(16)

Computer Network

M=6

* 2nd - layer in TCP
* 4th - layer in OSI

PAGE NO.:
DATE: 11/08/22

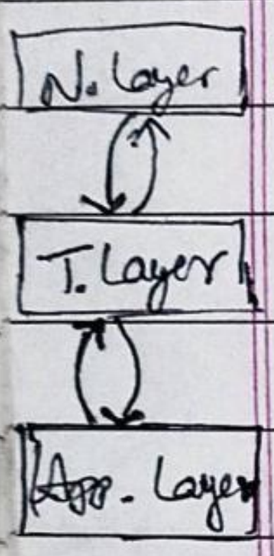
Transport layer

* Segmentation * Adding Sender & Receiver Port Num.

(1) Process to process :-

=> Transport layer is an end-to-end layer used to deliver message to a host. It is termed an end-to-end layer because it provides a point-to-point connection rather than hop-to-hop Between the source host & destination host to deliver the services reliably. The unit of Data encapsulation in the Transport layer is a segment.

Works :-



* Responsibilities OF a Transport Layer :-

(1) Process to Process Delivery :- While Data-link layer requires the MAC address (48 bit address OF NIC), of Source - Destination host to correctly deliver a frame and the N/W layer requires the IP add. for appropriate routing of packets, in a similar way. Transport Layer requires a port No. to correctly process amongst the multiple processes running on a particular host. A port Num is a 16-bit add. used to identify any client - server program uniquely.

(Port-Port delivery)

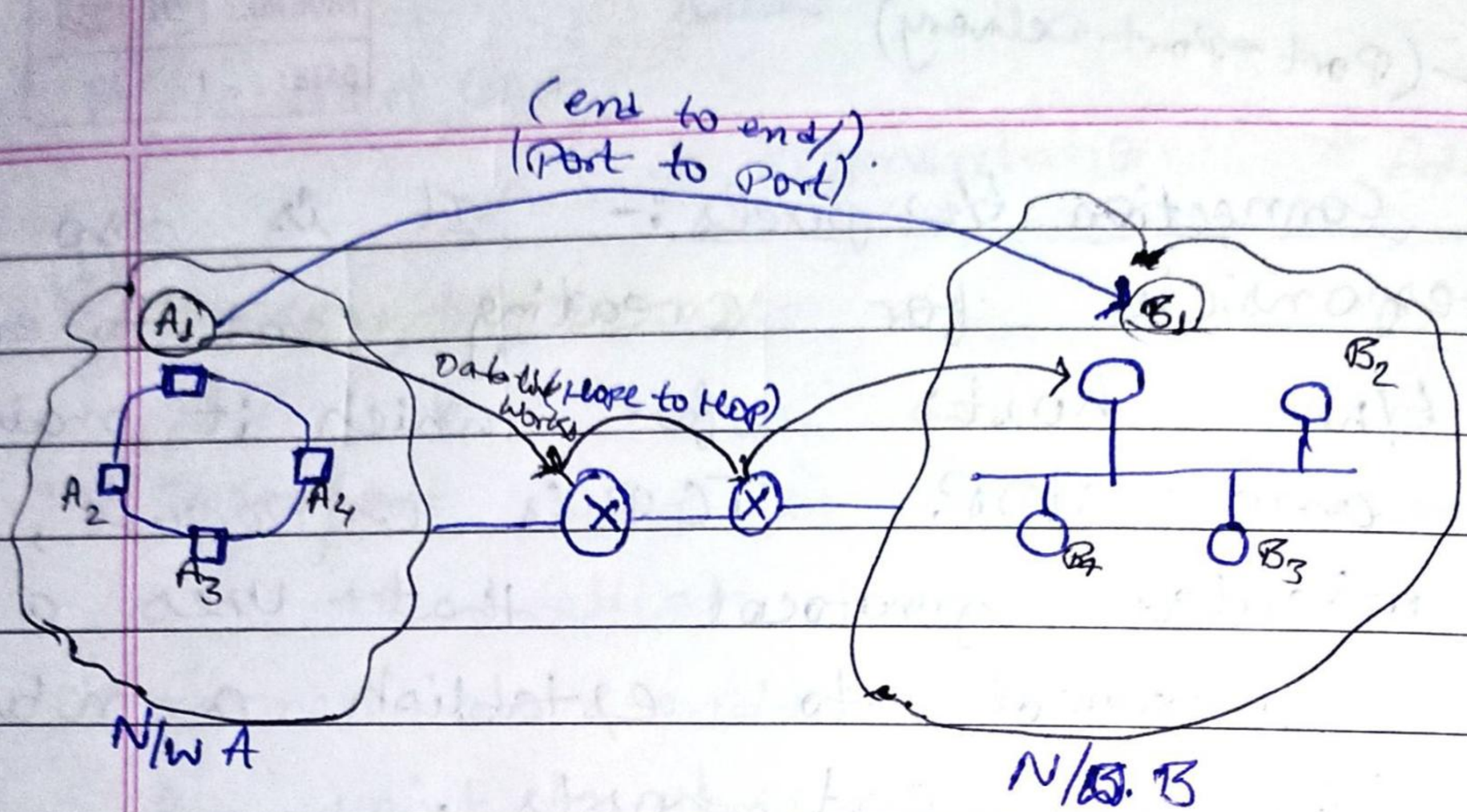
② End-to-End connection b/w pixels :- It is also responsible for creating end-to-end connection b/w hosts for which it mainly uses TCP and UDP. TCP is a secure, connection oriented protocol that uses a handshake protocol to establish a robust connection b/w two end hosts.

TCP ensure reliable delivery of message and is used in various applications that have little concern with flow or error control & requires sending the bulk of data like video conferencing. It often used in Multicasting protocol.

③ Multiplexing and Demultiplexing
④ Congestion control. (situation when large No. of Hosts start to share data)
⑤ Data Integrity and error Correction (ACK & NACK).
⑥ Flow control. (SW, SR).

▷ Protocol OF Transport layer :- • TCP • UDP • SCTP • DCCP • ATP • FCP • RDP • SST • SPX

Transport Layer - Role (Inorder delivery).



* No-loops of Data.

TCP (M-6)

** TCP header can be :- (160 bit to 420)

PAGE NO: 10
DATE: 11/08/22

* Transmission control protocol :- a communication standard that enables application programs and computing devices to exchange messages over a Network (n/w)

• It designed to send packets across the internet & ensure the successful delivery of data and messages over Network.

(TCP-Header).

Source Port (16bit)									Destination Port (16bit)
Sequence Number '32bit'									
Acknowledge No. '32bit'									
HLen 4bit	6 bit	U R G	A C K	P S H	R S T	S Y N	F I N	Window Size '16 bit'	
Checksum '16bit'									Urgent Pointer 16 bit
Options & Padding 40 (Bytes)									

Note **

* Header leng. = 4bit (20-60)

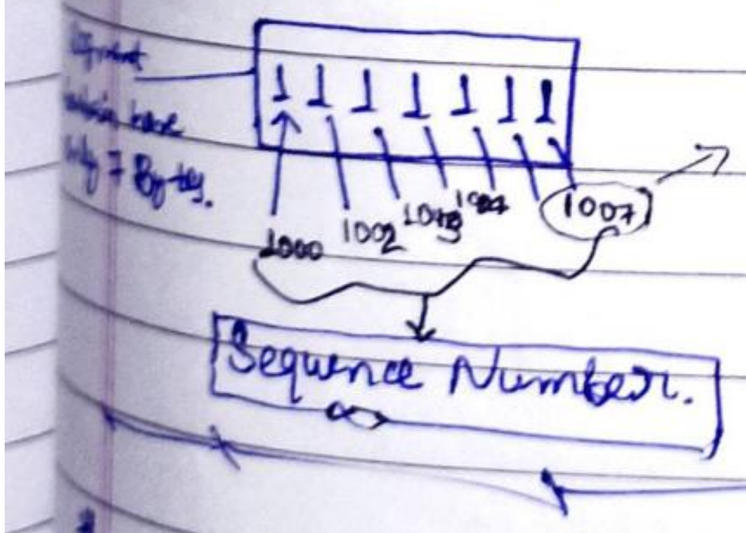
* Size of S/D Port No :-

2^{16} = 0-65535

where, 0-1023 is well standard Port No.

eg :- Port 80 for HTTP, Port 25 (4) SMTP.

* Segment = (Collection of Bytes)



* Header bit :- use* the scale of 4.

eg :- (1) (1111) × 4 = (60) max

(2) (0101) × 4 = (20) min limit

* Window Size :- Limit or capacity to receive or sends Data & also No. of Datas.

* Check Sum :- used to controll error.

* Urgent Pointer :- Defines range of Urgent Data/ormessage.

{ * option Padding :- (MSS) maximum Segment Size
* Ethernet MSS has - (1500B).

(20-60 Byte)

A.L

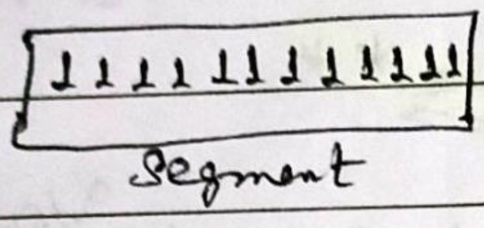
T.L

① Header | Data/msg.

Can be TCP/UDP

* Here, T.L attach a header to the Data. That can be TCP Header or UDP Header.

▷ TCP converts the continuous Data/msg in 'Bytes' then combine in 'Segments'.



▷ TCP Header must be Under (20-60) Byte i.e 20×8 = 160 bits

TCP (Features).

- Byte Streaming
- Connection oriented → Reliable
  - 3-way Handshaking
- Full Duplex → Both can send Data at a time
- Piggy backing → GBN, SR
- Error Control.
- Flow Control
- Congestion Control.

TCP Connection Establishment and Termination

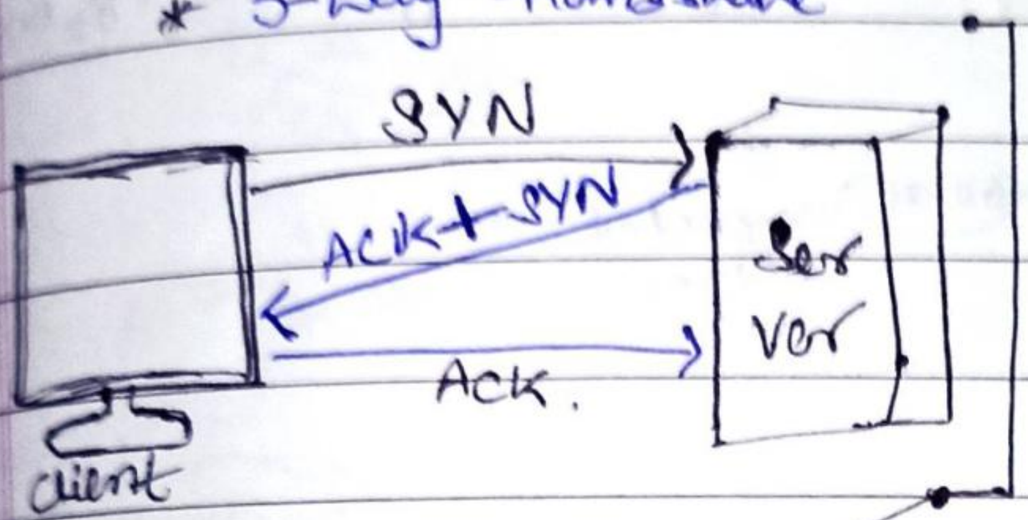
3 way

BW, CPU, Buffer

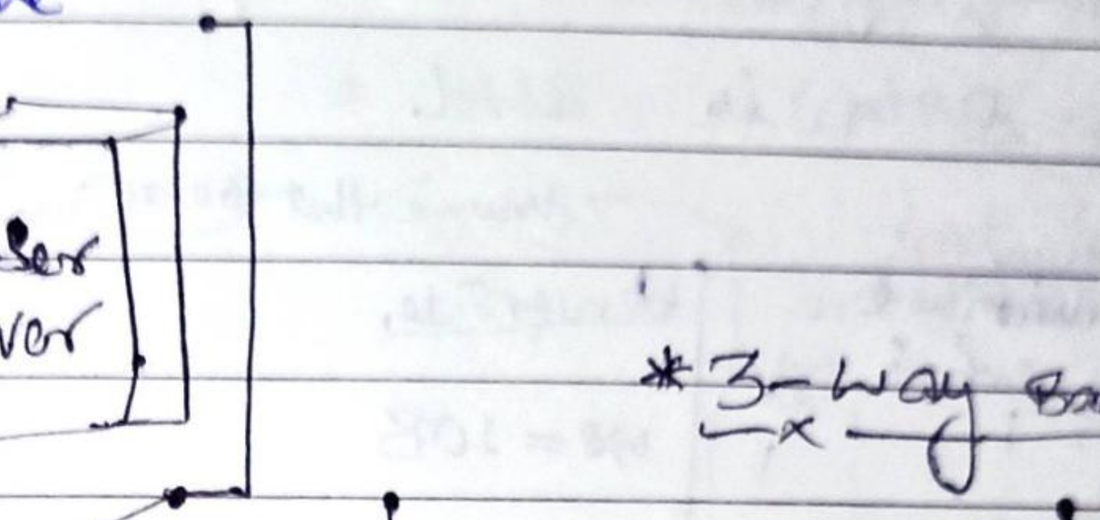
PAGE NO: 11
DATE: 11/09/22

* Random Port No. assigned by the Operating System
* Window sized used to control the flow of Data.
* SYN(1) means want to make connection (Synchronize).

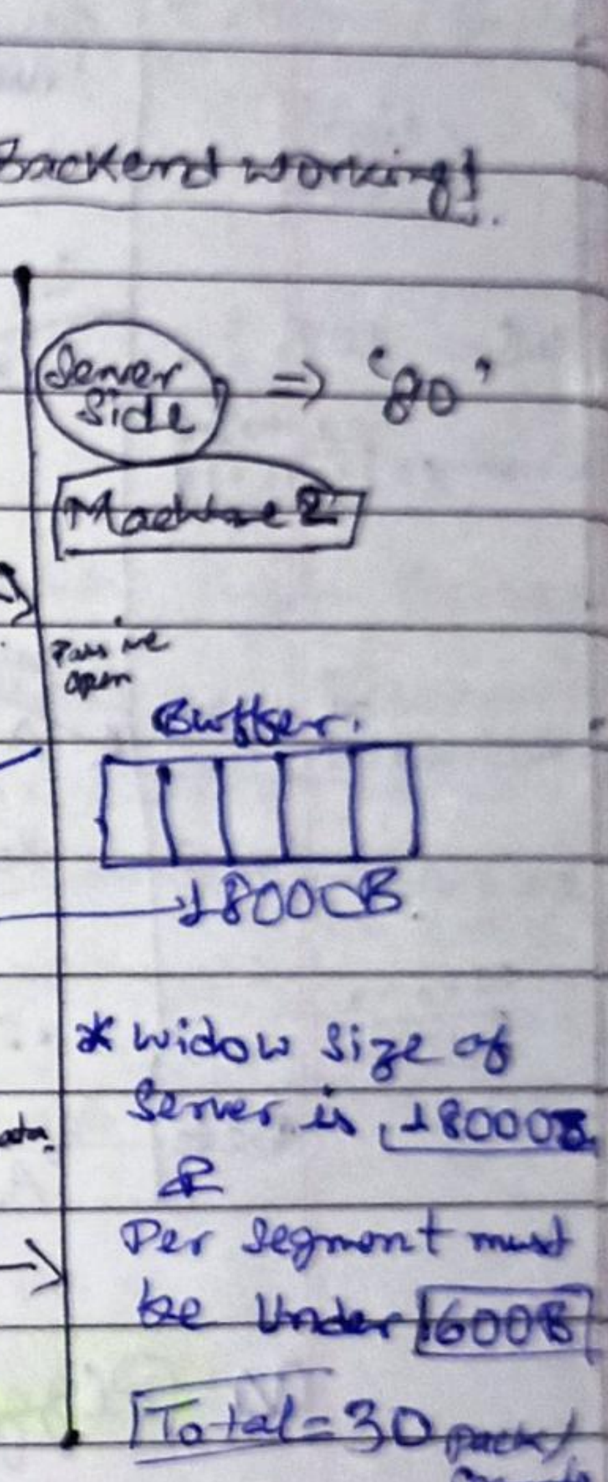
* 3-way - Handshake



* SYN flag is used first make/first time connection.
# ACK flag used '1' to acknoledge the (Port A+B).



* 3-way Backend working!



⇒ Positive ACK with Retransmission (PAR).
⇒ Protocol Data Unit (PDU) called as Segments.

M=6
CN

TCP - DATA Transfer

1970

A ⇄ B Full Duplex

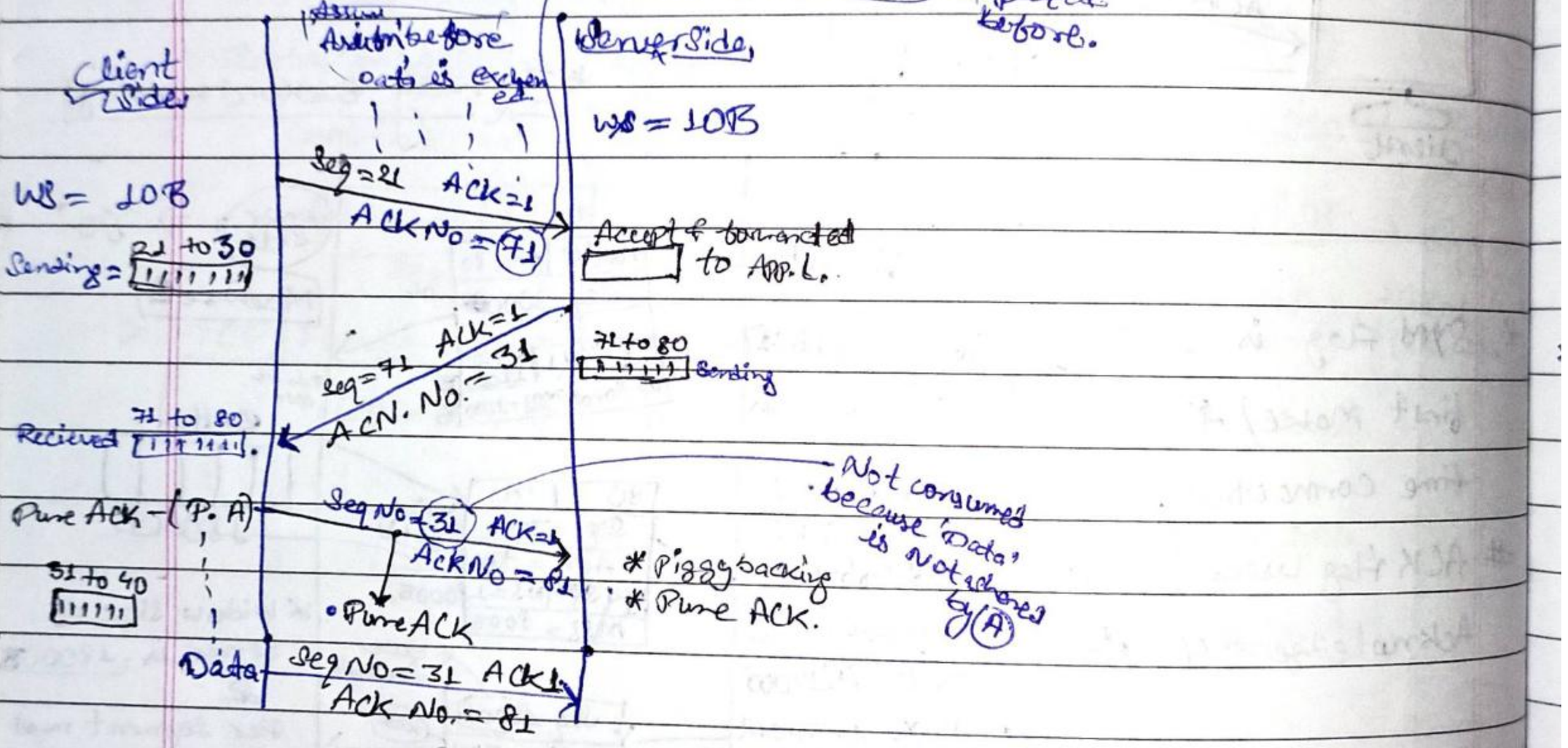
AFT 3-WAY connection

PAGE NO.:
DATE 10/08/22

▷ During connection establishment (b/w client & server) Both reserved the resources like, Bandwidth (BW), CPU, Buffer etc.

** ACK Flag is '1' ~~if~~ at every Sending / Receiving Data, if before Data is Sent.

Forex:
WS = 10B,



∞ Piggybacking :- This techniques used to reduce No. of packets Transmission in N/w.

• which the outgoing ACK is delayed temporarily is called Piggybacking

Piggybacking = (Data + ACK) — Both transfer in single packet/segment

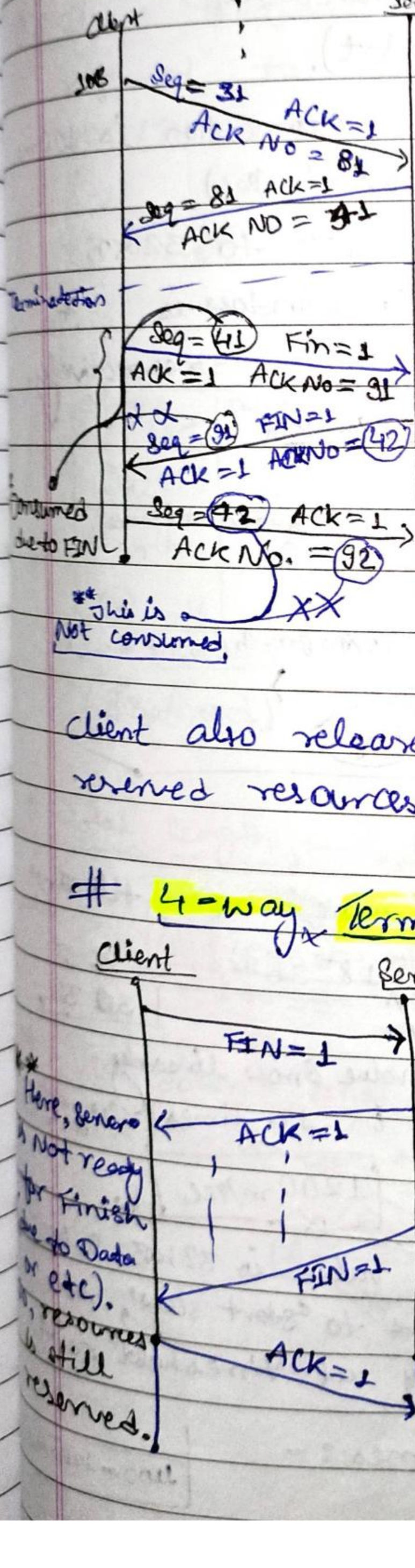
* It improve efficiency of the bi-directional protocol (Full-Duplex)

∞ Pure ACK :- when Host doesn't have data to send, so this time they 'Pure ACK' to confirm that last transaction is received. ACK.

TCP - Connection Termination - 3-4 steps

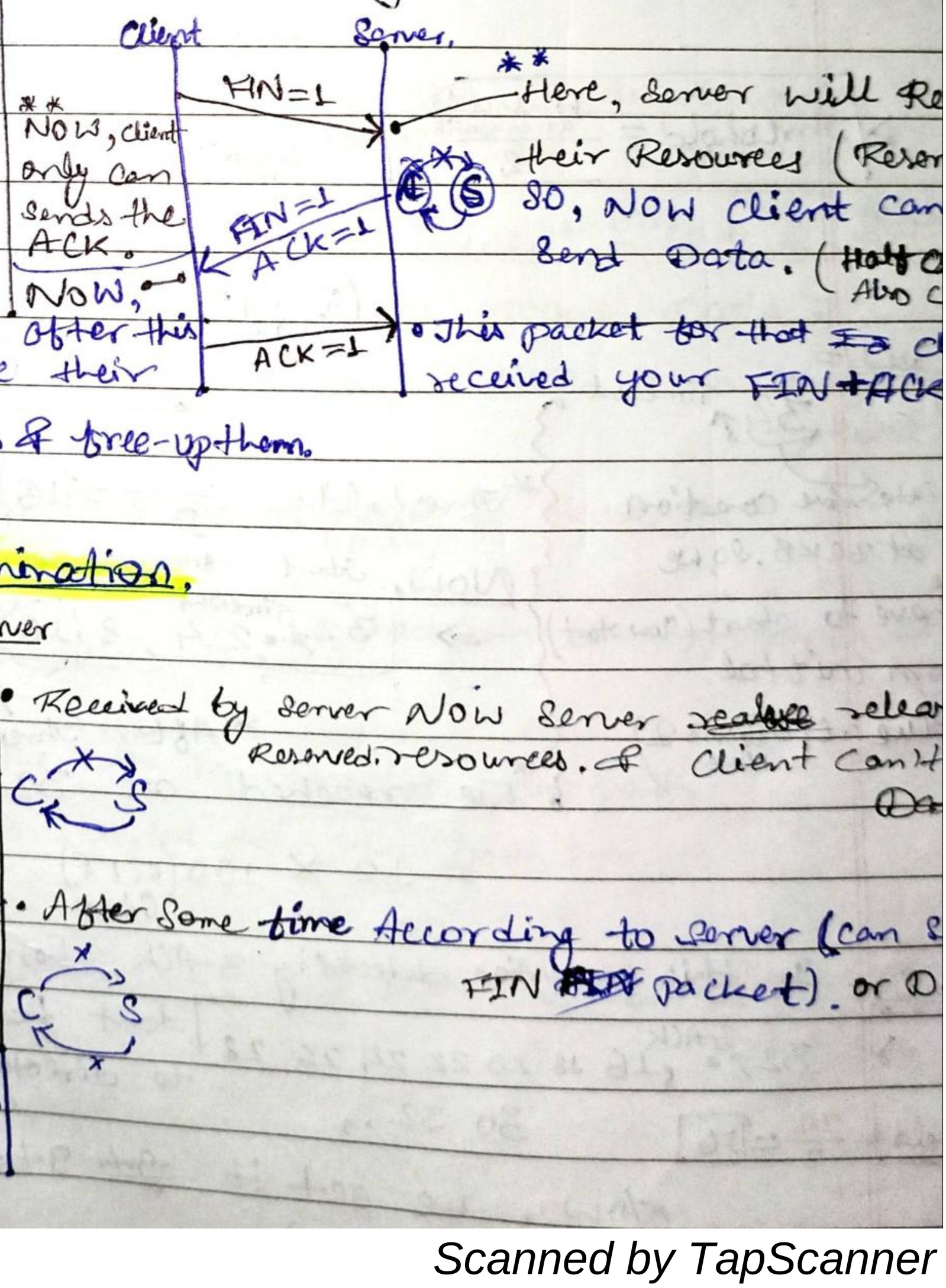
PAGE NO.: 22
DATE: 11 / 08 / 22

* 1. Can be FIN with Data. * 2. only FIN message



* FIN Flag (1) is always consumes the 'Sequence No' eg- FIN = 1 (FIN + ACK)
* ACK Flag (1) doesn't consume 'Sequence No.' but if it Alone/

3-way Termination :-



4-way Termination.

CN | Important

UDP - TCP Questions (congestion control)

Examples :-

** Let the size of congestion Window of a TCP connection in two cases where maximum segment size is 32 KB. The RTT of a connection is 100ms and MSS = 2KB. The time taken (mSec) by TCP - connection to get back to 32KB

Congestion window is ____ & ____ respectively.

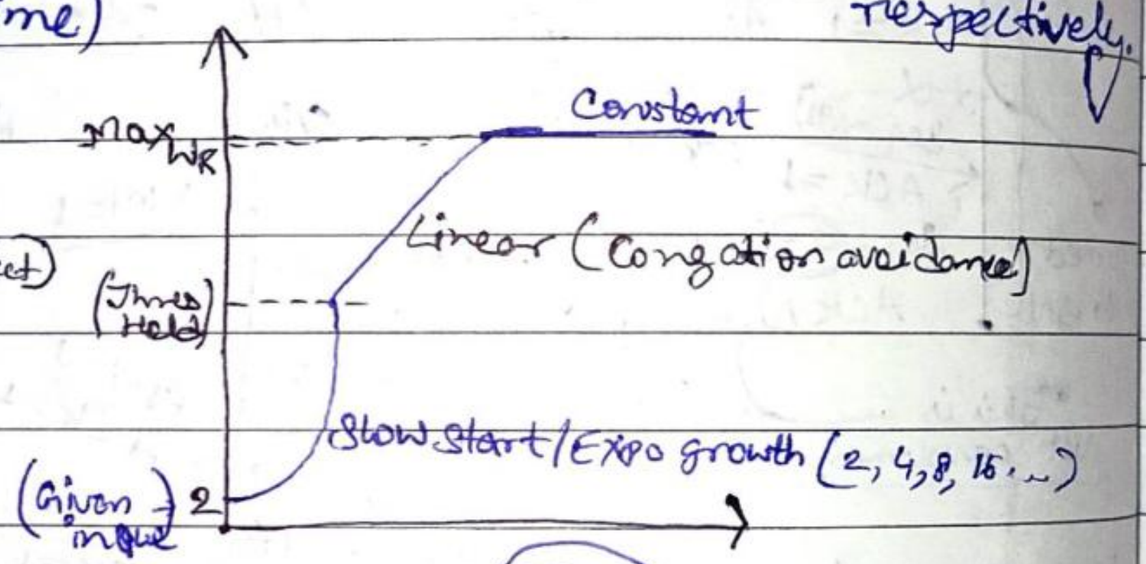
▲ Case 1: Timeout occure — (Severe) (Hard)

▲ Case 2: 3-ACK Received — (light).

Case 1: (Soln)

* RTT (Round Trip Time)

▷ Threshold = (Window size at congestion effect) / 2



** We-Determine congetion when at 32KB. So, we (32) have to start (Slow start) from initial value i.e (Given=2 Here)

* Threshold = 32/2 = 16

Now, start from 2 & move in expon. till Thresh value

⇒ 32 (Timeout) 2 4 8 16 +2 18 +2 20 +2 22 24 26 28 30 32.,

* After Thres value Grow linearly

* We reached at 32 in 12 times (Segments sents)

= 12 * 100 (RTT Given) = 1200 m/sec Ans.

Case 2:- In this congestion detected by 3-ACK when value is 32KB. So we dont have to 'start slow', move to directly at 'threshold' point

⇒ 32 (3-ACK) 16 18 20 22 24 26 28 30 32.,

Threshold = 32/2 = 16

Now, we get it 9 times :-

= 9 * 100 (RTT)

= 900 msec (miliSecond).

Ans

(800 - 900) Ans

Note if threshold is equal to [16.5] or [17.5] etc. So take it as Round [16] / [17]

[16] - [17]

1100 - 1200 Ans

CN/

Used by Transport Layer

UDP - User Datagram Protocol

PAGE NO.: 17
DATE: 11/08/22

* No-order (seq.) → * connection-less Protocol * Un-Reliable

It is a part of the internet protocol suite, reffered to as UDP/IP suite, reffered to as UDP/IP suite.

* It is an 'Un-reliable' & connectionless Protocol

So, There is not need of Establish a connection.

UDP

Source Port	Destination Port
(16) = 2^{16}	(16) = 2^{16}
Total Length	Checksum (IPV4)
(16) = 2^{16}	(16) = 2^{16}

Header (8 Byte fixed)

* Range of Port = $2^{16}-1$ = (0-65535)

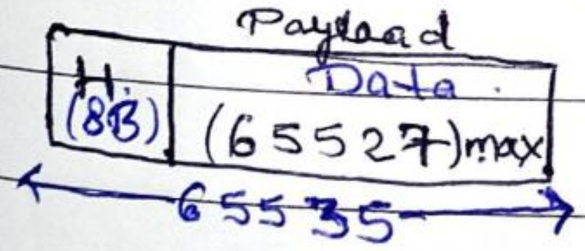
* checksum is used to detect/control the errors. checksum is optional in case IPv4

** Checksum = UDP Header + UDP Data + Pseudo header of IP

* payload means pure Data.

checksum is like Hash (Unique) value to detect error.

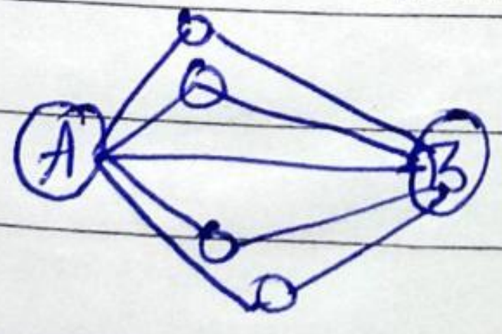
total length :- contains payload (Data) & Header):-



* checksum is mandatory in IPV6 only

UDP doesn't sends the data in same/on order path

Data can be transmitted from any path :-



▷ UDP is depends on IP & ICMP for error reporting.

CN 1

UDP ADVANTAGES :- Less overhead.

PAGE NO.:
DATE : 11/08/

Why UDP in use?

1) Query Responce (one request one reply) [DNS DHCP]
2) Speed (online games, Voice over IP)
Protocol.

3). Broadcasting / Multicast [RIP]

4) Continuous Streaming [YouTube, Skype].

5) State less

▷ Speed :- It makes useful for query-response protocols. Such as DNS, which Data packets are small & transactional.

* TCP	* UDP
1. Connection - oriented	1. connection less
2. Reliable (ordering)	2. Less Reliable (No-order)
3. Error control Mandatory	3. Error control is optional
4. Slow ~~transmission~~ transmission	4. ~~Error~~ Fast Transaction
5. More overhead.	5. Less overhead
6. Flow control, Congestion control.	6. No, FC, CC.

V = 6

TCP+UDP Jai Maa

RFC 2960

SCTP Congestion Control

2000

PAGE NO.:
DATE : 12/08/22

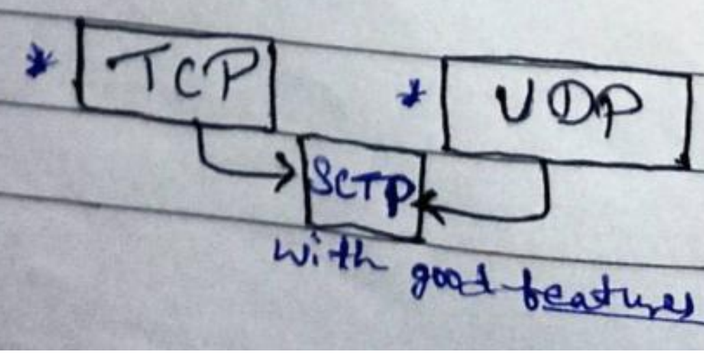
* Stream Control Transmission Protocol.

▷ It connection-oriented protocol with full-duplex capability.
▷ Multiple stream of data Transmission.
▷ Simultaneously, detect, lost/Duplicate & out of order Data.
▷ SCTP is also intended to make it easier to establish connection over wireless N/W & Managing transmission of Multimedia Data.

* **Characteristics:-**
1. Unicast with Multiple pro.
2. Reliable Transmission
3. Message-Oriented
4. Multi-homing (It can establish multiple connection Path b/w two end point)

ADVANT:-
- It is full-duplex
- It allows Half closed connection (A → B)
- ** It has both properties 'TCP' & 'UDP'.
- It doesn't rely on IP layer for resilience OF paths.

D-ADVAN:-
1. One of key challenges is that it requires changes in Transport Stack on Node.
2. Applications need to be modified to use SCTP instead of TCP/UDP.
3. Application need to be ~~multi~~ modified to handle multiple simultaneous streams.



CN

Quality OF Services

PAGE NO.:
DATE: 12/ 08

⇒ It is use of machanisms or technologies that work on a Network to control traffic and ensure the performance of critical applications with limited Network capacity. It enables organizations to adjust their overall Network Traffic by prioritizing Specific high - performance application.

NEEDS OF QOS? • Video & Audio Conferencing require bounded relay and loss rate.

• Time - critical applications (real - time control) in which bounded delay is considered to be an important factor.

• valuable applications should be provided better services than less valuable applications.

* **QOS Specification :-** Requirements :- 1. Delay 2. Error Rate 3. Delay Variation (Jitter) 4. Throughput

* **Types of N/w traffic :-** • Bandwidth • Delay • LOSS • JITter.

▷ **QOS Solution :-** 1. Stateless 2. Stateful.

1. Stateless sol^n :- Routers maintain No fine grained State about traffic, one positive factors of it is that it is Scalable and robust. But it has weak services. as there is no gurantee about the kind of delay or performance in a particular application which we have to encounter

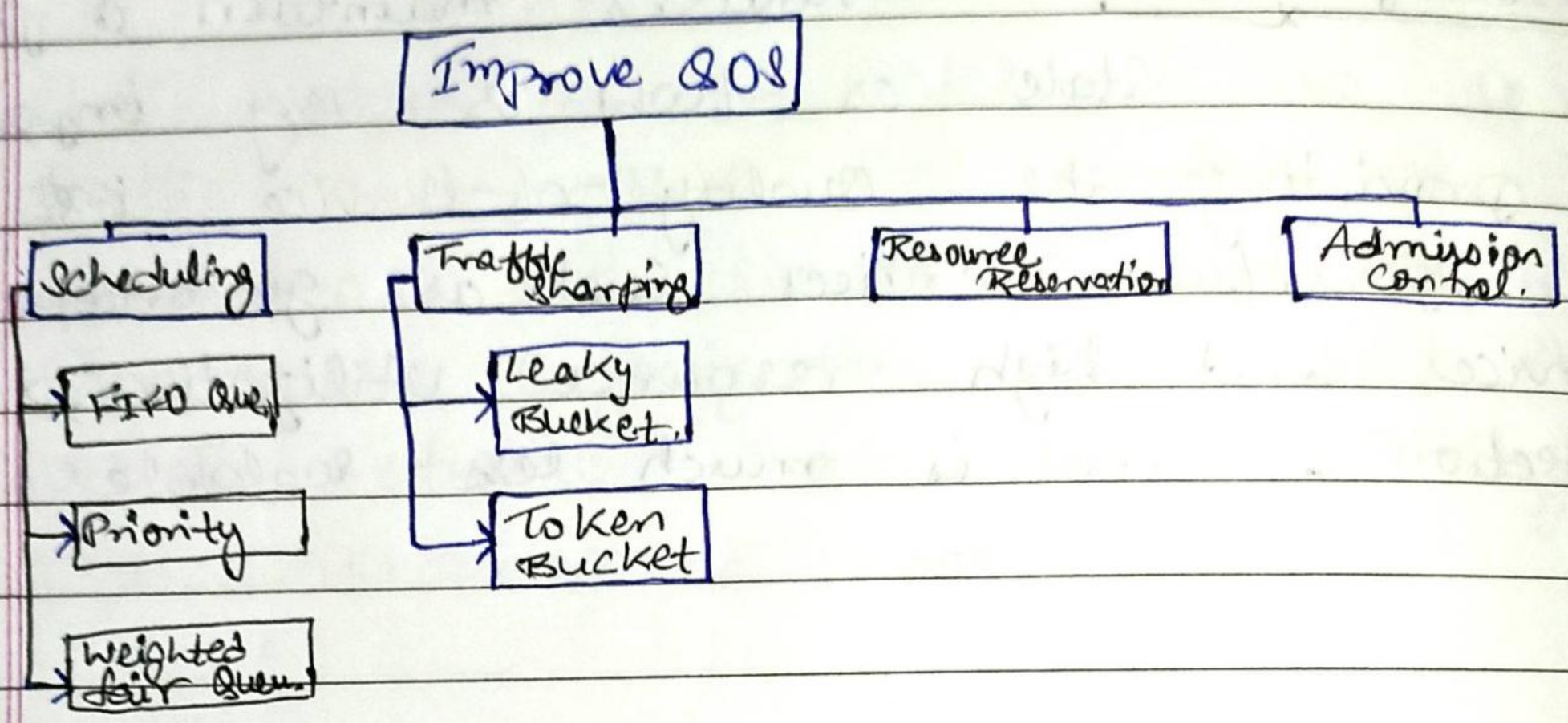
2. Stateful SDN :- Routers maintain a per-flow state as flow is very important in providing the Quality-of-Service. i.e providing powerful services such as guranteed services and high resource utilization, providing protection, and is much less scalable and robust

CN

QOS x Improving : Leaky Bucket & Token Algo.

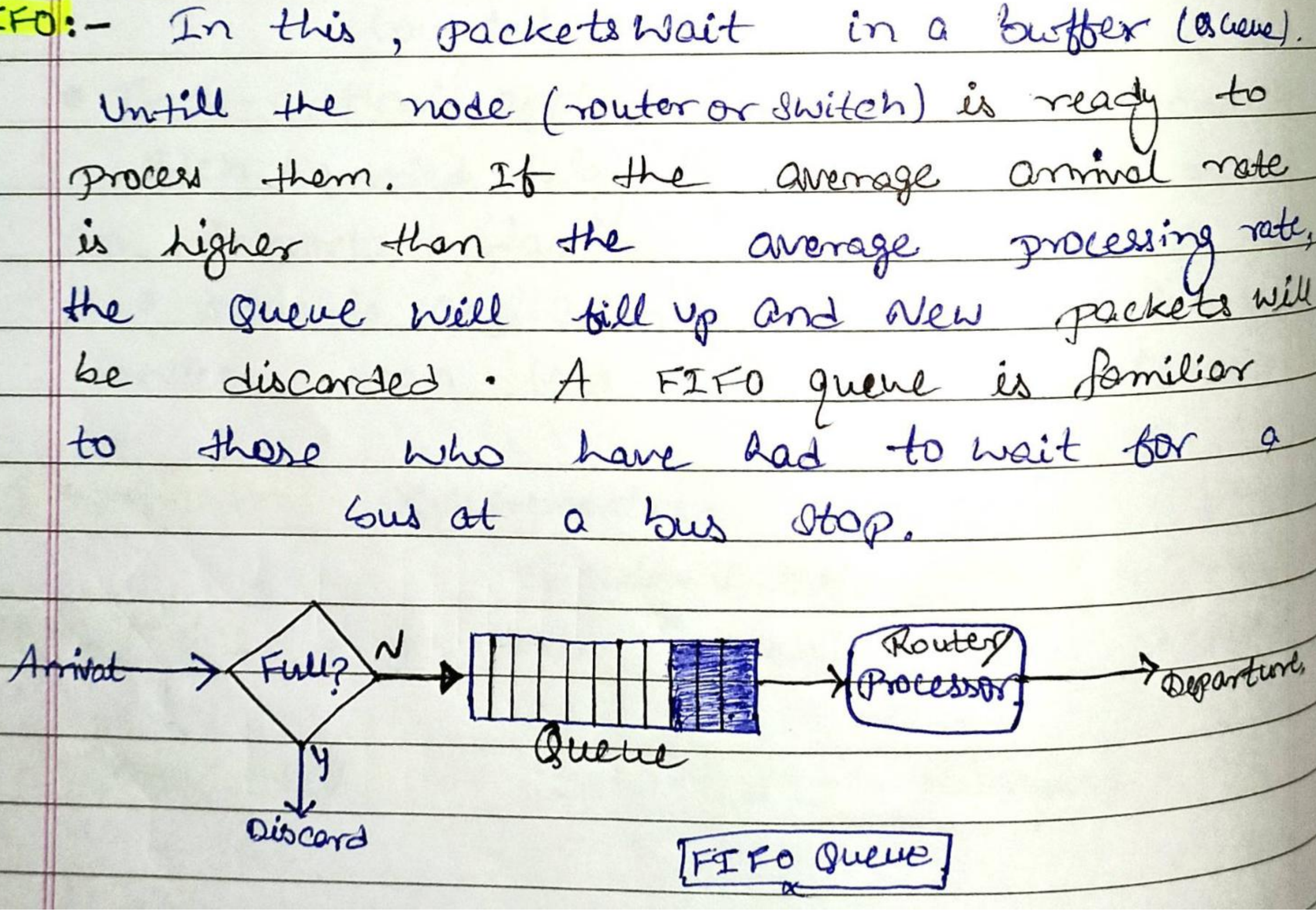
(Techniques)

PAGE NO.:
DATE : 12/08/20



1.

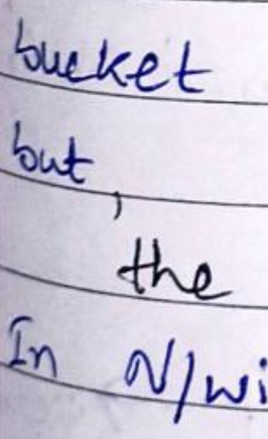
* **FIFO:-** In this, packets wait in a buffer (queue). Untill the node (router or switch) is ready to process them. If the average arrival rate is higher than the average processing rate, the queue will till up and new packets will be discarded. A FIFO queue is familiar to those who have had to wait for a bus at a bus stop.

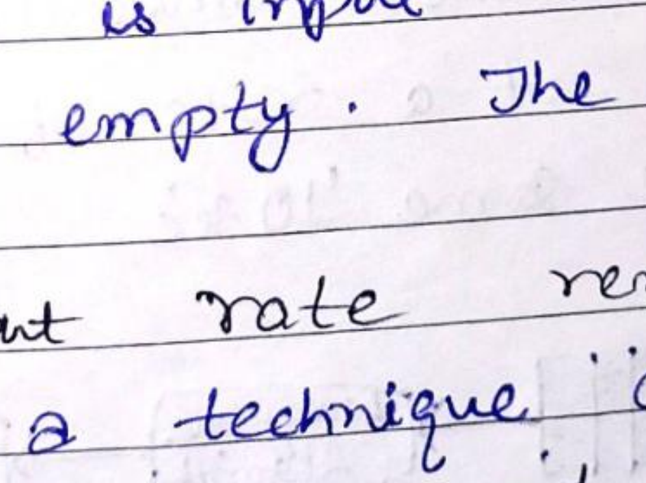


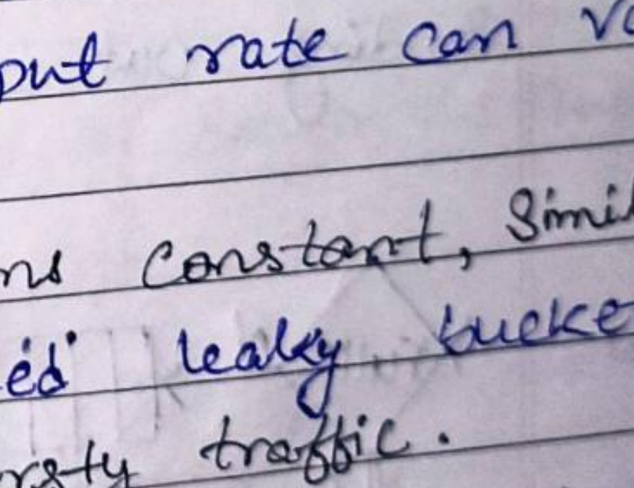
* Priority Queue * Weighted Fair Queue

Traffic Shaping :- It is a mechanism to control the amount and the rate of the traffic sent of the Network. Two Techniques can shape Traffic : Leaky and Token bucket.

* Leaky Bucket :- It a bucket has a small hole at the bottom, the water leaks from the bucket at a constant rate as long as there is water in the bucket. The rate at which the water is input to the bucket unless the bucket is empty. The input rate can vary, but, the output rate remains constant, Similarly In N/wing, a technique called leaky bucket can smooth out bursty traffic. Bursty chunks are stored in the bucket and sent out at an average rate







* Packet Discard :

Inlast page: We assume, the N/W has commited a bandwidth of '3Mbps for a host. The use of the leaky bucket shapes the input traffic to make it conform to this commitment.

In fig. - Host sends a burst of Data at a rate of '12 Mbps' for '2' sec. total of 24 Mbits of Data. The host is ~~this~~ silent For '5 sec.' & then sends Data at a rate of '2 Mbps' for '3" sec.

total = $\boxed{3 \times 2 = 6}$ Mbits. OF Data

* The host has sent ~~total~~ 30 Mbits of Data in '10' sec.

** SO, The leaky bucket smooth's the traffic by sending out data at a rate of '3 Mbps' during the same '10 sec'.

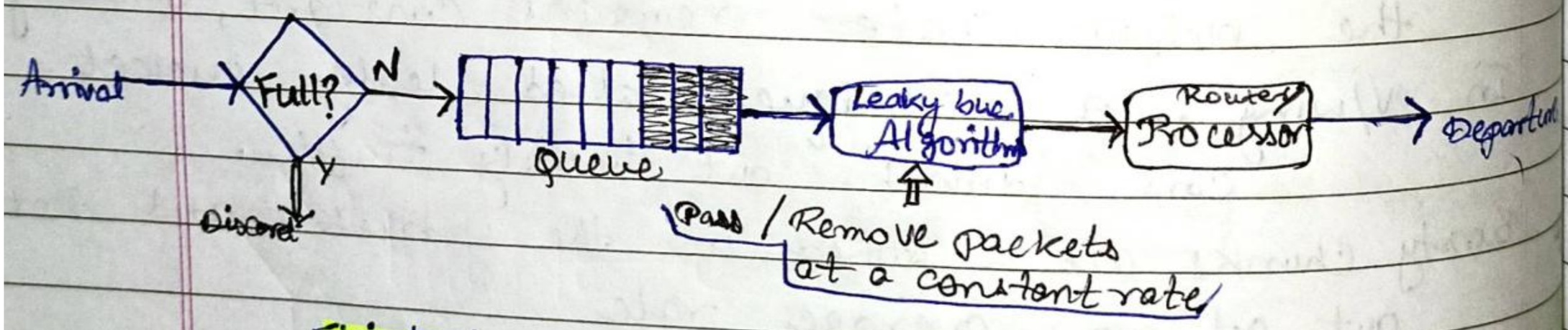


Fig:- Leaky bucket Implementation

* In above implementation, A FIFO queue holds the packets (e.g., cells in ATM N/Ws), the process removes a fixed number of packets from the queue at each tick of the clock. If the traffic consists of variable-length packets, the fixed output rate must be based on the No. of Bytes for bits.

Leaky Bucket Algo.

PAGE NO.: 10
DATE: 12/08/21

* Algorithm for variable length packet :-

1. Initialize a counter to n at the tick of the clock.
2. If n > is greater than the size of the packet send the packet and decrement the counter by the packet of size.
3. Repeat this ↑ step Until n is smaller than the packet size.
4. Reset the counter & go to step 1.

** Note :- A leaky bucket Algo. shapes bursty traffic into fixed-rate traffic by Averaging the Data rate.
• It may drop the packets if the bucket is ful

Now, (n < size of the packet at the head of the Queue),
i.e. - [n < 450].

** Therefore, The procedure is stopped.
Then, Initilize (n=1000) on another tick of the clock.
This procedure repeated Until all the packets are se

* Examples :-
Let 'n = 1000' ; packet = [200 | 700 | 500 | 450 | 400 | 2

① Since n > size of the packet at the head of the Queue,
= i.e. [n > 200]
Therefore, n = [1000 - 200 = 800]
Packet size 200 is sent into the N/w.

② ∴ n = [Current 'n'] - [Current Head Queue] = New('n').
Now, again
n > size of the packet at the head of queue, i.e., [n > 400]
Therefore, n = [800 - 400] = 400
So, Packet is sent into N/w.

CN-6

Token Bucket

PAGE NO.:
DATE : 12 / 08 /

* The Leaky bucket is very restrictive. It doesn't Credit an idle host is not sending for a while, its bucket becomes empty.

Now if the host has ~~was~~ bursty Data, the leaky bucket allows only an average rate. The time when the host was idle is not taken into account.

But, in Token Algo. allows idle hosts to Accumulate Credit for the future in the form of 'tokens'

जमा करना

For, ~~eva~~ each tick of clock, ~~the~~ System removes 'One token for every cell(byte) of Data sent.

⇒ If ('n' = 100) and host is idle for '100 ticks', the bucket collects 100 x 100 = 10000 tokens

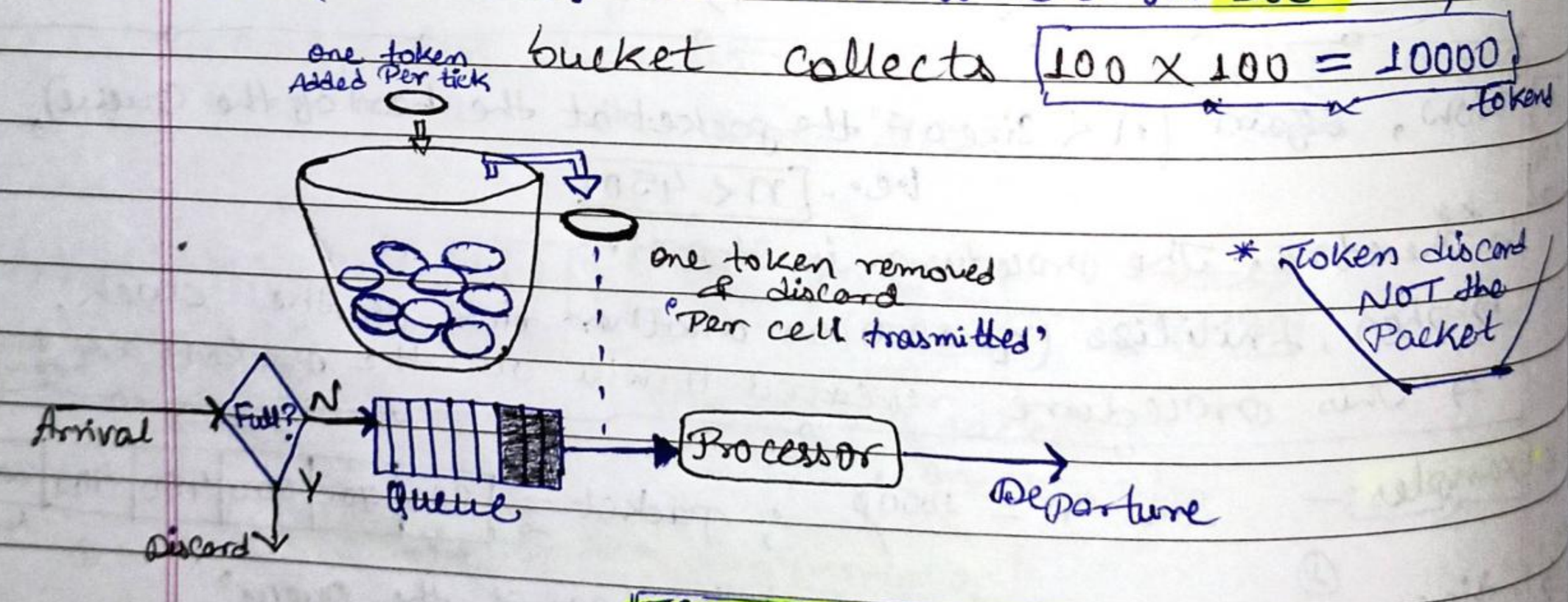


Fig.- Token Bucket

∞ The token bucket can easily be implemented with a counter.

The token is initialized ~~at~~ to zero(0).

∞ Each time a token is added, the counter is incremented by 1.

PAGE NO. : 19
DATE : / /

Each time a Unit of Data is sent, the counter is decremented by 1. When the ~~con~~ counter is zero, the host cannot send data.

Note :- The token bucket allows bursty Traffic at a regulated Maximum rate.

Combining "Token bucket and Leaky bucket"

↳ To Credit an idle host and at the same time regulate the traffic.

⇒ The Leaky bucket is applied after Token bucket; the rate of the bucket needs to be higher than the rate of token dropped in bucket.