2. OSI Layer Software Architecture

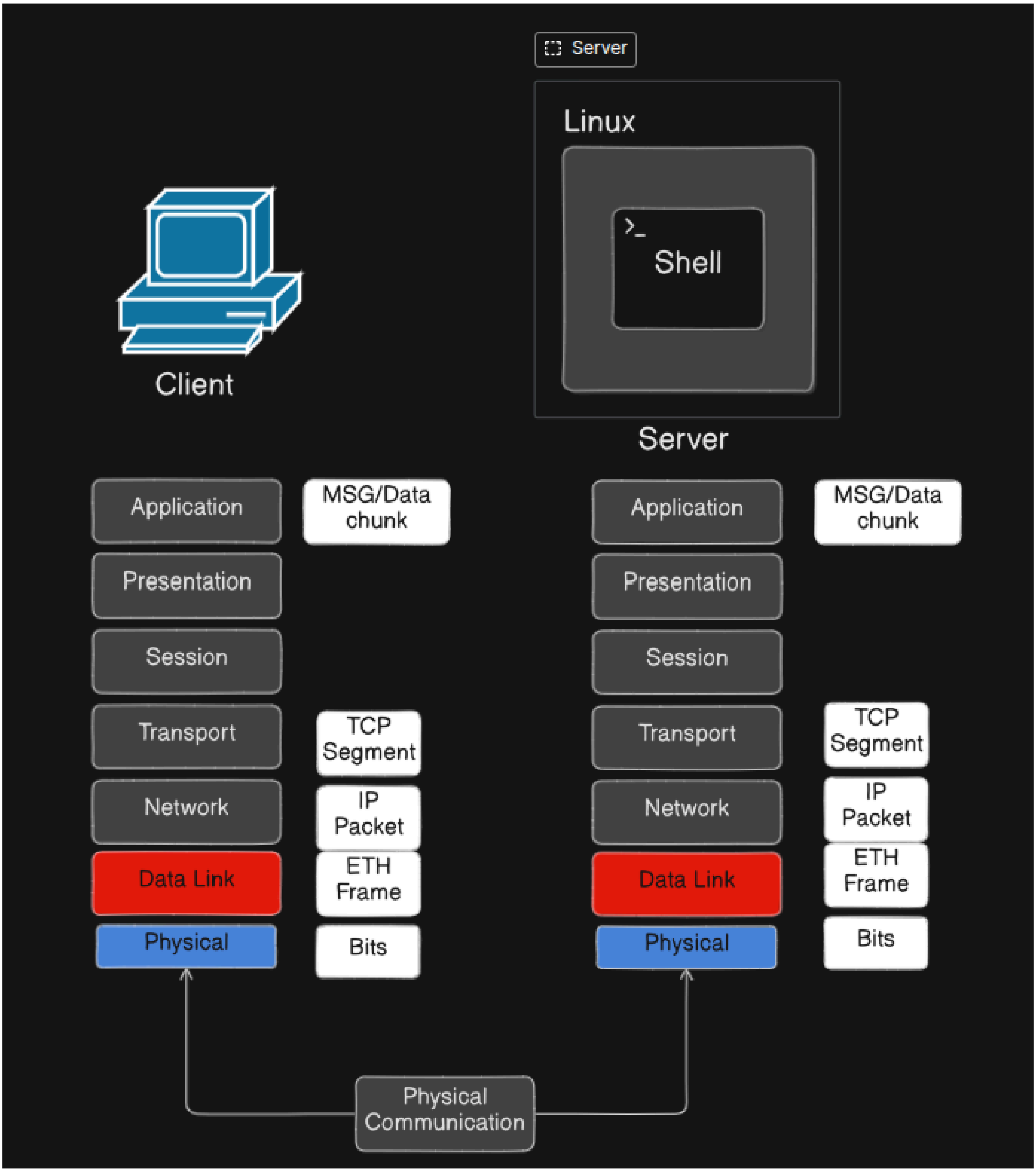
OSI

Open systems interconnection was found to make all the founders of network devices and all the developers developing applications over the network follow one standard.

This standard exists on each device.

OSI Model

OSI model consists of 7 ***logical layers*** not ***physical layers*** for organization purposes in networking.



Each layer is meant to do a set of functions these functions can be:

- **SW Functions**
- **HW Functions**
- **Firmware Functions**
  Software implemented to deal with certain protocols in the HW.

The naming convention of data at each layer is very important as it identify the data plus protocol installed over, meaning if we are saying:

- we have a problem in the MSG this means that the `http` message has a problem.
- We have a problem in the segment this means that the `TCP` header is corrupted.

Real Life Application on Networking

Opening a Web Site use case.

When you search for a website and press enter your request will pass through the **7 OSI** layers in your machine and then gets forwarded towards the server on receiving the request at the server side it pass through the **7 OSI** layers in reverse order.

Application Layer

Application here doesn't stand for the GUI interface that you are using it stands for the protocols like:

- `HTTP`
- `HTTPS`
  The `https` contains a lot of things one of them is a message format which interpret that this request is an `https` and what does it demands.

The request is ready but we need to make host name resolving to change the website name to its corresponding IP address by DNS;
In the domain `www.google.com`:

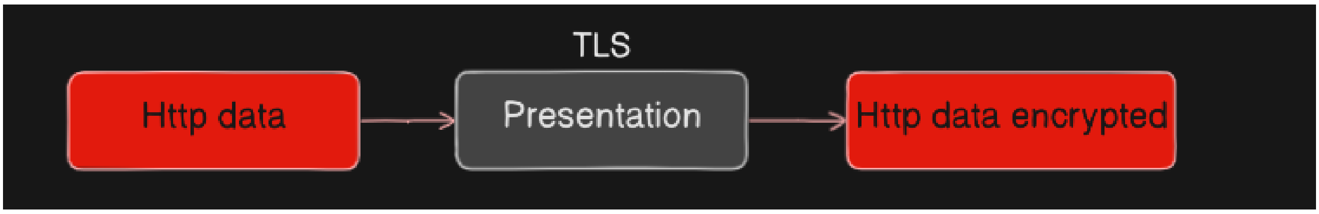
- `google` = **Second-level domain**
- `.com` = **Top-level domain (TLD)**

This process is divided into steps:

1. **DNS recursor(resolver).**
   Which communicate with Root nameserver.
2. **Root nameserver**
   Which communicate with **TLD nameserver** which has the suffix in the Domain name for instance `.com`.
3. **TLD nameserver**
   It stands for ***Top Level Domain***.
   It's the enterprise which is responsible for `top level domain` like `.com`
   Which communicate with Authoritative nameserver that have sold the ***Second-level domain*** and ask for its IP to return to the application layer with it.
4. **Authoritative nameserver**
   Returns the IP address corresponding to the ***Second-level domain***.

These servers communicate together to fetch the IP corresponding to the DNS of the website.

Presentation Layer



Is the layer where the request formed by the `http` is encrypted in by the `TLS` protocol.

After resolving the DNS successfully the `https` is the secured version of `http` thus, `TLS` / `SSL` (previously) who does the encryption.

ال Presentation هنا معناها هيئة البيانات بمعنى أن هنا ده المكان البعمل في Encryption و Decryption و بغير فيه في شكل ال Data.

Session layer

Asking the OS to open a **network socket** and here **where the connection begins actually** in the previous two layers the connection hasn't started we just:

- Constructed the request message(Application Layer).
- Resolved the DNS.(Application Layer).
- Encrypted the Message(Presentation Layer).

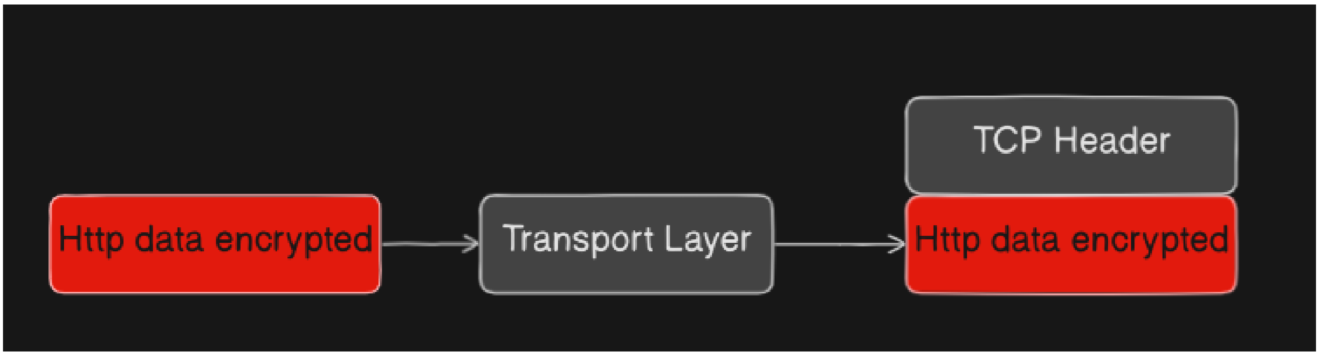
The network socket can be considered as a data structure object that gathers information about:

- **Requester**
- **Provider**
- **Ports that serves this connection**
  1. Port provided by the OS to receive the data on.
  2. Port to request/send the data over.

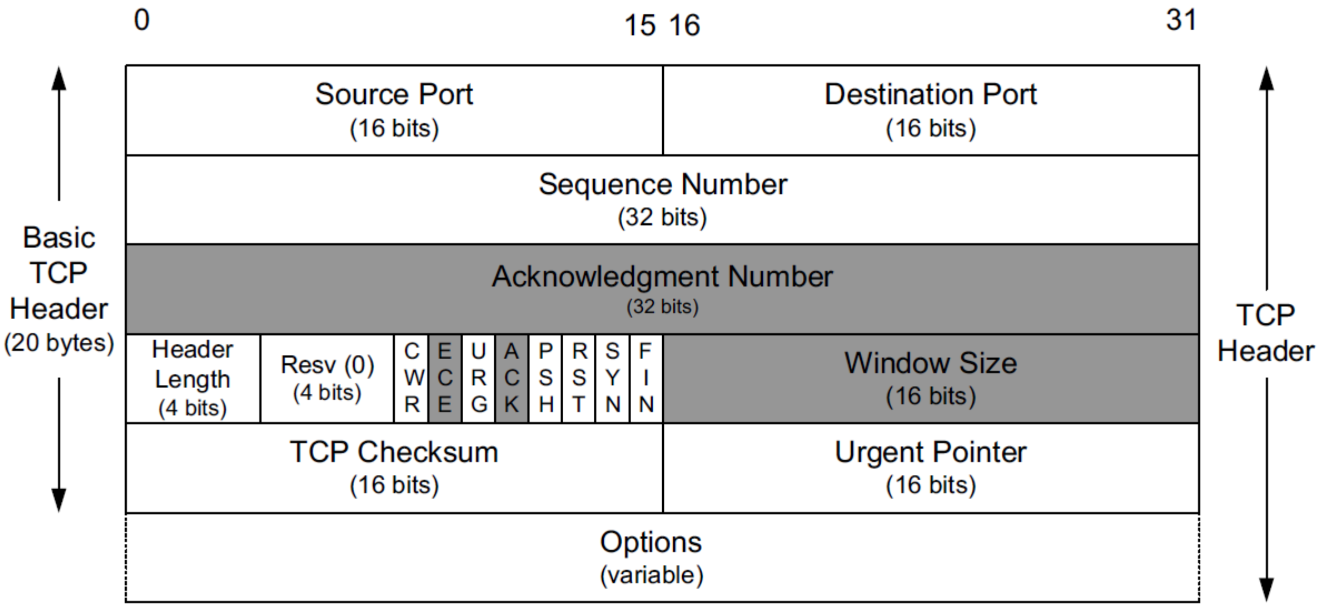
This data is saved in the OS to send and receive data based on.

Also in this layer you determine the protocol that you will use in the layer it follows.

Transport Layer



We take the HTTP data and put over the `TCP` header which can displayed in the following picture:



The `TCP` header is different from the `http` :

The `TCP` header is a ***binary based protocol*** written all in binary and the location where the binary is placed in is crucial.

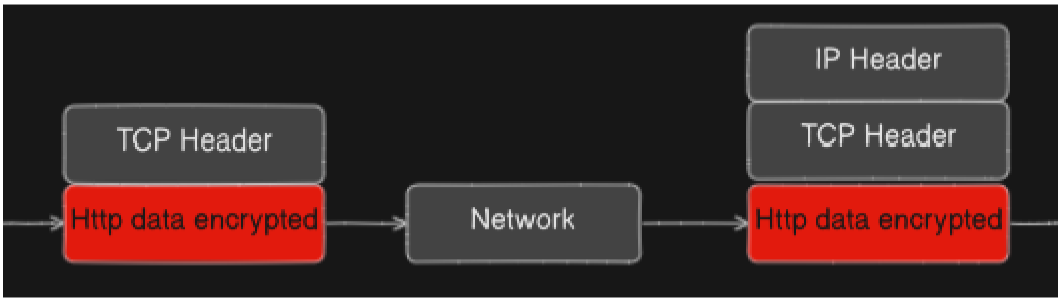
While the `http` is a ***text based protocol*** which is based on header and value attached to it and the file arrangement doesn't matter as we search for the header to read the value attached to it.

The `TCP` is the first protocol which divides the data into ***segments*** each having its ***sequence number***.

Network Layer (IP)

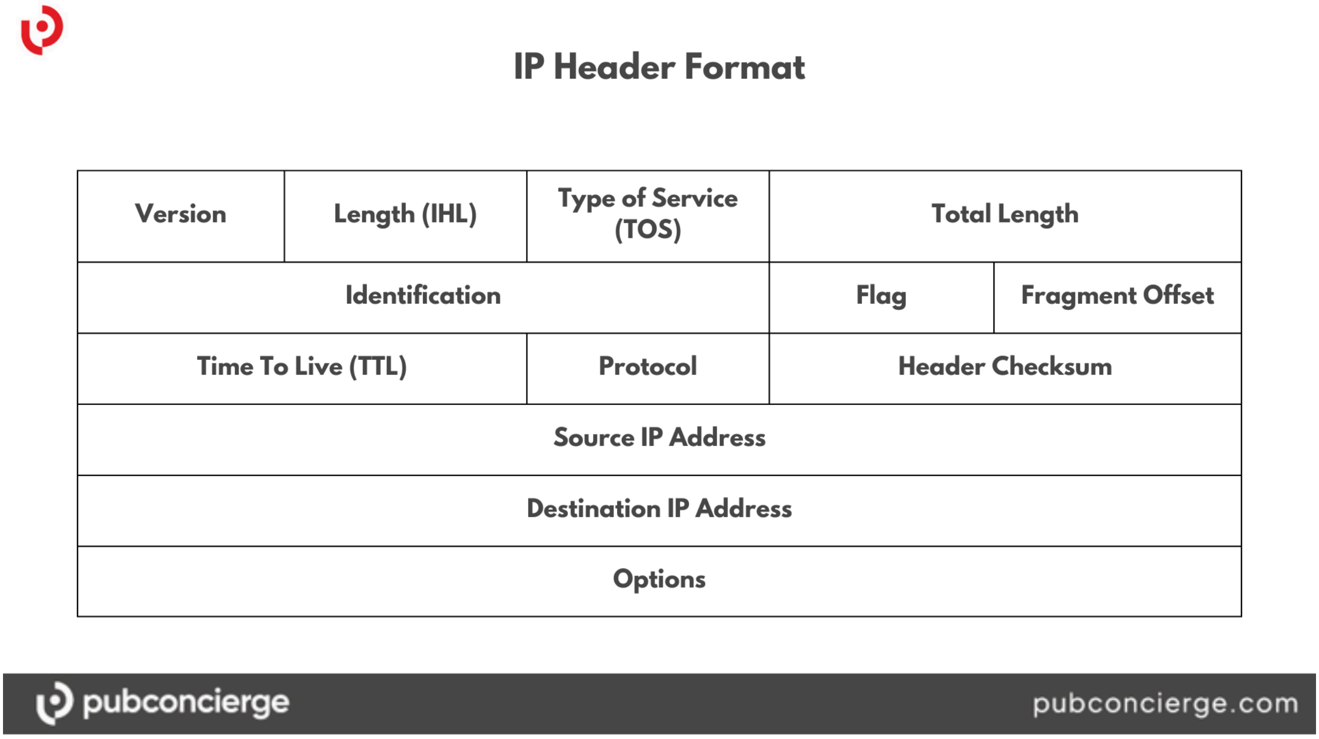
It does:

- **IP addressing**.
- **Packet Switching**.
- **Routing Protocols**.



Receive the data from the transport layer which is:
`http` data encrypted by `tls` and added over the `tcp` header and maybe got segmented.

And then place over the IP header which is a binary based header which each binary value should be placed in its correct location at the header to give its correct meaning.



IP Header Format

<table>
<tr><td>Version</td><td>Length (IHL)</td><td>Type of Service (TOS)</td><td colspan="2">Total Length</td></tr>
<tr><td colspan="3">Identification</td><td>Flag</td><td>Fragment Offset</td></tr>
<tr><td colspan="2">Time To Live (TTL)</td><td>Protocol</td><td colspan="2">Header Checksum</td></tr>
<tr><td colspan="5">Source IP Address</td></tr>
<tr><td colspan="5">Destination IP Address</td></tr>
<tr><td colspan="5">Options</td></tr>
</table>

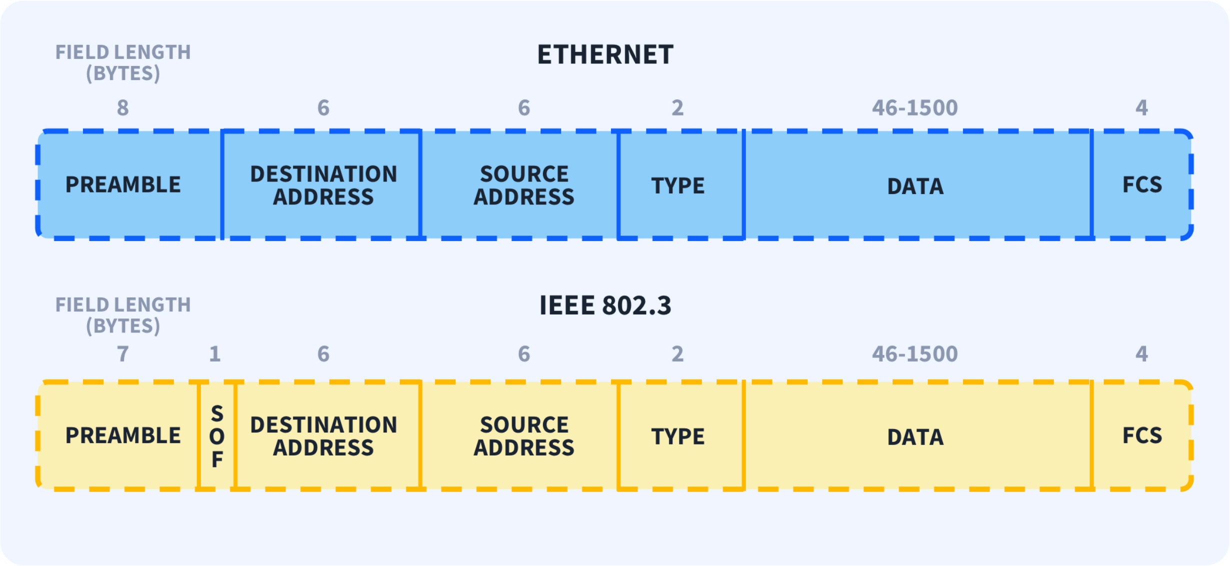
Protocol refers for the protocol which is used in the Transport Layer.

Destination IP address is which we previously resolved in the application layer when we were making an `http` request.

Data Link Layer (Ethernet Protocol)

It does:

- **Data flow control for the Physical Layer.**
- **It does error detection after receiving data from Physical Layer.**

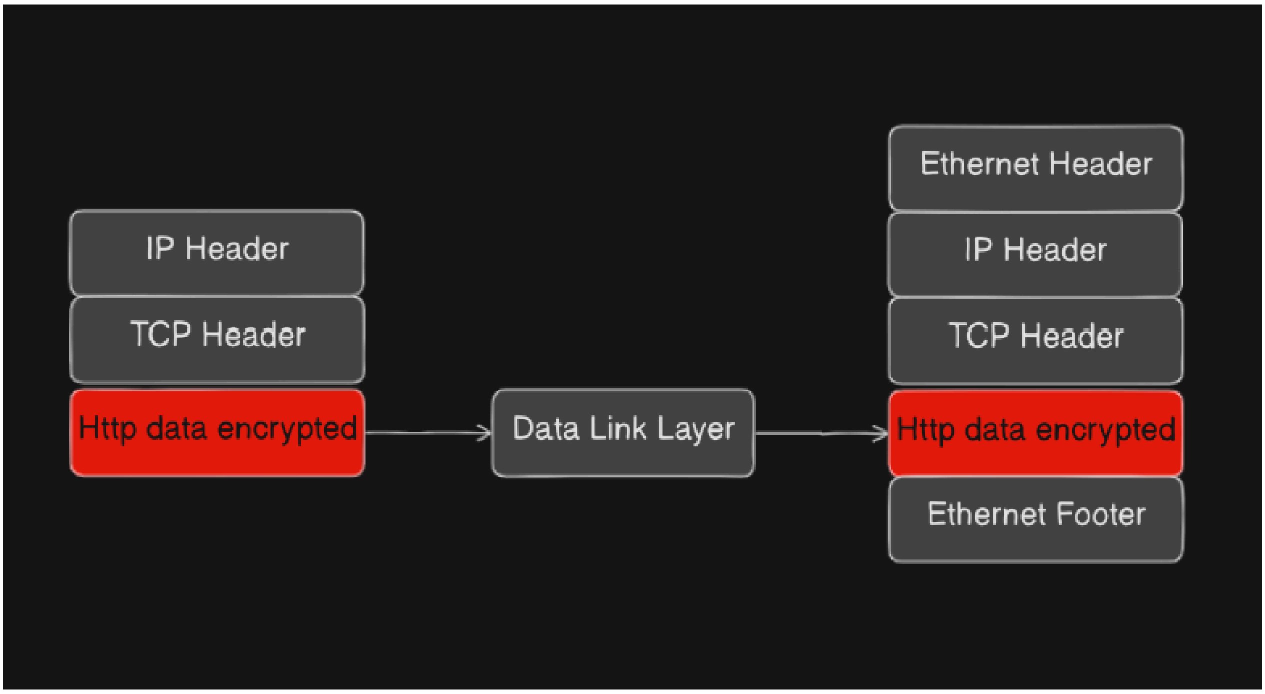


ETHERNET

FIELD LENGTH (BYTES)	8	6	6	2	46-1500	4
	PREAMBLE	DESTINATION ADDRESS	SOURCE ADDRESS	TYPE	DATA	FCS

IEEE 802.3

FIELD LENGTH (BYTES)	7	1	6	6	2	46-1500	4
	PREAMBLE	SOF	DESTINATION ADDRESS	SOURCE ADDRESS	TYPE	DATA	FCS



The data is encapsulated by:

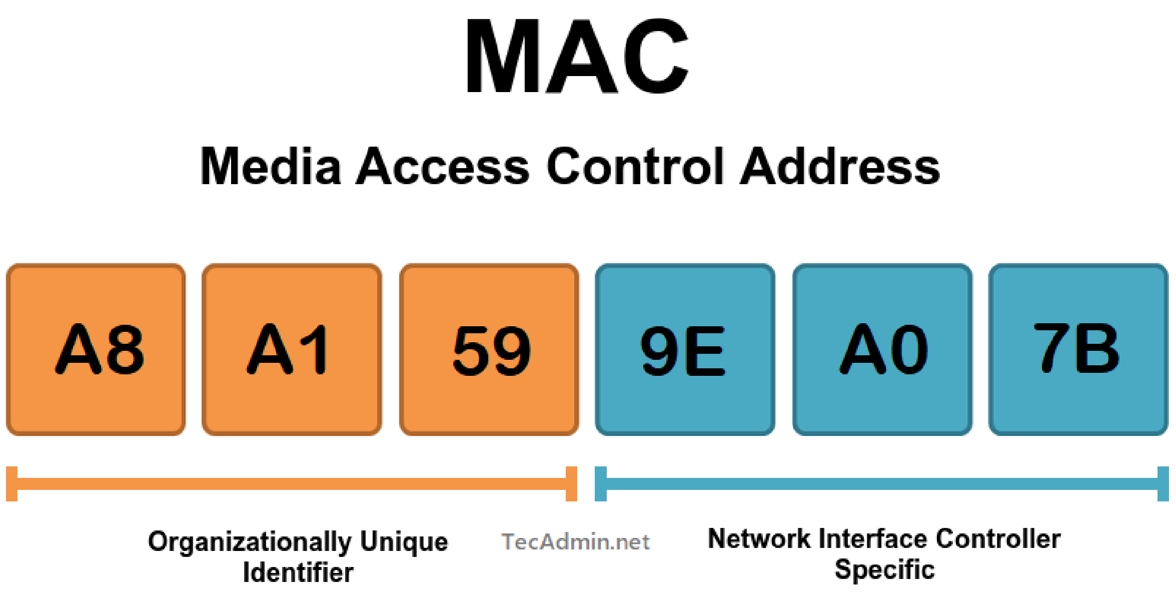
- **Ethernet Header**.
- **Ethernet Footer**.
  For error detection using various algorithms.
  The Ethernet frame is no more than binary bits placed in specific locations to have a physical meaning for the receiver.

The Ethernet header contains:

- **Preamble**
  Which is a 56 bit data of `1's` and `0's` alternatively for ***Synchronization***.
- **Destination MAC address**.
  Is a physical address unique and assigned for your device.
- **Source MAC address**.

The destination MAC address which the data is going to be transported with is the MAC address of the router(gateway); the server destination MAC address will be known in the internal network for security reasons.

MAC Address Structure


MAC
Media Access Control Address
A8
A1
59
9E
A0
7B
Organizationally Unique Identifier
TecAdmin.net
Network Interface Controller Specific

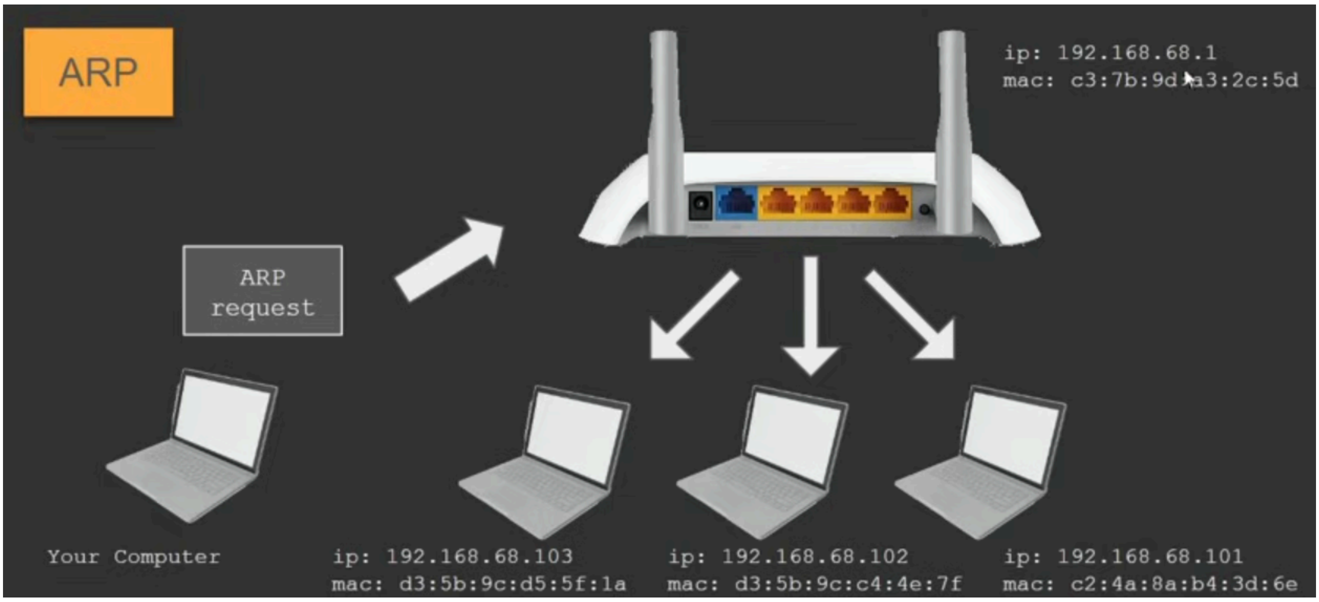
ARP(Address Resolution Protocol)

It is a network protocol used to determine the **MAC address** (hardware address) from any **IP address**.

In other words, ARP is used to mapping the **IP Address** into **MAC Address**.
When one device wants to communicate with another device in a LAN (local area network) network, the ARP protocol is used.

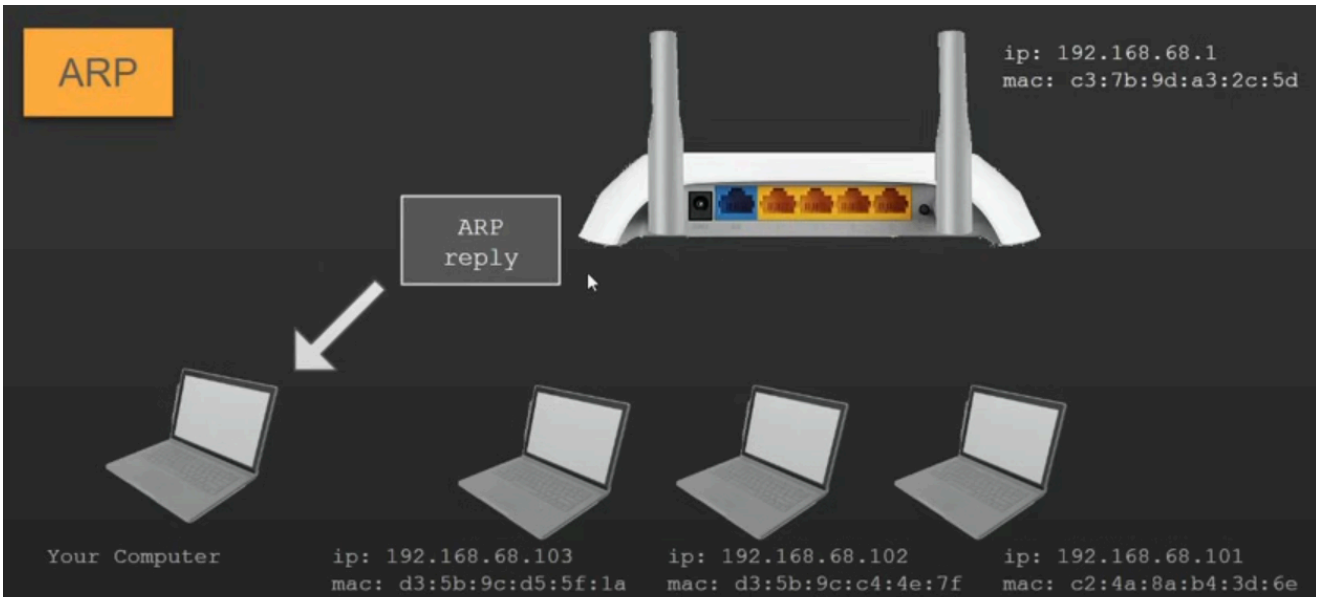
When your device boots up and gets an IP via DHCP, it also receives the **default gateway IP address** (your router).

ARP request:



This is a request message sent by the device with a destination MAC address which wants to know the MAC address corresponding for that device; The router do as following take this message forwards it to all the devices connected on the LAN and waits for reply if no device replied this means that this message wasn't headed to any of them and it was targeting the router thus, the router will respond with **ARP reply**.

ARP reply



The reply message will contain the MAC address of the router.
After receiving it on my machine we will be capable to complete the Ethernet frame which had had the destination MAC address missing.

Physical Medium

- **Copper Cables**.
- **Optical Cables**.
- **Radio Wave**.
  What happens also in this layer is the line coding which is related to communication level for representing binary bits with the correct code for:
- **Saving BW**.
- **Synchronization**.
- **Error detection**.