→ different Application /program running on different ports & services

5. Transport Layer (process-process delivery)

→ Heart of the OSI Layer System

→ process to process delivery System

→ Comes at 4th layer of OSI

→ provides Interface betn S/W & H/W

* process : one type of Application / program running called as process

* TL is responsible for deliving entire data to appropriate applicate process to one host to another host

→ 1) Service point Addressing : port number

→ 2) Sequentian & reerranging : provide sequence no msg

→ 3) Connection control : Connection oriented / Connectionless

→ 4) Flow control : Buffer mechanisim (speed. S→R)

→ 5) Error control : Error Control mechanism

* UDP : Connectionless, Unrelliable, faster

* TCP : Connection-oriented, reliable, slower ← (establish virtual path)

* SCTP : services (Stream CTP)

* (Accept data from session layer)

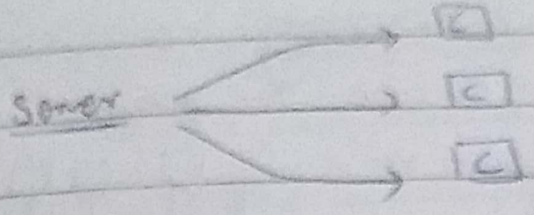
* manage end to end delivery of data

* no of application client server ⟶ * no of application

⟵ server side

* Different application using different services

*



1) Data Link layer : Frames are transported between node

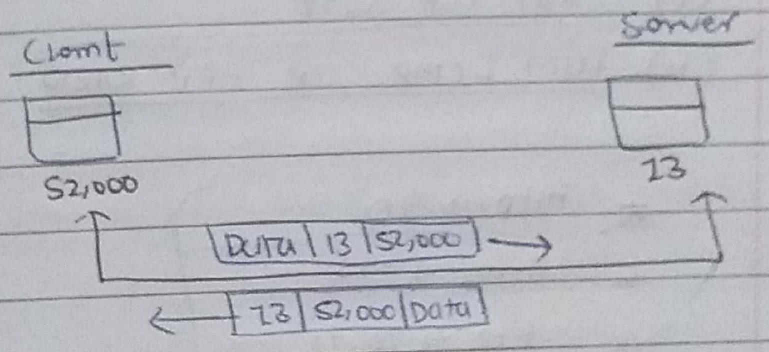
2) Network Layer : Packets are transported between host

3) Transport Layer : Process to process delivery

* Different types of Application using different processes

* Transport layer ⟶ port addresses

(0 to 65,535) (ports)



* Internet Assigned number Authority

0 – 1023 (wellknown)

1024–49151 (Registered)

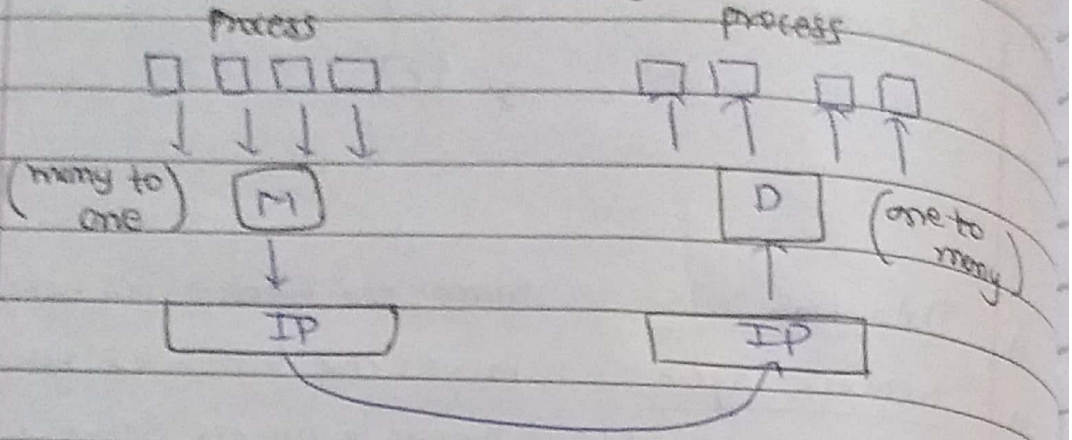
49152 - 65535 (Dynamic)

} Port ranges

* Socket address : combination of IP & port number

↳ 200.23.56.8 69

* Multiplexing & demultiplexing



* UDP → Unreliable / connectionless
* TCP → reliable / connection oriented

(A) SMTP FTP TFTP DNS SNMP

(T) UDP TCP SCTP

(N) IGMP ICMP IP ARP RARP

* Http => 80
* Https => 443
* Ftp => 20, 21
* DNS => 53
* SMTP => 25
* POP3 => 110
* IMAP => 143
* Telnet => 23
* SSH => 22

port numbers

* msg is independent in UDP (Connectionless service)
(Identity routes automatically)

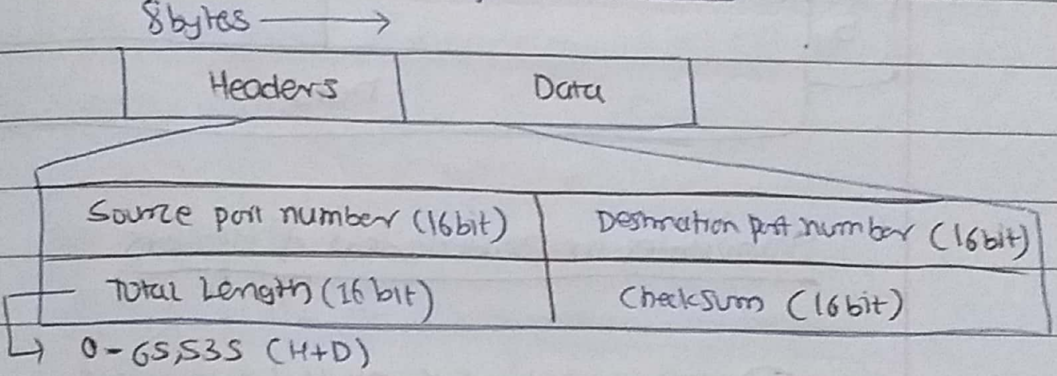
Page No.:
Date:

* Userdiagram Protocol (UDP) :

→ connectionless, unreliable transport protocol, It does not add anything to the services of IP except to provide process to process communication instead of host to host communication

→ well known ports used with UDP
(Echo, Discard, RPC, NTP, SNMP, TFTP)

* UDP format



8bytes →

Headers	Data

Source port number (16bit)	Destination port number (16bit)
Total Length (16 bit)	Checksum (16bit)

↳ 0-65,535 (H+D)

Source port & Destination

→ process → client → Epiphemal port → 49152

→ server → well known port → 0-1023

* process running on client & server

* Total Length of User datagram

UDPL ⇒ IPlength - IPheader Length

* Checksum ⇒ error detection

* Cheksum *

Pseudo headers	32 bit source IP Address		
	32 bit source to Destn. IP Add		
	All 0S	8-bit protocol	16 bit UDP length
Headers	Source port address	Destination port address	
	UDP total length	Checksum	
	Data		

Similarities in pseudoHeaders & Headers

① Take addition of all 16 bit numbers

Sum == chevksum

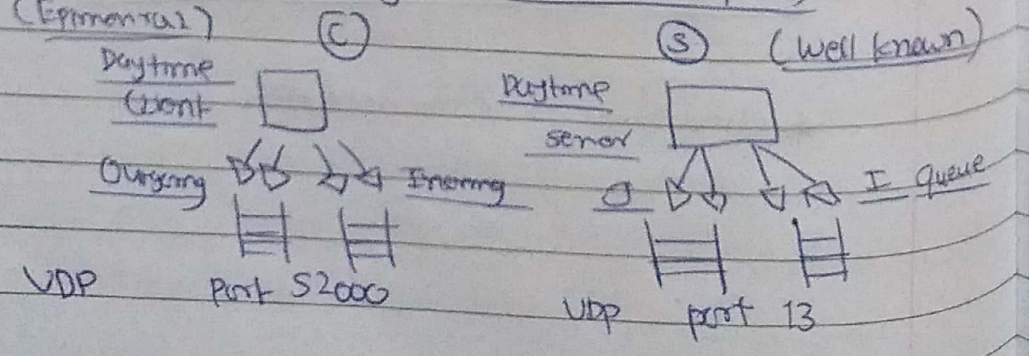
* UDP Operations →

i) Connection less services

ii) flow & error control

iii) Encapsulation & Decapsulation

iv) Queuing (Epimephal port)



Use OF UDP ⇒

* → UDP is suitable for a process that reaures Simple reauest & responmse Communication with Little concem for Flow & error control

→ It is not usually used for process such as FTP that needs to send bulk data

→ UDP is suitable transport protocol for multicasting

→ UDP is used for management process suchas the SNMP

→ UDP is suitable for process with interal Flow & error Control mechanismm (TFTP It can easily use UDP)

→ No need to create connection

→ Connectionless & Unreliable

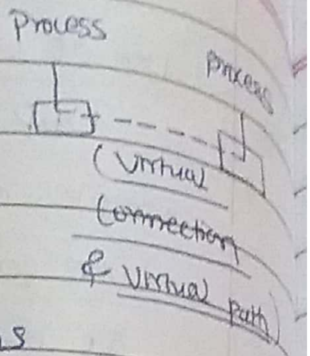
→ Dosent give gaunentee to deliver data

→ Process to Process communication

→ NO Flow & error control mechanism

23/11/20

* TCP *



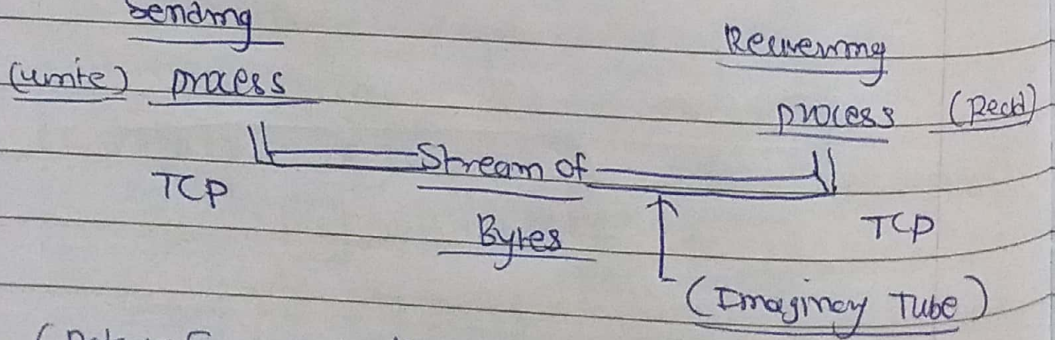
- → Transmission control protocol
- → Connection-oriented protocol
- → Reliable & requires Connections
- → Gives Gaurentee to deliver data
- → Process to process mechanism
- → Error & flow Control mechanism

→ It creates virtual connection between Two TCP to send data & also It uses flow & error control

→ Services ⇒

- * process to process
- * Stream Delivery
- * Sending & Recieving Buffers
- * Segments
- * Full Duplex Communication
- * Connection oriented services
- * Reliable services

→



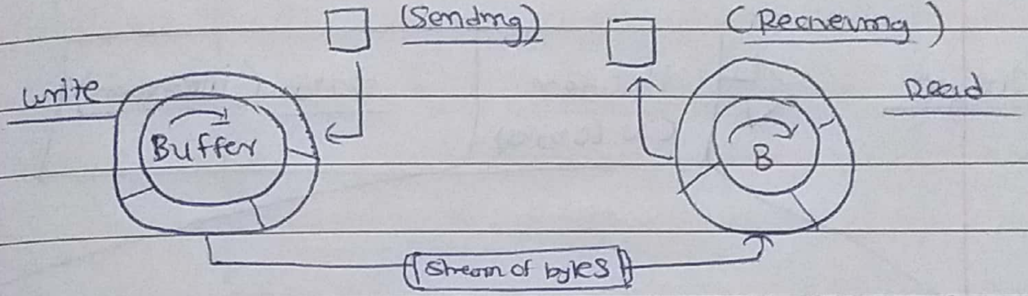
(Data is forwarded in the form of stream of bytes) through Imaginary tube

* → Needs Buffer for storage mechanisim process, Buffer has 3 Chambers : white ⇒ Empty Chambers

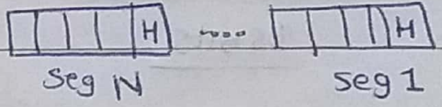
grey ⇒ Bytes which are sent

pink ⇒ Bytes which are not sent

* In Communication if Sender & reciever has not same speed to Communicate so they need buffer for storing data 2 Buffers one for sending & other for recieving → They uses circular Array of 100/1000 bytes each, direction of data is oneside.



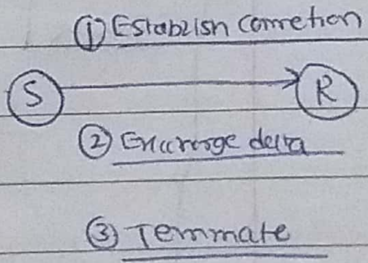
* Segments →



→ Group of Byres into packets → Called as segments

* full duplex Communication → At same time sender has send & Reciver has recieving data at a same time

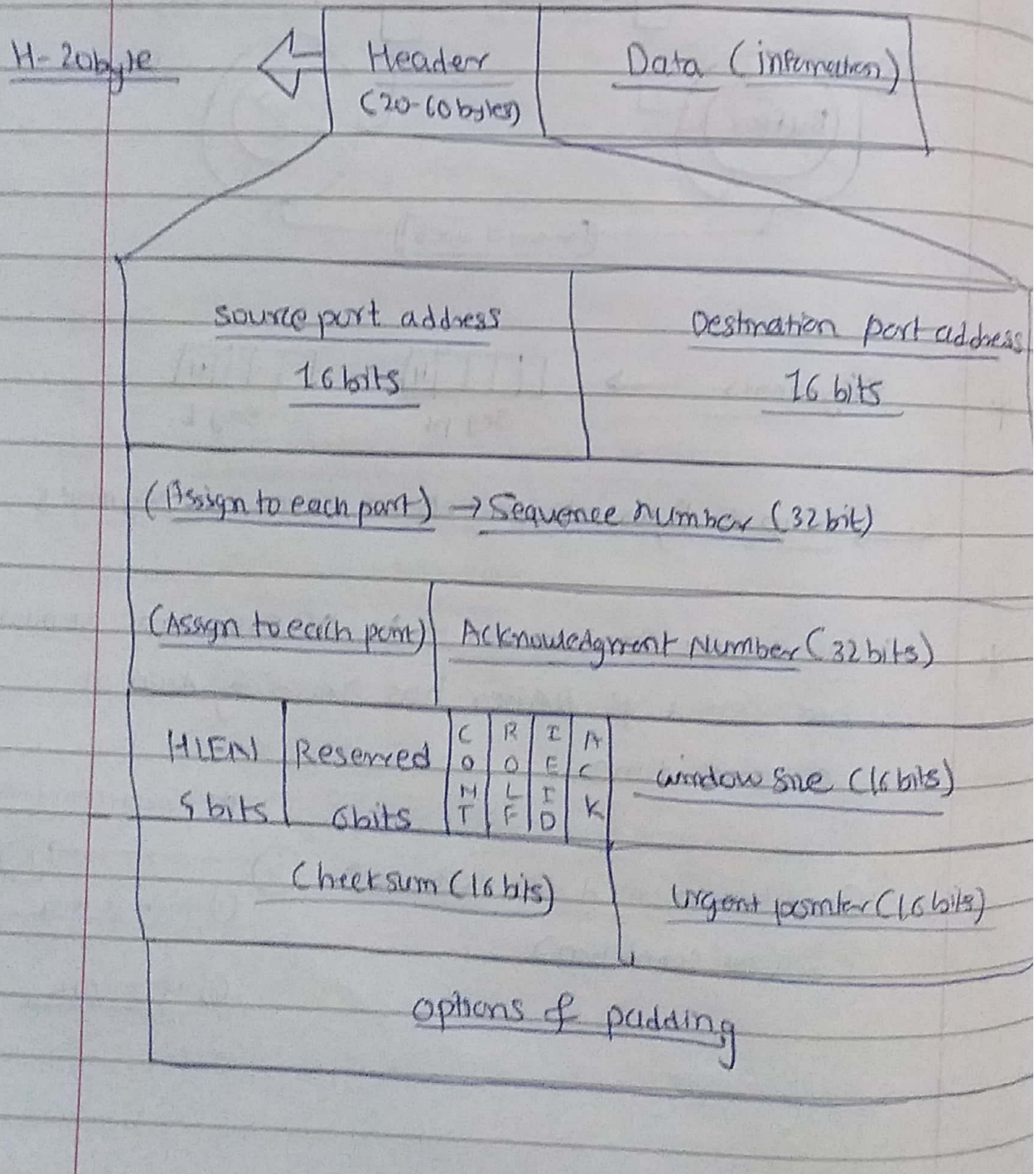
* Connection orinted services →
(virtual Connection)



* TCP Features *

- → Numbering System
- → Byte Number
- → sequence Number
- → Acknowledge Number
- → Flow control
- → Error control
- → Congestion control

* TCP Segment Format *



source port address 16 bits | Destination port address 16 bits

(Assign to each part) → Sequence number (32 bit)

(Assign to each part) | Acknowledgment Number (32 bits)

HLEN 5 bits | Reserved 6 bits | C O M T | R O L F | I E I D | A C K | window size (16 bits)

Checksum (16 bits) | Urgent pointer (16 bits)

options & padding

* TCP Connections :

→ 3 way Handshake protocol ⇒ (Passive open & active open)

↓ Client

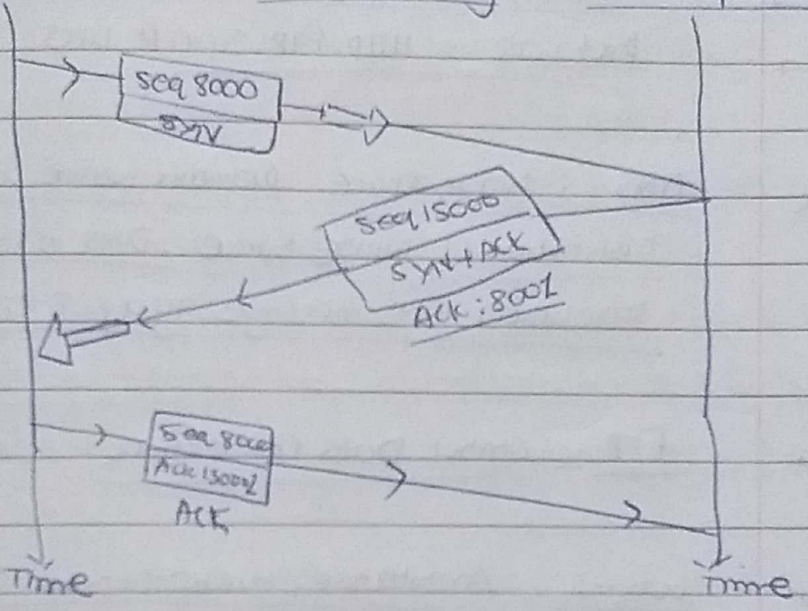
(I am ready to send request)

A : Ack Flag

S : SYN Flag

↓ Server

(I am ready to accept connection)



* Connection Establishment

* Data-transfer →

→ Server side it only sends one segment at a time