Computer Network

A Network could be wired or wireless

Network :→ A group or system of interconnected people or things.

Computer Network :→ A group or system of inter-connected computers.

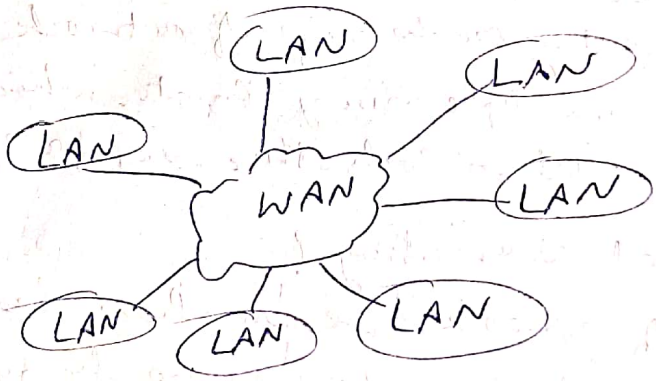
Internet is just a Giant Network.

Traditionally computer Network consisted of desktop, servers, printers etc but now we have vast nos of smart devices connected to our network.

LAN :→ Local Area Network, Any network in a single area is called land.

The LAN is usually connected to the outside world using a socket in the wall that your Internet Service provider has installed.

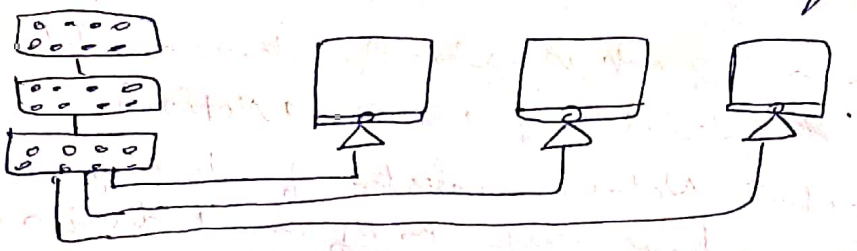
This socket connects us to the WAN



So The purpose of WAN is to connect various LAN together.

eg: internet.

Business type Network :-



These are our office computers and all of them will usually plug into a wall or floor socket.

All these sockets will have cables running through the floor or the ceiling (may be embedded inside wall) back to the special room called a wiring closet or a patch room or server room

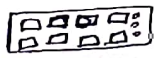
In wiring closet all these devices will connect to a stack of switches which are mounted onto racks depending on the size of network there might be multiple wiring closets, which are all connected to each other - say for eg: on different floors

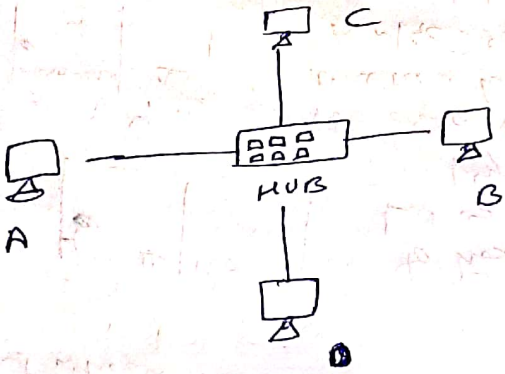
Q) So why do we need Networking ?

Ans» The Beauty of network is we can share resources.
If there was no network, their is no sharing of files!
think networks as human network, how easy our task becomes if we have a good network.

* Resources could be anything data, software, hardware etc.

HUB :→ It's job is to connect devices in your network.





Bandwidth → rate of data transfer in a network

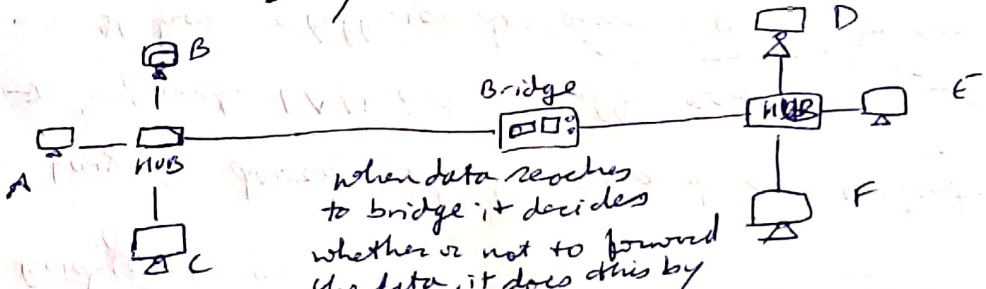
If host A sends data to host B hub will give this data to host C and D also but they will discard it.

- The hub has many downfalls, it wastes bandwidth as we see host C and D receives the data which is unwanted to them
- A hub uses half Duplex which means it can't send and receive data at the same time without data collision. Even if you try, it will cause collision and makes your data corrupt. i.e 1 collision domain.
- A hub is a Layer 1 device meaning, it has no knowledge of address, all it does is repeats any data it receives.
- It has security risks.

Hubs are old technology, they are now being replaced by switches

Bridges :→ Bridges were introduced to overcome some of the shortcomings of Hub.

They are used to segment networks into smaller sections.



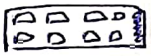
when data reaches to bridge it decides whether or not to forward the data, it does this by looking to address of destination and source (MAC address)

A Bridge is a

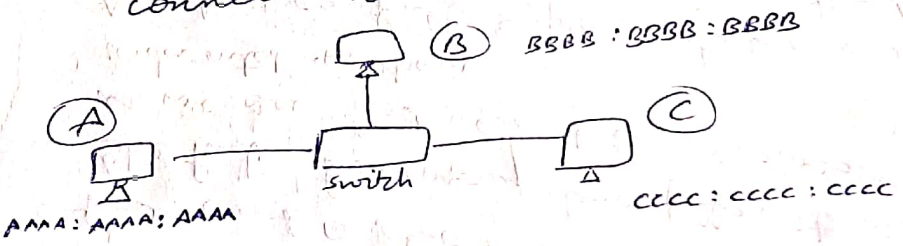
→ Layer 2 device which means it can understand MAC addresses
→ Segments LAN into smaller sections. → because it usually has 2 ports
→ It has 2 collision domains i.e. data can be send & received by each section of the network at the same time. (Full duplex)
→ It has fewer ports (Usually two)

Bridges aren't used anymore, they are being replaced by Switches.

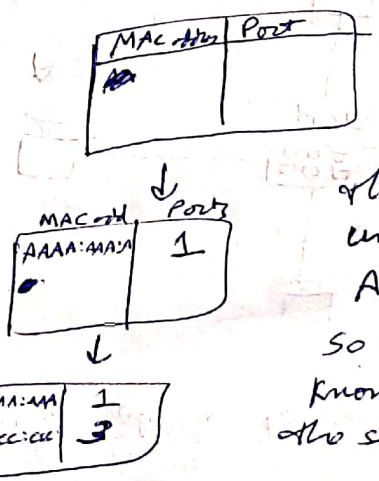
Switches :→ Think of Switch as a hub and bridge pulled inside a single unit.



Using it we can connect devices but switch knows the MAC address of each device i.e. which port connects to which host.



A switch has mac address table which list all MAC address and which port it connected to



Lets say host A sends some data to host C, the switch receives the data but as you can see the MAC address table is empty, in this case the switch will forward the data to B and C just like a hub but unlike a hub, the switch now learned the host A MAC address can be reached on Port 1.
So now if host C wants to send data to host A, the switch knows exactly where to send the data so now the switch learns that host C can be reached through Port 3. By using this process switch sorts things.

Switch

↳ It's a Layer 2 device i.e. it learns MAC address

→ It uses full duplex i.e can send and receive data at same time

→ Multiple Collision Domains

→ Saves Bandwidth

→ Security Increased.

Routers :→ you can think of them as doorway out of your internal networks into the outside world.

↳ It's a Layer 3 device i.e. it uses your MAC as well as IP address too.

→ Fewer ports

→ It's highly configurable and has many many different features

OSI Model → 7 layer model
→ standardized network.

- It's a theory Article, i.e. stack of seven layers that can be used as a reference to help understand how networks operate.
  or concept

- The model was introduced to standardised the network in a way that allow multi-vendor system
  - Prior to this you'll only able to have one vendor network because the devices from one vendor couldn't communicate with others.
  - We don't actually use the OSI model, we use TCP/IP model the concepts their are exactly same, just the layers are slightly different.

Q) So if we don't use this model, why to learn?
Ans → Because we still refer to it while troubleshooting or deciding network operation.

Layers

7.) Application [SMTP, FTP, Telnet] → this is where App and User communicate
6.) Presentation [Format Data, Encrypt] → It formats the data in a way that receiving app can understand it
5.) Session [Start & Stop Session] → It's responsible for establishing and terminating connections between devices
4) Transport [TCP, UDP, Port Numbers] → It adds the transport protocols, it also adds source and destination port numbers.
3.) Network [IP Address, Routers] → At this stage logical address i.e. IP address of source and destination are added.
2.) Data Link [MAC Address, Switches] → At this layer physical addresses are added to data i.e. from where its coming and to where its going.
1.) Physical Layer [Cable, hubs, Network Interface Cards etc.]

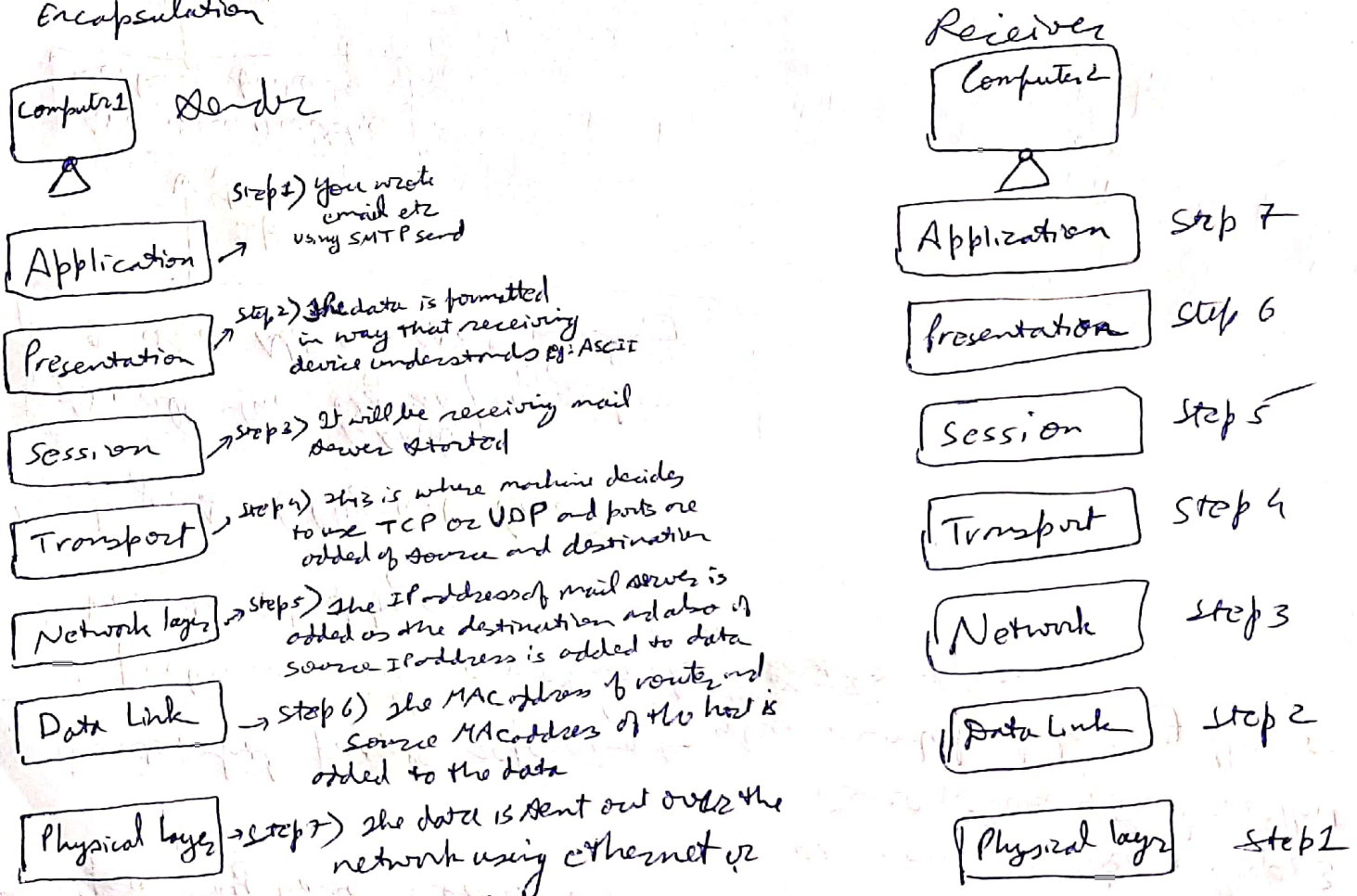
↓
It's key responsibility is to carry the data across physical hardware eg: Ethernet Cable

A way to remember these as

All	Application layer
people	Presentation layer
seem	Session.
to	Transport
need	Network link
data	Data Link
Processing	Physical layer

Eg: Say you send an Email and the data traverse using OSI Model

- adding and processing data on each layer. This Process is called Encapsulation

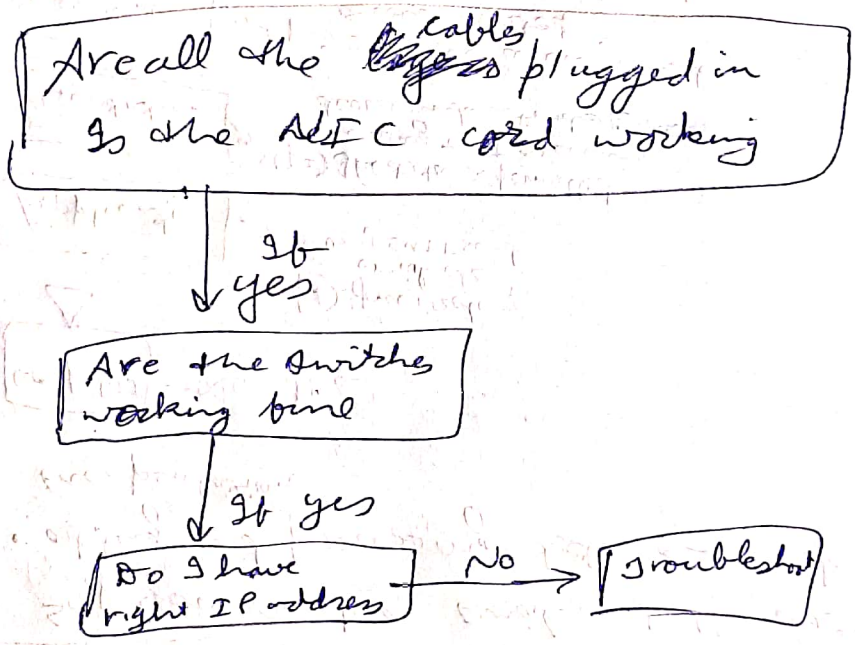


* Note when data gets to other side it gets processed in same way but in reverse order

Q) How it (OIC Model) can be used in Troubleshooting?

Ans) you may have heard the term → that's a layer 2 problem or sounds like a layer 3 issue when you hear that thats what people referring to this model, lets say there's a problem with a network layer ~~Level~~ then how to identify :=>

1) Layer 1 (Physical layer)



TCP/IP Model :→ It's a Model to standardise computer networking

Sounds familiar
Well it's the same as OSI model
now OSI model which is widely referenced
But not used in the real world.
The TCP/IP model is the real deal

Here is the TCP/IP Model :→

4 Application
3 Transport
2 Internet
1 LINK

Just like OSI model, its been numbered from bottom to up but the direction depends on if you're sending or receiving traffic.

This is the original model but it has been updated here is brand new model

5) Application
4) Transport
3) Network
2) Data Link
1) Physical

One extra layer and one renamed layer that is, the Link layer has been split into data link & physical layer
Internet has been renamed to Network

OSI Model

TCP/IP model

Application	Application layer	HTTP, FTP, SMTP
Presentation		
Session		
Transport	Transport	TCP, UDP
Network	Network	IP, Routers
Data Link	Data link	Ethernet, Switches
Physical layer	Physical layer	Cables, NIC

Encapsulation

Sender

Application
Transport
Network
Data link
Physical layer

Each layer will add its own bit of information this process is called Encapsulation

Decapsulation

Receiver

Application
Transport
Network
Data Link
Physical layer

when Receiving device gets the data it starts to decapsulate it. it first checks destination MAC address in that frame and if its for our computer, it processed further, then computer checks IP info of Packet and so on.

Data → here data is called Data only

TCP | Data → Once the transport info is added its called a Segment

IP | TCP | Data → Network layer info make segment a Packet

Ethernet | IP | TCP | Data | Ethernet → Once we add Physical addresses Packet becomes Frame

Head contain destination and source MAC address

It contain some error Check information

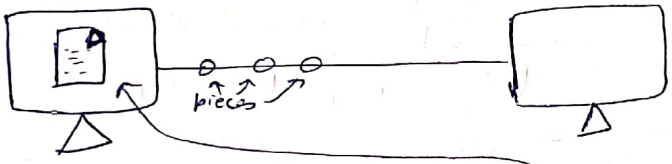
* It's right to note that at each layer data has different specific names

TCP vs UDP comparison :→

When sending data we work our way down in TCP/IP model
We'r going to ignore the application layer as it's largely out of our control and start at the transport layer.

④ Transport → The two primary protocols at this layer are TCP & UDP.
A Network application needs to choose that how to send the data, that choices comes down to do I want to send it reliably or do I want to send it unreliably.
Both have their benefits and drawbacks

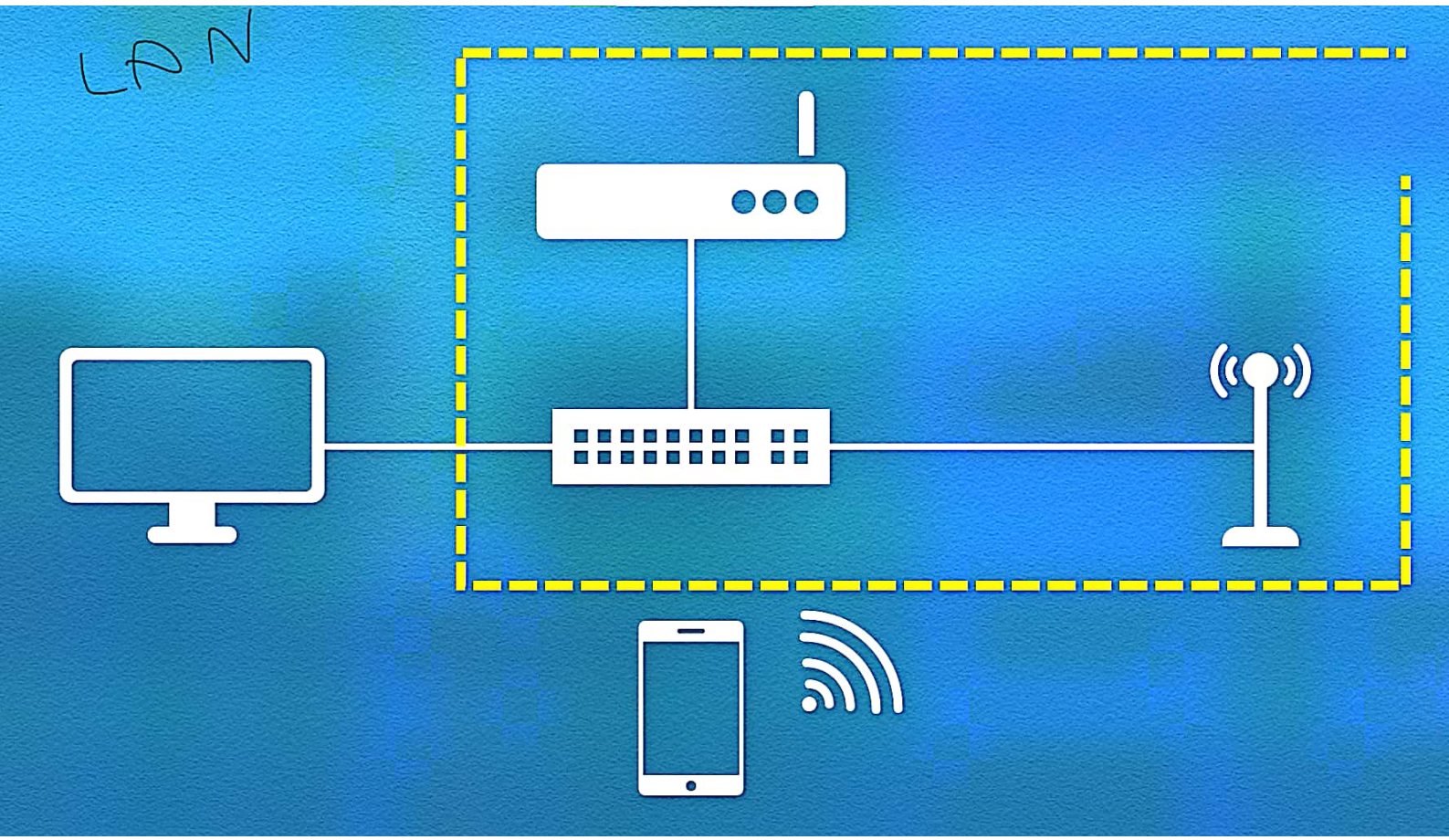
TCP (Transmission Control Protocol) :→

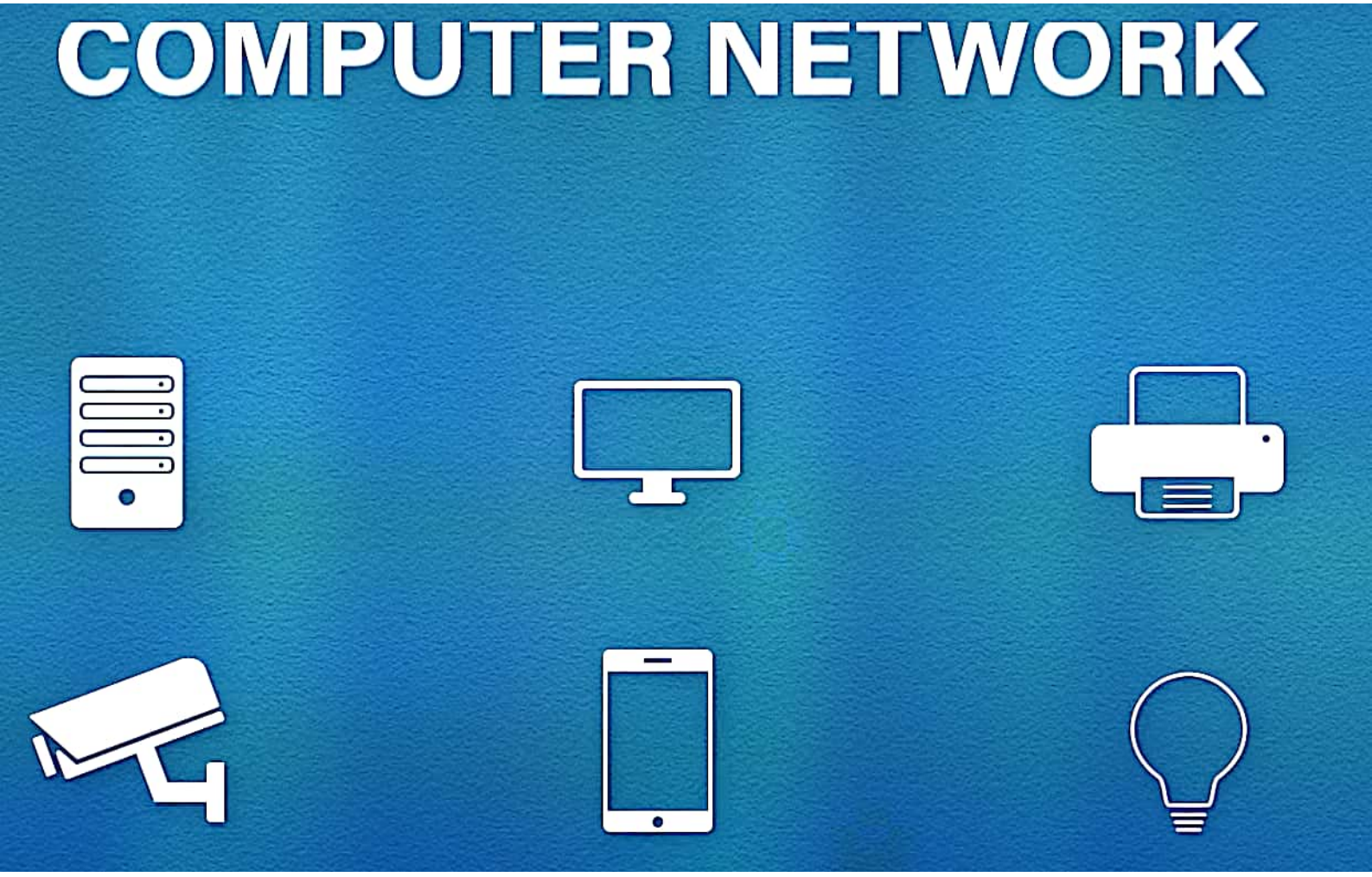


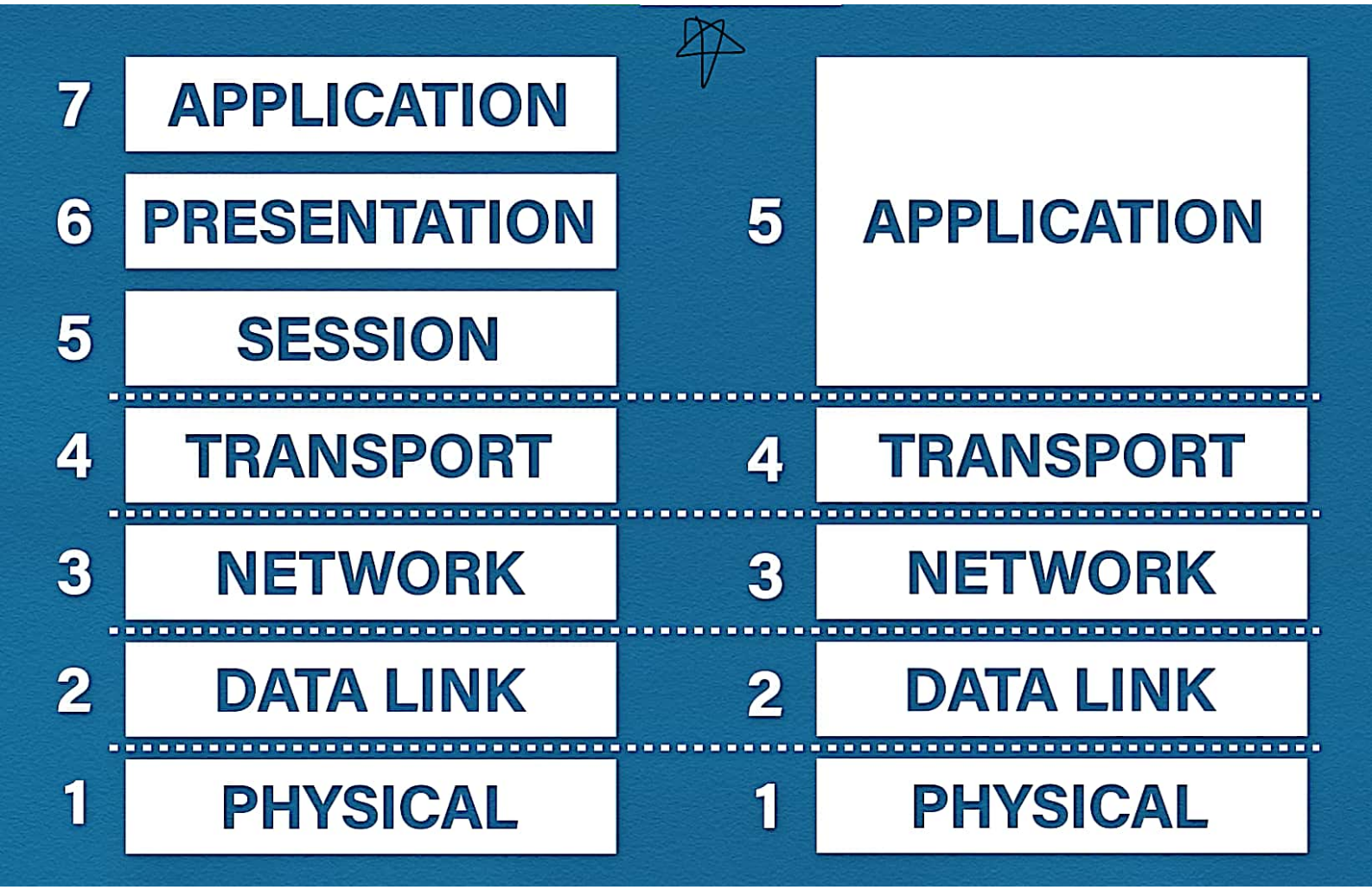
* When we send some data, this file for eg: is not sent all as one piece, its being sent in lots of different pieces of data
The receiving computer then gets these pieces, puts them together and then spits out the file. That's fine
but what if some of the data piece gets lost in the way.
The receiving computer won't be able to put pieces together and will end up with a corrupt file.

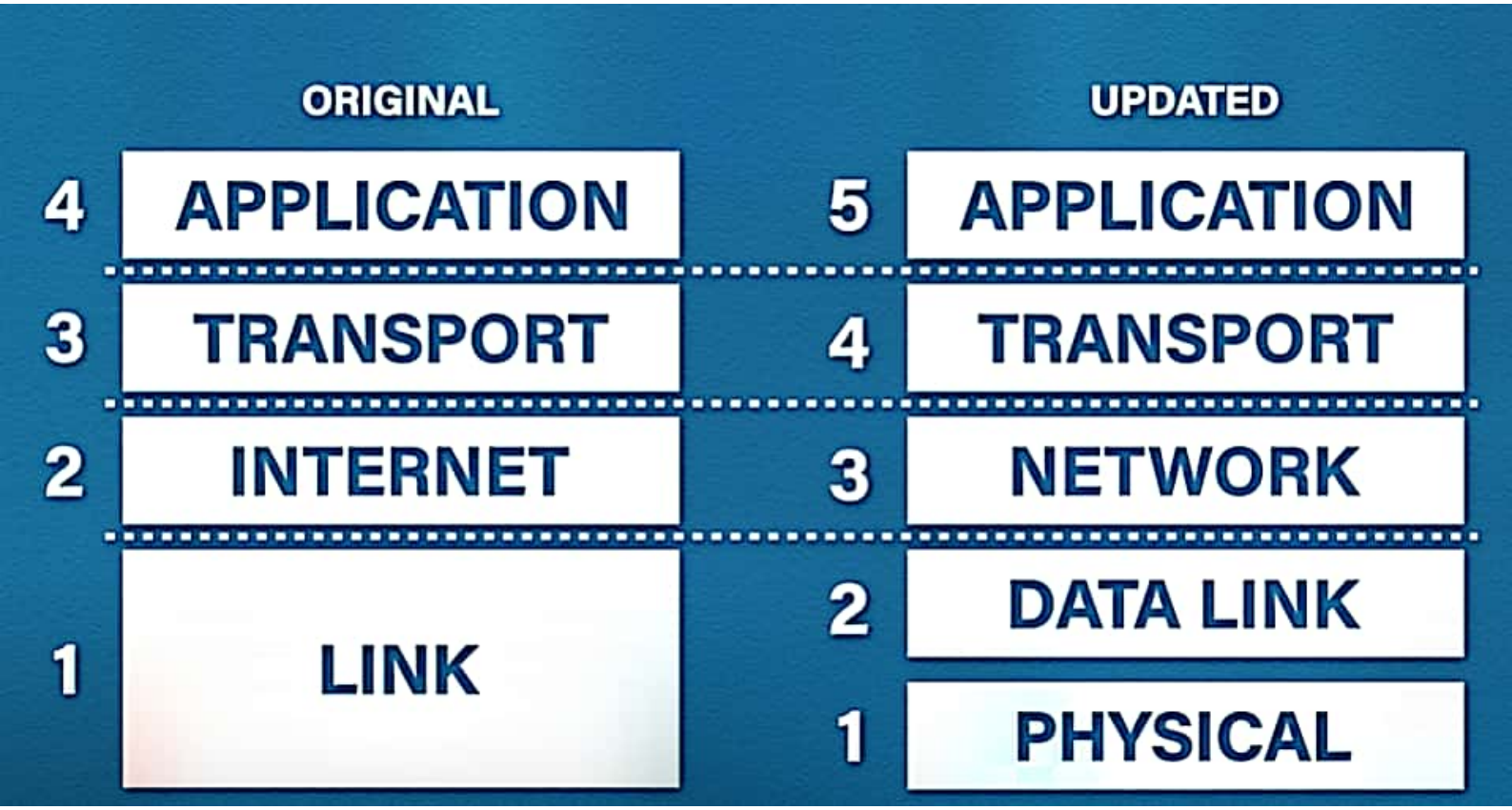
So this is the reason we need some method to reliably send the data, something that can resend or resolve missing or corrupted data. and this is what exactly TCP does.

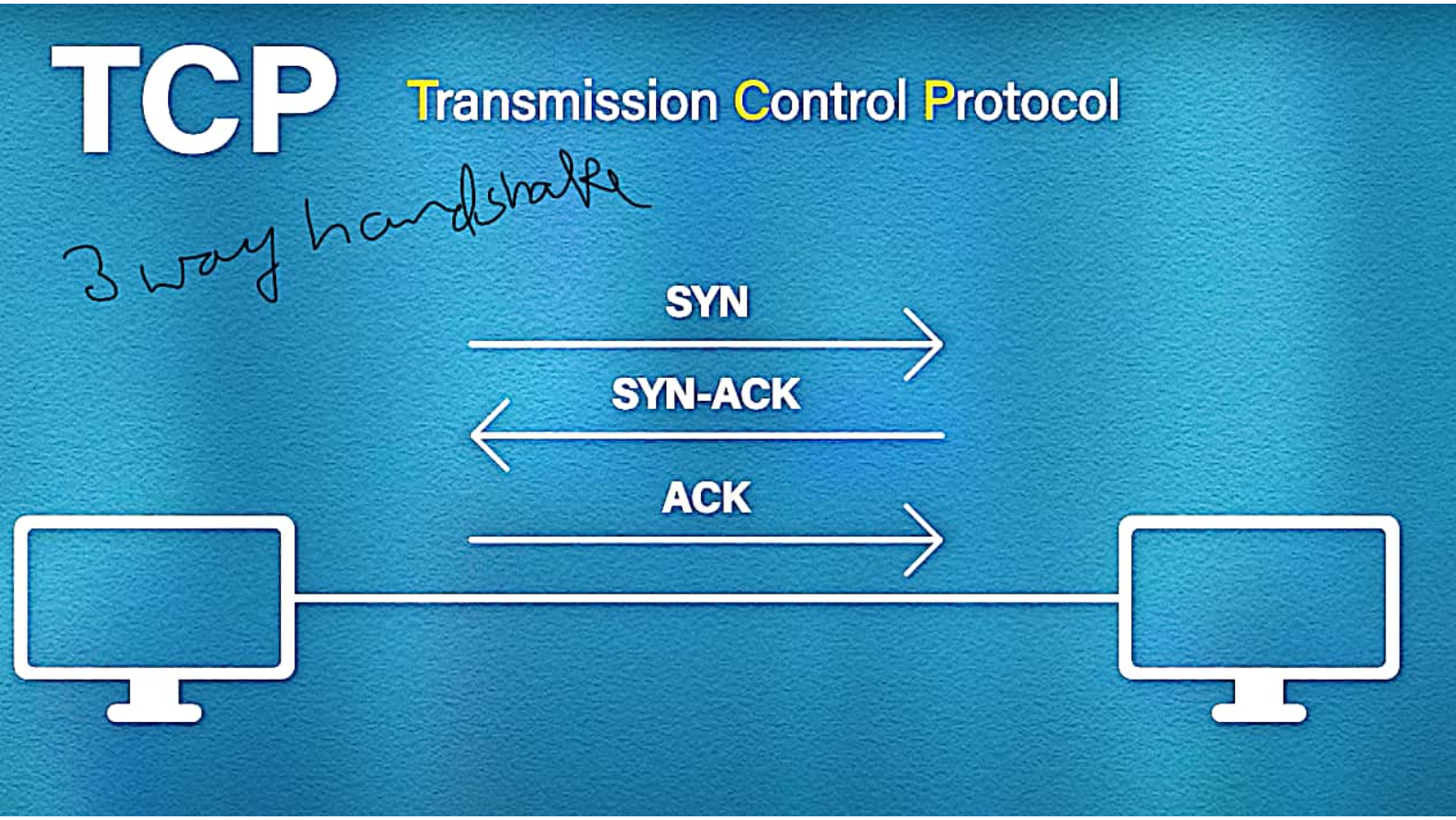
TCP is widely used as it can reliably send and receive data, If there is a network blip and some of the segments are lost along the way TCP can recover them, meaning user experience shouldn't be compromised with half loaded websites or corrupted document.


LAN


COMPUTER NETWORK


7
APPLICATION
6
PRESENTATION
5
SESSION
4
TRANSPORT
3
NETWORK
2
DATA LINK
1
PHYSICAL
5
APPLICATION
4
TRANSPORT
3
NETWORK
2
DATA LINK
1
PHYSICAL


ORIGINAL
UPDATED
4
APPLICATION
3
TRANSPORT
2
INTERNET
1
LINK
5
APPLICATION
4
TRANSPORT
3
NETWORK
2
DATA LINK
1
PHYSICAL


TCP
Transmission Control Protocol
3 way handshake
SYN
SYN-ACK
ACK

The way TCP can provide such a stable and reliable connection is in 3 ways :=>

1) It uses Acknowledgement numbers
2) It uses Sequence numbers
3) TCP adds a Checksum.

But before any of these could happen we need to start a reliable connection, TCP does this by what's called a Three-way handshake

Three-way handshake

↓

It looks something like this :=>

1st the sender computer sends a message called a SYN which is short for Synchronize, the receiving computer then replies with a ACK message, short of acknowledgement and it also includes a SYN message of its own. so this message together is called SYN-ACK

Lastly the sending computer Acknowledges with an ACK message

Now we have an open TCP connection.

A similar Process is followed when closing this connection

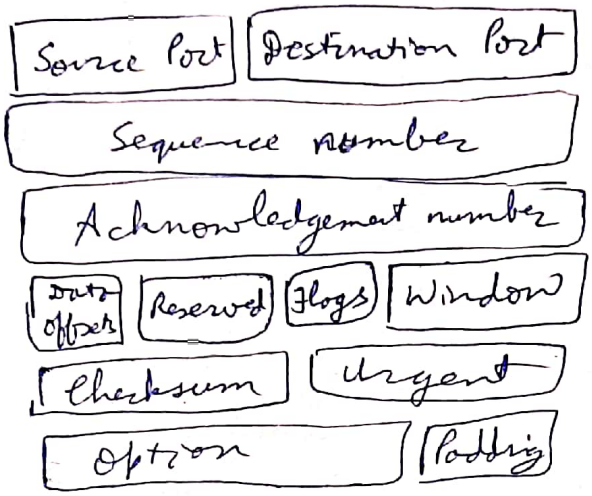
Sequence numbers & Acknowledgement numbers.

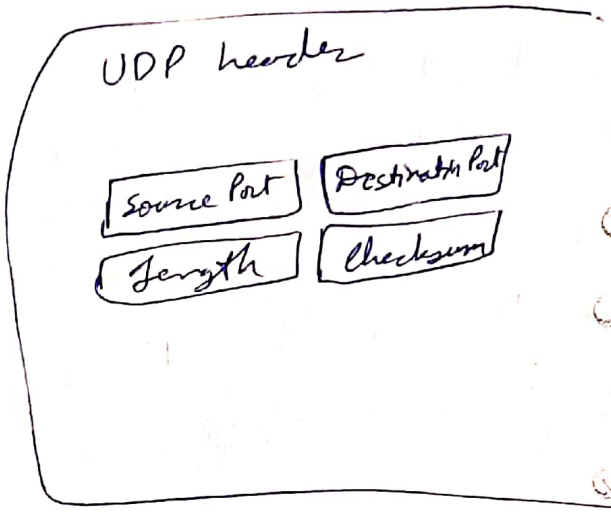
⊙ TCP assigns each segment with some number when they are sent. This way the receiving device can collect these segments, re-order them correctly and determine if any segments are missing. The sequence number are just one field in TCP header, these messages then Acknowledged i.e sending comp says sended this nos, receiver says got it ... and so on.

⊙ Checksum is a simple calculation against the data, If the data has been transmitted successfully, the checksum ran on receiving side should match that of sending side

⑥ If there is some problem along the way or some bits got messed up then there will be a Checksum mismatch and the segment will be discarded.

A TCP header is added to Application data to create a segment
This is what a TCP header look like :→





Unreliable communication comes in the form of UDP or User Datagram protocol. UDP has none of the error handling, sequencing or reliability of TCP.

you can think of UDP as machine gun, which is just firing and firing data to other side, not caring abt where the data is going, it has one goal i.e to send the data

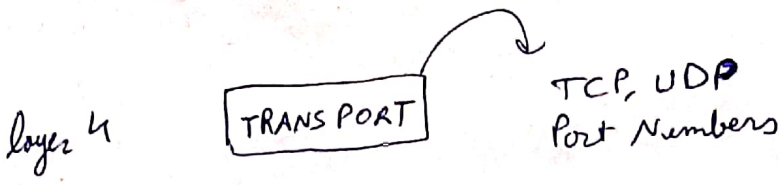
Q So Why do we ever use UDP if its so unreliable?

Ans → Well, TCP comes with reliability and great connection but it all comes at a price of resources and latency.
(lag) or delay in communication

TCP :→ This is great for most common task such as Web-browsing, file transfers where we don't mind the latency issue (time issue) in return for a stable connection.

UDP :→ Where UDP is useful is when we need live real-time connection, for eg: voice calls, video calls, gaming we can't afford latency in these situations.

Port Numbers =>



Q Why do we need port numbers

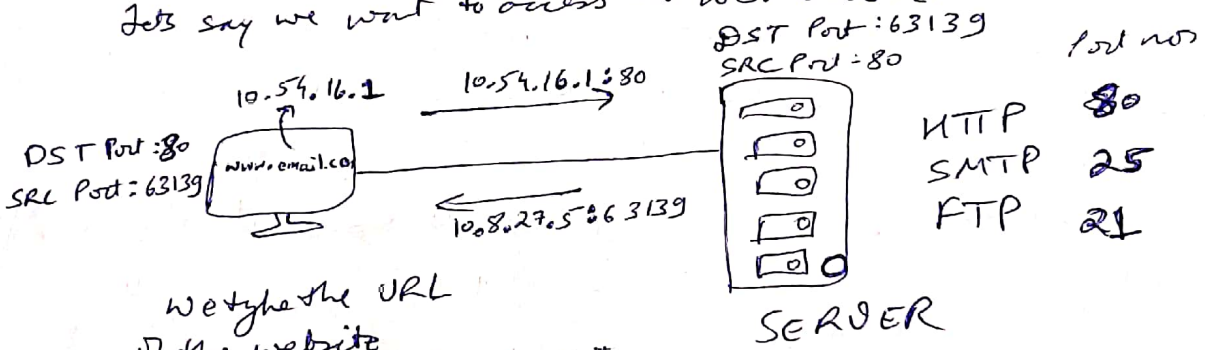
(Ans)

Let's say you live in this house, you probably want to receive post here, so what you do, you put a letter box outside your house so that when post comes it can get access to your house.

A port works in similar way.

Instead of a house there might be a server, and this server can be running mail, web or any other service we want to send data to

Let's say we want to access the web-server.



We type the URL of the website and the 1st thing the computer will do is, convert this URL into an IP address, this is done by using DNS

A Computer then sends the request to the web-server but this server might not just be hosting websites using HTTP, it may also be a mail server using SMTP or even a file server using FTP, so how does the server know which application to send the request to? Well these applications have something called a well known port number assigned to them. HTTP is assigned port number 80. SMTP is assigned port no 25, FTP is assigned port nos 20 and 21.

So Because the port nos are standard numbers all comp. know about it. When we made our web request our comp knew we were trying to access a http site, so it added the destination port nos 80 to the TCP header, the computer will also choose a randomly generated nos eg: 63139 as SRC to receive reply. Source Port nos

It sent the web servers IP address and the well known port for that service. The IP address and port numbers are often written like this

10.54.16.1 : 80

↑ IP address : ↑ Port nos (destination)

The server would then receive this request, look at the destination port number, realise the request is for the web and pass it to that application. The server would then respond. This time the port numbers are reversed, the DST (Destination) port is the randomly generated nos and the source port is our well-known port number 80. ~~The first~~ Again when we receive this response our computer looks at the port number, the port nos lets our computer know which application to send it to and in this case our browser to display it on and even which tab of that browser to display on.

So its the IP address which gets the data to the computer but its the Port Number that get's the data to the right application. This all is just a brief. There is much more complex process which runs in the background.

- Port nos →
  0 to 1023 are well-known Port nos

It doesn't mean you need to remember them all but you should the ones used everyday.

- Registered Port nos which companies have registered :→
  1024 - 49151

- Dynamically assigned Port numbers (These are used for local computers)
  49152 to 65535

Number System :- It's a way to represent nos.
The value remains same but the representation changes.

* Upto now the nos system which we had use is decimal number system
i.e from 0 to 9

① Decimal Nos system (Base 10) → nos of digits are 10 from 0 to 9
(0 to 9)

② Binary Nos system (Base 2)
(0 and 1)

③ Octal nos system (Base 8)
0 to 7

④ Hexadecimal (Base 16)
(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)

Binary to Octal (2 → 3 column)

Octal is 8

8 / 4
2

put 4 zeros and 4 ones in 1st column
* start with 0

4 / 2 → 2
2

now put 2 zeros and 2 ones combination in second column
* start with 0

2 / 1 → 1
1

now put 1 zero and 1 one combination

table

Octal			
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

~~# Octal to Binary~~

Q) Convert
eg: $(110010110 \cdot 10101)_2$ to Octal

626

010 ← this zero is place by us to satisfy group of 3.

. 52

So 626.52 in Octal

eg: (10 10) convert to octal

010

001

make group of 3 but extra zeros if some group is not satisfying

now look in the table

010 = 2

001 = 1

* So nos is 12

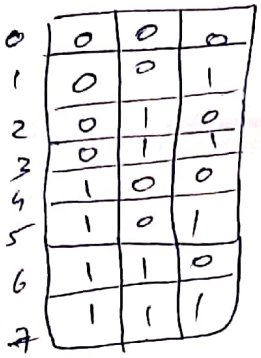
Octal to Binary

Q) Convert $(67)_8$ to Binary ?

Check the table

6 7

110 111



$(110111)_2$

Q) Convert $(67.32)_8$ to Binary ?

Ans) $(110\ 111 . 011\ 010)_2$

Hexadecimal to Binary

See table and get answer

Q) $(EF2)_{16}$ to Binary ?

Ans)

E → 1110
F → 1111
2 → 0010

So

$(1110\ 1111\ 0010)_2$

Q) Convert $(F2.D)_{16}$ to Binary?

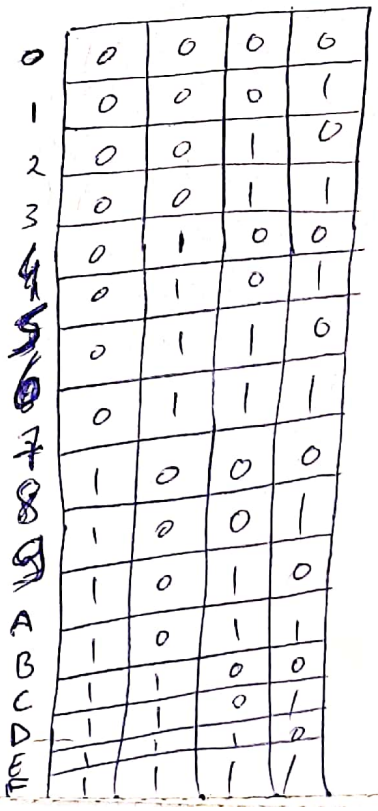
F → 1111
2 → 0010
D → 1101

So

$(1111\ 0010.1101)_2$

Binary to Hexadecimal

16 is 2^4 → 4 columns in table



Q) Convert $(11010110)_2$ to Hex.

Ans) Now hexadecimal is 2^4

so we will make groups of 4 and find answer using table

1101 0110

D 6

* Always make group from left side

» D6

Q) Convert $(11010110.1101)_2$ to Hexadecimal

D6.D

Conversion from Octal to Hexa
$( )_8$ to $( )_{16}$

Q) Convert $(26)_8$ in Hexadecimal nos.

Ans) Here also we can use the table i.e. 1st we can convert $(26)_8$ to $( )_2$ (Binary) and then (Binary)$_2$ to Hexadecimal.

$11 = 2^{④}$ → group of 4

Octal-Binary table

0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

8/2 → 4, 4/2 → 2, 2/2 → 1

$(26)_8 = (010\ 110)_2 = (\ )_{16}$

To complete grp. group of 4

0 0 0 1 0 1 1 0

1 6

$(16)_{16}$

Binary-Hexadecimal table

0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
A	1	0	1	0
B	1	0	1	1
C	1	1	0	0
D	1	1	0	1
E	1	1	1	0
F	1	1	1	1

16/2 → 8, 8/2 → 4, 4/2 → 2, 2/2 = 1

Conversion from Hexa to Octal.
$( )_{16}$ to $( )_8$

So 1st convert $( )_{16}$ to $( )_2$ then to $( )_8$ (Simple 😊)

Q) Convert $(AE)_{16}$ to Octal

$2^{③}$ → grp of 3

$(AE)_{16} = (\ )_2 = (\ )_8$

A → 1010
E → 1110

so $(10101110)_2 = (256)_8$

0 1 0 1 0 1 1 1 0

2 5 6

Conversion from Decimal to Binary
to Octal
to Hexadecimal.

* when we convert from Decimal we divide
* to decimal we multiply

Q) Convert $(16)_{10}$ to Binary (base 2)?

Ans) So whenever we convert some number from Decimal to that nos we just divide our nos with base of the system in which it needs to be converted.

		remainder
2	16	0
2	8	0
2	4	0
2	2	0
2	1	1

as soon as we get 1 we stop

So

$(16)_{10} = (10000)_2$

we can also cross check

10000

$2^4 \times 1 + 2^3 \times 0 + 2^2 \times 0 + 2^1 \times 0 + 2^0 \times 0$

$16 + 0 + 0 + 0 + 0$

$(16)_{10}$

Convert $(16)_{10}$ to Hexadecimal (base 16)

16	16	0
	1	

$(16)_{10} = (10)_{16}$

Q) Convert $(16)_{10}$ to Octal ? base 8

Ans)

8	16	0
	2	

$(16)_{10} = (20)_8$

cross check

$(20)_8$

$8^1 \times 2 + 8^0 \times 0$

$(16)_{10}$

Convert $(256)_{10}$ to Hexadecimal

16	256	0
16	16	0
	1	

$(256)_{10} = (100)_{16}$

Conversion from Decimal to Binary
Octal
Hexadecimal
and vice versa

Q) Convert $(.25)_{10}$ to Binary?

Ans ⇒

$(.25)_{10}$

Octal

$0.25 \times 2 = 0.50$

$0.50 \times 2 = 1.00$

Integer	Fraction
0	.50
1	00

Step when we get zeros in fraction part, or repetition of itself, or if its not coming find till 5 times.

$(0.25)_{10} = (.01)_2$

Q) Convert $(85.42)_{10}$ to Binary?

Ans ⇒

2	85	1
2	42	0
2	21	1
2	10	0
2	5	1
2	2	0
	1	

$(1010101.011010111000010101$

	Int	Frac
.42 × 2	0	.84
.84 × 2	1	.68
.68 × 2	1	.36
.36 × 2	0	.72
.72 × 2	1	.44
.44 × 2	0	.88
.88 × 2	1	.76
.76 × 2	1	.52
.52 × 2	1	.04
.04 × 2	0	.08
.08 × 2	0	.16
.16 × 2	0	.32
.32 × 2	0	.64
.64 × 2	1	.28
.28 × 2	0	.56
.56 × 2	1	.12
.12 × 2	0	.24
.24 × 2	0	.48
.48 × 2	0	.96
.96 × 2		.92

Network layer

To see how the data actually gets to that location.
This layer is responsible for IP address.

Q So what is an IP address
Ans ⇒ A unique identifier assigned to each device connected to a computer Network.

Each computer in a network needs to have a unique address called as an IP address, Given below is ipv4 address

192 . 168 . 32 . 152 ← Each section is called a Octet.
These octet's are seperated by dots or periods.

32-bit in length
which means it contains 32 binary digits.

Each Octet can contain any nos between 0 to 255

Q Why 255?
Ans ⇒ 255 is the largest no that can be made with 8 bits.

Number System Rule
from decimal so divide

2	192	0
2	96	0
2	48	0
2	24	0
2	12	0
2	6	0
2	3	1
	1	

2	168	0
2	84	0
2	42	0
2	21	1
2	10	0
2	5	1
2	2	0
	1	

2	32	0
2	16	0
2	8	0
2	4	0
2	2	0
	1	

2	152	0
2	76	0
2	38	0
2	19	1
2	9	1
2	4	0
2	2	0
	1	

11000000 . 10101000 . 00000 . 10011000

11 2 extra zero's before just to complete octate and they are before nos do wont change its value

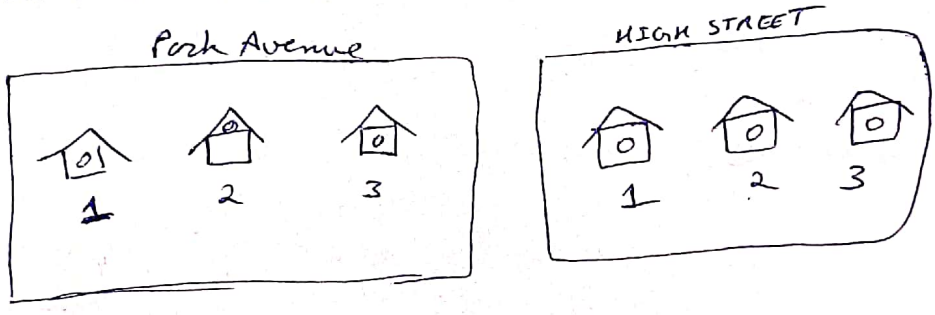
IP address is also made from two parts

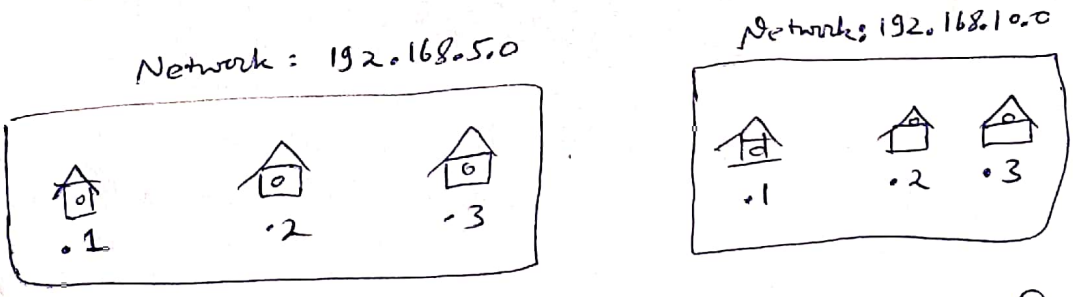
① Network ② Host

A subnet mask is always paired with an IP address and is used to identify the network section and host section

In the simplest form, whenever we see 255, it's the network part of the address.
and whenever you see 0, this is the host part of the address.

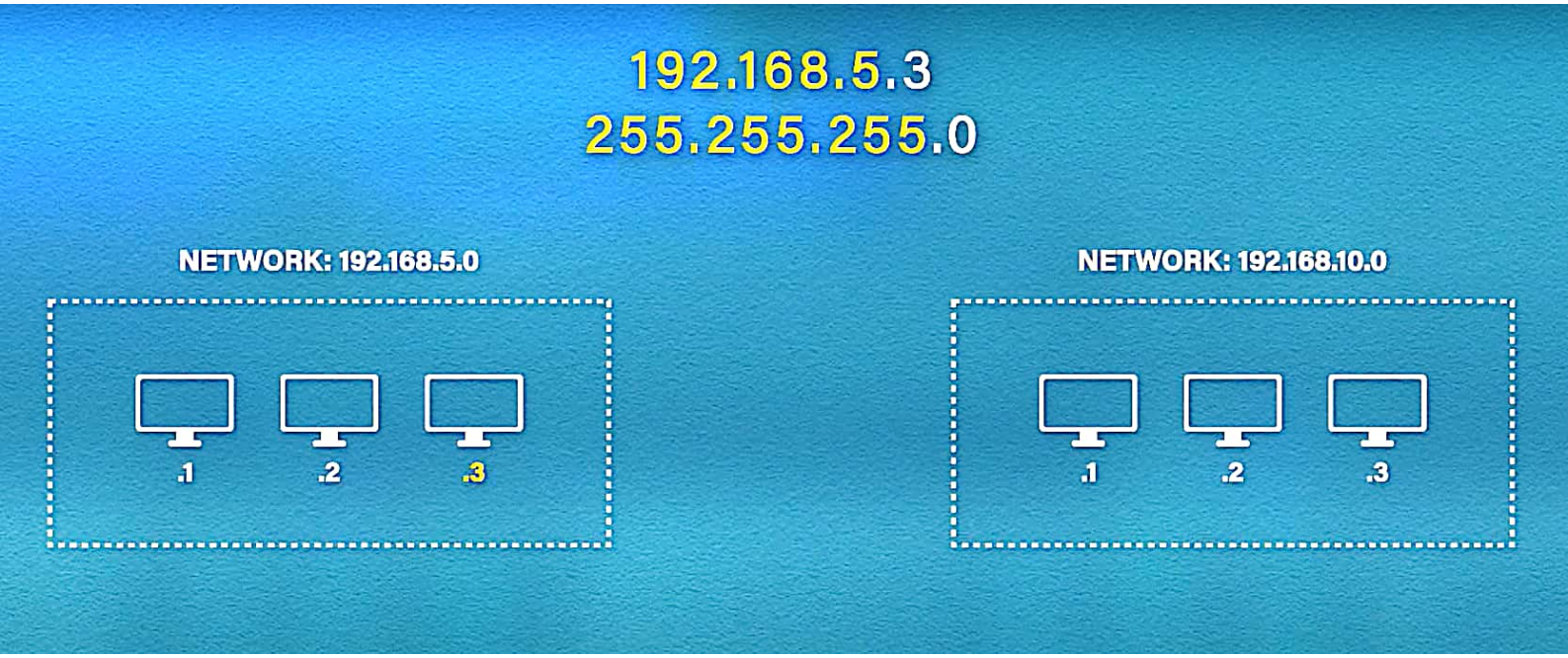
eg: You share the same street names as your neighbours but, it's the house nos which make your address unique.
On the other hand we can have multiple streets with same house nos and this is the same in computer Networks too.





In Computer Networks, instead of street address we have network nos. and instead of house nos we have host numbers.
here in the above diagram both have a subnet of 255.255.255.0
Inside our network we have host .1, .2, .3
Inorder these IP addresses would be

192.168.5.1 , 192.168.5.2 ... so on.


192.168.5.3
255.255.255.0
NETWORK: 192.168.5.0
.1
.2
.3
NETWORK: 192.168.10.0
.1
.2
.3

FTP Data	TCP	20
FTP Control	TCP	21
SSH	TCP	22
Telnet	TCP	23
SMTP	TCP	25
DNS	TCP/UDP	53
DHCP	UDP	67, 68
TFTP	UDP	69
HTTP	TCP	80
HTTPS	TCP	443
POP3	TCP	110
SNMP	UDP	161

Well-Known Port Nos

Back in early days of IT's, it was decided to split all of the available addresses into groups and these group's were called classes.

The idea was to make address allocation scalable.

The main classes are

Class A, Class B and Class C

There is also Class D for something called Multicast addresses and Class E which is reserved for experimental use.

A	1.0.0.0 to 126.255.255.255 Subnet: 255.0.0.0	Hosts: 16,777,214
B	128.0.0.0 to 191.255.255.255 Subnet: 255.255.0.0	Hosts: 65,534
C	192.0.0.0 to 223.255.255.255 Subnet: 255.255.255.0	Hosts = 254

This was to control the number of host available on each network.

Class A has 3 octets available for host allocation.
massive massive network.

Class B has 2 available octets for host allocation.
Still a Big nos

Class C has one available octet for host allocation.
i.e small network

Due to massive expansion of computer and internet there are no more IPv4 address left. This is why new IP version 6 has been designed, it will provide enough number of IP addresses to us. But fear not, there is a "Sol" to prolong life of IP version 4, the "Sol" is to carve out small sections from all classes. and call them private IP addresses. All other addresses are known as public address. They still use the same subnet mask for that class, and they can still have some nos of Hosts

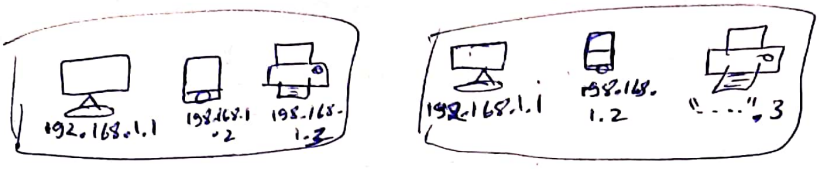
	PUBLIC	PRIVATE
A	1.0.0.0 - 126.255.255.255 SUBNET: 255.0.0.0	10.0.0.0 - 10.255.255.255
B	128.0.0.0 - 191.255.255.255 SUBNET: 255.255.0.0	172.16.0.0 - 172.31.255.255
C	192.0.0.0 - 223.255.255.255 SUBNET: 255.255.255.0	192.168.0.0 - 192.168.255.255

#

The difference is where public IP addresses need to unique, private address can be used over and over and over again, thus saving millions of billions of Public IP address.

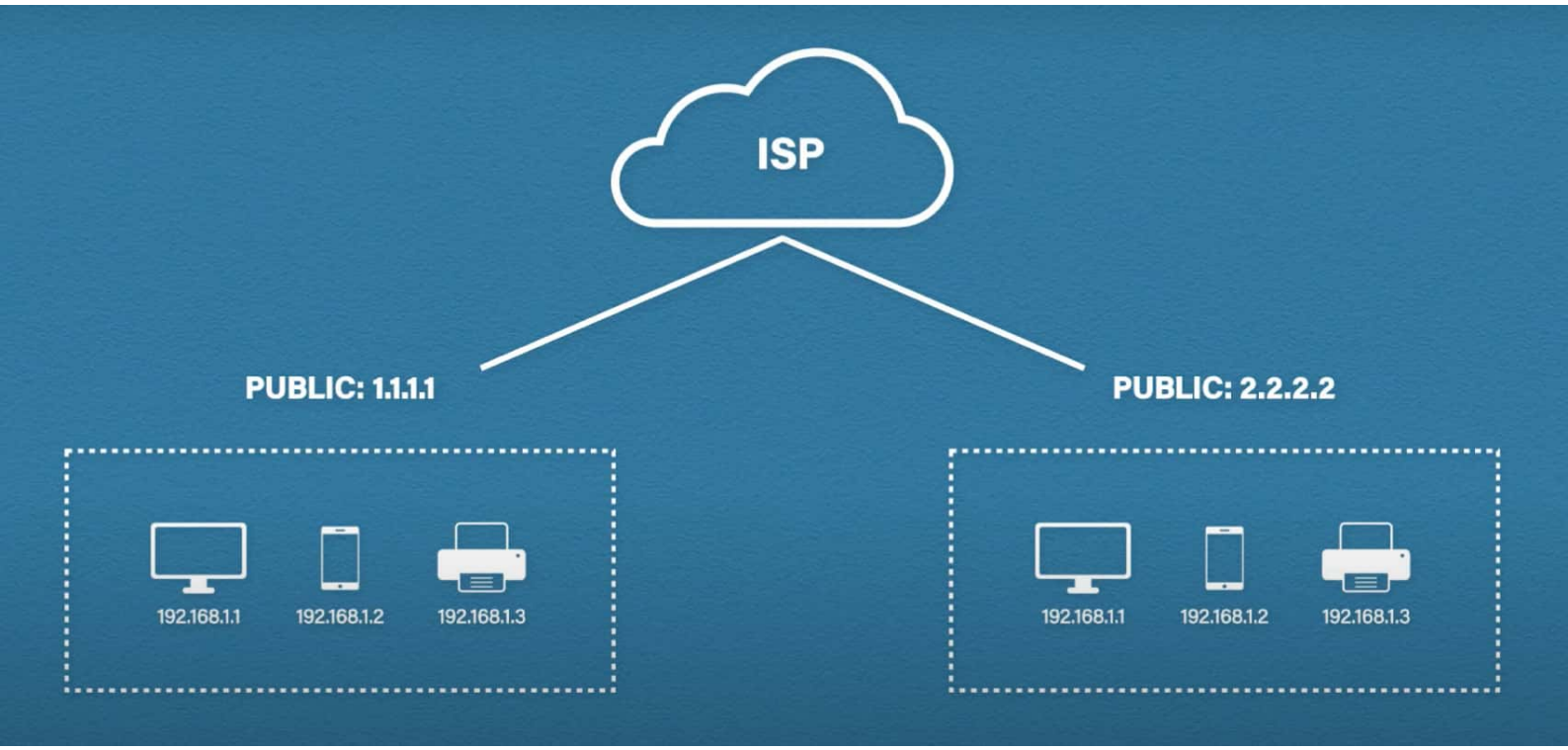
Private IP address can't be used over the internet.

Private IP address should be unique in your own network



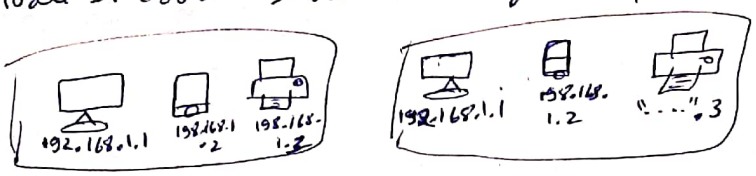
So Only Public IP address could be used over the internet. These must be unique

when you sign up with your ISP (Internet Service Provider) they will issue us with a public IP address, that we can use over the internet.


ISP
PUBLIC: 1.1.1.1
PUBLIC: 2.2.2.2
192.168.1.1
192.168.1.2
192.168.1.3
192.168.1.1
192.168.1.2
192.168.1.3

The difference is where public IP addresses need to unique, private address can be used over and over and over again, thus saving millions of billions of Public IP address.

* Private IP address can't be used over the internet.
* Private IP address should be unique in your own network



* So Only Public IP address could be used over the internet. These must be unique

when you sign up with your ISP (Internet Service Provider) they will issue us with a public IP address, that we can use over the internet.

* Previously we said, wherever there's 255 that will be network section. and wherever there is is a Zero that's the host section.

eg. ⇒ 192.54.103.29

It class C so subnet mask is

255.255.255.0

compare so
network ID = 192.54.103.0
Host ID = 0.0.0.29

in IP address when comparing with subnet mask

but the real way to check which is Network ID and which is Host, is by looking at binary bits of subnet mask so anytime you see one (1) that's network section and anytime we see zero that's a host section.

eg. 192.54.103.29
11000000 00110110 01100111 00011101

11111111 11111111 11111111 00000000
255.255.255.0

now instead of a full subnet mask we'll often see it written with a forward slash and then the no. of network bits

eg: 192.54.103.29/24

Let's take a closer look at host section of 192.54.103.29/24

29
000 1110

It consists of 8 bits, so how many host address it can have with 8 bits.

So we can put nos from 0 - 255 here
i.e total nos of 256 possible combinations

We can't use the 1st and the last IP address to host as
i.e 192.54.103.0 → 1st one is network address
and 192.54.103.255 → second one is broadcast address.
they both are reserved as

Broadcast message is used to send messages to all computers in this network.

So we only have 254 usable hosts

#IPv6 Address

Due to complete allocation of IPv4, a new version is introduced and now it can support a huge huge huge amount of different IP address.

Apart from this, it brings a lot of improvement, it's efficient & practical.

eg: 2001:0db8:0000:0000:a111:b222:c333:abcd

This is IPv6 address but don't worry, we'll understand it also :->

1st lets do some comparisons with IPv4.

192.168.32.152

IPv4	IPv6
• It's 32 bits long	It 128 bits long
• It has 4 sections called as octet	It has 8 sections called as Hextets
• It uses dots to seperate octets	• It uses colon to seperate Hextets
• It contains decimal nos in it.	• It contains hexadecimal nos in it

How IPv6 works when it comes to binary.

⇒ Each hexadecimal (base 16) (2^4) is made up of 4 bits

So

	2^3	2^2	2^1	2^0
	8	4	2	1
2 →	0	0	1	0
8 →	1	0	0	0

So

2001 : 0db8 : 0000 : 0000 : a111 : b222 : c333 : abcd

0010 0000 1000

Each hextet is 16 bits long. Each character and is 4 bits long.

So d = 13

	8	4	2	1
13 →	1	1	0	1

So

d = 1101 in binary

1	
2	
3	
4	
5	
6	
7	
8	
9	
10	← a
11	← b
12	← c
13	← d
14	← e
15	← f

Now just like IPv4, IPv6 has Network Section & host Section.

Rectification The way IPv4 does this is by using a Subnet mask

However IPv6 has gotten rid of the Subnet Mask and instead it just uses a forward Slash and nos of network bits

2001 : 0db8 : 0000 : 0000 : a111 : b222 : c333 : abcd /64

This is the Subnet Mask.

How to make this long hexadecimal address little shorter and easier to write :→

Ans)

1st) Remove continues zero's, if any, in multiple hextets are adjacent.

eg: 2001 : 0db8 : 0000 : 0000 : a111 : b222 : 0000 : abcd

Replace them with a double colon. (::)

So

2001 : 0db8 :: a111 : b222 : 0000 : abcd

Computer see's it and realizes there are two missing hextets and hence it replaces them by zero's on its own.

* The rule is we can only use it once,

2nd) We take all leading zeros from each Hextet

2001 : 0db8 :: a111 : b222 : 0000 : abcd

and remove them.

2001 : db8 :: a111 : b222 : 0 : abcd

now computer see's it and realizes there is a missing character so, it adds a zero at missing place on itself.

Note : Even if its all zero's to because we already use double colon, we need to leave the last zero.

Now, its the address we're left with, still pretty large but its an improvement to old address hahaha.