Date
Page

**Flow Control:** occurs when speed of sender is more than speed of Receiver.

→ Open Loop → prevention Policies
- ① Re-transmission Policy
- ② Window Policy
- ③ Acknowledgement Policy
- ④ Discarding Policy
- ⑤ Admission policy

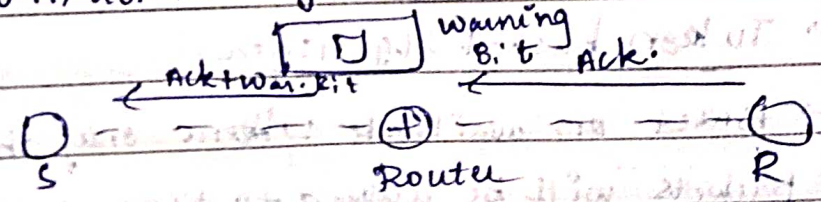
→ closed Loop → Removal Policies:
- ① Back Pressure
- ② choke Packet
- ③ Implicit Signaling
- ④ Explicit Signaling

14/11/23

Congestion Control Techniques:
- 1) warning bit
- 2) choke Packet.
- 3) Load shedding
- 1) RED (Random early Discard)
- 2) Traffic shaping
  - Leaky bucket algo.
  - Token bucket algo.

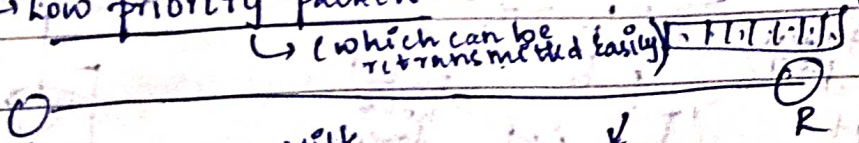
① **warning Bit**

→ A special bit in packet header is set by Router to warn the sender when there is a conjection. (warning Bit is Piggy bagged with acknowledgement).




③ **Load shedding**

→ Low priority packets will be discarded first.
↳ (which can be retransmitted easily)

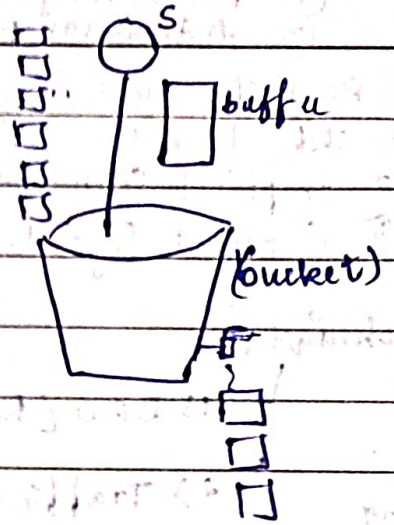



Appn → Milk, wine

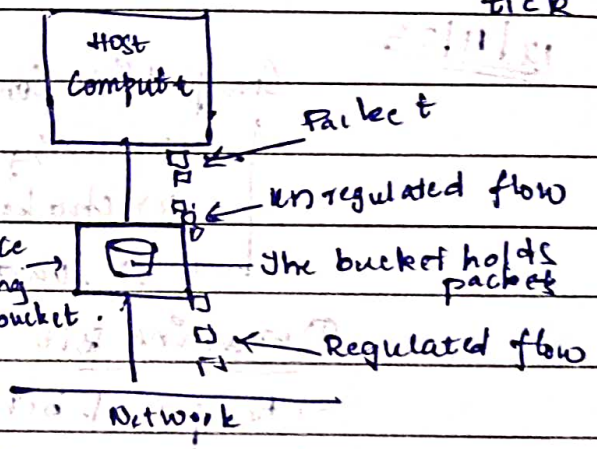
when buffer is full, low priority data is discarded

(1) RED → We discard the data before buffer is filled. We fix a threshold value, if buffer size exceeds threshold val. we start discarding data. This method stops congestion.

(2) • Leaky bucket algorithm:




→ When sender sends large amount of data, it is stored in the bucket and data is sended to reciever at particular clock tick

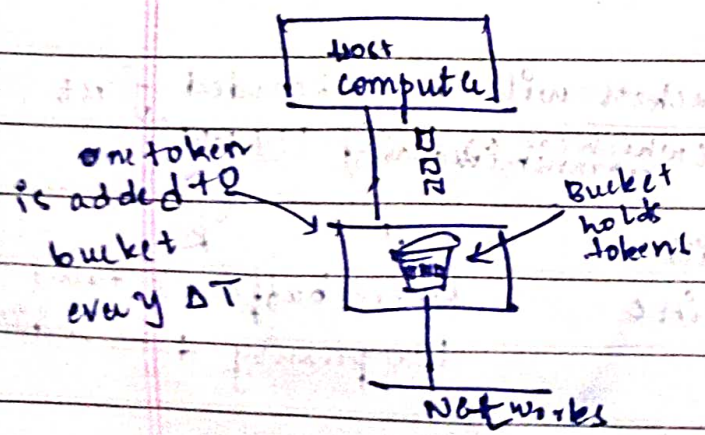


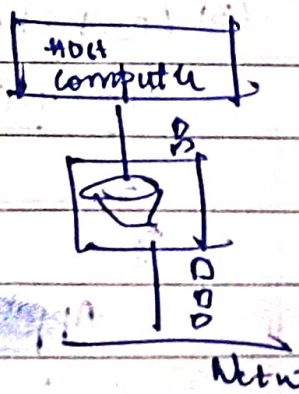

→ At every (fixed) clock tick, one packet is transmitted unless the queue is empty.

→ Unregulated flow of data will be regulated.

→ Token bucket algorithm

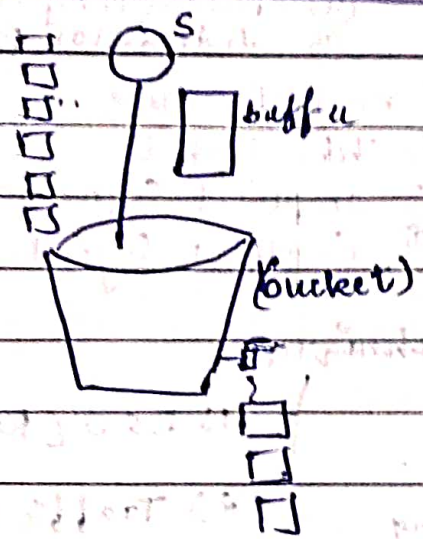
→ Based on available tokens only those no. of packets will be allowed to transmit to N/w



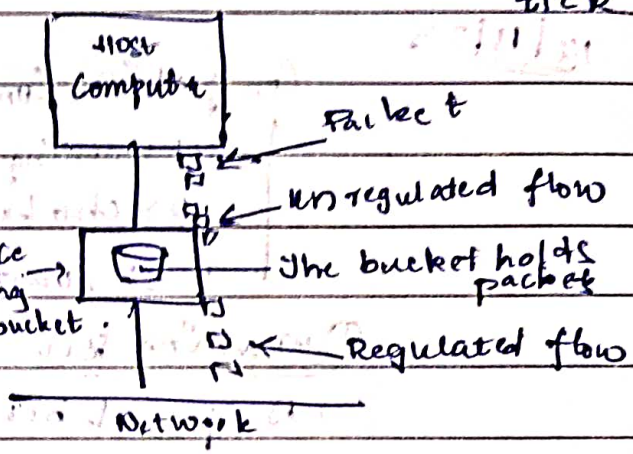



(1) RED → We discard the data before buffer is filled. We fix a threshold value, if buffer size exceeds threshold val. we start discarding data. This method stops congestion.

(2) • Leaky bucket algorithm:




→ When sender sends large amount of data, it is stored in the bucket and data is sended to reciever at particular clock tick.

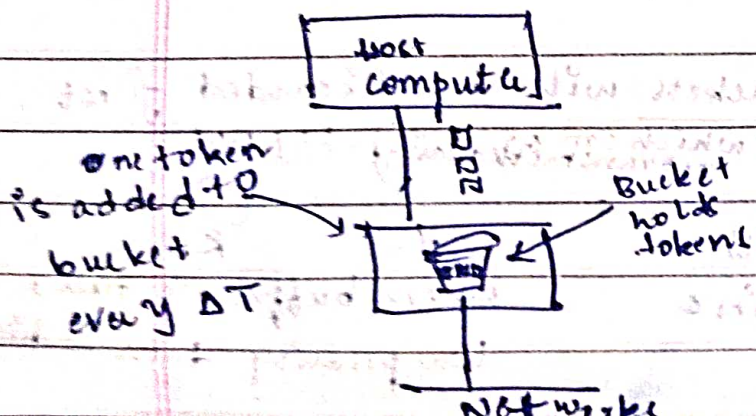


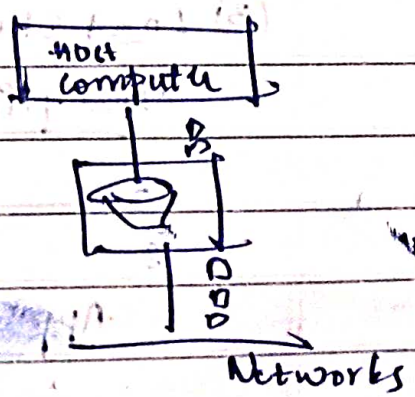

→ At every (fixed) clock tick, one packet is transmitted unless the queue is empty.

→ Unregulated flow of data will be regulated.

→ Token bucket algorithm

→ Based on available tokens only those no. of packets will be allowed to transmit to N/W



16/11/23

Date
Page

# Quality of Service (QOS)

→ Requirements: A stream of packets from source to Dest. is called a flow. The need of each flow can be characterized by 4 parameters:

1) Reliability → No bits may be delivered incorrectly (can be achived by error checking)

2) Delay

3) Jitter → uneven delay in packet delivery

4) Bandwidth

| | Reliability | Delay | Jitter | Bandwidth |
|---|---|---|---|---|
| Email | High | Low | Low | Low |
| File Transfer | High | Low | Low | Medium. |
| Web Access | High | Medium | Low | Medium |
| Remote Login | H | M | M | L |
| Audio on demand | L | L | H | M |
| Vedio on demand | L | L | H | H |
| Telephony | L | H | H | L |
| VedioConfuerence | L | H | H | H |

→ Together these parameters determine QOS the flow requires.

Techniques for providing QOS:

① Over Providing → provide so much router capacity, Bandwidth & Buffer Space.
(give more than needed.)
drawback: It is Expensive

② Buffuring → It only ~~soothes~~ smoothe out the jitter it doesn't affect Reliability, Bandwidth & Delay.

③ Traffic Shaping: Regulating avg. rate of data transmission. Traffic Policing → monitoring traffic flow is Traffic Policing (2 tech → Leaky, Token bucket)

④ Resource Reservation: Possible only with connection oriented service. Reserve all resources req. for data transfer. Resources that are reserved are Bandwidth, Buffer space, CPU cycles.

⑤ Admission control: (act accordingly)

⑥ Proportional Routing: Split your packets (traffic) equally among the routers.
Simple method to divide is → divide packets equally proportional to capacity of outgoing links.

⑦ Packet Scheduling: A scheduling algorithm is Fair Queuing algo. also uses Round Robbin

Jitter:
→ The variation in packet arrival time.
→ This prblm is seen in Audio & Vedio Streaming.
→ some packets 20ms & others taking 30ms to arrive will give an uneven quality to sound & movie
→ For audio data Router checks to see how much the packet is behind/ahead of it's schedule. This info. is stored in packet & updated at each hop.
→ If a packet is ahead of it's schedule, it is held to get it back on schedule, if it's behind the schedule router tries to get it out the door quickly.
→ For vedio data, Jitter can be avoided by buffering at reciever

Jitter control

→ For vedio confuences & live streaming, buffuing is not acceptable.

Internetworking: → IPV4 → IPV4 addressing
- Logical Addressing: → IPV6 → IPV6 addressing

IPV4 Addressing:

It is a 32-bit address that universely & uniquely defines connection of a device.
(uniquely ↓ no 2 connec. will have same IP address)

for Eg: (A computer / Router to the internet)

Address Space:

(How many IPV4 addresses are possible → ($2^{32}$ bits)
= 4, 294, 967, 296.
→ is the total no. of addresses used by the protocol.

17/11/23 Notations:

① Binary Notation
& 32 bits are represented in binary form.

② Dotted Decimal Notation
& rep in decimal, each octet (8bits) are seperated by dot(.)

Binary → 01110101 10001101 11001100 10110011
Decimal → 117 . 141 . 204 . 183
Decimal → 128 . 11 . 3 . 31
Binary → 10000000 . 00001011 . 00000011 . 00011111

→ IP V4 uses class-full addressing.
$2^{32}$ address → divided into diff Groups.

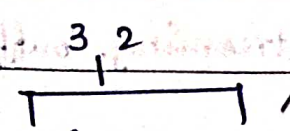
Class Full Addressing

→ (Here) This architecture is called Class full addressing
→ In class full addressing, total address space is divided into 5 classes (class A, B, C, D, E)

Class A designed for → Large Organisations.
class B → Medium Range Companies.
class C → Small Organisations.

Class D → Future purpose
Class E → special purpose

Date
Page




Address of N/W

| | | |
|---|---|---|
| | $2^8$ | $2^{24}$ → Class A |
| | 16 | 16 |
| | 216 | 216 → Class B |
| | 24 | 8 |
| | $2^{24}$ | $2^8$ → Class C |

Class A drawback:
The no. of connections are more than required.
→ We solve this using "Subneting".
↓
div. large N/w to smaller N/w's.
(adv) we can give access rights diff.

Class C drawback: No. of connections less than req.
→ We solve this using "supernetting" → combining N/w's
→ Each class occupies some part of address space.

| | Binary Notation | Dotted Decimal |
|---|---|---|
| Class A | 0 | 0-127 |
| Class B | 10 | 128-191 |
| Class C | 110 | 192-223 |
| Class D | 1110 | 224-239 |
| Class E | 1111 | 240-255 |

Classes And Blocks:

| | No. of blocks | Block Size | Application |
|---|---|---|---|
| A | 128 | $(2^{24}) \rightarrow 16,777,216$ | Uni Casting: |
| B | 16,384 | $(2^{16}) \rightarrow 65,536$ | " |
| C | 2,097,152 | $(2^8) \rightarrow 256$ | " |
| D | 1 | 268,435,456 | Multi Casting |
| E | 1 | 268,435,456 | Reserved |

Class full addressing ← Limitation → No. of addresses are getting Wasted.

Classless Addressing:
Block of addresses are given based on the demand/organization need.

21/11/23

Date
Page

Rules:

↳ Wasting so many connections in classful addressing.
↳ Address Depletion

→ classless addressing is designed to overcome address depletion & give more org. access to internet.

→ In this schema there are no classes but addresses are still granted in blocks.

→ The size of block varies based on nature & size of organization

Restrictions / Rules of classless Addressing:

1) The addresses in a block must be contiguous one after the other

2) The no. of addresses in the block must be power of 2.

3) The first address must be evenly divisible by no. of addresses.

Block

| a) Decimal | b) Binary |
| --- | --- |
| 205.16.37.32 | 11001101 00010000 00100101 00100000 |
| 33 | ⋮ |
| ⋮ | ⋮ |
| (Last) → 205.16.37.47 | 00101111 |

⇒ **Mask** → It is a 32-bit address in which no. of 1's rep N/w id & no. of 0's rep. Host id

CIDR - Slash Notation

205.16.37.40 / 28

← N.id → ← H.id →

11 001101 10000 100101 100111

Date
Page

Eg) 205.16.37.39 /28 · Find block size, starting & Ending address

Block size = $2^{32-28} = 2^4 = 16$ (Mask = 4)

Starting Address : (last 4bits → 0) → 10011100 → 32

Ending Address : (last 4bits → 1) → 10011111 → 47.

(B) 197.15.34.61 /26

Block size = $2^{32-26} = 2^6 = 64$ (Mask = 6)

61 → 111101 → starting = 0

→ ending = 63

Starting address = 197.15.34.0

ending " = 197.15.34.63

Network address Translation :

↳ can be done by Internet Service Provider.

N/w address translate maps local IP address to static IP addresses.

→ As the no. of home users & small business users are increasing day by day, it is not possible to give each & every user to one IPV4 address due to shortage of IPV4 addresses.

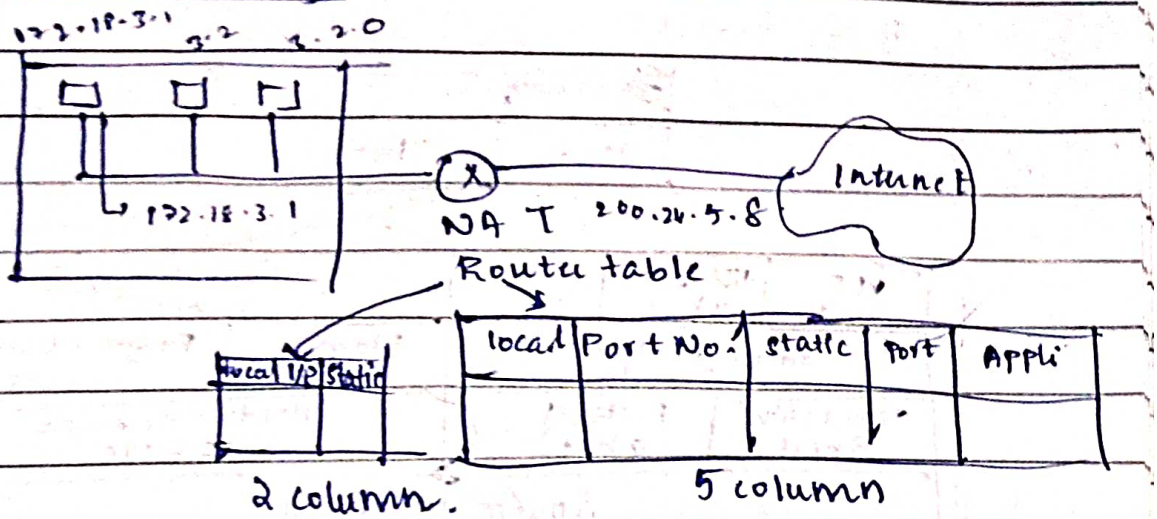
→ In order to overcome this problm developers designed the concept of Private (local) IP addresses & NAT

→ NAT enables a user to have large set of addresses internally & small set of addresses externally

Private Addresses:

| Ranges | total |
|---|---|
| 10.0.0.0 to 10.255.255.255 | $2^{24}$ |
| 172.16.0.0 to 172.31.255.255 | $2^{20}$ |
| 192.168.0.0 to 192.168.255.255 | $2^{16}$ |

Address Mapping




IPV6 Addressing:

→ In order to overcome the problem of address depletion, to eliminate concept of NAT & private addresses IPV6 was designed.

→ In IPV6 there is no need for classless addressing DHCP (Dynamic Host Configuration Protocol)

IPV6 structure:

→ 128 bits are div. into 8 sections each of 4 Hexa digits separated by a Colon (:)

→ IPV6 specifies Hexadecimal Colon Notation -

original Rep. ← FDEC:0034:0000:0000:0000:BOFF:0000:FFFO

→ FDEC:34:0:0:0:BOFF:0:FFF6

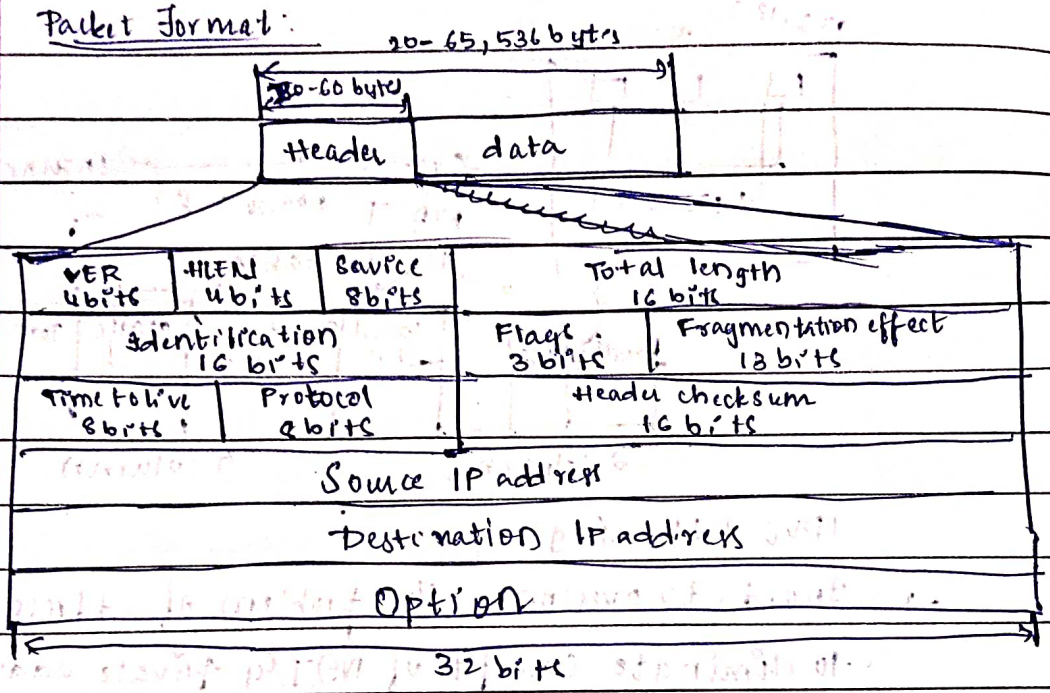
abbreviated Rep. → FDEC:34::BOFF:0:FFFO

(Continuous are rep. using "Gap").

more abbre. Rep.

IPV4 Packet Delivery:

Packet Format:




→ IPV4 delivery mechanism is used in TCP/IP Protocol.

It is an unreliable & connectionless datagram protocol

→ If reliability is imp. IPV4 must be paid with a reliable protocol such as TCP.

Data Gram:

→ Packets in IPV4 layer are called "DataGrams".

→ A data gram is a variable length packet consisting of 2 parts: Header & Data.

→ The header is divided into 4 sections: Version, Header length, Services & Total length (→ Header + data len).

Version:

→ It identifies version of protocol. Currently, 4th version of Internet Protocol is using.

HLEN (Header length):

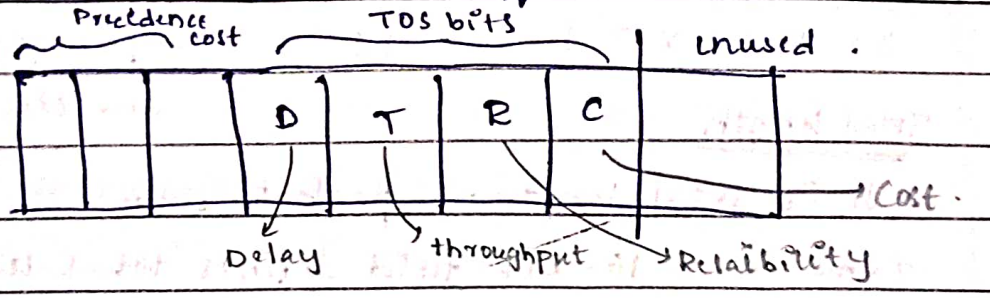
→ It defines total leng. of datagram header in 4-byte words.

→ The length of header is variable b/w 20-60 bytes.

Date
Page

## Services:

→ Previously this field is called "Service type" but now name changed to "Differentiated Services".




→ Precedence bits rep. priority of packet.

→ TOS (Type of Service bits) → Last bit not used.

→ Precedence bits range from 0-7 defines priority of data gram in issues such as conjution

→ if all 4 bits are 0000 it have default Service (Packet)

0001 - minimize cost.

0010 → maximize Reliability

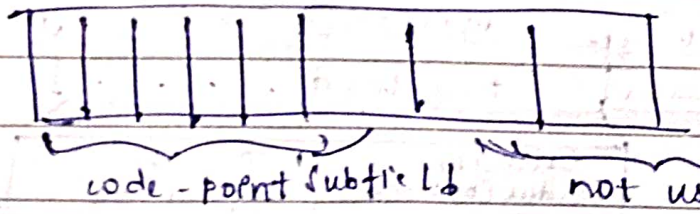
0100 → " throughput

1000 → minimize delay

Time to Live
How many routers it need to visit to reach the destination

## Differentiated Services

→ In this interpretation first 6 bits will make a code point sub-field




→ The code-point sub-field can be used in 2 diff. ways:

1) When right most 3 bits are "0" left most 3-bits acts as precedance bits

2) When 3 rightmost bits are not all zeros. The total 6 bits define 64 services ($2^6$).

→ GA services are categorized into 3 types:

| | | | |
|---|---|---|---|
| 1) x x x x x 0 | 32 | Internet Authority. |
| 2) x x x x x 1 1 | 16 | Local " |
| 3) x x x x 0 1 | 16 | Temporary / experiment services. |

**Total length**

→ It is total length of packet.

→ This is a 16-bit field defines total length of IPV4 datagram in terms of bytes.

**Fragmentation:**

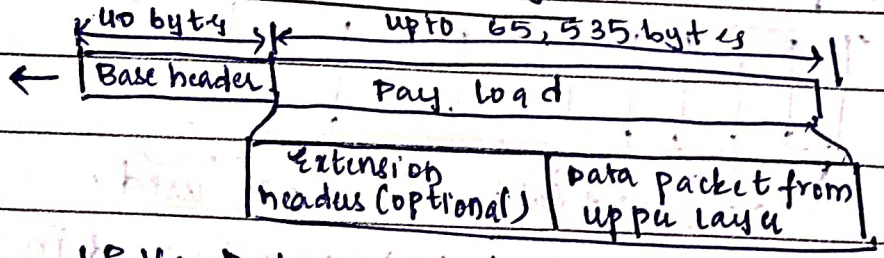
→ Identification: It is of 16-bits. It defines/identifies a datagram originating from the source host. It will give some unique Id to each & every packet.

→ Flag: 3 bits → 1) Reserved 2) Do not Fragment bit (if bit is 1 datagram should'nt be fragmented, if 0, it can be fragmented) 3) More Fragment.

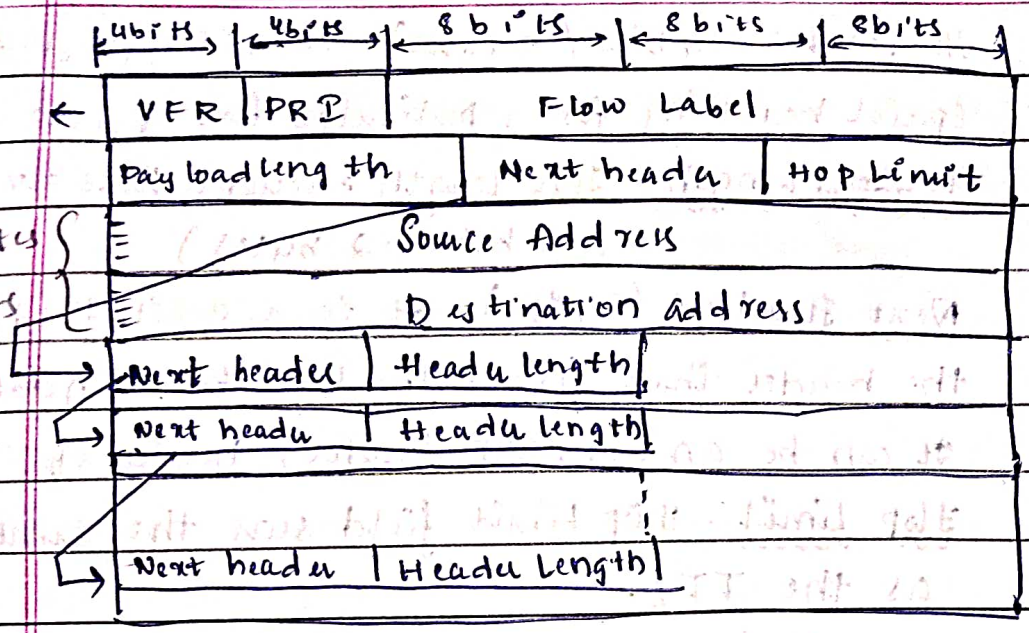
→ Fragmentation Offset: It shows the relative position of this packet w.r.t whole datagram. It is measured in units of 8 bytes.

24/11/23

**IPV6 Header Formatting:**




IPV6 Datagram header and Pay Load.




Disadvantage of IPV4 addressing:

Shortage of I/p address

Real time audio-video transmission → (unable to provide)

It is not providing anytype of encryption & authentication

⇒ IPV6 is introduced to overcome the deficiency of IPV4

IPV6 also called as → Ipng → Internet protocol next generation

Adv:

1. larger address space.
2. Better Header format.
3. New options.
4. Allowance of extensions.
5. Resource Allocation.
6. More Security.

Packet Formate:

→ In IPV6 each packet is composed of a mandatory base header & pay load ↔ optional extension header. ↔ actual data from upper layers.

Version: IPV6 → value of version is '6'

Priority: Defines priority of packet w.r.t traffic congestion

Textbooks:
Andrew S Tane..
A. Forouzan..

Date
Page

**Flow label**: (24 bits / 3 bytes). It is designed to provide special handling for a particular flow of data.

**Payload Length**: This length excludes Base Header length. (16 bits → 2 bytes)

**Next Header**: (8 bits). It is a 8 bit field defining the header that follows the base header in datagram. It can be an extension header / header of TCP/UDP.

**Hop Limit**: Hop Limit field serve the same purpose as the TTL.

**Next Header Codes**:

| | |
|---|---|
| 0 - Hop-by-Hop option | 43 - Source Routing |
| 2 - ICMP | 44 - Fragmentation |
| 6 - TCP | 50 - Encrypted Security Payload |
| 17 - UDP | 51 → Authentication<br>59 → Null - (No next header)<br>60 → Dest. Option. |

**PRIORITY**:

IPV6 divides traffic into 2 categories: 1) Conjestion Controlled (0-7)
2) Non-Conjestion Controlled (8-15)

① Priority: 0 → no specific traffic (Least Priority)
1 → background data (delivered in background)
2 → unattached data traffic (Eg: e-mail)
3 4 5 → Reserved
4 → attended bulk data traffic (Eg: FTP)
6 → Interactive Traffic (Eg: TELNET
7 → Controlled Traffic (Highest Priority)
(Eg: SNMP) (Simple N/w management protocol)

(carry real time audio vedio data.)

② 15, 8 are only used:

15 ← Less redundancy data
8 → high redundancy data

5/12/23

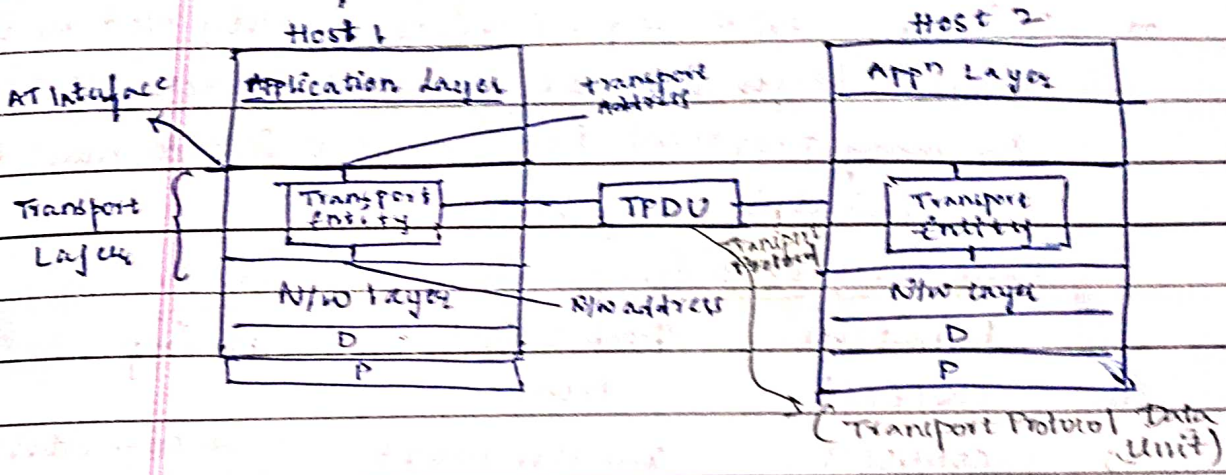
# UNIT-IV
## TRANSPORT LAYER

→ Transport Layer task is to provide reliable, cost effective data transport from source machine to Destination machine independent of physical (or N/w layer) N/w's.

**Services:** 1) Services provided to upper layers
2) Transport Service Primitives (commands)
3) Berkeley Sockets (used in Berkeley UNIX OS).

An eg. of Socket Programming is Internet File Server.

① **Services provided to Upper Layers:**
Provides 2 types of Services: ① Connection Oriented transport service ② Connection-less transport Service.




**Transport Entity:** Hardware / Software within the transport layer are called transport entity (It can be located in OS Kernel, Separate User Process, in a library package as a DLL / on the NIC (N/w Interface Card)).

**The Need for Transport Layer:**

→ The transport code runs entirely on user machines whereas N/w Layer mostly runs on routers which are frequently crashes everything is gone, a poor quality cannot be solved by better Routers. So, we need a layer on top of N/w layer that improves QOS.

With respect to transport layer first 4 layers are called Transport Service Providers, the remaining are called Transport Service Users.

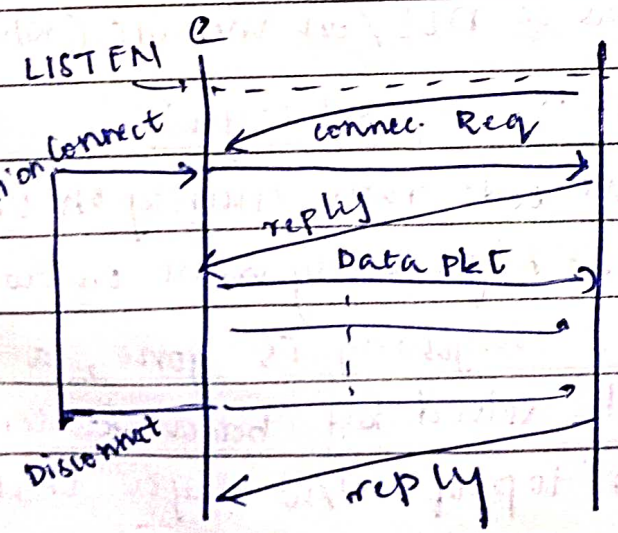
Transport Service Primitives :- To allow users to access the transport service transport layer must provide some opn to app programs. (possible through Transport Service Interface)

→ Transport Service is similar to N/w Service but there are some imp. diff. (1st diff: N/w service is intended to model the service offered by real N/w's) where as the Transport layer service is Reliable.

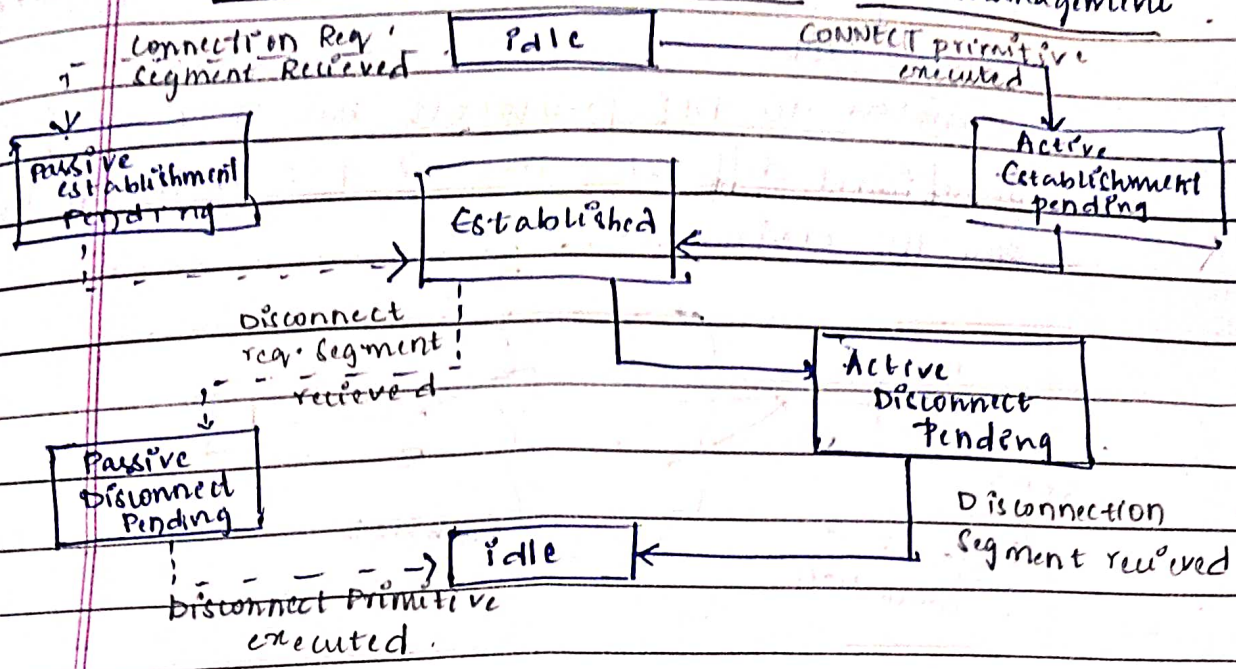
→ (2nd diff: N/w layer services are only used by the transport layer whereas transport services are seen by many programs i.e, Transport Service must be convenient (& easy to use))

Primitives of a Simple Transport Service :

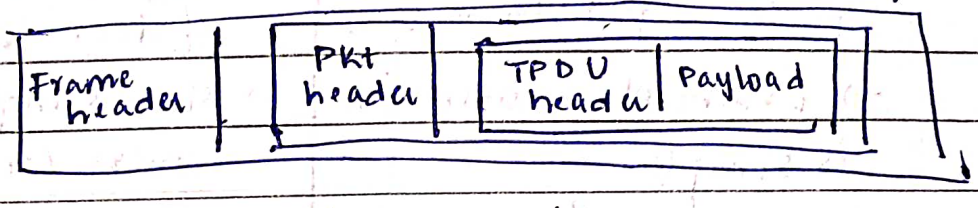
| | Primitives | Packet Sent | Meaning |
|---|---|---|---|
| 1) | LISTEN | None | |
| 2) | CONNECT | Connection request pkt | Sends conn. establishment req. to server. |
| 3) | SEND | Data Packet | waiting for the serv |
| 4) | RECEIVE | None | |
| 5) | DISCONNECT | Disconnect req. pkt | closing conn. |



## State diagram for simple connection Management.




TPDU – Transport Protocol Data Unit - A msg from Transport Entity to Transport Entity




## Berkeley Socket Primitives: 7/12/23

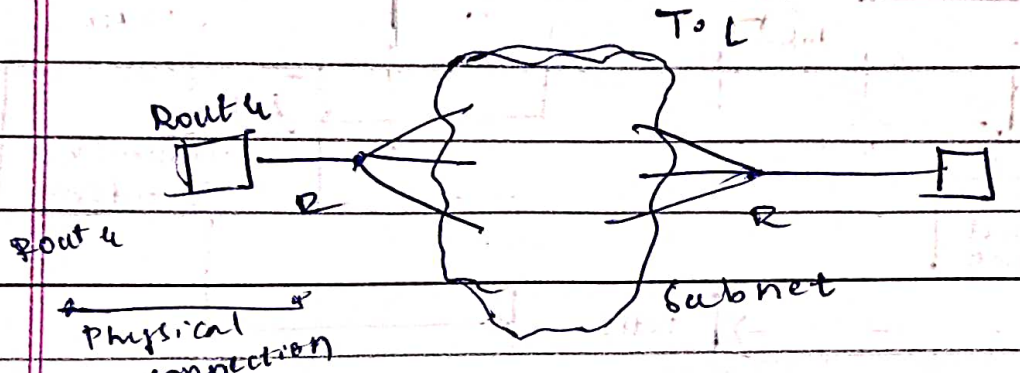
1) SOCKET → Create a new communication end point.
2) BIND → Associate a local address with a socket.
3) LISTEN → Announce willingness to accept connections. (give Queue size)
4) ACCEPT → Passively establish an incoming connection.
5) CONNECT → Actively attempt to establish a connection
6) SEND → send some data over the connection.
7) RECIEVE → Recieve data from the connection.
8) CLOSE → Release the connection.

## Elements of Transport Protocols:

→ Transport service is implemented by a Transport protocol b/w 2 transport entities.

→ These protocols deal with error control, sequencing & flow control.

→ Similar to DLL protocols but there are few significant diff. b/w these 2 due to the dissimilarity b/w the environments.




| DLL | TL |
|---|---|
| → Each outgoing line uniquely specifies a particular router. | → Explicit addressing of dest. is required. |
| → Establishing connection is simple. | → Establishing connection is more complicated. |
| → Doesn't require any Buffer Capacity. | → It requires buffer capacity for the subnets |
| → Implementation of flow control is diff from TL | → Implementation of flow control is diff from DLL |

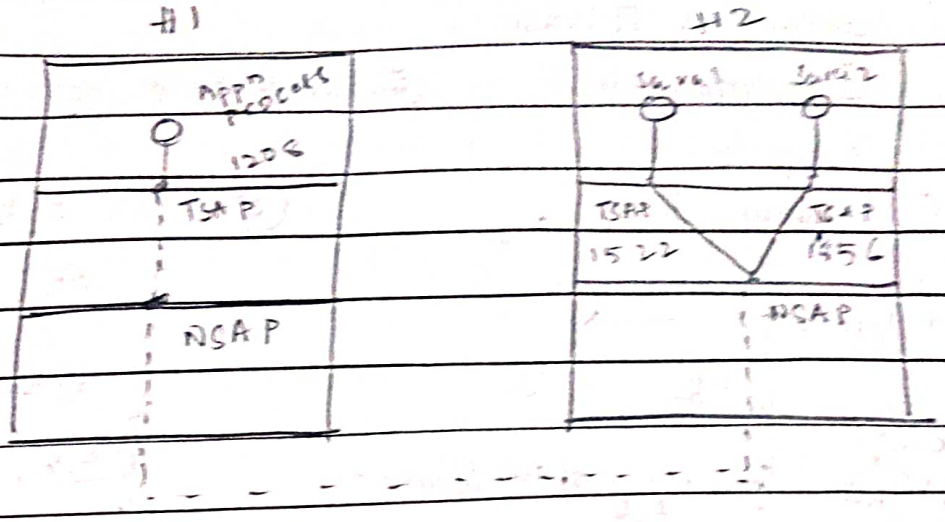
elements of TL ↙ ① Addressing ② Connection establishment ③ Connection Release

Addressing:

→ In internet the end points are called "ports".

→ In general they are called 'TSAP' ("Transport Service Access points")

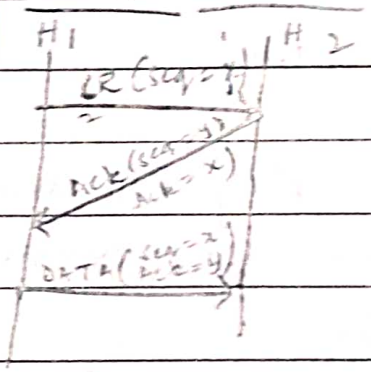
(Network Service Access Point) → (NSAP)



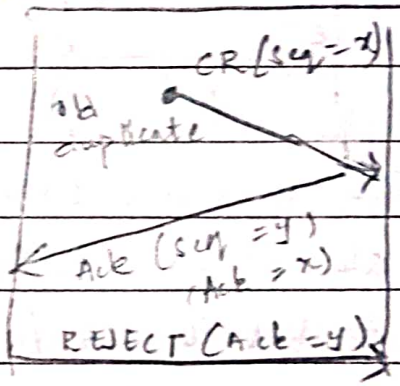

Connection establishment:

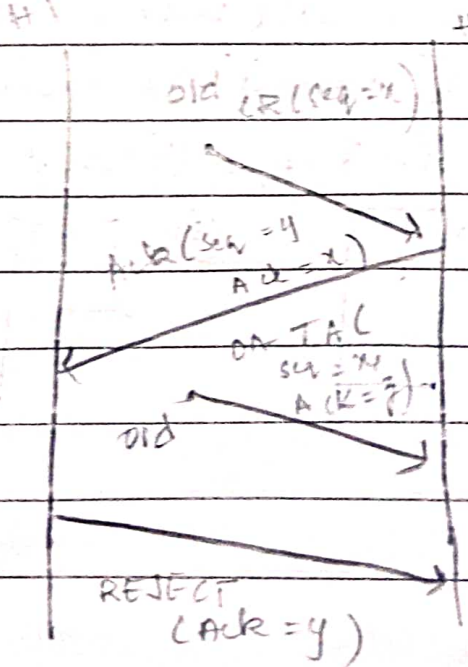
→ It is simple in normal Scenario but the problem occurs when N/W loss loss store duplication & heavy congestion occurs in the N/W. To solve this we use "three-way Hand Shaking" for connection establishment

① Normal Scenario:




② Duplicate Connection Request:



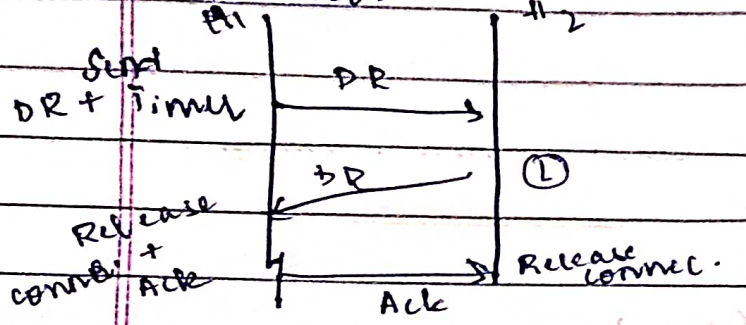

→ ③ Duplicate CR & Ack.

Connection Release :

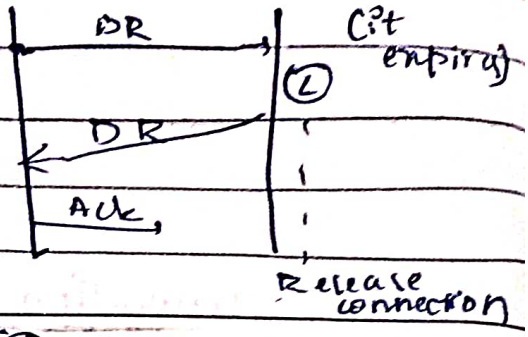
① Assymetric ② Symmetric

Symmetric:

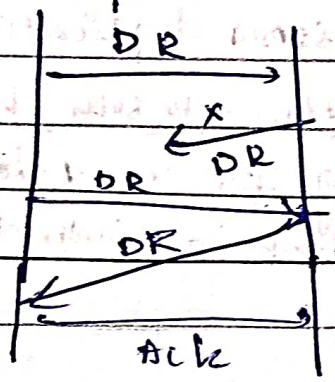
① Normal



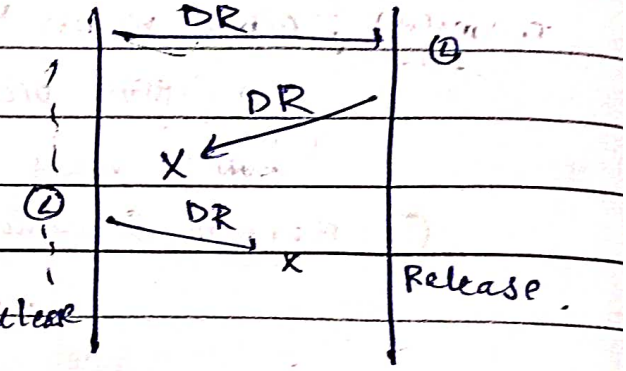

② Find Ack




③ Response




④ Both lost




8/12/23

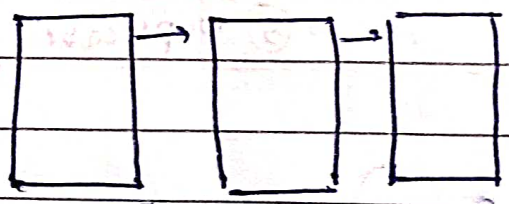
→ TL provides Error Control using Checksum & CRC & also it uses ARQ mechanism for retransmission (if packet is lost/discarded).

→ TL provides Flow control using "Buffers".

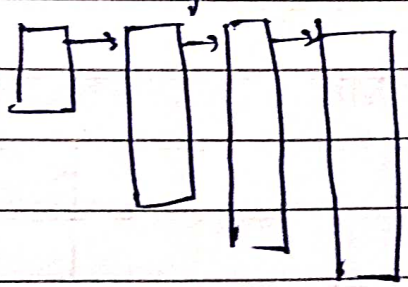
3 types of buffers: ① Chained fixed size Buffers.

② Chained Variable Size Buffers ③ One large Circular Buffer

①



②

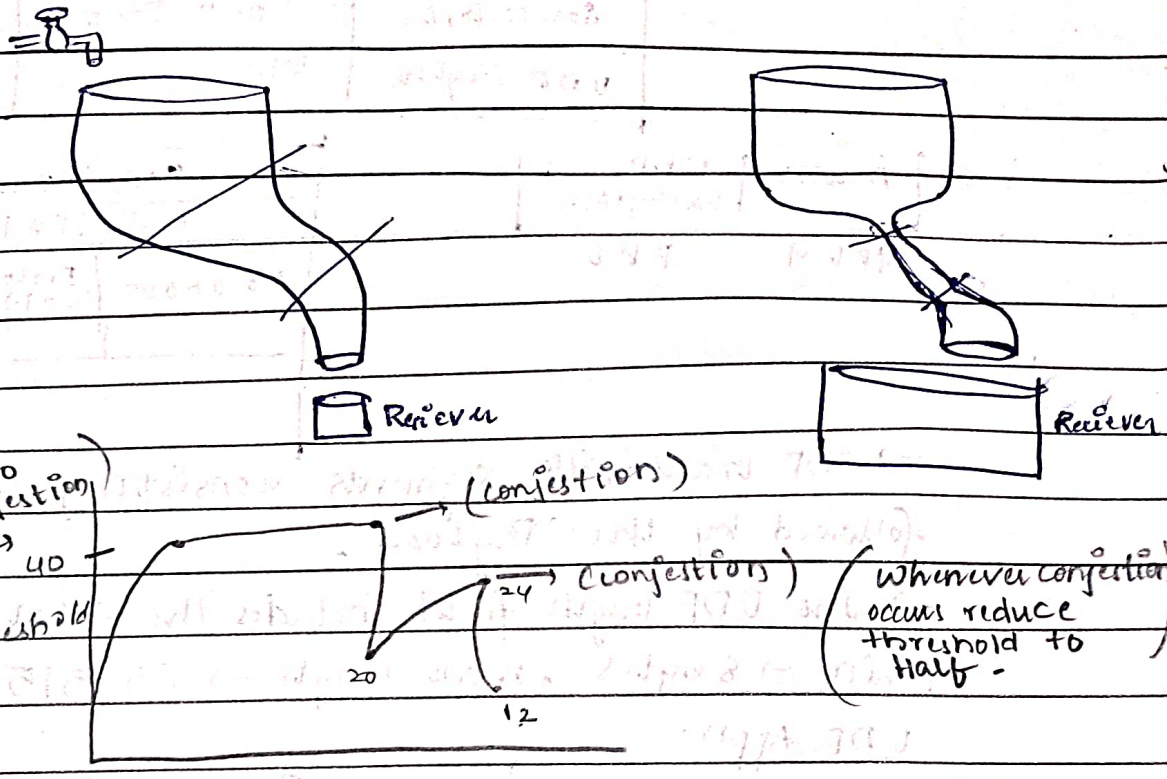


③

→ In TL congestion control can occur in 2 ways:

① You have a fast N/W & slow Receiver.

② Slow N/W and fast Receiver. (High capacity Receiver)




Multiplexing: Mul. app$^n$ data is combined together as a data unit & given to NL.

## The Internet Transport Protocol: UDP (User DataGram Protocol)

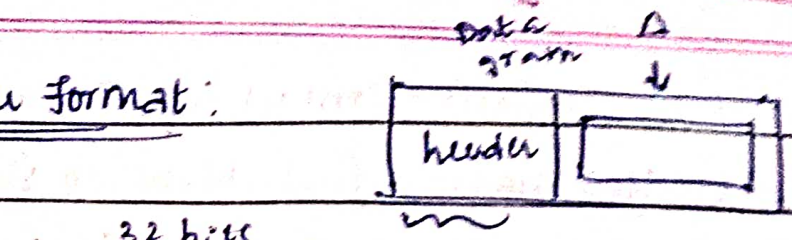
→ The internet has 2 main protocols in TL:

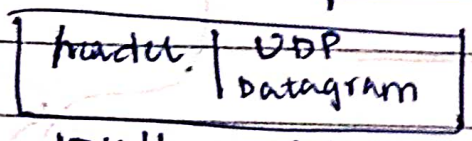
1) Connection Less Protocol – UDP

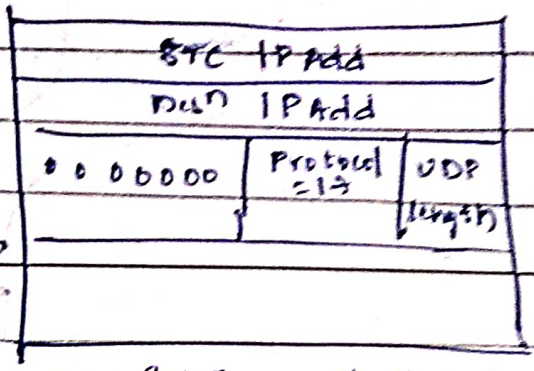
2) Connection-Oriented Protocol – TCP.

UDP: → UDP protocol does almost nothing beyond sending packets b/w app$^n$.

## UDP header format:



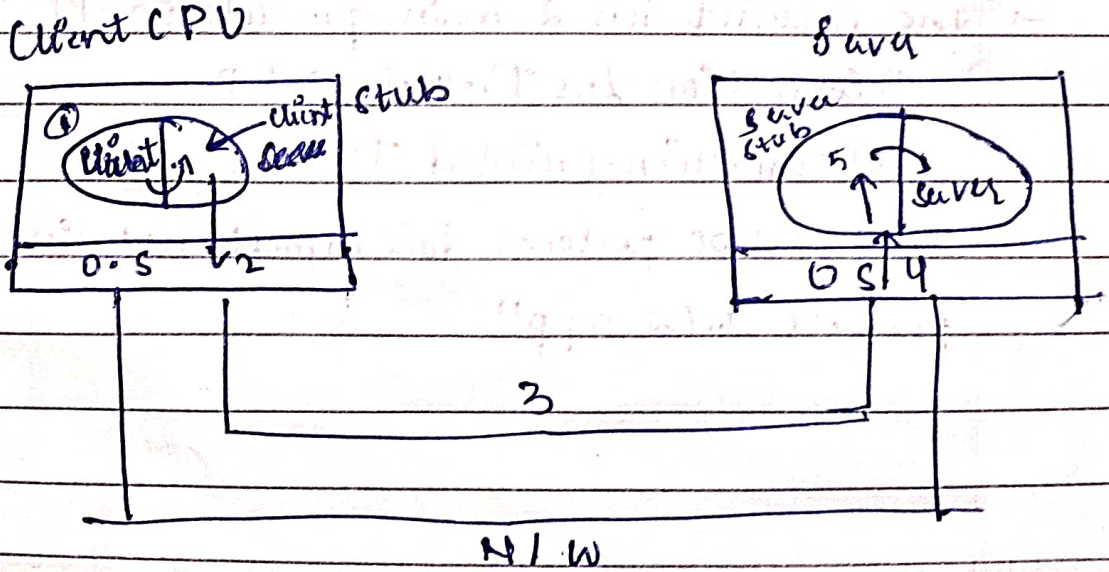




→ UDP transmits segments consisting of 8-byte header followed by the 'Payload'.

→ The UDP length field includes the total size of UDP. (min. → 8 bytes, max. length → 65,515 bytes).

UDP Appn:

1) Query Response Protocol. (Eg: DNS)
2) Speed (Eg: Online Games)
3) For Broadcasting Messages
4) Streaming (Live Telecasting, Vedio Conferencing)




Combining a msg along with Parameter: Marshalling

Date
Page

## TCP:

→ It was specially designed to provide a reliable end-to-end byte stream over a unreliable internetwork.

→ TCP was formally defined in Sep, 1981.

Assigned Ports of TCP.

| Port | Protocol | Use |
|---|---|---|
| 20,21 | FTP | → File Transfer |
| 22 | SSH (Secure Socket Shell) | → Remote Login |
| 25 | SMTP | → E-mail |
| 80 | HTTP | → WWW |
| 110 | POP-3 | → Remote Email Access |
| 143 | IMAP | |
| 443 | HTTPS | → Secure Web |
| 543 | RTSP | → Media player control |
| 631 | IPP | → Printer sharing |

→ TCP uses Full duplex point-to-point service.

→ TCP uses 'Stream Transmission' instead of 'Byte Transmission'.

Port Numbers: These port addresses are assigned by IANA (Internet Assigned Number Authority).

→ Port addresses range from 0 - 65,535.

↳ categorized into 3 types:

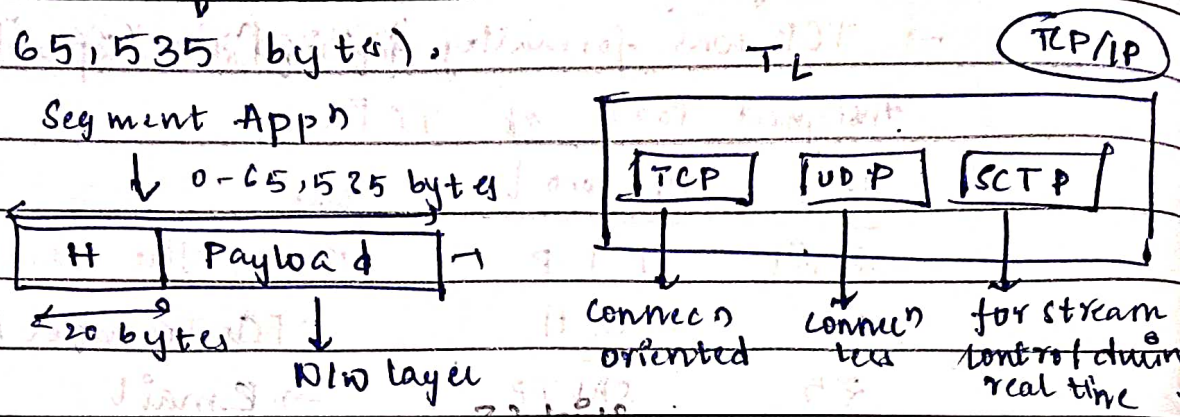
(i) 0-1023 → Well-known Port Numbers (Reserved for standard services) → assigned & controlled by IANA.

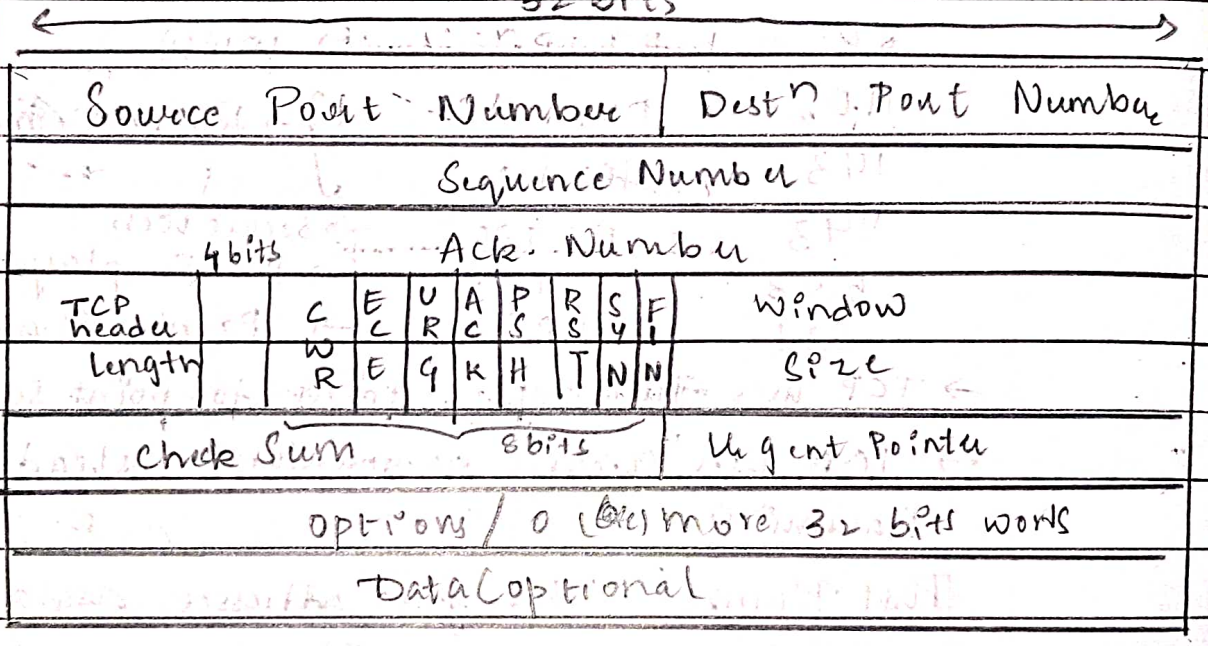
(ii) 1024-49,151 → Registered Port No.'s → Not assigned / controlled by IANA but registered by IANA to avoid 'duplication'.

(iii) 49,152 - 65,535 → Dynamic / private port no.'s. Not assigned, Not controlled & Not Registered by IANA, it can be used by any app^n Process.

Date
Page

TCP segment size should be 0 - 65,535 bytes since, when we send this segment to N/w layer it need to fit (since, N/w can accomodate only till 65,535 bytes).




(explicit congestion notif. Echo)

→ TCP uses a 32-bit sequence no.

→ CWR & ECE are used to signal the congestion when explicit congestion notif. is used.

(Congestion window Reduced)

→ URG (Urgent) → '1' → need to be sended immediately

→ ACK (Acknowledgement) → '1' → Segment using Ack. Service.
'0' → " not " " "

→ PSH (Push) → Similar to URG but sometimes TCP ignores this command. (more priority for URG)

→ RST (Reset) → Reset a connection

→ SYN → Used to establish a connection.

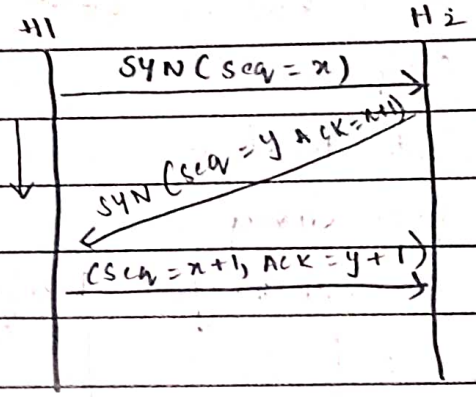
→ FIN → To release a connection.

→ Window Size → Flow control in TCP is handled using "sliding window". (how many bytes of data shld i send).

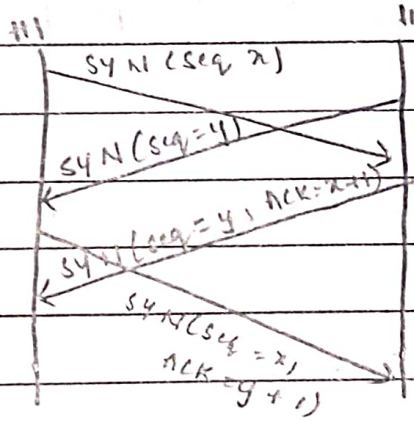
→ checksum → used for error control

→ Options → It provides a way to add extra facilities not ~~covered~~ provided by regular header.

## Connection Establishment:

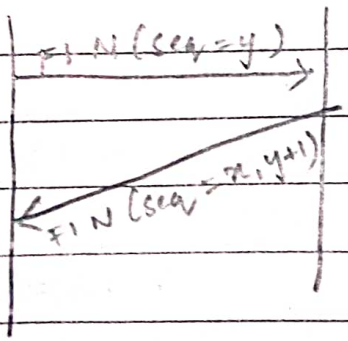



(connection establishment)




(simultaneously connection establishment on both sides)

## Connection Release:




(Connection Release)

Date
Page

## Performance Issues:

1. Performance problems → In complex N/W.
2. N/W performance measurements.
3. System design for better performance.
   1. CPU speed is more imp.
   2. Reduce Packet count.
   3. Minimize Context Switching.
   4. minimize copy
   5. use more bandwidth.
   6. Avoid Congestion
   7. Avoid Time-out.
4. Fast TPDU processing:
5. Protocol for Future high Performance.

14/12/23

Date
Page

# UNIT-4

## Application Layer

### DNS: (Domain Name System)

→ Although programs theoretically could refer to host, mailbox & other resources by their N/w addresses (IP address).

Ex: tana@ 128.11.24.41,

→ This addresses are hard for people to remember.

→ To decouple m/c names from m/c addresses, ASCII names were introduced.

Ex: tana@ ari.ucsb.edu.

→ The N/w only understands numerical addresses, so, some mechanism is req. to convert ASCII string to N/w address.

→ To solve this DNS was invented i.e, DNS is used to map the ASCII string to the N/w address.

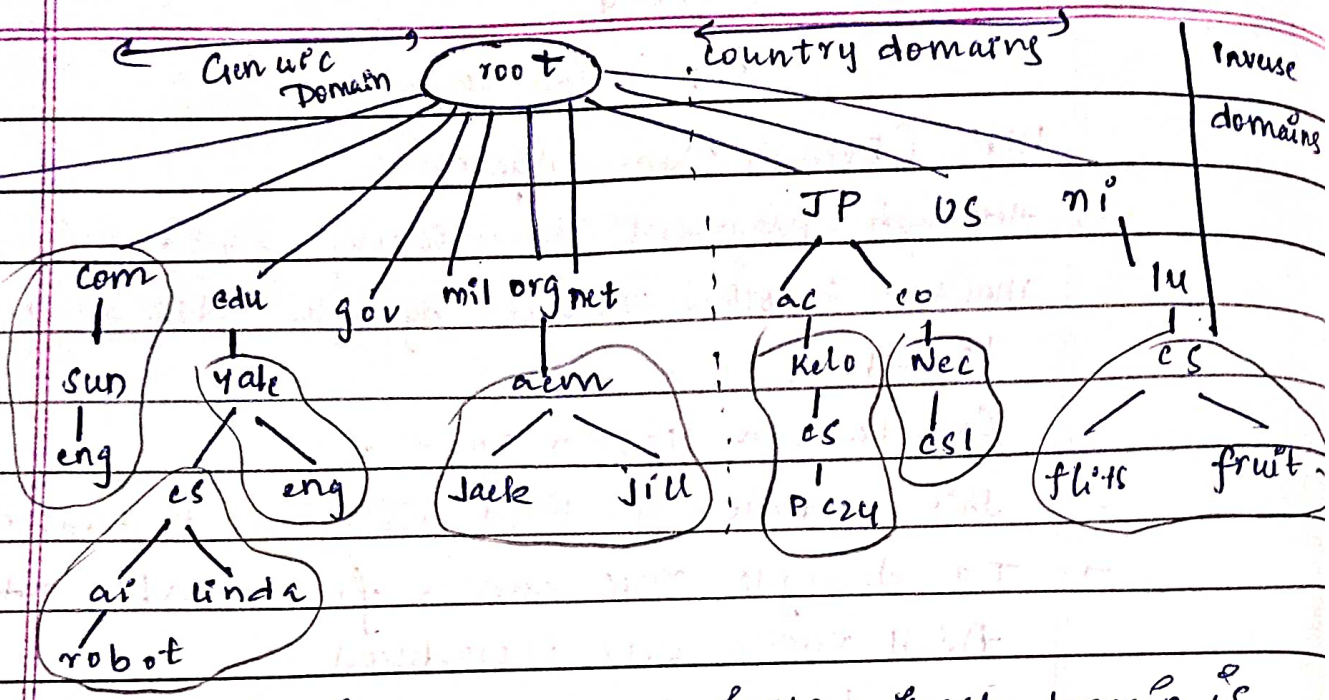
→ The essence of DNS is the invention of a hierarchical domain based naming schema & a distributed db for implementing naming scheme.

→ To map the name onto IP, app? program calls a library procedure called 'Resolver' passing it the name as the parameter.

→ DNS Name Space:

Managing a large constantly changing set of names, is a non-trivial problem, to solve this DNS uses hierarchical addressing.

- The Internet is divided into over 200 top level domains.




→ Each domain covers many hosts, Each domain is partioned into sub-domains & these are further partioned into so on.... all these domains can be rep. by a "Tree".

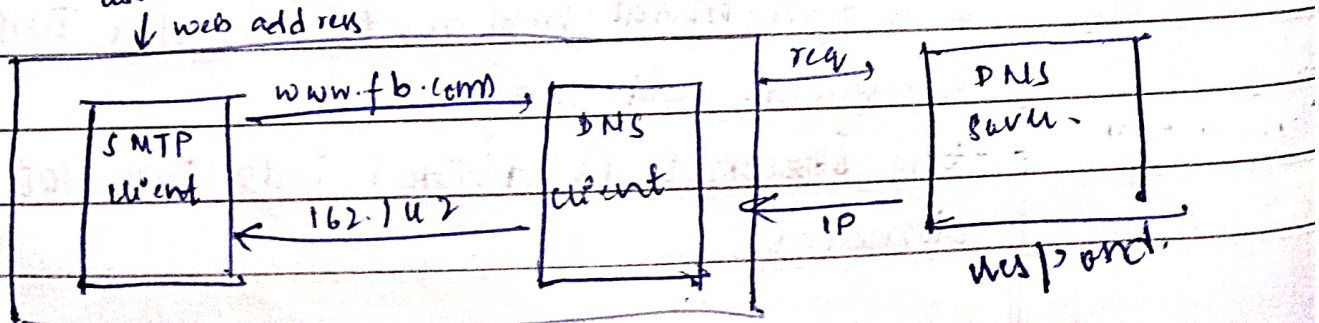
→ It has 128 levels, uses labels, domain names

→ The levels of tree represents, the hierarchy of the domain, the leaves rep the domains that doesn't have any sub domains.

→ A leaf domain contain a single host/may rep. a company. (contains 1000's of hosts)

→ Top level domains come in 3 flavours. Generic Domains, Country domains, Inverse domains

NOTE: In 2000 ICANN approve 4 more general top level domains they are: biz (for business), info (for information), pro (for prof., docs.), name



→ Each domain is named by a path upwards from it through the root (eg: Jack.acm.org).

→ Domain names can be absolute / relative.

→ Domain names are case insensitive.

→ Full path shld not exceed 255 chars. Component names shld not exceed 63 chars.

→ Domains can be inserted into tree in 2 ways:
① Generic ② Country.

15/12/23

## Resource Records:

→ Every domain can have a set of resource records associated with it.

→ For a single host the most common resource record is just it's IP address (Many other resource records also exists)

→ When a resolver gives a domain name to DNS what it gets back are the resource records associated to it's name.

① Domain Name → Domain to which the record applies

Resource Records: 5 Tuples

Domain_Name time-to-live class type value.

② time-to-live: How stable the record leaves.

high stable → 86,400 (no. of secs. per day)

low stable → 60 (1 min " ")

③ class → The class will be IN for Internet Info.

④ type field → Type field tells you what type of record that is.

| TYPE | Meaning | Value |
|---|---|---|
| SOA | Start of Authority | |
| A | IPV4 Address | 32 bit |
| AAAA | IPV6 " | 128 bit |
| MX | Mail Exchange | Priority |

(Start of Authority)

| | | |
|---|---|---|
| NS | Name Server | name of the server |
| CName | Canonical Name | Domain name |
| PTR | Pointer | Alias for IP address |
| SPF | Sender Policy framework | Text Encoding |
| SRV | Service | Host that provides |
| TXT | text | ASCII text. |

Ex:

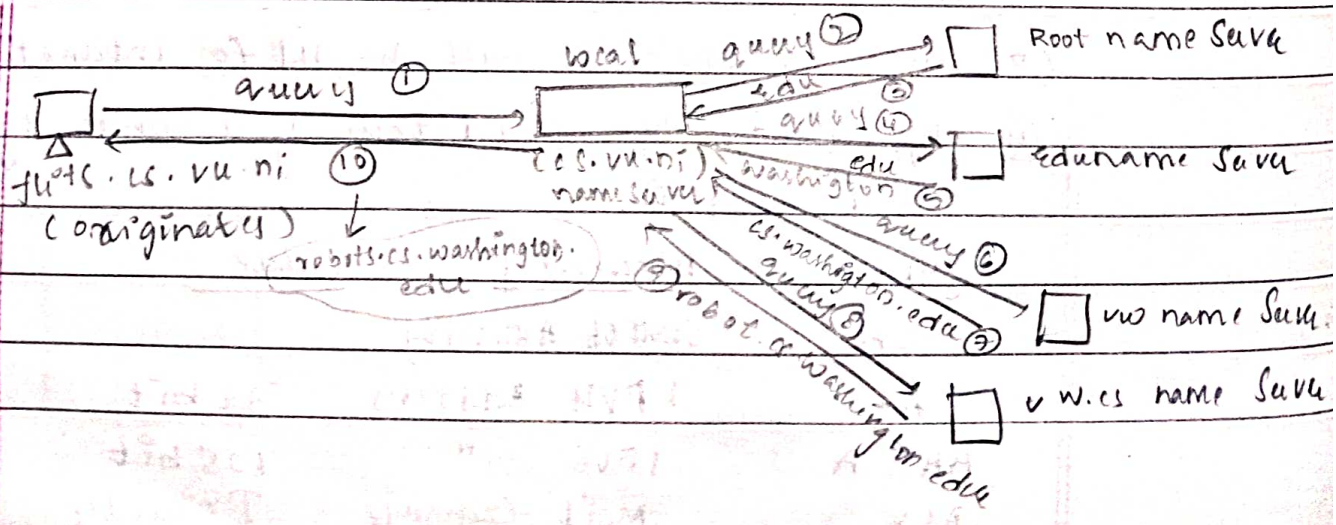
| | | | | |
|---|---|---|---|---|
| cs.vu.nl | 86400 | IN | SOA | star boes (9527, 7200, ... |
| Zephyr | 86400 | IN | SOA | 130.37.56.205 |

## Name Servers :

→ A single name server could contain the entire DNS db & respond to all queries abt it, but in practice the server would be so overloaded, if server crashes entire internet would go down. To avoid this prob. the DNS namespace is div. into non-overlapping zones

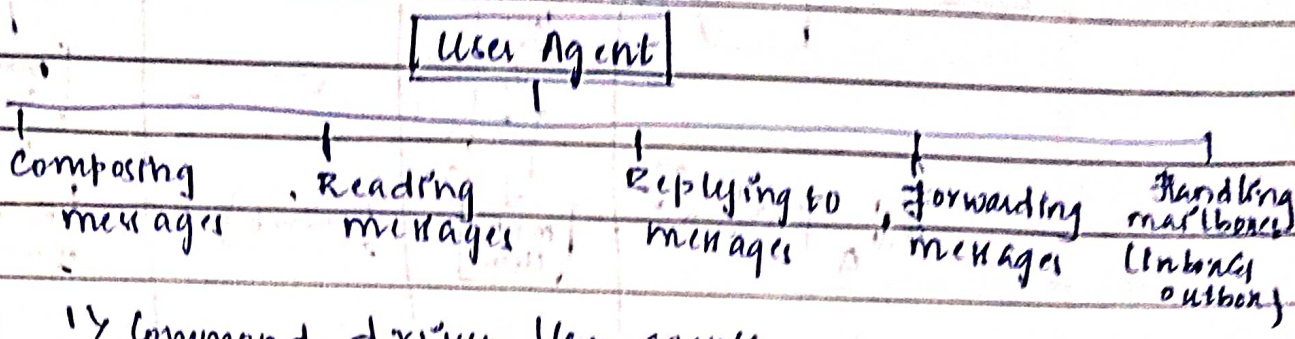
→ Each zone contains some part of the tree, also contains name servers, holding the info. abt the zone.

(Same diag. as Name Servers)



LDAP (Lightweight Director Access Protocol)

E-mail:

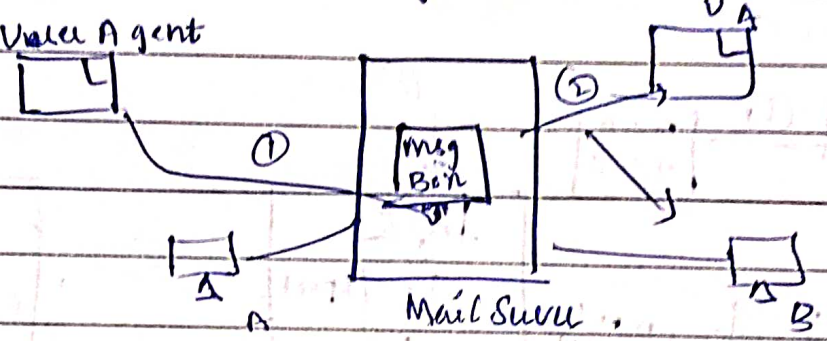



1) Command driven User agents.

2) GUI - User Agent.

21/12/23

1st Scenario:

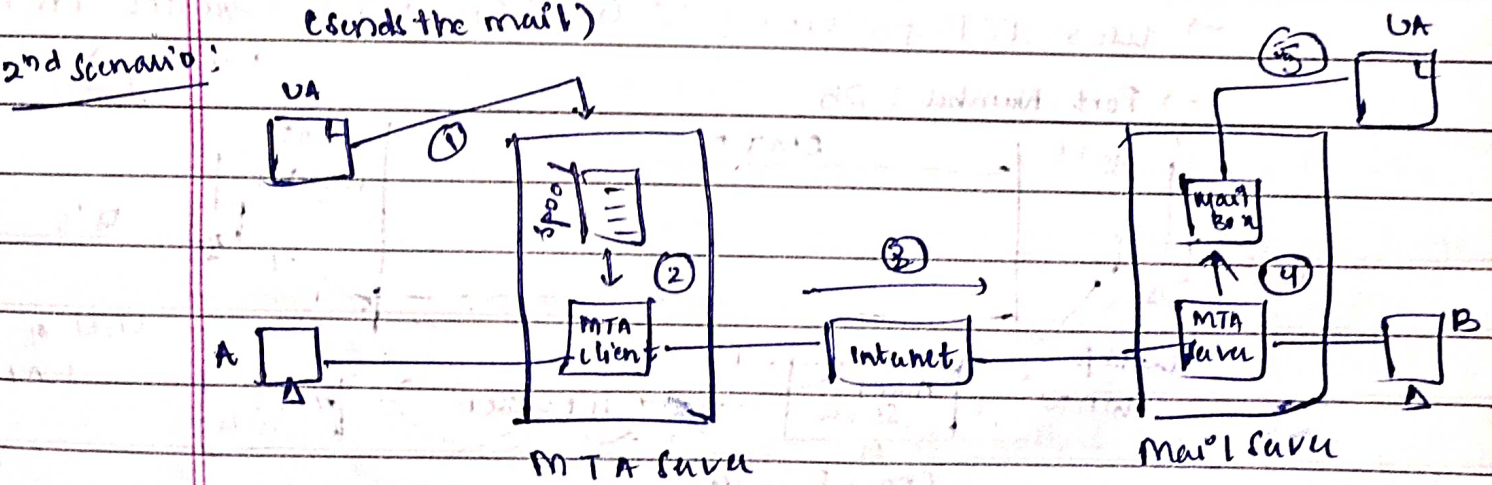



M.T.A → Message Transfer Agent

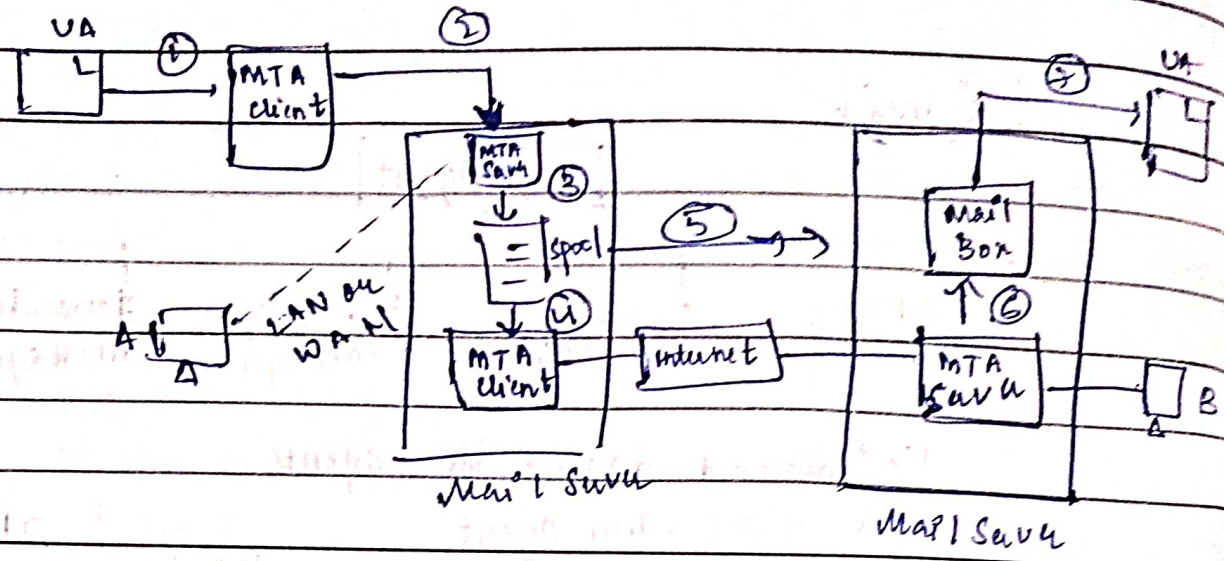
MTA client (sends the mail)

MTA server. (Receive the mail)

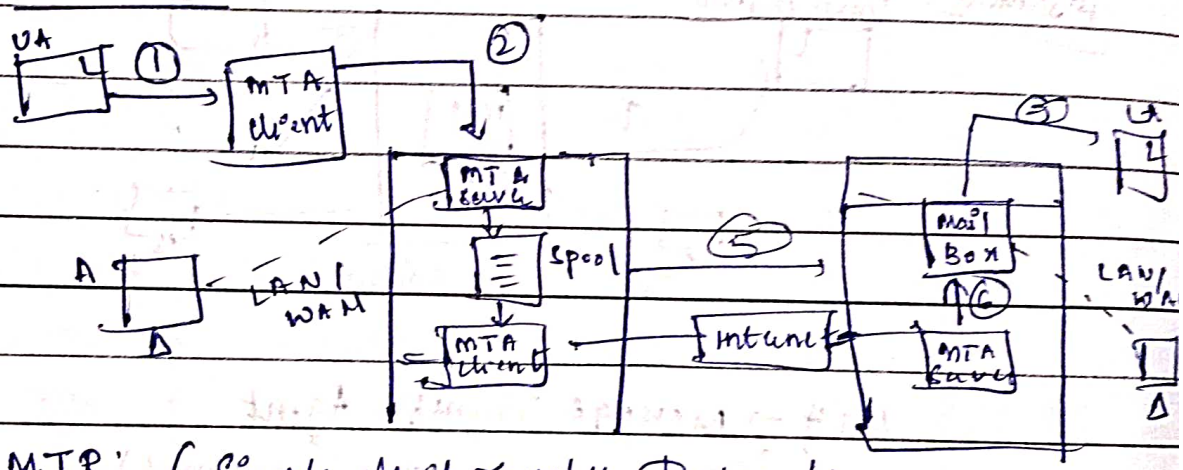
2nd Scenario:



3rd Scenario:



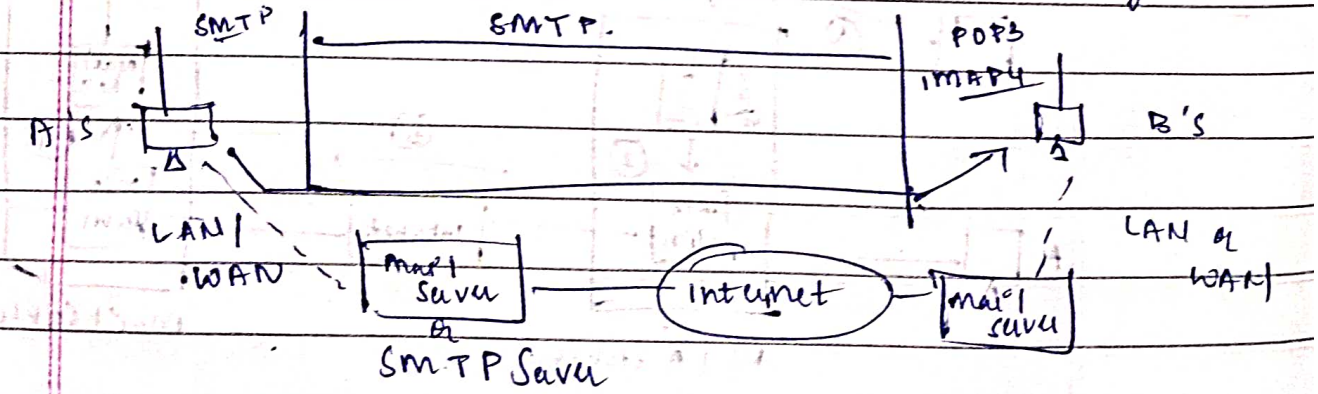

4th Scenario:




SMTP: (Simple Mail Transfer Protocol)

→ Uses TCP protocol (∵ Conn° Oriented → Guarantee in msg. delivery)

→ Port Number: 25




→ Used 2 tyms in Msg. delivery

i) b/w Sendr. & Sndr. Mail Server.

ii) b/w sendr's M.S & Reciever M.S.

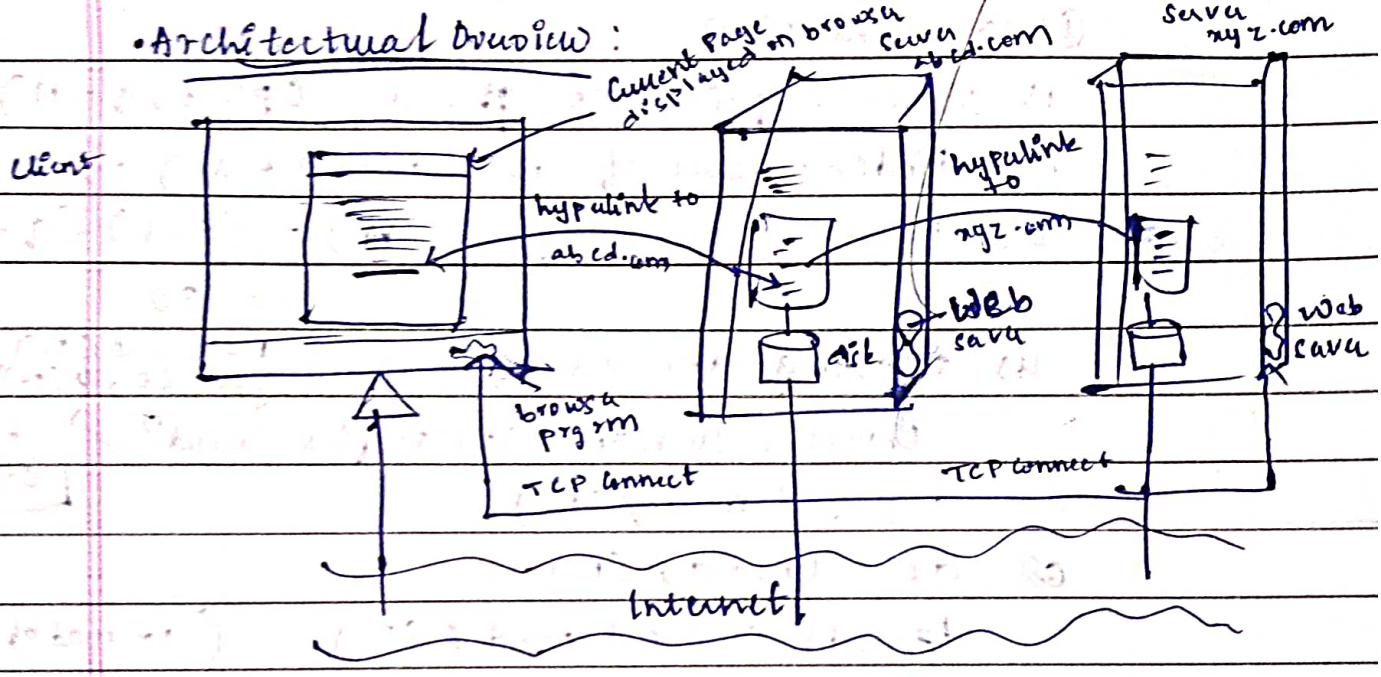
Date
Page

Command: HELO (Host Name) QUIT
MAILFROM
RCPTTO
DATA

## WWW (World Wide Web)

→ It is an "architectural framework" for accessing linked documents spread over millions of machines all over the internet. 'Tim Bunes Lee' → 1989.

• Architectural Overview:




### Client Side Programming:

→ From user's POV, the web consists of a vast world wide coll^n of docs (web pages) often just called as pages, each page contains link to other pages anywhere in the world
→ 1948 → "Vannevar Bush" → dev. concept to hypertinking web pages.
→ Pages are viewed with a prgrm called 'a browser'.
→ The browser fetches the page req. & displays the page properly formatted on screen.
→ Web Page contains Title & some info. & ends with email of the maintainer.
→ Hyperlinks are highlighted by a special color/underlined.

Date
Page

=> The URL has 3 parts : ① Protocol ② DNS Name
③ Name of the file containing the Page.

Eg: http://www.abcd.com/product.html
① ② ③

Client Side + Server Side + URL + Static Web Pg's + Dynamic Web Pg's
HTML, formi, XML, RSL

## RSA

↳ Rivest, Shamir, Adleman

1> Key Generation 2> Encryption 3> Decryption

### ① Key Generation

1> Choose 2 prime no. p & q. (p = 3, q = 11)

2> Finding value of n (n = p * q) = 33

3> Finding value of $\phi(n)$ = (p-1)(q-1)
= 20

4> Finding the value of e $\left[1 < e < \phi(n), \gcd(e, \phi(n)) = 1\right]$
Choose e such that it satisfies cond$^n$ → e = 3, 7...
e = 7

5> $d = e^{-1} \bmod \phi(n)$

$de \equiv 1 \bmod \phi(n)$

$d e \bmod \phi(n) = 1 \bmod \phi(n)$ (1 x mod of anything = 1)

$d * 7 \bmod (20) = 1$

6> Public Key = {e, n} = {7, 11}

7>

### ② Encryption

c = cypher text m = plain text.

$c = m^e \bmod n$

Let m = 31 $(m < n$ i.e., $m < 33)$

$c = 31^7 \bmod 33$

$c = 4$

$\boxed{c = 4}$

A → [ ] → B
m = 31 c = 4

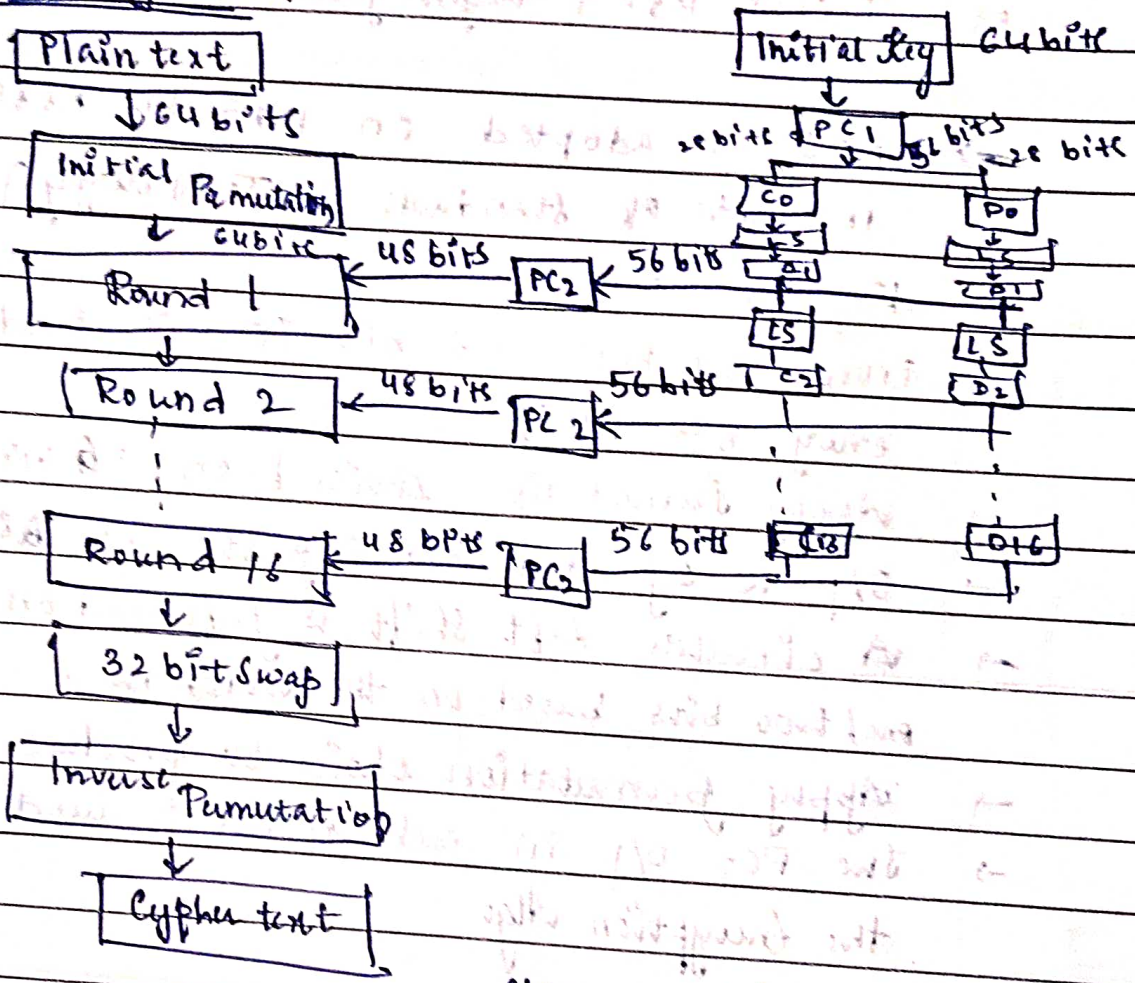
③ Decryption:

$m = c^d \bmod n = (4)^3 \bmod 33 \Rightarrow m = 3$

Adv: ① There is no sharing of key.
② More Security for data.

Dis: More slower than other algos.

DES Algorithm:




Algo.

→ DES is a symmetric Encryption which involves out combination substitutions & permutations. It is a Block Cyphering Tech. where, Block size = 64 bit, Key size = 64 bits, Number of rounds = 16.

→ 16 sub keys are generated from $K_1$ to $K_{16}$ which are used for 16 routes.

Algo Implementation

1) Divide data into Blocks of 64 bits
2) Perform Initial permutation on blocks.
3) Break Block into 2 parts. (L → Left Part & R → Right Part).
4) Permutation & substitution steps repeated 16 times.
5) Refer in left & Right parts. Perform Inital

→ DES is adopted in 1977 by NIST (National Institute of Standards & Technology)

Key Generation:

→ From 64 bits, 56 bit Key is produced (ignore every 8th bit).
→ Apply permutation choice 1 on 56 bits
→ O/p is seperated into 2 parts of 28 bits ($C_0$, $D_0$)
→ A circular Left shift is performed on $C_{i-1}$ & $D_{i-1}$ is one/two bits based on the round no.
→ Apply permutation choice to produce 48 bits.
→ The $PC_2$ o/p in each round is used as I/p to the Encryption Algo.

Algorithm:

1) The plain text of 64 bits will get pass through an initial permutation.
2) Permuted I/p split into 2 parts each of 32 bits (left Part (L), Right Part (R))
3) Expand 'R' 32 bit into 48 bits by performing Expansion Permutation (E)
4) Perform XOR b/w sub key & 'E'

5) The result will pass through substitution func. & produces 32-bit o/p's.

6) Combine the results of each S-box & the result will pass through a permutation func. 'P'.

7) Perform XOR of 'P' with $L_i - 1$

8) Put the result in $R_i$, $R_{i-1}$ in $L_i$

9) Continue the process for 16 times & then perform Swap func. & Inverse Initial Permutation to produce 64 bits Cypher text.