CN [COMPUTER NETWORKS]

Dated: 03/Sept/2025

- Internet → "Network of Networks"
- Network ← LAN, WAN

WHAT IS "INTERNET"?

.. a "nuts and bolts" view Billions of connected computing devices :-

- hosts = end systems
- running network apps at Internet's "edge".

PACKET SWITCHES :- forwards packets (chunks of data).

.. routers, switches

Communication links :- ← fibre, copper, radio, satellite
transmission rate : bandwidth.

Networks :- collection of devices, routers, links :- managed by an organization

INTERNET : A "SERVICES" VIEW :-

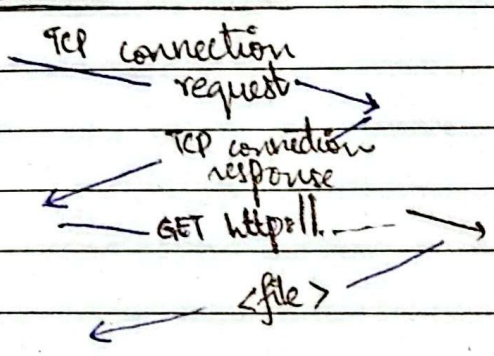
.. Infrastructure that provides services to applications :-
web streaming, email, games etc.

.. provides programming interface to distributed applications :-

PROTOCOL :- WHAT IS PROTOCOL ?

Human protocols :- What time is it? [request]
other person responds. [response].
Professor asks something [request]
Students respond. [response].

Computer Network Protocol :-



.. Protocols define the format, order of messages sent & received among network entities, & actions taken on message transmission & receipt.

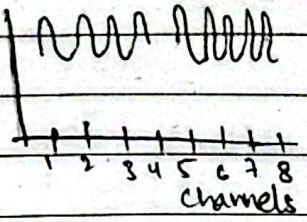
Dated:

Network edge :- → data centres.
- hosts → clients n servers

Access networks, physical media :- wired / wireless

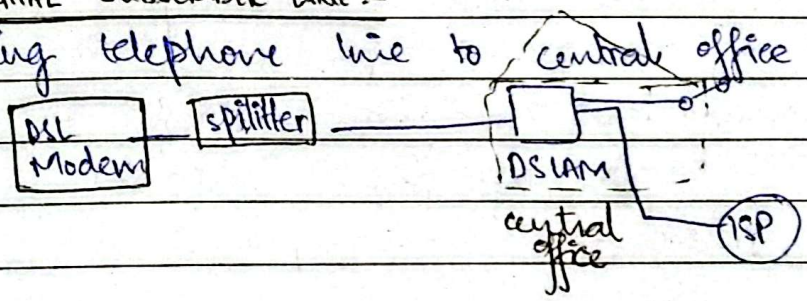
Q. How To Connect End Systems To Edge Routers?
- residential access nets. } ACCESS NETWORKS
- institutional access networks
- mobile access networks.

ACCESS NETWORKS :- CABLE - BASED ACCESS :-
FDM (FREQUENCY DIVISION MULTIPLEXING) :- different channels transmitted in different frequency bands.

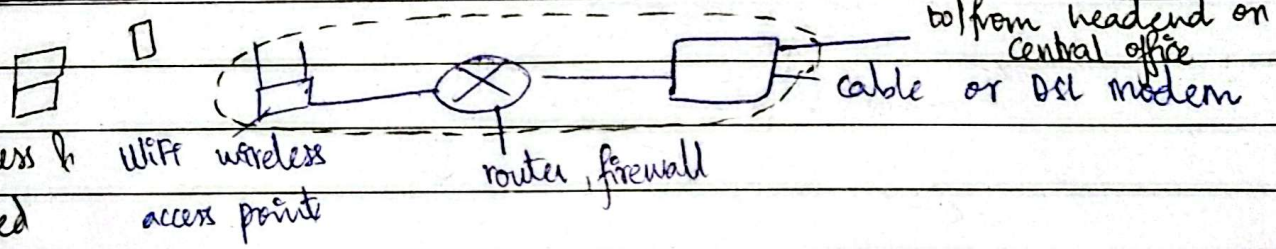


- HFC : hybrid fiber coax
  ↳ asymmetric

- DSL DIGITAL SUBSCRIBER LINE :-
use existing telephone line to central office DSLAM.



ACCESS NETWORKS : HOME NETWORKS.

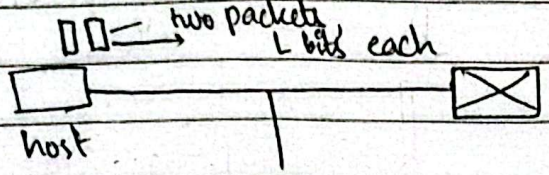


WIRELESS ACCESS NETWORKS (~ 100ft)
WIDE-AREA CELLULAR ACCESS NETWORKS (10's km)

Dated:

Host : sends packets of data :-

↳ host → takes application message
.. breaks into packets of length L bits.
.. transmits packet into access network at transmission rate R.
.. link transmission rate, aka link capacity, aka link bandwidth.



R : link transmission rate

$$\text{Packet transmission delay} = \frac{\text{time needed to transmit L-bit packet into link}}{} = \frac{L \text{ (bits)}}{R \text{ (bits/sec)}}$$

Physical media :-

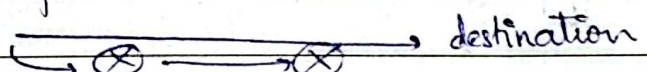
.. physical link :- lies between transmitter & receiver.
.. Guided media :- signals propagate in solid media : copper, fiber, coax.
.. unguided media :- signals propagate freely, e.g. :- radio.
.. Twisted pair (TP) :- 2 insulated copper wires.

Coaxial cable :-
↳ bidirectional
↳ two concentric copper conductors

Fiber optic cable :-
.. high speed operation
.. low error rate.

THE NETWORK CORE :-

↳ mesh of interconnected routers.



packets forwarded, from one router to another so that packets are received by destination.

Forwarding ——— routing algorithm

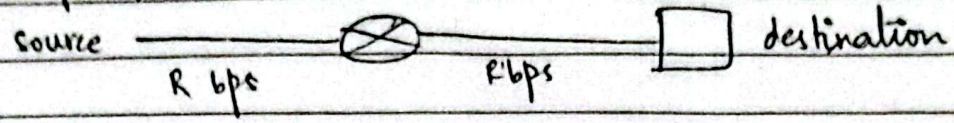
local forwarding table	
header value	output link
0100	3
0101	2
0111	2
1001	1

Dated:

•- local forwarding action ——— global routing algorithm implemented.

Packet - Switching :- store - and - forward :-

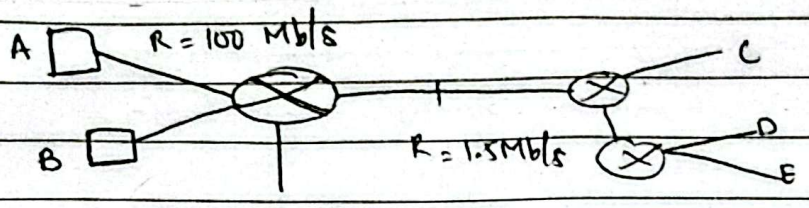
L bits per packet



•- pkt transmission delay :- L/R.

•- store and forward :- entire packet must arrive at router before it can be transmitted on next link.

Packet - Switching : Queing



queue of packets waiting for transmission over output link.

Packet Queing and Loss :-

•- if arrival rate (bps) >> transmission rate (bps).

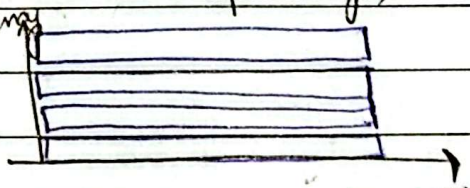
•- packets will queue, waiting to be transmitted on output link.

•- packets can be dropped (lost) if memory (buffer) in router fills up.

Circuit Switching :- end-end resources allocated to, reserved for "call" between source and destination.

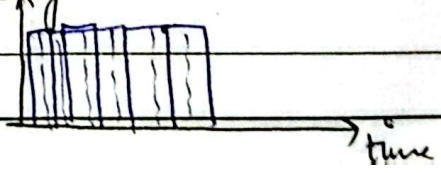
FDM (Frequency Division Multiplexing) :-

f → band frequency



TDM (Time Division Multiplexing) :- frequency

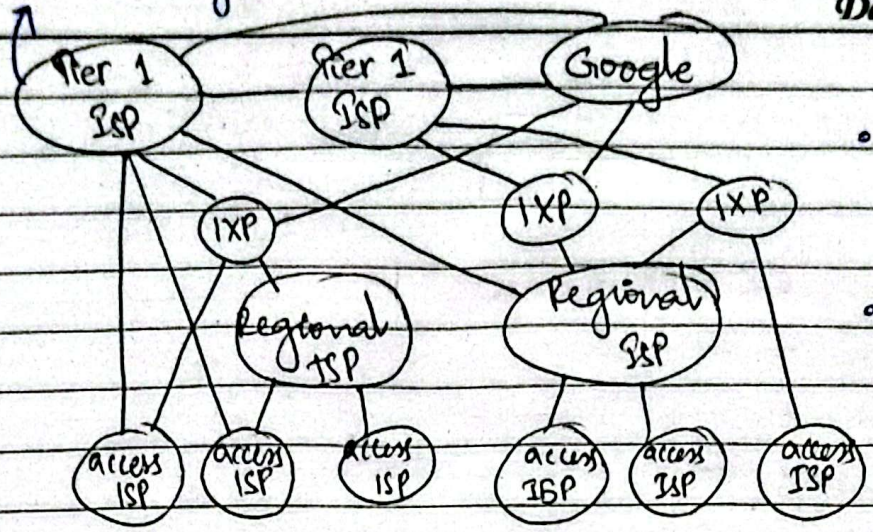
time → slot



commercial ISPs
national & International coverage

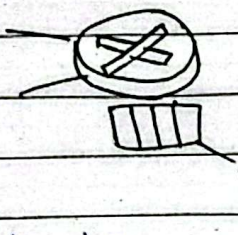
google, Microsoft
content provider Network.

Dated:



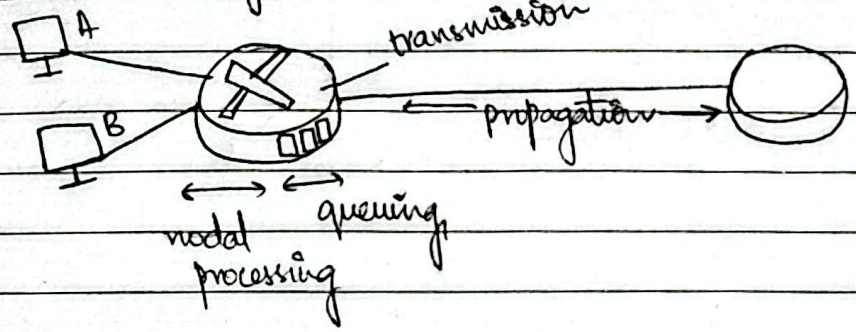
- private network that connect its data center to Internet.
- IXP = Internet Exchange Point

PACKET DELAY & PACKET LOSS :-



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

Packet delay :- four sources



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

d_{proc} : nodal processing
- check bit errors
- determine output link

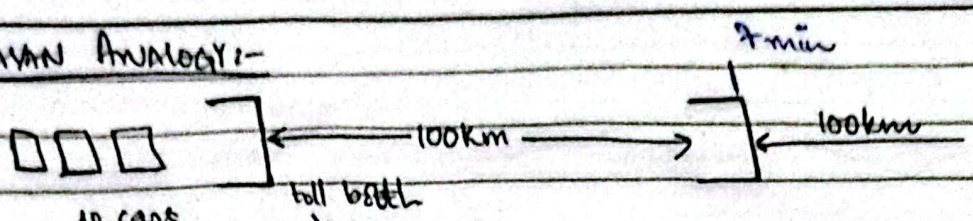
d_{queue} : queuing delay
time waiting at output link for transmission.
depends on congestion level of router.

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

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

Dated:

CARAVAN ANALOGY:-



one min to service a car

total time to service all = 10 * 1min = 10min

$d_{prop} = \frac{100}{1000} = 0.1$ hr * 60 = 6 mins

After 7 mins first car will arrive at the second booth.

Packet Queuing Delay:-

$\frac{L \cdot a}{R}$ = $\frac{\text{arrival rate of bits}}{\text{service rate of bits}}$ "traffic intensity"

a: average packet arrival rate

L: packet length (bits)

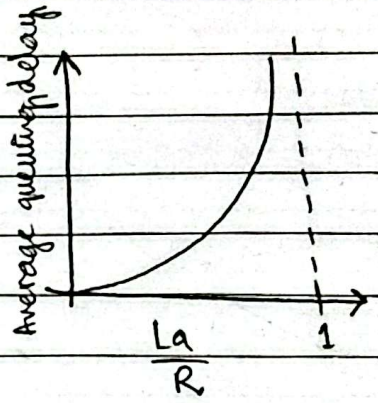
R: link bandwidth (bit transmission rate)

$\frac{La}{R} \sim 0$ avg. queuing delay small.

$\frac{La}{R} = 1 \rightarrow$ avg. queuing delay large

$\frac{La}{R} > 1$ avg. delay infinite

more "work" arriving than it can be serviced.



traceroute:- provides delay measurements from source to router along end-end Internet path towards destination.

THROUGHPUT:- rate (bits/time unit) at which bits are being sent from sender to receiver

- instantaneous: rate at given point in time
- average: rate over longer period of time

R_s: sending rate of the server.

server client

Dated:

$R_s < R_c$ What is average end-end throughput.

Actual Throughput = R_s (limited by sender).

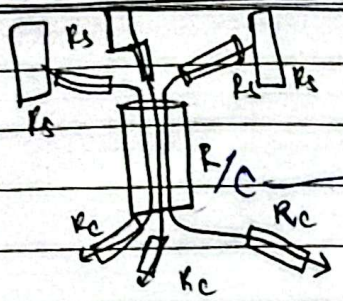
$R_s > R_c$ what is average end-end throughput.

Actual Throughput = R_c (limited by receiver).

key idea: The slowest part of the path determines the overall throughput.

Throughput = min (R_s, R_c)

Network scenario :-



throughput
↳ min (R_s, R/c, R_c)

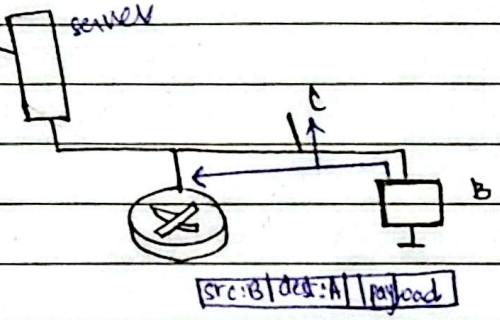
# of connections fairly sharing backbone bottleneck link

Bottleneck link → min (R_s, R/c, R_c)

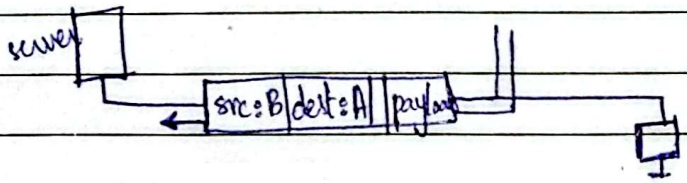
link utilization = $\frac{\text{Actual throughput}}{\text{Capacity}}$

Packet "sniffing" :-

- process of capturing & analysing data packets as they travel over a computer network.



IP spoofing :- injection of packet with false source address.



DOS (Denial of Service) :- attackers make resources (server, bandwidth) unavailable to legitimate traffic by overwhelming resource with bogus traffic.

Dated:

- select targets
- break into hosts around the network
- sends packets to target from compromised hosts.

Layered Internet protocol stack :-

Application layer :-

- Programmable layer.
- Cookies work on application layer.
- supports network applications.
- defines how processes in different end systems communicate by exchanging messages.
- They are rules that describe message types, structure, meaning & typing.

Ex :- HTTP, IMAP, SMTP, DNS

- Application specific headers are appended to the message.

Transport Layer :-

- Process to process data transfer.
- Connection control - can be connection oriented TCP or connectionless UDP.
- Error detection & recovery - detects lost/corrupted data & retransmits TCP.
- Adds header information such as source/dest no., seq. no. & checksum etc.

Ex: TCP, UDP

- Connection oriented
- reliable
- Used in Web browsing, emails, file transfer.
- Congestion control

- unreliable
- connectionless
- faster
- streaming/gaming
- No congestion control

Network Layer :- ~~- End-to-end datagram delivery across multiple links.~~ ~~- Process-to-process data transfer~~

- adds info. regarding source/dest. IP addresses as well as other routing information.
- ID, routing protocols

Dated:

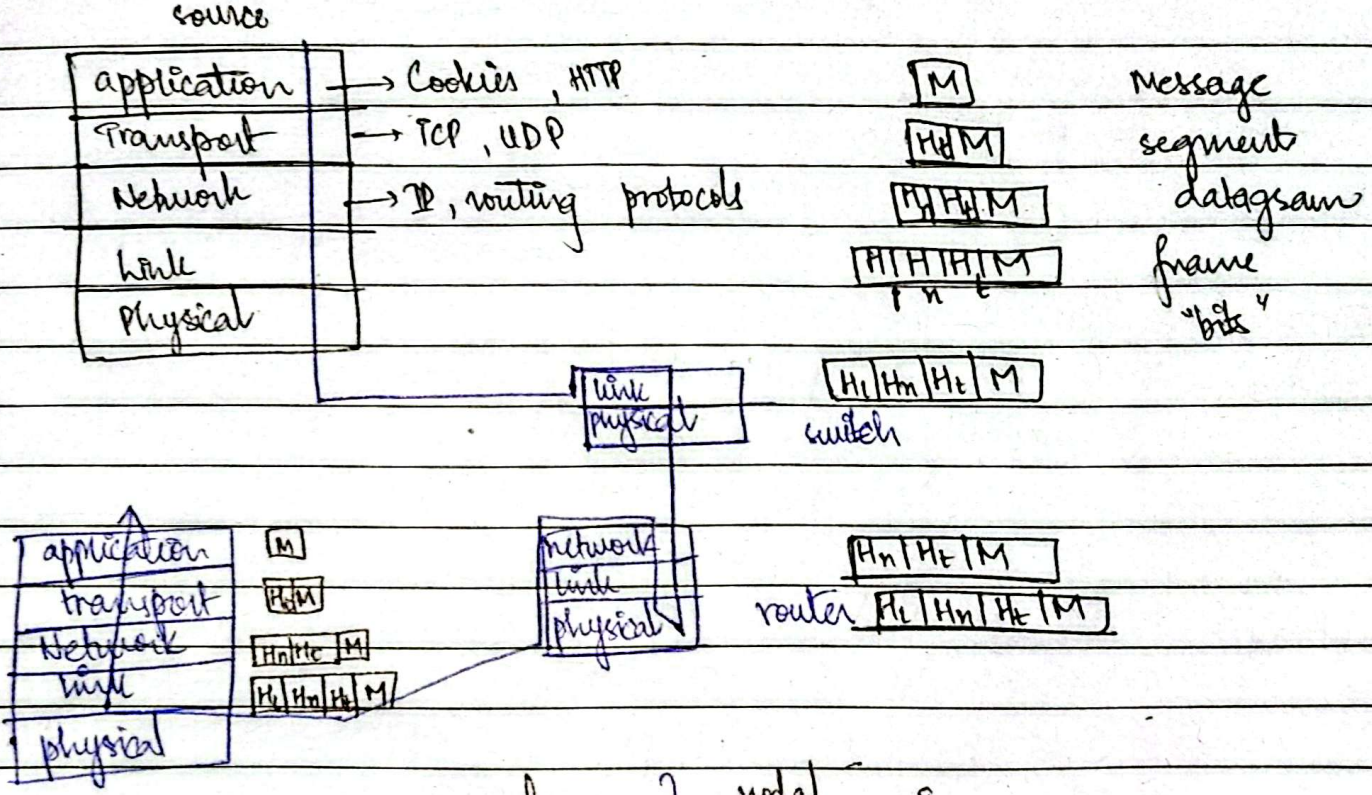
- routing → Determines the best path for data to travel from source to destination across networks.
- detect routing issues or packet loss & report back to the source

Link layer :-

- The link layer encapsulates each IP packet into a frame suitable for transmission over the physical network.
- Frames includes source & destination MAC addresses.

Physical Layer :-

- Converts frames to bits so that it can be sent over the transmission medium.
- Defines how bits flow between devices.
  - ↳ simplex, half duplex, full duplex



delay components :-

- d_{proc}
- d_{trans}
- d_{prop}
- d_{queue}

→ nodal processing

→ constants.

→ variable.

Problems From Book Dated:

P2. length L, N links of transmission rate R.

$= \frac{N*L}{R}$

- Time for 1st packet $N\frac{L}{R}$
- Each of the P-1 packets adds one more $\frac{L}{R}$

$$\text{Total} = N\frac{L}{R} + (P-1)\frac{L}{R} = \boxed{\frac{(N+P-1)L}{R}}$$

time slot	1	2	3	4	5
link1:	P1	P2	P3	P4	....
link2:	-	P1	P2	P3	P4
link3:			P1		

P3. steady rate, relatively long period of time

a) Circuit switching → more better. It reserves capacity along the entire path for the duration of the connection.
↳ Gives guaranteed throughput (the reserved bandwidth = r per call)
↳ Gives low & predictable delay.

b. $\sum_i r_i < C$, no congestion occurs.
no need for congestion control.

Queuing Condition :-

$R_{total} > C$: persistent congestion
capacity
The link cannot transmit all arriving bits as fast as they arrive.

$R_{total} = C$: on average link keeps up so no congestion

$R_{total} < C$: no persistent congestion.

Ex :-
link capacity $C = 10$ Mbps
$r_1 = 2$ Mbps, $r_2 = 3$ Mbps, $r_3 = 4$ Mbps
$R_{total} = 2+3+4 = 9$ Mbps < 10 Mbps.

Dated:

RG A —— R bps, L bits, s m/s, m meters —— B

a. dprop = m/s

b. dtrans = L/R

c. dend-to-end = L/R + m/s

d. Last bit is just leaving it at the start of the link, entering the medium.

e. if dprop > dtrans

At t = dtrans, the first bit is still in flight & has travelled a distance:- $d = s \cdot d_{trans}$

$$d = s \cdot \frac{L}{R}$$

f. if dprop < dtrans

at t = dprop, since dprop < dtrans, by time t = dtrans the first bit has already arrived at B.

g. $s = 2.5 \times 10^8$ m/s, $L = 1500$ bytes $= 1500 \times 8 = 12{,}000$ bits

$R = 10$ Mbps $= 10 \times 10^6$ bits/s

dtrans = dprop, find m

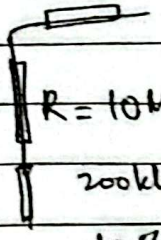
$d_{trans} = L/R = 12000 / 10000000$ s $= 0.0012$ s $= 1.2$ ms

set dprop = dtrans, so $m = s \cdot d_{trans}$

$m = 2.5 \times 10^8 \times 0.0012$

$$\boxed{m = 3.0 \times 10^5 \text{ m} = 300{,}000 \text{ m} = 300 \text{ km}}$$

P8. a. when ckt switching is used, how many users can be supported? Max users $= \frac{C}{n} = \frac{10000}{200} = 50$



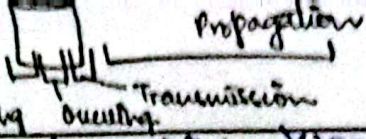
b. P(user transmitting) = 0.1

c. Probability that exactly n users transmit simultaneously

Let X = number of users transmitting at a given instant.

$X \sim \text{Binomial}(N = 120, p = 0.1)$

$$P(X = n) = \binom{120}{n} (0.1)^n (0.9)^{120-n}$$



Dated:

c) $P(X \geq 51) = \sum_{n=51}^{120} \binom{120}{n} (0.1)^n (0.9)^{120-n}$

$E(X) = np = 120(0.1) = 12$

$d_{prop,i} = \frac{d_i}{s_i}$, $d_{trans,i} = \frac{L}{R_i}$

P10. $d_{proc} = 3$ ms , $d_1 = 5000$ km $d_2 = 4{,}000$ km , $d_3 = 1{,}000$ km

$s = 2.5 \times 10^8$

packet = 1500 bytes, transmission rate of all three links are 2.5 Mbps.

$d_{end\text{-}to\text{-}end} = d_{trans,1} + d_{prop,1} + d_{proc} + d_{trans,2} + d_{prop,2} + d_{proc} + d_{trans,3} + d_{prop,3}$

$= \frac{L}{R_1} + \frac{d_1}{s_1} + d_{proc} + \frac{L}{R_2} + \frac{d_2}{s_2} + d_{proc} + \frac{L}{R_3} + \frac{d_3}{s_3}$

b. $d_{trans,i} = \frac{12000}{2.5 \times 10^6} = 4.8$ ms

All three links have the same R, so each $d_{trans} = 4.8$ ms

$d_{prop,i} = d_i / s_i$

Link 1 : $d_{prop,1} = 5 \times 10^6 / 2.5 \times 10^8 = 0.025 = 20$ ms

Link 2 : $d_{prop,2} = 4 \times 10^6 / 2.5 \times 10^8 = 0.016\,s = 16$ ms

Link 3 : $d_{prop,3} = 1 \times 10^6 / 2.5 \times 10^8 = 0.004\,s = 4$ ms

$d_{total} = (4.8 + 20) + 3 + (4.8 + 16) + 3 + (4.8 + 4)$

$= \boxed{61.4 \text{ ms}}$

P11. In above P10, suppose $R_1 = R_2 = R_3 = R$, $d_{proc} = 0$. no store-&-forward, immediate forwarding.

$d_{end\text{-}to\text{-}end} = d_{trans,1} + d_{prop,1} + d_{prop,2} + d_{prop,3}$

$4.8 + 20 + 16 + 4 = \boxed{44.8 \text{ ms}}$

P12. $L = 1500$ bytes $= 12{,}000$ bits , $R = 2.5$ Mbps

Currently transmitting packet → halfway done → remaining bits $= 1/2 = 6{,}000$ bits

$d_{trans} = \frac{12{,}000}{2.5 \times 10^6} = 4.8$ ms

Dated:

Queuing delay = total time our packets wait before starting to transmit.
Remaining bits of currently transmitting packet $\frac{L}{2}$

6000 bits $\rightarrow \frac{6,000}{2.5 \times 10^6} = 2.4$ ms

Four packets in queue : each takes 4.8 ms So total 4.4.8 = 19.2 ms.
total queuing delay = 2.4 + 19.2 = 21.6 ms
d_{queue} = 2.4 + 19.2 = 21.6 ms
Remaining bits of current packet L-x.

$$d_{queue} = \frac{L-x}{R} + n \cdot \frac{L}{R}$$

P13. $d_{trans} = L/R$

Queuing delay per packet:-
Packet 1 : no waiting → delay = 0
Packet 2 : waits for 1 packet → delay = d_{trans}
Packet 3 : waits for 2 packets → delay = $2d_{trans}$
⋮
Packet N : waits for N-1 packets → delay = $(N-1)d_{trans}$.

$$\boxed{d_{queue,avg.} = \frac{N-1}{2} \cdot \frac{L}{R}}$$

CHAPTER 2. Dated:

E-MAIL :-

three components
- user agents → mail reader.
- mail servers
- SMTP

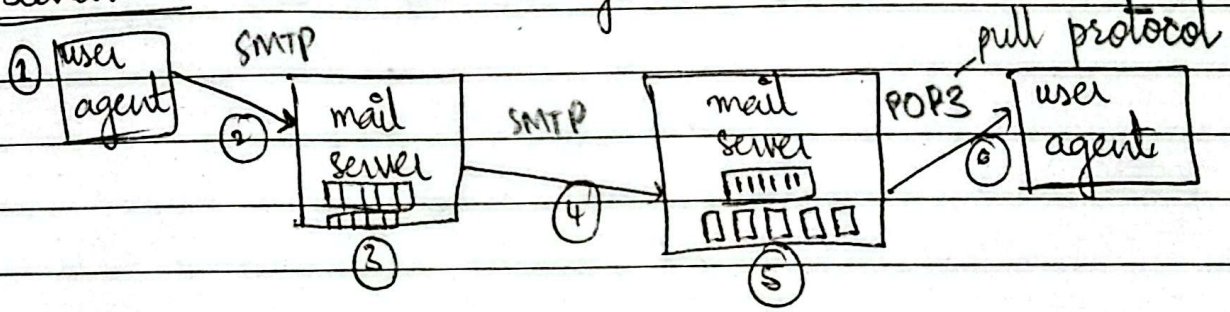
User Agent :- "Mail Reader"
↳ composing, editing, reading mail messages.

Mail Servers :-
- mailbox → contains incoming messages for user.
- message queue → of outgoing (to be sent) mail messages.
- SMTP protocol between mail servers to send email messages
  client :- sending mail server.
  "server" :- receiving mail server.

SMTP → uses TCP to reliably transfer email message from client to server, port 25
direct transfer : sending server to receiving server.
three phases of transfer
- handshaking (greeting)
- transfer of messages
- closure

- messages must be in 7 bit ASCII.

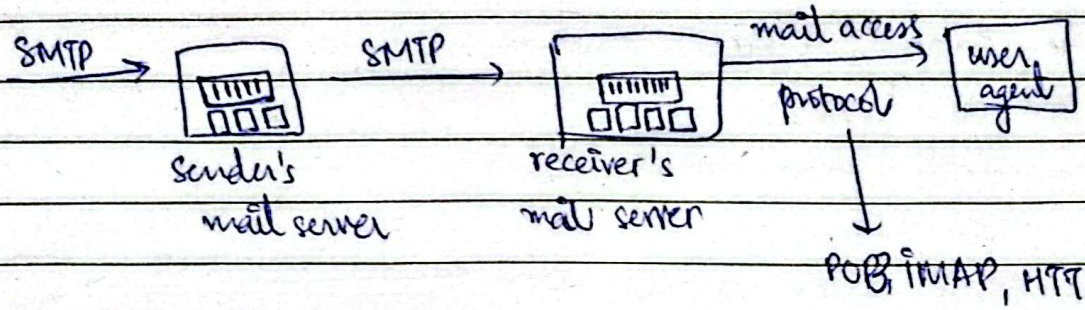
Scenario :- Alice Sends Message to Bob



- Alice uses User Agent to compose message to bob@nu.edu.pk
- Alice's User Agent sends message to her mail server;

message placed in message queue.
- client side of SMTP opens TCP connection with Bob's mail server.
- SMTP client sends Alice's message over TCP connection.
- Bob's mail server places the message in Bob's mailbox.
- Bob invokes his user Agent to read message.

SMTP
↳ uses persistent connections
↳ push protocol : SMTP , HTTP : Pull protocol
↳ SMTP & HTTP both are FTP.
↳ HTTP transfers files from web pages.
↳ Smtp transfers mails from mail server.
↳ Both have ASCII command/respone, status code.



POP3 :-

authorization phase	transaction phase
* client commands:	list : list message numbers
- user : declare username	retr : retrieve message by number
- pass : password	dele : delete
* server responses	quit
+ OK	
- ERR	

POP3 → 2 modes → "download and delete" (cannot reread email if he changes client)
↓
"download and keep" (copies of messages on different clients).

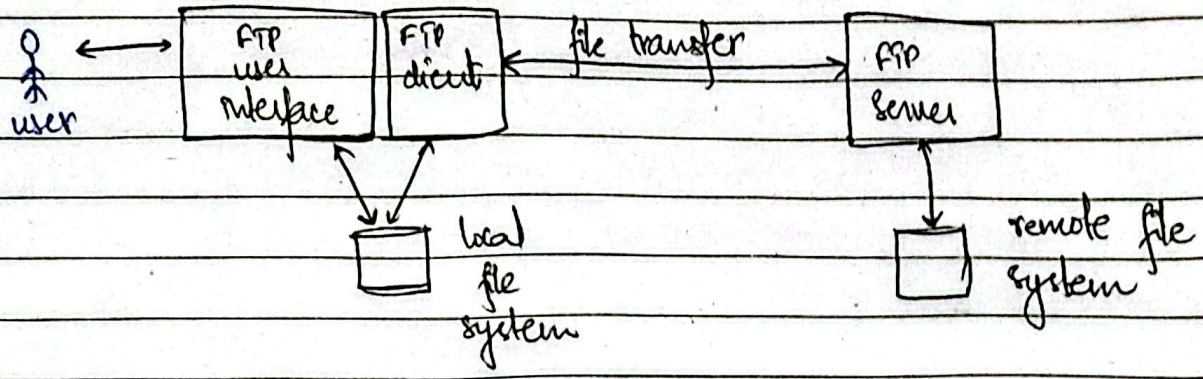
POP3 → stateless across sessions

Dated:

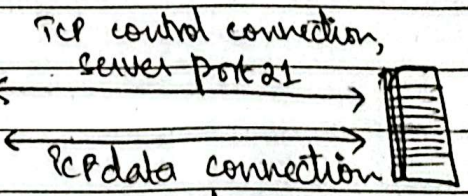
IMAP :-

↳ keeps all messages in one place : at server.
↳ allows users to organize message in folders.
↳ keeps user state across sessions :
names of folders & mappings between message IDs & folder name.

FTP [File transfer Protocol] :- Application layer protocol



- client : side that initiates transfer.
- server : remote host.
- transfer file to/from remote host.



- client authorized over control connection.
- when server receive file transfer command, server opens 2nd TCP data connection.
- after transferring one file, server closes data connection.
- server opens another TCP data connection to transfer another file.
- control connection : "out of band".
- FTP server maintains "state": current directory, earlier authentication.

Dated:

FTP COMMANDS, RESPONSES :-

sample commands

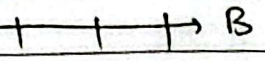
- USER username
- PASS password
- LIST return list of file in current dir.
- RETR filename retrieves (gets) file.
- STOR filename stores (puts) file onto remote host.

sample return codes :-

- 331 Username OK, password required
- 125 data connection already open; transfer starting
- 425 can't open data connection
- 452 Error writing file.

Problems

2.1.



a) $R_1 = 250$ kbps, $R_2 = 1$ Mbps, $R_3 = 500$ Kbps

$R_1 = 250,000$ bps, $R_2 = 1,000,000$ bps, $R_3 = 500,000$ bps

Throughput $= \min(R_1, R_2, R_3) = 250,000$ bps

b) File size = 8 Million bytes $= 8,000,000$ bytes $\times 8$

$= 64,000,000$ bits

250,000 ——— 1S.

64,000,000 ——— ↓

256S $= 4$ min 16 S

c) $R_2 = 200000$ bps

Throughput $= \min(R_1, R_2, R_3) = 200000$ bps

$t = \frac{64,000,000}{200,000} = 320$ S $= 5$ min 20s

Assumption :- ignored d_{queue}, d_{proc} & d_{prop}.

2.2. 1500 bytes, $d = 3100$ km, $R = 3$ Mbps, $s = 2.5 \times 10^8$ m/s.

a) $d_{prop} = \frac{3100 \text{ km}}{2.5 \times 10^8 \text{ m/s}} = \frac{3100 \times 10^3 \text{ m}}{2.5 \times 10^8 \text{ m/s}} = 0.0124$ S $= 12.4$ ms

b) $d_{trans} = \frac{1500 \times 8}{3 \times 10^6} = \frac{L}{R} = 250000$ S $= 4$ ms $= 4 \times 10^{-3}$ S

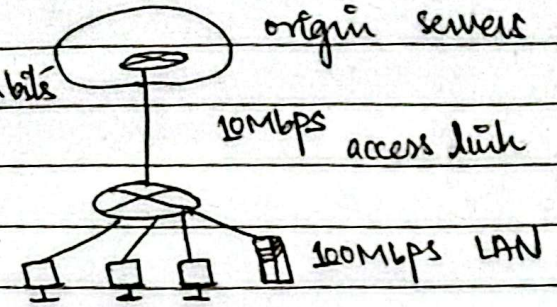
$t = d_{prop} + d_{trans} = 12.4$ ms $+ 4$ ms $=$ **16.4 ms**

length Dated:

$d_{prop} \rightarrow \frac{d}{s}$ doesn't depend on packet(L) & transmission rate(R).

$d_{trans} \rightarrow \frac{L}{R}$ depends on (L) packet length & (R) transmission rate.

2.3



(a). Average data rate to browsers over the access link.

Object size = 600K bits = 600,000 bits.

Request rate = 20 req/s.

Rate = 600,000 x 20 = 12,000,000 bits/s = **12Mbps.**

(b). LAN utilization = $\frac{12}{100}$ = 0.12 = **12%**

(c). Access link utilization = $\frac{12}{10}$ ← Average data rate to browser over the access link.

→ R of access link

= **120%** PROBLEM!!! CONGESTION OCCURS!

Web cache → 33% requests are satisfied locally, 67% requests are forwarded to origin servers.

Now only 67% of request go to origin server through the access link.

0.67 x 12Mbps = 8.04 Mbps

Access link utilization = $\frac{8.04}{10}$ = **80.4%**

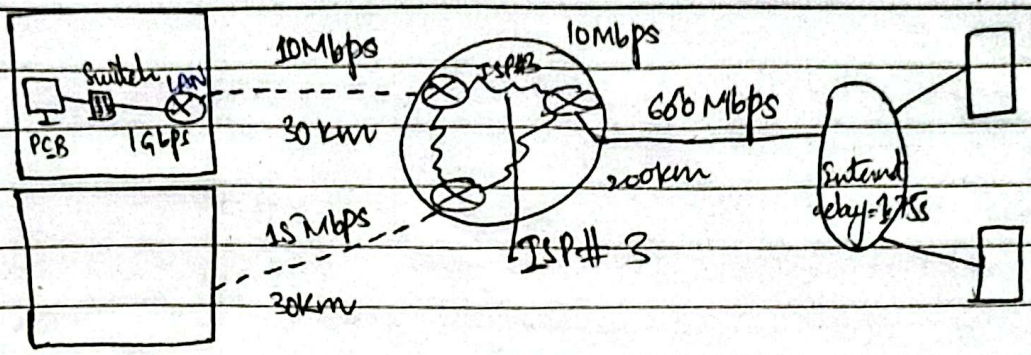
LAN utilization = 12% , LAN utilization remain unchanged.

$\frac{12 \times 100}{100}$

- Caching reduces external (access link) load & relieves the internet link, but the LAN still carries the full delivered content to browser (so LAN utilization stay the same).

Dated:

Q)



a) RB, file size = 1 Mbits, Avg. internet delay as 1.75s, ISP# 3 delay = 0.85s, 18 requests offered to service ISP # 3 per second. link is usable upto 90% capacity.

Average data rate = $1 \times 10^6 \times 18 = 18 \times 10^6$ bits reqs/s

file size / object size → Request/s

Access link = $\frac{18 \times 10^6 \times 10^6}{10 \times 10^6 \times 10^6} = 1.8$ s

10 Mbps access link rate

Link LAN = $\frac{18 \times 10^6 \times 10^6}{1 \times 10^9 \times 10^9} = 0.018$ s

Total delay → = 0.018 + 0.85 + 1.75 + Access link delay (1.8 s)

= 2.62 + unbounded.

Web cache placed on ISP #2.

b) hit ratio = 50%.

= 0.50 (LAN delay 0.018) + 0.5 (0.02 + 0.40 + 0.85 + 1.75)

= 0.01 + 1.33 = 1.34 (with cache).

CLIENT - SERVER PARADIGM :-

- Central server → provides services or resources
- Client → send requests to server.
- server processes the request & sends back the response.
- Web browsing (Web browser = client, Web server = server).

Client → Request → Server → Process → Response → Client.

server → permanent IP address, always on-host.

clients → may have dynamic IP addresses, don't communicate directly with each other.

Dated:

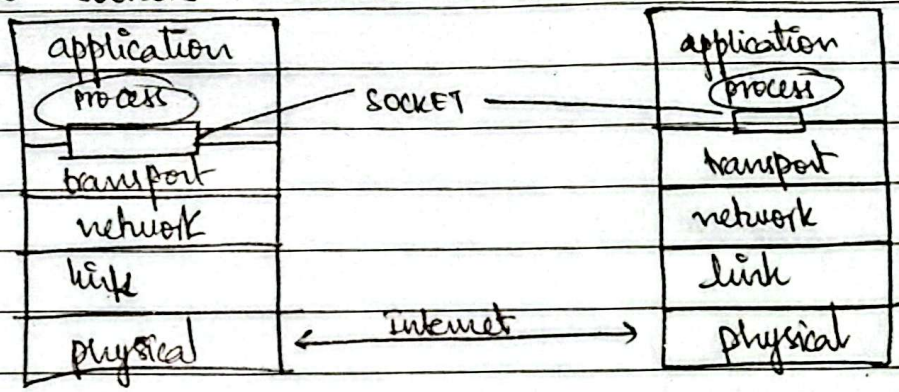
Peer-2-Peer :-

- end-systems directly communicate.
- peers acts as clients & server.
- peers are connected & change IP addresses.
- peers request & provide services from each other.
- less secure that client server, complex management since IP addresses of clients are dynamic (changing).

Sockets :-

- processes sends/receive messages to/from its socket.
- socket is like a door.
- sending processes shoves message out of door.
- sending process relies on transport infrastructure on other side of door to deliver message to socket at receiving process.

two sockets involved :- one on each side.



To receive messages, processes must have an identifier :-
IP address + Port no.

Port no. above is not enough because alot of processes can be running on the same host/same port no.

Application layer protocol defines :-

- types of message exchanged :- e.g request, response
- message syntax :- what fields in a message & how fields are structured.
- message semantics :- meaning of information in fields.

Dated:

- rules :- how process send & receive message
- timing :- when processes send & respond to messages
- open protocols :- everyone has access to protocol definition.
  - ↳ allows for interoperability.
- proprietary protocols e.g :- Zoom

Transport Service :- TCP model, UDP model

- ↳ data integrity
- ↳ timing
- ↳ throughput
- ↳ security

file transfer, download → FTP
e-mail transfer, email → SMTP
web documents, text-messaging → HTTP

no loss, elastic throughput, Not time sensitive → TCP

audio, video, gaming → HTTP

loss tolerable, throughput minimum, Yes : time sensitive, UDP

TCP, UDP = transport Layer protocols :-

TCP	UDP
- reliable	- less reliable (unreliable data transfer)
- congestion control	- no congestion
- flow control	- no flow control
- connection oriented (Hw sender & receiver)	- connectionless
- slower	- faster
- overhead → more	- overhead - less

- Both dont have timing, minimum throughput guarantee security
- TCP mostly used in sending email, docs etc
- delay insensitive.
- UDP mostly used where less is tolerant such as audio, video, gaming
- delay sensitive.

Securing TCP :-

TLS [Transport Layer Security] :- provides encrypted TCP connections

- data integrity
- end-point authentication.
- implemented in Application layer.

Dated:

Web page → contains objects

www.someschool.edu/someDept/pic.gif

hostname | pathname

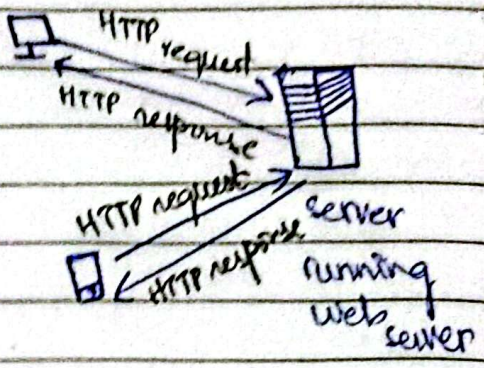
HTTP :-

client → receives web objects, REQUEST

Server → send web objects, RESPONSE

HTTP uses "TCP" → encryption TLS

Application control protocol



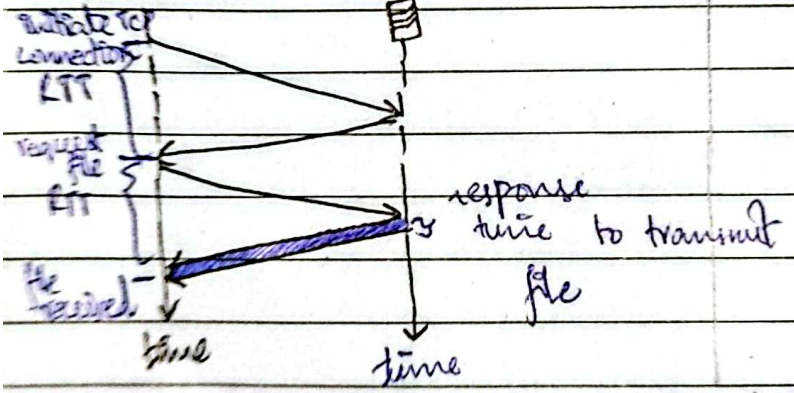
Steps :-

1. TCP connection opened.
2. HTTP request / response.
3. TCP connection closed.

- HTTP is stateless!

server maintains no info. about past client requests.

Non-Persistent HTTP	Persistent HTTP
1- TCP connection opened.	1- TCP connection opened to a server.
2- almost one object sent over TCP connection	2- Multiple objects can be sent over single TCP connection b/w that client & server.
3- TCP connection closed.	3- TCP connection closed.
Response time :- 2RTT + file transmission time	Response time : 1 + N
	1 → to open TCP connection
	N → no. of object to be requested

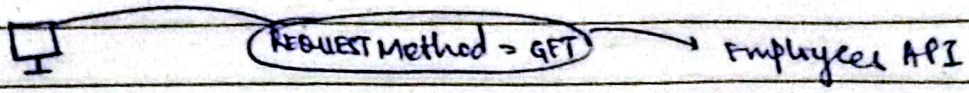


type of HTTP message :- REQUEST, RESPONSE

Dated:

HTTP Request Methods :- GET, POST, PUT, HEAD :-

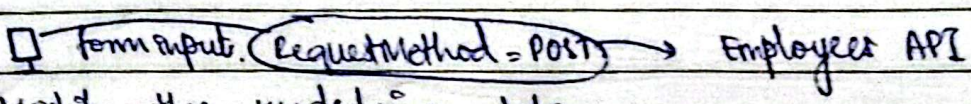
HTTP GET REQUEST :-



- retrieves data GET

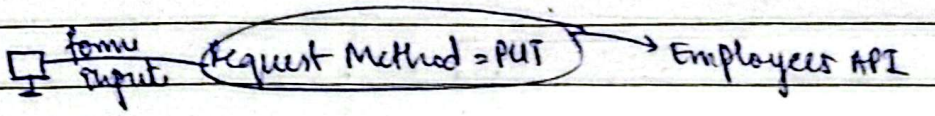
URL ⟶ / Get request message

- requesting / getting data from server.



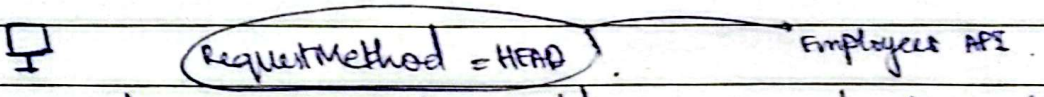
- Modify the underlying data
- Create new resource

for e.x :- new employee fills out input form & we want to add that employee.



- Modify underlying data
- update existing resources
- uploads / new file / object to server.

Ex :- update employee dept, job_title



- requests headers (only) that would be returned if specified URL were requested with an HTTP GET method.

Status codes :-

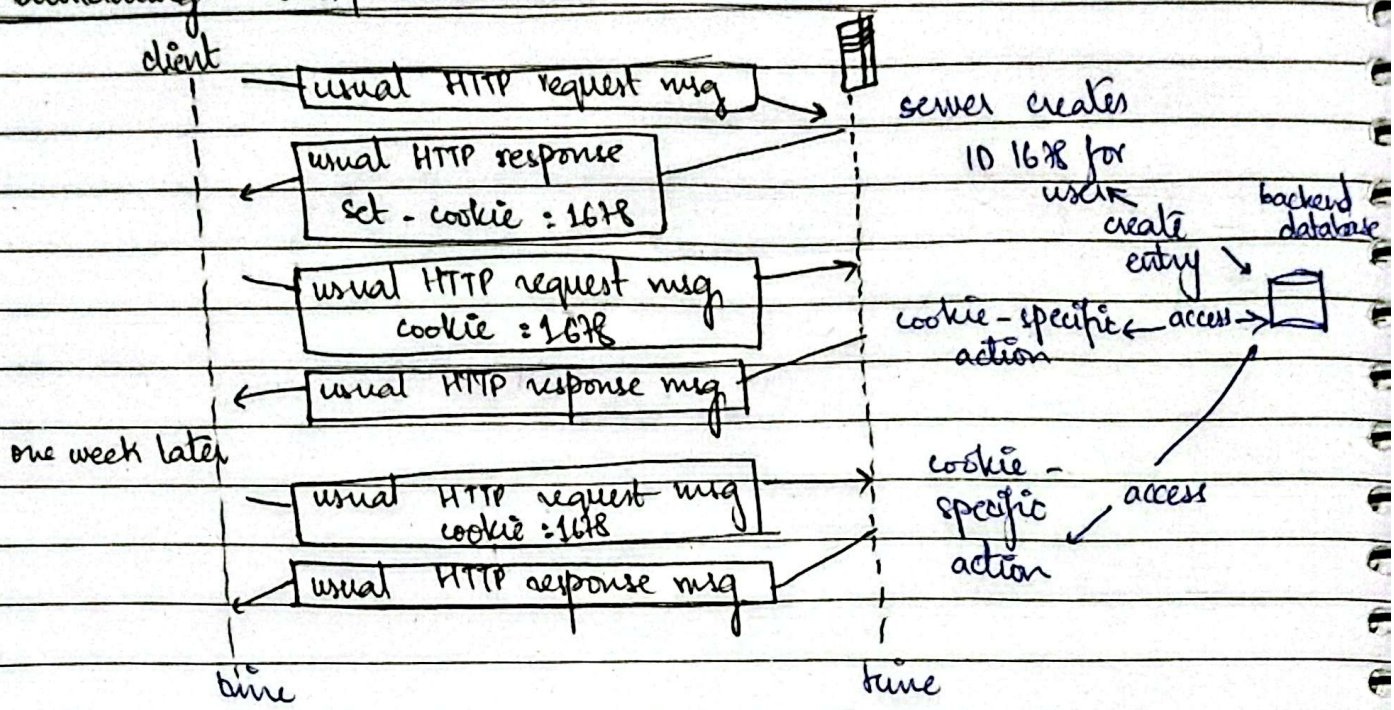
200 OK 301 Moved Permanently, 400 Bad Request, 404 Not Found 505 HTTP version Not Supported.

Cookies :- Web site & client browser use cookies to maintain some state between transactions.

four components :- Dated:

1) cookie header line of HTTP response message.
2) cookie header line in next HTTP request message.
3) cookie file kept on user's host, managed by user's browser.
4) backend database at web site.

Maintaining user/server state : cookies



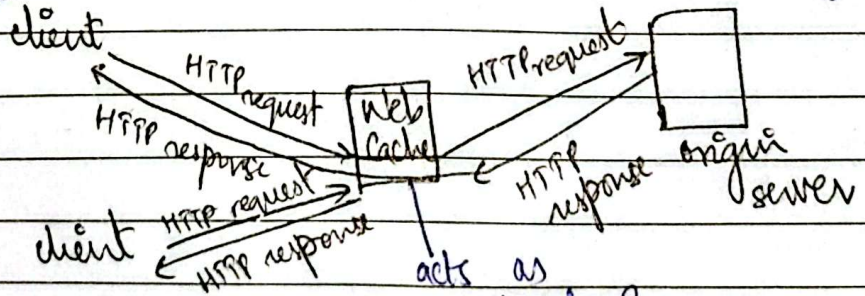
Cookies :- can be used for authorization, shopping carts, recommendations, user session state.

Q: How to keep state?

- at protocol endpoints : maintain state at sender/receiver over multiple transactions.
- in messages : cookies in HTTP messages carry state.

Web Caches :- aka Proxy Server

Goal : satisfy client requests without involving origin server.

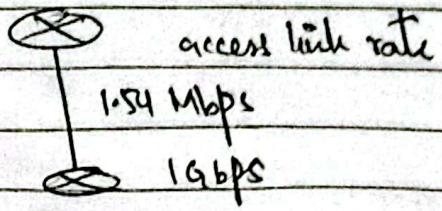


web cache
↳ reduces load/congestion on origin server/access link.
↳ speeds up
↳ reduces response time for client request

Dated:

Caching example:-

- 15 req/s , 100k bits = object size
- RTT from institutional router to server : 2s



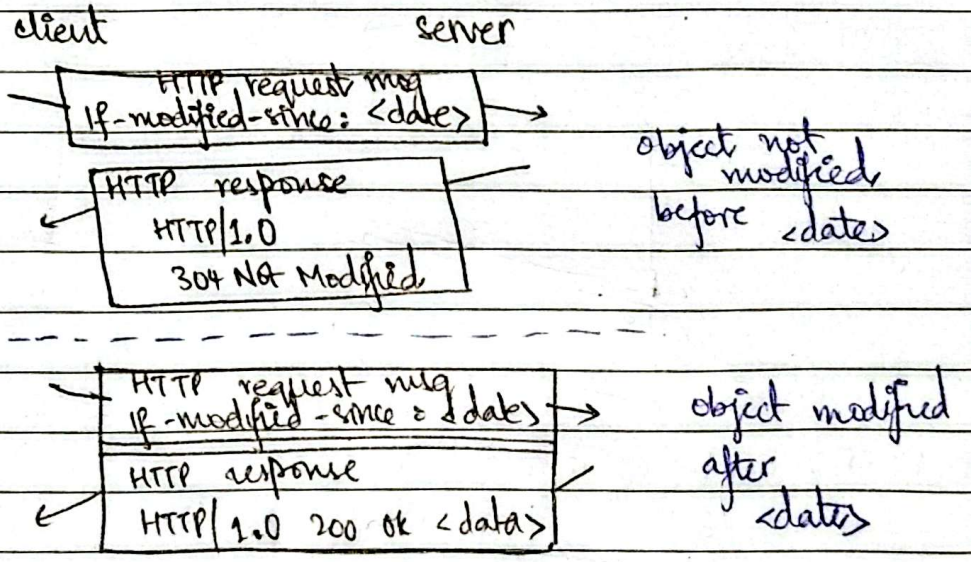
Average data rate = 100 000 × 15 = 1500000

utilization, access link: $\frac{1500000}{1.54 \times 10^6} = 0.97 \rightarrow$ 97.4%

LAN utilization = $\frac{1500000}{1 \times 10^9} = 1.5 \text{ bps} \times 10^{-3} = 0.0015$

web cache
60% request → origin server, 40% requests → served by web cache.
0.6 (1500000) = 900000
access link utilization = $\frac{900000}{1.54 \times 10^6} = 0.584 \rightarrow$ 58.44% (97.4%)

CONDITIONAL GET:-



PROBLEMS OF HTTP 1.1 :-

- server responds in-order (FCFS : first come first served scheduling) to GET requests.
- with FCFS, small object may have to wait for transmission (head-of-line (HOL) blocking) behind large

object(s).

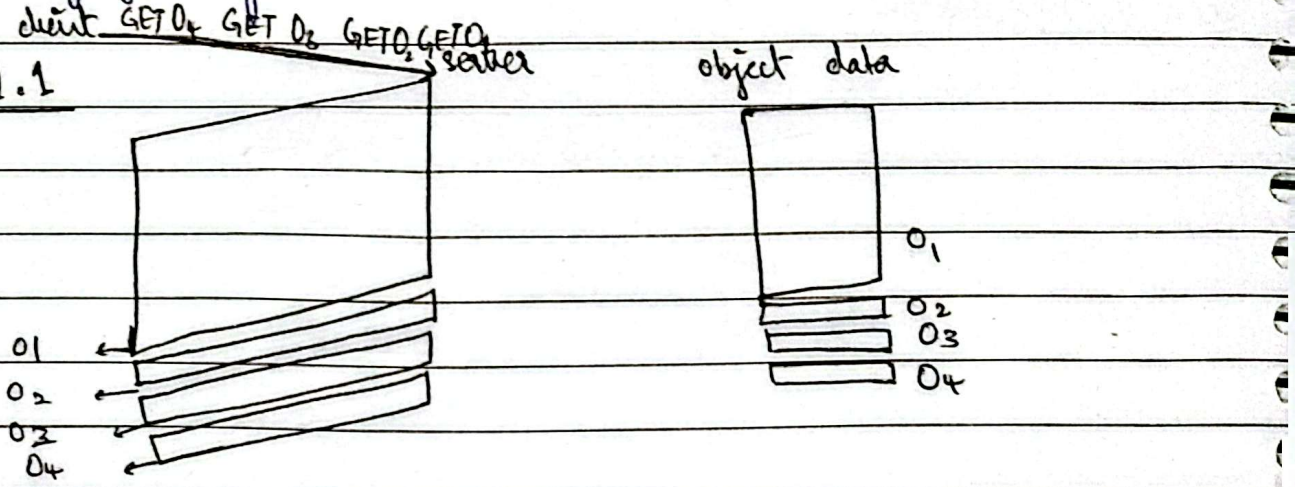
- If a TCP segment is lost, the transmission stalls until lost segment is retransmitted.

HTTP/2 (key goal: decreased delay in multi-object HTTP requests.)

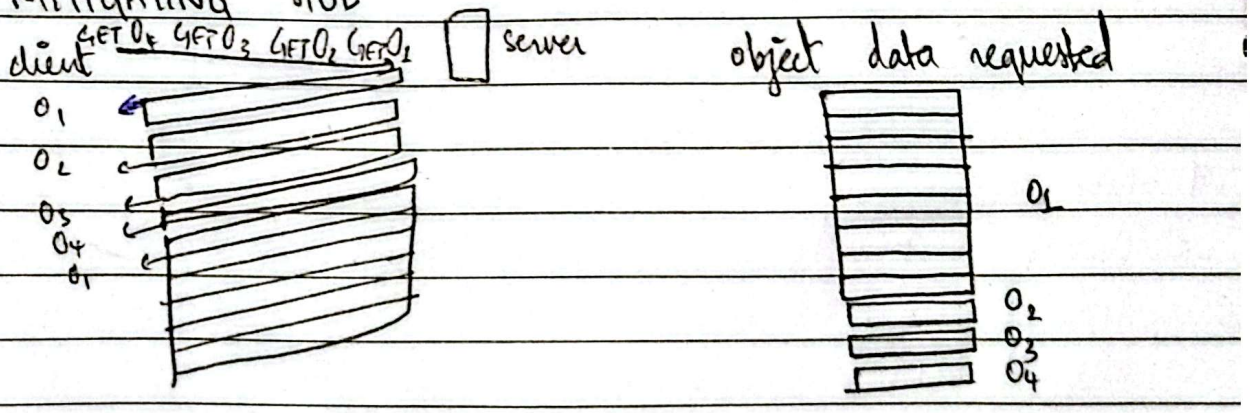
HTTP/2 IMPROVEMENTS:-

- **Multiplexing:-** Multiple HTTP requests and responses can be sent concurrently over a single TCP connection.
- Responses do not need to be in order, avoiding HOL blocking at the application.
- **Header compression:-** HTTP/2 compresses headers to reduce overhead in requests/responses.
- **Stream prioritization:-** Priority to streams can be assigned (e.g small objects can be sent first).
- **Reduced Latency:-** By multiplexing streams & allowing out-of-order responses, HTTP/2 decreases delay in multi-object request significantly.

HTTP 1.1



HTTP/2 : MITIGATING HOL

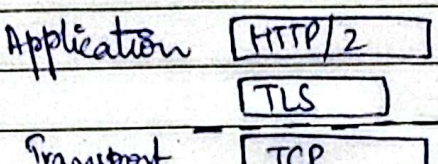


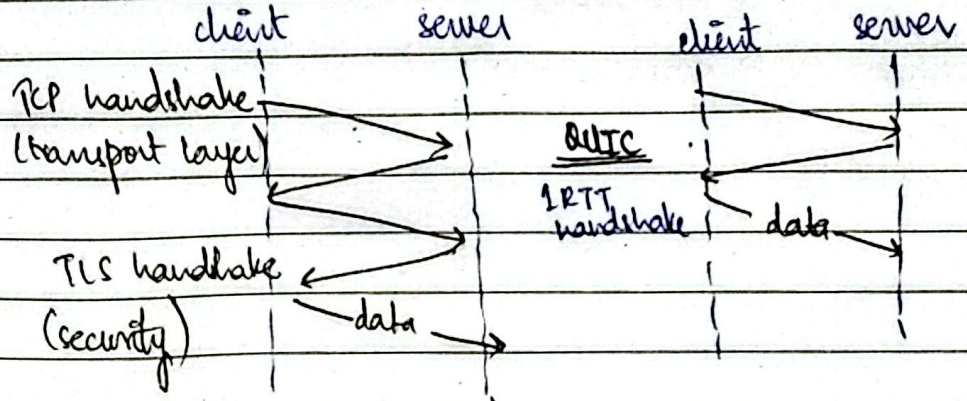
Dated:

HTTP/3 :- adds security, per object error object & congestion control (more pipelining over UDP)

QUIC : QUICK UDP INTERENET CONNECTIONS :-

o- Application layer protocol
↳ increase performance of HTTP





TCP (reliability, congestion control) + TLS (authencation, crypto state)
2 serial handshake

QUIC: reliability, congestion control, authentication, crypto state.
1 handshake
↓ ORTT handshake delay
↳ faster connection establishment
↳ runs on top of UDP
↳ Encryption
↳ lower latency compared to TCP.

Q: Object size = 850,000 bits, Average request rate : 16 req/s
internet = 3s

avg data rate = 850000 x 16 = 13600000

a). $\Delta = \frac{L}{R}$

$$\Delta = \frac{850{,}000}{15 \times 10^6} = 0.0567s$$

traffic intensity on the link, $= \Delta\beta = 16 \text{ req/s} \times 0.0567$

$= 0.907$

avg. access delay $= \frac{\Delta}{1-\Delta\beta} = \frac{0.0567}{1-0.907} = 0.6s$

The total average response time is $0.6 + 3 = \boxed{3.6s}$

b). 60% cache hits, 40% miss rate

new traffic intensity $= \frac{40}{100} \times (16 \times 0.0567) = 0.36288$

Average access delay $= \frac{0.0567}{1-0.36288} = 0.089s$

Total Average time for cache misses $= 3 + 0.089 = 3.089s$

Since there is 0.6 probability of cache hit → response time x 0s, & a 0.4 probability of utilizing a 3.089 response time

Dated:

Total average response time = 0.6(0) + 0.4(3089)

= 1.24s

Thus Average response time is reduced from 3.61s to 1.24s when using the cache option.