcasted to all devices present in a network to find the DHCP server. This message is 342 or 576 bytes long


Any DHCP Server
DHCP DISCOVER
Dest MAC Address: FFFFFEFFEFFF
Source MAC Addr: 08002B3EAF2A
Source IP Address: 172.16.32.12
Dest IP Address: 255.255.255.255 O
Client Identifier: 08002B2EAF2A
DHCP Client

As shown in the figure, source MAC address (client PC) is 08002B2EAF2A, destination MAC address(server) is FFFFFFFFFFFF, source IP address is 0.0.0.0(because PC has no IP address till now) and destination IP address is 255.255.255.255 (IP address used for broadcasting). As the discover message is broadcast to find out the DHCP server or servers in the network, therefore, broadcast IP address and MAC address is used.

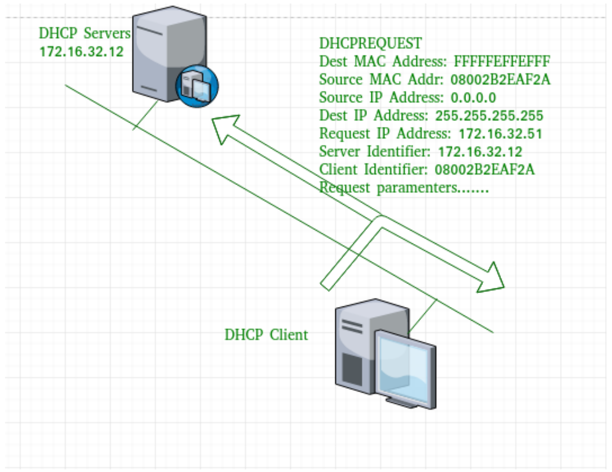
2. **DHCP offer message -** The server will respond to host in this message specifying the unleased IP address and other TCP configuration information. This message is broadcasted by the server. Size of the message is 342 bytes. If there are more than one DHCP servers present in the network then client host will accept the first DHCP OFFER message it receives. Also, a server ID is specified in the packet in order to identify the server.


DHCP Server
172.16.32.12
DHCPOFFER
Dest MAC Address: FFFFFEFFEFFF
Source MAC Addr: 00AA00123456
Source IP Address: 172.16.32.12
Dest IP Address: 255.255.255.255
Offered IP Address: 172.16.32.51
Server Identifier: 172.16.32.12
lease Length: 72 Hours
Client Identifier: 08002B2EAF2A
DHCP Client

Now, for the offer message, source IP address is 172.16.32.12 (server's IP address in the example), destination IP address is 255.255.255.255 (broadcast IP address) ,source MAC address is 00AA00123456, destination MAC address is FFFFFFFFFFFF. Here, the offer message is broadcast by the DHCP server, therefore, destination IP address is broadcast IP address and destination MAC address is FFFFFFFFFFFF and the source IP address is the server IP address and MAC address is server MAC address.

Also the server has provided the offered IP address 192.16.32.51 and lease time of 72 hours(after this time the entry of host will be erased from the server automatically) . Also the client identifier is PC MAC address (08002B2EAF2A) for all the messages.

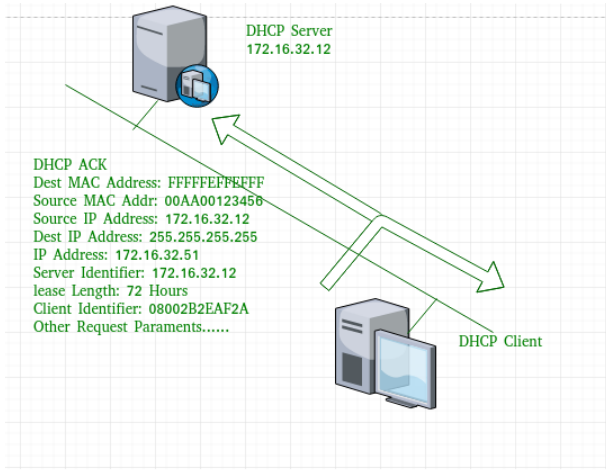
3. **DHCP request message -** When a client receives a offer message, it responds by broadcasting a DHCP request message. The client will produce a gratitutous ARP in order to find if there is any other host present in the network with same IP address. If there is no reply by other host, then there is no host with same TCP configuration in the network and the message is broadcasted to server showing the acceptance of IP address .A Client ID is also added in this message.


DHCP Servers
172.16.32.12
DHCPREQUEST
Dest MAC Address: FFFFFEFFEFFF
Source MAC Addr: 08002B2EAF2A
Source IP Address: 0.0.0.0
Dest IP Address: 255.255.255.255
Request IP Address: 172.16.32.51
Server Identifier: 172.16.32.12
Client Identifier: 08002B2EAF2A
Request paramenters.......
DHCP Client

Now, the request message is broadcast by the client PC therefore source IP address is 0.0.0.0(as the client has no IP right now) and destination IP address is 255.255.255.255 (broadcast IP address) and source MAC address is 08002B2EAF2A (PC MAC address) and destination MAC address is FFFFFFFFFFFF.

**Note -** This message is broadcast after the ARP request broadcast by the PC to find out whether any other host is not using that offered IP. If there is no reply, then the client host broadcast the DHCP request message for the server showing the acceptance of IP address and Other TCP/IP Configuration.

4. **DHCP acknowledgement message -** In response to the request message received, the server will make an entry with specified client ID and bind the IP address offered with lease time. Now, the client will have the IP address provided by server.



DHCP ACK
Dest MAC Address: FFFFFFFFFFFF
Source MAC Addr: 00AA00123456
Source IP Address: 172.16.32.12
Dest IP Address: 255.255.255.255
IP Address: 172.16.32.51
Server Identifier: 172.16.32.12
lease Length: 72 Hours
Client Identifier: 08002B2EAF2A
Other Request Paraments......

Now the server will make an entry of the client host with the offered IP address and lease time. This IP address will not be provided by server to any other host. The destination MAC address is FFFFFFFFFFFF and the destination IP address is 255.255.255.255 and the source IP address is 172.16.32.12 and the source MAC address is 00AA00123456 (server MAC address).

5. **DHCP negative acknowledgement message -** Whenever a DHCP server receives a request for IP address that is invalid according to the scopes that is configured with, it send DHCP Nak message to client. Eg-when the server has no IP address unused or the pool is empty, then this message is sent by the server to client.

6. **DHCP decline -** If DHCP client determines the offered configuration parameters are different or invalid, it sends DHCP decline message to the server .When there is a reply to the gratuitous ARP by any host to the client, the client sends DHCP decline message to the server showing the offered IP adc already in use.

7. **DHCP release -** A DHCP client sends DHCP release packet to server to release IP address and cancel any remaining lease time.

8. **DHCP inform -** If a client address has obtained IP address manually then the client uses a DHCP inform to obtain other local configuration parameters, such as domain name. In reply to the dhcp inform message, DHCP server generates DHCP ack message with local configuration suitable for the client without allocating a new IP address. This DHCP ack message is unicast to the client.

**Note -** All the messages can be unicast also by dhcp relay agent if the server is present in different network.

**Advantages -** The advantages of using DHCP include:

- centralized management of IP addresses
- ease of adding new clients to a network
- reuse of IP addresses reducing the total number of IP addresses that are required
- simple reconfiguration of the IP address space on the DHCP server without needing to reconfigure each client

The DHCP protocol gives the network administrator a method to configure the network from a centralised area.
With the help of DHCP, easy handling of new users and reuse of IP address can be achieved.

**Disadvantages -** Disadvantage of using DHCP is:

- IP conflict can occur