day / date:

August 21, 2023
Lecture 1

→ Application layer 1) presentation 2) session 3) application
→ Transport layer
→ Network layer } routing
→ Data Link layer
→ Physical layer

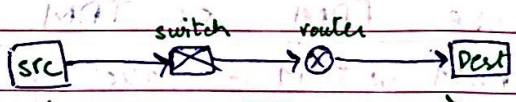
Switching devices of Datalink Layer
- Hub
- Bridge
- Switch

Hardware:
- End host (source n dest.)
- Switches / Host

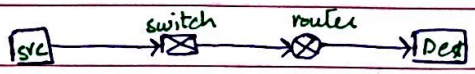
Software:
- Protocols of above 5 layers

Application Layer Protocols:
- HTTP
- FTP
- SMTP

Transport Layer Protocols:
- TCP (connection oriented)
- UDP (connectionless protocol)



message reaches dest at any cost
TCP is used in emails



complete message may not reach dest
UDP is used in vedio streaming

Network Layer Protocols:
- IP

Data Link Layer Protocols:
- Ethernet
- ARP

day / date:

Wednesday, August 23, 2023
Lecture 2

Endhost :- devices which are used by users.

Mobile Phones	Stalelitte Phones
analog signals	radio signals
call specific numbers which charge accordingly	specific extensions available without charges

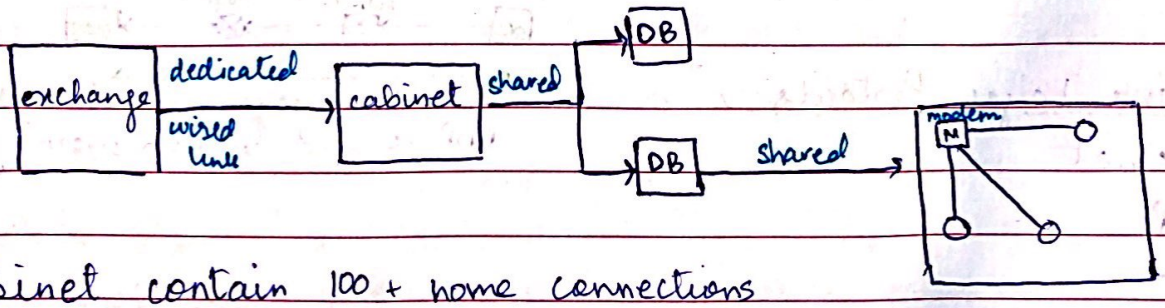
Physical
Hub
Datalink
Bridge/Switch
Network
Router

- 802.11 a RFC document
  802.11 n available for 5G

Edge router :- which connects one network to another
↳ edge / internet service gateway / gateway

Cable Based Network
use FDM or TDM
FDM vs TDM :- have diff bandwidth

FDM :- frequency distributed multiplexing
TDM :- time distributed multiplexing



cabinet contain 100 + home connections
DB contain 10 home connections

DSL :-

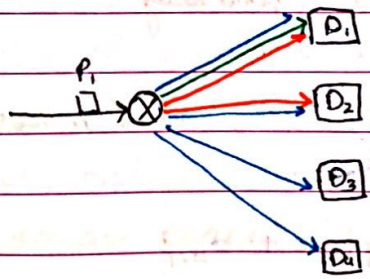
day / date:

Monday, August 28, 2023
Lecture 3

unicaste:
If a packet is forwarded by the switching device to a single digital device

multicaste:
If a switching device forwards a packet to more then one digital device

broadcaste:
If a switching device forwards a packet to all devices in the range to that switching device but is upto the digital device to accept or reject the packet



- unicaste
- multicaste
- broadcaste

Transmission delay
the time a packet takes to completley push onto the link

radio range
88 – 108 MHz

one hop numerical example

$$\frac{L}{R} = \frac{10 \times 10^3 \text{ bits}}{100 \times 10^6 \text{ bps}}$$

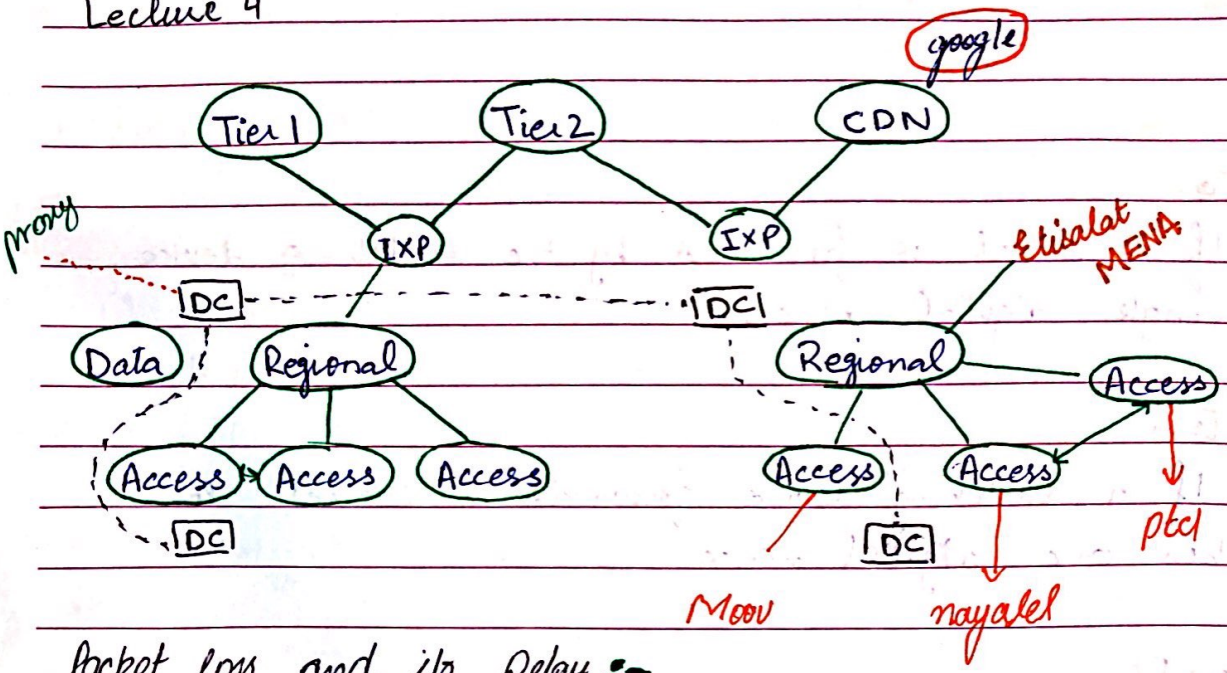
$$= 0.1 \times 10^{-3} \text{ s}$$

$$= 0.1 \text{ ms}$$ Transmission delay

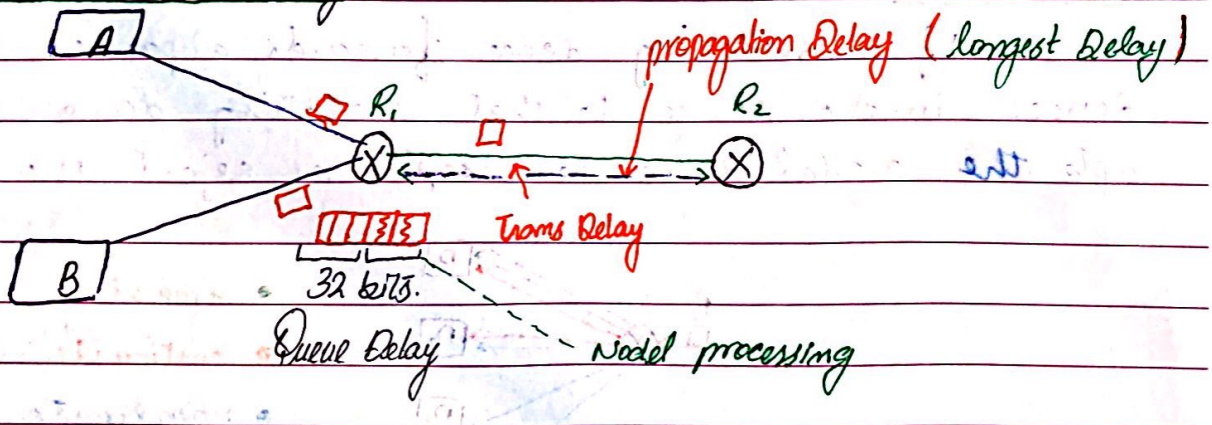
day / date:

Wednesday, August 30, 2023

Lecture 4



Packet loss and its Delay :-



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

Packet Queuing Delay :

$\frac{L \cdot a}{R}$: arrival rate of bits / service rate of bits

day / date:

CHAPTER 1

Nauman Moazzam

Lecture 1.1

Network Edge :- comprises of end systems / hosts

End Systems :- either request services or provide services
client server

- server is not part of the network core

Server

- provide access to network resources
- always present on host

e.g I request some service on google. Here google is the host from which we are (client) want some kind of service. Here the server is present at google which would provide the service which was requested by the client

- server have permanent IP address

↳ for websites or services which are accessed 24/7 around the world if the IP address is dynamic then routing table would have to be updated constantly.

* static IP address makes it easier to find you via DNS.

DNS :- Turns domain names into IP address

Client

- may request services
- have dynamic IP addresses

whenever a device is connected to the internet an IP is assigned to that device. When we shut down the device the IP is returned back to the pool of addresses & a new IP is assigned when connected again.

- do not communicate with each other

day / date:

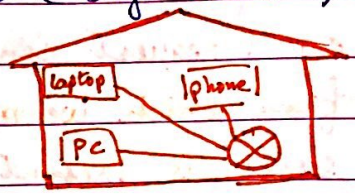
Access Network & Physical Media

↳ how host connect to any network or internet
connect end systems to edge router / first router

- Edge Router :- last internally managed routing device that connects one network to another
- First Router :- the first device which is connected to the device or server in a network
  - local ISP router

telephony or video conferencing apps are mostly peer-to-peer

Access Network: is a network which physically connects and end system to the immediate router (edge router)
e.g in a home network where internet is connected by stromfiber, the edge router is not present in the house, the edge router is present at the office of stromfiber & the edge router is connected to the devices via fiber cables



Transmission Modes

- unicaste :- one to one communication
- multicaste :- one to a selected group communication
- broadcaste :- one to all members of the group communication

Access Network Technologies

1. DSL (digital subscriber line)
   - use the telephone wires to transmit voice & data signal
   - the data over DSL line goes to Internet
   - & voice over DSL line goes to telephone net
   - using splitter
   - DSLAM (digital subscriber line access multiplexer)
     multiplex & demultiplex signals through DSL
     internet signals are forwarded to ISP & voice signal

dedicated access network

day / date:

2. Cable Modem / Network

- similar to DSL
- same network is used to provide data signals & TV signals
- splitter is used at home to split the signals accordingly
- & at the cable network end CMTS is used to demultiplex the signal to cable network or ISP
- the difference b/w DSL and Cable network is that DSL is a dedicated access network & Cable is a shared access network

shared access network

shared network is a broadcaste medium

3. Ethernet (enterprise access network)

- commonly installed wired LAN technology
- used to connect devices together on a Network to transfer broadband data.
- upto 10 Gbps transmission rates

4. Wireless Access Network

- Shared
- connects end systems to router via access points

5. Physical Media

Guided Media	Unguided Media
signals propogate in solid media	signals propogate freely
e.g copper, fiber, coax	e.g radio

day / date:

Nauman Moazzam
Lecture 1.2

Physical Media

- coaxial cable
- fiber optic
- radio

Approaches to Sharing

Reservation

- we reserve a link for permanent use through physical medium

circuit switching
e.g landline

On demand

- whenever we need to send / recieve a link is created & after delivery link is returned back

packet switching
e.g Internet

Packet Switching

- data is broken in packets
- & packets are created on demand
- If I need to send some data only then I will use the link otherwise link will be free from my side
- it is not neccesary for each packet to take the same route to reach it's destination
- each packet is considered as individual data
- each packet while have it's own src address & dest address
- then the network edge device will reassamble the data & then used

→ ~~addtni~~ admission control : per packet

Packet Transmission Delay = $\frac{L \text{ bits}}{R \text{ bits/sec}}$ no. of bits of packet - length

data network ⇒ routers
telephone network ⇒ switches

day / date:

Circuit Switching

circuit : connection

- when two device want to communicate a circuit is established
- now even if no data is send or recieved the circuit is reserved by the device / user

e.g If I call someone, a circuit is established b/w the two devices. Now if we talk for 5 mins & for the next 15 mins both muted their devices & noone talkes. No data is transmitted in those 15 mins but a circuit is reserved even if no data is transmitted since the call is not terminated.

- since the circuit is reserved b/w devices, the resources are dedicated to them & no other device can share these resources.
- doesn't route around trouble

FDM vs TDM

FDM: Frequency division multiplexing
The frequency (bandwidth) is divided equally among the users showing a link & each user use that allocated line to share data. If a user is not sending data then at that time the frequency is being wasted.
The allocated frequency is not shared.

TDM: Time division multiplexing
Instead of frequency, time is divided in among users If there are 4 users then each gets 15 secs of 1 min ~~until~~ in a cycle until time is completed. In that allocated time the user will use full frequency to send data. If there is no data for the user then that time cannot be used by other users.

day / date:

Numerical

How long does it take to send a file of 80 Kbytes from host A to host B over a circuit-switched network? All links are 1.536 Mbps. Each link uses TDM with 24 slots/sec. Time to establish end-to-end circuit is 500 msec

80 Kbytes = 640,000 bits
1 Kbyte = 8000 bits
1 Kbyte = 1024 bytes

Link transmits 1.536 Mbits per sec
There are 24 slots per sec

Each circuit has $\frac{1.536 \times 10^6}{24}$ = 64000 bps (one slot has)

Total data : 640,000 bits
It takes : $\frac{640{,}000 \text{ bits}}{64000 \text{ bps}}$ = 10 sec (To transmit)
To establish link : 500 msec = 0.5 sec
1 msec = 0.001 sec

Total time : 10 sec + 0.5 sec
= 10.5 secs

day / date:

Packet Switching : Store and Forward

entire packet must arrive at router before it can be transmitted to the next link.

Packet Loss and Delay

- delay occurs when packet arrival rate to link exceeds output link capacity
- router temporily store / buffer packets in its queue
  - ↳ queuing
- If the queue fills up & packets keep arriving then the router drops those packets which leads to loss
  - → happens in case of congestion in links

4 types of Delays

- → queuing : first in first out
- → processing : router extracts the src & dest IP address & index the IP
- → transmission : either given or negligible in numerical
- → propagation : time taken from src to dest affected by speed of light

to get end-to-end delay of a packet we need to calculate all the delays & add them

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

$$d_{trans} = \frac{L}{R}$$

$$d_{prop} = \frac{d \text{ length of physical link}}{s \text{ propagation speed } (\sim 2 \times 10^8 \text{ m/s})}$$

Transmission delay is the amount of time required for the router to push out the packet. Propagation delay is the time it takes to travel from 1 router to the next.

day / date:

Traffic Intensity

$$\frac{La}{R}$$

L: packet length
a: arrival rate
R: link bandwidth

- if La/R ~ 0 avg queuing delay is small
- if La/R ~ 1 avg queuing delay is large
- if La/R → 1 more work is arriving than it can be processed.

Throughput

rate at which bits are transferred b/w sender & reciever

e.g while downloading a video a packet travels through diff links will diff bandwidths. Since each packet may take a diff route, each packet will take diff time to reach the dest. The avg time taken by these pkts is throughput.

- The throughput is generally the rate of the link which is minimum.

day / date:

CHAPTER 2

Nauman Moazzam
Lecture 2.1

Application Layer

- specifies the shared protocols & interfaces methods used by hosts in a communication network
- helps to identify communication partners & synchronise communication

ensures an application can effectively communicate with other applications on different computer systems & networks.

- application layer do not exist on core network devices e.g router & switches
meaning a developer doesn't have to write a code for routers / switch

core network device	end user device
network layer	application layer
data link layer	transport layer
physical layer	network layer
	data link layer
	physical layer

Application Architechure
- client - server
- peer - to - peer → self scalability

day / date:

Process Communicating

processes communicate within a host using inter-process communication
↳ set of rules defined by OS

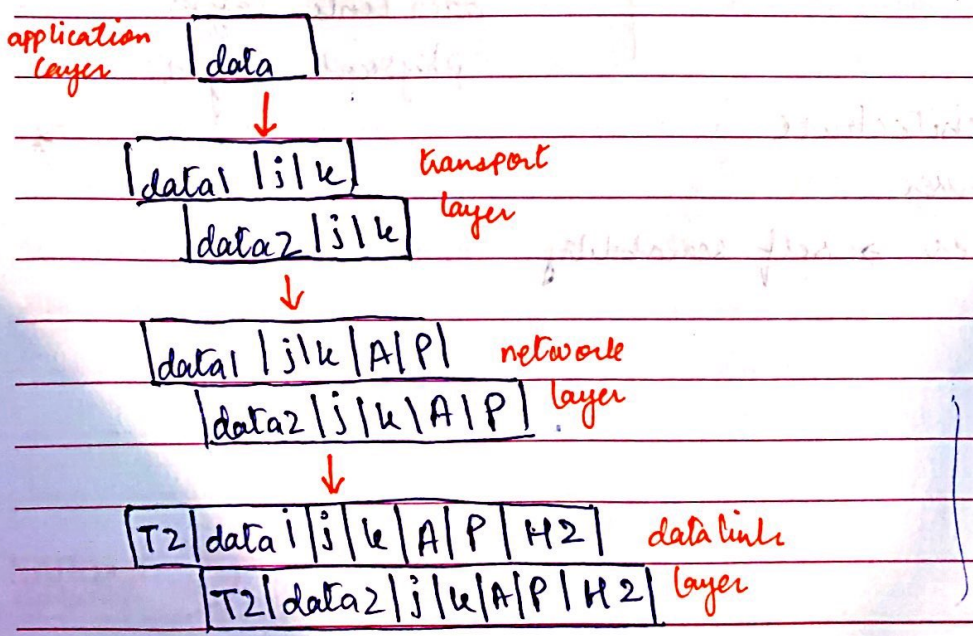
OS on servers in mostly Linux

process in different hosts communicate by exchanging messages

client process: process that initiate communication
server process: process that awaits to be contacted

each process opens a socket in the transport layer & send its data in that socket

socket: communication link b/w 2 processes running on the network
combination of port number & IP address



day / date:

- host A send data to application layer
- application layer breaks the data into packets & send it to transport layer through sockets
- transport layer attach [port address] j & k to the data packets
  - j :- address of sending process
  - k :- address of recieving process
- then send it to network layer
- network layer append the source & destination [IP address] to the data packets
- then send to data link layer which attach the header & trailer to the packets
- then the physical layer converts the data into bits & send to the internet

- port number helps the end host to identify which process is the correct process to send the data

Transport Services

- data integrity
  - e.g apps for file transfer require 100% reliable data transfer
- timing
  - e.g internet, telephony require low delay to be "effective"
- throughput
  - e.g multimedia
- security
  - e.g bank transfer

day / date:

Transfer Layer Protocols

- TCP
- UDP

UDP exists because some apps still do not require so many services which TCP provide. UDP is a light-weight protocol which do not require a large header

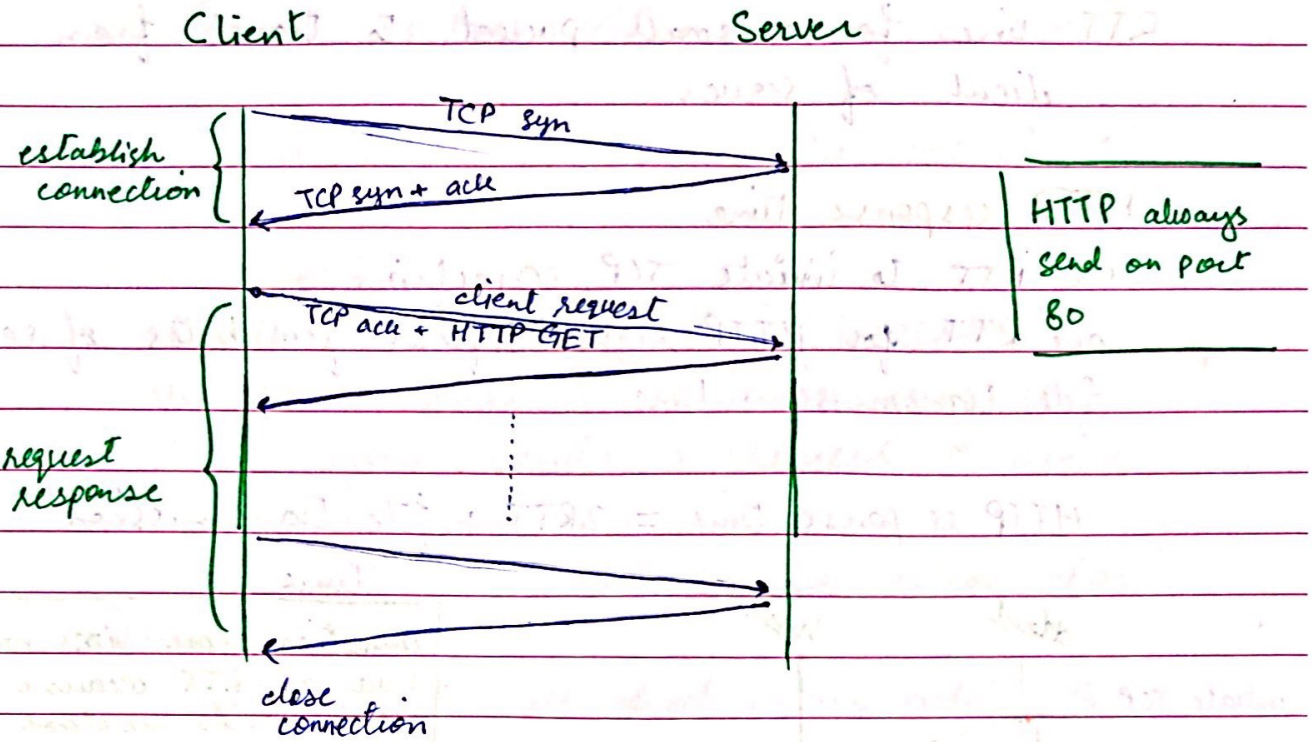
Lecture 2.2

- differentiating aspect of Internet from other competitor networks was on-demand

HTTP

- web's application layer protocol
- when we enter somekind of URL in the browser, it send a message to the server known as HTTP request
- This request is recieved on port number 80
- the server in return sends objects
- HTTP uses TCP protocol
- HTTP is stateless meaning that the server maintains no prior information about the client searches if information needs to be saved the cookies is used

day / date:



Non-Persistence HTTP

at most only 1 object is sent over a TCP connection

- TCP initiate

→ client initiate TCP connection on port 80
→ server at host accept connection on port 80 & notify client
→ client send request message into socket
→ server forms response message to the request & send the message into its socket
→ client recieves response message
server closes TCP connection

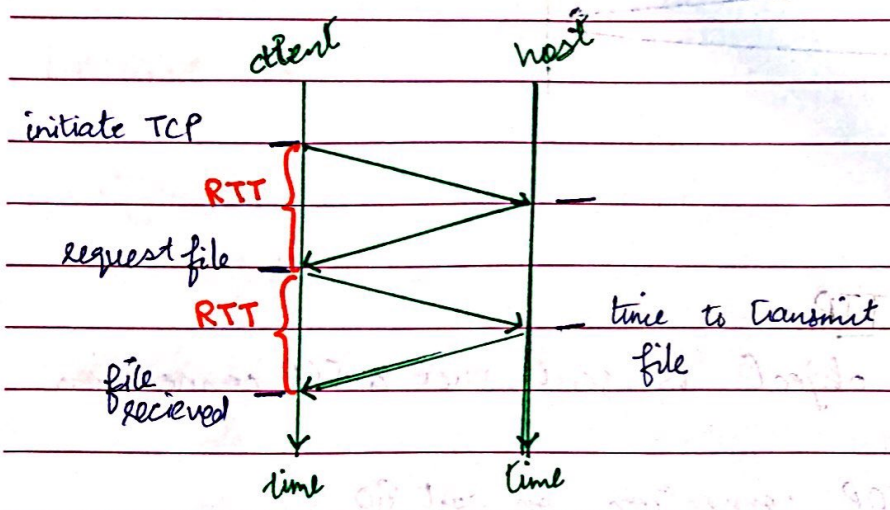
If a message request contain 10 jpeg images then theses steps will be repeated for all 10 images

day / date:

RTT:- time for a small packet to travel from client of server

HTTP response time :-
one RTT to initiate TCP connection
one RTT for HTTP request & first few bytes of response
file transmission time

HTTP response time = 2RTT + file transmission time



connection close does not have an RTT because server sends the close msg along with the last bits

Persistent HTTP

- server sends the base HTML file to client but keeps the connection open
- if a request message contain 10 more refrence objects server will use the same TCP connection to send those files/objects
- all this takes as little as 1 RTT

day / date:

persistant pipeline often

persistant HTTP often use the concept of pipeline

Non-pipeline:-
clients sends 1 request & in response the server sends 1 response message.
client again sends a request & client sends a response.
this continues untill the connection closes.

Pipline:-
client requests all objects in one batch & the server will recieve all the requests and then send the response objects

HTTP request message format

- first line is request line

<method> <URL> <version> <cr> <lf>

method include: POST, GET, PUT, HEAD
version of HTTP: 1.0, 1.0, protocol version
cr: carraige return \r
lf: line-feed \n

- the next couple of lines are header lines
- a line with only cr & lf indicate the end of header lines

day / date:

Form Input

when we type something in the webpage

→ POST method :

Some webpages include some kind of form e.g while shopping online, that input is acquired by the server using POST method which is optional in the body of request message

→ GET method :

when we type something in the search bar what we type is appended in the URL

Method Types

HTTP / 1.0	HTTP / 1.1
GET	GET
POST	POST
HEAD :- often used for debugging, do not return any object	HEAD
	PUT :- client upload a file in server
	DELETE :- client delete a file in server

day / date:

HTTP response message

→ first line contain the status line
<method> / <version> <status code> <cr> <lf>
some examples of status code include
(2xx) 200 OK : request succeeded
(3xx) 301 Moved Permanently : object moved
(4xx) 400 Bad Request : message not understood by server.
(4xx) 404 Not Found : requested doc not found on server
(5xx) 505 HTTP Version Not Supported
→ next couple of lines are header lines
DATE :- when the client made the request to server
Last-modified:- when the requested object was updated
→ at the end is the data objects which were requested

Cookies :-

- used to maintain the state of stateless HTTP
- client stores small state on behalf of the server
- when accept cookies our state is saved in a small database on the server coressponding to the id assigned to us by cookies
when we again request the same HTTP the cookies id is sent to the server along with our request message

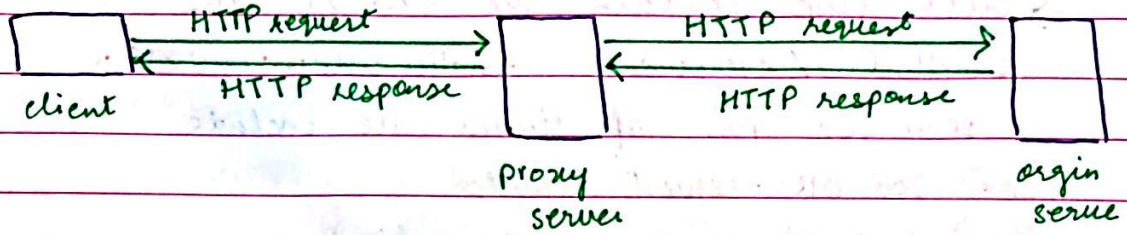
e.g in daraz, we first accept cookies, our information is saved in the database. After sometime if again add something to the card & checkout our information is saved because of the cookies.

cookies is maintained it is deleted
or the cookies expire

day / date:

Web Caches

complint HTTP protocol to reduce traffic



- whenever a client performs a request if the objects are stored in the cache server.
  if another client performs the same request then the proxy server do not forward the request to the origin server, but it returns the objects requested by the second client from as they are stored in the cache of proxy server
- In this way the traffic is reduced.
- proxy server acts as both server & client
- typically cache is installed by ISP

Example

avg object size : 100k bits
avg request rate from browser to origin server : 15
avg data rate to browser : 1.50 Mbs
internet delay RTT from institutional router to origin server : 2sec
access link rate : 1.54 Mbps

LAN utilization : 1.50Mbps / 10 Mbps = 15 %.

or

La/R = (100k bits * 15) / 10 Mbps = 0.15 (negligible)

(negligible)

day / date:

Access link utilization : 1.50 Mbps / 1.54 Mbps = 97%.

or

La/R = (100 kbits * 15) / 1.54 Mbps = 0.97 (Huge)

Total delay = internet delay + access delay + LAN delay

= 2 sec + minutes + μ sec

97% utilization is equal to 0.97 traffic intensity

Now suppose that,

local web cache is installed

~~char~~ cache hit rate is 0.4

40% requests satisfied at cache

60% requests satisfied at origin

Access Link utilization: (0.6 * 1.50) / 1.54 = 0.58 ≈ 58%.

Total delay = internet delay + access delay + LAN delay

= 0.6 (2 + msec + ~~LAN~~ μsec) + 0.4 (μ sec)

= 0.6 (2.01) + 0.4 (~msec)

= ~ 1.2 sec

Lecture 2.4

email is asychronous

Electronic Mail

- user agents
  - present at user end systems
- mail servers
  - contain dedicated mailbox & message queue
- SMTP (app layer protocol)
  - simple mail transfer protocol, push protocol

day / date:

- the sending user agent push the mail in the outgoing queue of mail server
- mail server use SMTP to push the mail to the user mail box to the recieving user agents mail serve
- hence SMTP is a push server
- now the mail is in the mail box of recievers mail server
- the reciever pulls the mail from the mail box at their convenience, this pull protocol MAP

pull protocol means that if the reciever do not have internet connection & the recievers mail box got a mail then, when the reciever will get internet connection then the mail will be downloaded, means that it will be pulled at the recievers convenience.

SMTP : RFC

- uses TCP
  - ↳ error message is displayed
- sender & reciever maintain a direct link
- 3 phases of transfer
  - ↳ SMTP handshaking — initiate TCP
  - ↳ SMTP transfer of message — persistent TCP
  - ↳ SMTP closure

in-band :-

out-band :-

SMTP is out-band

7-bit ASCII response / command

day / date:

Comparison with HTTP

HTTP	SMTP
pull protocol	push protocol
ASCII command / response	ASCII command / response
object encapsulated in its own response mssg	multiple objects sent in multipart mssg

DNS

domain name system

- acts as an identifier for IP address
- user type a domain name in web browser; DNS server translate requests for names into IP addresses; control which server an end user will reach
- FTP invoke an IP address against a domain name
- a single web server may have multiple DNS servers, different DNS queries must be distributed among these servers

in the same way a web services have multiple web servers & every web server have an IP address. DNS distribute the load among multiple web servers

e.g millions of people access google.com at a time DNS distribute the load by assigning IP address of web servers to those DNS queries

- some web services have a canonical name & an alias name; we type the alias name & DNS converts it to its canonical name e.g CNN.com

day / date:

DNS servers are in 3 layers:

- root DNS server layer
- TLD Top level DNS server
- authoritative server layer (server of an particular organisation)

13 DNS root servers

- when we type something in browser
- DNS service translate domain name to IP address
- HTTP protocol invoke DNS protocol
- DNS protocol sends the DNS query to Root DNS server
- Root DNS transfer the query to a correct TLD server
  e.g we type FAST.edu in browser the root DNS will send the query to edu DNS server at TLD layer
  Root DNS will transfer the query corresponding to the Domain name
- Each domain name have a TLD server layer
- TLD server will then send the query to correct organisation server

- Local DNS server
  works similarly to proxy ~~server~~ web cache
  when a DNS query goes to DNS server from an organisation the response is stored against the domain

Iterated Query

in response to a DNS query, it provides refrence to another DNS server

day / date:

Recursive Query

when one server communicates with several other DNS server to hunt down an IP address & return it to the client.

Caching DNS Information

DNS records

(name, value, type, ttl)

- canonical name of mail server
- host name :- canonical name / real name of server
- domain name :- alias name
- host name of authoritative name serve

type = A : authoritative DNS
type = NS : TLD server
type = CNAME mapping b/w alias & canonical name
type = MX similar to CNAME but for mail server

day / date:

CHAPTER 3

Nauman Moazzam

Lecture 3.1

- Transport layer provides services to the application layer
- provide logical communication b/w (apps) in diff devices
- run in end systems — 2 processes
- when data packets are in transport layer they are called segments
  - ↳ data packet + transport layer header
    - header ↓ Port addresses

Transport layer :- process to process delivery
Network layer :- responsible for node to node delivery
Link layer :- responsible for delivery b/w neighbouring nodes
e.g host to first router

a single node contain many processes
e.g server & client are using html app so the process responsible for that html app ; communication b/w those processes is ~~resp~~ done by transport layer

Functions of Transport Layer

- provide process to process communication
  - ↳ accomplished by port number
- provide end-to-end error checking (both TCP & UCP), error control (only TCP) & and flow & congestion control (only TCP)

TCP is a reliable protocol because it provides error control and flow & congestion control whereas as UDP is unreliable since it doesn't provide these

~~TCP~~ TCP & UDP both tell if error occurs but only TCP tells how to resolve the error

day / date:

Multiplexing and Demultiplexing

- multiplexing takes place on senders side
  demultiplexing takes place on recievers side
  → on port number in transport layer

Port

- Transport layer identifier
- to choose among the multiple processes running on same host
- is a 16 bit integer
  0 – 65,535

→ on a single client (laptop) there are multiple processes running like HTTP, spotify, skype, etc. Every process will have a port number which will help to identify which process send a message

IANA had divided port numbers into 3 ranges

- → Well known: 0 – 1023 ; assigned to processes e.g HTTP – 80
- → Registered: 1024 – 49,151 ; registered not assigned
- → Dynamic: 99152 – 65353 ; can be used by any process

Multiplexing :- gather data from multiple sockets ; evelop data with a header

Demultiplexing :- recieve segments ; deliver data to correct app layer processes by using info from header & identifing correct sockets ; remove header

day / date:

How Demultiplexing Works

- Suppose A is the sender & B is the reciever
- both have hike & WhatsApp installed
- A wants to send WhatsApp message & to B
- A must mention IP address of A & B & and the port number of WhatsApp
- If A wants to send message on hike as well
- A must mention IP address of & B and the port number of hike while sending the message
- The message from boths apps will be wrapped up along with appropriate headers

  header :- source IP address, dest IP address, source portno, dest port no
- a single message will be send to B
- This is called multiplexing

- At B, the recieved message will be unwrapped.
- messages from WhatsApp & hike are sent to appropriate apps by looking at dest port no.
- This is called demultiplexing

Connection Connectionless Demultiplexing

- used by UDP
- client select an epiheral port no randomly e.g 9157
- transport layer will attatch the source port no : 9157 & the dest port no (well known) : 6428 to the packet
- now at the server side transport layer will strip the header & from the dest portno: 6428 send the packet through correct socket to the appropriate app

day / date:

- UDP socket is identified by 2-tuple
  - dest IP address
  - dest port no.

- at host transport layer
  - UDP segment arrives
  - checks ~~port no~~ dest port no
  - directs UDP segment to appropriate socket

- UDP voilates the layering principle
  - network layer headers include source & dest IP address
  - transport layer uses the information of network layer

  several voilations exists.

- in case of same dest port no & diff source port no. the data packets will be delivered to the same socket

Connection-oriented Demultiplexing

used by TCP

TCP socket is ~~identified~~ identified by 4-tuple
- source IP address
- source port no
- dest IP addres
- dest port no.

- now in case of same dest port no, transport layer can perform segregation on the basis of source IP address & source port no.
- every socket will create a diff socket in server

day / date:

User Datagram Protocol UDP

TCP is connection oriented so it have extra cost & extra delay

barebone protocol

connectionless

↳ no handshaking

apps which require no delay but can afford unreliable data transfer prefer UDP

↳ e.g streaming multimedia, DNS,

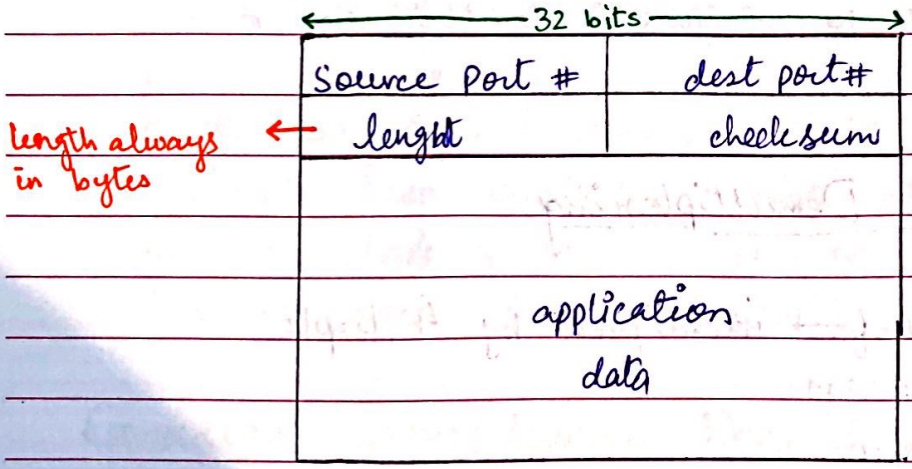
even if we use UDP does not mean that reliability can not provided

↳ we can add reliability but with in the app

each UDP segment is handled independantly

↳ UDP does not ensure & ordered delivery

UDP segment



length always in bytes

header length = 4 × 16 bits = 64 bits = 8 bytes

length :- total length of segment including header & data

data length = length − header length

checksum used for error detection

day / date:

UDP checksum

each segment is treated as a sum of 16 bit words

sender :- add all the words & take it's one's complement & store in the checksum

reciever :- takes one's complement of the data recieved & match with the checksum value

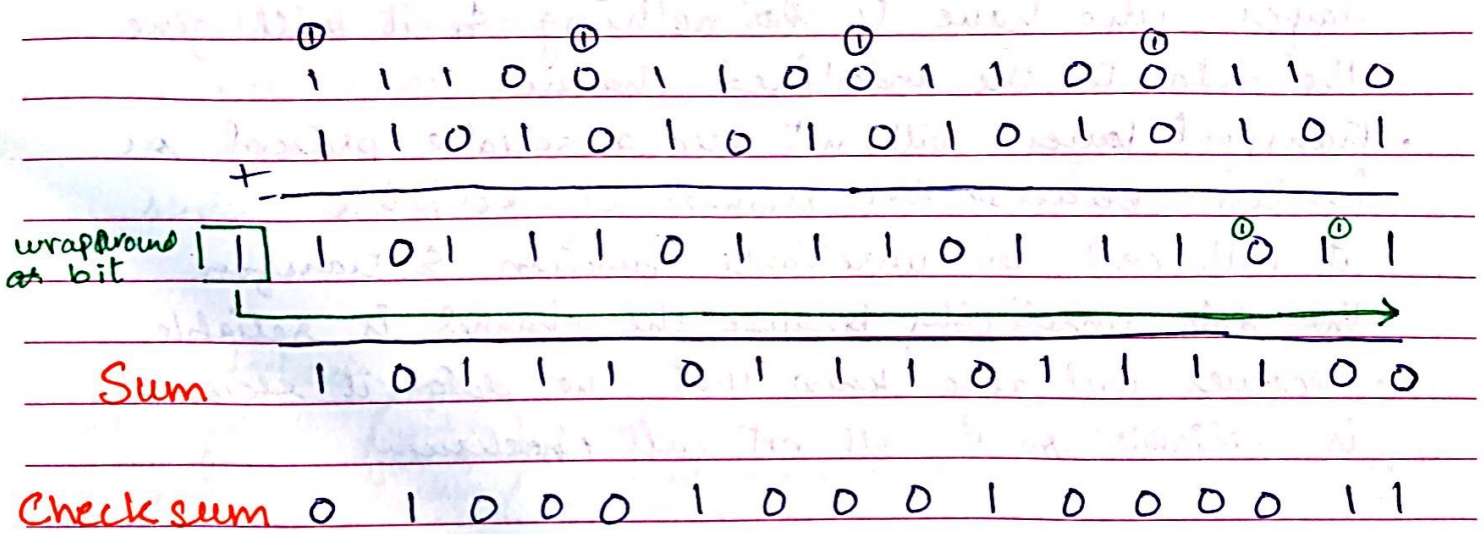
if matched == no error

else error occurred.

Example

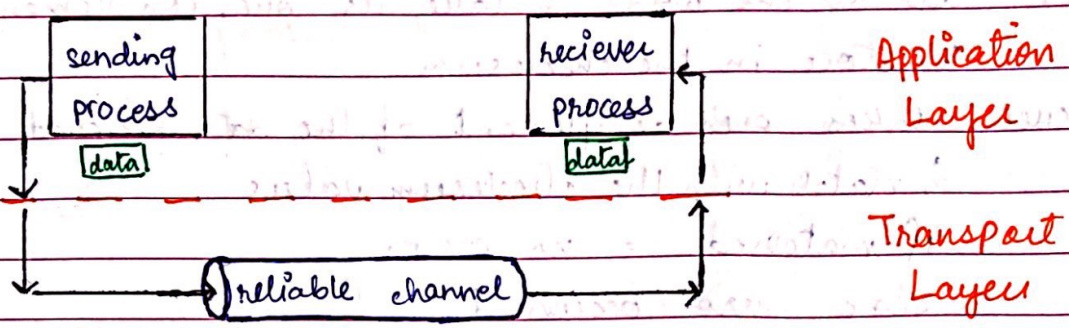
data of 32 bits

2 16 bit words

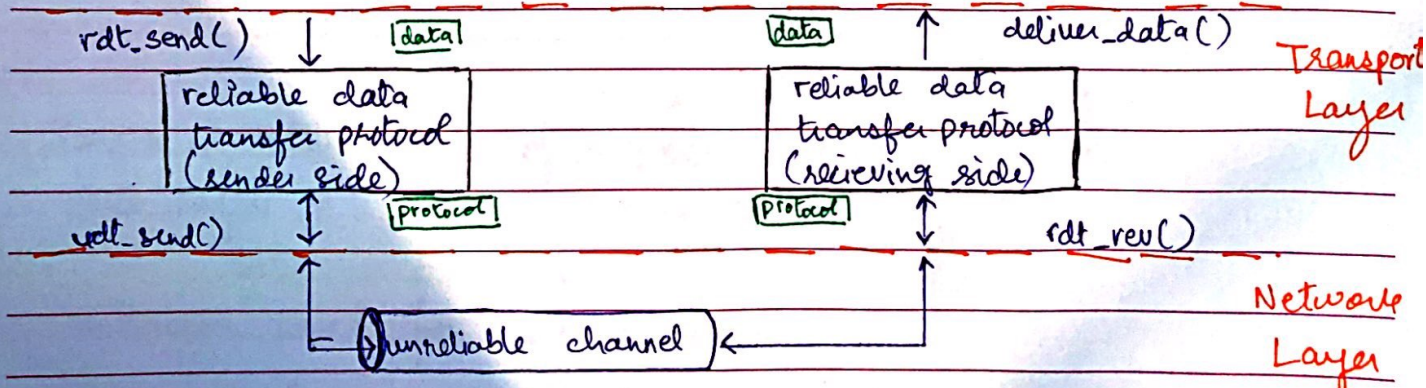


day / date:

Principles of Reliable Data Transfer



- in an ideal world the data will always be reliable there would be no data loss or corruption
- sending process will give the data to the transport layer who have to do nothing & it will give the data to the underlined channel
- transport layer will not need a reliable protocol or function because the channel is reliable.
- it will call an unreliable function & transfer the data unreliably because the channel is reliable
- reciever will also know that the data it recieved is reliable so it will not call checksum.



10 sec : 15
1 : 15/10 = 1.5

day / date: 5263 : ?

- application layer send ddata
- it calls a reliable data transfer function in transpor layer
- transport layer also call a function in network layer which is unreliable because network layer have unreliable channel
- reliability is ensured at transport layer
- at the reciever side, transport layer recieves the data through a reliable function because it has to ensure that data is sent to application layer reliabily

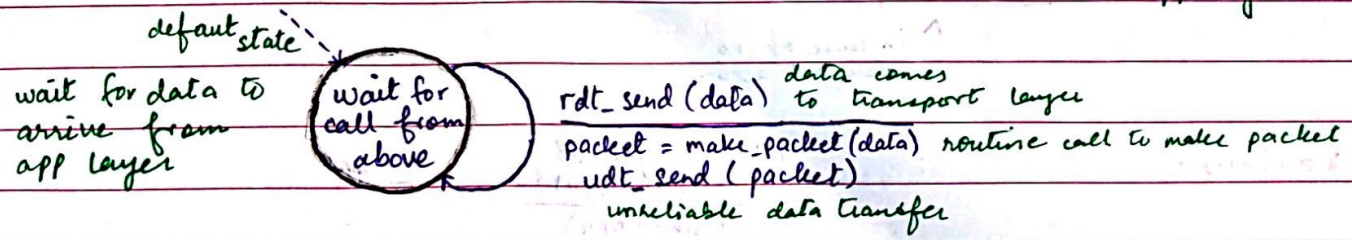
rdt 1.0

underlined channel is reliable

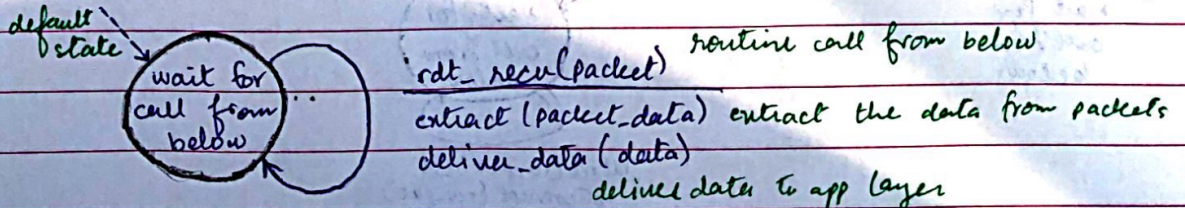
- no error
- no loss

sender & reciever have seperate FSM

above : app layer



^ Sender



^ Reciever

sender or reciever don't change state because they know that data will be transfered reliably, hence don't wait for any signal / acknowledgment

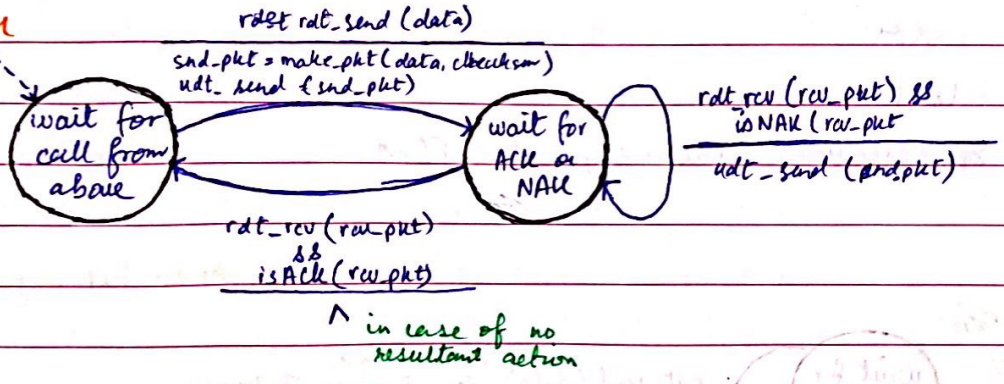
day / date:

rdt 2.0

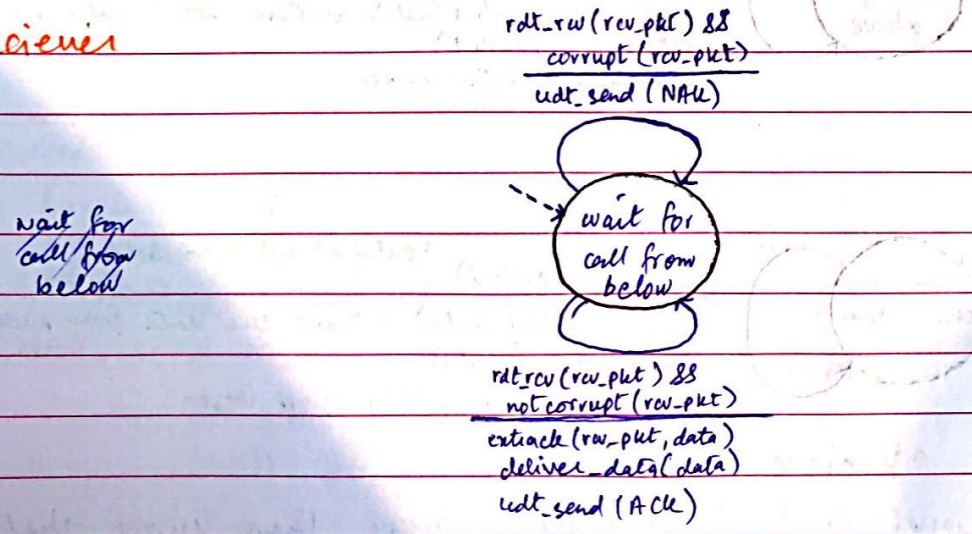
underlined channel is unreliable
- ↳ bit error

3 new functionalities to recover from error
- ↳ error detection
  - ↳ checksum
- ↳ feedback
  - ↳ ACK : positive acknowledgment
  - ↳ NAK : negative acknowledgment
- ↳ retransmission
  - ↳ packet in error is retransmitted by sender

Sender



Reciever



day / date:

incase of no error
at sender data arrives from above
packets are formed & checksum is attached
sends data & change state

at reciever data arrives from below
check the checksum ; data not corrupted
data/packets are extracted
send ACK to sender

at sender , waiting for ACK or NAK from reciever
accepts ACK
changes state to default

incase of corrupted data
at sender side data arrives
packets are formed ; checksum is attached
sends data & change state

at reciever side data arrives
check the checksum ; data is corrupted
send NAK to sender

at sender , waiting for ACK or NAK
accepts NAK
sends packet again

at reciever data arrives : check the checksum
if not corrupted , extract the data & sends ACK to sender

day / date:

Flaw in rdt 2.0

- ACK or NAK are bits
- they can also be corrupted
- if reciever sends ACK and on the way it is corrupted & sender recieves NAK and resends the packet
- reciever will get duplicated data
- to solve duplications sender and sequence number
- in this way reciever can identify duplicate packets and discard them

rdt 2.1

the flaw in 2.0 is solved in rdt 2.1
add sequence number

- at sender, waits for data at seq#0
- data arrives and made into packets
  sndpkt = make_pkt(0, data, checksum)
- send data & change state
- if sender recieves NAK0 (data at reciever was corrupted)
- resend the 0 packet again
- if sender recieves ACK0 from reciever
  packet reached reciever without error
- changes state for 1 (next data with seq#1)
- if reciever sends NAK & becomes corrupted & sender gets ACK ; sender recieves ACK 1
- sender was waiting for ACK0 but recieved ACK1
  hence it knows that ACK was corrupted
- retransmits the packet

Why sequence # 0 & 1 only?

day / date:

rdt 2.2

- Same functionality as 2.1
- remove NAK
- when reciever needs to send negative ACK, reciever will send ACK of the previous sequence no.

reciever is in wait state 1 but recieved corrupt data for seq # 1 then reciever will continue to send ACK 0 untill correct data for 1 is recieved

- when sends gets ACK 0 when it sends the packet with seq# 1 sender will know that reciever is not getting correct data for seq#1

rdt 3.0

- underlying channel with errors and loss
- in this case checksums, seq#, ACK, retransmission will be of no help
- in this case when sender sends a packet it will start a timer, if sender recieves any ACK - then well n good but if sender does not recieve any ACK before the timer expires it will send the packet again

- rdt 3.0 gives poor utilization
  stop n wait protocol
- limiting the use of resources because the sender can not send another packet untill it recieves the acknowledgement of of previous packet

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

day / date:

Go - back - N

- window size = N
  consecutive transmitted, unACKed packets
- reciever sends cumulative ACK (ACK of last correctively recieved packet)
- when sender recieves ACK(n) window is forwarded to n+1
- sender has a timer for oldest unACKed packet
  - when timer of a packet expires resend all unACKed packets

- seq # range depends on sequence number field in header.
  bits allocated to the sequence # in header is the limit for the range of seq #
  $0 , 2^k - 1$
  k :- no of bits allocated for the sequence # in header

e.g if $k = 3$

$2^k - 1 = 2^3 - 1$

$= 7$

0 - 7 in binary

the window size would also be limited by this

0 - 7

$N = 8$

- if N = 1 then Go-back-N would become stop n wait since pipeline main ēik hi packet bhej sakta

- base : sequence # of oldest inflight packet
  inflight packet : sent but not acked
- nextseqnum : next available seq # which can be sent in pipeline

day / date:

- There are 4 ranges
  - 0 — (base-1) already ACKed
  - base — (nextseqnum-1) sent but not ACKed ; inflight
  - nextseqnum — (base+N-1) usable but not sent
  - >= (base+N) not usable

Selective Repeat

reciever sends individual ACK of each packet
sender maintains a timer for each packet
reciever maintains a buffer of size N
↳ window size

Dilemma

- suppose that there are seq # 0, 1, 2, 3
- and window size N is 3
- sender send pkt 0, 1, 2
- recievers recieves them & send acknowledgment ack 0, ack 1, ack 2 ; & move the buffer along
- but the ACK's are lost before reaching the sender.
- timer for packet 0 expires on sender side & it retransmitt pkt 0
- at reciever side the buffer moved to 3, 0, 1
- since it does not know that ACK's were lost
- when pkt 0 reaches reciever, it does not know that it retransmission or a new packet

Solution :- reduce window size
ideally window size should be less than or equal to half the sequence number space

N <= sequence number/2

day / date:

	Stop n Wait	Go Back N	Selective Repeat
sender window size	1	N	N
reciever window size	1	1	N
min seq number	2	N+1	2N
efficiency	$1/(1+2a)$	$N/(1+2a)$	$N/(1+2a)$
type of Ack	individual	cumulative	individual
order at recieving side	–	inorder only	out of order as well
num of retransmission	1	N	1

a = ratio of D_{prop} and D_{Trans}

- as Go Back N ack the pkt cumulatively, it rejects out of order pkts
- as Selective Repeat supports supports recieving out of order packets
  - ↳ sorts the window after recieving packets
  - it uses independent acks

day / date:

Transmission Control Protocol TCP

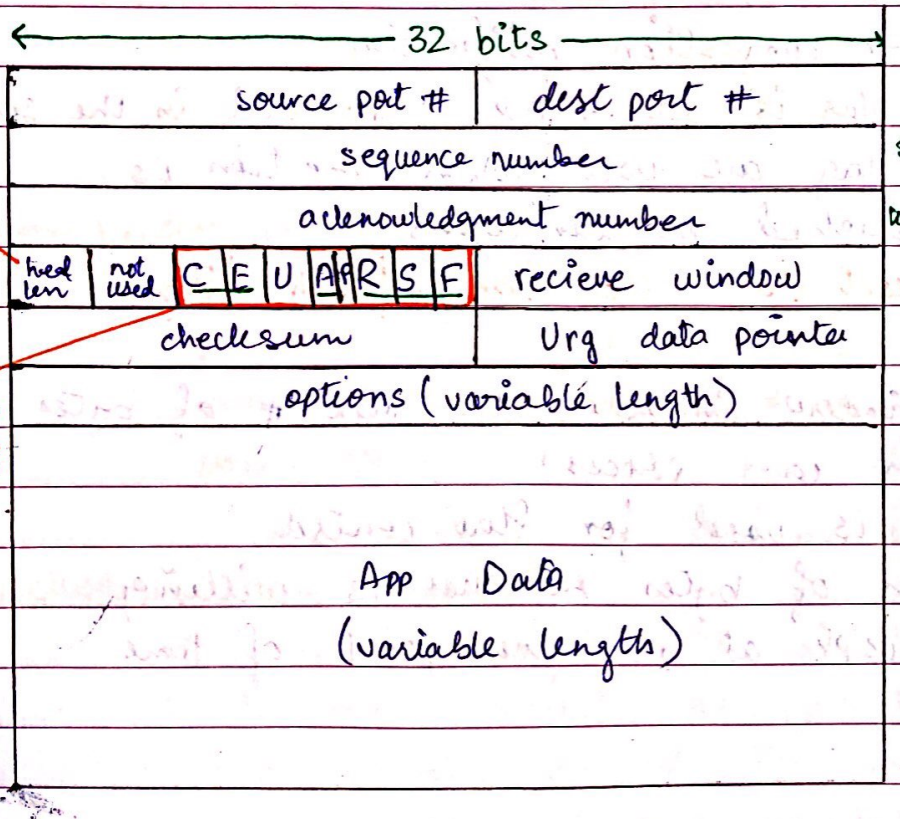
- connection oriented protocol
- inorder Transmission
- full duplex
  - ↳ bi-directional flow of data
- multicasting is not possible
  - ↳ one sender, one reciever
- data flows in byte stream

MSS :- max amt of data (payload) which ca be send in the segmen

TCP header size can be variable

not overload reciever flow control & congestion control are diff not overload the channel/ links

TCP Segment



sequence # of segment

kis seq ka kis seq # ka ACK

6 flags of 1 bit

- no. of bits available for seq # is the range of seq in this case available bits = 32 $(0, 2^{32}-1)$ range of seq # eik pipeline $2^{32}-1$ segments ja saketay
- seq # is basically byte num, data stream buffer se byte grab karti aur segment mein dal deti

day / date:

- header length is variable because the field of options in the segment is optional
there are diff TCP options which when used will vary the TCP header length
if this is not used then the header length is standard i.e 20 bytes

Data length = Total segment size - header length

- There are 6 flags in TCP segment of 1 bit
  - C, E → congestion notification
  - if A flag is set → ACK is included in the segment
  - &RSF flag are used when conection is established or tear down
    reset flag, sync flag, finish flag

- recieve window includes the max # of bytes a reciever can process
  ↳this is used for flow control
  number of bytes reciever is willing to accept at any given point of time

day / date:

Sequence Number

- TCP views data as byte stream
- TCP app layer se data aa raha hai sending side pr aur sender buffer main ata ja raha
- TCP sender grabs chunks of data from the buffer & make packets / segments
- Seq # have to be A-one in order for TCP to assure inorder delivery as network doesn't care about the order
- sequence number is the first byte in the first chunk of data which TCP grabbed

500k bytes data in buffer

MSS 1000 bytes

no. of segments = $\frac{500k}{1000}$ = 500 segments

assuming, each segment size is 1000 as

seq # of 1st segment 0

0 - 999 :- 1st segment seq # field / range

1000 - 1999 :- 2nd segment seq # range

2000 - 2999 :-

Acknowledgment Number

Ack for segment 0-999 will be 1000

- seq # of next expected byte from the sender
- this means that bytes till 999 have been correctly recieved and next expected byte is 1000
- TCP sends cummulative Ack
- sender send 3 segments 0-999, 1000-1999, 2000-2999 TCP will send the Ack for the last correctly recieved byte
- Ack in this case will be 3000

day / date:

TCP RTT, timeout

- if timeout value < RTT
  - ↳ timer expires before ACK reaches sender
    retransmission
- if timeout value > RTT
  - → agar ACK gets lost tou jab timer expire nahi hota retransmission nahi ho gi
  - → slow reaction to ACK / segment lost
- optimal value of RTT
  - when connection develops a sample RTT (variable) is stored
  - this variable is used to set the timer
  - during the connection, sample RTT is measured with some intervals & update the variable
  - timer is computed using the variable
  - SampleRTT is not measured for retransmissions.

$$EstimatedRTT = (1-\alpha) * EstimatedRTT + \alpha * sampleRTT$$

where, $\alpha = 0.125$ typically

depends on network designer

$$TimeoutInterval = EstimatedRTT + 4 * DevRTT$$

where,

$$DevRTT = (1-\beta) * DevRTT + \beta * |SampleRTT - EstimatedRTT|$$

where, $\beta = 0.25$ typically

day / date:

TCP ACK generated at reciever

1. inorder segments arrive at reciever
   reciever waits upto 500ms for the next expected inorder segment so it can send a cumulative ACK
2. inorder segment arrive, along with another unACKed during the 500ms wait another inorder segment arrives at the reciever, it will immediately send a cumulative ACK
3. out of order segment arrives, higher than expected seq #
   reciever expects seq# 6, but recieves seq# 8 this means that seq# 7 was lost so reciever will send the ACK for last correctly recieved segment i.e. 6
4. arrival of segment that fills the gap
   immediately sends the ACK
   reciever recieves seq# 7 which fills the gap & send ACK for 7
   sends new ACK only if gap is filled at the lower end

day / date:

Chapter # 4

Network Layer

Data plane: local per routing functions primarily forwarding moving a Datagram from an input link to an output link at a router

Control plane: network-wide logic, end-to-end getting packets one end of the network to the other end of the network coordination of devices

Services and Protocols

edge of network layer

Sender Host: network layer will take a transport layer segment from UDP or TCP and attatch it's header to it and encapsulate the segment in a datagram

Reciever Host: recieves the datagram, checksum inf like the checksum, extracts the payload & multiplexes to the correct transport layer protocol

Router:- recieves datagram from neighbouring host/ router from the network layer & pass it to the appropriate router/host

day / date:

Network Layer Function

Local :- decision or action made at an individual router

Global :- network wide

Forwarding :- router local action
forwarding the datagram from a routers input port to its output port
implemented in hardware

Routing :- global action
determining the route taken by a packet from source to destination
implemented in software

Dataplane :- per router, per IP ~~address~~ device. local function
Forwarding

Controlplane :- ~~dot~~ network wide logic
Routing

Router :- the router matches the bits in datagram header with the bits in the forwarding table to determine the appropriate output link the datagrams should be forwarded.

how does these local forwarding tables be computed?

- entered by hand by the network developer
- traditional routing algo
  distributed routing algo is run in all the routers
  routing algo function communicates with the routing algo function in other router to compute values in forwarding table
- Software Desige Networking
  remote software controller process, computes & distributes forwarding table used by each router

day / date:

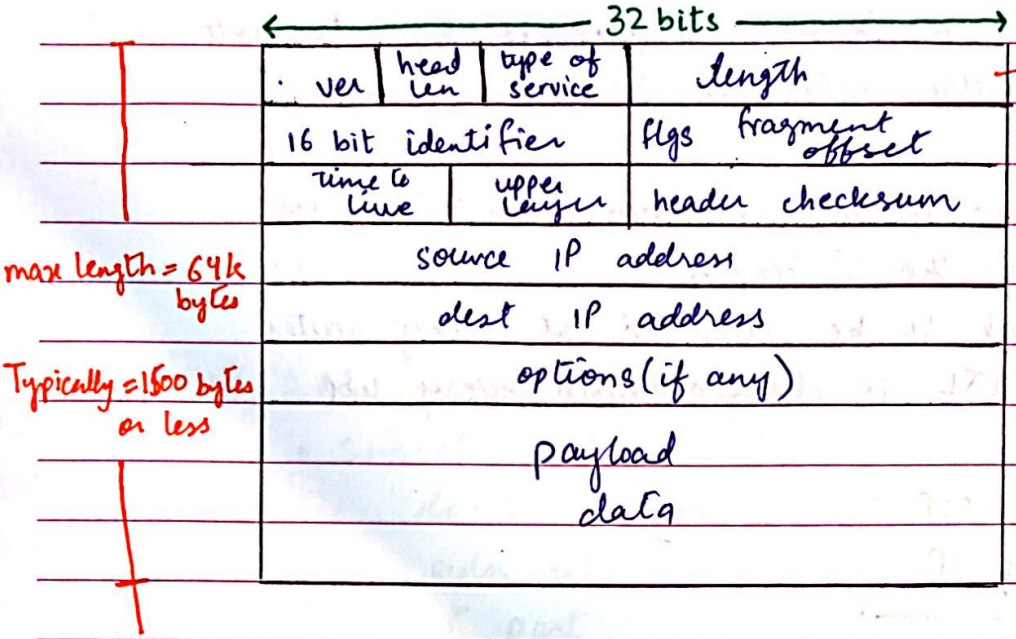
Services
- guaranteed delivery with specific delay
- in-order delivery
- flow gaurnteed in min bandwidth

Best Effort Services
- simplicity
  - easy To add a new host to add a new network
  - easy to manage
- provisioning of bandwidth
  - enough capacity allowed performance of real-time facilities
- Replicated
  - distributed application level infrastructure
  - services provided from multiple locations.

Internet Protocol
about
- datagram format
- IP address structure
- packet handling

day / date:

IP Datagram Format



ver :- IP protocol version number (4 bits)
ipv4 header

head len :- length of header

Data length :- Total Datagram Length − Header Length

generally 20 bit if options field is not used

type services :- to distinguish diff types of datagram from each other
ECN :- congestion ~~con~~ notification
diffserv :- 6 types of diff bits

time to live :- decremented each time datagram passes through a router
if TTL reaches 0 the datagram must be dropped at the router

upper layer :- Transport layer protocol
6 :- TCP segment
17 :- UCP segment

day / date:

- 16 bit identifier, flags, fragment offset :-
  used when a large datagram is segments into multiple smaller datagrams

- header checksum :- internet checksum computed over the contents of the IP header
  checksum need to be computed at every router since the TTL is changed with every hop

20 bytes of TCP
+ 20 bytes of IP

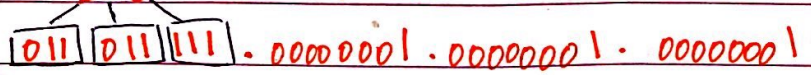
40 bytes + app layer overhead

IP Addressing

IP address does not identify a host or router per say rather our link layer interface of router or host
interface :- connection b/w host/router & physical link
router have more than 1 interfaces
multiple incoming & outgoing links
~~but how does our interfaces actually connected~~

IP address are in dotted-decimal notation

e.g 223 . . 1 . 1 . 1



day / date:

Subnet — router to router connection is also a subnet

peice of a network that contains all devices that can reach each other without passing through intervening network layer devices router that are directly connected to each other via some link layer technology

IP Address has 2 part

- subnet part
  device in same subset have common high order bits
- host part
  IP address will have diff host part which is the lower order bits

223 . 1 . 1 . 1

network prefix | subnet ID | host ID

for example duniya ke liye FAST ki ID hai 223.1 jo usko ICAN se mili hai ab local administrator isko subnets main divide kare gi

FAST ka network tou identify ho gaya ab subnet specify karne ke liye bhi bits chahiye wo mile gi host ID se (8 bits) last 8 bits jo host ID ki hoti hain us main se subnet ID aye gi aur host ID ki bits kum reh jaye gi agar

Agar 3 subnets banane hain tou 3 bits chahiye

$2^x = 3$

$\therefore x = 2$

host ID se 2 bits le le ge for subnet ID

day / date:

max IPadd = 2^{32} = 4B

IP addresses in Class A, B, C have 2 levels
- IP address = net ID , host ID
- IP address = net ID , subnet ID, host ID

Delivery of IP ~~address~~ packets
- delivery to the site router
- delivery to the subnet router
- delivery to end host

Subnet Mask
process of extracting the network address / net ID (if no subnetting is done) or subnet ID (if subnetting is done)
32 bit pattern
"1" in every net ID location & "1" in every in every subnet ID (if any) & "0" in every host ID location

e.g 141 / 14 . 21 . 21
10001101 . 00001110 . 00000010 . 00010101

subnet mask
11111111 . 11111111 .

day / date:

Defaults Masks with subnetting

Class A	255.0.0.0
Class B	255.255.0.0
Class C	255.255.255.0

Net ID = IP address AND Subnet Mask

CIDR

Classless InterDomain Routing

format : a.b.c.d/x

↘ no. of bits of subnet part

there is no rigid boundry b/w host and network & subnet part

11001000 00010111 00010000 00000000

network IP ← subnet part → host part

200.23.16.0 / 23

host part : 9 bits

total IP add available : 2^9

usable : $2^9 - 2$

host part which is all 0 / all 1

first & last bit is not usable

maanj thi IP address jis main 500 IP address hoon, uhaan ne 500 ke closest $2^9 = 512$ de di

day / date:

DHCP

Dynamic Host Configuration Protocol

dynamically obtain IP address from server when it joins the network using DHCP server

when device leaves the network the IP is returned so it can be reused

DHCP uses UDP

client uses port 68

server uses port 67

day / date:

Chapter 3

Flow Control

reciever tells the sender that rate at which segments need to be sent
reciever window variable (rwnd)

LastByteRcvd − LastByteRead <= RcvBuffer

- LastByteRcvd: last byte of data stream recieved from network
- LastByteRead: last byte in the stream to be read by app
- RcvBuffer: size of buffer

rwnd = RcvBuffer − [LastByteRcvd − LastByteRead]

LastByteSent − LastByteAcked <= rwnd

↳ variables maintained by sender

Connection Management

TCP connection have 3 way handshake
SYN bit is used to ~~is~~ in 3 way handshake
SYN bit is set to 1

FIN bit is used to close the connection
FIN bit is set to 1

day / date:

TCP Tahoe

timeout, duplicate ACK, data loss
cwnd will always go to 1

TCP Reno

timeout → cwnd goes to 1
duplicate ACK → cwnd will be half

ss_thresh :- will be half in any case.

linear growth :- AIMD / congestion avoidance
exponential growth :- slow start
if cwnd becomes half :- duplicate ACK
fast recovery

day / date:

150 . 100 . 80 . 0 / 22

150 : 100 : 01010000 : 0000000

= host

001
010
011 . . . . .

- convert into bits
- / x 32 − x = host ID ki bits
- dekho kitne IP add = y

  $2^x = y$

  x = no of bits required for IP addres
- subset bits = host ID bits − x
- update mask
  - → net ID + subnet
  - → ~~host~~ 32 − x

day / date:

Chapter # 5

Network Layer : Control Plane

Routing Algorithms Classification

- Global
  - compute paths at a global level using global information of the network
  - link state algorithms
- Decentralized
  - calculations based on their vicinity
  - based on the information of their neighbours
  - distance vector algorithms
- Static
  - routes once calculated change very slowly over time
- Dynamic
  - quick changes in response to link cost changes

How is the cost of a link computed

- cost of the link itself
  - e.g fiber optic link will have high cost compared to submarine link
- congestion
  - e.g high congestion on a link would mean higher cost so routers don't use that link

day / date:

Distance Vector Algorithm

Bellman-Ford Equation

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

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

$c_{x,v}$: cost to neighbour v

$D_v(y)$: cost of neighbour v to dest y

min taken over all neighbours v of x

Hierarchical Routing

- autonomous system (AS)
- an AS is typically under 1 same administrative control
- routers in an AS will be running the same routing algorithm & will have information of each other
- Intra-AS :- ~~router~~ protocol within an AS

AS:- set of Internet routable IP prefixes (of same subnet) belonging to a network or a collection of networks that are managed by a single entity or organization

gateway router :- edge router of an AS which helps in communication b/w two AS

- Inter-AS :- different AS can have different protocols communication b/w AS is done by gateway routers.

day / date:

- one AS can have multiple getway routers
- forwarding tables have both Intra-AS & Inter-AS routing protocols

In case of multiple gateway routers which how would a router know to which gateway router to forward the datagram?

- there are 2 tasks performed by Inter-AS protocols
  1. obtaining reachibility info from neighboring AS
     must know which network is accessible through which gateway router
  2. propogating this info to all internal routers
     every subnet is connected to routers & these routers are connected to gateway routers
     end host will a packet to its subnet router, now how will this router know to which gateway router to forward that packet to reach it's destination. This information will be provided by the gateway routers. (gateway routers bataye ge ke kis network ya subnet kee sathe connected hain)

2 communicating AS must be running on the same Inter-AS routing protocol generally BGP.v4

- hierarchical routing saves table size

if all Internet network is considered as a single set of routers, so all routers will be able to communicate with each other. This will result a flood of routing information on links that there will be no space for data. Secondly, every forwarding table will have millions of entries

day / date:

Inter-AS	Intra-AS
Policy	
admin controls traffic; policy can also control traffic which is not destined for it e.g AS1 & AS3 communicate via AS2, so AS2 have control wheather to allow it or not.	single admin so no policy decisions needed
performance	
policy dominates performance since need to protect the network, change, & restrict the traffic	focus on performance

Hot Potato Routing

- suppose a packet at AS1 needs to go to dest x
- X is reachable through AS2 and AS3
- the router at AS1 must decide which path to take
- sends the packet with the least cost link

OSPF (open shortest path first)

- usually implemented in upper tier ISP's
- use Dijkstra's algorithm (link state)
- open:- routing protocol specification is publically available
- each router creates a map of the entire AS not only its neighbor
- network administrator configure chooses the individual link cost
  - choose link cost to be one :- minimum hop routing
  - cost can be inversely propotional to the links capacity.

day / date:

security:- all messages are authenticated
simple authentication
MD5

multi-path:- multiple paths to the dest with same cost
OSPF allow multiple paths to be used
load sharing b/w paths

TOS:- for each link, multiple cost metric for different service can be used
complex cost computation

multicast:- 1 to a group of nodes in AS

Hierarchical OSPF

- AS can be configured hierarchically into areas
- each area runs its own OSPF routing algo withe eachoth
- each router in an area broadcaste its link states to all other routers in that area
- 1 or more area border routers are responsible for routing packets outside that area
- exactly 1 OSPF area in the AS is configured to be the backbone area
- backbone area contains all the area border routers and may also contain non area border routers
- inter-area routing within AS packet is first routed to an area border router then it is routed to the backbone router and then to the dest area
- flooding of advertisment is contained only in its area
- broadcaste link state every 30 mins or whenever a link state changes
- stateless

day / date:

checks if links are operational via "Hello messages" send to an attatched neighbors

BGP (Border Gateway Protocol)

- inter-AS routing protocol
- dominately implemented over Internet
- RFC 4271, 4724

functions of BGP:
- eBGP :- obtain subnet reachibility info from neighboring ASes, may be connected directly or indirectly
- iBGP :- propogate this info to all the routers in the AS and not only the gateway router.
- determine "good" routes
  not only on basis of cost but also policy

BGP session:
- BGP connection exist b/w 2 routers (a pair)
  - semi-permanent TCP connection
  - BGP peers
- use port 179
- exchange BGP messages

- BGP session b/w 2 ASes is called external BGP (eBGP)
- BGP session b/w routers in the same AS is called internal BGP (iBGP)
- BGP sessions do not always correspond to the physical links
  not neccessary that there is a BGP session when there is a direct link ; might exisist when there is not direct link

- an AS is identified globally by a unique Autonomous System number and is listed in RFC 1930
- a stub AS do not need an AS number
  ↳ an AS on which traffic originates or terminates.

day / date:

- attributes of BGP

AS-Path :- identifies the AS's through which a route has passed ; used to detect & prevent looping advertisement

Next-Hop :- indicates specific internal-AS router to next hop AS.

router interface that begins AS-Path (Next-Hop value)

used by routers to properly configure forwarding table

a link b/w inter & intra-AS protocol

BGP Route Selection

more than 1 route to destination AS

1. local prefrence value (assigned by network administrator)

based on policy decision / cost

routes with the highest prefrence values are selected

if only 1 route is selected routing will be done on that, next steps won't be performed.

2. shortest AS-Path
3. closest Next-Hop router

hot-potato routing

least cost path

at this path generally a single path is filtered out

4. additional criteria

more BGP attributes

SDN Control Plane

- network's forwarding devices :- packet switches
- forwarding decisions are made on the basis of transport layer, network layer & link layer packet header fields

day / date:

- logically centralized controller computes & distributes the forwarding tables
- controller interacts with control agent (CA) of each router
- interacts using well-defined protocol to configure & manage routers flow table
- Control Agent
  communicate with controller & do what it commands
  CA's don't directly intract with each other
  don't take part in computing forwarding table

- Characteristics of SDN
  Flow-based forwarding
  - packet forwarding rules are specified in switch's flow table
  - SDN control plane compute, manage & install flow table entries

  Segregation of data plane & control plane
  - dataplane consist of network's switches
  - control plane consist of servers & software that determine and manage switches flow tables.

→ Network control function
- external to dataplane switches.
- controller maintains accurate network state info
- provides this to the network-control applications
  provide the means through which these apps can monitor, program & control the underlying network

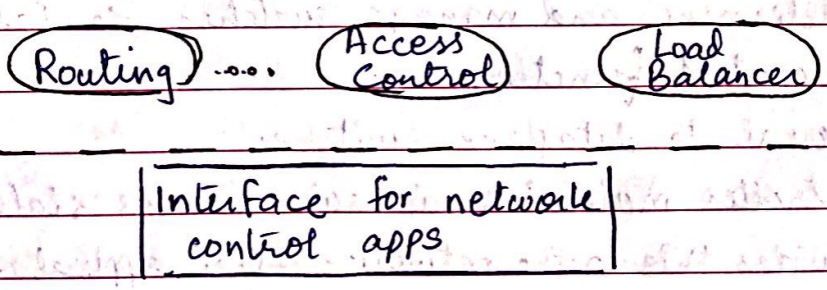
- Programmable Network
  - network control application might determine end-end paths
  - specify & control dataplane in network device

day / date:

- SDN control plane is divided into 2 components
  - SDN controller
  - SDN network-control applications

SDN controller

- control the operation of a remote SDN enabled switch / host
- a protocol is needed to transfer info b/w controller & device
- communication b/w controller & device is known as controllers "southbound" interface
- controller require up-to-date info about the state of network hosts, links, switches & other devices
  - ↳ for control decisions
- controller might maintain a copy of flowtables
- controller interacts with network control applications using "northbound" interface
  - ↳ allows network-control apps to read/write network state & flow tables



day / date:

Internet Control Message Protocol (ICMP)

- used mostly to signal error conditions b/w hosts & routers
- Example: ping / traceroute
- communicate network level info
  - error reporting
    - unreachable host, protocol, network, port, etc
  - echo request
    - ping
- ICMP messages are carried directly inside payload
  - ↳ UDP / TCP message are carried as payload in IP datagram
- upper layer protocol
- protocol number: 1
  - ↳ used for demultiplexing up from IP
- message consists of
  - 1 byte type field
  - 1 byte code field
  - 2 byte checksum field
  - upto 8 byte header field
    - of IP datagram that caused the message to be issued
    - e.g first 8 bytes of a datagram whose ttl exceeded

- type: 11, code: 0 TTL expired
  - router recieved a datagram, decremented TTL, & now TTL is 0

day / date:

Traceroute

- works by sending a set of 3 udp datagrams to dest
- 1st set is sent with an IP TTL field value set to 1
- 2nd set is sent with a TTL value of 2
- 3rd is sent with a TTL value of 3

⇒ when a router forwards datagram, always decrement the TTL value by1

- when TTL value decremented to 0, datagram is dropped at that router
- The router may send an ICMP message back to the source indicating that TTL value has expired
- IP add of the message containing that ICMP TTL expired message is the IP address of the router where the packet was dropped

if a packet have TTL value = n
reply back from the router is the router which is n hops away

- source also records the amount of time when it sent an IP datagram to the time when the corresponding ICMP message is recieved

measurement of the RTT from the host to that router

- when segment finally reach the destination dest host may return ICMP "port unreachable" message type: 3, code: 3
the source knows the segment / packet reached the end of its path

- ICMP is a tool for network management

day / date:

Chapter 6

Link Layer

- responsibility of transferring datagram from one node to physically adjacent node over a link
- encapsulate datagram in a frame & transfer it from 1 node to another adjacent node over link.
- each link may use different protocols and each protocol provide different services

Network Layer :- end to end

Link Layer :- adjacent node

Link Layer Services — every protocol do not provide all services

- encapsulate datagram into frame & add header AND trailer
  transport layer & network layer only add its header
- access channel can be a point-to-point link or a shared medium e.g Ethernet
- reliable delivery between 2 adjacent nodes
  transport layer also provide rdt but between to processes
- don't use rdt on low-bit-error links ~~but~~ like fiber

why both link layer & end-to-end reliability?

- efficiency is improved
- burden off TCP :- TCP doesn't have to perform retransmissions
- bit error corrects locally so TCP find the packet without bit error
- but in an end-to-end transmission some link might not provide reliability so rdt is ensured at TCP

- provide flow control
- error detection
- error correction
- half duplex & full duplex services

day / date:

most of its portion
- link layer is mostly implemented in the NIC card or the ethernet card or LAN card
- many of the link layer services are implemented in a chip at the heart of the NIC card
  controlling functionalities are implemented in the hardware
- the part of the link layer which implemented on the software runs on the hosts CPU

Most of the link layer is implemented in a chip in the NIC card (hardware) rest of the link layer is implemented on the software which runs on the hosts CPU.

Error Detection

EDC :- error detection & correction

D :- Data

- Datagram is forwarded to the link layer, which attatches EDC bit to the payload, then this payload is forwarded to the link which may be prone to error or not
  At the reciever side D' & EDC' is recieved which runs runs an error detection algorithm, if error occurs datagram is dropped else forwarded to the upcomming layer.
- more EDC bits, higher error detection reliability
  less EDC bits, lower error detection reliability
- There are primarily 3 error detection techiniques
  - parity checking
  - internet checksum
  - CRC

day / date:

Parity Check

- single EDC parity bit is added
- even parity
  EDC bit is 1 :- even number of 1's in the data + parity bit
  EDC bit is 0 :- odd number of 1's in the data + parity bit
  parity bit is choosen so the total number of 1's in data + parity bit is even
- this is error prone ; is not suitable for bursts of error
- to overcome this we use 2D bit parity
- in this we convert a stream of bits in rows n columns.
- each row & each col will have its own parity bit

Internet Checksum

- same as we did in transport layer
- add 2 bit words & take 1's complement of the sum

Cyclic Redundancy Check (CRC)

- forward error correction :- when reciever detects error & resolve it
- backward error correction :- when reciever detects error but don't correct it, instead sends NAK to sender, sender re retransmits
- D : data bits R : CRC bits G : pre agreed generator code
- reciever compute the generator code to check for error
- D+R should be exactly divisible by G
  if remainder is 0 then no error
- G is an algebric polynomial which needs to be converted into bits

day / date:

Multiple Access Protocols (MAC)

- There are 2 types of links
  - Point - to - Point
    - have 1 sender at one side of the link and 1 host at the other side
  - Broadcast
    - can be wired or unwired
    - shared channel
- problem: simultaneous transmission by node over shared channel
  solution: multiple access protol, algorithm to determine how nodes share a channel

Ideal MAC characteristics

- one node should be able to utilise complete capacity of the link / rate - R bps
- if M nodes want to transmit, the rate would be divided R/M bps
- decentralized, no one node to decide

Categories of MAC

- channel partitioning
  - e.g TDM and FDM
- random access
  - allow collisions but can recover from them
- taking turns

day / date:

Random Access Protocols

- transmitting at full rate R bps
- no prior agreement between node sharing a link
- when a node needs to transmit, it does so without checking if link is free which leads to collisions
- but there is mechanism for detection & recovery from collisions
- when collision is detected each node waits a random amount of time before transmitting again collision probability decrease

1- Slotted Aloha

- time is divided into slots
- 1 slots' length is atleast 1 frame wide
  1 frame should be transmitted completly in 1 slot
- if a slot have data to transmit, it will for the beginning of a slot (always)
- Analogy : each node flip a coin ; tails :- won't transmit
  heads :- transmit in the beginning of next slot
  head corresponds to probability p (transmit)
  tail corresponds to probability (p-1) (do not transmit)
- each node transmits at full bandwidth
- max efficiency is 37%. (at best performance par bhi low efficiency)
  ↳ only 37% of the slots do useful work
- Probability a given node has success :- $p(1-p)^{N-1}$
  probability the remaining node do not transmit :- $(1-p)^{N-1}$
  probability that any of N nodes has a success :- $Np(1-p)^{N-1}$
  ↳ efficiency

max efficiency :- $\lim_{N\to\infty} Np^*(1-p^*)^{N-1}$:- $1/e = 0.37$

2- Pure Aloha

- fully de-centralized, no synchronization, high collision rate
- node immediately transmits a packet
- probability that ~~any of N~~ a given node has success :- $p(1-p)^{2(N-1)}$

day / date:

3- CSMA (carrier sense multiple access)

- before transmission node listen before transmitting if senses channel is idle it transmits else defer packet
- collision can still occur nonetheless due to propagation delay
- node may not hear that another node have transmitted, so it senses that channel is free and starts its own transmission
- propagation delay is introduced due to the distance b/w 2 nodes
- even after collision nodes transmit the whole packet/frame
  ↳ no collision detection

4- CSMA/CD

- it further perform collision detection
- a transmitting node continues hearing on the channel and if detects another transmition, ceases its own transmission
- waits a random amount of time before retransmission
- ethernet uses CSMA/CD, and it is implemented in NIC
- the random time is choosen using binary (exponential) backoff
- node choose K randomly from $(0, 1, 2, \cdots, 2^{n-1})$ where 'n' is the number of collisions experienced previously by the node for that frame.
- the amount of time node waits is K·512 bit times if K=1, it waits 512 bit times ie 512 ms for a 100 Mbps ethernet
- more collisions experienced, larger the interval from which K is choosen

- efficiency $= \dfrac{1}{1 + 5\, t_{prop} / t_{trans}}$
- less t_{prop}, greater efficiency since collidings nodes will abort immediately efficiency = 1 (100%)
- greater t_{prop}, less efficiency

day / date:

Taking Turns

- using channel partion protocols and MAC protocols
- mechanism is used so nodes take turns to transmit
- 2 mechanisms
  - polling
  - token passing

Polling

- one node on the channel is master & the rest are slaves
- master control the turns, assign turns
- usually polls using round robin fashion
- goes around asking each node if it wants to transmit, and if a node do want to it is assigned a number a certain number of frames, after the limit moves to the next node.
- if a master detects transmission is finished transmitting then move to the next node

disadvantage:
- highly centralized
- polling delay

example : bluebooth

Token Passing

- there is a token exchanged b/w nodes, whichever node has the token it will transmit and only hold unto the token if it have any frame(s) to transmit using full capacity
- if the node which have token crashes then the whole system crashes since token will also be gone
- in this case, token recovery process is also needed

disadvantage:

a router have 2 interfaces with diff IP add & MAC add

day / date:

- ARP is a simple request - response protocol
- Host A broadcaste a request packet containing IP address of B & MAC address FF.F.FF.FF.FF
- all hosts/routers on the LAN will accept the frame & extract the datagram
- B will match the IP address with its own IP & send a respone packet only to A containing it's MAC address
- A will save the MAC address of B in its ARP table
- the rest of the nodes will discard the frame
- ARP table entry have 3 things IP add : MAC add : TTL
  TTL is normally 20 min after which the entry will be discarded from the table
- ARP request is broadcaste
  ARP response is unicaste

Ethernet

- bus topology in mid 90's, broadcaste LAN
- hub-based star ~~tology~~ topology in late 80's
  hub is a physical layer device which that acts on bits not frames
  boosts the bits energy strengh & transmit it onto other interfaces
- switched ethernet prevails today
  active switch in centre rather than a hub

Frame Structure

preamble	dest address	source address	type	data (payload)	CRC

- data contains IP datagram ; 46 bytes ≤ payload ≤ 1500 bytes (max MTU)
- if a payload < 46 bytes then ethernet protocol stuff bytes in the payload to make it a minimum of 46 bytes
- if payload > 1500 bytes network layer perform fragmentation

day / date:

- dest address is a 6 byte field, contains MAC add of dest
  source address is 6 byte field, contains MAC add of soace
- type is a 2byte field, used to multiplex network layer protocol
  a host may support multiple network-layer protocols for diff applications
  so the adapter at any host needs to know to which network layer
  protocol to should it pass the frame, this is identified in type field
- CRC is a 4 byte field
- Preamble is a 8 byte field, 56 bits of alternating 1's & 0's
  first 7 bytes 10101010 (always)
  last byte is SFD (8 bits) which is also alternating expect last 2 bits
  10101011
- 1st 7 bytes are used to wakeup the recieves adapter & to
  synronize with senders clock
- clocks are out of synronisation because there will always be a
  drift b/w target rate & transmitted rate, this drift is not known
  by the reciever or any other adapter on the LAN
- the last 11 of the SFD tells the reciever that preamble byte is
  about to end and actual /important stuff is coming

Ethernet Switches

- layer 2 switches, ~~network layer~~ link layer device
- do not need to configured unlike routers
- perform 2 functions :- filtering & forward
  determines whether to forward the datagram or filter/drop
  if forward then unto which interface to forward
- each switch table entry have 3 things
  MAC address
  interface which leads to that MAC address
  TTL / time at which the entry was made

day / date:

- suppose a frame arrives at a switch at interface x
- there are 3 possible cases
  1- there is no entry for the dest MAC address in the table in this case the switch broadcaste the frame to all interfaces except x
  2- there is an entry in the Table associating dest add to interface x, so there is no need to forward the frame, and filter function is performed (src add == dest add)
  3- there is an entry associating dest add with interface other then y, selective forwarding is performed for that interface only.

Self Learning

- switching table is initialy empty
- the frames recieved by the switch at any interface source MAC add, interface, time it arrives is stored in the table
- in this way it populates the table
- switch deletes an entry after sometime if no frames are recieved with that address as source this time period is known as aging time

- there are no collisions
- full duplex
- allow simultaneous switching (transmition) without collisions as long as dest is different
- transmition with same dest add can not happen simultaneously

day / date:

- suppose a frame arrives at a switch at interface x
- there are 3 possible cases
  1- there is no entry for the dest MAC address in the table in this case the switch broadcasts the frame to all interfaces except x
  2- there is an entry in the Table associating dest add to interface x, so there is no need to forward the frame, and filter function is performed (src add == dest add)
  3- there is an entry associating dest add with interface other then y, selective forwarding is performed for that interface only.

Self Learning
- switching table is initialy empty
- the frames recieved by the switch at any interface source MAC add, interface, time it arrives is stored in the table
- in this way it populates the table
- switch deletes an entry after sometime if no frames are recieved with that address as source this time period is known as aging time

- there are no collisions
- full duplex
- allow simultaneous switching (transmition) without collisions as long as dest is different
- transmition with same dest add can not happen simultaneously