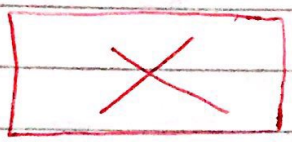
PAGE 24th. Jan.
DATE

Hardware
- host = end systems : PC, mobiles, servers, any device connected to the internet.
- Communication links.
- Packet switching devices.
  ↳ devices receives units of data (packets) and determines in which direction the packet should be sent to deliver it to the final destination.

- ISP: Internet Service Provider
  ↳ a way to make you connected to the internet.
  ↳ is nothing but set of packet switching devices highly connected to each other "which means not only one one two wires between packets".

Router



Switch.

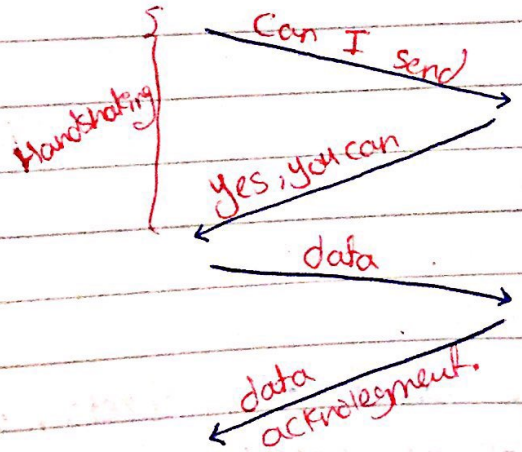
Network
* Protocol : set of rules that define format of packets, what is the meaning of each part of the packet, what to do when packet received.

- In networks and internet, there are set of standards each standard is written in a document.
"RFC"

* Infrastructure: end systems, servers, packet switching, devices communication links, protocols & standards.

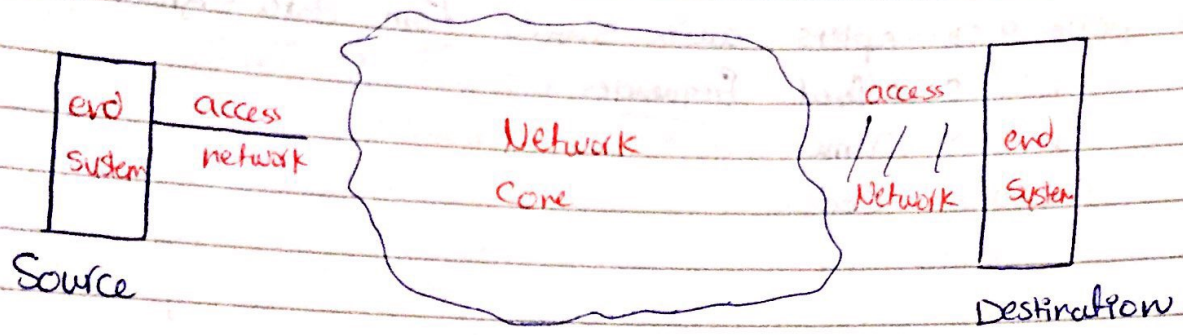
- I need programming interface to encapsulate and hide all the details below.
  ↳ e.g. Socket programming
  RMI
  RPC

* TCP is connection oriented protocol:
  ↳ before sending any data, a connection should be opened.



* Server: any device that provides a service.

* Network edge : end systems
* access network : the first connection to the internet.
* network core : packet switching devices.



* Physical media : the medium through which the data is propagated
  - ↳ guided : the data is propagated in physical mediam
  - ↳ non-guided : the data is propagated in the air.

* Bandwidth : data rate (bits per second).

- Shared : you take at most the claimed bandwidth, but if there are other endsystems using the internet then the bandwidth decreases.

- dedicated : you take all the claimed bandwidth.

PAGE 27th. Jan.
DATE

* ADSL : Asymmetric Digital Subscriber Liner
  - ↳ ISP: Telephon network.
  - ↳ Asymmetric: Upload < Download.

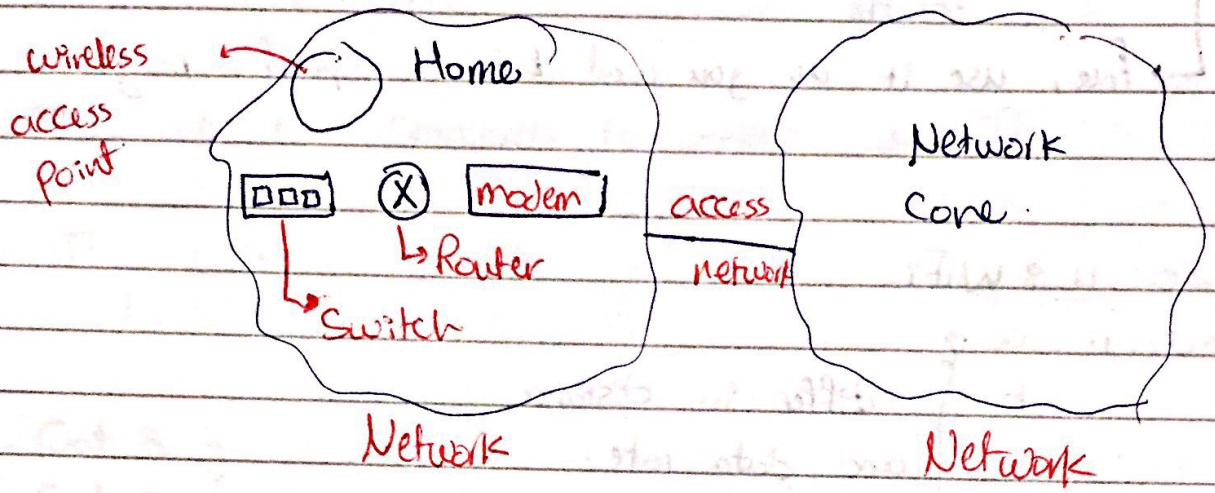
→ One Line is connected to the home
- Splitter : it splits audio signals from data signal
  - FDM : different frequencies
  - TDM : Time
  - CDM : Code.

- Home access Networks.
  1) ADSL : Tele communication
  2) Cable :
     - ↳ ISP: TV companies.
     - ↳ Coaxial : Multiplexing : data from video.
     - HFC : hybrid fiber coaxial
     - ↳ Assymetric.
  3) 3G/4G
  4) Satellite.

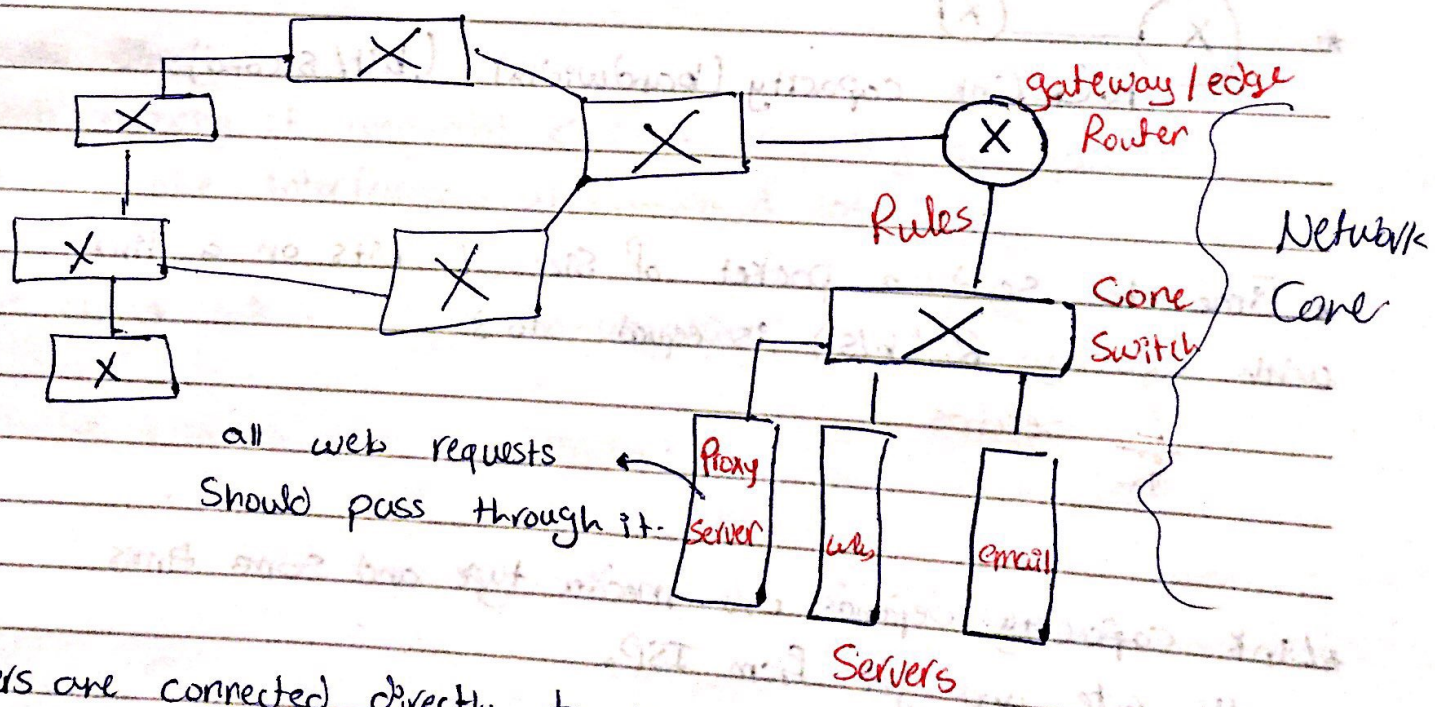
- modem : modulation/Demodulation
  - ↳ carry data on frequency.

- Router : is a packet switching device that connects 2 networks or more

- Switch: to provide ethernet connection
- Wireless Access point: to provide wireless connection

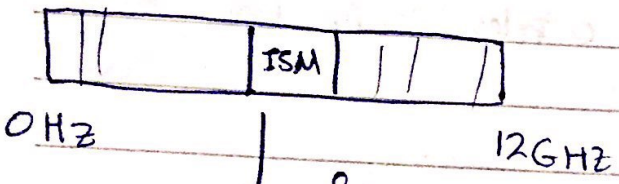


* Gateway: is the router that connects a network to the internet.



- Servers are connected directly to the gateway server to achieve speed/ security and desired rules.

- Wireless spectrum.

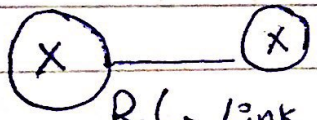


→ free, use it as you want but in specific range

- IEEE 802.11 & WiFi

802.11 a
b
g
N

differ in distance and data rate.

* 

R → Link capacity (bandwidth) (bit/second).

* Time to send a packet of size L bits on a link with capacity R (b/s) is equal to

$\frac{L}{R}$ seconds.

* Link capacity depends on media type and some times on the rate you get from ISP.

PAGE 29th. Jan.
DATE

* Twisted pair :- (TP)
4 paired colored.
أكثر انتشار.
- Orange, white orange, green, white green, blue, white blue. Brown, white brown.

- There are two standards to connect the TP "RJ45"

- TP → UTP : Unshielded TP
  ↳ STP : Sheilded TP

- Cat 3
  Cat 5
  Cat 6
  } they differ in link capacity.
  Cat 3 < cat 5 < cat 6.

* Link capacity of unguided < Link capacity of guided due to noise, interference, attenuation & loss.

- bit error rate is bigger in unguided.

- Unguided → in the air. (no cables).

PAGE
DATE

* Network Core :-
  ↳ is an interconnected packet switching devices (Routers)

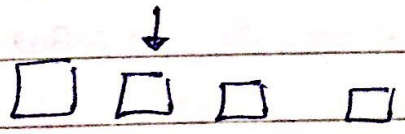
- Two main types of network core
  ↳ Packet switching
  ↳ Circuit switching.

* in network layers, each layer adds a header for the packet.

Headers (H)

- Packet switching core :
  ↳ it divides the big message into smaller size packets and each is sent independently.

- When someone wants to send an email that has an attachment of 1 GB size



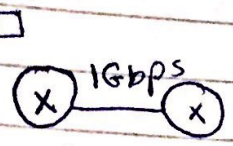
Packets.

1000 packets of 1 MB each, 1000 H will be added.

* In Packet switching network core, each packet is sent in full link capacity. in the router.

- In Circuit switching network core, the message is sent not in full link capacity.

1Mps



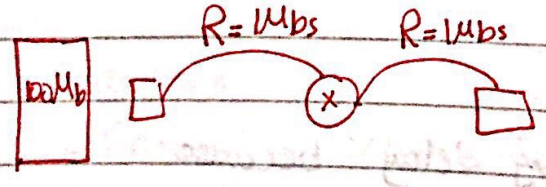
↳ in packet switching, it takes time $= \frac{1 * 10^6}{1 * 10^9} = 1ms$.

- When the message is divided into smaller packets 8 -
  ↳ added headers (disadvantage)
  ↳ on error, smaller re-transmission (advantage)

* All routers use Store & forward technique
  ↳ don't forward the packet except after fully recieving it.

* #hops = # transmissions.

- Ex if a path that is composed of 2 hops with link capacities 1Mbps. You want to send a file of size 100 Mb. Assume the header size = 100 b
What is the total transmission delay if you send the file as one packet and as 100 packet?



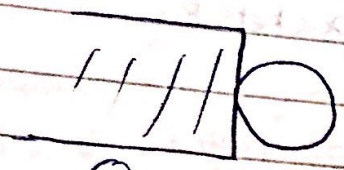
$\frac{100 * 10^6 + 100}{1 * 10^6} + \frac{100 * 10^6 + 100}{1 * 10^6}$ = Time to send the file as one packet.

- to send the file as 100 packets:-

Packet size = $\frac{100 * 10^6}{100} + 100 = 10^6 + 100$.

Time = $\frac{10^6 + 100}{10^6} + 100 * \left(\frac{10^6 + 100}{10^6}\right) \rightarrow$ أسرع.

*



Queue : first in first out.
↳ in the RAM.

- RAM can afford limited number of packets.

- Queuing delay at router:-
  ↳ Because sometimes the router is busy in forwarding other packets, therefor the packet waits in the queue.

- when the queue is full, new incoming packets are lost.

- the internet is packet switching core, so no guantee on delay or loss.

- No body can determine the queuing delay because it depends on the traffic at the router.

PAGE 31st. Jan.
DATE

- Router has 2 main function in packet switching :-

1- Routing : is a planning phase

2- Forwarding : work phase.

The result of routing is routing table

Destination address	interface #

in forwarding, when the router receives a packet it checks the routing table to see where to forward it and forwards the packets.

* Circuit Switching :-

↳ before sending any message, the source and the intermediate must setup a connection & reserve bandwidth for this connection.

- the connection along the reserved bandwidth is called circuit.

- the source & routers do not send in full link capacity.

↳ Features :-

- Connection setup (disadv)
- no queuing (Adv)
- Can afford less # of users (because of limited bandwidth) (disadv)
- No sharing of reserved resources (disadv).

- In general, packet switching is better than ckt switching.

* Slide 30

each user needs 100 Kbps

each user is active 10% of the time

when will circuit sw be better than packet sw?

Ans when 10 or more users are active

$$P(>10 \text{ active}) = 1 - P(<10 \text{ active})$$

$$= 1 - \sum_{i=0}^{9} P(i \text{ users is active}).$$

$$= 1 - \sum_{i=0}^{9} \binom{35}{i} * (0.1)^{i} * (35)^{35-i}$$

- Internet & Interconnected network

PAGE
DATE

* Performance metrics:-

↳ ① delay: the time from packet transmission until packet delivery (in seconds)

delay = Σ delay on each hop (link)

- delay on each hop.

↳ Processing delay: reading the headers and do some processing to forward the packet (like checking errors & destination).

↳ very small, because it depends on the processor

↳ Propagation delay: time for one bit to be delivered between two nodes on one link.

$$\left(\frac{\text{distance}}{\text{Speed of propagation}}\right)$$ (very small).

سرعة الضوء

↳ transmission delay: time to send one packet on one hop (push out time)

$$\left(\frac{\text{Size of packet}}{\text{Link capacity}}\right) \left(\frac{L}{R}\right)$$

↳ dependent on link capacity.

↳ Queuing delay: When the link is full (how busy the network is).

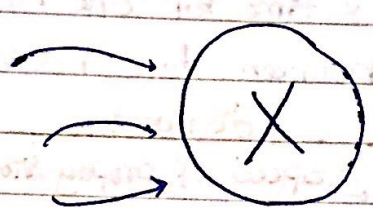
↳ ② throughput: # of packets delievered per second.

↳ ③ loss: # of packets lost.

- two packets with same size, same source & same destination but they may differ in delay, why?
  - they may have different paths.
    ↳ what if they follow the same path?
      - Queuing delay makes the delay different.

* Queuing delay
  ↳ dependent on how congested is the network.

- Arrival packet rate (a) bits/sec



and all packets to be sent on a link with capacity R bits/sec.

- each packet of size (L) bits/packets

$$\text{Average Arrival Rate} = a\left(\frac{\text{packets}}{\text{second}}\right) * L\left(\frac{\text{bits}}{\text{packet}}\right)$$

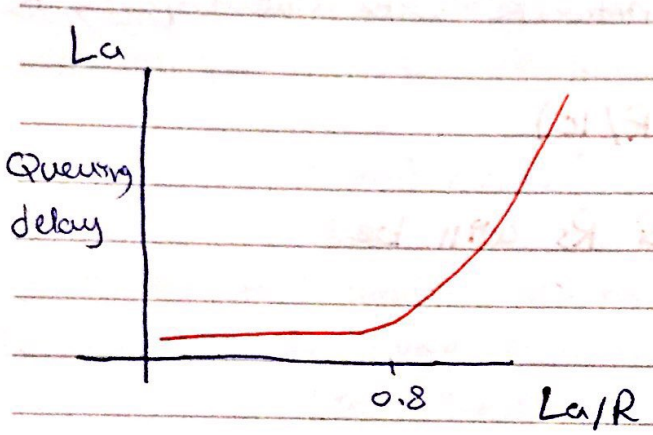
$$= L * a \left(\frac{\text{bit}}{\text{sec}}\right)$$

avg transmission rate = R (bits/sec).

- if at some moment
  $L_a << R$
  ↳ the avg queuing delay is Small

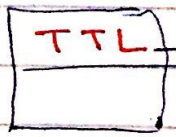
- if $L_a \simeq R$
  ↳ queuing delay is very large

- if $L_a > R$
  ↳ queuing delay goes to ∞ ⟹ Unstable network.



$L_a < 0.8R$
↳ Queuing delay is acceptable.

* Traceroute:
  ↳ command to measure the delay.



→ time to live: in terms of # hops

(x) TTL --
if TTL == 0
drop the packet and send msg back to source (TTL expired)
else forward the packet.

* Throughput:

- Bottleneck Link: the weakest link in terms of link capacity along a path.

↳ it determines the throughput.

* 10 end systems (sources), each with access network of link capacity R_s and 10 end systems (recievers) each with access network of link capacity R_c

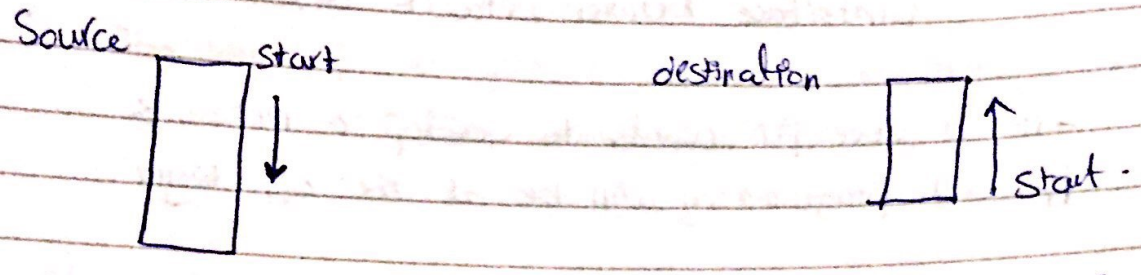
Network core is composed of network with link capacity R

Throughput $= \min(R_c, R_s, R/10)$

- most probably R_c or R_s will be the bottleneck.

* Layering System

1- divide & conquer & reduce complexity.



2- modularity : modification on one layer will not affect other layers → modification is easy.

3. encapsulation : each layer adds header that is checked only by the same layer on the other end.

- Layering considered harmful?
  - ↳ overhead data
  - ↳ lack of information gives less optimal decisions and actions
    - ↳ because each layer in the destination reads only one information from the source

* Cross layer protocol : a protocol that uses the information from two or more layers.

* Two layering systems:
  - ↳ 1) TCP/IP protocol stack : 5 layers & sometimes 4 layers.
  - 2) OSI model

- TCP/IP Protocol Stack.

① - Application layer : deals with all user applications (interface between network and user).

- if a developer wants to develop a network app, all programming will be at the app layer.

② - transport layer : end-to-end communication

Src : Src process

dst : dst process.

process-to-process communication.

from the process on the source to the process at the destination.

process : Program in execution.

③ - Network layer : node to node communication

Src : initial Src

dst : final dst.

routing & forwarding.

④ - Data link layer : Responsible of transmission data on one link.

Src & dst are the two ends of a link.

⑤ - Physical layer : Convert message into signal and send it.

- ISO/ OSI refrence model.

OSI : Open System Interconnection

- Session layer : to open a Session and make all communication in this session related.

- Presentation layer : for presenting the data to the app layer
  - ↳ encoding
  - ↳ encryption/ decryption
  - ↳ Compression.

- all the layers are downloaded with the OS.

- In TCP/IP, any required feature from session and presentation should be done at the app layer (developer).

- The internet uses TCP/IP.

- google.com

↳ at each layer, the data (packet) has specific name.

↳ Application layer : Message

↳ Transport layer : Segment

↳ Network layer : Datagram

↳ Link layer : Frame.

* Switch: is a transparent device that doesn't modify the frame

↳ Switch is layer "2" device because it reads layer "2" headers and make desicion based on the header

* Layer "X" device means that all headers in the layers up to layer "X" are checked in this device

* Router : is a layer "3" device

* Link layer header : is totally changed at each link However, the network layer header is only modified.

App.	Layer 5
transport	Layer 4
Network	Layer 3
Link	Layer 2
Physical	Layer 1

* Application layer and transport layer
  ↳ only existing in end systems

* Network layer
  ↳ exists in end systems & routers

* Link layer
  ↳ exists in all (Switch, Routers, end systems)

end of CH#1

PAGE 7th . Feb
DATE

CH #2

- Application Layer

- Application layer: Deals with all network applications.
  The user only sees this layer.

- Run on different systems: no many restrictions.
  - You should use protocols and standards when writing your own application
  ↳ I may define my own protocol.

- Communicate over network:
  ↳ Addressing.

- When you write any network application, you don't have to write anything for network core because the network core has only up to layer-3.

* Two main types of network applications:
  ↳ Client - server
  ↳ Peer - to - Peer.

- Client-Server:-
  ↳ Server that is always on.

- IP address → A.B.C.D ~ Decimal Dotted Notation (32 bits)
  ↳ max = 2^{32} = 4 billion IP
  ↳ Permanent IP address for server. However, the client has dynamic IP.

- IP address:
  ↳ Permanent: always the same address
  ↳ dynamic: keeps changing.

* Scalability: what happens if size of clients increased.
  ↳ to achieve scalability, use data centres.

- e.g.: google.com.

- Peer-to-Peer:-
  ↳ no always on-server to get the service.
  ↳ when the clients gets a service, it becomes a server.
  ↳ Self-Scalable: as # of clients increase, # of servers also increase.

  ↳ e.g.: torrents

PAGE 10th . feb.
DATE

- In OS, processes may communicate with each other locally.

- The client initiates the communication.

- Client process and server process in Client/ Server.

* Socket : like a door between the app & transport layer.

* In interprocess communication over the network, the com is between client process & server process.
- Therefore, we need to address the process on the two ends.
- The IP addresses end system (machine).
Therefore, we need one more type of addressing to address the process on the end system
This is the (Port number)

PAGE
DATE

* Port number:-
  ↳ 16 bits ($0 \rightarrow 2^{16}-1$) ($1 \rightarrow 1024$ reserved).

* The client can't communicate with any server without knowing the IP address & port # of the server.

* Some protocols use well-known port numbers.
  ↳ HTTP: 80
  ↳ FTP: 20 & 21
  ↳ DNS: 53
  ↳ SMTP: 25

  → Those are for servers only not for clients.

- The port numbers in the client are random.

* IP address → Address للجهاز
  Port number → Address for process.

* Socket Address:- IP + Port number.

* message Syntax: the format of the msg
* message semmantic: the meaning of the msg.

* what are the services that the application layer may need them from the transport layer?

↳ ① data integrity : no loss, no error & in-order

↳ ② Time (delay)

↳ ③ Throughput

↳ ④ Security.

* Transport layer protocols services

↳ ① TCP protocol

services: ↳ reliable transport : no loss, no error, in order

↳ flow control : when the recever is over whelmed by the sender

↳ congestion control

↳ when the networ (router) is over whelmed by the sender.

↳ connection-oriented : open connection befor communication

② UDP protocol

↳ Unreliable data transfer

↳ doesn't provide security, timing, security, ...

→ why we use UDP?

↳ it has less overhead due to smaller headers so it's faster.

PAGE 11th. Feb.
DATE

* Securing TCP
↳ TCP doesn't provide security, so they made a socket called Secure Socket Layer "SSL".

* HTTP is a client server protocol that is used for web pages.

⇒ HTTP://www.ju.edu.jo
↳ the base HTML file.
↳ HyperText Markup Language.

↳ it contains links to refrenced objects.

- each web page consists of several files (audio, images, ---) but we only request the base HTML file which includes the links of these files.

* each file has it's own URL
- URL: Uniform Resource Locator

* ju.edu.jo / file 1

↳ / path / refrenced object name.

* HTTP : Hyper Text transfer protocol.

- the client is web browser
- the server is web server

* HTTP traffic uses request/response messages.

* The most used web services.
  ↳ Apache web server
  ↳ IIS " Internet Information service"

- ~~[illegible]~~

* HTTP works on top of TCP port 80

* TCP is connection oriented
  ↳ before exchanging any data, a connection should be opened, and after exchanging the data the connection is down.

* Two types of connections for HTTP.

↳ ① non-persistent : open connection for each object exchanging the object, then close the connection.

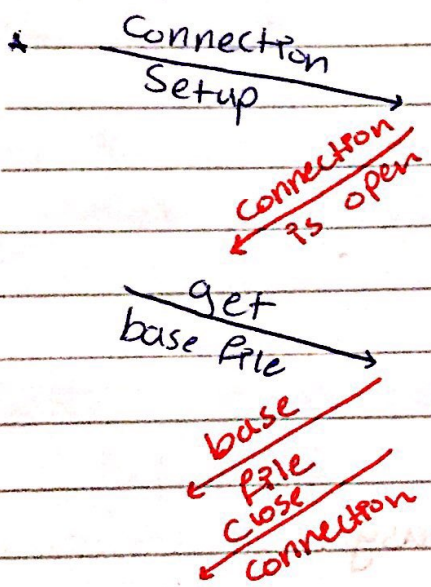
↳ ② persistent : open one connection, exchange all objects then close the connection.

- if we want to open a web page with ~~more~~ 9 objects using non-persisted connection, how many connections are opened?

9 + 1 = 10 connections.

objects: ↳ base file

* HTTP is Stateless
  ↳ does not hold any information about previous connections.



— Server
— Client

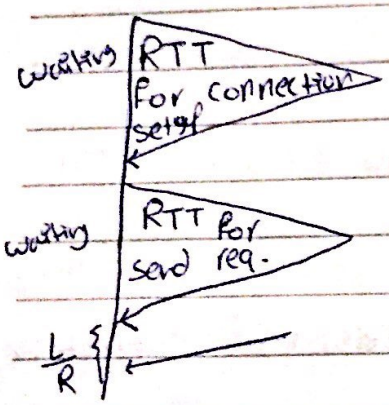
⇒ Non-persistent connection.

↳ it's not practical because each time ~~a~~ ~~ob~~ when want to open a web page, a new connection must be opened and steps are repeated.

* Round Trip Time (RTT)
↳ time to send one bit from source to destination and recieve back the response
(This includes all types of delays on all hops).

- I can't determine RTT for one packet or the exact RTT, so we calculate avg RTT.

- Avg RTT :-



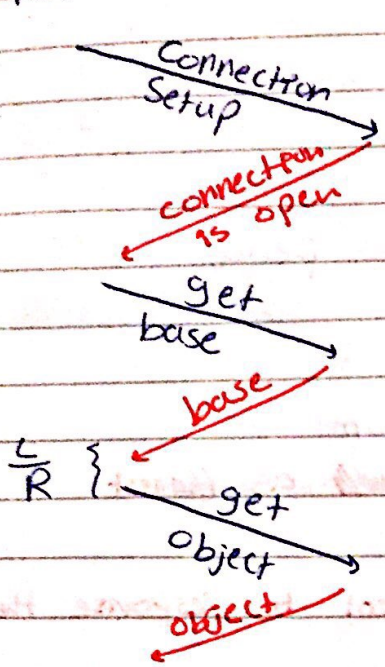
- what is the utilization of the client?
↳ how much it's busy.

$$\frac{L/R}{L/R + 2RTT}$$

- each file to be exchanged in non-persistent requires
$2RTT + \frac{L}{R}$

- For a page of 10 referenced objects each of the objects is of size L.
→ time to open the page $= 11 * (2RTT + \frac{L}{R})$
$= 22\,RTT + 11 * \frac{L}{R}$

* Persistent connection.



RTT (for connection) + $11 * (RTT + \frac{L}{R})$

$12 * RTT + 11 * \frac{L}{R}$

→ still slow.

- The solution is to do parallel requsts after connection setup

* HTTP Request message
  ↳ 3 parts:
    1) Request Line
    2) Header
    3) Body.

- Request Line ~ Method Line
  ↳ Object
  ↳ HTTP version

PAGE 14th. Feb
DATE

- How do I know when the header ends?
  \r\n\r\n after the header.

* Keep Alive 115
  ↳ keep the connection open for this period then close it.

- When the client types www.google.com
  ↳ the browser respones the request as request header and body.
  - the browser must include HTTP protocol to response the req.

* Uploading from input.

- Forms are sent to the server in two ways:
  1) POST method
  2) URL

* Head method: Similar to get method but the response will be with empty body.
  ↳ I ask for the file but I don't get it
  why we use it?
  For debugging and checking the file.

PAGE
DATE

* HTTP Response messages -
  ↳ 3 parts:
    ↳ Response Line
    ↳ Header
    ↳ Body.

- Response Line
  ↳ HTTP version
  ↳ Response Code
  ↳ Status phrase.

* telnet: Terminal Network
  ↳ to look at get /post methods.

* Cookies:
  ↳ when the client requests a web page from the server the server responses the request with something called "Set-cookie" which is a number and it will be saved on the server and browser which the client is using and the next request is set with that cookie number.

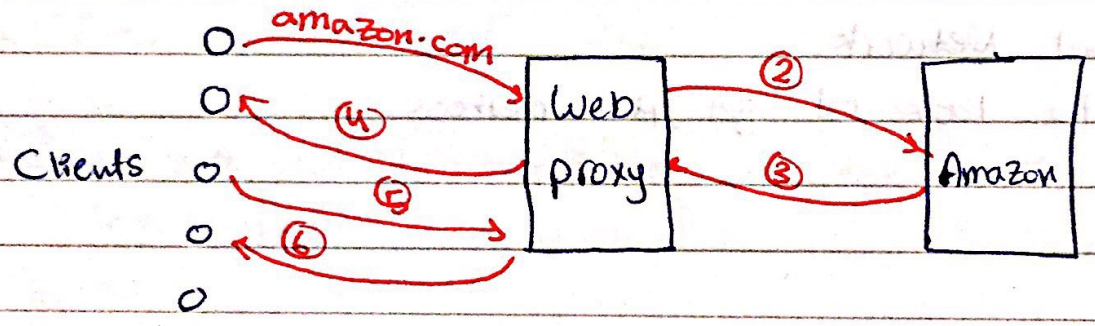
- ex: if you open Amazon to search for laptops, Amazon will set a cookie number for you and the next time you open Amazon, it will show you laptops based on your previous search because of cookies.

* Proxy [redacted] 8-
  ↳ door that you use to get out

- [redacted]

- Web proxy
  ↳ All web traffic should go through this server.

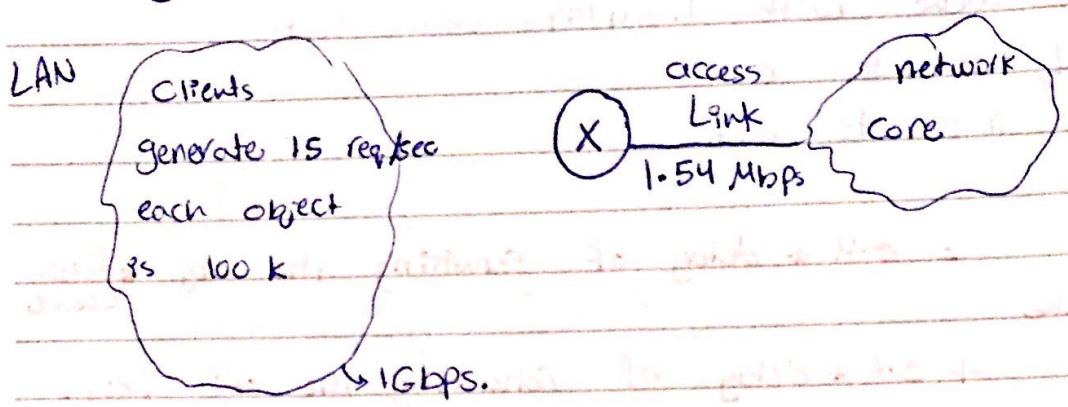
- Advantages of web proxys
  1) Cache
     ↳ Will not send every request to the server if these requests are to the same server.



2) Security
   ↳ I can control all the trafic sent from proxy.

PAGE 19th. Feb.
DATE

* Cashing example



- Delay = LAN delay + access Link delay + Network core delay.

$= \frac{100 * 10^3}{1 * 10^9}$ + minutes + 2 sec - mins.

- arrival rate = $15 \frac{req}{sec} * 100 \frac{k10}{req}$

= 1.5 Mbps

↳ high access link utilization.

$\left(\frac{L*a}{R} = \frac{1.5}{1.54} \approx 0.99 \right)$ → delay is minutes.

delay of access link.

- One solution is to make the access link 1.54 Mbps.

$\frac{L*a}{R} = \frac{1.5}{154} = 0.01$

new delay with access link of capacity 154 Mbps = 100 μs + ms + 2 sec ≈ 2 sec

- the problem of this solution is that it costs alot

* another solution
↳ to use cache with 1.54 Mbps access link.
↳ hit rate = 0.4
miss rate = 0.6

avg delay with cache = 0.4 * delay of returning the obj from cache
+ 0.6 * delay of returning the obj from network core

$= 0.4(100\ \mu s + 0 + 0) + 0.6(100\ \mu s + ms + 2\ sec)$

↓ the sum of the three delays.

$= 40\ \mu s + 1.2\ sec$

$= 1.2$** seconds

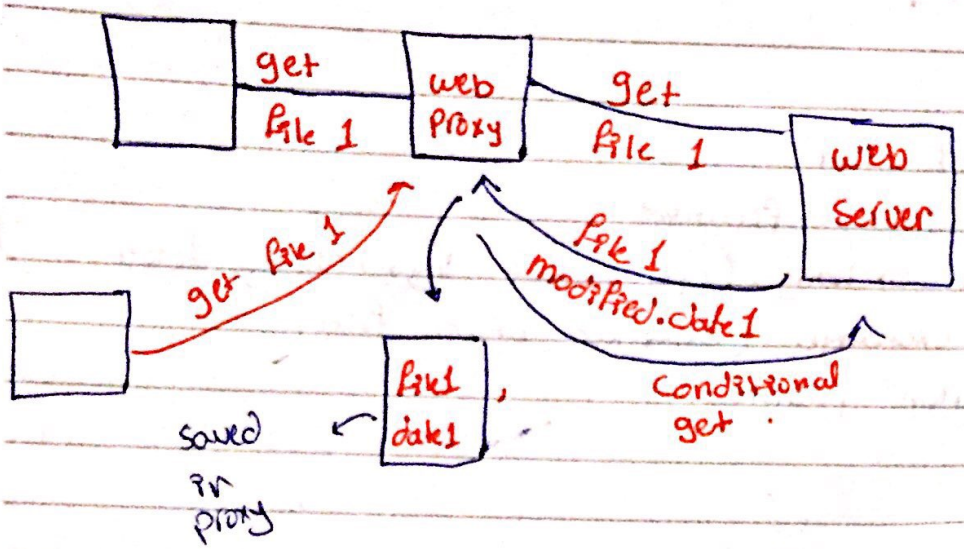
- why zero? because the request doesn't reach the network core or the access link

- Arrival rate = 0.6 * 15 * 100
= 0.9 Mbps

$\frac{La}{R} = \frac{0.9}{1.54} = 0.58 < 0.8$ → threshold of the delay.

→ delay = ms
↳ the access link delay

* Conditional GET.



* when a client requests a file from proxy, the proxy checks if the file is up-to-date or not from the web server using the conditional get.

↳ if the response is "not modified" → the file is up-to-date and the proxy send it to the client.

↳ if the response is "data"

↳ the file is not up-to-date and the server sends the data ↳ modified. to the proxy.

* FTP
↳ file transfer protocol

- FTP on top of TCP
- at ports
↳ 21 Control exchange (open, close, login, change folder, ...) → persistent
20 data exchange (download / upload files) ↳ non-persistent.

out of bond control channel.

- FTP maintains state, not stateless, to remember that the user logged in, the last opened folder.

* DNS

- the network uses the IP address but the client requests a website name "domain name" because it's easier to remember. However, the network uses IP because it's routable, can be used efficiently in routing.

- So I need sth to convert from domain name into IP → DNS!

PAGE
DATE

* Hirarechal
↳ composed of several levels and each level is responsible about part of the database.

* The server that has part of the DNS database is called name server

* Benefits of DNS:

1) Resolve name into IP address.

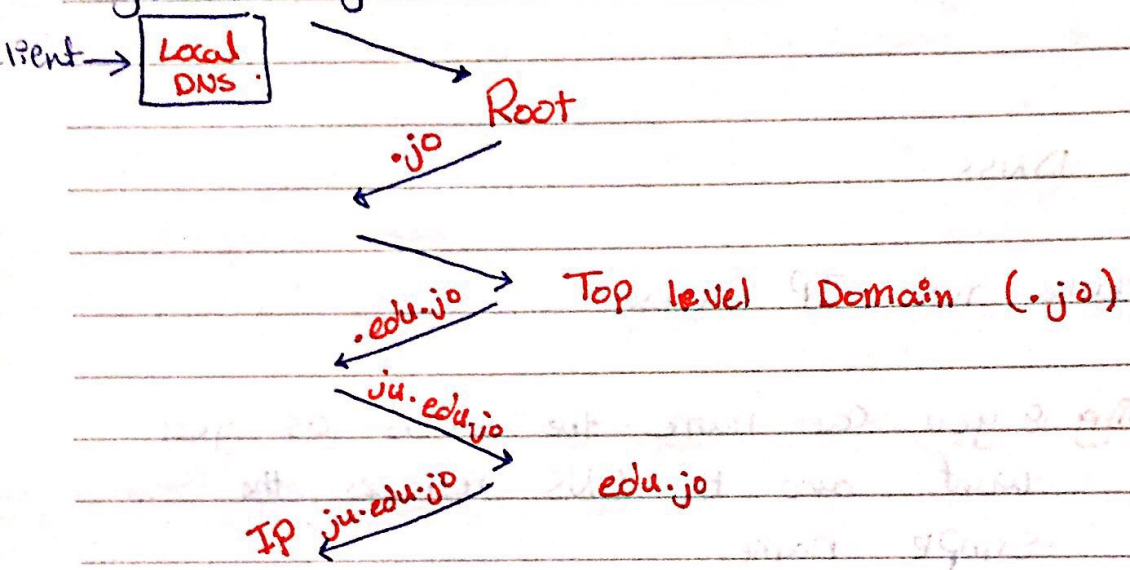
2) Host aliasing: you can name the servers as you want, and the DNS retrieves the simple name

3) Mail server aliasing: allows making the webservers name same as email server name.

4) Load balancing: Allow many servers with different IP addresses to be mapped into the same name.

* To resolve the IP address of specific host name, the client is may be required to ask several Name Servers because each Name Server is responsible about part of the DB

* ju.edu.jo



- when the requests a website it happens in level each level responses with it's domain.

- Any org or company that have public servers must own their own authoritative DNS on their premises. which is responsible about retrieving the IP addresses of these servers.

Root → TLD → authoritative DNS. always existing in DNS hirarechy (may include other levels).

PAGE
DATE

* LOCAL DNS (default name server).
  ↳ Similar to web proxy.
  - any DNS query is sent to the local DNS and if the answer is in the cache it's returned directly, otherwise, the query is sent out.

* To resolve the IP address of a specific host name, the client asks the local DNS, and after that there are two ways to retrieve the IP address.

1) Iterative.
   ↳ the local DNS is responsible about contacting several name servers iteratively until getting the IP address.

2) Recursive.
   ↳ Local DNS only contacts the Root level name server.

PAGE 24th. Feb
DATE

* DNS Recordss.

Name	Value	Type	TTL
- host	IP	A	
- domain name	Name of authoritative DNS	NS	
- Alias Name	Cannonical name	CNAME	
-	Name of mail Server	Mx	

* In DNS, the query & reply has same format but they are different in some flags.

* The local DNS Addresses are learned through DHCP

- DHCP Server gives information about local resources.

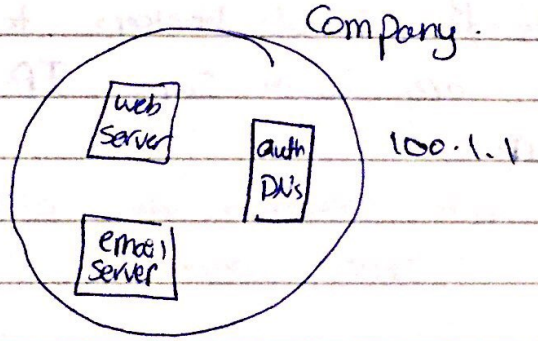
* If I want to create a website "mycompany.jo" with email "@mycompany.jo", what should I do?

- First, check if the domain reserved already.
  ↳ this is done by ICANN.

- ICANN is responsible about distributing all domain names & addresses.

- In jordan we contact المركز الوطني للمعلومات to check the domain.

- I must create an authoritative DNS for my Company.



Adds two records. مركز المعلومات الوطني

mycompany.jo	Auth DNS-name	NS
Auth DNS_name	100.1.1	A
web server		A
email server		MX

* Attacking DNS:

- DDoS: Distributed denied of service
  - ↳ easy to discover it because roots have firewalls, traffic filtering, ---

- Redirect Attack
  - ↳ Some one in the middle changes the response from IP1 to IP2.

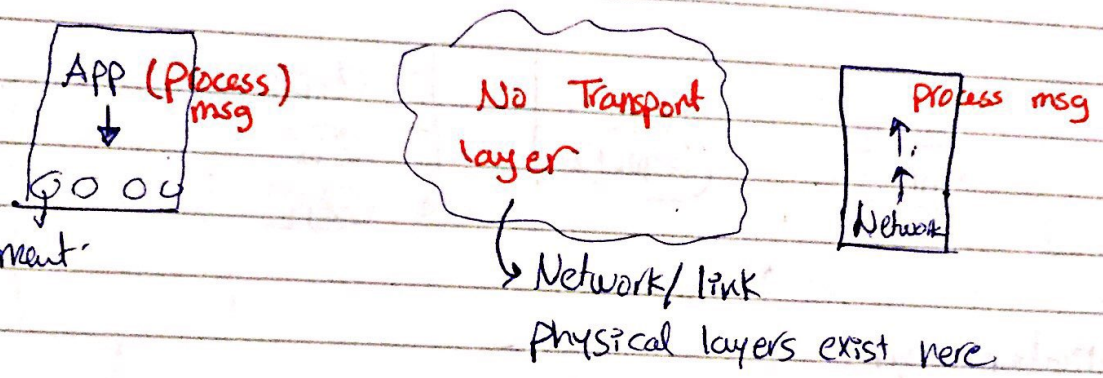
- Exploit DNS for DDoS.
  - ↳ all requests for 1000 hackers to all servers are with same IP
    IP = Victim IP.

end of CH#2

CH#3 26th Feb

Transport Layer

- Transport layer is responsible about process-to-process communication.
- process is at the end system. Does not exist in the network core. Therefore, the transport exist only at the end systems.



* At the transmitter, the transport layer breaks the msg into segments sent to Network layer.

- At the reciever, the transport layer assemble the segments into msg and send them to Application layer

- Network layer is responsible about sending the packet to the end system (final destination).
  - it exists in network core & end systems.

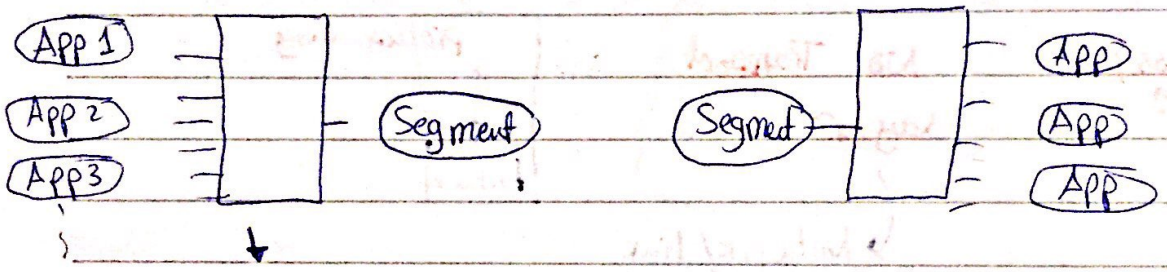
- The networ layer uses IP protocol which doesn't have any services (Security, reliability, loss, ---) So the transport layer tries to acheive these services, makes the transport layer work harder.

* **Best-effort:** No guarntee on any service, but it tries it's best effort.

- **Multiplexing** / **Demultiplexing**.

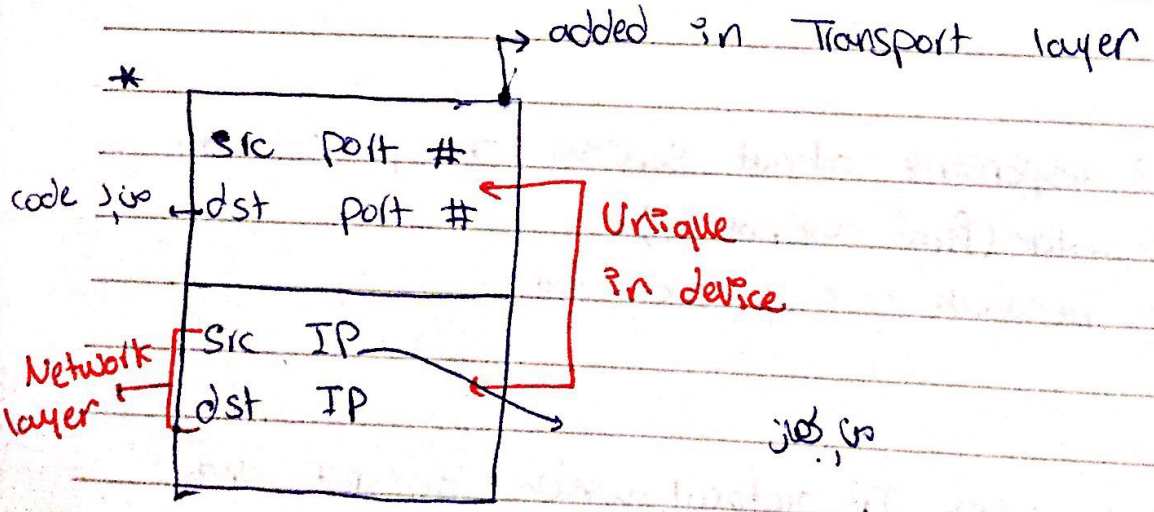
Multiplexing: happens at the Source

Demultiplexing: happens at the destination



the application layer msgs are sent in same segment format.

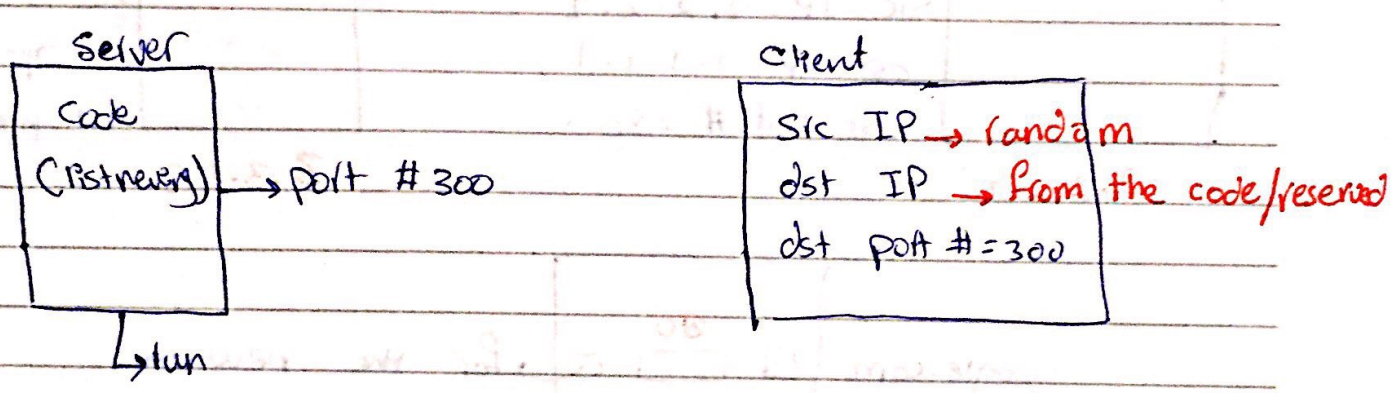
* In one Segment it's impossible to find payload from differed msgs.



* port # 16-bit
↳ reserverd HTTP.
↳ not reserverd

* For network communication to happen, The source should know the dst IP & dst port #.

*



- keep listening at port 300.

- if I sent 301 → Unreachable DSN port.

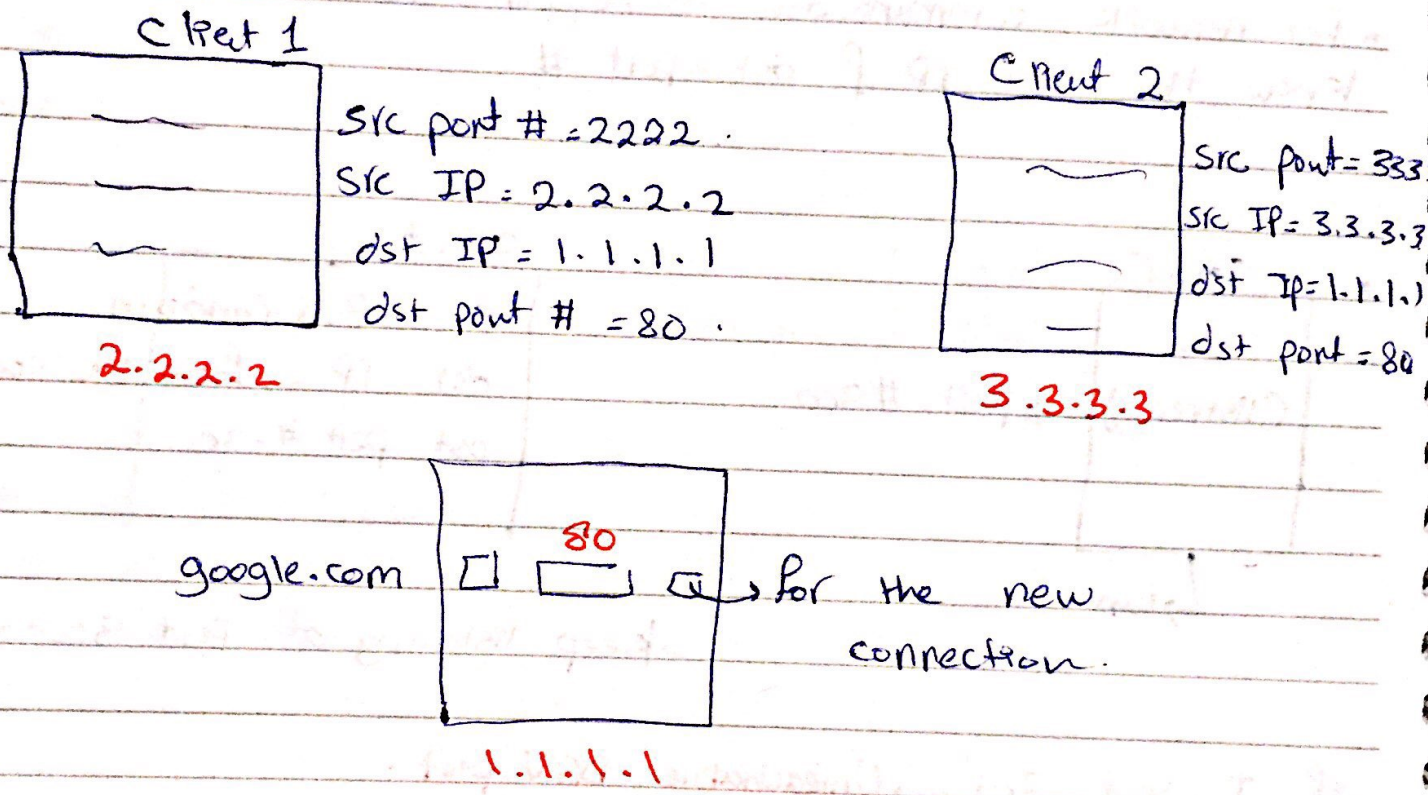
* DMUX → UDP → connection less
↳ TCP → connection oriental

- UDP
↳ DMUX is based only on dst & IP port #

- TCP
↳ for each connection, new thread is opened
DMUX is based on dst IP, dst port #, src port # & src IP.

- TCP is a multi-thread server

* on the arrival of request from new IP or new port #, The server creats new thread to serve the client.



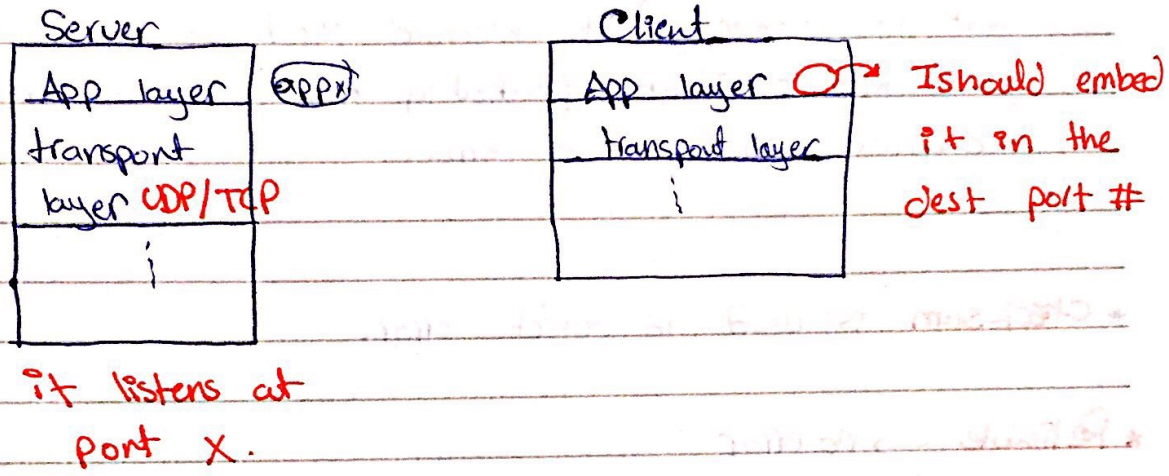
- for client 1 & client 2 new threads and new sockets are created

↳ because the port # and IP address are different

UDP → TCP

it should provide at least mux/Demux.

-Connectionless: the segments are not related to each other
each segment is handled independently.



*Length: To identify the size of the size of the segment, because the segment is of a variable size.

*Payload: is the message or part of it that arrived from the application layer.

*Checksum: is a 1-bit parity.

↳ Even parity: # of 1's including the parity bit should be even.

↳ Odd parity: # of 1's including the parity bit should be odd

- ex: data = 1011 0001 → even parity should be zero.

- In checksum calculation
  ↳ adds words of 16-bits together at the end
  the check sum equal the complement of the final sum

  - at the receiver, if the received checksum equal the calculated checksum, probably there is no error
  otherwise, there is an error.

* checksum is used to detect error.

* Reliable → no error
  ↳ no loss
  ↳ in order

- the complexity of the TCP and reliable data transfer (rdt) depends on how much the under layer is unreliable.

* the Network layer should know whether the segment is sent through TCP/UDP.

* Ideal scenario: no error, no loss & in order, so no need to add headers to acheiver rdt.

- At the receiver, there's no need to check anything, just pass the segment to the app layer.

5th. Mar

* rdt1.0 : ideal → app layer. مباشرة بطلع على ال
rdt2.0 : with error, no loss.

- if the msg sent with an error, the Tx detects it and ask the Rx to resend the data.

- acknolegment → data is correct, no need to resend.
- negative acknoledgment → data is not correct, need to resend!
  ↳ the Rx must have a buffer to save the data if the Tx ask for it again no need to take from app layer, take the data from the buffer.

* rdt2.1 : we add seq #
- If the transmisson is stop and wait.
  ↳ don't send next packet until recieving acknoledgment for previous one ⇒ needs one bit seq #

* rdt 2.2 : NACK free, It doesn't send NAK only sends Ack.

- what happens if the packet recieved is wrong and no NAK?
the Rx sends Ack for the last correct packet
  ↳ if packet 0 was sent and was correct, then packet 1 was sent with error, the Rx sends Ack for packet 0, so the Tx understands that packet 1 is wrong!

* rdt 3.0 & error & loss.
- to deal with loss, **timer** is added.
- on timer exping, the packet is assumed lost and retransmission is done.

- **Timeout**
  ↳ Should guarntee that the packet is sent from the sender to the receiver & the Ack gets back including the processing time

- **Premature time out**
  ↳ if the time out is short and the Tx resends the packet befor the ack from Rx is recieved.

- Time out
  ↳ too large → delay
  ↳ too short → premature timeout

- Usually, the time out is function of last RTT of packets.

* the problem of send & wait → very slow.

* Utilization of Sender $= \frac{L/R}{L/R + RTT}$

- Ex Slide 42

$R = 1\,Gbps$

$L = 8000$ bits

$RTT = 30ms$

Utilization of sender $= \frac{8000/1*10^9}{8000/1*10^9 + 30*10^{-3}} = 0.00027$

↳ very bad Utilization.

- Solutions: pipelined transmission (Sliding window).

*pipelined transmission:

↳ Send multiple packets before Ack is recieved but Ack is still needed.

⇒ Seq# > 1 bit.

* piplened transmission forms

↳ Go-Back-N (GBN)

↳ Selective Repeat (SR)

* Go-back-N:
  - ↳ one timer for oldest unacked packets
  - ↳ Ack is called Comulative Ack
    - ex: ↳ Ack 7 → this means that all packets with seq# up to 7 were recieved correctly.
  - ↳ on time-out: Send the oldest packet and all subsequent packets.
  - -Adv: One timer, send one Ack for multiple packets

* Selective Repeat:
  - ↳ one timer per unacked packet.
  - ↳ Ack is individual Ack
    - ex: ↳ Ack 7: only packet 7 was recieved.
  - ↳ on time out: only send the packet that caused the time out
  - -Adv: Resend only the lost packet.

PAGE 7th. Mar
DATE

*GBN & S.R both of them can send up to N packets before recieving Ack.
N = window size

* In Go-Back-N, there is no buffer at the reciever
- when a packet out of order is recieved in GBN, it's discarded, and sends Ack for newest good packet received in order

- receiver always has state: expected Seq #

↳ if the expected seq# was 3 and the packet received was 4 packet 4 will be discarded because it's out of order and sends Ack 2.

* In GBN, the sender has a window that shows:-
1- Sender - base
↳ oldest sent packet that not yet Acked.
2- N = 14

- when does the window slide?
when the receiver sends any Ack (comulative Ack).

- On time out for the segmented that has seq # = send base "Slide 48"
will retransmit all 8 segments colored in yellow.

* In Selective repeat, there are 2 windows views
- There is a buffer at both sender & receiver
  ↳ At sender & the buffer contains all segments not yet Acked.

  ↳ At receiver & the buffer contains all segments out of order.

- The two windows views & -
  1- At sender
    ↳ the window refers to the acceptable seq #s. that can be sent.

    ↳ the window slides when Ack on send-base is received, the sliding is up to next unacked segment.

  2- At receiver
    ↳ The window refers to the acceptable seq #s that can be received from sender

    ↳ The window slides when a segment with seq# = rcv-base and slides up to next unreceived segment

PAGE 10th. Mar
DATE

* TCP
  - ↳ Unicast : point-to-point
  - ↳ Multicast : one sender, multiple receivers.
  - ↳ Broadcast : one sender, all are receivers.
  - ↳ Anycast : one sender, and any receiver out of a group

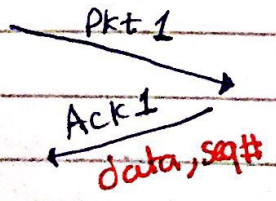
* window size in TCP is determined by flow control and congestion control

* full-duplex : on the same opened connection, both sender and receiver can send data.

* Sequence-Number : is the seq# of first byte in the segment.
  - ex: Seq # = 100
    Size = 50
    Seq # = 150.

* Ack-Number : is comulative Ack.
  - ↳ Positive Ack
    - ex: Ack 100 → the receiver received up to byte #ack correctly and is expecting byte 100.

*



* **Header length** : determines the length of the header because it is of a variable size due to options.

* **U** : Urgent data, has higher priority
  - not implemented.

* **A** : this segment contains acknolegdment.

* **P** : push segment, please pass the segment as soon as possible
  - not used.

* **R, S, F** : for connection setup and shut-down.

* **Receiver window** : window size.
  # of bytes that can be sent before receiving Ack.

* **Checksum** : detect error

* **Urgent data pointer** : refers to the place where the urgent data exists in the segment
  - not used.

* <u>Note</u> Transport layer exists in OS
  App layer exists in apps.

* Slide 60

it sends only one byte "c"

Seq # = 42 → it means that the seq# of the first byte = 42.

Ack = 79 → it means that it received up to 78 and waiting 79.

* Time out & to solve the loss
  - ↳ too long & delay of retransmission
  - ↳ too short & premature - time outs → many unnecessary retransmission.

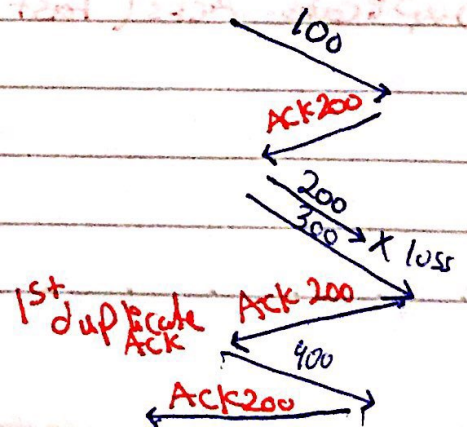
* Time out = weighted average RTT + safety margin.

- Avg RTT for 1000 segments = $\sum_{i=1}^{1000} \frac{RTT_i}{1000}$

* Time out in TCP = weighted avg RTT + safety margin
  - ↓ 4 * DevRTT

- new avg RTT = $\frac{1}{8}$ * last sample RTT + $\frac{7}{8}$ old avg RTT

* TCP always send Ack for last good received in-order segment.



*Slide 65:

- Retransmission is for oldest unacked segment in case of the following:-
  1. Time out
  2. Duplicate Acks.

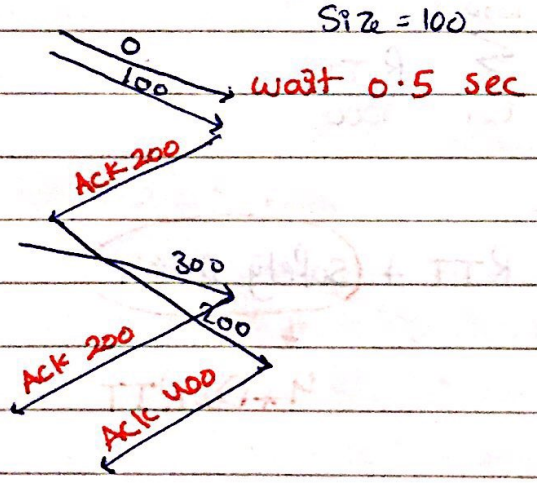
- Slide 66

*Single timer in TCP like GBN

- Slide 70

- Case #1 : delay 0.5 second to wait for subsequent segments and send one ack for all of them.

- Case #2 : TCP tries to send one Ack per 2 segments



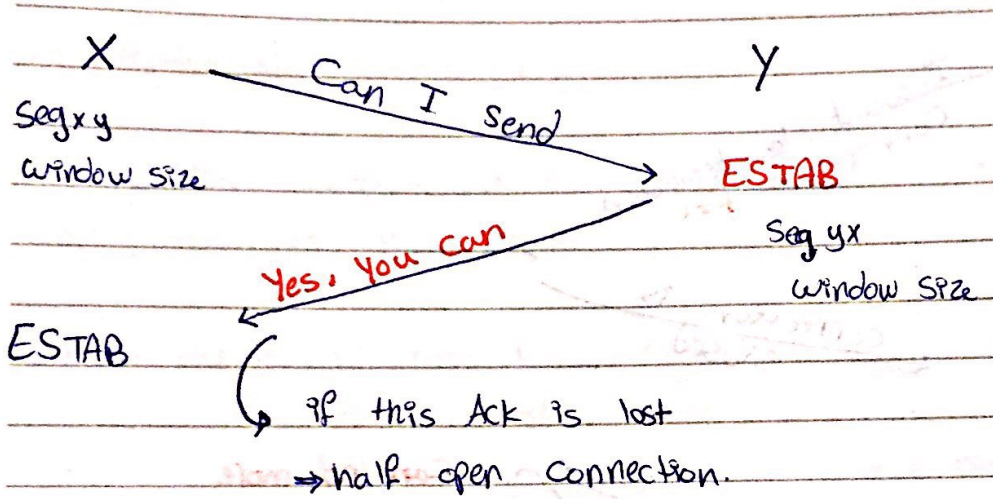
*Retransmission in TCP is due to :-
1. Time out
2. Duplicate Acks (fast retransmission).

* To reduce data rate, we reduce window size
* At connection setup, sender and receiver agree on window sizes.

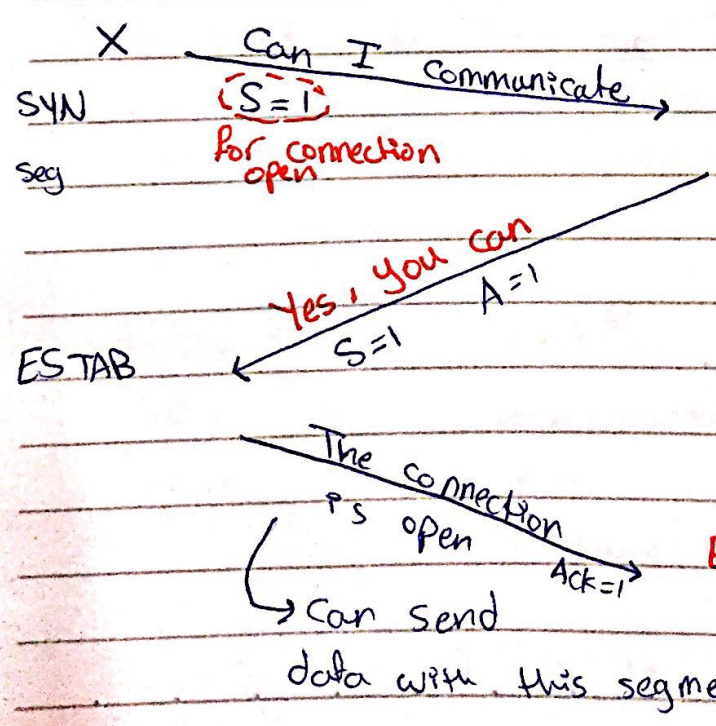
-Slide 77

Connection management

* 2-ways Handshaking



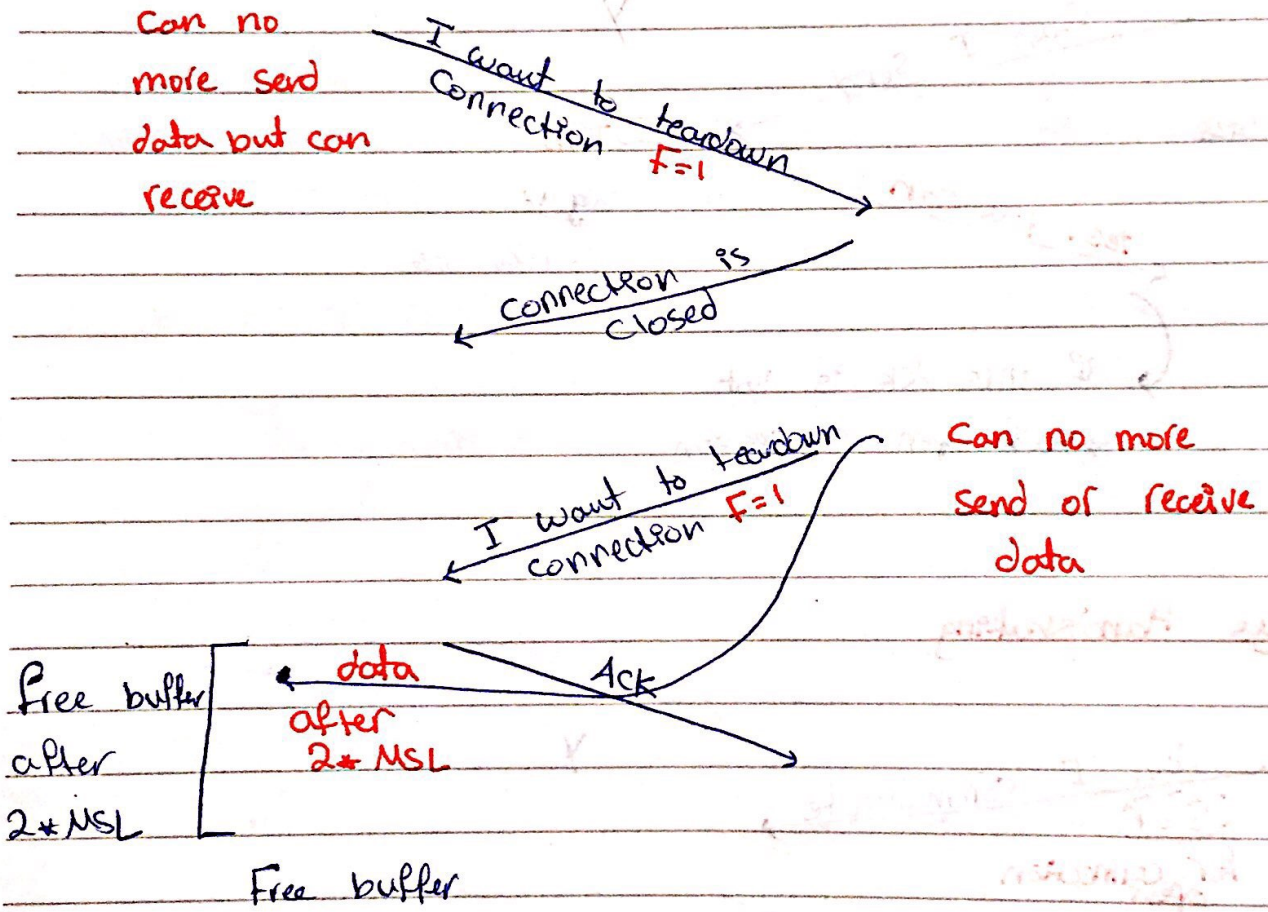
* 3-ways Handshaking



* when R=1, the sender is trying to open a connection with non-open port.

Slide 82

* Connection Teardown : should be initiated from the sender and receiver



MSL : Maximum Segment Lifetime.

- Slide 85

Congestion Control

↳ it's done by managing window size which is the same window controlled by flow control

- Flow control → window size = X
- Congestion Control → window size = Y

→ window size = min (X, Y)

- Congestion happens at the router, However, the router will not tell the sender.

* How the sender will know that the router is congested?

↳ from congestion effects

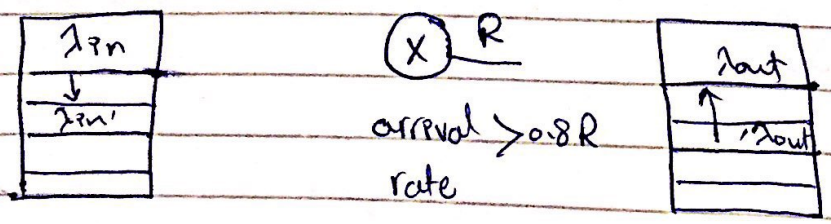
1. too long delay until receive ack
2. loss (3-duplicate Acks, time out)

- Slide 86

λ_{in} : data rate from App layer to transport layer at sender

λ_{out} : data rate from Transport layer to App layer at receiver

ideally $\lambda_{in} = \lambda_{out}$



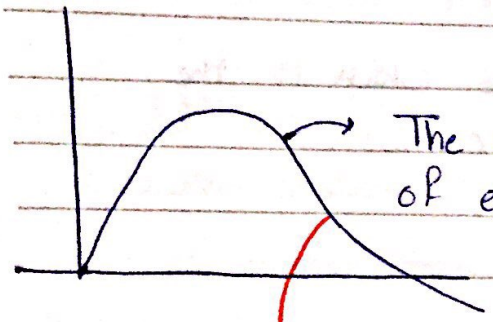
arrival rate > 0.8R

queing delay → ∞

$\lambda_{in}' > \lambda_{in}$, $\lambda_{in}' = \lambda_{in}$ + rate of retransmission

↳ due to loss or premature timeout

- Slide 94



The network should raise a flag of emergency (fast solution)

because most of transmitted data is just retransmission due to large delays.

* In TCP/ IP protocol Stack
  ↳ congestion is solved only by sender & receiver.

- Slide 99

TCP uses AIMD & Slow Start.

AIMD: Additive Increase ~~Multiplicat~~ Multiplicative Decrease

Slow Start: Start with small window, duplicate window size on each received Ack, up to a threshold
After this threshold, start additive increase.

- Slide 100

On Congestion:

↳ Tahoe: new window size = 1 MSS (Maximum Segment Size)

↳ RENO

↳ time-out: new window size = 1 MSS

↳ 3-duplicate: new window size = $\frac{1}{2}$ window size at congestion point.

- How do I know that there's a congestion?
  ↳ Time out / 3-duplicate Acks.

  - Time out is more dangerous because the network is congested

  - 3-duplicate Acks, the network is functioning well, but at some moment one segment was dropped.

- Slide 101

How to determine the threshold after congestion?
↳ The threshold is half of last window size at congestion point.

19th . Mar

* CH#48 Network Layer

- **Network Layer functionality:** take the segment from the transport layer, encapsulate it into datagram and deliever it to the final destination through many hops (routers).

- The network layer exists in all hosts + routers.

- Slide 5

* Network layer functions:-
  - ↳ Routing } internet
  - ↳ forwarding }
  - ↳ Connection setup in some types of networks like ATM networks. (Virtual Circuits) between routers.
    - ↳ huge overhead on the network layer level because all routers will participate in setting up connections.

- **Routing:** is a planning phase that plans each packet to be delievered to its final destination, on which interface it should be forwarded, and it results a routing table.

- **forwarding:** when the datagram is received, use routing table to forward the packet to its final destination.

- If no matching is found in the routing table, the datagram is dropped.

*Routing is done in two ways

↳ Static : the admin sets each destination on which interface can be reached.

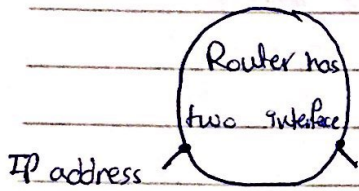
↳ Dynamic : by exchangin frequent hello packets which contain routing table or part of it.

-Slide 6

Routing table.

range

destination address	interface #
Static	
dynamic.	



-why range?

because the IP address is 32 bits and if each IP is sent alone (not range) there will be 2^{32} destinations → too large table. and huge delay when retrieving.

-we use IP V4 (32-bits)

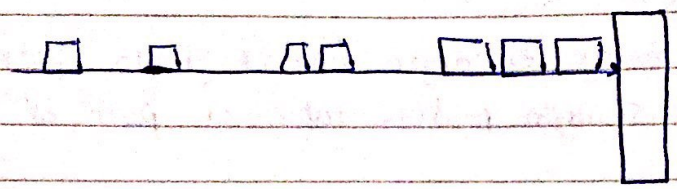
* Default Static Path (Default Static route)

↳ otherwise path (when there is only one ISP for the network it goes to the default path because there is no matching entries in the table).

-Dynamic routing is set by routing algorithms and protocols.

- Slide 8

- Inter packet spacing (Jitter)
  ↳ time between two packets to be delivered.



- Usually handled by buffering.

Slide 11

IP = W.X.Y.Z (it's written in decimal not binary, not hexadecimal).
↳ each is 8 bit

32 bits

Network part → hosts within the same network share network part, but differ in host part

Host part.

dst IP range	interface #
Network part	

dst IP = Network part + host part → (X)

-Why Network part in routing table?
because it's more important to get to the final hop then get inside the network.

-Slide 11
in link interface 0
21 bits are common between starting address and last address
(11001000 00010111 00010XXX XXXXXXXX)
↳ don't care means they are different (maybe 0 or 1)

-I have to write in decimal
How?
-I write the common bits in decimal, then the don't cares I make them zeros and write the decimal number then (/) and after it the number of common bits.

↳ 200.23.16.0/21
↳ # of common bits
# of bits in network part

⇒ this number represents
(2^{11}) hosts.
host bits = 32-21

* when a datagram arrives at a router, it must contain destination host IP. The forwarding function of the router checks this host IP belongs to which range and find which interface the datagram should be forwarded on.

-if the interface is busy, what to do?
Queuing, each interface has queuing.

-Slide 12
Ex:
the dst IP = 11001000 00010111 00011000 10101010
to which interface I should pass it when interface 1 & 2 are very similar?

-We follow the longest prefix matching rule which means when a datagram with destination address matches 2 entries in the routing table, it will be forwarded on the interface that has the largest prefix match. أكبر عدد bits متشابهين من اليمين.

-Slide 13 X محذوفة

-Slide 16
- Ver: Version (we have IP V4 and IP V6) (4 bits)

- header length: Variable length due to options if added.

- type of services: Quality of service (Real time guarnteed throughput ....) (not used).

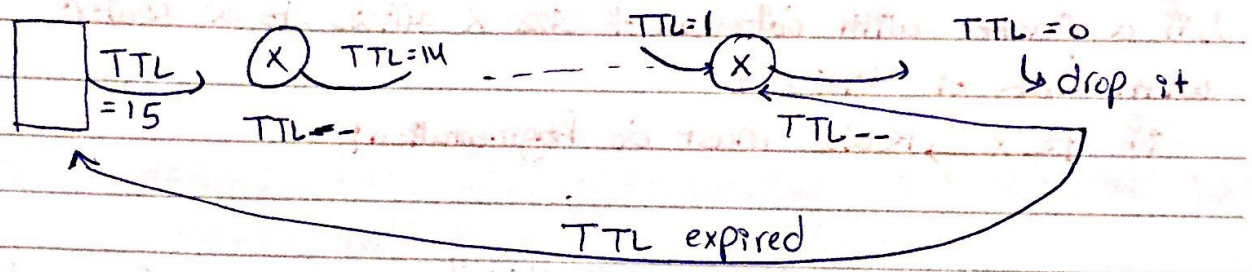
-length: length of datagram.

-16-bit identifier + flags + fragment offset
↳ for fragmentation and reassembly
they are needed when the router in enforced to fragment a segment

* when a msg is received from App layer to transport layer it's broken (fragmented) into segments, the router tries its best effort not to fragment the datagram, but sometimes its forced to fragment them so it uses 16 bit + flags + fragment offset.

* reassembly happens at the final destination.

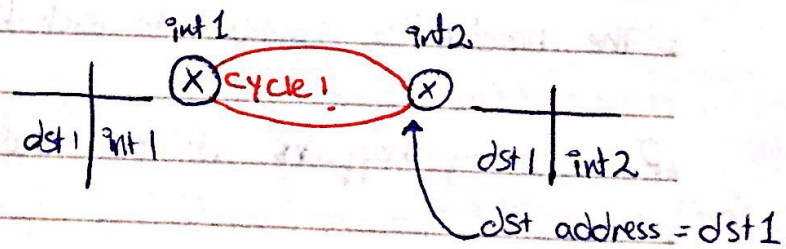
- Time to live : in terms of # of hops.
if =15 → this datagram can be forwarded at most 15 hops.



it's sent from the router that decremented the TTL to zero.

- Upper layer : the protocol in Transport layer.
- checksum : detect error but take no action (best effort)

- why there is TTL?
- To prevent cycles.



- Src IP and dst IP : to deliever the packet to it's final dst.

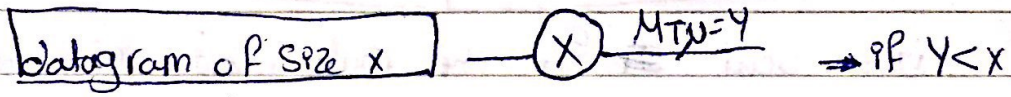
* minimum size of datagram header = 20 bytes.
↳ maybe more because of options added.

- Slide 17

* Sometimes, router will be enforced to fragment the datagram.
- Each type of link has a characteristic called **Maximum Transmission Unit (MTU)**
  - ex:

Fiber → MTU = X
UTP → MTU = Y

* when the router is enforced to fragmentation?
  ↳ if a packet with datagram of size X arrives to a router with link of MTU = Y
  if Y < X → Router must do fragmentation!



→ if Y < X ⇒ fragmentation!

⇓

This datagram is sent as several datagrams each of them of size = MTU!

* The header is copied into each fragment.

* Reassembly happens at the final dst.

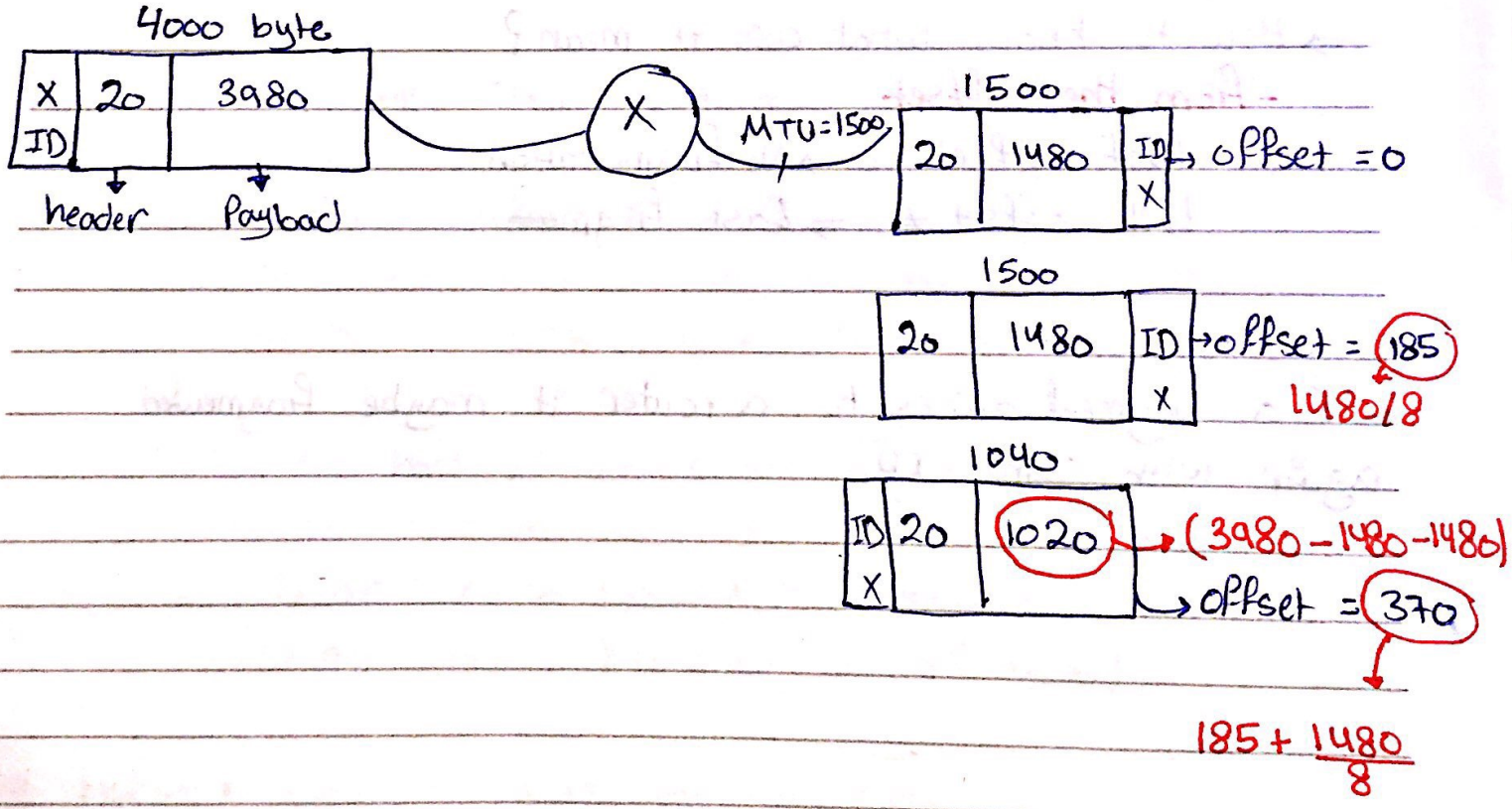
Slide 18

* 16-bit identifier, flags and fragment offset will be used in fragmentation and reassembly.

* ID: the same ID will be used for all fragments.
  ↳ but different datagrams have different IDs.

- why ID is used?
  ↳ So when the packet arrives at the dst for reassembly it knows that these fragmentations are for the same datagram if they have the same ID.

* ID is different from datagram to another, but it's the same for all fragments that belong to one datagram.



* Frag flag = 1 ⇒ this is a fragmented datagram and there will be more fragments following this fragment

* offset → offset of the first byte in the fragment
It counts * 8 bytes unit.

* Frag flag shows two things
1) This fragment is fragmented
2) There is next following fragments.

- what if Frag flag = 0 ??
- may be it means that this datagram is not fragmented or maybe it means that this is the last fragment.

⇒ How to know what does it mean?
- from the offset
↳ if offset = 0 ⇒ No fragmentation
↳ if offset ≠ 0 ⇒ Last fragment.

* if a fragment arrives to a router it maybe fragmented again with same IP.

- Slide 20

IP addressing.

- IP address Version 4 (V4) : 32-bit address
  ↳ Up to 2^{32} IP addresses.

W.X.Y.Z

Decimal Dotted Number.

- Classful Addressing :-
- IP addresses are divided into classes (determined by the W part) :-

① Class A ⇒ if W Starts with 0
- 8 bits for network part
- 24 bits for host part
- $(2^{24} - 2)$ host IP addresses.
- W in the range [0-127]
- private IPs : 10 . X . X . X
  ↳ decimal
- Public IPs : remaining.

- what does private IP mean?
  ↳ It means that it's free, can be used without coordination with anyone. It's used locally only!

- Why the host IP address are $2^{24} - 2$, why -2?
  because there are two reserverd IPs. in host part
  ↳ all zeros → Network IP (reserved)
  ↳ all ones → broadcast address (reserved).

ex Network part = X, host part = all ones.
↳ this means send the packet to all hosts which have network part = X.

② Class B ⇒ if W Starts with 10
- 16 bits for network part
- 16 bits for host part
- $(2^{16}-2)$ host IP addresses
- W in the range [128-191]
- Private IPs: [172.16.x.x — 172.31.x.x]
- public IPs: remaining.

③ Class C ⇒ if W starts with 110
- 24 bits for network part
- 8 bits for host part
- $(2^{8}-2)$ host IP addresses
- W in the range [192-223]
- Private IPs: [192.168.x.x
- Public IPs: remaining.

LAN

* only private IPs can be used

(x) → only public IPs.

*LAN Limits is to the interface to router in Slide 20

↳ there is 3 LANS.

how?

Start counting from router and count each interface seperately.

PAGE 28th . Mar
DATE

* Different LANs must differ in network part!
- in Slide 20 we have 3 LANs but they all start with 223, how they are different?
- They must differ at least in one bit.
↳ (223.1.2.9), (223.1.3.27), (223.1.1.4)
- to what class do they belong?
Class C

* Classful: when you want to buy block of IP addresses, you must buy either class A, class B or class C.

- what's the problem in classful addressing?
- if I want to buy 1000 IPs using classful addressing I must buy class B which gives me $2^{16}-2 \simeq 65{,}500$ IPs and I only need 1000, so underutilization happens (64000 IPs are not used).

↓ solution
* Classless: You can assign any # of bits to the subnet and the remaining for host
ex: 27 bits for subnet part, 5 bits for host part.
⇒ 30 hosts.

* in classless if I want 1000 hosts
↳ 10 bits for host. ($2^{10}-2 = 1022$ hosts) -

* In classful addressing, I can know to which class it belongs, host part and network part

ex 223.1.3.1 ⇒ This is class C
subnet part = 223.1.3 (24 bit for network)

- ex 223.1.4.5
223.1.3.2
Can there hosts communicate directly without router?

↳ No, they differ in subnet part ⇒ different LANs
⇒ they need router to communicate.

- ex 223.1.3.1
223.1.3.2.
↳ Same subnet part, they don't need router to communicate.

* In classless, I can't directly know subnet part and host part because they can be any number of bits. So I must add the number of bits in subnet part.

ex 200.2.3.7/16
200.2.6.5/16 # of subnet part bits.

↳ Same subnet, because the left most 16 bits are common so they don't need router to communicate.

PAGE
DATE

* What is the wrong with the following IP addresses?

1) 200.300.100.1 / 24

300 > 255

↳ each dotted part is 8 bits ($2^8 = 256$)

so numbers must be between [0 - 255].

2) 200.200.200.0 / 24

This address is reserved (Network address)

↳ 24 bits for network part & 8 bits for host part.

those 8 bits cannot be all zeros or all ones.

3) 100.100.100.31 / 27

This address is reserved for broadcasting.

↳ 5 bits for host.

31 = 000 1 1111

all ones → reserved (for broadcasting).

* IP Config

↳ IP: 200.3.7.5

↳ Subnet mask: 255.255.255.0 } → 200.3.7.5 / 24

→ if I count the ones from the left → # of network part.

* / 27 in subnet mask form.

255.255.255.224

1110 0000

* If IP is given and subnet mask ~~[illegible]~~?

Use AND Gate.

200.3.7.5 AND 255.255.255.0 = Subnet IP address.

*Ex: Assume a company has the block of IP addresses 100.1.8.0/23, and it is willing to divide its network into 8 subnetworks of equal size where each subnetwork must have 50 hosts.

Sol: it's classless with 23 bits for network.

- first determine # of hosts.

$2^{32-23} = 2^9 - 2 = 510$ hosts.

we want 8 subnetworks each with 50 hosts

$50 * 8 = 400$ hosts

$510 > 400$ → so this can be done with no problem! (the network can be divided)!

- Subnetting: divide the big network into smaller subnetworks

- How to do subnetting?

IP = subnet bits, host bits.

Subnetting is taking i bits from host to subnet.

(23) (9)

i bits

↳ 2^i subnets, each can have up to $(2^{9-i} - 2)$ hosts

- How to determine i?

$\log_2$ (# of subnetwork) = $\log_2(8) = 3$ bits

IP address = 100.1.8.0/23

1 bit from here + 8 bits from here

convert the parts that contain the host bits into binary

23 bits for network decimal → 100.1. 000b 1000 . 0000 0000

9 bits for host.

they don't change (common)

we want to take 3 bits (left most 3 bits).

100.1. 000b 100 0.00 00 0000

000 ①
001 ②
010 ③
⋮
111 ⑧

① 100.1.8.0/26 → 3 bits from host to network
② 100.1.8.64/26
③ 100.1.8.128/26
④ 100.1.8.192/26
⑤ 100.1.9.0/26
⑥ 100.1.9.64/26
⑦ 100.1.9.128/26
⑧ 100.1.9.192/26

* $2^{32-26} - 2 = 62$ hosts in each subnetwork.

* what is the broadcast address of network 3?

100.1.8.1000 0000

26 bits for network → for broadcast they must be all ones.

address = 100.1.8.191/26

* what is the first IP in network 4?

100.1.8.193/26 because the first one is for network

* IP 100.1.8.130/26 belongs to which network?

Network 3

PAGE DATE 2nd. Apr

*Ex Assume you have the IP block 100.0.64.0/22 and you want to divide this block into several subnets as follows:-
400 hosts, 200 hosts, 100, 50, 25, 12, 12

Sol different sizes of subnetworks.
7 subnets.

- in this IP we have 10 host bits so we have up to $2^{10} - 2 = 1022$ hosts.
if we borrow 3 bits from the host bits
$\Rightarrow$ 7 host bits up to $2^7 - 2 = 126$ host per subnet!
(This is not enough because some subnets need more than 126 hosts).

- How to solve problems with variable size hosts?
[1] Order the subnets descending
(400, 200, 100, 50, 25, 12, 12)

[2] Determine how many bits needed for each subnet.

[3] Determine how many bits needed to be taken from host part to network part.

→ 100.0.64.0/22

0100 0000.0000 0000 → host bits

I change this bit (left most bit of host)

① 400 → $\lceil \log_2(400) \rceil = 9$ host bits

② 200 → 8 bits

③ 100 → 7 bits

④ 50 → 6 bits

⑤ 25 → 5 bits

⑥ 12 → 4 bits

⑦ 12 → 4 bits

① 0 → 100.0.64.0/23

1 → 100.0.66.0/23

0 → ② 100.0.66.0/24

1 → 100.0.67.0/24

1 → 100.0.67.128/25

0 → 100.0.67.0/25 ③

1 → 100.0.67.192/26

0 → 100.0.67.128/26 ④

0 → ⑤ 100.0.67.192/27

1 → 100.0.67.224/27

0 → ⑥ 100.0.67.224/28

1 → ⑦ 100.0.67.240/28

* Supernetting (Router aggregation).

↳ You have several Subnets and you want to merge them into one network ⇒ to reduce routing table size.

* IP = Subnet bits Host bits

Subnetting

Supernetting.

- How to know what bits to take?

Least from network part (different bits).

Ex

①	200.23.16.0/23	0001 0000
②	200.23.18.0/23	0001 0010
③	200.23.20.0/23	0001 0100
④	200.23.22.0/23	0001 0110
⑤	200.23.24.0/23	0001 1000
⑥	200.23.26.0/23	0001 1010
⑦	200.23.28.0/23	0001 1100
⑧	200.23.30.0/23	0001 1110

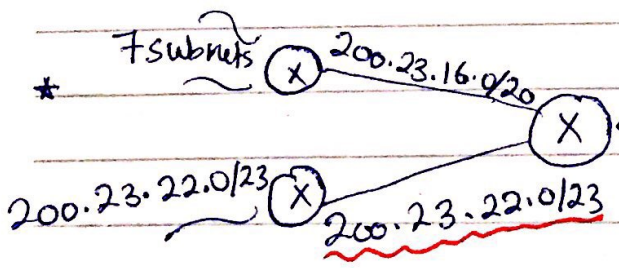
Changing bits

↓

replace them by zeros.

⇒ 200.23.16.0/20

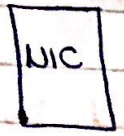
*



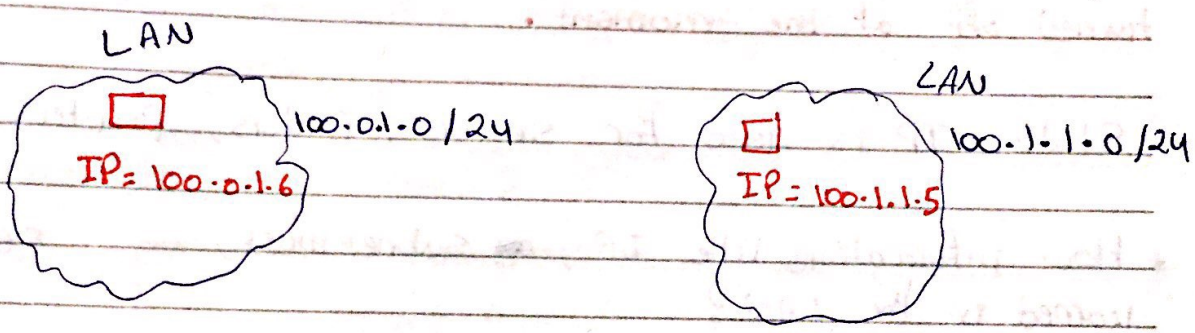
dst adress = 200.23.22.5.

Longest prefix match rule

* IP address must be unique!
- How many addresses can be given for a device?
  ↳ depends on number of Network Interface Card (NIC) but usually one.



IP: Logical → change (because it takes the IP LAN).
Macs: Physical → which means that it doesn't change (C.I.P)



- How to set an IP address on any host?
- We have two ways
  1. Static: the admin (user) sets the IP
  2. Dynamic: the host automatically learns IP address.

- What are static problems?
  ↳ each time you connect to a network you must set the IP for it, subnet mask, gateway and local DNS and it's difficult when the network is big. Choosing a unique IP is hard!

* To connect to any network, what do we need?
  1. IP address
  2. Subnet mask
  3. IP address of gateway (to get outside the LAN)
  4. Local DNS

* When setting the IP address by the user (static) if the address was not unique the LAN sends an IP conflict msg, so another IP address must be chosen.

- If no conflict msg is sent, does that guarntee that the IP address is unique (no conflict)?
- Maybe there is a conflict with a device that is turned off at the moment.

- Static IP is used for servers, routers, printers.

* How information like IP, subnet mask, --- are learned in the LAN?
- By broadcasting.

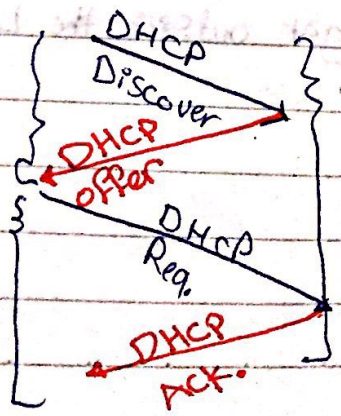
* Router cuts the broadcast (ما بوصلها برا ال LAN)

Slide 31

DHCP: Dynamic Host Configuration Protocol.
↳ the one responsible of distributing IP addresses + other information.

- How to know where is the DHCP?
Host does broadcasting for the network.

- To find (optional) a DHCP server
- Request an IP address from a DHCP server (mandatory)



Broadcasted, why?
- To tell all other DHCP servers that this DHCP is offering a service to the requesting host

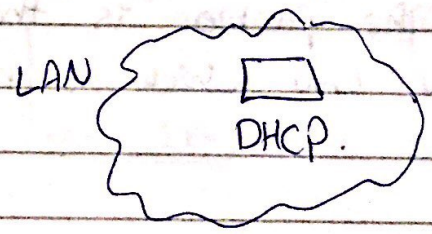
* After these four msgs, the host got an IP address and any other needed information

- Why the first two msgs are optional?
- Because I may already know the DHCP server, so I don't have to search for it.

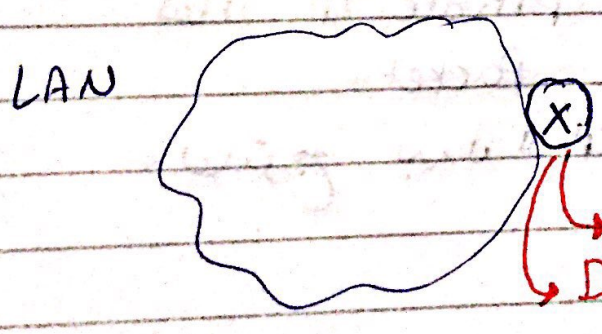
* To know whether the IP is static or dynamic
Use the command IPconfig/all.

* What we do on a DHCP?
1. Specify pool of IP addresses
2. Reserve some IP addresses
3. Lease time (depends on number of active users on the network)
4. Other information

* Where to put the DHCP server?
- DHCP can't be outside the server, it must be in LAN or on router to receive the broadcast msgs.



* To put the DHCP outside the LAN, it doesn't work directly. we must set settings on the router.

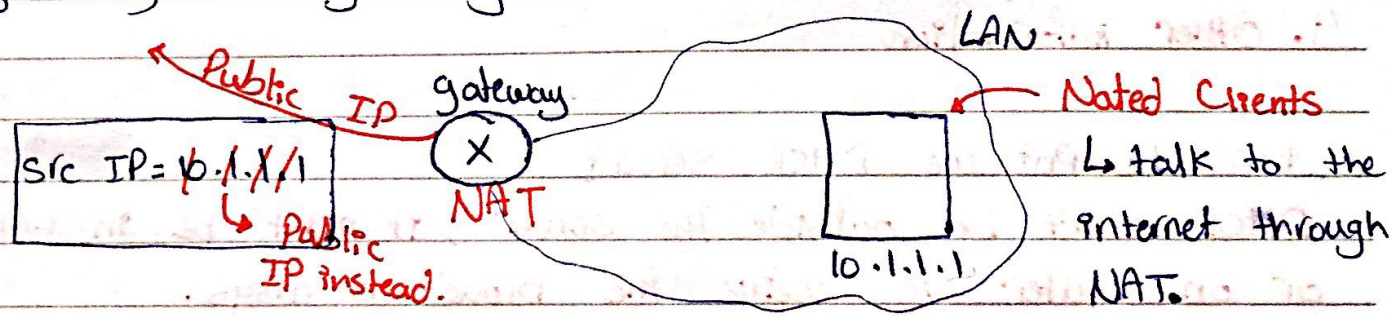


the router cuts the broadcast and send the msg only to the DHCP.
→ the IP address of DHCP = X.X.X.X
→ DHCP relay agent.

* NAT (Slide 38
- We have 32 bits (can't all be used)
  but we may have up to 2^{32} IP addresses
  ↳ public IP
  ↳ Private IP → free, no need for coordination, non-routable
    ↳ Class A: 10
    ↳ Class B: 172.16 - 172.31
    ↳ Class C: 192.168

Non-routable: Only inside the LAN, why?
because outside the LAN it becomes not unique.

* If we have Src IP = 10.1.1.1 and wants to browse google, it can't browse it because it's private and cannot go through the gateway.



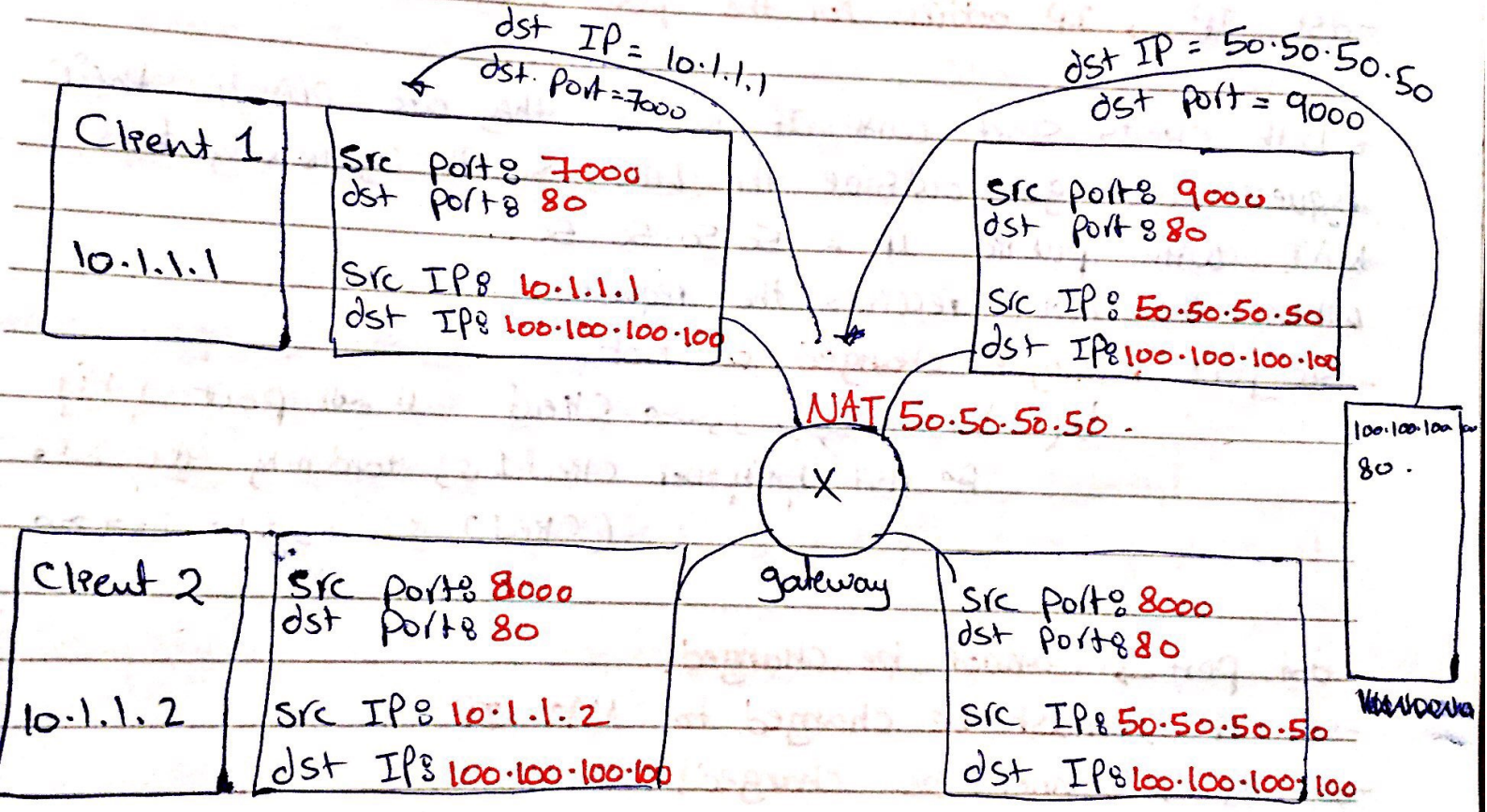
- The problem is that each private IP can only be inside the LAN which is not desired, the solution is NAT.

* شو فكرة ال NAT ؟
فكرة ال NAT إنّه كل ال Private IPs اللي جوا ال LAN
بشوفهم العالم كأنهم one public IP .
لما ال Private IP تطلع بيدل ال Src IP بواحد Public
Packet
ولما ترجع بيدل ال dest IP

* All the private IPs go through the gateway as one public IP when responding to the LAN, how to know each request for those several private IPs from one public IP?

(يعني لو عندي بال LAN أكتر من private IP وكل واحد طلب صفحة بشكل مثلاً (google, facebook)، كل ال requests رح يطلعو ب one public IP لما يجي ال response، كيف بده يعرف مين طلب facebook ومين طلب google وعندي بس one public IP)؟

- The solution is one extra field, port number.



Src IP	Src port	mapped port #
10.1.1.1	7000	9000
10.1.1.2	8000	8000

* شرح المثال السابق

- Client 1 & 2 want to browse a page with IP = 100.100.100.100 listrening on port 80.
each Client has
- Src Port → chosen randomly
- dst port → port # for the page they want
- Src IP → the client's IP address
- dst IP → IP address for the page they want.

* both clients start with IP = 10, so they are private, their request can't go outside the LAN, so the gateway has NAT with public IP = 50.50.50.50.
when the NAT receives the request
- Src port → maybe changed or not
إذا ال port اللي لل Client محجوز بغيره ، رجعها)
واحد تاني randomly زي Client 1 أما إذا ال port مش
محجوز ما بغيره زي Client 2).

- dst port → cannot be changed!
- Src IP → must be changed to NAT IP
- dst IP → cannot be changed!

* after this a table with the mapped port #, src port # and src IP is created in the NAT. (عشان يرجع كل response لل Client)

* the web page responses to the NAT and the NAT checks the table to deliver it to the correct client.

* max # of connections through NAT = 2^{16} TCP connections
↳ why connections not clients ?
because the mapping is based on port number and each connection has different port number

* what is the main reason for the existiness of NAT?
There is not enough IP addresses!

* Disadvantages of NAT:-
1) Single point of failure (إذا الـ NAT خرب كل الشبكة بتخرب)
2) Overhead (because of mapping)
3) Remote connection to a Nated client (ما بقدر أشبك على جهاز جوا الـ LAN من برا)

* Advantages of NAT:-
1) Reduce required # of IPs.
2) Security + (nobody knows the real IPs of internal hosts له بتعامل مع الـ public Nat ما بعرف الـ private)

3) Simplify changing IP addresses (بقدر أغير الـ public IP بدون ما أغير الـ IP للـ clients)

* NAT can have many public IPs not only one.

* Slide 42 حفظ

* Solutions to Remote connection to a NATed Client (Slide 43-45).

1) Static

↳ I add an entry to the table (Src IP, Src port, mapped port) and it stays permenantley in the table, so anyone can access the Client with the public IP and port number.

ex

Src IP	Src Port	mapped Port
10.1.1.2	8000	9000

Public NAT = 50.50.50.50

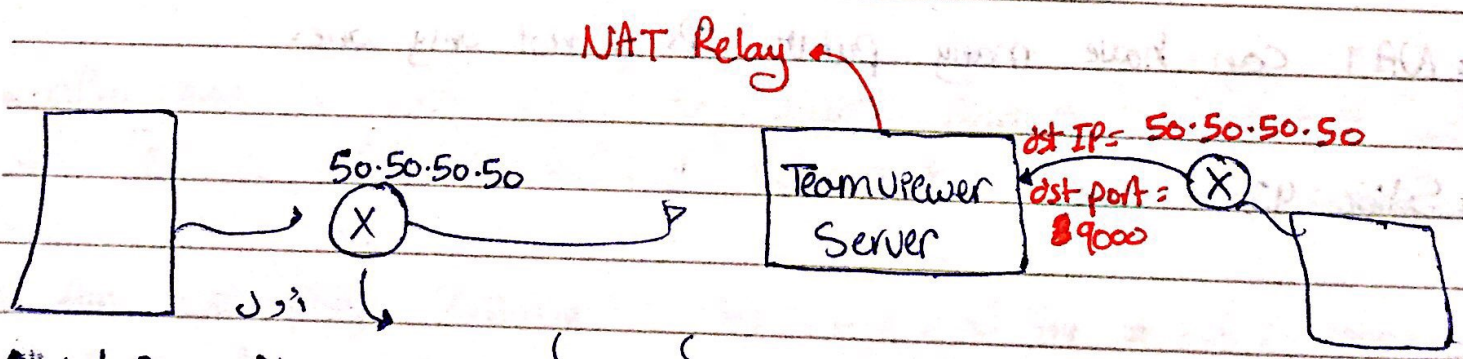
people access 50.50.50.50 with port # = 9000 and with the mapping it reaches the desired client (10.1.1.2, 8000).

2) Plug & Play

↳ Same as static, but client uses a command to put the entry in the table (done by the client).

3) Relay agent

↳ example: Team Viewer.



أول ما أشبك

10.1.1.2 8000 بتخزن

10.1.1.2	8000	9000

9th. Apr

*ICMP: (Internet Control Message Protocol)

- ICMP packet is encapsulated in IP datagram, Therefore, it is assumed that ICMP is on top of IP protocol.
  ↳ mainly used for network debugging and troubleshooting.
  ↳ for error reporting.

*Ping command is a command that sends a packet (small) to the dst and sends a respond.
if a respond arrives then the connection is well (to check an error in the network).

ex: Ping www.yahoo.com
Ping gateway IP
ping local DNS
ping any other PC in the LAN.

* Tracerout → each time TTL expired it sends ICMP msg.

* IPV6 (Slide 49)
  ↳ IPV4 addresses are limited and the wanted IPs are increasing, so price will increase in a huge way. (2^{32} IPs are not enough).
  → Solution is to move to IPV6!
    ↳ (IPV6 has 2^{128} IPs, so it's enough)!

* IPv4 header had some complexities!
  ↳ So they wanted to reduce complexity of IPv6.

* Fixed-length 40 byte header in IPv6 so this reduced complexity.
  - 16 byte src
  - 16 byte dst

- Slide 50

Ver: Version (to determine the format of the packet).
Pri: Priority (like Quality of Service).
Flow label: أزرق لماذا بيعمل (for quality of service).
Payload length: because the payload is of a variable size.
Next header: Upper layer protocol
Hop limit: TTL (Time to Live).

* Checksum, flags and fragmentation are removed in IPv6 but they existed in IPv4.

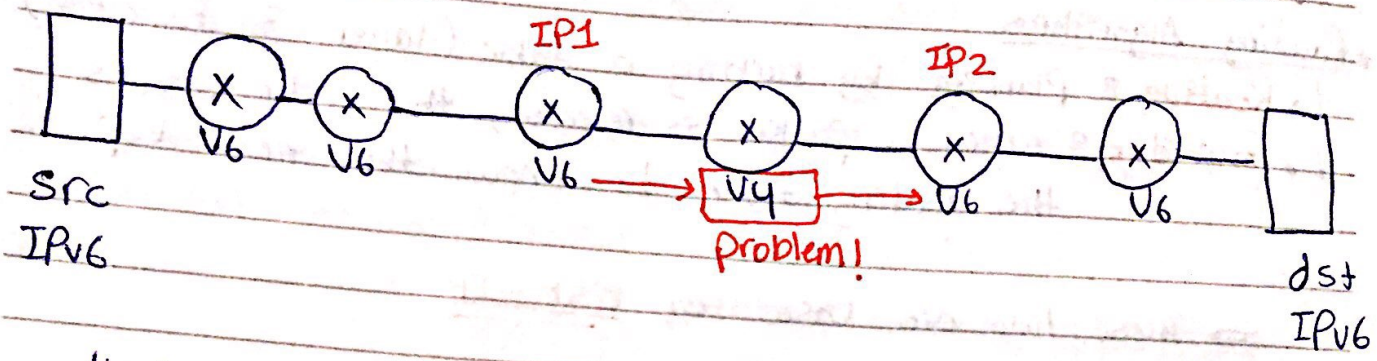
- when do I need fragmentation?
when size of datagram > MTU

- If no fragmentation is used in IPv6, how to deal with the case of datagram size > MTU?
- Send ICMP message "packet is too large"
So the src will reduce the size of the packet.
↳ this type of ICMP msg didn't exist in IPv4
So ICMPV6 was created to include such msgs.

* options are part of the payload.

* If a router supports IPv6 it must support IPv4!
↳ how the router know what version is the packet?
From the "ver" field in the format

- Slide 52



→ the problem happens when some routers support only IPv4, So it may not understand the packet and sends an ICMP msg.

⟹ Solution:
the previous router (IP1) knows that the next router is V4, so it encapsulates the packet inside IPV4 datagram and adds a V4 header with src IP & dst IP

Src IP: IP1 dst IP: IP2	IPV4 datagram
IPv6	→ this is called (Tunneling).

- when the packet arrives to the next router (IP2) it also know that the previous router was V4 so it removes the IPv4 datagram and take IPv6.

Tunneling: it means that IPV6 datagram is carried as payload in IPv4

* Routing Algorithms

↳ **Routing**: Planning by building a table (layer 3 function).

↳ **Forwarding**: when a packet is recieved, the router checks the routing table to know the next hop.

⇒ These two are based on dst IP

- Routing table:

range of IP | interface #

Commands are saved in Hard Disk.

Static

dynamic ↳ the router learns these items.

if no entry matches the table, a msg (dst network unreachable) is sent to the src and packet is discarded.

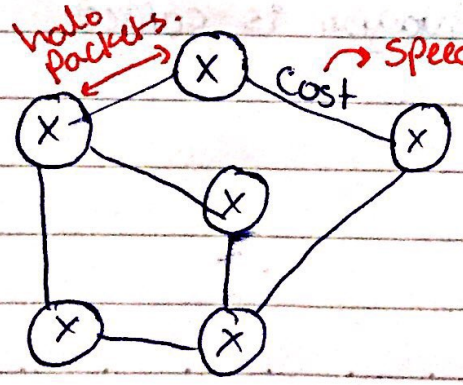
- The table exists in RAM (Volatile).

* Dynamic has two main families
  - ↳ **Link State** → like: OSPF
  - ↳ **Distance Vector** → different types, like: RIP, IGRP

* Routing: we try to model the network as Graph.

$G = (V, E)$ and sometimes cost.

Vertices (routers) ↳ Edges (connections).



- To find a path to any destination

→ Apply shortest path Algorithm.

PAGE 11th.Apr
DATE

* if there is no cost on edges, all links are of the same cost (the same).
  ↳ So the shortest path algorithm is # of hops.
- if there is cost → Σ cost on edges.
Cost could be → distance
→ Constant throughput (because we want min cost).
→ Reliability
→ Security

* Link State: Based on global knowledge.
  ↳ Each node (router) has a complete view of the network. it knows each router connected to which routers and the cost on each edge.

* Distance vector: Based on local knowledge
  ↳ Each router knows part of the information.
  - links costs that are connected to the router
  - Some information about the neighbours.

Problems of link state:-
* Halo packets in link state are huge
* updates exchange on link cost changes (frequent).
  ↳ كل ما تتغير connection لازم كل النتورك تعرف

* Each router uses link state have a full topology of the network.

14th. Apr

- Slide 61

Link State Dijkstra algorithm.

↳ it finds the shortest path (min cost path) from a node to every destination.

↳ it's iterative & multiple iterations, in each iteration the node learns the shortest path to one of the destinations.
(if we have 100 nodes → 99 iterations are needed)

* Routing algorithms: - are implemented on each router while it's trying to build a routing table

* On boot-up, what is inside the routing table?
- Static entries.

* Distance (v) "D(v)":
↳ current value of cost of path from source (the current router) to destination (v).
(Sum of costs of links on the path from the current router to the node v).

* p(v): next hop to v.

* N': Set of nodes that the router knows the minimum cost path to them.

* C(x,y): cost from x to y.
- intially, each router knows the path to neighbours
- if not neighbours, $C(x,y) = \infty$

- Slide 62-65.

- initially the routing table is empty (because we are now dealing with dynamic not Static).
- we have 6 routers (Slide 63) so we need 5 steps

5 Steps because it's iterative and in each iteration it learns the min cost for one of the destinations.

- in Step 0, I only know the cost for myself and I know the links am connected to.
  - ↳ Starting with "u", it's connected to "v", "w" and "x".

$D(v)=7$, $D(w)=3$, $D(x)=5$, $D(y)=\infty$, $D(z)=\infty$

because they are not connected to u

- in the next iteration, I check the minimum $D(v)$ from the previous iteration ($D(w)=3$) and I add it to the list, so now I know the minimum cost from u to w $=3$

*min cost path to each remaining node $= \min\{C_{i-1}, C_{to(w)} + C_{(w) \text{ to the node}}\}$

- min cost to $X = \min\{5, 3+4\} = \min\{5, 7\} = 5$.

- min cost to $Y = \min\{\infty, 3+8\} = \min\{\infty, 11\} = 11$

and this is done in every iteration to each node.

* Distance vector algorithm.

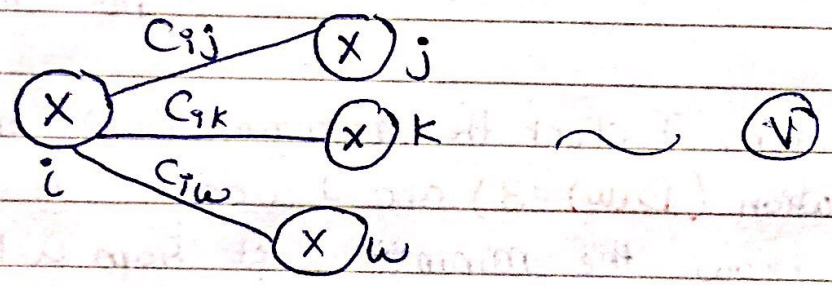
→ Distance vector: the minimum cost from a source to each destination. (Routing Table).

$\{\underbrace{0, 0, 0, 0}\}$

elements: each element is the min cost to one of the destinations.

- in ~~[illegible]~~ distance vector algorithm, each node knows:-
  1) Link cost to neighbours
  2) Distance vectors of neighbours.

*



- min cost path from node (i) to destination (v) =

$$\min\left\{C_{ij} + \text{Cost}_{j-v},\ C_{ik} + \text{Cost}_{k-v},\ C_{iw} + \text{Cost}_{w-v}\right\}$$

⇒ This is called Dynamic Programming (Recursion).

⇒ This algorithm is Bellman-Ford.

* exchanging data in link state is bigger.

- Slide 78

Hierarchical routing

↳ divide the world into Autonomous systems (AS).
↳ each AS could be a country or an ISP.

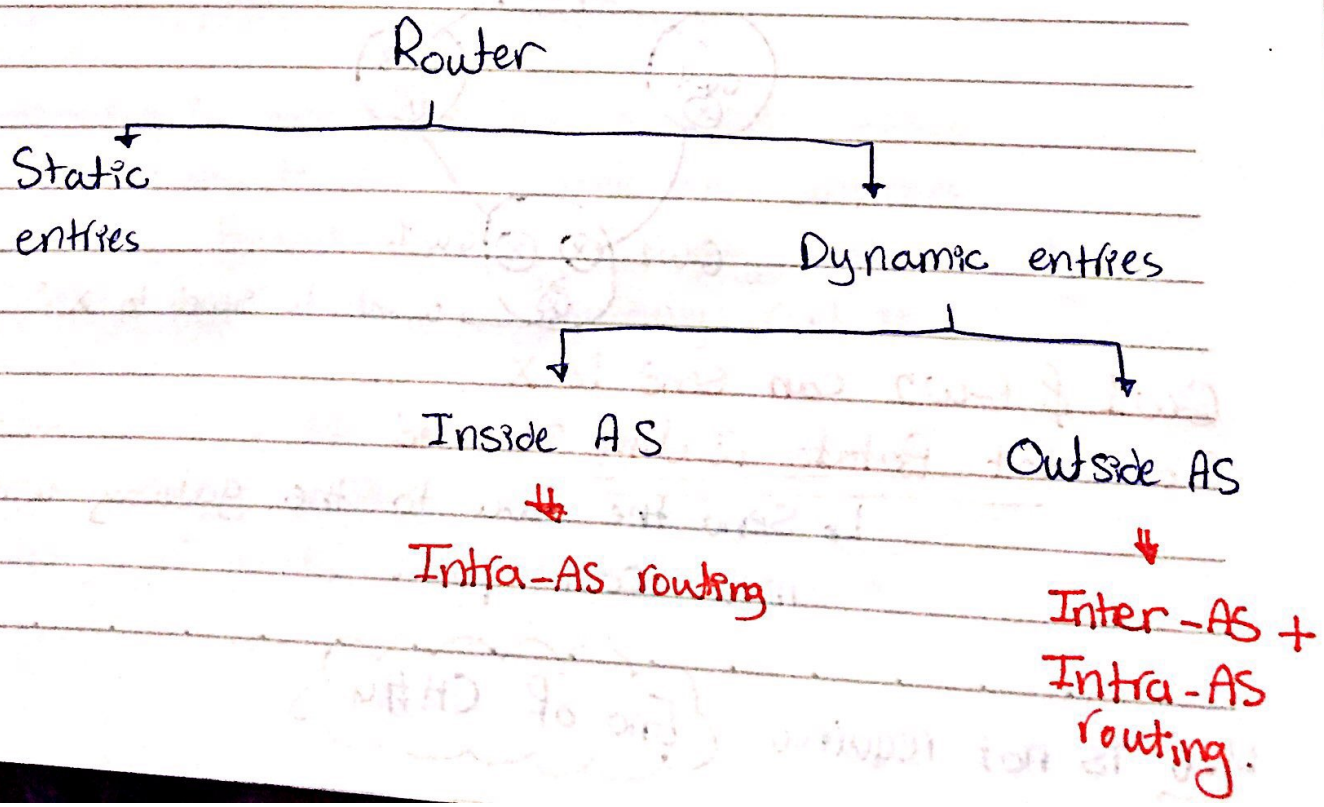
- Two Types of algorithms in Hierarchical Routing:-

1) Inside AS (Intra-AS routing)
   ↳ you try to find a path from any router to any destination inside the AS, or to the gateway router.
      ↳ many routing protocols (RIP, OSPF, IGRP, ---)

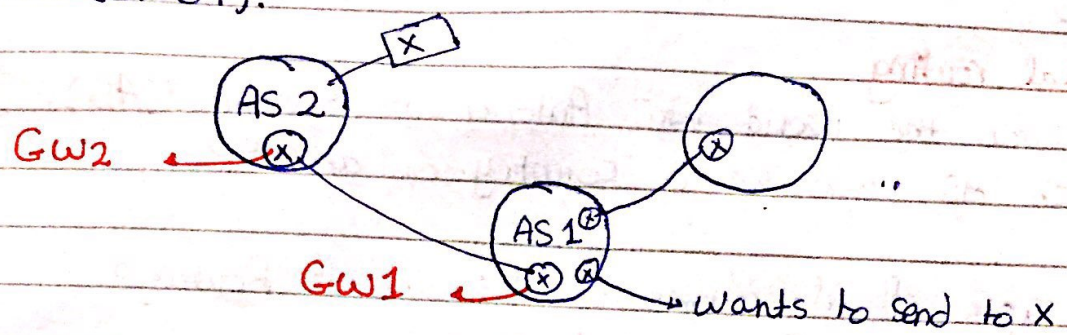
2) Outside AS (Inter-AS routing)
   ↳ there should be a Standard way of routing because the ASs are scattered in the world and managed by different organizations.
      ↳ One single Protocol (BGP Protocol).

* To send any data outside the AS, you need to arrive to the gateway router.

*



- Slide (81-84).

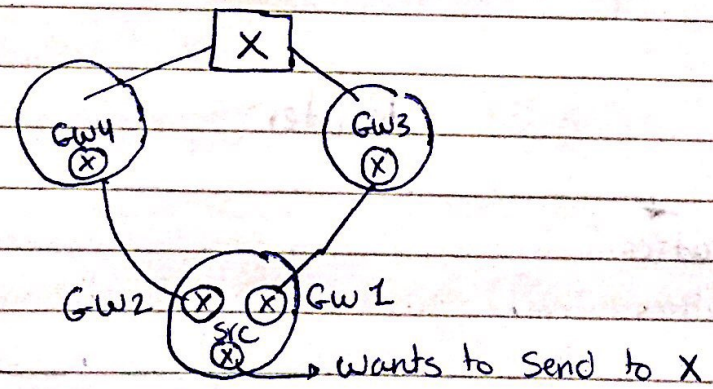


- Intra-AS routing: learns which destinations are reachable through which gateways.

Inter AS routing
- GW1 informs GW2 about all destinations reachable via AS2, then GW2 sends this information to all routers in AS 2

Intra AS routing
Then to send data to X, the router should find the min cost path to GW 2.

- Suppose you want to send data to X and X is reachable from two gateways.

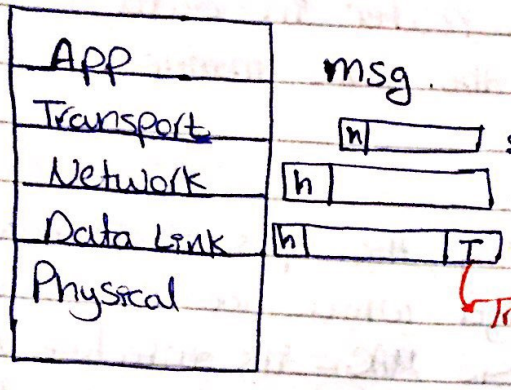


GW1 & GW2 can send to X
So Hot Potato routing is used
↳ Send the data to the gateway with min cost path.

4.6 is not required

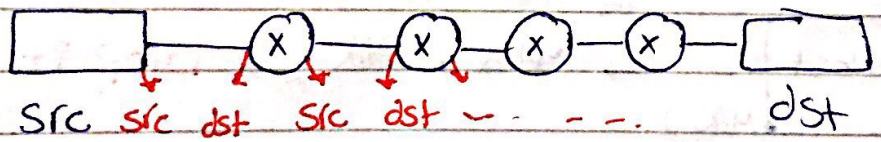
End of CH#4

CH #5 Link Layer.



* **Link Layer:** is responsible about communication between two physically adjacent nodes, and the protocols in this layer are dependent on the link type.
  e.g. Ethernet, WiFi, Fiber, ---.

- In data link layer, the source and destination are the two edges of a hop.



* The addressing in data link layer is MAC Address.
* Layer 2 exists in routers, hosts and switches.
* Switches are transparent devices, they don't modify data link layer header, they only read it.

* Host name $\xrightarrow{\text{DNS}}$ IP address (for translation).
  IP address $\xrightarrow{\text{ARP}}$ MAC
* Data Link layer works only inside the LAN.

- Slide 4

why hosts and routers are now called nodes?

- Because the host and the router in data link layer work exactly in the same manner.

1) * dst IP = IP_x → (X) IP1 → $IP_x \neq IP_1$ ⇒ Routing, then pass it to data link layer which adds new header with src MAC = its interface MAC and dst MAC = next hop interface MAC

dst IP = IP_x → [host] IP2 → $IP_x \neq IP_2$ ⇒ Drop it.

2) *

dst MAC = MAC_x → (X) MAC 1 → $MAC_x \neq MAC_1$ ⇒ Drop it

dst MAC = MAC_x → [host] MAC 2 → $MAC_x \neq MAC_2$ ⇒ Drop it

3) *

dst MAC = MAC_x → (X) MAC 1 → $MAC_1 = MAC_x$ ⇒ pass it to network layer

dst MAC = MAC_x → [host] MAC 2 → $MAC_x = MAC_2$ ⇒ pass it to network layer.

PAGE 18th . Apr
DATE

-Slide 6

-Link Layer services:-

1) Framing: take the datagram from network layer and adds header and trailer to it (becomes frame).

2) Medium Access Control:

↳ Links

↳ Shared: if two or more nodes accessed the link at the same time ⇒ Collision.

- The collision will destroy the packet even if it's in one bit.

⇒ I need Medium Access Control (MAC) who can send at any time to prevent collision.

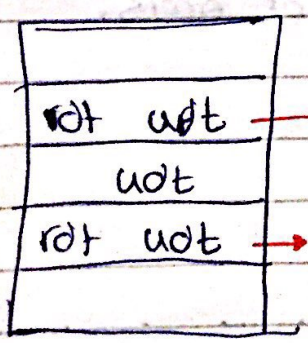
- if link is not shared → no need for Medium Access Control.

* Collision Domain:- the set of nodes that will collide with a node when it sends.

- Is Ethernet cable shared or not?

Not shared, because it has pairs.

3) Reliable delivery between adjacent nodes.



→ the error may happen in the router.

→ detect errors early (no need to wait until final destination to detect the error).

- why do we have rdt in data link layer?
to detect errors early with no need to wait until it reaches the final destinations.

- why do we have rdt in transport layer although we had rdt in data link layer?
Because Network layer has only udt to an error may happen in the router, so transport layer will detect it.

- Which is more reliable, Fiber or Ethernet?
Fiber

- which is more reliable, Ethernet or WiFi?
Ethernet.

- which is more reliable, WiFi or Satellite?
WiFi

4) Flow Control
↳ between adjacent nodes.

5) Error detection
6) Error correction
↳ for error detection and correction we need redundant bits.
↳ Size of redundant bits depends on number of errors.

7) Full-duplex : Ethernet.
Half-duplex : WiFi

PAGE
DATE

- Slide 8
- Data link layer is both software and hardware.
  - ↳ Hardware: Network Interface Card (NIC)
  - ↳ Software: NIC identification + protocols.

- Error detection and correction.
  - ↳ we need redundant data
  - ↳ Size of redundant data depends on number of errors detection ....

- Error detection: 1) Parity checking (Single bit, 2D bit)
  2) Checksum
  3) Cyclic Redundancy Check (CRC).
     - ↳ up to 3 bits correction.

⊛ Details are not required.

*Multiple Access Protocols:
Slide 17
- type of links:
  1) Point-to-Point: - only one is listening
     - no collision
     - no need for medium access control
  2) broadcast (shared): - more than one is sending data
     - there is collison
     - there's need for medium access control.
     - e.g.: old-fashioned Ethernet.
       - ↳ Because hop was used, it's layer 1 device which takes the data and send it to everyone.

21st .Apr

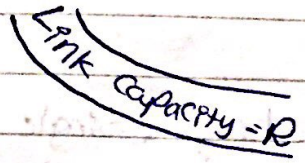
Slide 18

* Requirements of Medium Access Control:
  1) Distributed: no central authority that performs coordination.

  2) no out-of-band control channel.

- Slide 19

* M nodes in the same collision domain



1) when one node wants to transmit, it sends at rate R
2) when M nodes want to transmit, rate = R/M

These two are easy to achieve but under one condition which is having a centralized node.

3) fully decentralized
   ↳ this makes the ideal scenario hard.

- Slide 21

Channel Partitioning MAC Protocols.
   ↳ TDMA: Time Division Multiple Access
   ↳ FDMA: Frequency " " "
   ↳ CDMA: Code " " "

Problem: because we don't have centralized point, some slots may not have data but still have time or frequency.

- no collisions may happen.

* Random Access Protocols:- (most used).
  ↳ how to detect collision
  ↳ how to recover from collision

-Slide 24

- Slotted ALOHA
  ↳ all frames are of same size
  ↳ time divided into equal size slots (each time slot is enough to send one frame).

  ↳ nodes start transmitting only at slot beginning.
  ↳ nodes are synchronized (same clock) (disadvantage).
  ↳ if two nodes transmit at same slot, they may have collision.

  - when collision happens, each node will send with probability (P) until success.

* Probability (P): random # between (0) & (1)
  if # > P → don't send
  if # < P → send.

* Max efficiency = 37%

- Slide 27

- Pure (Unslotted) ALOHA
  - ↳ no need for time slots (to get rid of synchronization)
    - ↳ but it increases collision.

* Max efficiency = 18 %

- Slide 29

Carrier Sense multiple access (CSMA)
- ↳ listen before transmit
- ↳ very popular.

- CSMA/CD (Collision Detection).
  - ↳ if sent = received ⇒ no collision
  - if sent ≠ received ⇒ there is collision. (Stop transmission).

- How to recover from collisions?
  - ↳ Use Binary exponential backoff time.
    - ↳ after ~~nth~~ n^{th} collision,
      - wait time = random between $(0 - 2^m - 1)$ * time to send 512 bits.
      - m is number of collision.
      - m is up to 10 (11 collisions → Random (0-1023))
    - ⇒ much better performance.

* efficiency close to 1

- CSMA/CA (Collision Avoidance)
  - ↳ Request-to-send
  - ⇒ no collision.

PAGE DATE 23rd. Apr

- Taking Turns

1) Master node & Slaves nodes to transmit in turns.
   ↳ not desirable.

2) Token passing & small packet that is continuosly rotating in a network. The node that has the token is allowed to send data.

* Cable access network not required!

- Slide 42

- What's the need for MAC address?
  - IP address keep changing, so we need a permanent address → MAC Address.

- MAC address is given to NIC
  ↳ it's permenant
  ↳ physical Address.

- MAC Address (48 - bits) (in Hexadecimal).
  12 Hex Digits → Unique.
  - manufacturer ID
  - Card ID.

- Slide 43 - 46
- ARP works inside the LAN
- Learning inside the LAN is always done through flooding (broadcast)

*ARP Table IP { MAC { TTL
↳ because IP may change.

*broadcast ⇒ dest MAC = FF-FF-FF-FF-FF-FF
- ARP is Plug & Play → it means no need for configuration

- Slide 47
how A knows the IP address of first hop router (R)?
from DHCP, but DHCP doesn't learn MAC address only IP address.

- Slide 48
Dst MAC is for the gateway.
when the router sees that the MAC address is for it it takes the frame, but the IP is not for it, so it does routing.
MAC src becomes MAC address of gateway.
MAC dst becomes B MAC address.

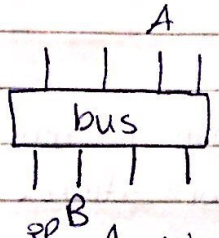
* Ethernet:

- ↳ very cheap
- ↳ reliable, good } → very popular.
- ↳ speed up to 10 ~~up~~ Gbps.

*

(a bus topology).

- takes the data and send it to everyone

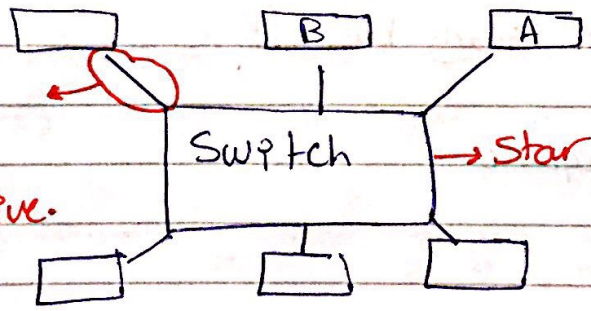


if A wants to send to B A sends to the bus and the bus sends it to B.

- in bus and hup all nodes are in the same broadcast domain.

- Switch was discovered then.

2 pairs:
a pair to send
a pair to receive.



→ Star topology, Layer 2 device.

- Switch table

MAC	interface #
A	1
B	2

*Advantages of switch:-

1) Reduce collision domain
2) Cheap
3) Self-learning device (Plug & Play).
   ↳ the switch learns the switch table.

-Slide 56

Preamble → 8 bytes with pattern
- for synchronization.

CRC → Error detection.
Type → higher layer protocol.
Payload → datagram.

Slide 58

- Connectionless
- Unreliable
- CSMA/CD with binary backoff

⊛ Slide 59 not required

* Core Switch: directly connected to the gateway router
directly connected to the server.

- What is the difference between switch & router?
1) Switch works on layer 2 while router on layer 3
2) Switch learns by flooding but router learns from exchanging hello packets with other routers.

* VLANS

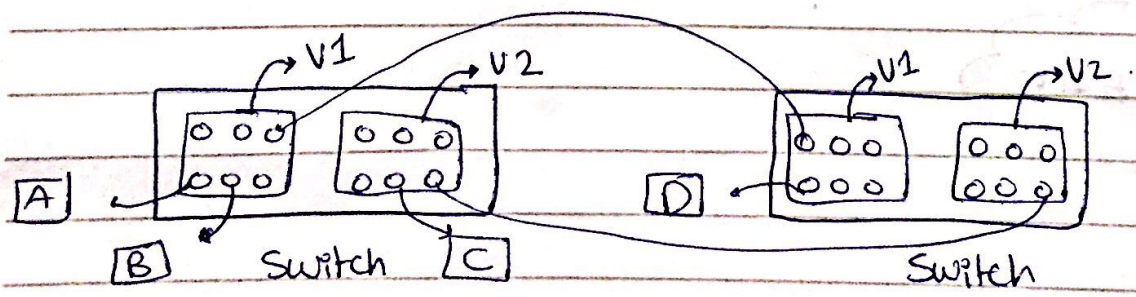
Slide 71 - 75

* We want to divide the LAN into sub-LANs without routers.
→ Virtual LANS.

- when you flood any frame in the LAN, it arrives to all hosts in the LAN. (VLAN) (VLAN)

- when two nodes in different LANS want to communicate they need routing. (VLANS).

* To do routing in VLANS → Router
→ Layer 3 switch



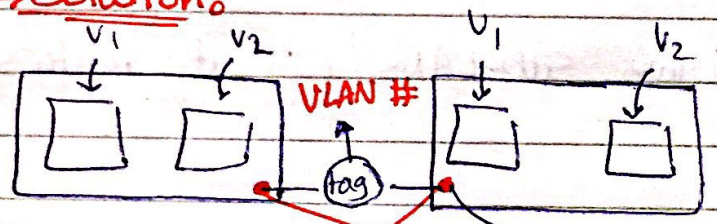
- if A wants to talk to B → if the switch doesn't know where B is, flooding in V1 to all ports except the one sent the data.

- if A wants to talk to C → Routing is needed!

* a VLAN may be on several switches not on a single switch.

- if A wants to talk to D → it can ✓
but what is the problem in this?
If I have 50 switches, I need 50 ports to connect the VLAN through the switches.

⇒ Solution:



Trunk port → belongs to all VLANS.

→ reads the tag and floods to all ports

* the Switch sends to all devices in VLAN except the one that sent the request plus to the Trunk port.

* The protocol that performs tagging (802.1Q)

End of CH#5