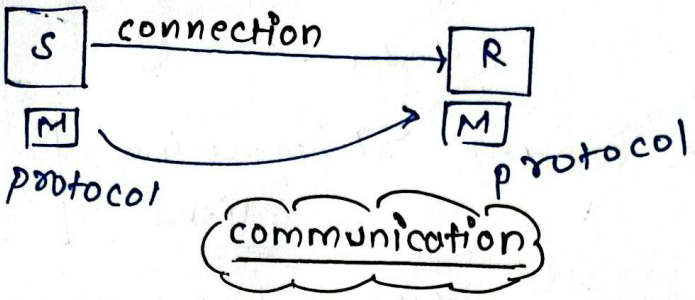
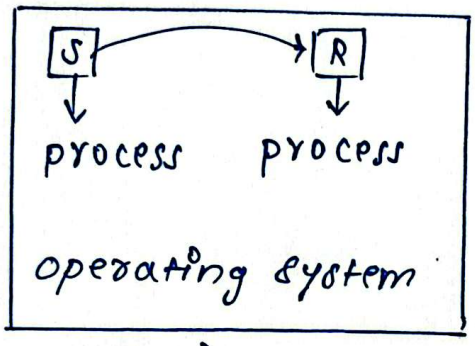
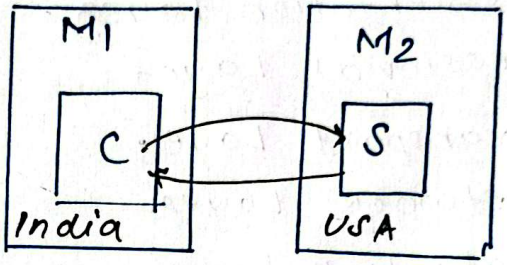
Computer Networks

Main purpose of computer networks to send data.

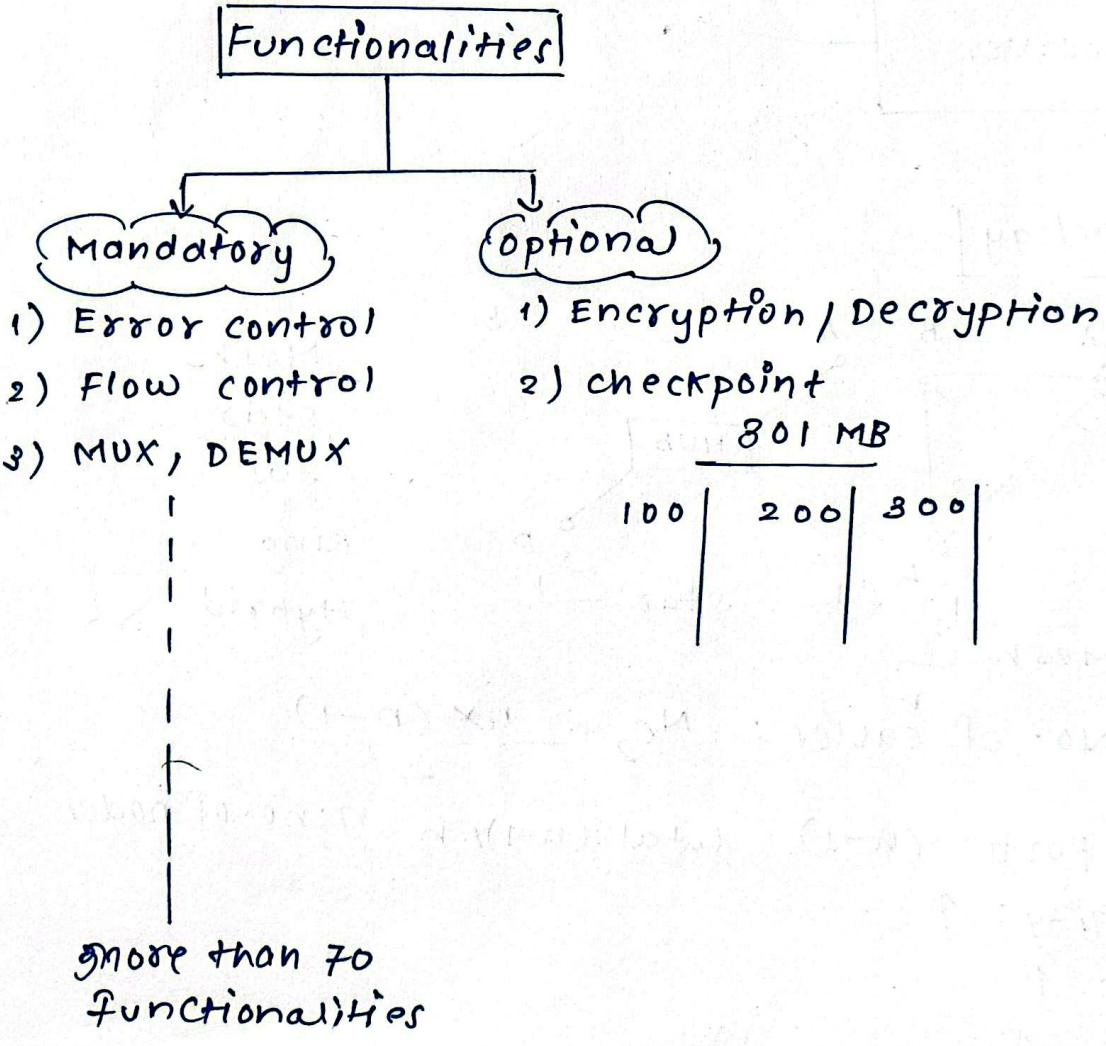




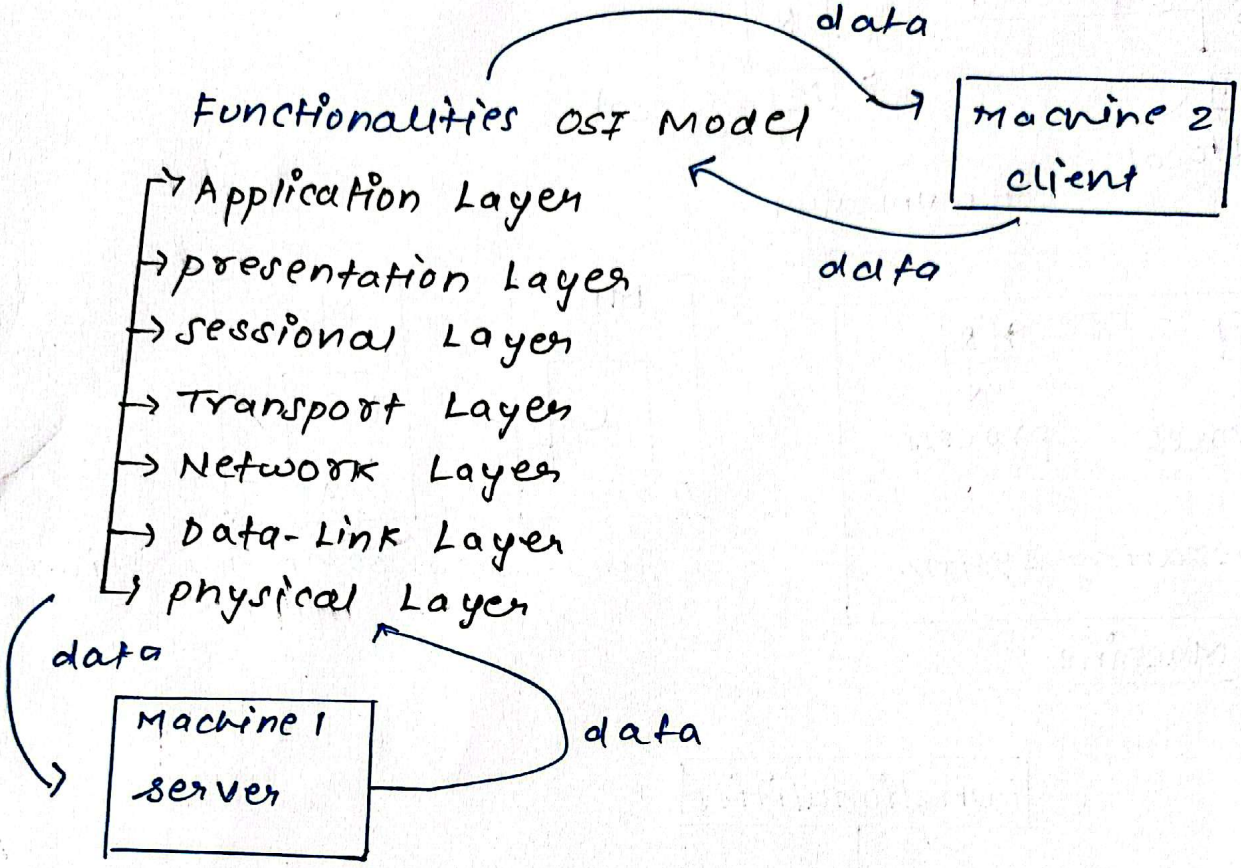
Machine



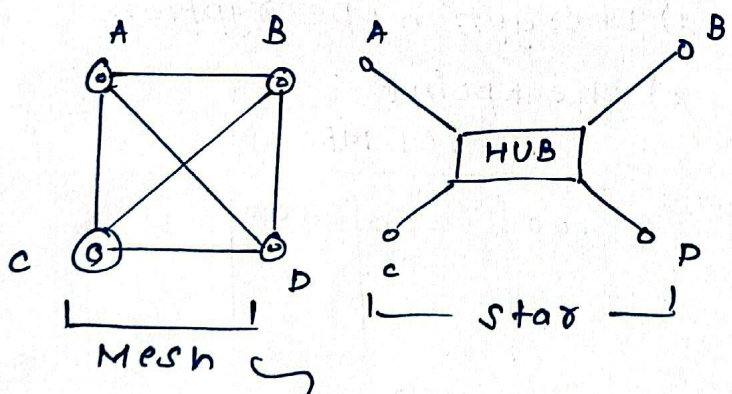
Two way communication



All the Above functionalities (optional & Mandatory) are compiled into one common imaginary model known as OSI model (open system interconnect)



Topology



Mesh
star
BUS
Ring
Hybrid

Mesh

No. of cables : $^{n}C_{2}$: $\frac{n*(n-1)}{2}$

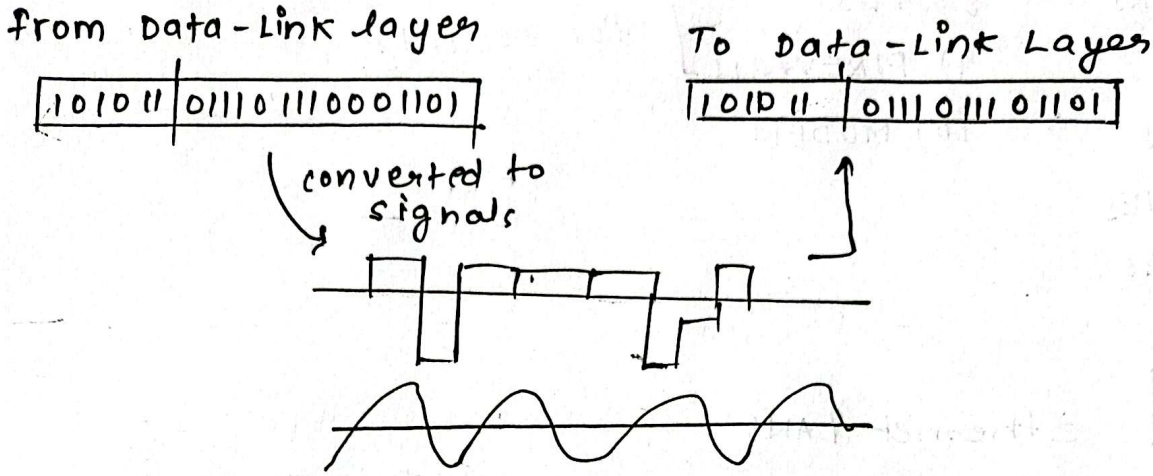
No. of ports: (n-1) total :(n-1)*n n: no. of nodes

Reliability: ↑

cost : ↑

security ↑

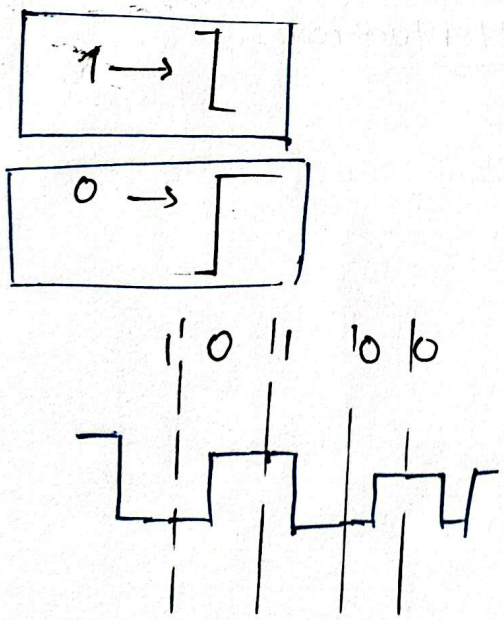
PHYSICAL LAYER.



Functionalities of physical Layer and its characteristics

- → Cables and connectors
- → physical Topology
- → Hardware (Repeaters, HUB)
- → Transmission mode
- → Multiplexing
- → Encoding

"Manchester vs Differential Manchester Encoding"

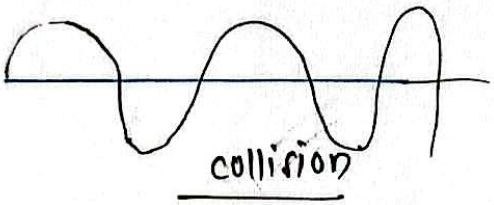


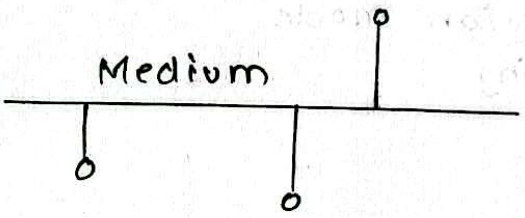
Various Devices in Computer-Networks

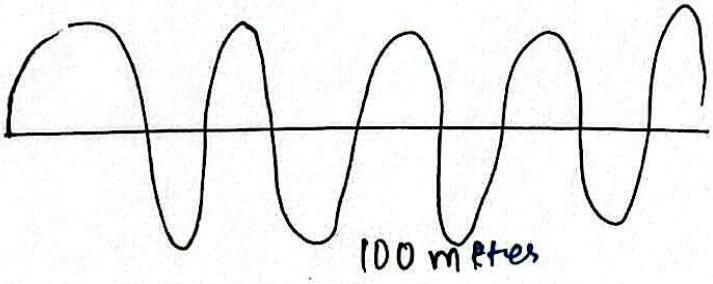
1) CABLES
2) REPEATORS
3) HUBS
4) BRIDGES
5) SWITCHES
6) ROUTERS
7) GATEWAY
8) IDS
9) FIREWALL } for security
10) MODEM

CABLES Ethernet LANS

- Unshielded twisted pair cables → 10 Base T, 100 Base T
- Coaxial cable → 10 Base 2 (10 MBPS, Base band) / 10 Base 5 (Broad band)
- Fiber optic – 100 Base Fx

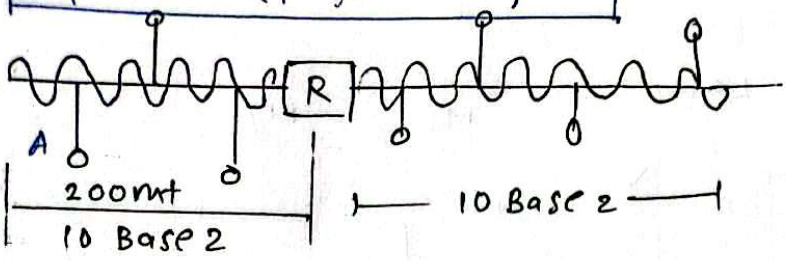






Attentuation

Repeaters (physical layer)



10 mbps – Bandwidth

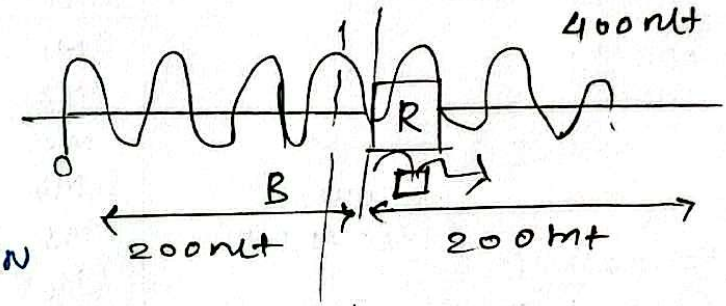
2 – 200mt

function of Repeater is to regenerate strength

x :→ $2x$ (Amplifier)

Repeater x → [R] → x

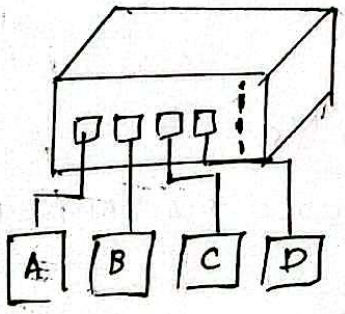
Attenuation is filtered

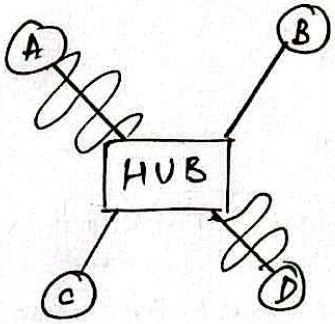


characteristics

→ 1) 2-PORT DEVICE
→ 2) FORWARDING
→ 3) NO-FILTERING
→ 4) COLLISION DOMAIN

HUB (PHYSICAL LAYER)



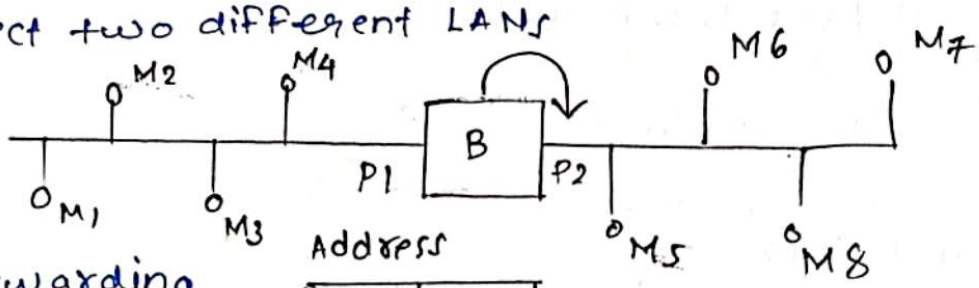


characteristics

→ MULTIPORT REPEATER
→ FORWARDING
→ NO FILTERING
→ COLLISION DOMAIN

BRIDGES (PHYSICAL AND DATA LINK LAYER)

→ connect two different LANs

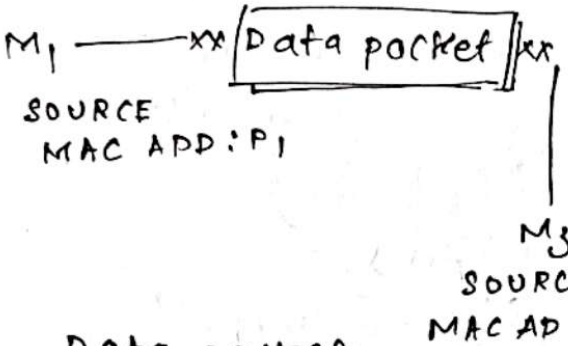


Address

MAC	MAC
SOURCE	DST

→ Forwarding
→ Filtering
→ collision Domain
→ Bridge Data Unit Protocol

Types → static, Dynamic OR Transplent



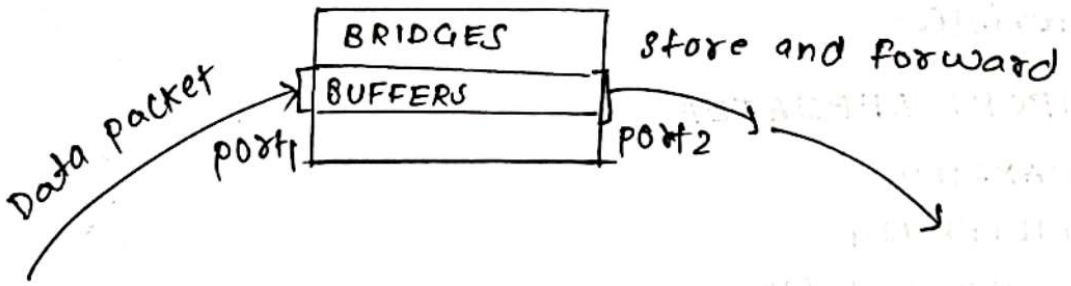
Data packet
Not send for unnecessary traffic.

MAC ADDRESS	PORT
M1	P1
M2	P1
M3	P1
M4	P1
M5	P2
M6	P2
M7	P2
M8	P2

But if

$M_1 \longrightarrow M_5$

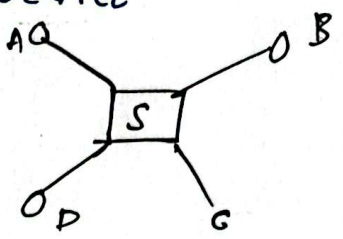
Data packet sent because source MAC ADD: P1 to MAC ADD: P2



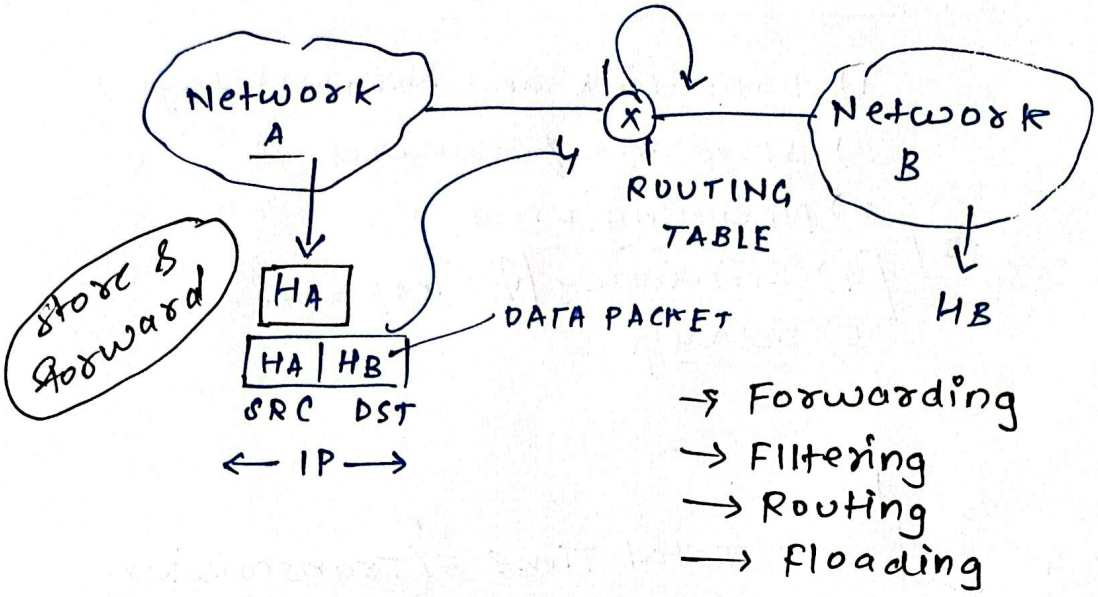
Because due to the Buffers present inside of Bridges there is negligible or No collisions in data-packets receving and sending.

SWITCH

→ LAYER-2 (Data-Link-Layer) Device
→ MULTIPORT BRIDGE
→ FULL DUPLEX LINKS
→ TRAFFIC IS MINIMAL
→ COLLISION DOMAIN IS 0



ROUTERS (PHYSICAL, DATA-LINK, NETWORK LAYER)

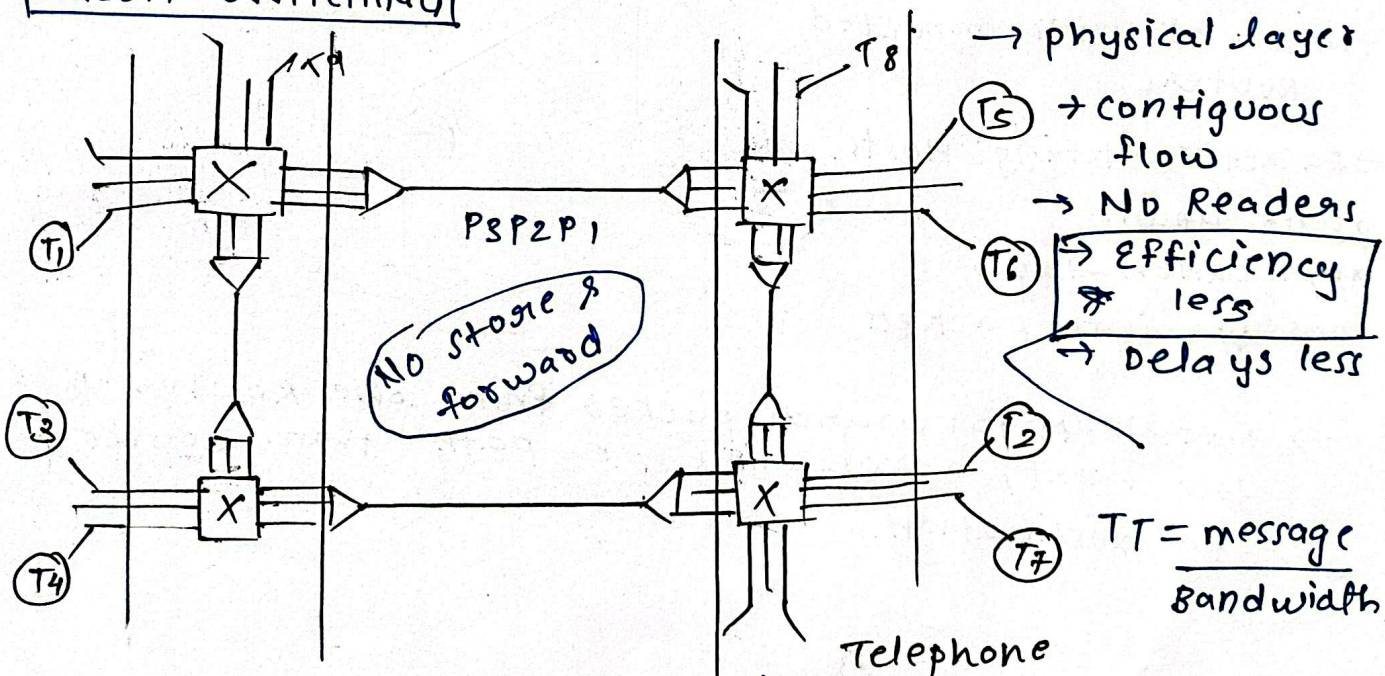


→ Forwarding
→ Filtering
→ Routing
→ Floading

COLLISION DOMAIN VS BROADCAST DOMAIN

Sr. NO.	Device Name	collision Domain	Broadcast Domain
1.	Repeater	No change	No change
2.	HUB	No change	No change
3.	Bridge	Reduce	No change
4	switch	Reduce	No change
5.	Router	Reduce	Reduce

CIRCUIT-SWITCHING



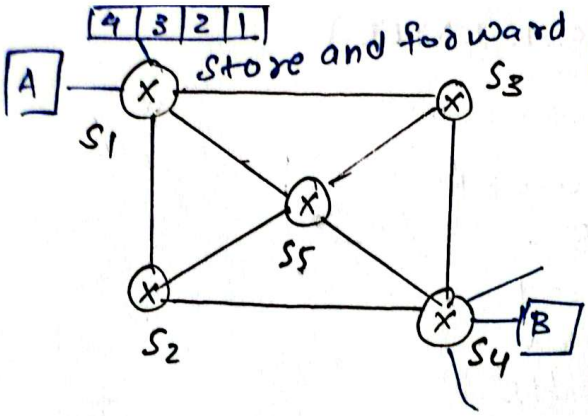
→ physical layer
→ contiguous flow
→ No Readers
→ Efficiency less
→ Delays less

TT = message / Bandwidth

propagation speed = D/v

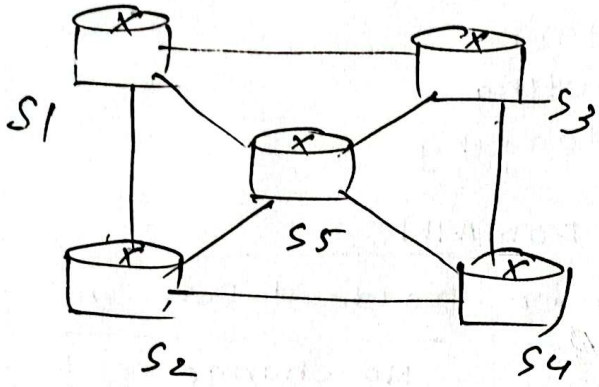
Total time = Setup time + TT + PD + Tear Down time

Transmission time



PACKET SWITCHING

1) Data-Link and Network Layer
2) Store and forward
3) pipelining used
4) Efficiency ↑ Important
5) Delay ↑



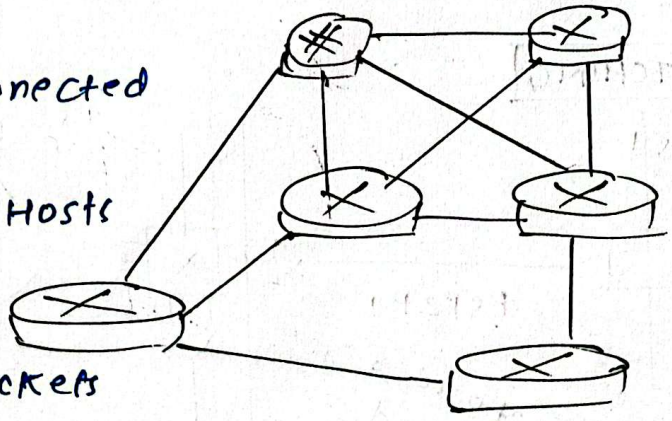
Total Time = $\sum_{n}$ (Transmission Time + Propagation Delay)

Network core : packet/circuit switching, internet structure.

Network core

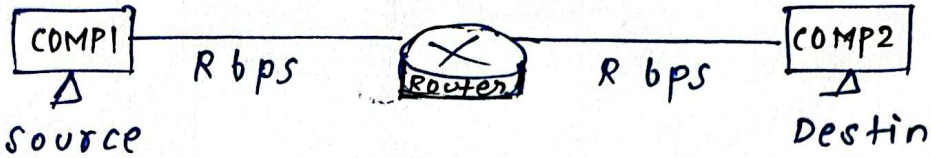
Mesh of interconnected routers

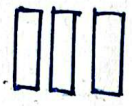
→ packet switching: Hosts break ~~down~~ application-layer messages into packets



→ Network forwards packets from one router to the next, across link on path from source to destination.

packet switching : store-and forward





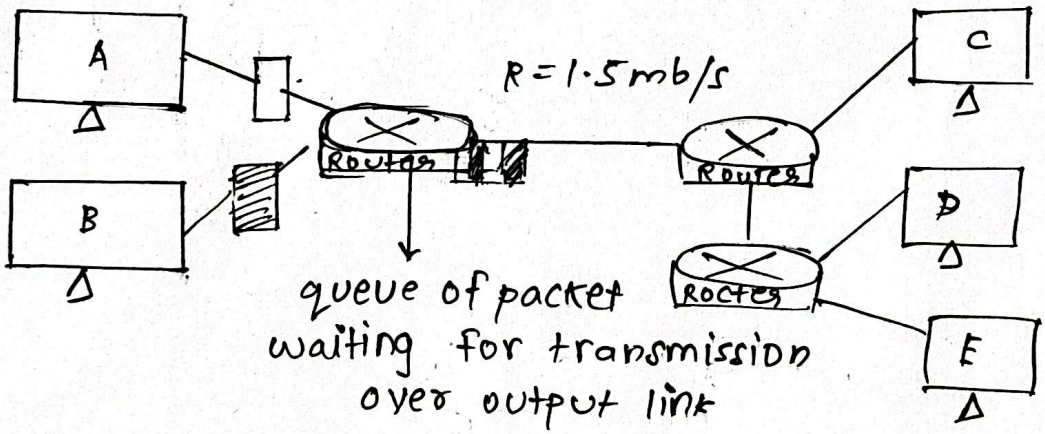
store forward : entire packet must arrive at router b/e it can be transmitted

packet transmission delay : takes L/R seconds to transmit (push out) L-bit packet into Link at R bps.

$L = 10$ Kbits $R = 100$ Mbps

packet transmission Delay $= L/R = \frac{10^{+4}}{10+6}$

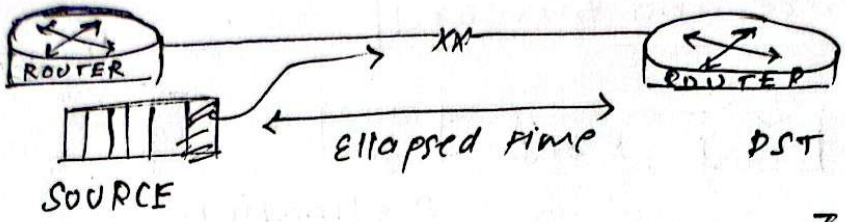
$10^{-2} = 0.01$ seconds



packet queuing and loss : If arrival rate (in bps) to link exceeds transmission rate (bps) of link for some period of time:

packets will queue, waiting to be transmitted on output link.

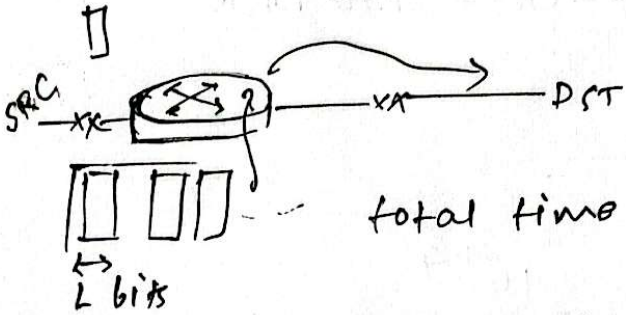
packet can be dropped (lost) if memory (buffer) in Router fills



Q1) Total time = setup time + TT + P.P

Transmission time = $\frac{D}{V} = \frac{L}{R}$

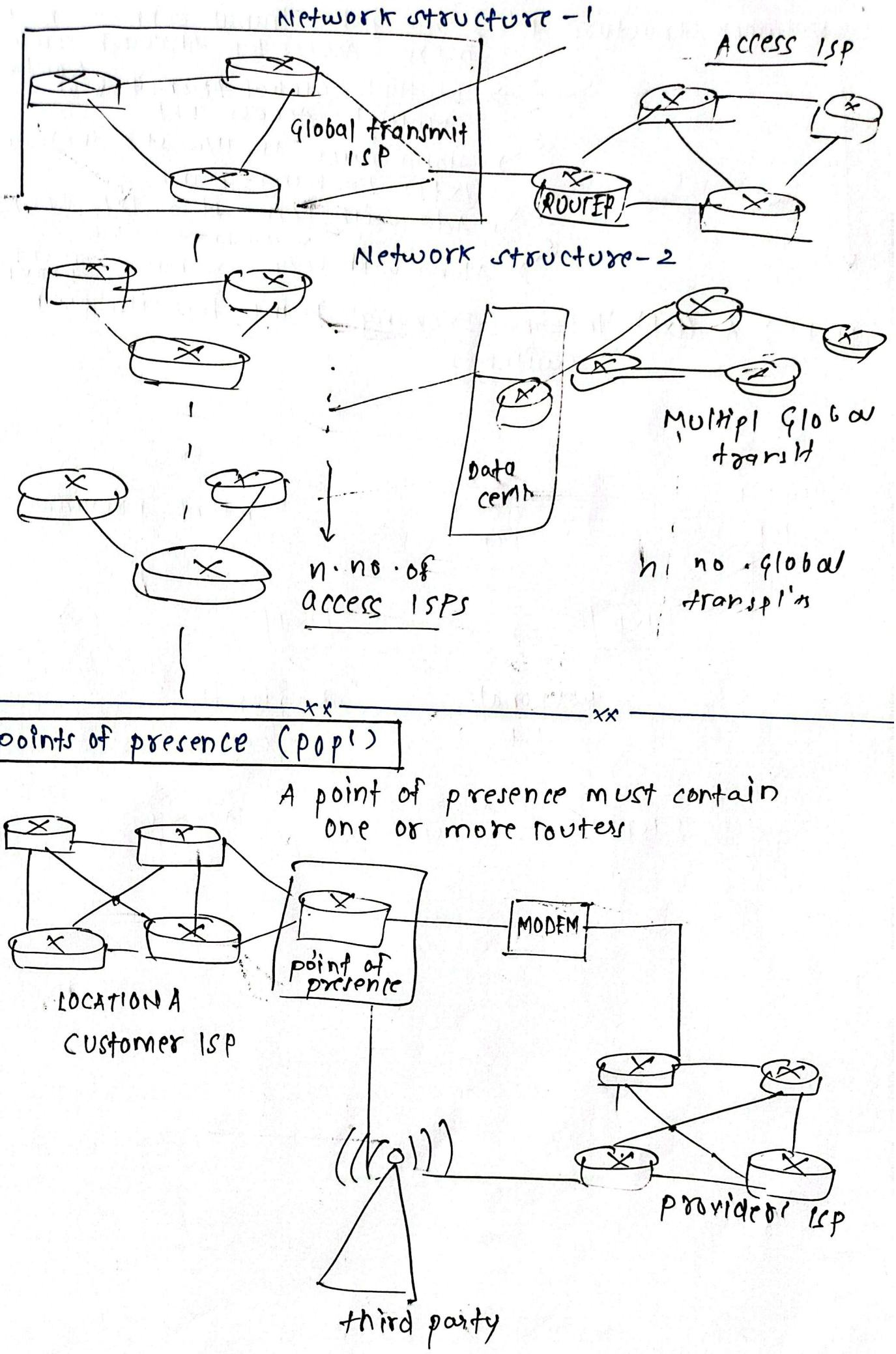
total time = setup time + $\frac{L}{R}$ = L/R



total time = T.T + P.P

Q2) transmission time = $\frac{3L}{R}$

total time = $\frac{3L}{R} + P.d$

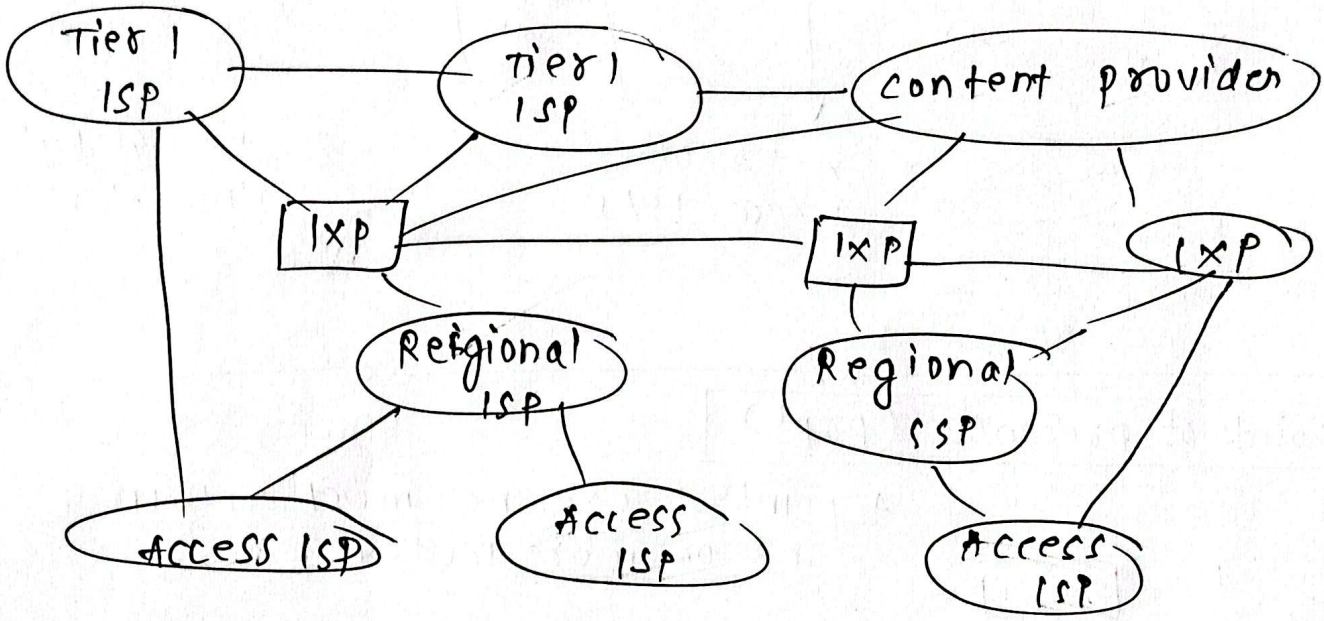


points of presence (popl)

A point of presence must contain one or more routers

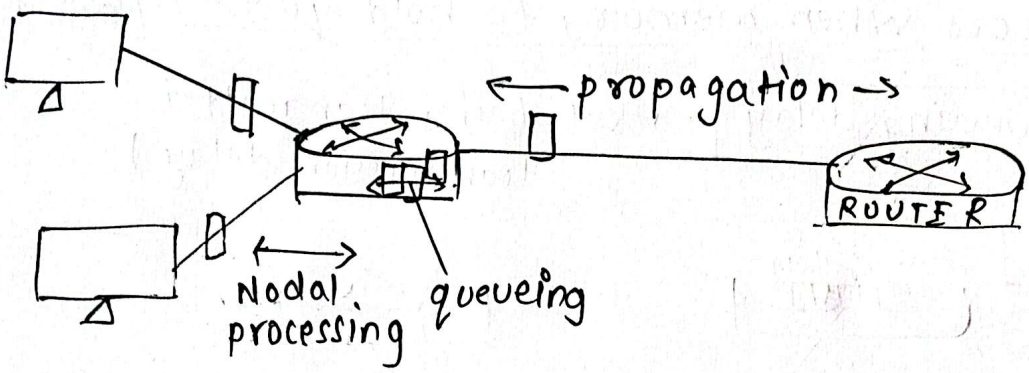
Network structure
1 → one Global transit connect one or more Access Isp through common Routers
2 → Multiple Global transit " "
Multiple Access ISP
3 → Along same as N/w st 2 there are
4 → (IXPs) Exchange point
→ Add with N/w st 3 IXPs, ISPs, POPs — " "
5 → Along with st/4 we have content provider.

Note : A (IXP) Internet Exchange point has its own switches



Transmission Delay : packet can only be transmitted only after all packets have arrived before it has been transmitted.

propagation Delay The propagation speed depends on physical medium of the link.



$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

d_{proc} : nodal processing
check bit ro errors
determine output link
typically < microsecs

d_{queue}: queing delay
time waiting at output link for transmission
depends on congestion level of router

d_{trans}: transmission delay
L: packet length (bits)
R: link transmission rate (bps)

$$d_{trans} = L/R$$

d_{prop} : propagation delay
d: length of physical link
s: propagation speed

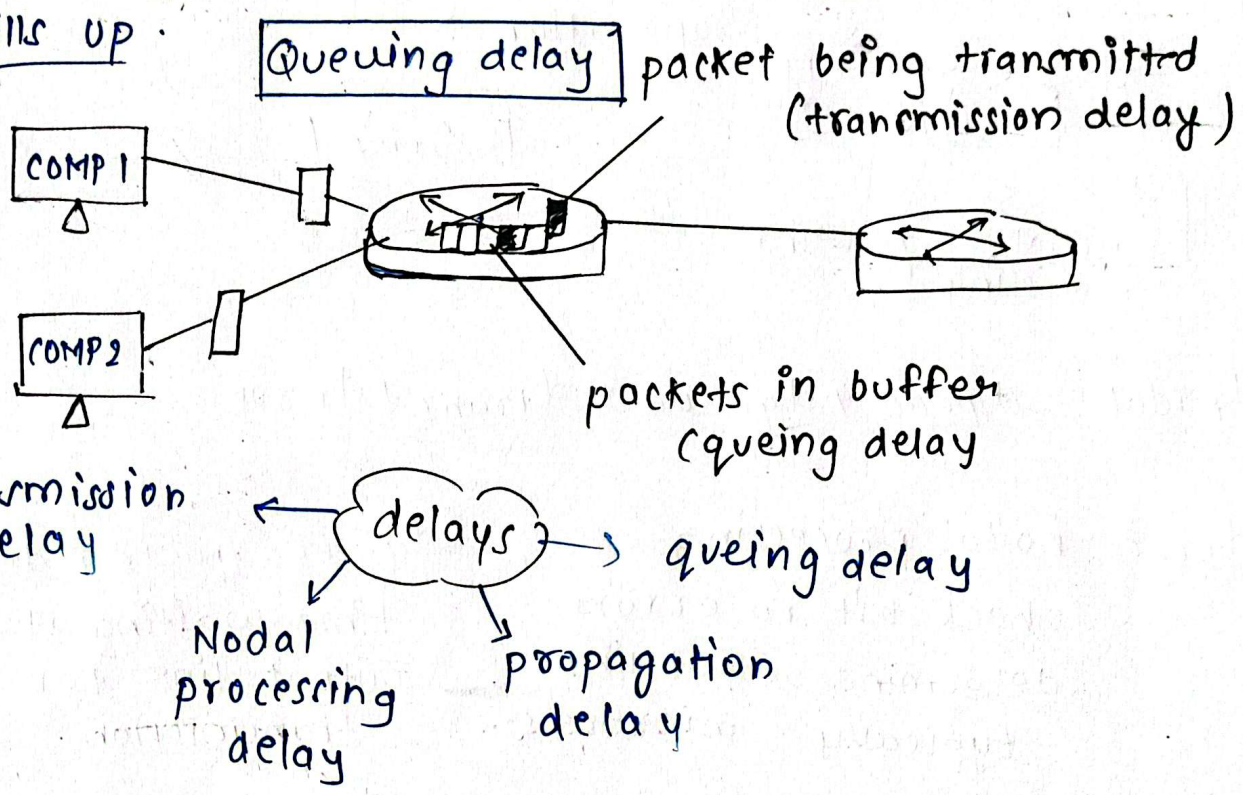
$$d_{prop} = d/s$$

How do packet delay and loss occur?

packet queue in router buffers, waiting for turn for transmission

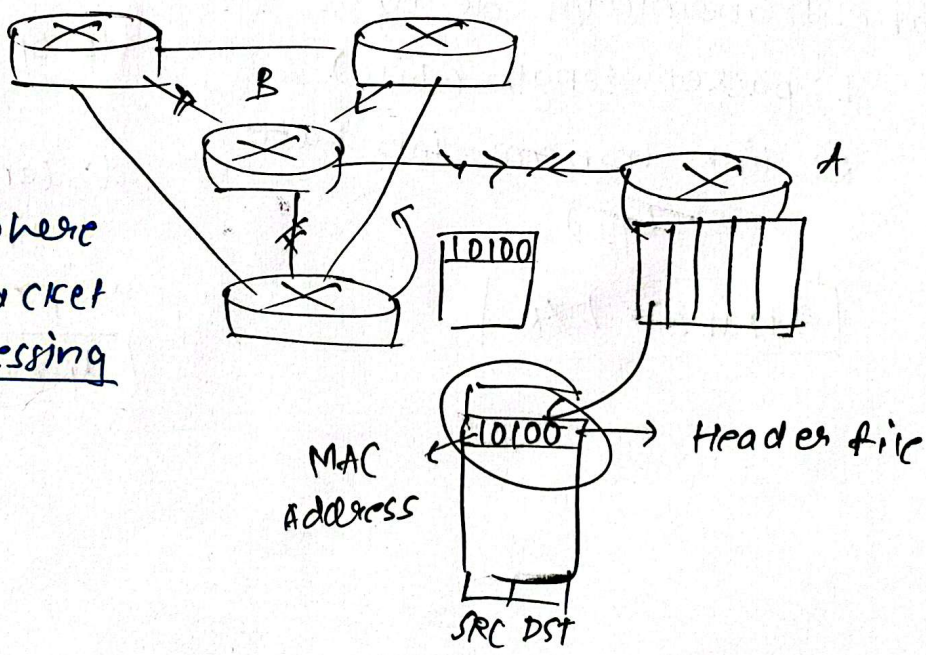
→ queue length grows when arrival rate to link (temporarily) exceeds output link capacity

→ packet loss occurs when memory to hold queued packet fills up.



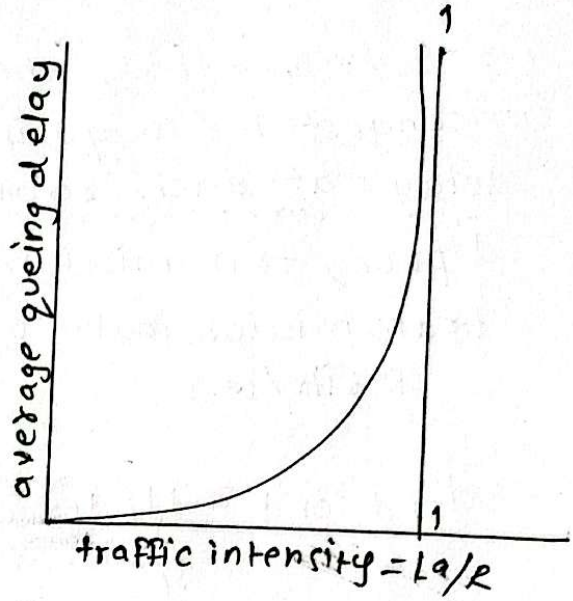
PROCESSING DELAY

Time required to examine the packet's header and determine where to direct the packet is part of processing delay.



defining the terms

Transmission ""	propagation delay
Transmission delay is amount of time required to push out the packet: its function of packet length nothing to do with distance b/w two routers	propagation delay is a time it takes a bit to propagate from one router to next; function of distance b/w two routers. nothing to do with packet length.



packet Queueing delay

a: average packet arrival rate : unit is packets/seconds

L: packet-Length

R: Link Bandwidth bits/s

$\frac{L \cdot a}{R}$ $\frac{\text{arrival rate of Bits}}{\text{service rate of Bits}}$

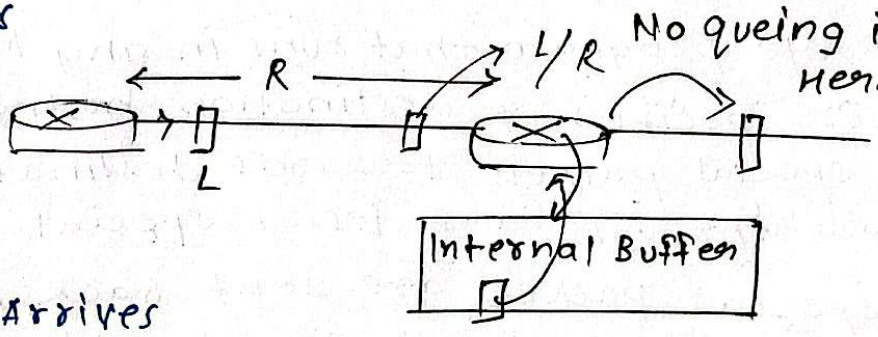
La/R ≈ 0 avg. queuing small

La/R → 1 avg. queuing delay large

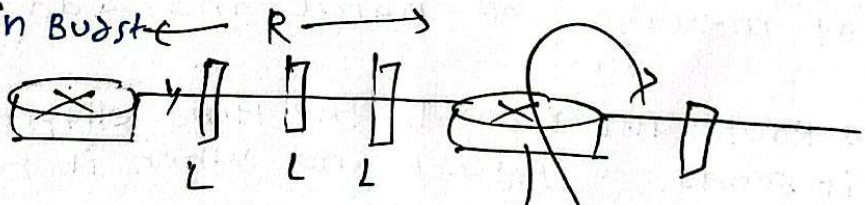
La/R > 1 more work than it can be serviced

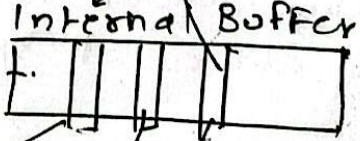
case of packets

IF $\frac{La}{R} \leq 1$



La/R → 1 Arrives in Burst





$n-1\left(\frac{L}{R}\right)$ delay

2L/R

L/R

queing is seen

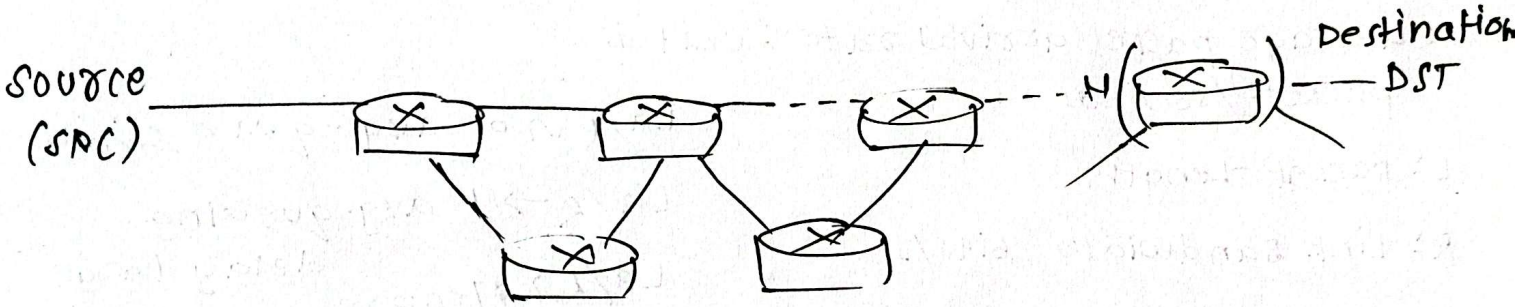
No queing delay

REAL INTERNET DELAY AND END-TO-END DELAY

Let us Assume that, the moment that network is uncongested (so queing delay ≈ 0) the processing delay at each router and at source host is d proc, transmission rate out of each router dproc transmission rate out of each router source host is R bits/secs

$$d_{end-end} = N(d_{proc} + d_{tran} + d_{prop})$$

N: No. of ~~routers~~ Links

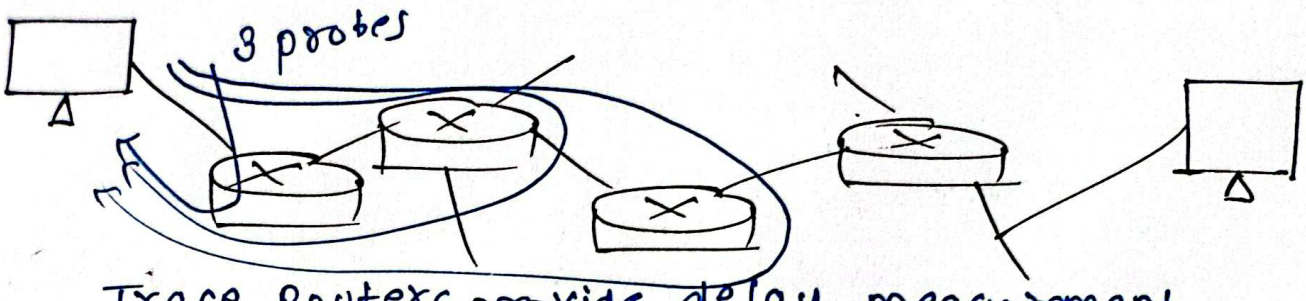


TRACE ROUTE

Trace Route program that runs in any Internet host when user specifies a destination hostname, program sends special packets towards destination. As they pass through a series of special routers these special packets are sent back to source with a short message of name and address of Router

→ Trace Route also records the time ellapses between when it sends a packet and when it receives the corresponding return message: it also records the name and address of the router

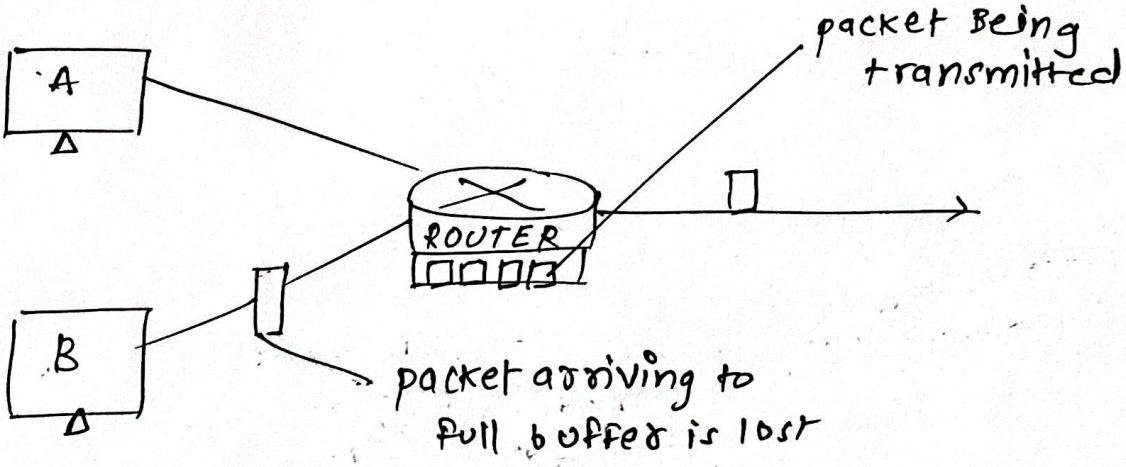
→ Trace Route actually repeats the experiment three time so the source actually sends 3.N packets to destination.



Trace Routers provide delay measurement from source to Router end to end for all i

1) A Router has finite capacity
2) Queing capacity depends on Router design and cost
3) A packet can arrive to find full queue, with no place to store, it can drop ∴ packet will be lost.

PACKET LOSS

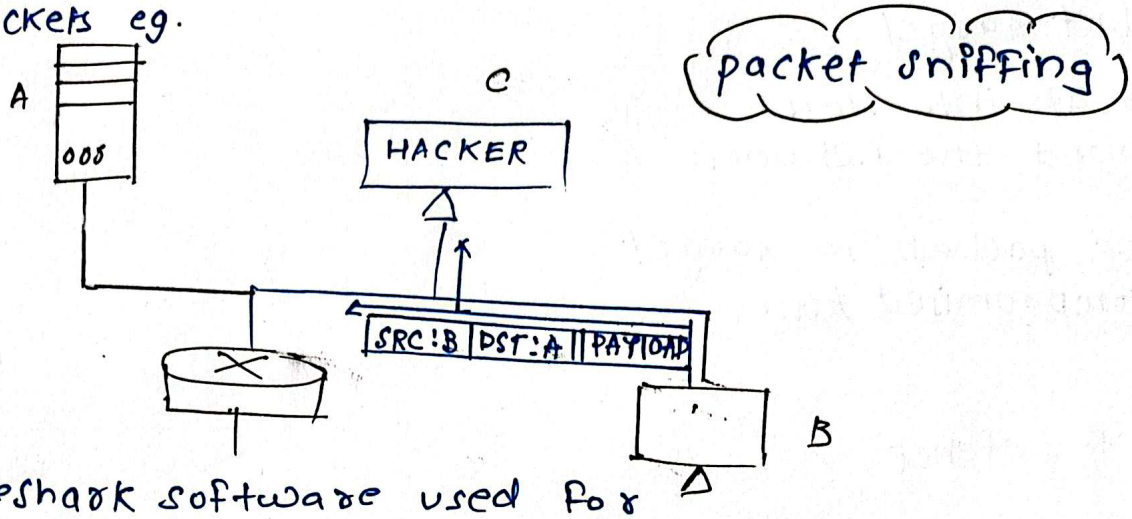


THROUGHPUT CONCEPT

NETWORK SECURITY

packet "sniffing"

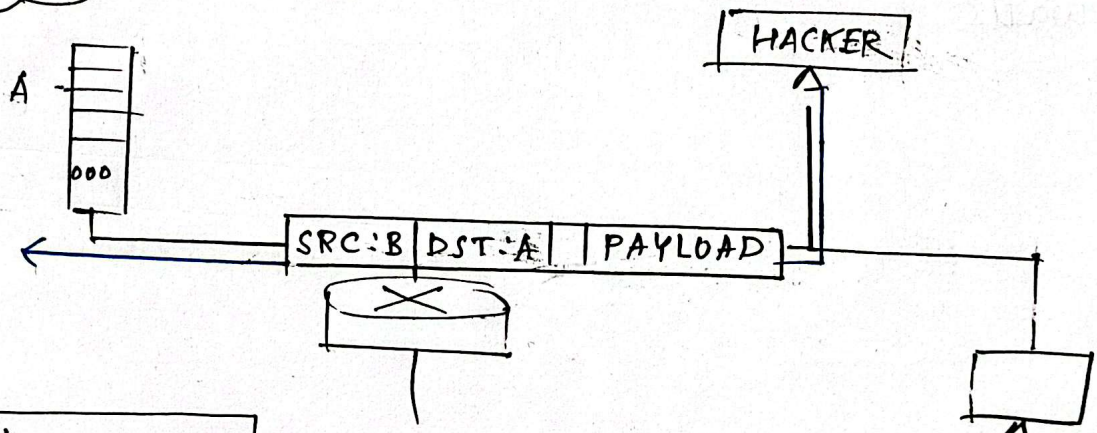
Broadcast media (shared Ethernet, wireless)
promiscous network interface reads/records
all packets eg.



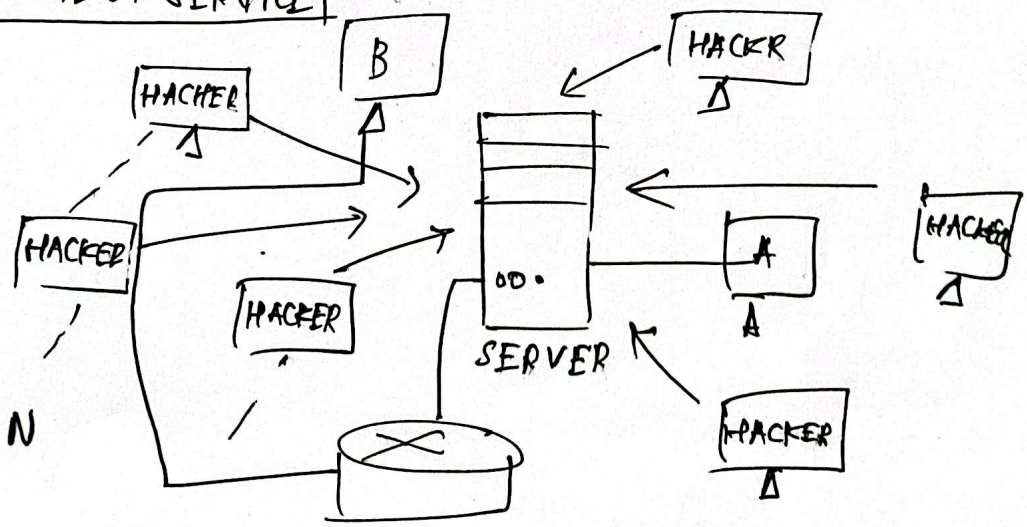
Wireshark software used for
end to end chapter labs is a free packet sniffer

BAD GUYS : FAKE IDENTITY

IP spoofing: injection of packet with false source address



DENIAL OF SERVICE



attackers make resources (server, bandwidth) unavailable to legitemate traffic by overwhelming resource with Bogus traffic

1. select target
2. Break into Hosts around the network
3. send packets to target compromised Host

Lines of Defense

1) Authentication
2) Confidentiality
3) Integrity checks
4) Access Restrictions
5) Firewalls

Layer internet protocol stack

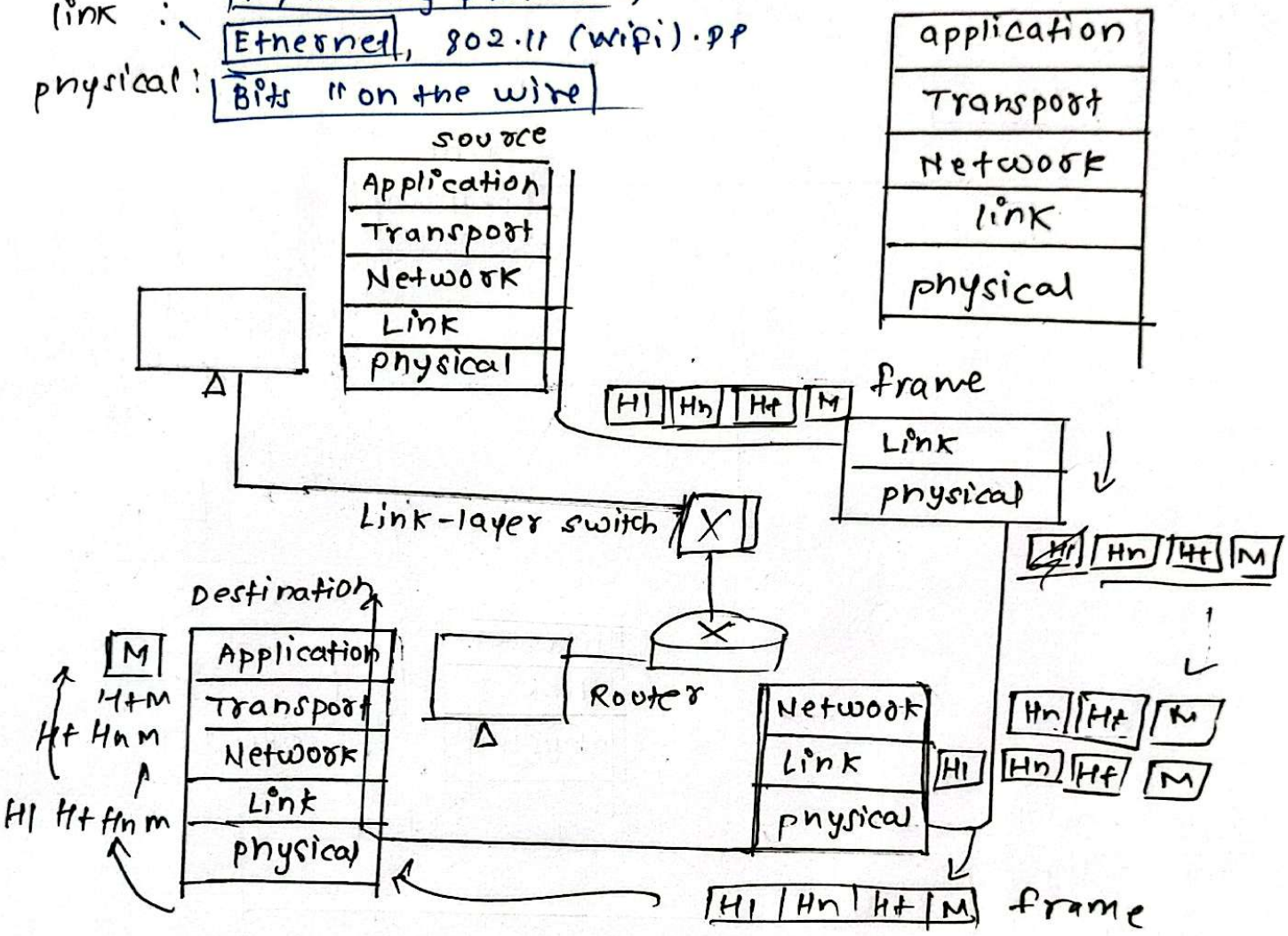
application : supporting network applications
(HTTPS, IMAP, SMTP, DNS)

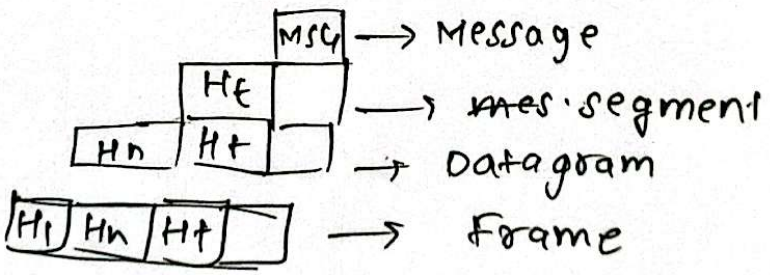
Transport : process-process data transfer (TCP, UDP)

Network : Routing of datagrams from source to dst
(IP, Routing protocols)

link : Ethernet, 802.11 (wifi), PP

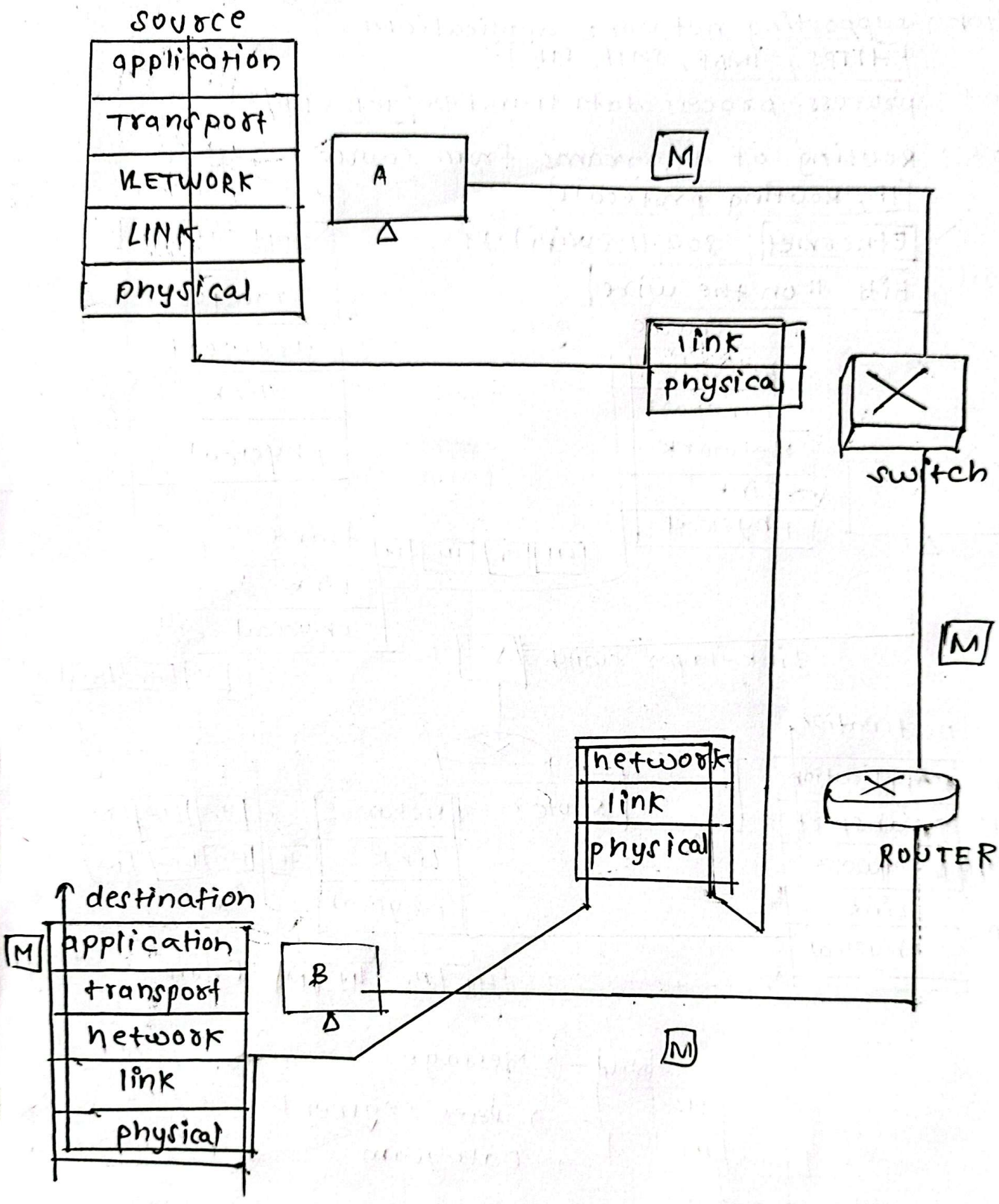
physical : Bits "on the wire"





t → transport
n → network
l → link

Encapsulation : An End-End view



An application-layer protocol defines

→ Types of messages
→ Message syntax
→ Message semantics
→ Rules
→ Open Protocols
→ proprietary protocols : Eg. Skype, Zoom

can be asked in midsem MCQs

Transport service requirements

Application	Data-loss	Throughput	Time-sensetive
file transfer / Download	no-loss	elastic	No
E-mail	no-loss	elastic	No
Web-Documents	no-loss	elastic	No
Real time-audio / video	loss-tolerant	audio : 5Kbps - 1Mbps video : 10Kbps - 5Mbps	Yes 10's msec
streaming audio / video	loss-tolerant	same as above	Yes few secs
interactive Games	loss tolerant	Kbps +	Yes 10's msec
Text messaging	No-loss	elastic	Yes and No

Internet application and Transport protocols

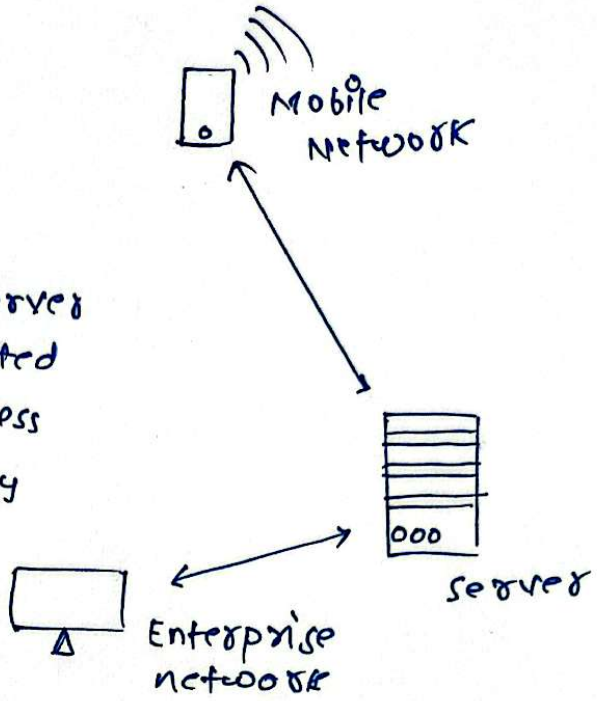
	application-layer protocol	Transport protocols
file transfer / Download	FTP [RFC 959]	TCP
E-mail	SMTP [RFC 5321]	TCP
Web documents	HTTP [RFC 7230, 9110]	TCP
Internet telephony	SIP [RFC 3261] RTP [RFC 3550]	TCP or UDP
streaming audio/vid	HTTP [RFC 7230], DASH	TCP
interactive Games	WOW, FPS ✓	UDP / TCP

CLIENT–SERVER PARADIGM

server: Always on host
permanent IP Address

clients
→ contact, communicate with server
→ May be intermittently connected
→ May have dynammic IP address
→ Do not communicate directly with each other

Example:- HTTP, IMAP, FTP



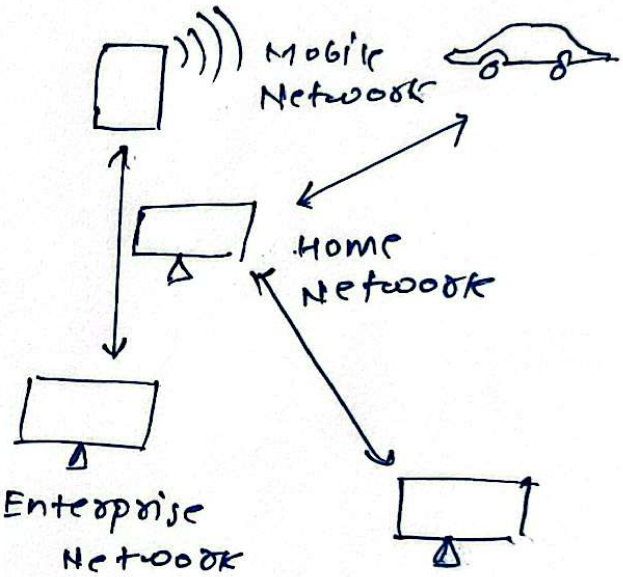
PEER–PEER ARCHITECTURE

→ Not always-on server
Arbitrary end-systems directly communicate

→ peers request service from other peers, provide service in return to other peers

→ peers are intermittently connected and change IP address
· complex management

example: P2P file sharing [Bit Torrent]



Internet Transport protocols services

TCP service

Reliable transport: Between sending and receiving process

Flow control : sender won't overwhelm receiver

congestion control : Throttle sender when network overloaded.

connection-oriented : setup required between client and server process

does not provide : Timing, minimum throughput guarantee, security

UDP servies

Unreliable data transfer between sending and receiving process

does not provide : Reliability, flow control, congestion control, timing, throughput gurantee, security, or connection setup.

WEB and HTTP

Web pages consists of objects stored in different web servers

They are referenced by addresable by a URI

www.manipal.edu / CSE Dept / manipallogo.gif

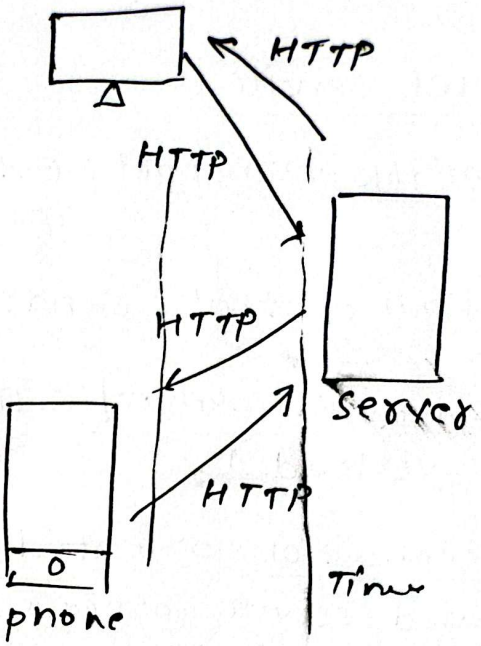
Host-Name. path name

HTTP : Hyper text transfer protocol

Web's application-Layer protocol

client : Browser that requests
Receives, (using HTTP protocol)
and display web object

server : Web server sends using
HTTP protocol objects
in request



HTTP USING TCP

Client establish TCP connection to server port 80

↓

Server Accepts TCP connection from client

↓

HTTP messages exchanged b/w browser msgs and web server

↓

TCP closed

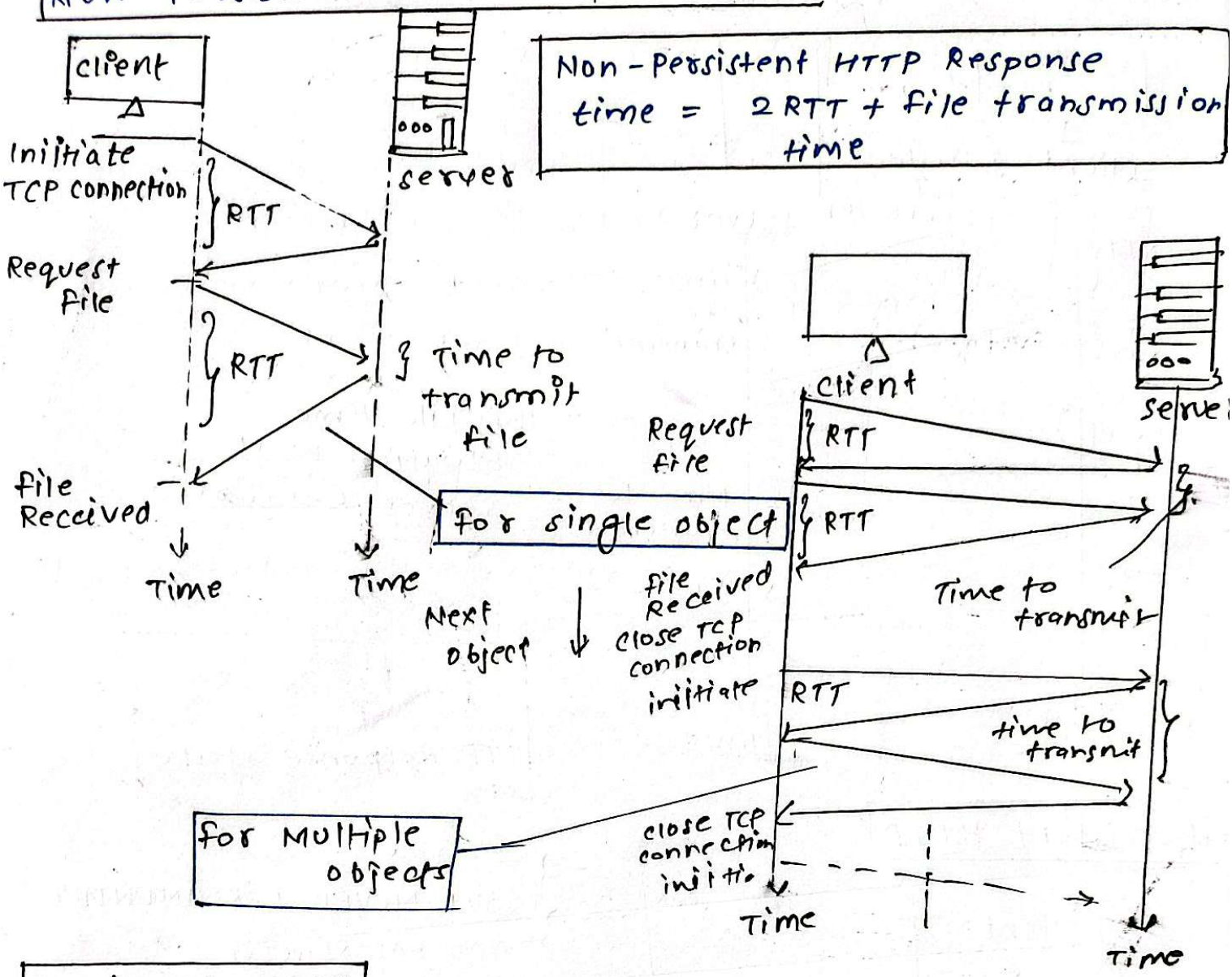
HTTP connections → Persistent
→ Non persistent

Note:

HTTP is stateless does not contain No info about past client

Non-persistent HTTP	persistent HTTP
1. TCP connection opened	TCP connection opened to a server
2. At Most one object sent over TCP connection	Multiple objects can be sent over a single TCP connection between client and that server
3. TCP connection closed	TCP connection closed.
Note: Downloading multiple object require multiple connections at TCP	

Non-Persistent HTTP: Response Time



persistent HTTP

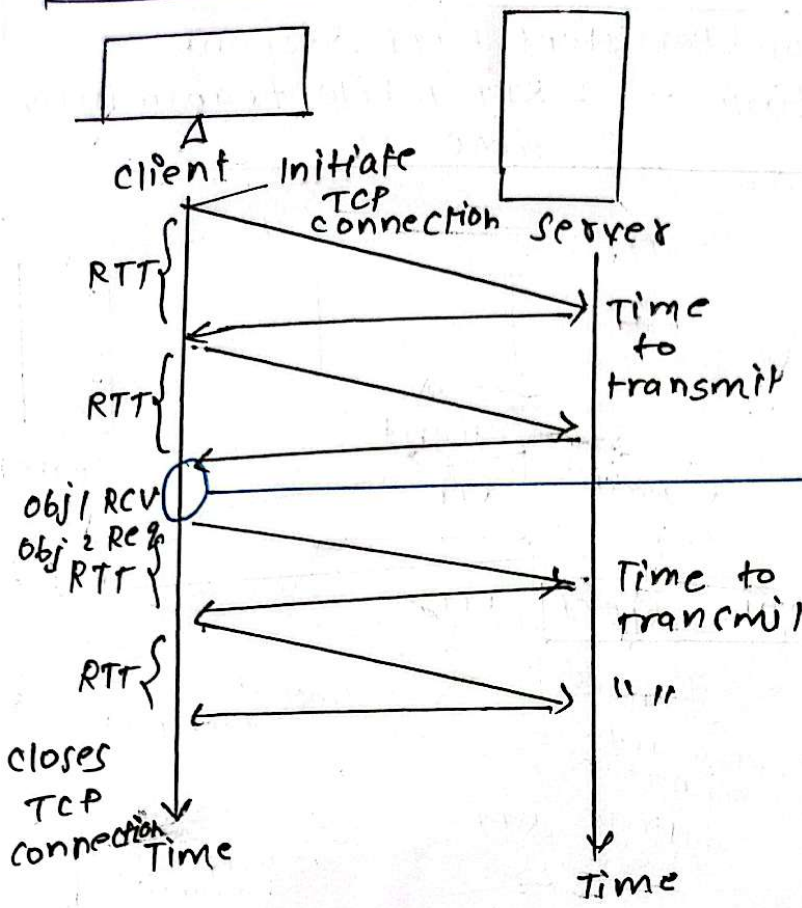
HTTP 1.1 made peristent HTTP connection to Default mode

- The server now keeps the TCP connection open for a certain period of time, so that it can make multiple request at same TCP.

Types of persistent connection

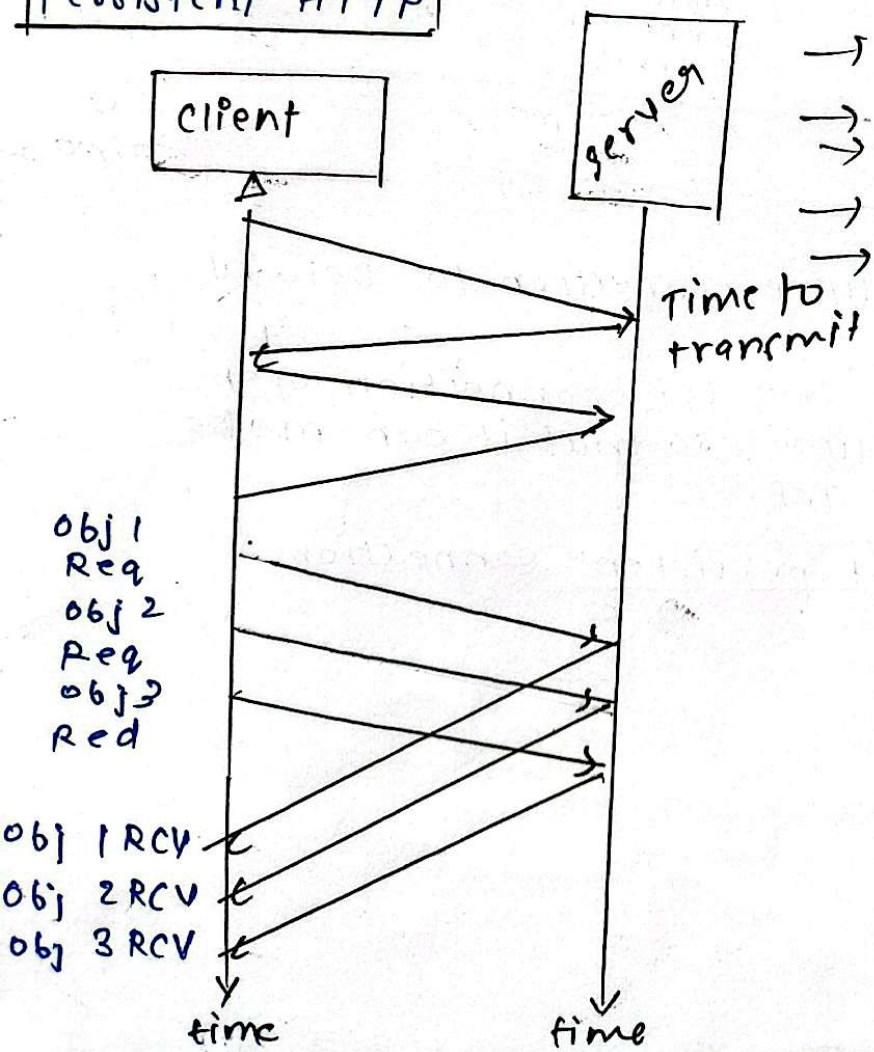
- without pipeling
- with pipeling

without Pipelining



This Pale time situation waste server Resources

Persistent HTTP

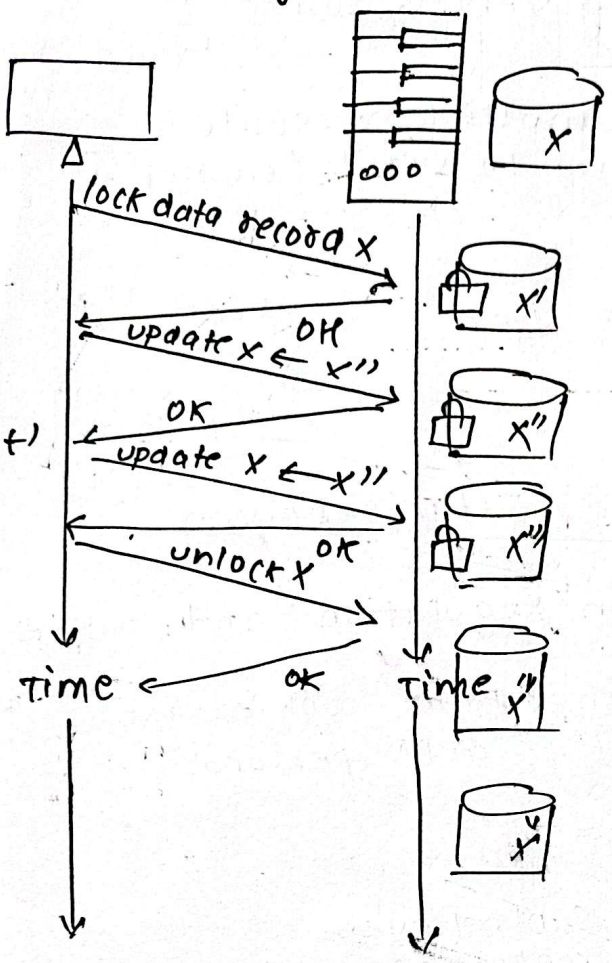


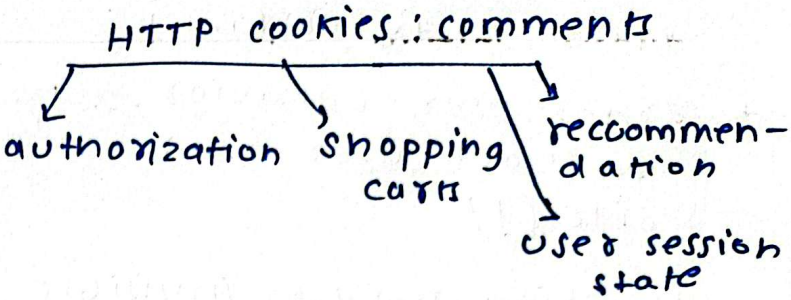
HTTP Response status code

→ 200 OK
→ 301 MOVED PERMANENTLY
→ 400 BAD REQUEST
→ 404 NOT FOUND
→ 505 HTTP VERSION NOT SUPPORTED

Maintaining user/server state cookies

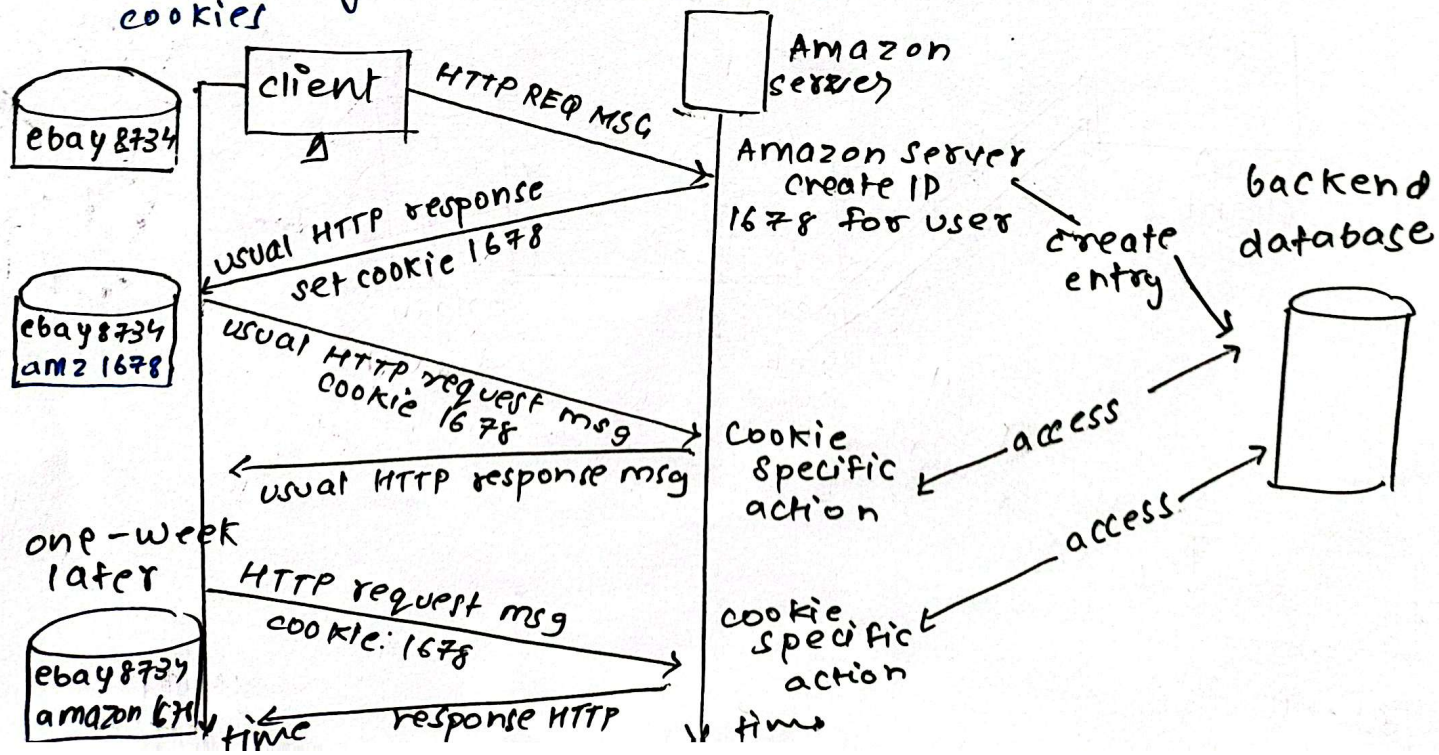
A stateful protocol: client makes two exchanges to x or None at all





Website and client browser use cookies to maintain some state between transactions.

Maintaining user/server state cookies



cookies: Tracking a user's Browsing behaviour

cookies can be used to

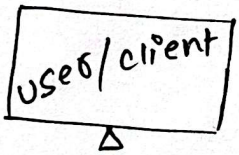
* track user behavior on a given website (First party cookies)
* track user behavior across multiple website without user ever choosing to visit tracker site (!)
* tracking may be invisible to user

third party tracking cookies:

Disabled by default in Firefox, safari browser

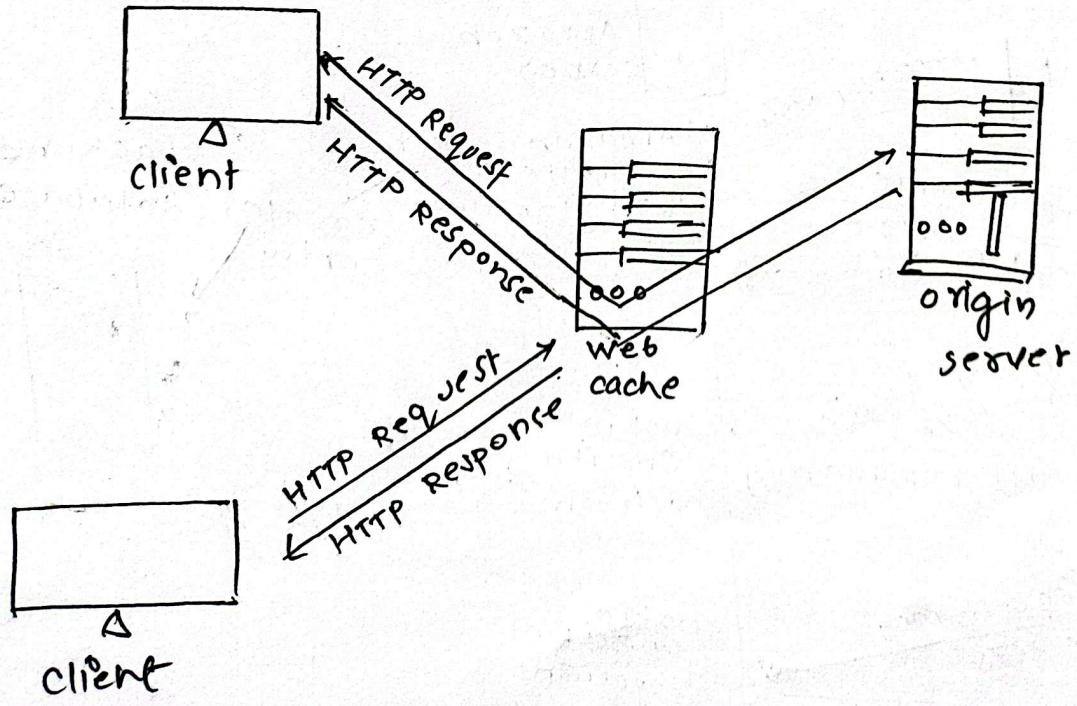
GDPR (Eu General Data Protection Regulation) and cookies

When cookies can identity an individual, cookies are consider personal data, subject to GDPR personal data regulation.

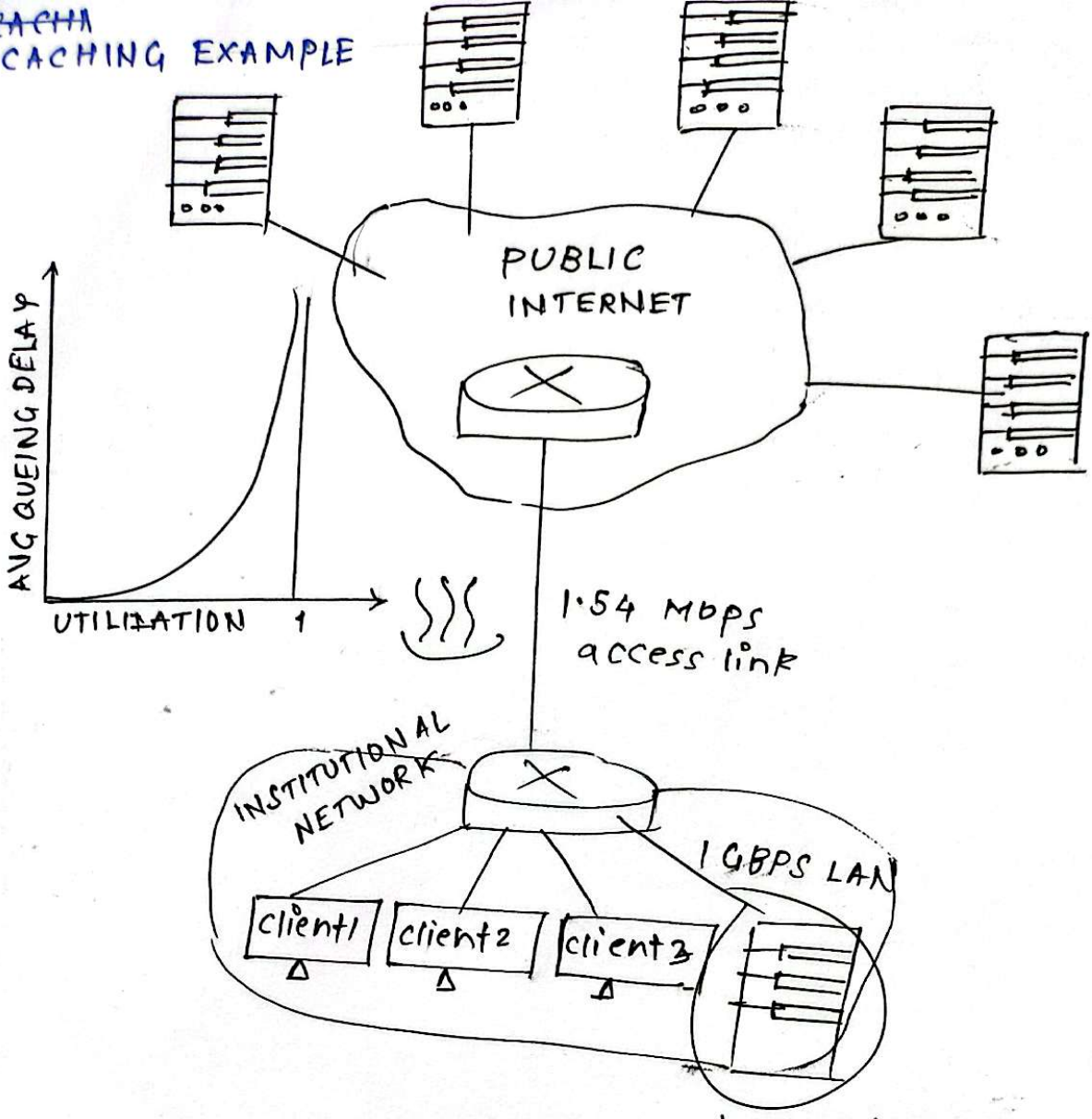


user has explicit control over whether or not cookies are allowed.

Web caches



~~CACHA~~
CACHING EXAMPLE



Calculating access link utilization

End-to-end delay

cache hit rate is 0.4

40% requested served by cache

60% requested satisfied origin

Rate to browsers over access link

$= 0.6 \times 1.50 \text{ Mbps} = 0.58 \text{ secs}$

Avg End-to-End Delay = 0.6 (delay from origin servers) + 0.4 (delay satisfied at cache)

$= 0.6 (2.01) + 0.4 \Rightarrow 1.2 \text{ secs}$

Example app : UDP client

SOCKET PROGRAMMING

Python : UDP client

```
from socket import *
serverName = 'hostname'
serverPort = 12000
clientSocket = socket (AF_INET, SOCK_DGRAM)
message = input ("Input lowercase sentence :')
clientSocket.sendto (message.encode(), (serverNam,
  serverPort))
ModifiedMessage, serverAddress = clientsocket.rcv
from (2048)
print (Modified Message.decode())
  clientSocket.close()
```

phython UDP client

```
    socket
from  import *
serverName = 'hostname'
server port = 12000
clientSocket = socket (AF_INET, SOCK_DGRAM)
message = Input ("Input lowercase message")
clientSocket.sendto (message.encode(), (serverName,
serverPort)
modifiedmessage, serveraddress = clientsocket.rcv (20M/2048).
  print (madifiedmessage.decode())
      clientSocket.close()
```

python UDP client

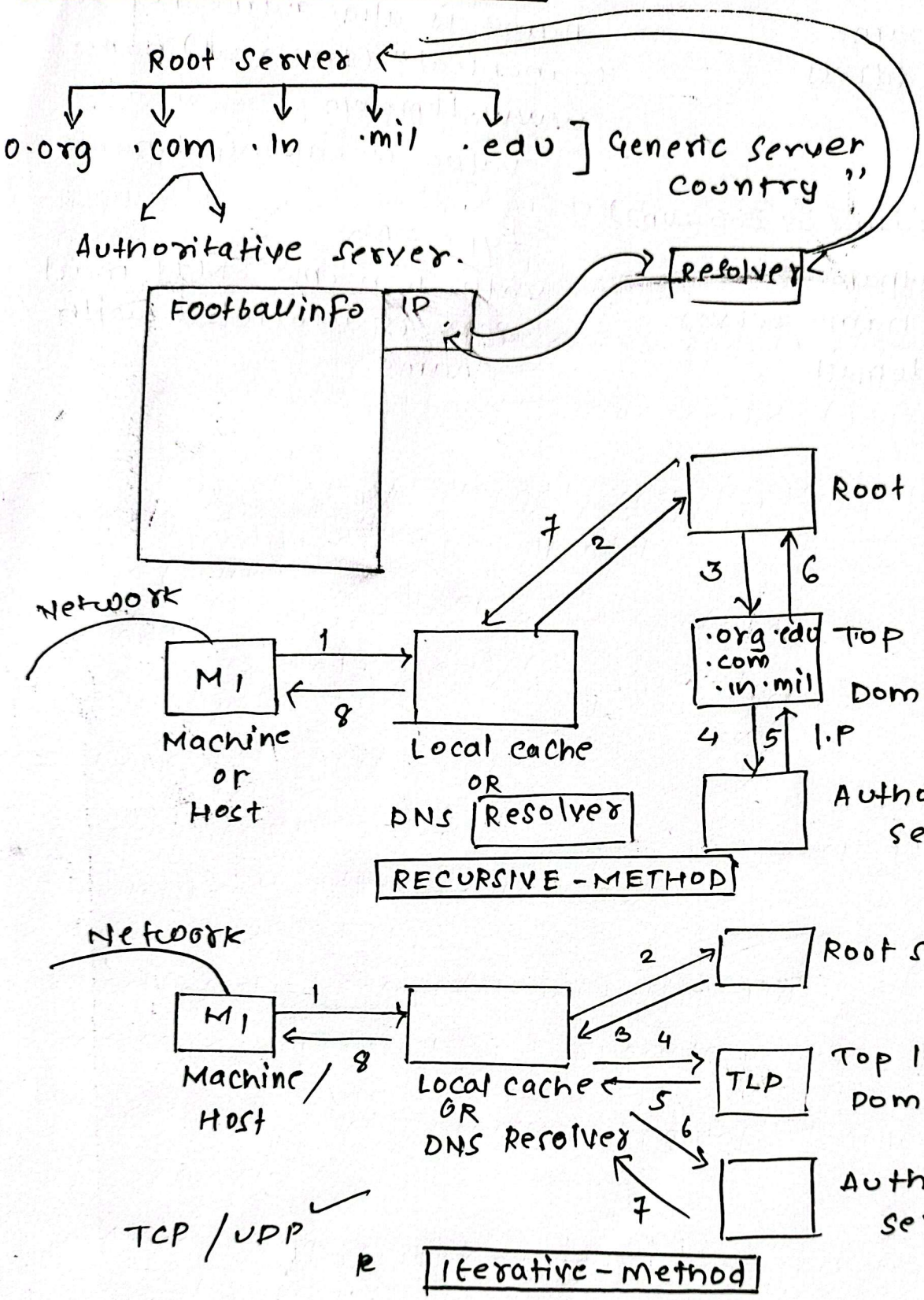
```
from socket import *
serverName = 'hostname'
serverport = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
message = input('input lowercase sentence?)
clientSocket.sendto(message.encode(), (serverName,
    serverPort)

modifiedmessage, serveraddress= clientSocket.recvfrom(2048)
print(modifiedMessage.decode())
clientsocket.close()
```

python UDPserver

```
from socket from *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', serverPort))
print('The server is ready to receive')
while
```

Domain Name Servers (DNS)



DNS Records : Distributed database storing resource Records (RR)

RR Format : (name, value, type, ttl)

type = A

name is Hostname

value is IP Address

type = B

name is domain (eg foo.com)

value is hostname of authoritative name server for this domain

type = CNAME

name is alias name for some "canonical" (the real) name

www.ibm.com

value is canonical name

type = MX

value is name SMTP mail server associated with name

UDP checksum

Goal: detect errors (i.e flipped bits) in transmitted segments

	1st No.	2nd No.	sum
Transmitted	5	6	11
Received	4	6	11

eg: Adding 16 Bit Numbers

Receiver-computed checksum ≠ sender-computed checksum (as received)

Summary: UDP

"No frills" protocol

- segments may be lost, delivered out of order
- Best effort service "sends and Hopes for the Best"

UDP has its plusses

- No setup/handshaking needed (no RTT incurred)
- can function when network service is compromised
- Helps with reliability (checksum)

Build additional functionality on top of UDP in application layer (eg. HTTP/3)

UDP: User-Datagram Protocol

UDP USE

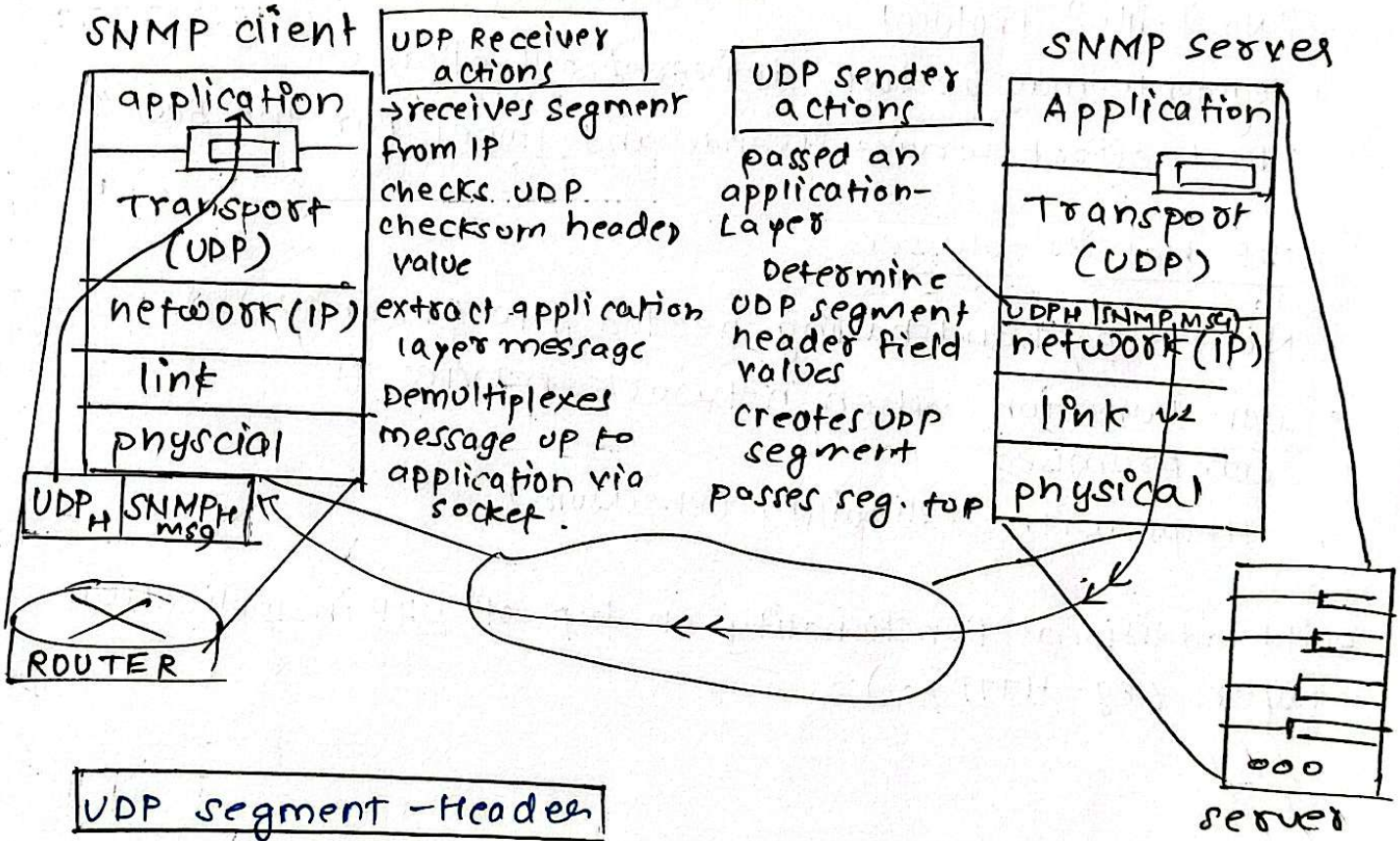
streaming multimedia apps (loss tolerant, rate sensetive)

- DNS
- SNMP
- HTTP/3

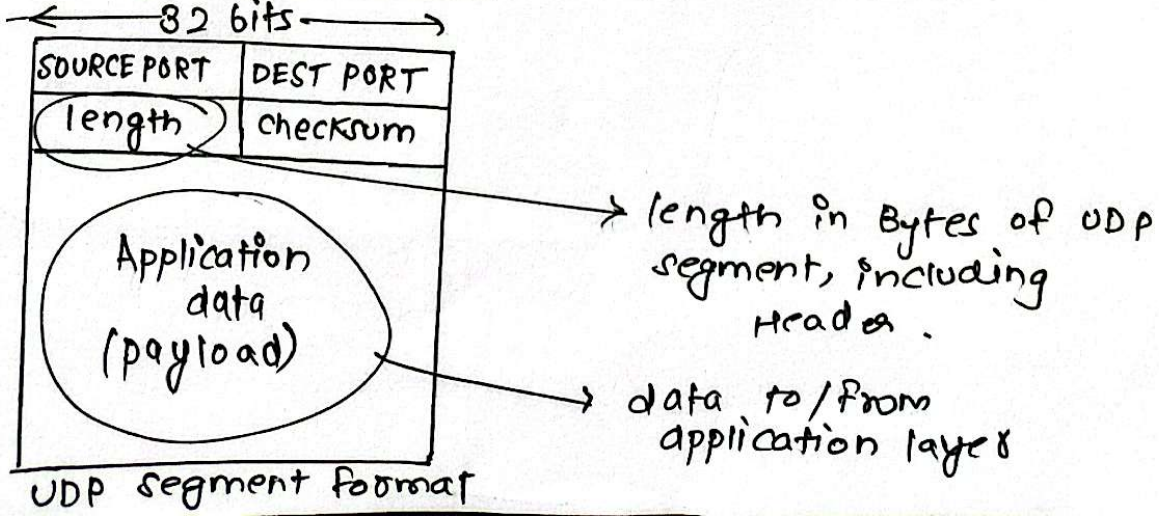
If reliable transfer needed over UDP (eg. HTTP/3)

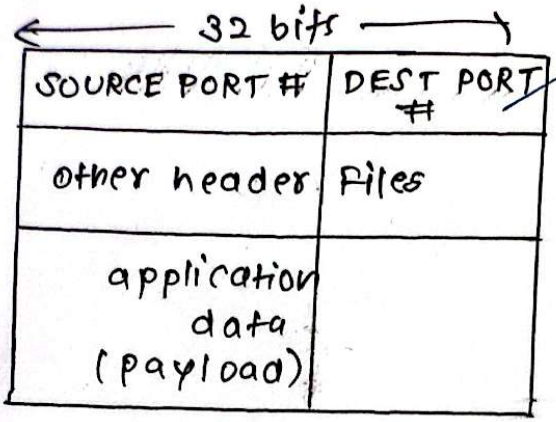
- add needed reliability at application layer
- Add congestion control at application layer.

UDP: Transport Layer Actions



UDP segment - Header





TCP/UDP segment format

IP ADDRESS PORTS

HOST USES IP addressing & port numbers to direct segment to appropriate socket.

create socket, must specify ~~local p~~ host-local port #

DatagramSocket mySocket1 = new DatagramSocket(1234)

connectionless demultiplexing

mySocket = socket (AF_INET, SOCK_STREAM)
mySocket.bind (myaddr, ~~417~~ 9157);

SRC:9157 | DEST:6428

mySocket = socket (AF_INET, SOCK_DGRAM)
mySocket.bind (myaddr, 6428);

SRC:6428 | DST:9157

mySocket = socket (AF_INET, SOCK_STREAM)
mySocket.bind (myaddr, 5775);

? | DST SRC |

SRC | DST

Summary

Multiplexing, demultiplexing: based on segment, datagrams header field values.

UDP: demultiplexing using destination port number (only)

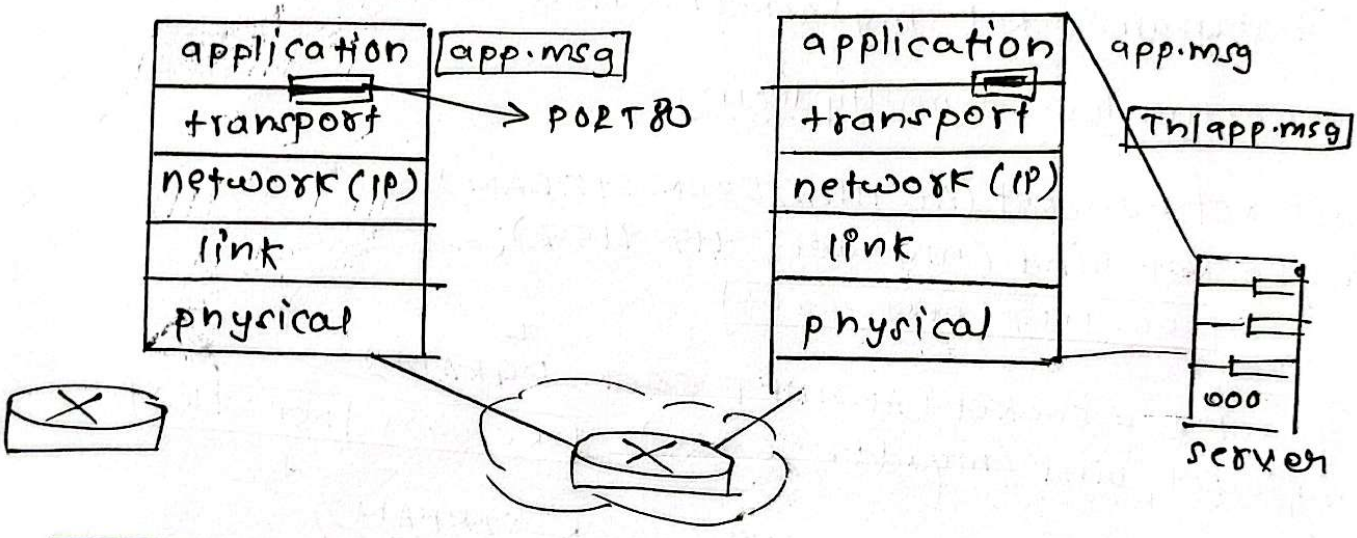
TCP: Demultiplexing using 4-tuple: source and destination IP address, & port numbers

Multiplexing/demultiplexing happens at all layers

TRANSPORT LAYER

TCP: Transmission Control Protocol
- Reliable, in-order delivery
- congestion control
- flow control
- connection setup

UDP: User Datagram Protocol
- unreliable, unordered delivery
- no frills extension of "best-effort" IP



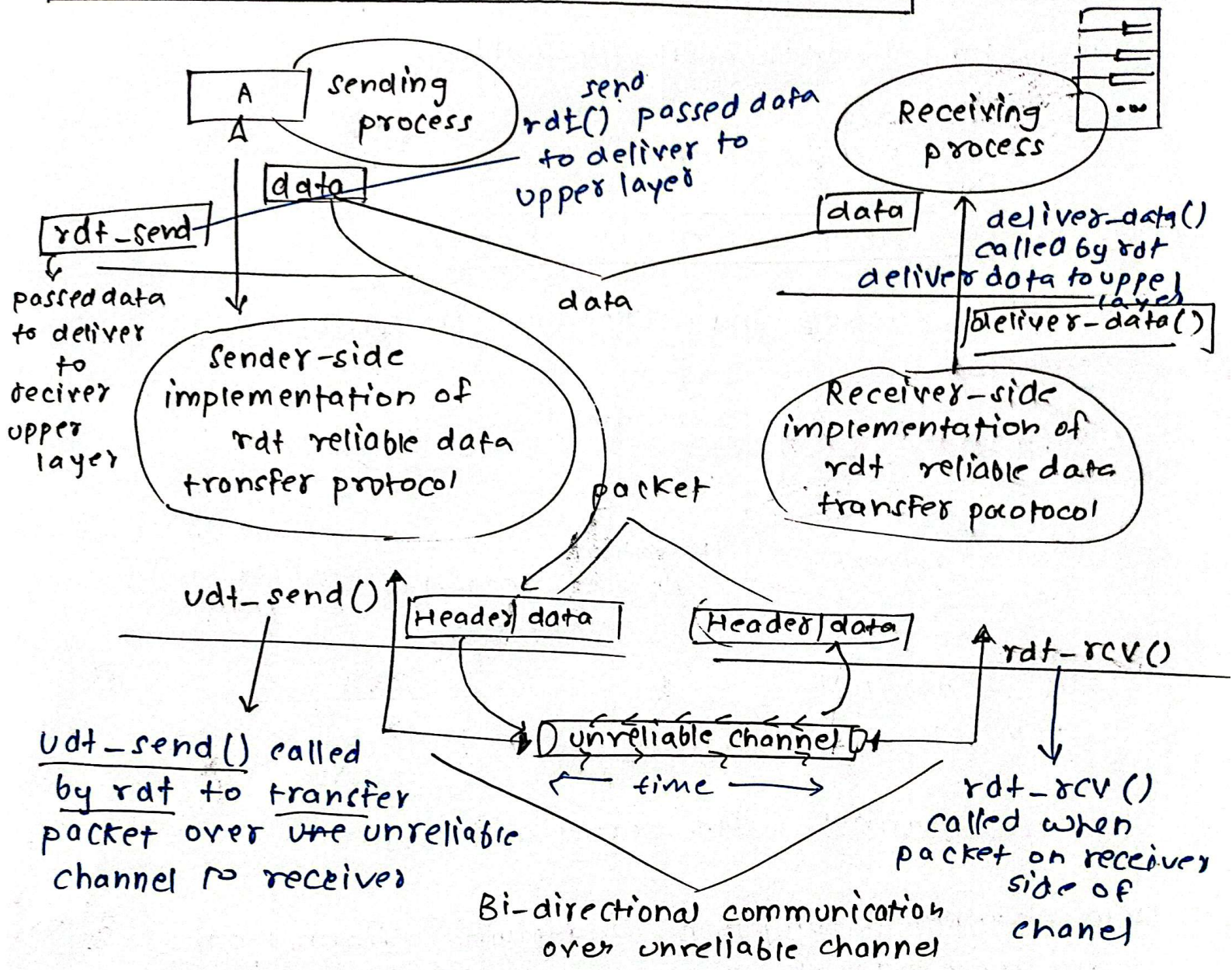
Sender

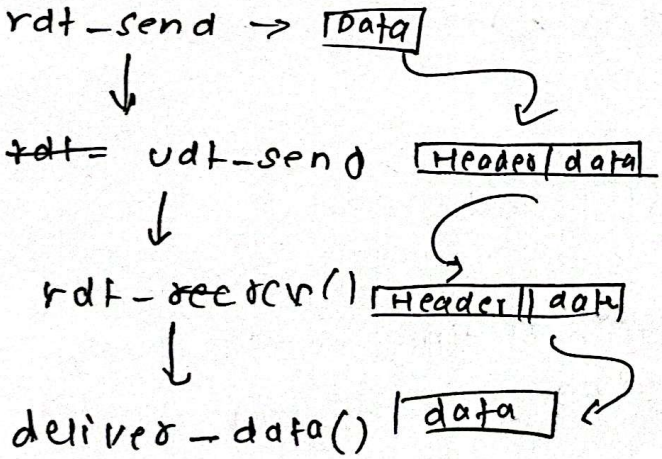
is passed an application layer message
determines segment header value
create segment
passes segment to IP

Reciever

Recieves segment from IP
check header values
extract application-layer
demultiplexes message up to application using socket.

Reliable data transfer protocol (rdt) interfaces





rdt 1.0 : Reliable transfer over a reliable channel

underlying channel perfectly reliable
- → No bit errors
- → No loss of packets

rdt 2.0 channel with bit errors

underlying packets may flip bits in packet
- → checks ↓ checksum

Other Errors
- ↙ Acknowledgement (ACKs) ↓ Tells sender pkt RCV OK
- ↘ Negative Acknowledged (NAKs)

rdt 3.0 channel with errors and loss

New channel Assumption : underlying channel can also loose packets (data, ACKs)

performance of rdt 3.0 Stop and wait

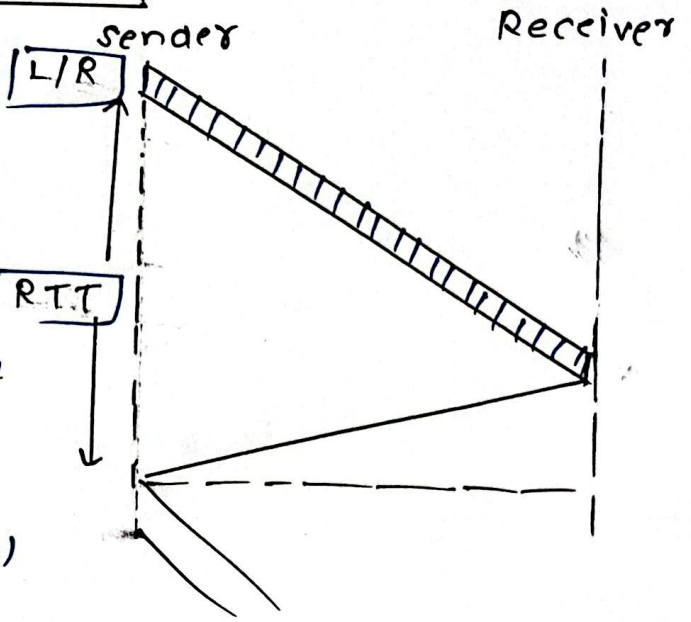
Usender : utilization – fraction of time sender busy sending.

1 GBPS link
15 ms prop. delay
8000 bit packet

$D_{trans} = \frac{L}{R} = \frac{8000 \text{ bits}}{10^9 \text{ bits/s}} = 8 \text{ ms}$ in channel

10^9 bits/s

rdt 3.0 stop and wait operation



$$U_{sender} = \frac{L/R}{RTT + L/R}$$

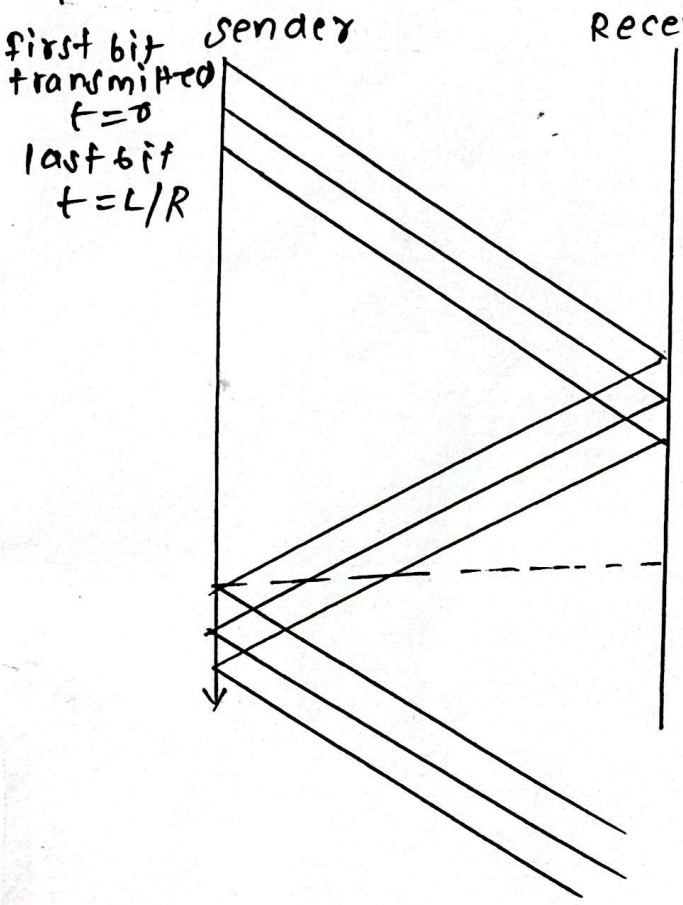
$$= \frac{0.08}{30.008} = 0.00027$$

rdt 3.0 protocol

↙

limits performance of underlying infrastructure. (channel)

pipelining : increased utilization



$$U_{sender} = \frac{3L/R}{RTT + L/R} = \frac{0.0024}{30.008}$$

$$= 0.00081$$

↑

3-bit 3 packet pipeling increases utilization by a factor of 3)

TCP : overview RFC 793, 1112, 2018, 5681, 7323

point-to-point: one sender, one receiver

Reliable, inorder byte stream no "message boundaries"

full duplex data: bi-directional data flow in same connection

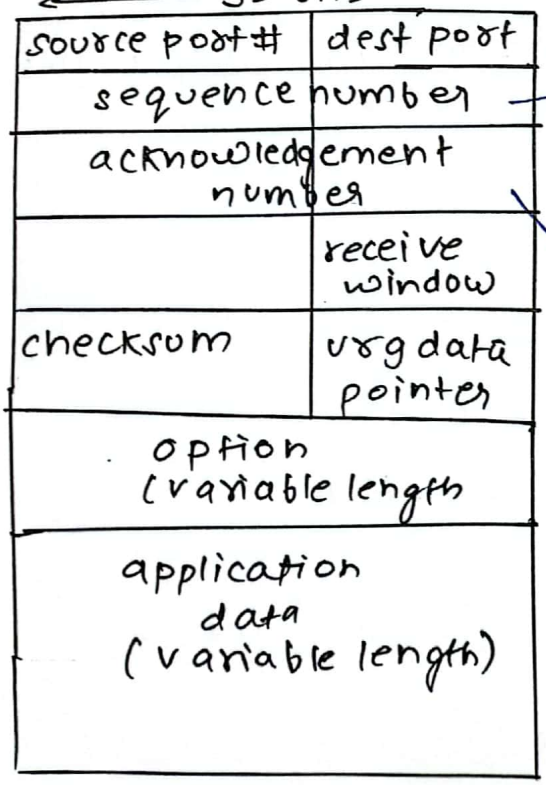
MMS: maximum segment size

pipelining : TCP congestion and flow control set window size

connection-oriented : handshaking (exchange of control message) initializes sender, receiver state before data exchange

flow control : sender will not overwhelm receiver

TCP segment structure



sequence-number
byte stream "number" of first byte in segment's data

Acknowledgement
seq # of next byte expected from other side

TCP Round Trip, Timeout

$$\text{EstimatedRTT} = (1-\alpha)*\text{EstimatedRTT} + \alpha*\text{sample RTT}$$

timeout interval : EstimatedRTT plus "safety margin"

large variation EstimatedRTT: want a larger safety margin.

$$\text{Timeout Interval} = \text{Estimated RTT} + 4 * \text{DevRTT}$$

Estimated RTT ; "safety margin"

$$\text{DevRTT} = (1-\beta)*\text{DevRTT} + \beta*|\text{sample RTT} - \text{Estimated RTT}|$$

typically $\beta = 0.25$

principles of congestion control

congestion: "too many sources sending too much data too fast for network.

manifestation
- long delay
- packet loss

causes/cost of congestion: scenario 1

simplest scenario:

one router, infinite buffer
input, output link capacity: R
two flows
no retransmission needed

causes/cost of congestion: scenario 2

one router, finite buffers

sender retransmits lost, timed-out-packet

application-layer = application-layer
input output

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

Transport-layer
input included retransmission

$$\lambda'_{in} \geq \lambda_{in}$$

causes/cost of congestion: scenario 3

four senders
Multi-hop path
timeout/retransmit

Network Layer in computer Networks

Network-Layer functions:

forwarding: move packets from a router's input link to appropriate router output link.

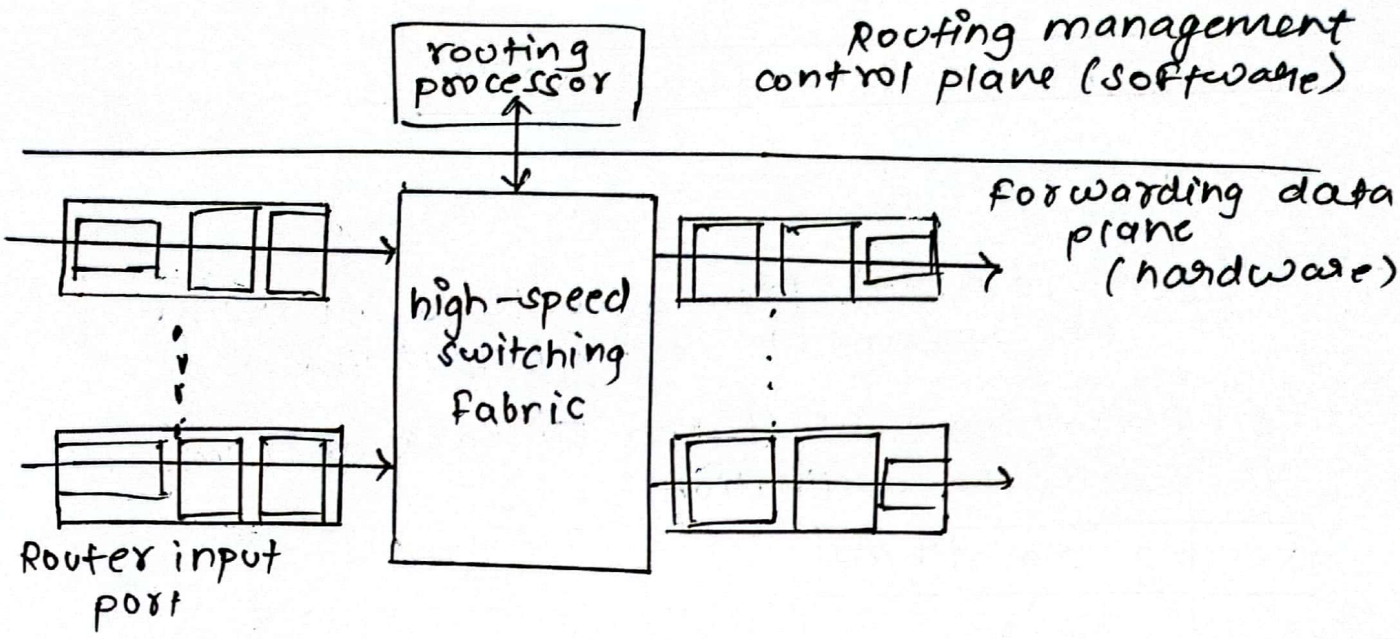
Routing: determine Route taken by packets from source to destination

- Routing algorithms
  - per-Router control plane
  - software Defined Networking (SDN)

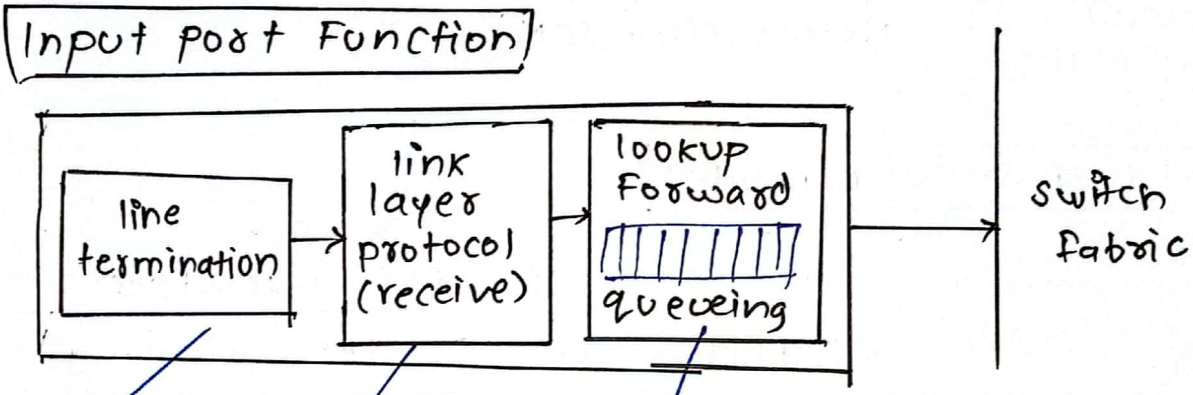
Network-Layer service model

Network Architecture	service Model	Quality of Service (QoS) Guarantees?			
		Bandwidth	Loss	order	Timing
Internet	best effort	none	no	no	no
ATM	constant Bit Rate	constant rate	yes	yes	yes
ATM	Available Bit Rate	Guaranteed min	no	yes	no
Internet	Intserv Guaranteed (RFC 1633)	yes	yes	yes	yes
Internet	Diffserv (RFC 2475)	possible	possibly	possibly	no

Router Architecture overview



Input port Function



decentralized switching

→ using header field values, lookup output port using forwarding table in input memory

goal: complete input port processing at 'line speed'

input port queing: if datagrams arrive faster than forwarding rate into switch fabric

Switching fabrics

Transfer packet from input link to appropriate output link

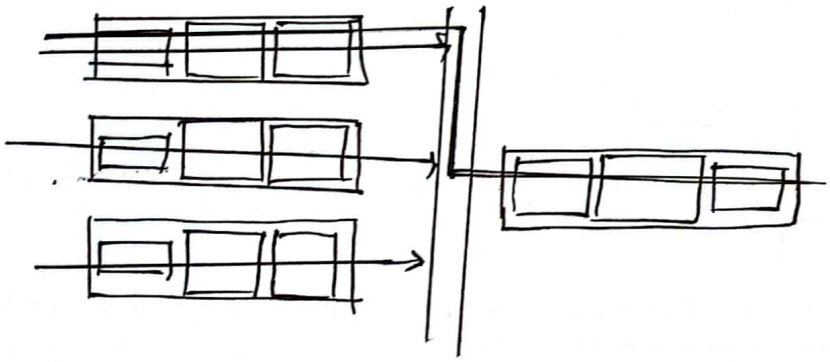
switching outputs :

- often measured as multiple of input/output rate
- N input: switching rate N times line rate desirable

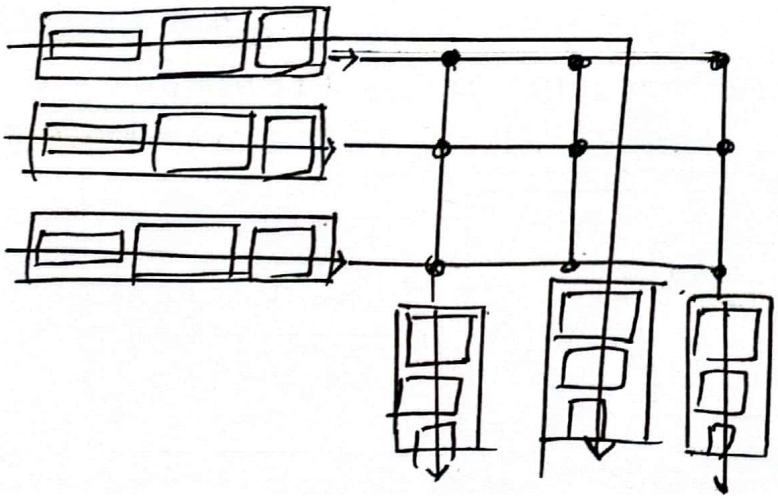
Switching via a Bus

datagram from input port memory to output port memory via a BUS.

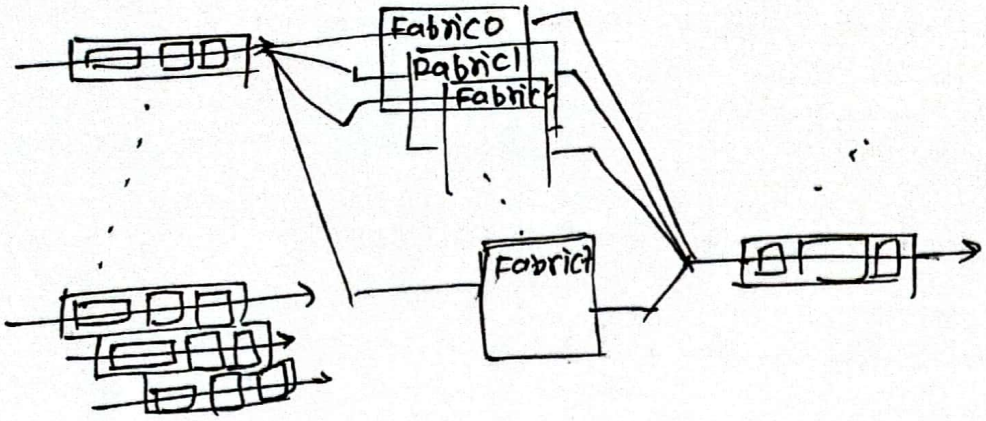
BUS connection : switching speed limited by BUS Bandwidth
32 GBPS



switching via interconnection network



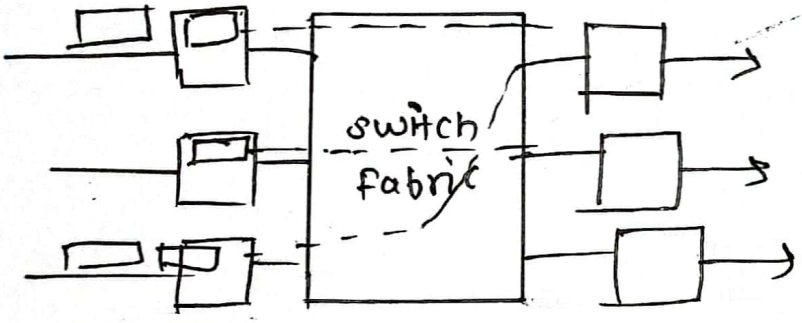
Switching via interconnection network



Input port queing

If switch fabric slower than ports combined → queing may occur at input queues

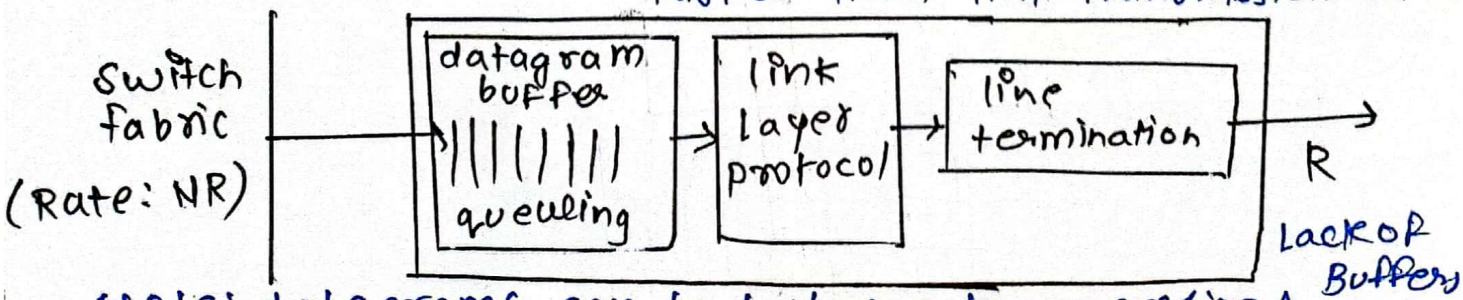
- queueing delay and loss due to input buffer overflow



Head-of-line (HOL) Blocking: queued datagram at front of queue prevents others in queue from blocking.

Output port queing

Buffering required when datagrams arive from fabric faster than link transmission rate.



Note: datagrams can be lost due to congestion

FCFS: packets transmitted in order of arrival to output port

First-in-First out

priority policies

priority scheduling: send packet from highest priority queu that has buffered packets.

Round Robin (RR) scheduling: Arriving traffic classified, queued by class.

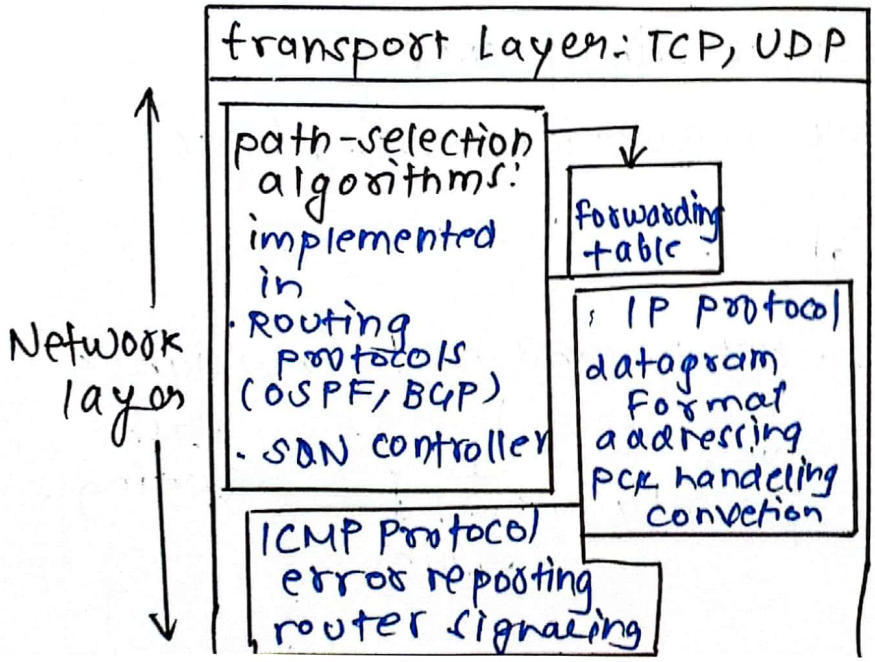
server cyclically, Repeadly scan class queues, sending one complete packet from each class

Weighted Fair queuing (WFQ)

generalized round Robin
each class, i has weight wi
and gets weighted amount of service in each cycle

$$\frac{w_i}{\sum_j w_j}$$

Network: Layer



Buffering Desirability

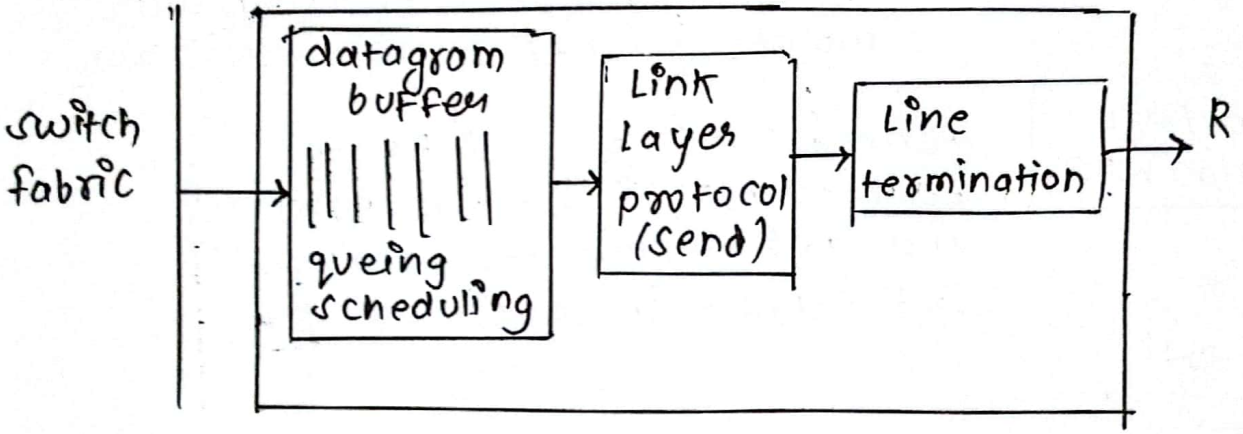
Recommendation: with N flows buffering equal to

$$\frac{RTT \cdot C}{\sqrt{N}}$$

Too much buffering can increase delays.

long RTT: poor performance for real-time apps, sluggish TCP response.

Buffer management



buffer management

drop which packet to add, drop when buffers are full.

tail drop: drop arriving packet

priority: drop/remove on priority basis

packet scheduling

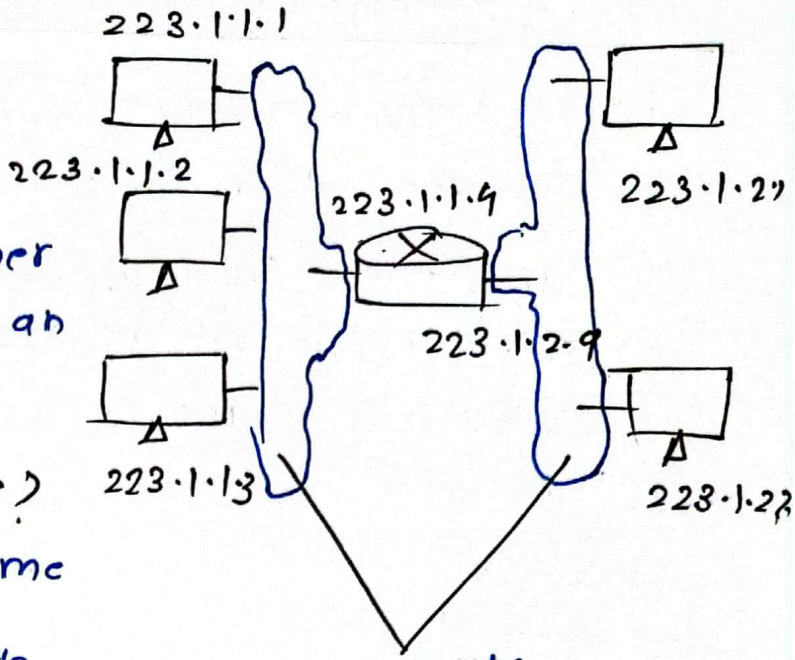
deciding which packet to send next → weighted fair queing.

- First come-first serve
- priority
- Round Robin

SUBNETS

What are subnets?

Device interface that can physically reach each other without passing through an intervening routers.



IP address have structure?

subnet part: devices in same subnet have common high order bits.

IP addressing : CIDR

CIDR: Classless Interdomain Routing

subnet portion of address of arbitary length

address format: a.b.c.d/x where x is # bits in subnet portion of address.

11001000 00010111 00010000 0000000
← subnet part → ← Host part →

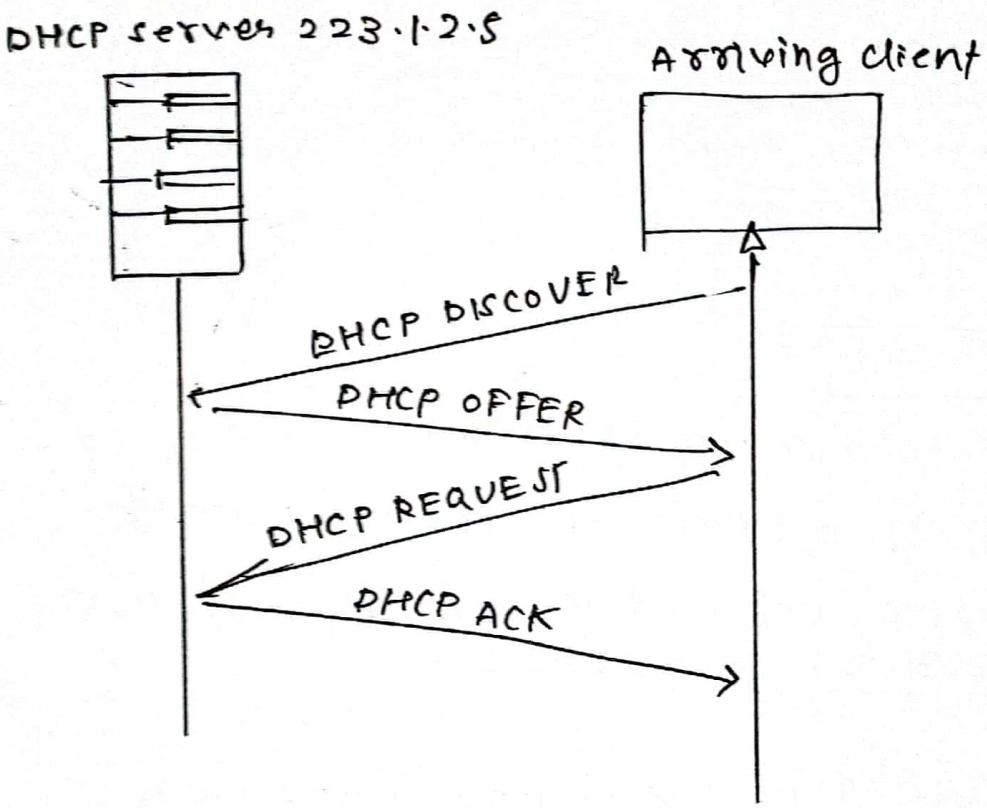
How Host get IP address?

- Hard coded by sysadmin in config file (example UNIX)
- DHCP: Dynammic Host configuration protocol: dynammically get address from a server

DHCP : Dynammic Host Configuration Protocol

goal: Host Dynamically obtains IP address from network server when it joins the network

1. can Renew its lease on address in use
2. Allow reuse of addresses
3. support For mobile users who joins/leave network



DHCP : more than IP address

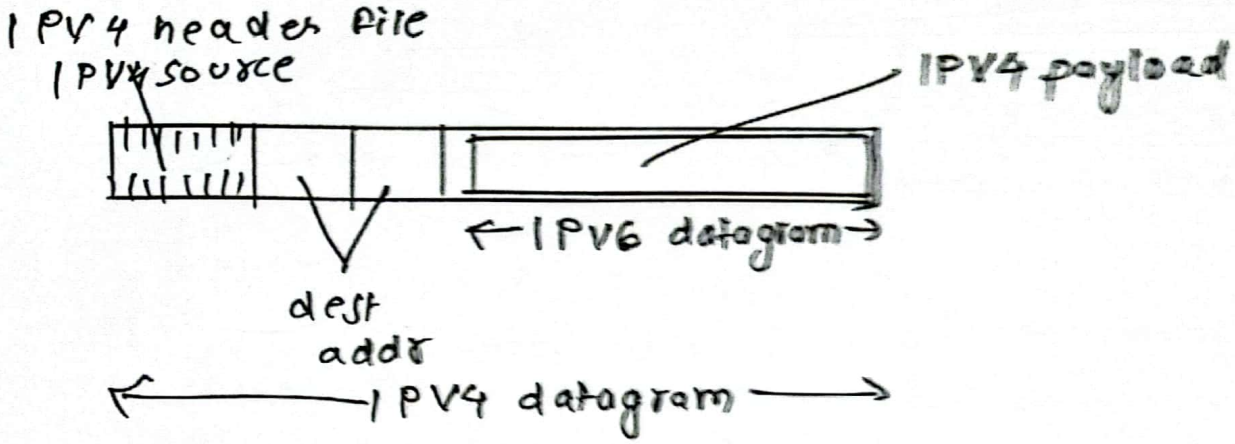
DHCP can return more than just allocated IP addres on subnet

1. Address of First-hop router for client
2. Name and IP address of DNS server
3. Network mask.

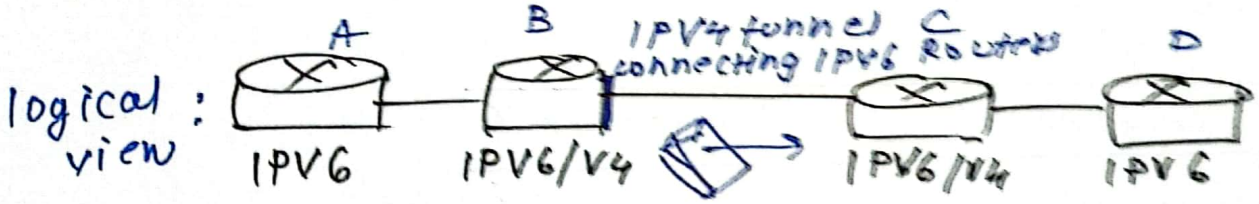
How Networks operate b/w mixed IPV4 and IPV6 Routers?

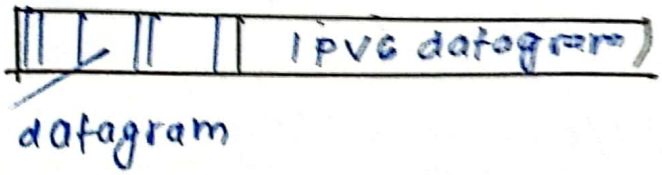
Tunneling: IPV6 datagram carried as payload in IPV4 datagram among IPV4

tunneling used extensively in other contents (4G/5G)



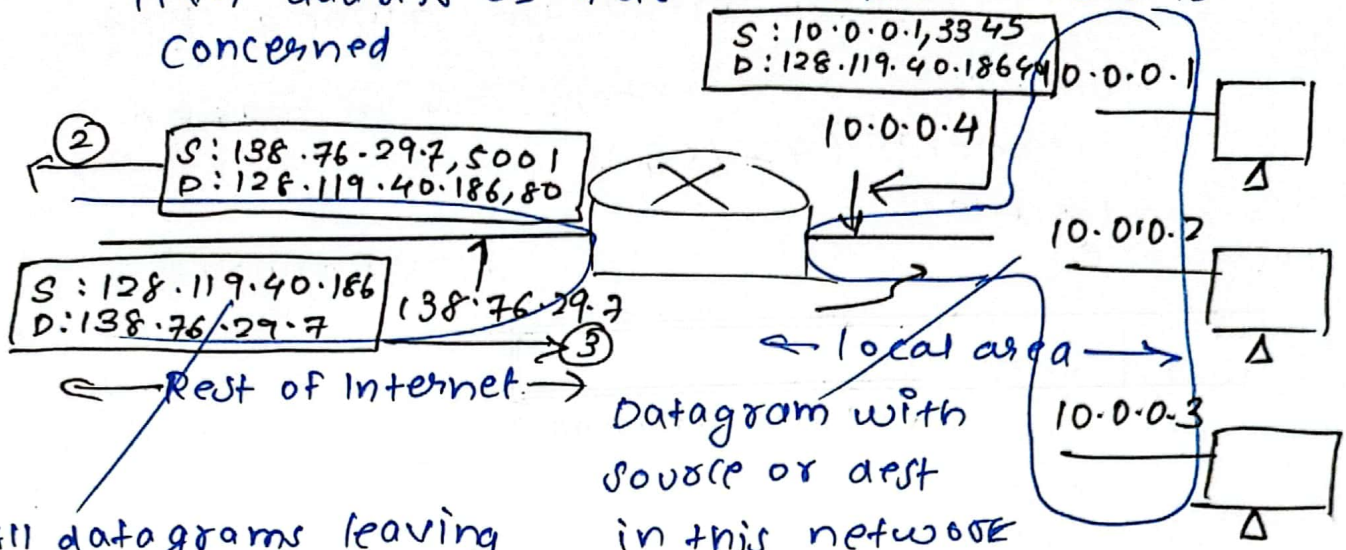
IPV4 tunnel connecting two IPV6 Routers





NAT: Network address translation

NAT: All devices in local network share just one IPV4 address as far as outside world is concerned



All datagrams leaving local area network have same source NAT IP address 138.76.29.7 But different source port Numbers.

IPV6 : Internet Protocol version 6

32-Bit IPV4 address space would be completley allocated

speed processing/forwarding
enable different network layer treatment of "flows"

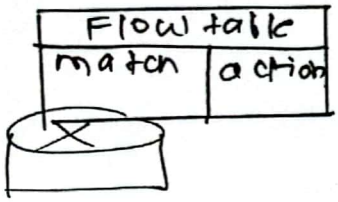
Transition from IPV4 to IPV6

Not all Routers can be upgraded simultaneously
no "flag delays"

Flow table abstraction

generalized Forwarding simple packet handing Rules

match : match pattern in packet header fields
action : matched packet : drop Forward, modify
priority : disambiguate overlating patterns
counter



Open Flow : abstraction

Routes :
match : longest destination IP prefix.
action : Forward out a link

switch
match : destination MAC address
action : Forward or flood

Firewall
match : IP/address and TCP-/UDP port numbers

NAT
match : IP address and port
action : Rewrite address and port

Middle Boxes

"Any ~~fterm~~ intermediatery box performing functions apart from normal, standard functions of an IP router on the data path between a source host and destination.

Network-Layer · Control plane

1) Host to Host delivery (source to Destination. Delivery

Two key network-layer functions

↙

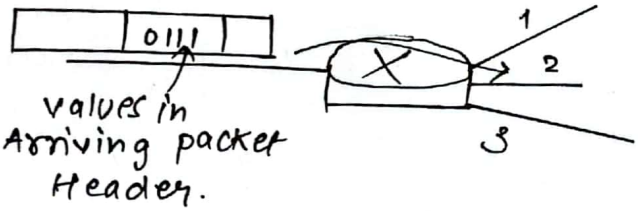
forwarding: Move packets from Router's input link to appropriate router output link.

Routing: Determine route taken by packets from source to destination

· Routing algorithm

Network Layer: Data Plane, control Plane.

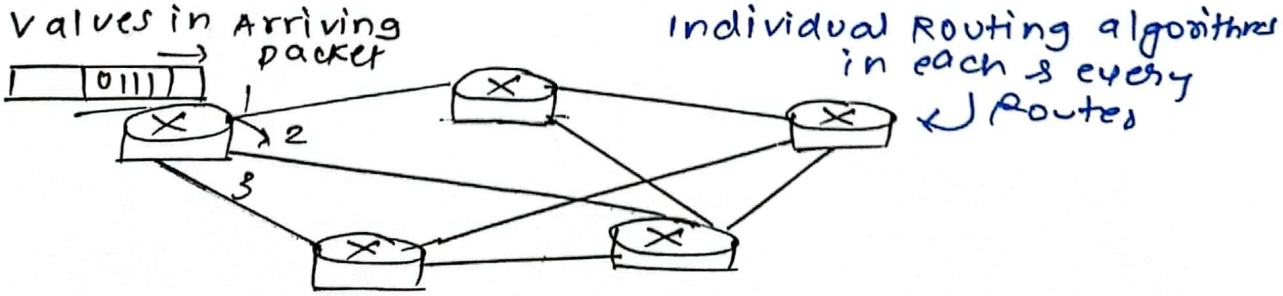
Data plane Determines how datagram arriving on router input port is forwarded to router o/p port



control plane - Determines how datagrams is routed among end-to-End From source host to destination host

↙ Traditional Routing Algorithms

↘ (Implemented in servers)
Softwared-defined Networking (SDN)



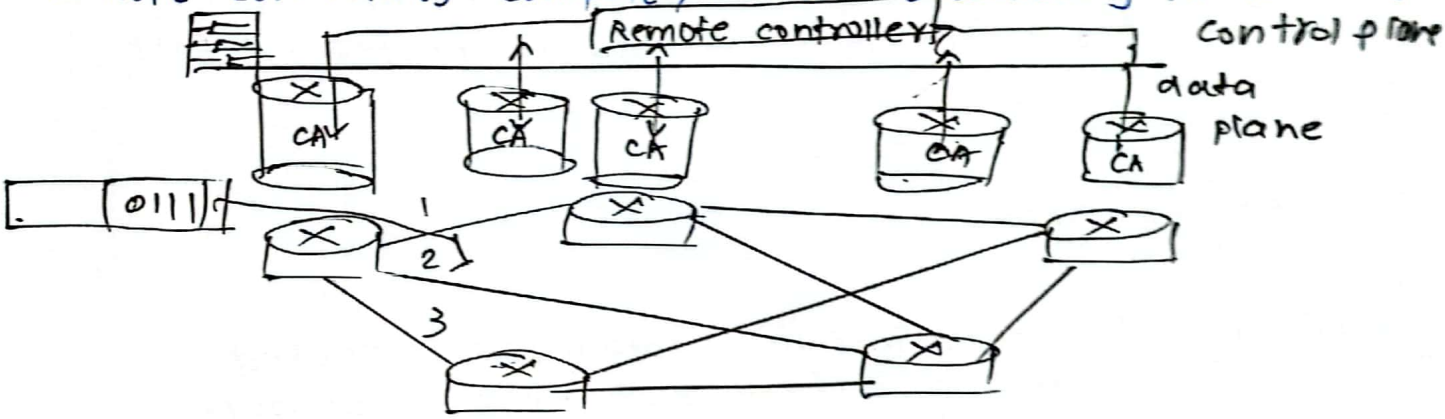
(SDN) Software-Defined Networking

Remote controllers compute, install forwarding table in Routers

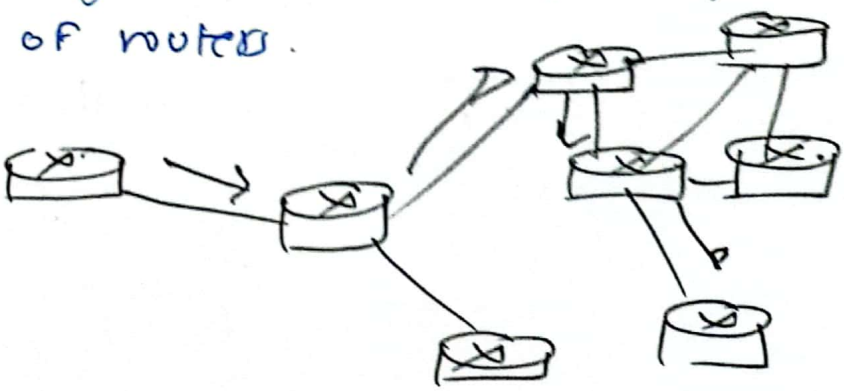
- Network-service-Model
  - Guaranteed delivery
  - Guaranteed delivery with bounded delay
  - Inorder packet delivery
  - Guaranteed minimal bandwidth
  - Guaranteed minimum jitter
  - Security service

Software-Defined Networking

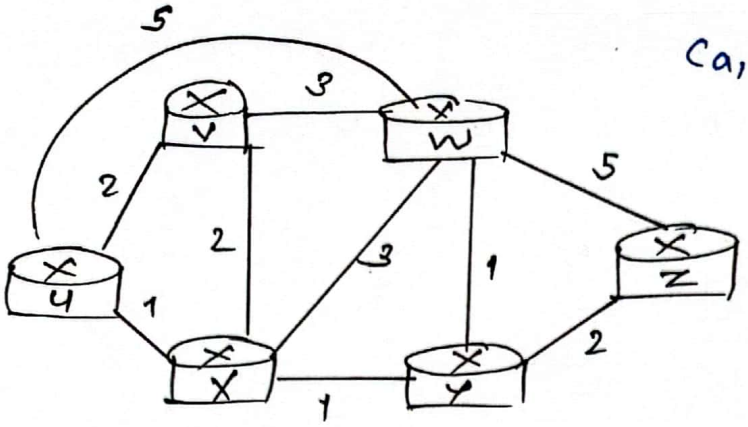
Remote controllers compute, install forwarding in routers



Routing protocol goal: determine "good" path from sending host to receiving host to receiving host, through network of routers.



Graph abstraction : link cost



$c_{a,b}$: cost of direct link connecting a and b

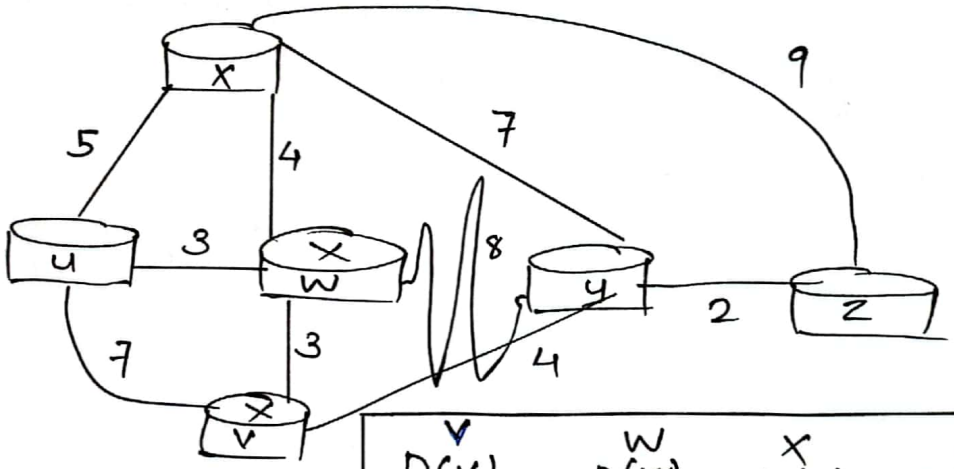
e.g : $c_{w,z} = 5$ $c_{u,z} = \infty$

graph : $G = (N, E)$

N : set of Routers = {u, v, w, x, y, z}

E : set of link = {(u,v), (u,x) (v,x), (v,w), (x,w), (x,y), (xw,y) (w,y), (w,z), (y,z)}

Dijkstra's Algorithm : Another example



Step	N'	v D(v) P(v)	w D(w) P(w)	x D(x) P(x)	y D(y) P(y)	z D(z) P(z)
	u	7, u	3, u	5, u	∞	∞
	uw	6, w		5, w	11, w	∞
	uwx	6, ~~w~~ x			11, ~~w~~ x	16, x
	uwxv				10, v	14, x
	uwxv					12, y
	uwxyz					

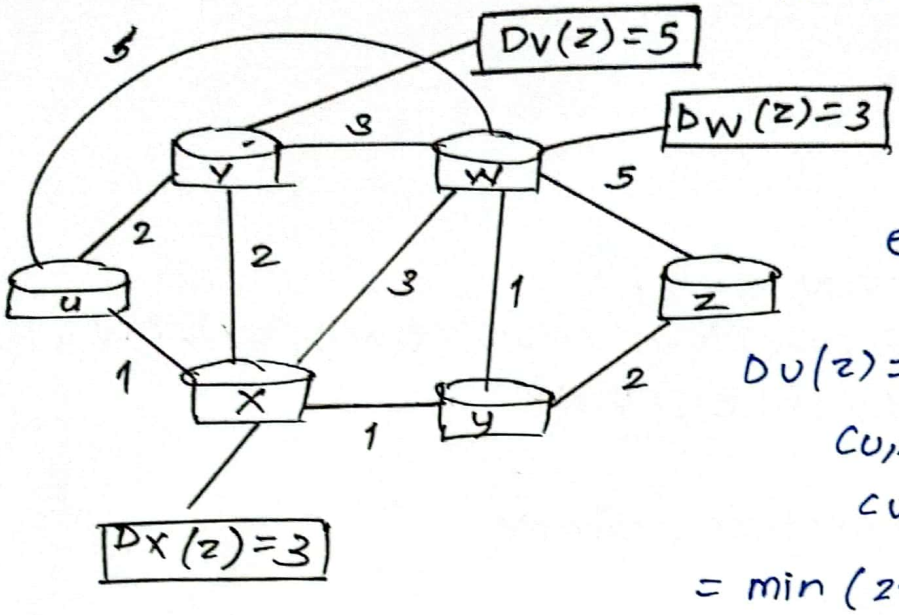
Distance vector algorithm

Bellman-Ford (BF) equation (dynamic programming)

$D_x(y)$: cost of lead-cost path from x to y

Then

$$D_x(y) = \min_v \{c_{x,v} + D_v(y)\}$$



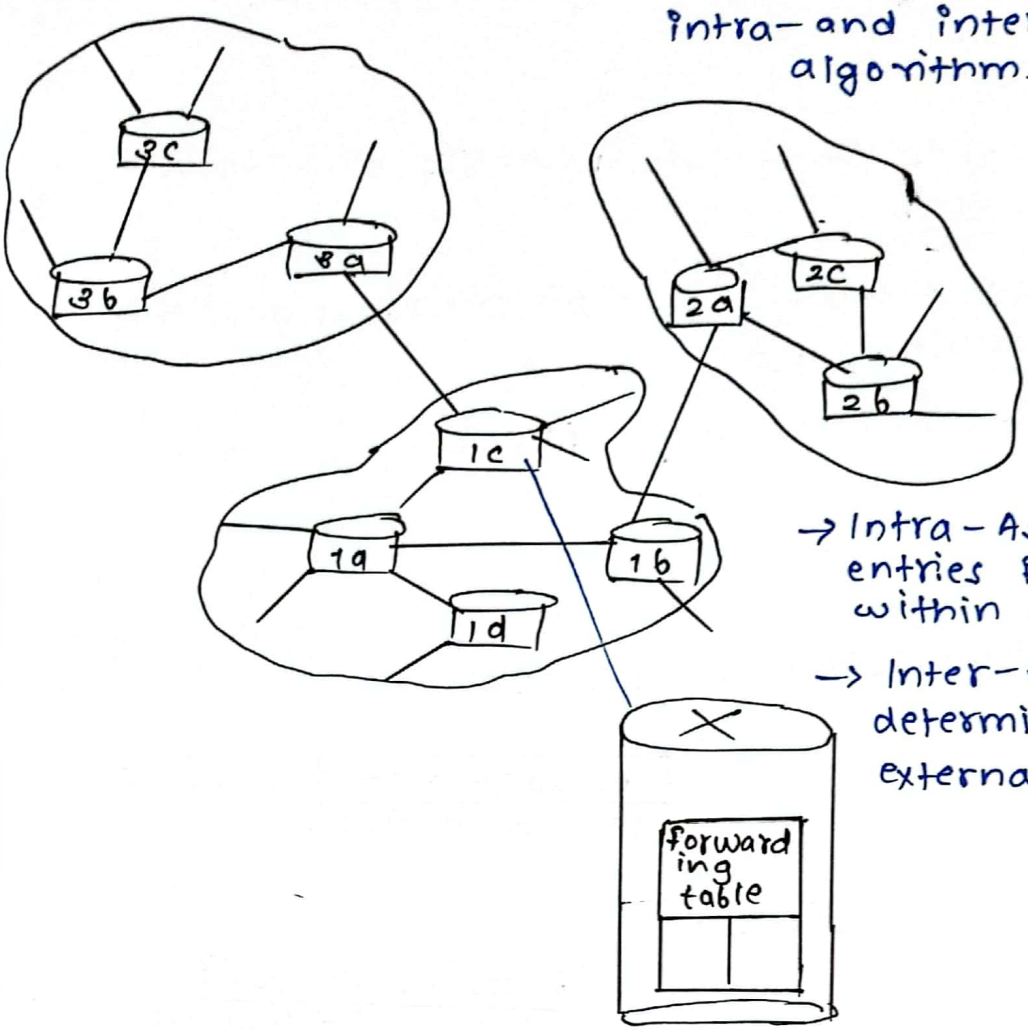
Bellan -ford equation says

$$D_u(z) = \min\{c_{u,w} + D_y(z), c_{u,x} + D_x(z), c_{u,w} + D_w(z)\}$$

$$= \min(2+5, 1+3, 5+3) = 4$$

Interconnected ASes

forwarding table configured by intra- and inter-AS routing algorithms



→ Intra-AS routing determine entries for destination within AS

→ Inter-AS & intra-AS determine entries for external destination.

Intra-AS Routing : Routing within an AS

most common Intra-AS routing protocols

* RIP : Routing Information Protocol [RFC 1723]
  - Follows classic Distance vectors every 30seconds
  - no longer used

* EIGRP : Enhanced Interior Gateway Routing Protocol
  - DV based
  - Formly cisco-proprietary for based for decades

* OSPF : Open Shortest Path File [RFC 2328]
  - link-state Routing
    IS-IS protocol essentially same as OSPF

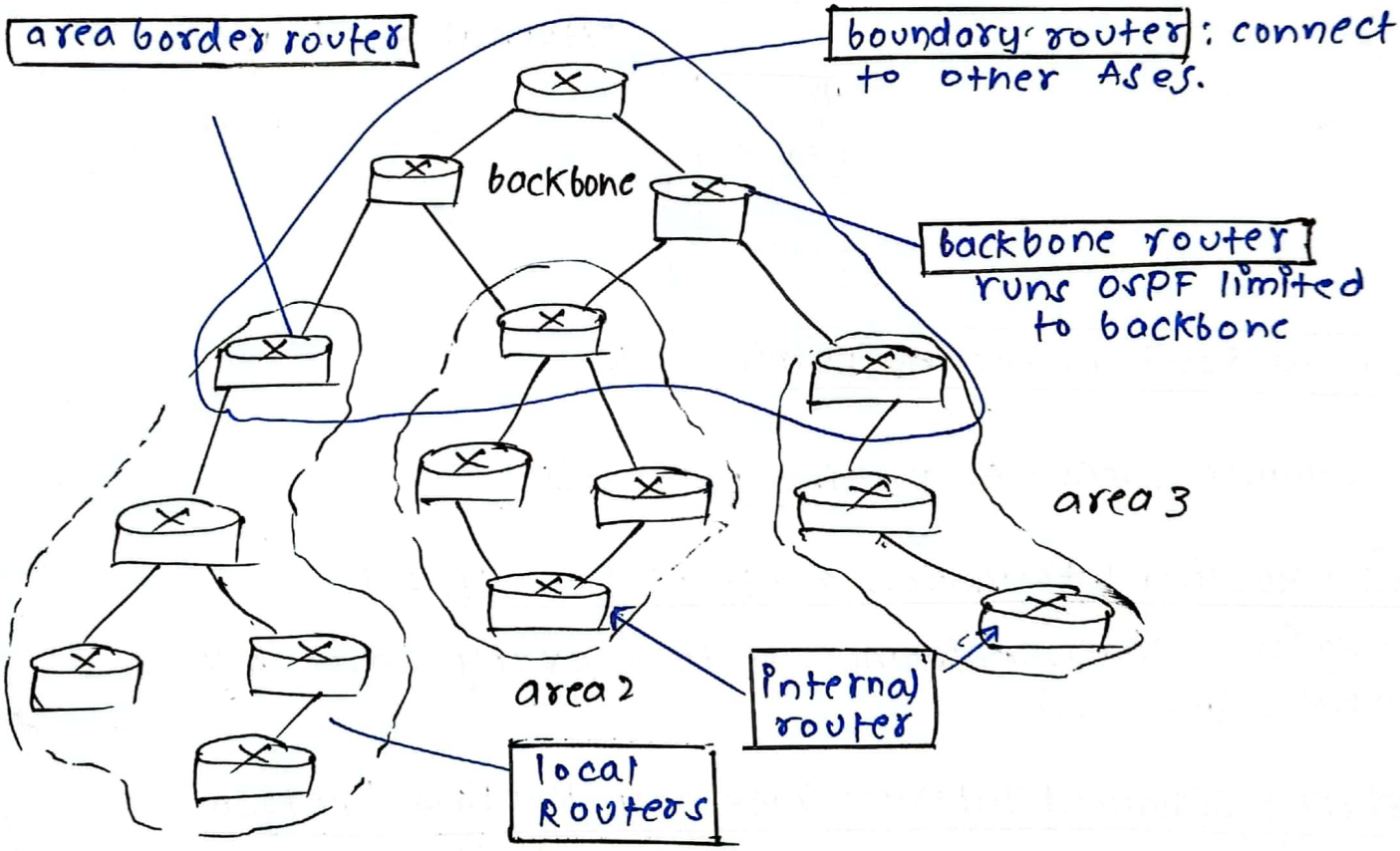
Classic link-state

- Each Router flood OSPF link-state advertisements to all router in entire AS
- Each router has full topology, uses Dijkstra's algorithm to compute forwarding table

security : all OSPF messages authenticated.

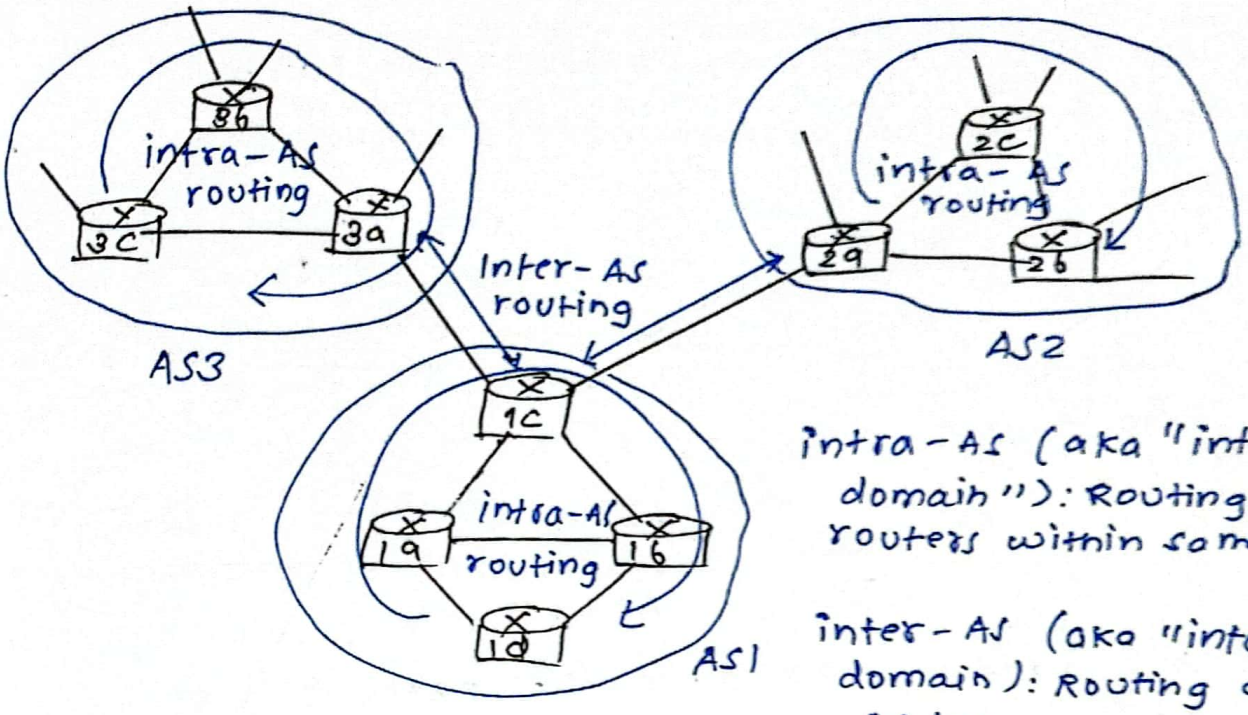
Hierarchial OSPF

Two-level-hierarchy : local area, backbone



- Flood LS in area only
- compute routing within area
- forward packet to outside via border router

Interconnected ASes



intra-AS (aka "intra-domain"): Routing among routers within same AS

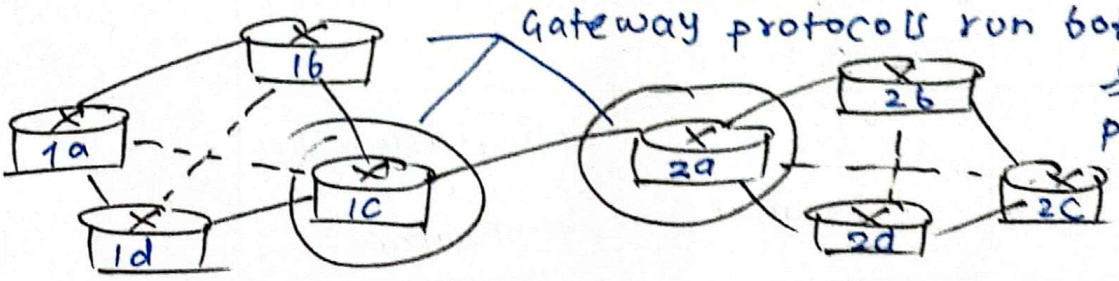
inter-AS (aka "inter-domain): Routing among AS 'es

BGP (Border Gateway Protocol): "glue that holds Internet together".

Allows subnet to advertise its existence and destination it can reach of internet.

BGP provides :-

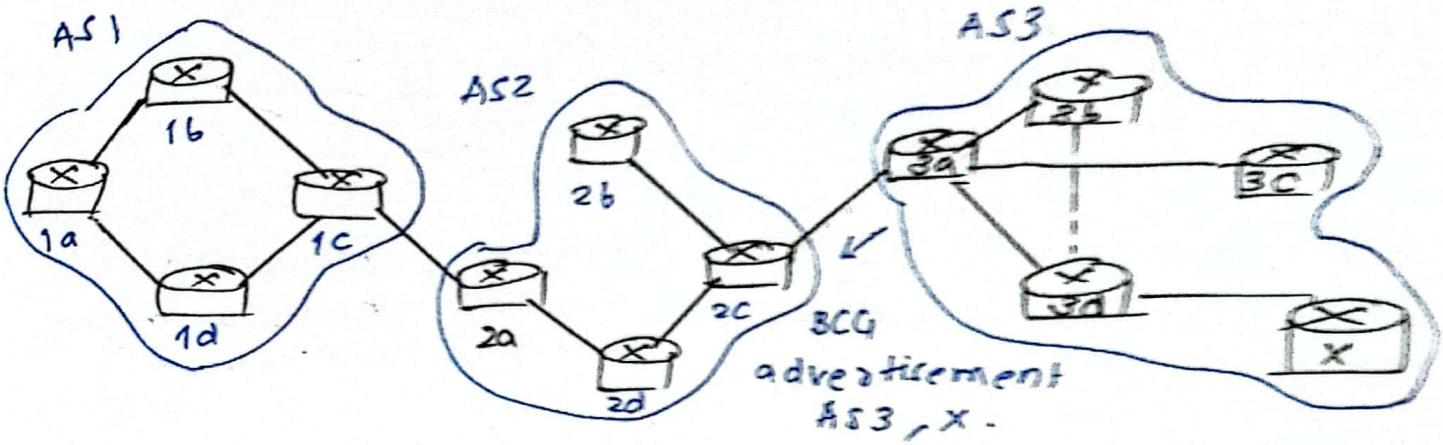
1. Obtain destination network reachability info from neighboring. (eBGP)
2. Determine Routes to other network based on reachability information
3. Propogate reachability information to all AS-internal routers (iBGP)
4. advertise destination reachability info.



Gateway protocols run both eBGP & iBGP protocols

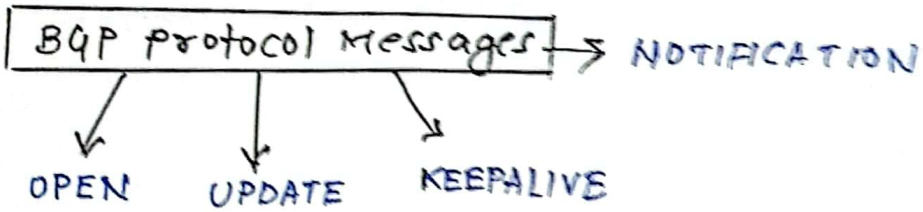
BGP basics

BGP session: two BGP routers ("peers") exchange BGP message over semi-permanent TCP connection



When AS3 gateway 3a advertises path AS3, x. to AS2 gateway 2c

AS3 promises to AS2 it will forward datagrams toward x



Hot potato Routing: choose local gateway that has intra-domain cost.

	Intra-AS Routing	Inter-AS Routing
policy	admin wants control over how its traffic is routed.	single admin, so policy less of an issues.
scale:	Hierarchical Routing saves table size, reduced update traffic	
performance	can focus on performance	policy dominates over performance

Software Defined Networking (SDN)

Why a logical centralized control plane?

① easier network management: avoid router misconfiguration
greater flexibility of traffic flows.

② Table-based forwarding (recall OpenFlow API) allows "programming" router

Centralized "programming": easier: compute tables centrally & distribute.

Distributed: "programming" more difficult: compute table as result of distributed algo (protocol) implemented in each-&-every router.

Data-plane switches

① API for table-based switch control
defines what is controllable, what is not

② protocol for communicating with controller (eg. openFlow)

SDN controller (network operating systems)

① Maintain network state information.

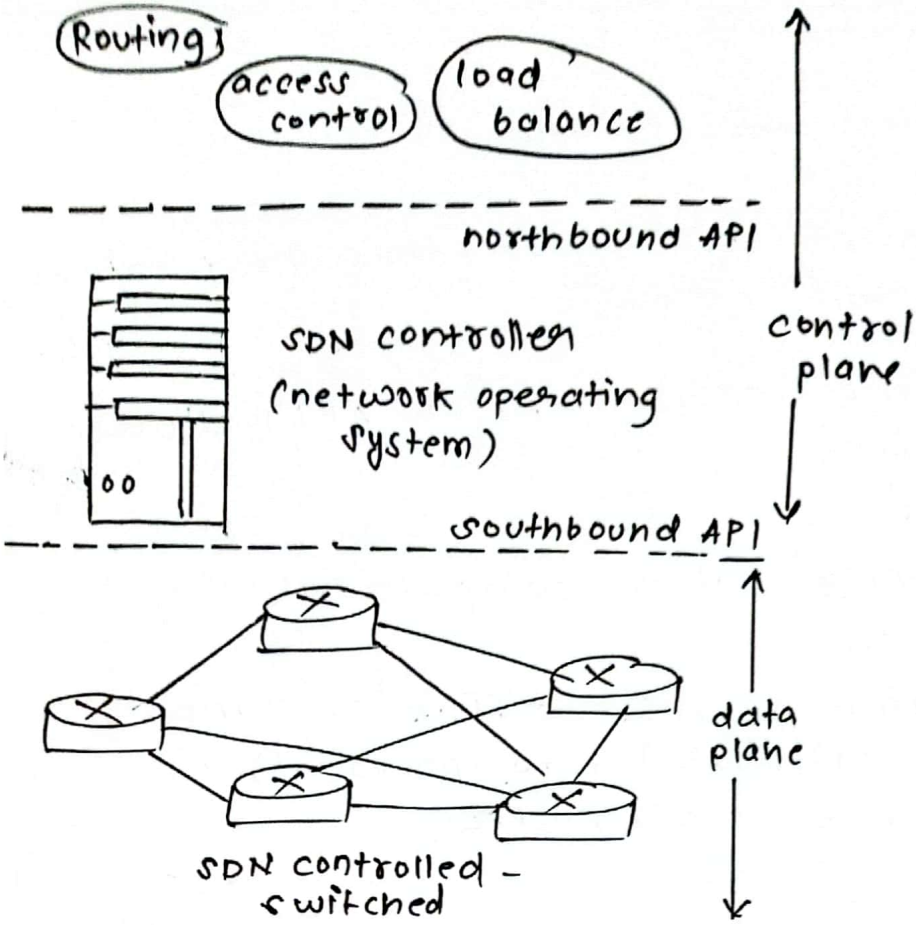
② Interacts with control applications "above" via northbound API

③ Interacts with network switches below via southband API

Network-controlled Apps

"brains" of control: implement control functions using lower-level service, API provided by SDN controller

unbundled: can be provided by 3rd party: distinct from routing



OpenFlow Protocol

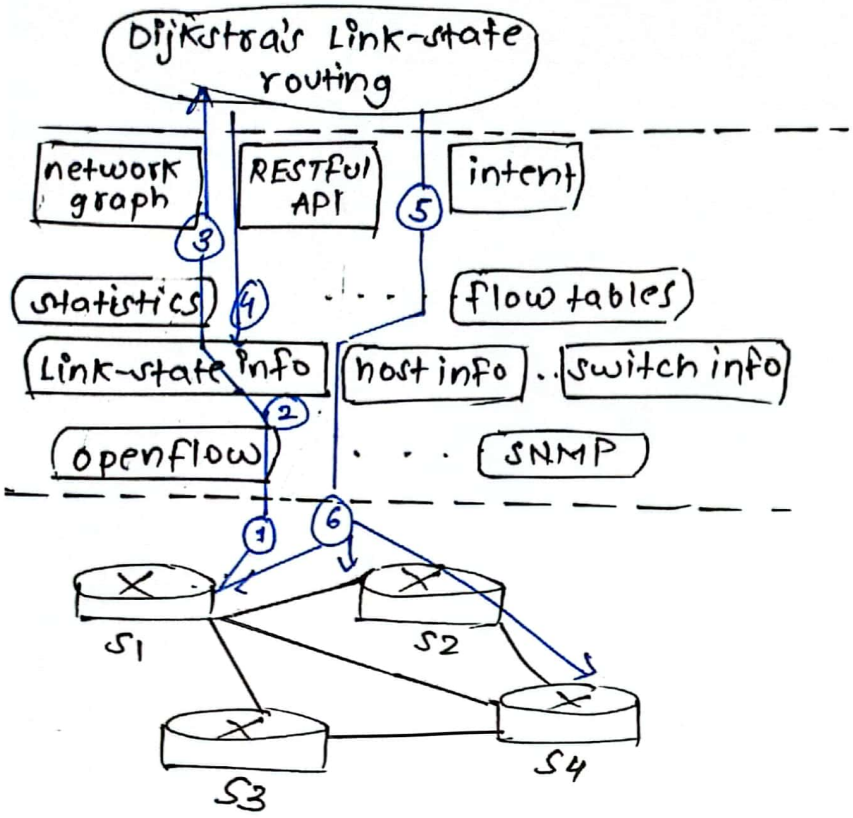
operates between controller, switch

TCP used to exchange messages
optional encryption.

openflow messages → symmetric

- **controller to switch**
  - → Configure queries/ set switch config
  - → Modify-state: add/delete entities in openflow tables
  - → packet/out: controller can send pkt out of specific port.
- **Asynchronous**

SDN: control/data plane interaction example:



wireless and Mobile Networks

wireless network Taxonomy

	single hop	Multiple hop
infrastructure	host connects to base station (wifi, cellular) which connects to large internet	host may have to relay through several wireless nodes to connect to larger internet: mesh net
no-infrastructure	No base station, no connection to larger internet (Bluetooth, ad hoc nets	no base station, no connection to large internet. may have to relay other a given wireless node MANET, DANET

wireless link characteristics

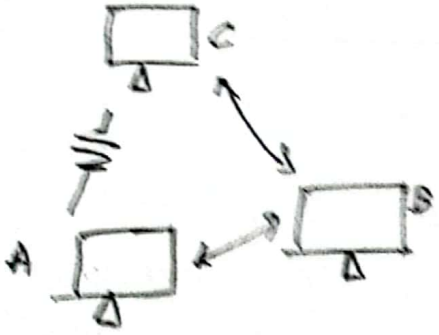
① Wireless radio signal attenuates (loses power) as it propagates (free space)

free space path loss (fsl)

② Multipath propagation: Radio signal reflects off objects ground, built environment arriving at destination

③ Interference from other sources on wireless network

④ → Hidden terminal
↳ Attenuation "hidden terminal"



code Division Multiple Access (CDMA)

↳ unique "code" assigned to each user.

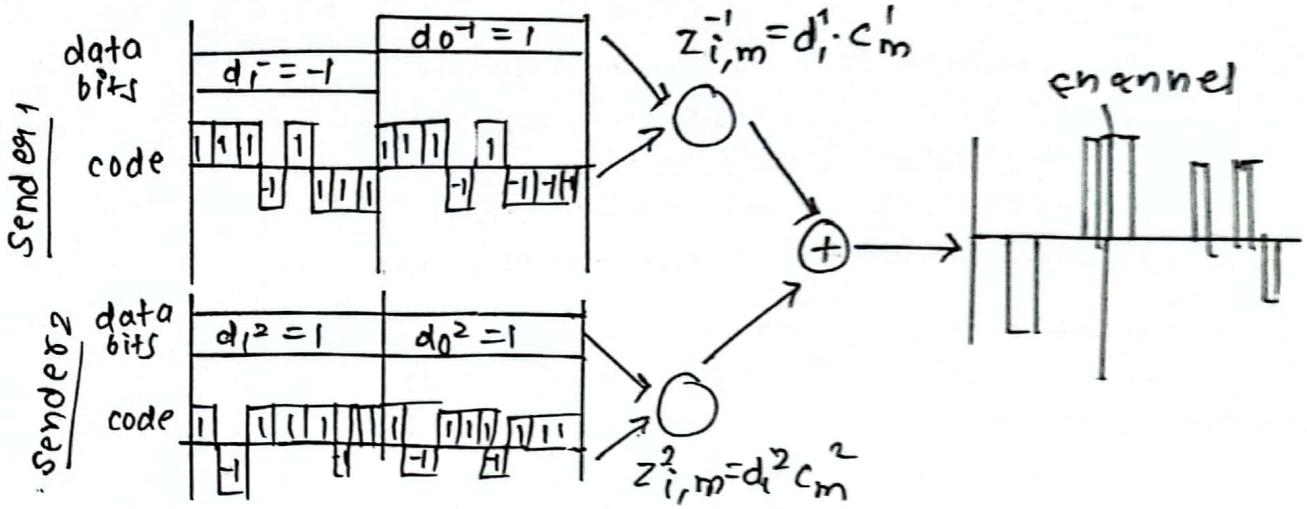
→ All users share same frequency, but each user has own "chipping" sequence

∀ transmit with minimal interference
helps to data

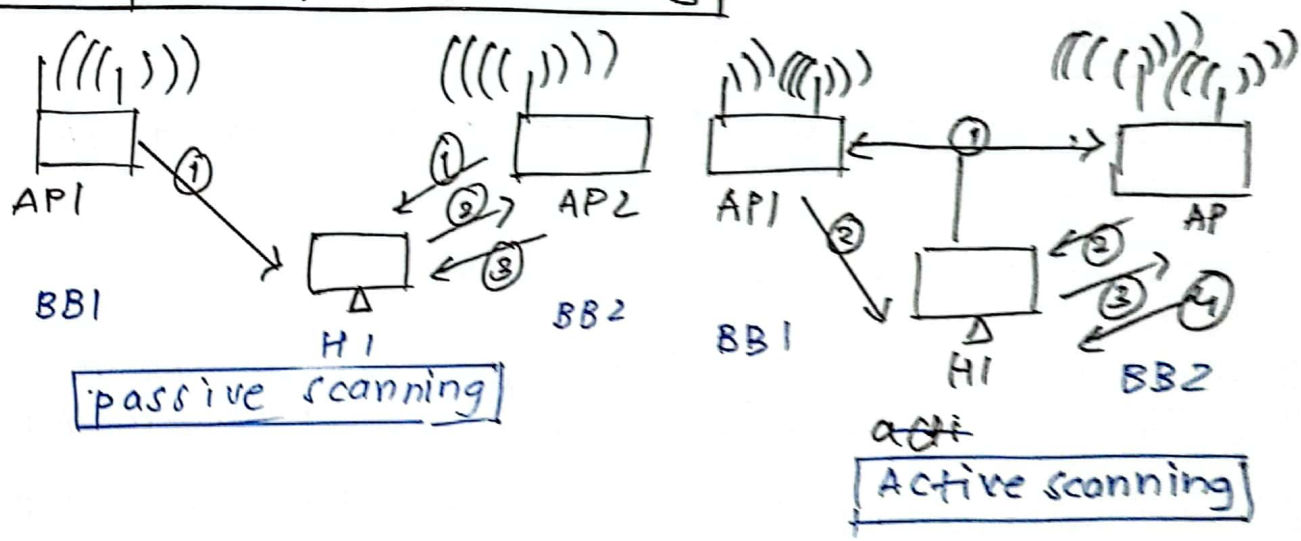
encoding: inner product

decoding: summed inner product

CDMA: two-sender interference



802.11 passive/active scanning



IEEE 802.11: Multiple access

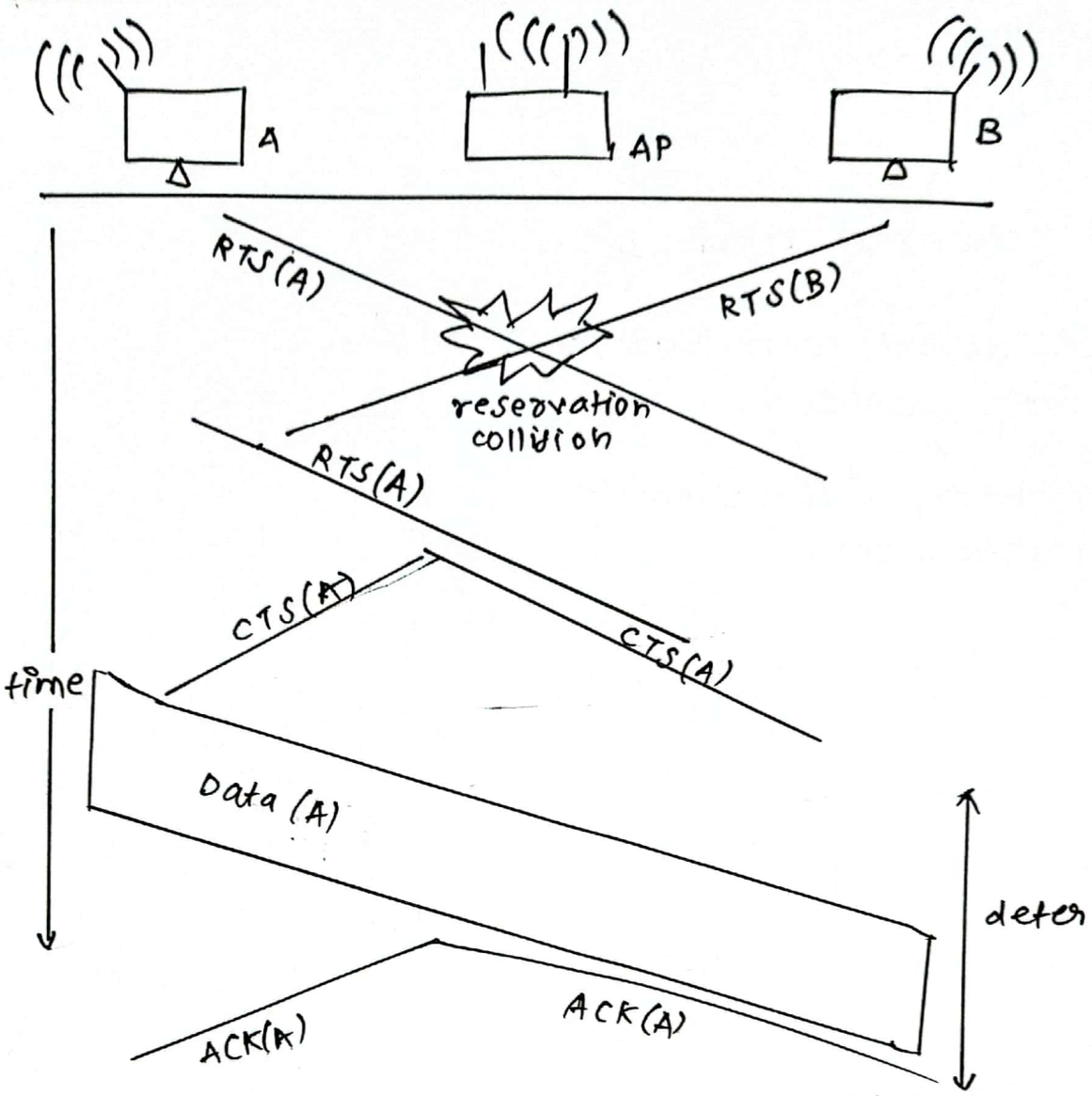
802.11: CSMA: sense before transmitting

disadvantage:

① High transmitting signal, weak received signal due to fading

cant' sense all collisions in any case

Collision Avoidance : RTS - CTS exchange



802.11 frame: addressing