Transport Layer

chapter 3

- Transport Layer is located between application layer and network Layer. (fig 3.1, Slide no 3.5
- It provides a process to process communication between two application layer.

Transport Layer Services:

(1) Process-to-process communication:

- A process in application layer is an program that uses the services of transport Layer
- Transport layer protocol is responsible for delivery of the message to appropriate process.
- Fig. 3.2 (Slide no 3.10

(2) Addressing:

- To define a process, identifiers are needed called as port numbers.
- In TCP/IP protocol suite, port numbers are bet^n 0–65535.
- Total 2^{16} number of ports exists.
- Client has ephemeral ports (short lived)
- Ephemeral ports are greaterthan 1023.
- In TCP/IP universal port numbers are used by server called as well-known port numbers.
- ICANN: Internet corporation for assigned names and numbers
- ICANN divide port numbers into three range
  - well known port number
  - Registered " "
  - Dynamic " "
- fig. 3.5 (slide no 3.13

(3) Encapsulation and Decapsulation

- fig. 3.7 (Slide no 3.16
- Transport layer protocol encapsulate and decapsula the messages.
- Encapsulation happen at sender site.

- Decapsulation happen at reciever site.

(4) Multiplexing and demultiplexing:

- An entity accept items from more than one source is called multiplexing (many to one)
- An entity deliver items to more than one source is called demultiplexing. (one to many)
- fig. 3.8 (Slide no 3.17
- Multiplexing happens at source and demultiplexing happens at destination.

(5) flow control

- When an entity produces items, and another entity consumes them, there should be balance between production and consumption rates.
- If items produce faster, consumer will be overwhelmed.
- " " " slower, " " wait and become less efficient.
- Fig. 3.9 (Slide no 3.18
- Delivery of item from producer to consumer is in 2 ways-
  - Pushing
  - Pulling
- If sender deliver without request from reciever it is pushing.
- If sender deliver item after request from reciever, it is pulling.
- Fig. 3.10 (Slide no 3.19
- Four entities are involved in transport layer communication:
  (i) Sender process
  (ii) " transport layer
  (iii) Reciever " "
  (iv) " process.

- Sending procen at applicat^n layer produces message and push to transport layer.
- Sending transport layer is both producer and consumer.
- Sending tranport layer consumes the menage push by applicat^n ".
- Sending transport layer encapsulate the menage in packet at push to recieving transport layer.
- Recieving transport layer is also both producer and consremer.
- flow control in transport layer occur from sending transport layer to sending application layer and from recieving transport layer to sending transport layer.
- flow control is implemented using 2 bubbers:
  - One at sending transport layer
  - Other " recieving " "
- Bubber is a set of memory locations that can ~~send~~ hold packet at sender and reciever.
- when bubber of sending transport layer is full, it inform applicat^n layer to slow down.
- when bubber of recieving ~~applicat~~ transport^n layer is full, it inforns the sending tranport layer to slow down.

6) Error Control:

- Transport layer needs to be reliable because underlying protocol IP is unreliable
- Reliablity can be achieved through error control
- Error control at transport layer includes:
  i. Detecting and discarding corrupted packet

ii. Keeping track of lost and discarded packets.
iii. Recognizing duplicate packets and discard them.
iv. Buffering out of order packet until the missing packet arrive.

- Error control involves sending and recieving transport layer.
- Sequence number:
  - Sending transport layer should know which packet to resent
  - Recieving " " " " " "
  are duplicate or out-of-order.
  - So packets are need to be numbered called as sequence number.
  - Transport layer header has a field for giving sequence number i.e. m bits
  - So sequence numbers range from 0 to $(2^m - 1)$.
  - Sequence numbers can be wrap around.
  - " " are modulo 2^m.
- ACKnowledgement number:
  - Reciever send ACK after recieving a packet.
  - If ACK does not arrive before timer expire, sender will resent the packet.
  - Duplicate packets are discarded
  - Out-of-order packet either stored or discarded.
  - Sliding window - imaginary window that contain packets.
- Fig. 3.13 (Slide no - 3.23
- window slides when ACK of a packet recieved.

7) Congestion Control:

- Congestion in a network occur if the load of the n/w is greater than capacity of the n/w.
- Congestion control refer to the mechanism and technique

that control the congestion and keep the load below the capacity of the network.
- If router cannot process the packet at the same rate at which they arrive the queue become overloaded and congestion occurs.
- Congestion at transport layer is due to congestion at network layer which manifest to transport layer.

(8) Connectionless and connection oriented service
- Transport layer provide two types of services:
  - Connectionless
  - Connection oriented
- In connectionless service, packets are created and delivered
- " " " , " " treated as single unit.
- Fig. 3.14 (slide no 3.24
- In connection oriented service, client and server first establish a logical connection
- Then data transfer will occur.
- fig. 3.15 (slide no 3.25

Transport Layer Protocols:

- There are 4 protocols designed for transport layer:
  1. Simple protocol → Unreliable protocol
  2. Stop and wait protocol
  3. GO-BACK-N "
  4. Selective Repeat "

  } Reliable data transfer (RDT) protocols

(1) Simple Protocol:

- Its a simple connectionless protocol
- It does not provide flow control or error control
- Fig. 3.17 (slide no
- Reciever immidiatly handle the packet it recieve
- Transport layer at sender and reciever provide serice to application layer.

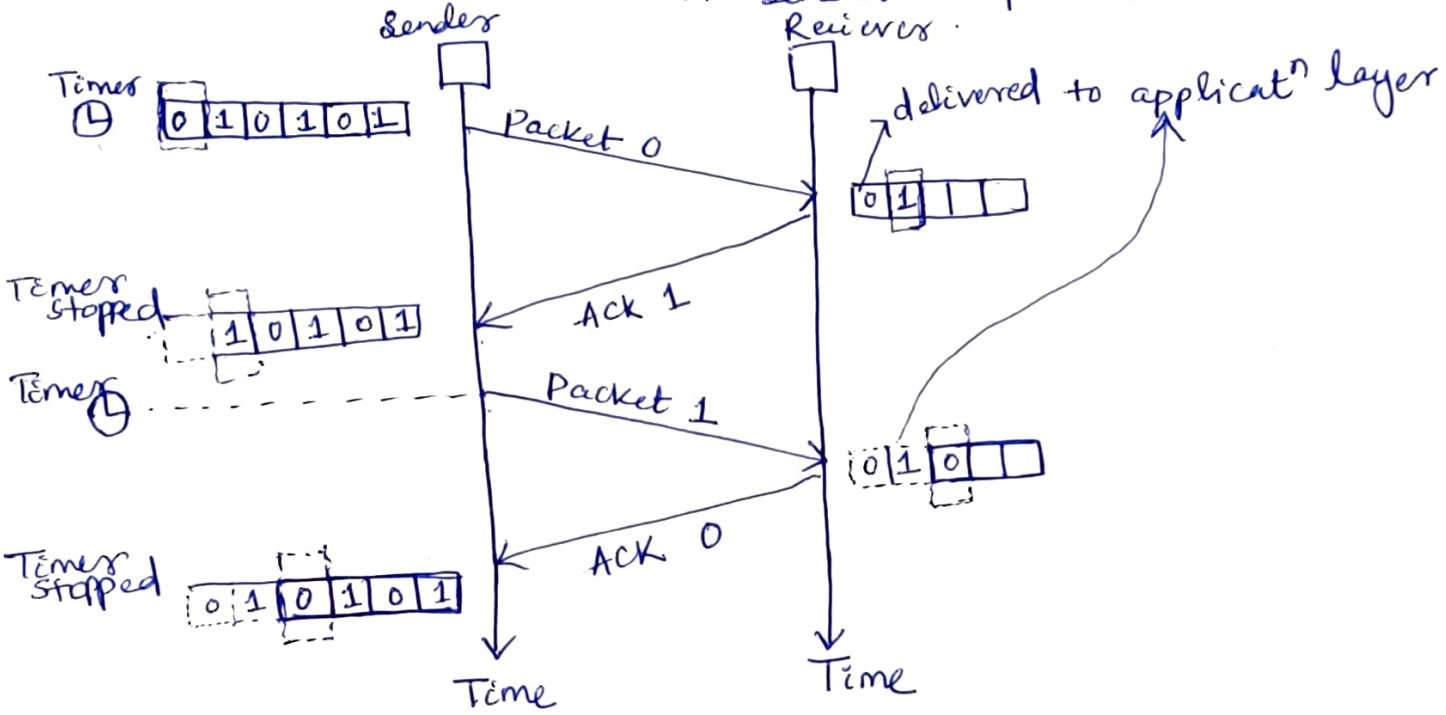
(2) Stop and Wait Protocol : (SAW)

- It is connection oriented.
- It provides flow control and error control.
- Sender and reciever sliding window size is 1.
- Sender send one packet and wait for acknowledgement before sending the next packet.
- To detect corrupted packet checksum field is added to packet.
- Sender starts a timer before sending packet.
- If acknowledgement (ACK) arrive before time expires, timer stoped.
- If timer expire and the sender resend the packet if ACK not arrived assuming that packet is lost.
- fig. 3-20 (slide no
- To prevent duplicate palket, sequence numbers and ACK numbers are used.

⑦

- In this protocol, only two sequence numbers can be used i.e. x & $x+1$.
- If $x=0$, then the sequence numbers of packet are

- It is called modulo 2 arithmatic
- Communication between sender and reciever is

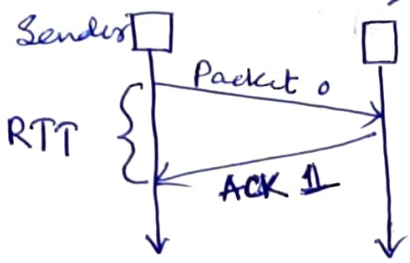


- ACK number is the sequence number of next packet expected by the reciever.
- If packet 0 recieved at reciever, it will send ACK 1 to the sender.
- Scenario of packet Loss:
  - Fig. 3.22 (Slide no
  - Packet 0 is sent and ACKed.
  - " 1 " " " Lost
  - " " " resent after timeout
  - Resent packet 1 is ACKed and timer stopped.
- Scenario of ACK Loss:
  - Fig 3.22 (Slide no

- Packet 0 is sent & ACK1 is sent by reciever.
- But ACK 1 is lost.
- Sender have no idea packet lost or ACK lost.
- After timeout, it resent packet 0.
- But as packet 0 is a duplicate packet and reciever is expecting packet 1 not packet 0, it will be discarded.
- ACK 1 will be sent by recievers as it recieved a packet and its expecting packet 1.

– Efficiency of SAW protocol:

- It is inefficient if it have high data rate and long round trip delay.
- Bandwidth delay product = Data rate × Round trip time
- Round trip time (RTT)



* RTT = 2 × Propagation delay

- Efficiency is measured in terms of link utilization (U).

$$U = \frac{\text{Time the link is utilized to transmit}}{\text{Total time}}$$

$$= \frac{\text{Transmission time}}{\text{Total time}}$$

$$= \frac{D_{trans}}{D_{trans} + RTT}$$

[Neglecting queuing and procuring delay

$$= \frac{D_{trans}}{D_{trans} + 2 \times D_{prop}}$$

$$= \frac{1}{1 + 2 \times \frac{D_{prop}}{D_{trans}}}$$

$$= \frac{1}{1 + 2a} \quad \text{where} \quad a = \frac{D_{prop}}{D_{trans}}.$$

⑨

- If L, R, D, V are given then U is

$$U = \frac{L/R}{L/R + RTT}$$

$$\Rightarrow U = \frac{L/R}{\frac{L}{R} + 2 \times \frac{D}{V}}$$

L = packet length
R = Data rate
D = Distance
V = velocity of medium

Q-1 - In a SAW system, the bandwidth of the line is 1 Mbps. 1 bit takes 20ms to make a round trip. If the system data packets 1000 bits length. what is the utilization percentage of link?

Sol^n - SAW protocol

$R = 1\,Mbps = 10^6\,bps$

$RTT = 20\,ms = 20 \times 10^{-3}\,s$

$L = 1000\,bits$

$U\% = ?\,\%$

$$U = \frac{L/R}{L/R + RTT}$$

$$= \frac{1000/10^6}{1000/10^6 + 20 \times 10^{-3}} = \frac{10^{-3}}{10^{-3} + 20 \times 10^{-3}}$$

$$= \frac{1}{21} = 0.047619$$

$U\% = 4.7619\,\%$

Q.2 - The distance from earth to a planet is 9×10^{10} m. what is the channel utilization if SAW protocol is used for frame transmission on a 64 Mbps. point to point link? Assume that the frame size is 32 KByte and speed of light is 3×10^8 m/s.

Sol^n - $D = 9 \times 10^{10}$ m

$R = 64\,Mbps = 64 \times 10^6\,bps$

$L = 32\,KB = 32 \times 10^3 \times 8\,bit$

$V = 3 \times 10^8\,m/s$

$$U = \frac{1}{1 + 2a} \quad \text{where } a = \frac{D_{prop}}{D_{trans}}$$

$$D_{prop} = \frac{D}{V} = \frac{9 \times 10^{10}}{3 \times 10^{8}} = 300s$$

$$D_{trans} = \frac{L}{R} = \frac{32 \times 10^{3} \times 8}{64 \times 10^{6}} = 4 \times 10^{-3} s$$

$$a = \frac{D_{prop}}{D_{trans}} = \frac{300}{4 \times 10^{-3}} = \frac{3}{4} \times 10^{5} = 75 \times 10^{3}$$

$$U = \frac{1}{1+2a} = \frac{1}{1+150 \times 10^{3}} = 0.0000667$$

Q-3 – Consider the use of 10kbit size frame on a 10Mbps satellite channel with 270 ms delay. What is the link utilization for SAW ARQ technique assuming $P = 10^{-3}$ where p is the probability of single frame error. ?

Sol^{n} – $L = 10kbit = 10 \times 10^{3} bit$

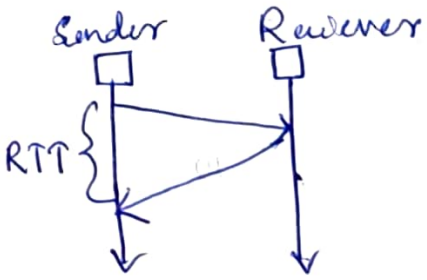
$R = 10Mbps = 10 \times 10^{6} bps$

$P = 10^{-3}$

$D_{prop} = 270ms = 270 \times 10^{-3} s$ $\quad D_{trans} = \frac{L}{R} = 10^{-3} s$

$$U = \frac{(1-P)}{1+2a} = \frac{1-10^{-3}}{1+2 \times \frac{270 \times 10^{-3}}{10^{-3}}} = \frac{0.999}{541} = 0.00184$$

Q-4 – Consider a computer network which sends the packet with size of 1000bits of data using SAW ARQ protocol. If the distance between source and destination is 400km and the propagation speed is 2×10^{8} m/s. Find the time required to send 10000bits.



Sol^{n} – $L = 1000bits$

$D = 400km = 400 \times 10^{3} m$

$V = 2 \times 10^{8} m/s$

1000 bit is 1 packet

10000 " " $\frac{10000}{1000} = 10$ packets

1 packet takes 1 RTT time

10 " " 10 " "

(11)

$$RTT = 2 \times D_{prop}$$
$$= 2 \times \frac{D}{V}$$
$$= 2 \times \frac{400 \times 10^3}{2 \times 10^8}$$
$$= 4 \times 10^{-3} s$$

Time required to transmit 10 packet $= 10\ RTT$
$= 10 \times 4 \times 10^{-3} s$
$= 40\ ms$

Hence time required to send 10000 bits in 40ms.

Q-5 – Assume a 100 Mbps link of 10000 m in length with 5 nanosecond per meter propagation delay. Assume a constant length of 400 byte data frame, 64 byte ACK frame, 10 µs of processing delay for each data frame and 5 µs processing time for each ACK. Calculate the link utilization assuming SAW protocol for sender.

Sol^n – $R = 100\ Mbps = 100 \times 10^6\ bps$

$D = 10000\ m$

Delay = 5 nanosec per meter

$D_{prop} = 10000 \times 5 \times 10^{-9}\ sec = 5 \times 10^{-5} \cdot sec$

$L_D = 400\ byte = 400 \times 8\ bit$

$L_A = 64\ byte = 64 \times 8\ bit$

$D_{proc\_data} = 10\ \mu s = 10 \times 10^{-6} s$

$D_{proc\_ACK} = 5\ \mu s = 5 \times 10^{-6} s$

$$U = \frac{\text{Time to transmit packet}}{\text{Total time}} = \frac{D_{trans\_data}}{D_{trans\_data} + D_{trans\_ACK} + RTT + D_{proc\_data} + D_{proc\_A}}$$

$$D_{trans\_data} = \frac{L_D}{R} = \frac{400 \times 8}{100 \times 10^6} = 32 \times 10^{-6} s$$

$$D_{trans\_ACK} = \frac{L_A}{R} = \frac{64 \times 8}{100 \times 10^6} = 512 \times 10^{-8} s$$

$$RTT = 2 \times D_{prop} = 2 \times \frac{D_{prop}}{} = 2 \times 5 \times 10^{-5}\,s = 10^{-1}\,s$$

$$U = \frac{D_{trans\_Data}}{D_{trans\_Data} + D_{trans\_ACK} + RTT + D_{proc\_data} + D_{proc\_ACK}}$$

$$= \frac{32 \times 10^{-6}}{32 \times 10^{-6} + 512 \times 10^{-8} + 10^{-1} + 10 \times 10^{-6} + 5 \times 10^{-6}}$$

$$= \frac{32 \times 10^{-6}}{32 \times 10^{-6} + 5.12 \times 10^{-6} + 1000 \times 10^{-6} + 10 \times 10^{-6} + 5 \times 10^{-6}}$$

$$= \frac{32}{32 + 5.12 + 100 + 10 + 5}$$

$$= \frac{32}{152.12} = 0.21036$$

Drawback of SAW protocol is that the link utilization is very less.

GO-BACK-N Protocol:

- To increase the efficiency of transmission multiple packets must be sent.
- GO-BACK-N (GBN) protocol sends multiple packet at a time.
- Sequence number are modulo 2^m, where m is the size of sequence number field in bits.
- Acknowledgement number is cumulative
- ACK define the sequence number of next packet expected.
- Fig. 3.23 (Slide no.
- Send window is an imaginary box covering the sequence number of data packets that can be in transit or can be sent.
- Fig. 3.24 (Slide no

- Maximum size of the window is $2^m - 1$ for sender
- Send window contains 4 types of packets -
  i. Packet sent, ACK recieved
  ii. " " not recieved → Outstanding packet
  iii. Processed " , " and can be sent next
  iv. Not Processed.
- Sb : First outstanding
  Sn : Next to send
  Rn : " " recieve
- Fig. 3.25 (Slide no
- Size of recieve window is 1
- Fig - 3.26 (Slide no
- Any packet arriving out of order is discarded.
- Timer is set for a group of packets
- After timer expire, all the outstanding packets are retransmitted.
- Why maximum size of send window is $2^m - 1$.
  - Fig. 3.28 (Slide no
  - Here $m = 2$, Hence size of send window is $2^m - 1 = 3$
  - Compare window of size $2^m - 1 = 3$ and $> 2^m - 1 = 4$
  - Size of window 3
    * All ACK are lost
    * After timer expire all packets are resent.
    * Reciver expecting packet 3.
    * So the resent packet 0, 1 & 2 will be discarded as they are duplicate.
    * So packets are correctly discarded.
  - Size of window 4
    * All ACK lost
    * After timer expire, packet 0, 1, 2 & 3 are resent
    * Receiver expecting packet 0.

* So packet 0 will be wrongly accepted although it is duplicate packet
* So it is wrong.

- Hence the size of send window must be less than 2^m.

- Scenario of ACK loss:
  - Refer Fig. 3.29 (slide no
- Scenario of Packet loss:
  - Refer Fig. 3.30 (slide no
- Link Utilization of GBN protocol:
  - N = window size = No. of packets in the window
  - m = Number of bits used for sequence number
  - P = Probability of frame error
  - link Utilization (U),

$$U = \frac{N(1-P)}{(1+2a)(1-P+NP)}$$

  - If P = 0 i.e. there is no frame error, then U is

$$U = \frac{N}{1+2a}$$

Q-1- Consider the use of 10kbit size frames on a 10Mbps satellite channel with 270ms delay. What is the link utilization for GBN protocol assuming $P = 10^{-3}$ and window size is 7?

Sol^n: $L = 10Kbit = 10 \times 10^3$ bit

$R = 10Mbps = 10 \times 10^6$ bps

$D_{prop} = 270ms = 270 \times 10^{-3}$ s

$P = 10^{-3}$

$N = 7$

$$D_{trans} = \frac{L}{R} = \frac{10 \times 10^3}{10 \times 10^6} = 10^{-3} s$$

$$a = \frac{D_{prop}}{D_{trans}} = \frac{270 \times 10^{-3}}{10^{-3}} = 270$$

$$U = \frac{N(1-P)}{(1+2a)(1-P+NP)}$$

$$= \frac{7(1-10^{-3})}{(1+2\times270)(1-10^{-3}+7\times10^{-3})} = 0.01285$$

Q-2: If you are designing a GBN protocol with bandwidth of 100kbps and has a one way delay of 4s. Assume each packet carries 1KByte of data, what is the minimum number of bits you need for sequence number?

Soln:

GBN Protocol

$R = 100\,Kbps = 100 \times 10^3\ bps$

$D_{prop} = 4\ s$

$L = 1\,KByte = 10^3 \times 8\ bits$

Number of bits in link = Bandwidth delay product

$= R \times RTT$

$= R \times 2 \times D_{prop}$

$= 100 \times 10^3 \times 2 \times 4$

$= 8 \times 10^5\ bits$

Total Number of packet = N = ~~length of one pa~~

$= \frac{\text{Total Number of bits}}{\text{Number of bits in one packet}}$

$= \frac{8 \times 10^5}{10^3 \times 8}$

$\Rightarrow N = 100$

Maximum window size in GBN $= 2^m - 1$

$\Rightarrow N = 2^m - 1$

$\Rightarrow 100 = 2^m - 1$

$\Rightarrow 2^m = 101$

$\Rightarrow m = 7$

Number of bits required for sequence number is 7.

Q-3: A 20Mbps satellite link has propagation delay of 400 μs. The transmitter employs GBN ARQ with window size N = 10. Assuming each packet is 100 Byte long, calculate the maximum datarate possible.

Sol^n:

$R = 20Mbps = 20 \times 10^6$ bps

$D_{prop} = 400\mu s = 400 \times 10^{-6}$ s

GBN ARQ Used

$N = 10$

$L = 100byte = 100 \times 8$ bits

Link Utilization $= U = \frac{N}{1+2a}$

$D_{trans} = \frac{L}{R} = \frac{100 \times 8}{20 \times 10^6} = 4 \times 10^{-5}$ s

$a = \frac{D_{prop}}{D_{trans}} = \frac{400 \times 10^{-6}}{4 \times 10^{-5}} = 10$

$U = \frac{10}{1+20} = \frac{10}{21}$

Maximum data rate $= R \times U$

$= 20 \times \frac{10}{21}$ Mbps $=$ Mbps

Q-4: Consider a computer network which sends the packet with a size of 1000 bits of data. Distance between source and destinat^n is 400km and the propagation speed is 2×10^8 m/s. If GBN ARQ protocol with window size 7 is used, what is the time required to send 10000 bits.

Sol^n:

$L = 1000$ bits

$D = 400km = 400 \times 10^3 m$

$V = 2 \times 10^8$ m/s

GBN protocol Used

Window size $= N = 7$

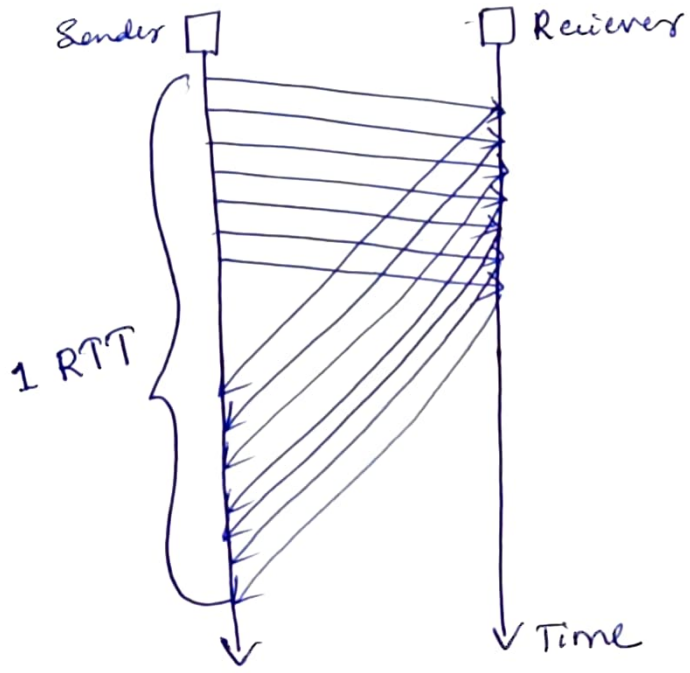
Total bits to be sent = 10000 bits

Total Number of packet = $\frac{\text{Total bits}}{\text{Size of one packet}}$

$= \frac{10000}{1000} = 10$

$D_{prop} = \frac{D}{V} = \frac{400 \times 10^3}{2 \times 108} = 2 \times 10^{-3}$ s

$RTT = 2 \times D_{prop} = 2 \times 2 \times 10^{-3} = 4 \times 10^{-3}$ s



window size = N = 7

7 packets transmitted in 1 RTT

10 " " " 2 "

So time required to transmit 10 packet is

$2 \times 4 \times 10^{-3}$ s $= 8 \times 10^{-3}$ s $= 8$ ms.

Selective Repeat Protocol : (SR)

- In GBN protocol, if a packet is lost or corrupted the sender resend all the outstanding packet (N) after the lost packet
- Hence the protocol name GBN.

- But the packets recieved correctly but out-of-order, so discarded.
- It is inefficient and may increase congestion.
- To remove this drawback of GBN protocol, selective repeat protocol is designed.
- In SR protocol, the sender resends only the selective packet that are actually lost.
- To do that, buffers must be implemented at the reciever side.
- Fig. 3.31 (slide no
- SR protocol uses 2 windows
  - Send window
  - Recieve window.
- Maximum size of send window is $2^{(m-1)}$.
- " " " recieve " " "
- Fig. 3.32 (slide no.
- Fig. 3.33 (slide no.
- SR protocol keeps the out-of-order packet instead of discarding the packet.
- SR protocol uses one timer for each outstanding packets.
- SR protocol treats packet individually while GBN protocol treats the packet as group.
- In SR protocol, ACK define the sequence number of the packet that recieved safe and sound. called as selective ACK.
- Why maximum window size in SR protocol is $2^{(m-1)}$.
  - Refer Fig. 3.36 (slide no.
- Scenario of packet loss:
  - Refer fig. 3.35 (slide no.

- link utilization of SR protocol:
  - Window size is N
  - Probality of frame error P
  - link utilization U,

$$U = \frac{N(1-P)}{1+2a}$$

  - If $P = 0$ i.e. no frame error occurs

$$U = \frac{N}{1+2a}$$

Q-1: Consider the use of 10KByte size frames on a 10Mbps satellite channel with 270 ms delay. Assume $P = 10^{-3}$ and $m = 3$. If SR protocol is used what will be the link utilization

Sol^n:

$L = 10 \times 10^3 \times 8$ bit

$R = 10 \times 10^6$ bps

$D_{prop} = 270$ ms $= 270 \times 10^{-3}$ s

$P = 10^{-3}$

$m = 3$

$N = 2^{m-1} = 2^{3-1} = 4$

$D_{trans} = L/R = \frac{10 \times 10^3 \times 8}{10 \times 10^6} = 8 \times 10^{-3}$ s $= 8$ ms

$U = \frac{N(1-P)}{1+2a}$

$a = \frac{D_{prop}}{D_{trans}} = \frac{270 \times 10^{-3}}{8 \times 10^{-3}} = 33.75$

$U = \frac{4(1-10^{-3})}{1+2 \times 33.75}$

$= \frac{4 \times 0.999}{1+67.5}$

$= \frac{3.996}{68.5}$

$= 0.058335$

Q-2: If you are designing a SR protocol with bandwidth of 100Kbps and one way delay of 4S. Assume each packet carries 1Kbyte data, what is the minimum number of bits needed for sequence number?

Soln:

$R = 100\text{Kbps} = 100 \times 10^3 \text{ bps}$

$D_{prop} = 4s$

$L = 1\text{Kbyte} = 10^3 \times 8 \text{ bit}$

SR protocol

No. of bits in link = Bandwidth Delay product

$= R \times RTT = R \times 2 \times D_{prop}$

$= 100 \times 10^3 \times 2 \times 4 = 8 \times 10^5 \text{ bits}$

Total number of packet, $N = \frac{\text{Total bits}}{\text{Bits in one packet}}$

$= \frac{8 \times 10^5}{8 \times 10^3} = 100$

Maximum window size in SR protocol = $2^{(m-1)} = \frac{2^m}{2}$

$\Rightarrow N = \frac{2^m}{2}$

$\Rightarrow 100 \times 2 = 2^m$

$\Rightarrow 2^m = 200$

$\Rightarrow m = 8$

Bidirectional Protocol: (Piggybacking)

- In practical scenario, data packets flow from sender to reciever and reciever to sender called as bidirectional.
- ACK also bidirectional.
- Piggybacking improve efficiency.

User Datagram Protocol : (UDP)

- UDP is a connectionless, unreliable transport protocol.
- Why a process use UDP?
  - Its a simple protocol using a minimum overhead.
  - If reliability is not required, UDP can be used.
- User Datagram:
  - UDP packets are called user datagram.
  - Fig. 3.39 (Slide no.
  - UDP packets has fixed size header of 8 bytes.
  - Fields in header :
    - * Source port Number - It is a 16 bit field.
    - * Destination port Number - " " " " " .
    - * Total length - Total length of user datagram i.e header and data combined. It is 16 bit.
    - * checksum - It is a 16 bit field used for error checking.
- Example 3.11 (Slide no
- UDP services :
  (1) Process to process communication
  (2) Connectionless service
  (3) Flow control - Does not provide flow control
  (4) Error " - " " " error "
  (5) Congestⁿ " - " " " congestⁿ "
  (6) checksum :
    - UDP checksum calculation includes 3 sections : a pseudoheader, UDP header and Data.
    - Fig. 3.40 (Slide no
    - Pseudo header is part of header of IP packet.
    - If the checksum does not include pseudoheader, user datagram may arrive safe and sound
    - But if the IP header is corrupted it may be delivered to wrong host.

- Protocol field is for protocol for UDP it is 17.
- If during calculation of checksum, error is detected packet will be dropped.

(7) Encapsulation & decapsulation
(8) Multiplexing & Demultiplexing.

- Advantages of UDP:

(1) It is suitable for process that require simple request response communication
(2) It is suitable for process with internal flow and error control mechanism.
(3) It is suitable for multicasting
(4) " " used " management process
(5) " " " " route updating
(6) " " " " interactive real time applications.

Checksum Calculation:

steps for calculating checksum

1. Divide the content of the segment to 16 bits parts
2. Add the 16 bits parts
3. If there is carry, add the carry to result.
4. Take 1's complement of result.
5. This will be the value of checksum.

Q-1: Find the checksum if the bits in the segment are

1101010101010101 1110011001100110.

Sol^n: After dividing to 16 bit parts, add the parts

```
  1110011001100110
+ 1101010101010101
-----------------
(1) 1011101110111011
+                  1
-----------------
  1011101110111100
```

carry

5's complement 0100010001000011 ⟶ checksum

Q-2: A client is using a UDP socket 153.18.8.105 : 1087 to connect a daytime server having socket 171.2.14.10 : 13. The UDP payload is "TESTING". Given data : ASCII values in decimal : E = 69, G = 71, I = 73, N = 78, S = 83, T = 84. Calculate the UDP checksum for the segment with above mentioned details.

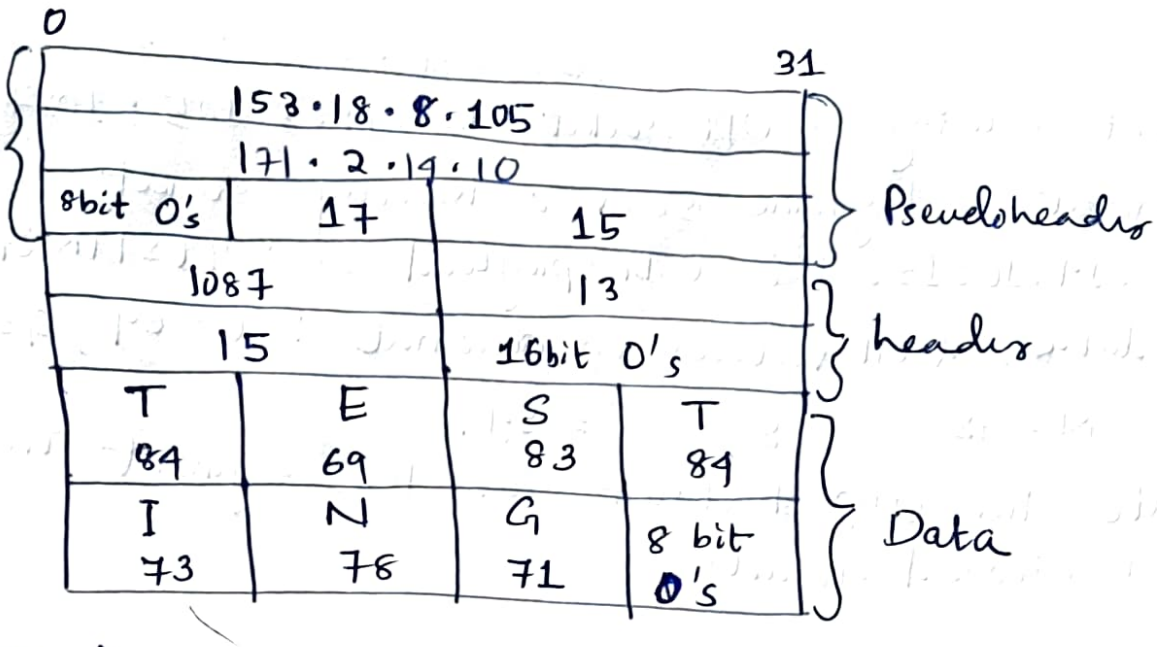
Sol^n:

- UDP checksum = pseudoheader + header + Data(payload)
- Refer Fig. 3.40 (Slide No -
- Parts of pseudoheader and its value from above are
  * 32-bit source IP addren - 153.18.8.105
  * " " Destinat^n " " - 171.2.14.10
  * 8 bit all 0's - 00000000
  * 8 bit protocol - For UDP - 17
  * 16 " UDP total length = 8 byte header + 7 byte data or payload (TESTING) = 15
- Parts of header:
  * Source port no - 1087
  * Destinat^n " " - 13
  * Total length = 15
  * Checksum = 0
- Data -

T	E	S	T	I	N	G
↓	↓	↓	↓	↓	↓	↓
84	69	83	84	73	78	71

Each will be represented in 8 bits

- Putting all the values in its place with appropriate bits in the format of Fig. 3.40



- Divide the segment into 16 bit parts.

153.18	→	10011001 00010010
8.105	→	00001000 01101001
171.2	→	10101011 00000010
14.10	→	00001110 00001010
0 17	→	00000000 00010001
15	→	00000000 00001111
1087	→	00000100 00111111
13	→	00000000 00001101
15	→	00000000 00001111
0	→	00000000 00000000
84, 69	→	01010100 01000101
83, 84	→	01010011 01010100
73, 78	→	01001001 01001110
71, 0	→	01000111 00000000
Sum	→	10010110 11101011
1's Complement	→	01101001 00010100 → checksum.

Transmission Control Protocol: (TCP)

- TCP is a connection oriented reliable protocol.
- TCP uses combination of GBN and SR protocol to provide reliability.
- TCP achieves reliability by
  - Using checksum for error detection.
  - Retransmission of lost or corrupted packet
  - Use of cummulative and selective ACKs.
  - " " timers.
- TCP services:

(1) Process - to - process communication.

(2) Stream delivery service:
- TCP is stream oriented protocol.
- " deliver the data as stream of byte.
- Fig. 3.41 (Slide no
- TCP needs buffer for storage - sending and recieving buffer.
- Fig. 3.42 (Slide no
- TCP groups a number of byte together into a packet called segment.
- Fig. 3.43 (Slide no

(3) Full duplex communication -
- Data flow in both direction at same time

(4) Multiplexing & Demultiplexing.

(5) Connection oriented service.
- Establish logical connection
- Exchange data
- Terminate the connection.

(6) Reliable service
- ACK is used to provide reliability.

- TCP features:

(1) Numbering System

- No field in TCP header for segment number
- TCP uses sequence number and ACK number.
- These are referred by byte number.
- TCP numbers all the data bytes.
- " " does not start from 0 always.
- It is a random number betn 0 to $(2^{32}-1)$ to number the first byte.
- If you have 100 byte data and number starts from 6, it will end at 105.
- TCP assigns sequence number to each segment
  - * Sequence number of first segment is called initial sequence number (ISN).
  - * ISN is a random number.
  - * Sequence number of next segment is the sequence number of the previous segment plus the number of bytes of data carried.
- Example 3.17 (Slide no
- ACKnowledgement Number
  - * ACK number defines the number of next byte reciever is expecting
  - * ACK number is cummulative
  - * New version of TCP uses selective ACK sometimes.

TCP Segment:

- A packet in TCP is called TCP segment
- Segment has 2 parts
  - 20 to 60 bytes Header
  - Data from applicatn layer or payload.

- Fig. 3.44 (Slide no.

- It, TCP header does not contain option field, it is fixed size of 20 bytes.
- TCP header will be maximum upto 60 bytes if it contains header.
- Fields of TCP header:
  - Source port number – Its a 16 bit field for sender port.
  - Destinat" " – " " " " " reciever "
  - Sequence number – It is a 32 bit field define the number assigned to the first byte of data in segment.
  - Ack number – Its 32 bit field that define the next byte expected by reciever.
  - Header length – It is a 4 bit field which define number of 4 byte word in TCP header. (HLEN)
    * TCP header is 20 to 60 byte ⇒ This field value is betn 5 to 15.
    * If HLEN = 1001, means 9 ⇒ 9 x 4 = 36 byte is the header length.
  - Control bits: (Flags)
    * 6 different control bit are there each of 1 bit
    * Fig. 3.45 (Slide no
    * These bits are used for flow control & connection set up.
    * URG: Urgent pointer
    * ACK: Acknowledgement
    * PSH: Request to push
    * RST: Reset the connection.
    * SYN: Synchronizat"
    * FIN: Terminate the connect".
    * IF flag bit values are 1, it is enabled
    * " " " " " 0, " " disabled.

Window size:
* Its a 16 bit field define the sender window size.
* Maximum size is 2^{16} bytes.
* Referred as recieving window (rwnd) as it is

determined by reciever.

- Checksum.
  * 16 bit field.
  * Fig. 3.46 (Slide no
  * It contains 3 parts: pseudoheader, TCP header & Data.
  * Checksum is calculated using same procedure as discussed previously.
- Urgent Pointer:
  * 16 bit field.
  * Valid when URG flag value is 1.
  * Used when segment contain urgent data.
- Options:
  * 40 bytes of optional information.

- The header of TCP is encapsulated with data recieved from applicatn layer combinedly called as segment.

TCP Connection:

- TCP is a connection oriented transport protocol which establish a logical path between source and distination.
- It has three phases:
  - Connection establishment
  - Data transfer
  - Connection Termination.

(1) Connection establishment:

- TCP transmit data in full duplex mode.
- Sender and reciever get approval from each other before data transmission.
- Connection establishment in TCP is called three way handshaking.
- Client want to make connection with server.
- The process starts with server.
- The server requests its TCP that it is ready to

, accept connection which is called passive open.
- Client program request for connection called as active open.
- Fig. 3.47 (Slide no
- TCP starts the three way handshaking procedure.

(i) SYN segment:
- Client send the first segment called SYN segment.
- SYN flag is set.
- This segment is for synchronization of sequence number.
- ISN is a random number.
- No ACK number.
- No window size.
- It is a control segment
- No data is carried.
- Consume one sequence number.

(ii) SYN + ACK segment:
- Server sends the second segment called SYN + ACK segment.
- SYN and ACK flags are set.
- Server use this segment to initialize sequence number of server side.
- Server Acked the reciept of SYN segment from client.
- It defines the recieve window size (rwnd) of server.
- Consume one sequence number.

(iii) ACK segment:
- Client send the third segment called ACK segment.
- It ACKed the reciept of segment from server.
- ACK flag set.

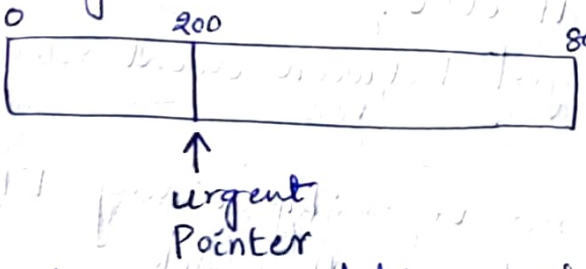
- If it do not carry data it will not consume seq. number.
- Some ACK segment carry data which will consume seq. number.

- SYN Flooding Attack:
  - One or more malicious attackers send a large number of SYN segments to server pretending that each of them coming from different client by faking IP address.
  - If number of SYN segment is large, server will run out of resources and unable to accept request from valid client.
  - This attack is also known as denial of service (DoS) attack.
  - Sol^n.
    - * Limit of connect^n request during specified period of time.
    - * Filter out datagram coming from unwanted source address.
    - * Postpone resource allocation until it is verified that request is coming from valid IP address using cookie.

(2) Data Transfer:

- After connection is established, bidirectional data transfer can takes place.
- Client and server send data and ACK on both direction.
- ACKs are piggybacked with the data.

- Fig. 3.48 (Slide no.
- Pushing Data:
  - Recieving TCP bubber the data when they arrive.
  - It delivers to applicat^n Layer when application layer is ready to recieve.
  - If sending TCP want to deliver the data immidiatly to receiving applicat^n layer, sender request a push operation by enabling the PSH flag.
  - Recieving TCP will know that the data should be delivered immidiatly to applicat^n layer.
- Urgent Data:
  - In some occasions urgent bytes are needed to be sent.
  - URG bit flag is set.
  - The urgent data are inserted at the begining of the segment.



200Byte of urgent data in the segment is there.

- Urgent pointer debines the end of urgent data.

(3) Connection Termination:

- Either server or client can close the connection.
- It is implemented in two ways.

(a) Three way handshaking:

- Fig. 3.49 (Slide no.
- Three segment exchanged between client and server.
- FIN segment
  - * Client TCP sends first segment, FIN segment.
  - * FIN flag set
  - * FIN segment can include last byte of data

* It is not a purely control segment.
* It will consume one sequence number.

- FIN + ACK Segment
  * Server sends
  * Confirm the reciept of FIN segment
  * Announce the closing of connection.
  * Can carry Last chunk of data from server.
  * Consume sequence number.
  * Not a purely control segment.
* ACK segment.
  * client sent Last Ack!
  * Confirm the reciept of FIN segment from server.
  * Cannot carry data, so purely control segment.

(b) Fourway handshaking with half close operation:

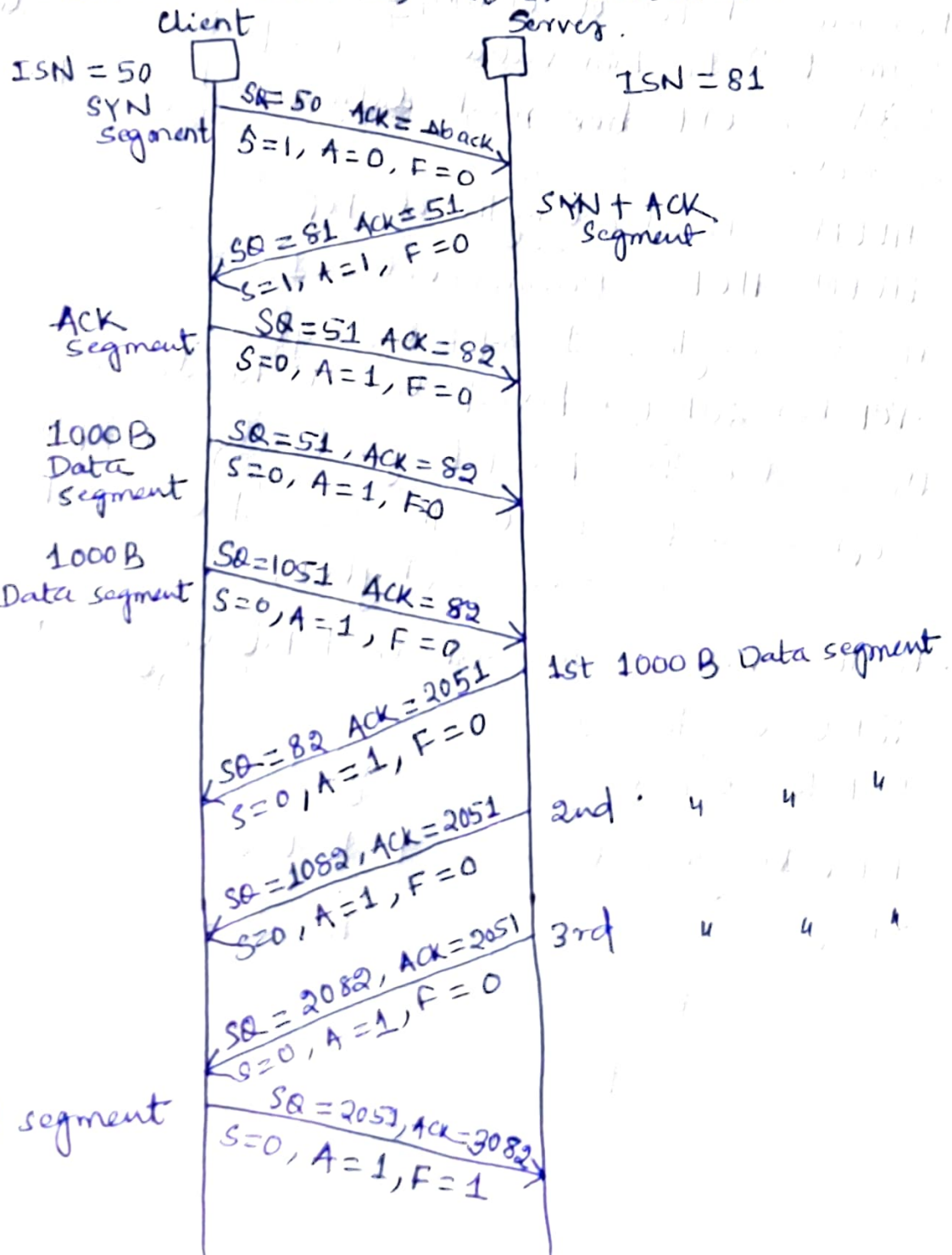
- In TCP one end can stop sending data while still recieving called as half close.
- four segments are exchanged between client-server.
- Fig. 3.50 (Slide no.
- client half close the connection by sending FIN.
- Server accept half close by sending ACK.
- Server can still send data.
- After sending all data, server send a FIN segment.
- client send ACK to server for FIN segment.

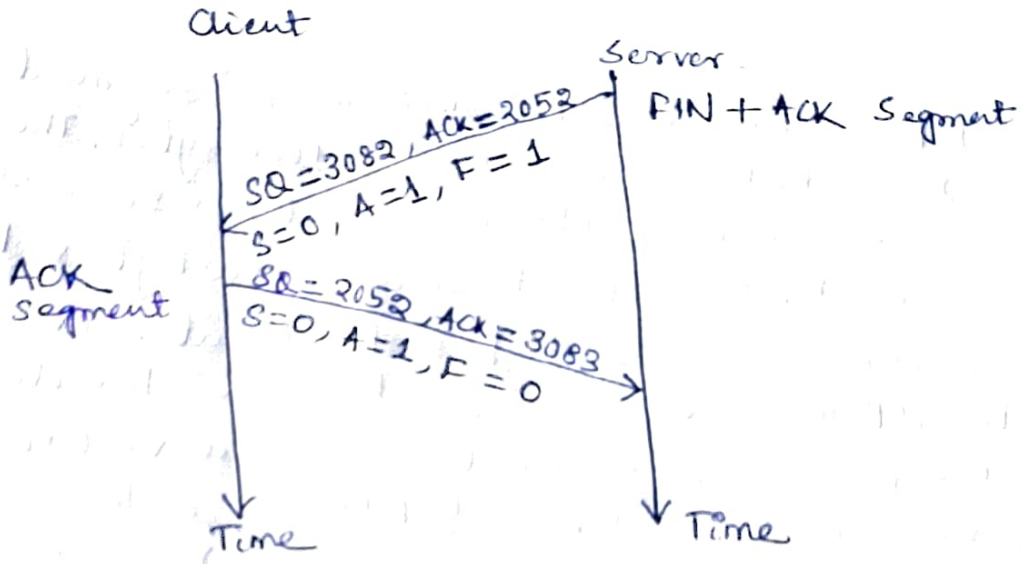
(4) Connection Reset :

- May terminate an existing connection.
- RST flag is set.

Q-1 - In a TCP connection, the maximum segment size (MSS) is both direction is 1000 bytes. The initial sequence number (ISN) for client is 50 and for server is 81. The client sends 2000 bytes to server and server sends 3000 bytes to client. Give the complete TCP message exchange between the client and server. For each segment draw a vector showing the value of SYN, ACK and FIN bit with sequence number (SEQ) and ACK number (ACK). Assume no packets are lost and the application consumes the data as soon as it is recieved.

Sol^n:





Q-2- If the value of HLEN field in TCP is 1101, how many bytes of option are included in the segment? If this value is used in total length field in UDP how much data in byte the segment carries.

Soln - HLEN = Header length of TCP

HLEN = 1101, Decimal value 13

Header length = 13 × 4 = 52 bytes

TCP has 20 Bytes fixed length header.

Header length = Fixed length header + options

⇒ options = Header length − Fixed length header

= 52 − 20 = 32 bytes.

- If this value is used in total length of UDP

Total length = 13

UDP header length = 8 bytes

Total length = Header + Data

⇒ Data = Total length − Header

= 13 − 8

= 5 bytes

TCP State transition:

- To keep track of event happening TCP specified a finite state machine.
- Fig. 3.51 (Slide no
- Both client and server states are represented.
- Round corner rectangle are called states.
- Transition from one state to another shown in directed lines.
- Each line has two strings separated by slash
  - First is what TCP receives.
  - Second " " " sends.
- Dotted line in server & solid line in client.
- Half close scenario
  - * Fig. 3.52 (Slide no.
- States of TCP
  - * Table 3.2.(Slide no
- Fig. 3.53 (Slide no.
  - Events at the client
    - * Client procm issue an active open.
    - * TCP send a SYN segment and move to SYN_SENT state
    - * It recieves SYN+ACK segment, send an ACK segment and move to ESTABLISHED state.
    - * Data transferred in both direction.
    - * Client has no data to sent, so issue an active close.
    - * client send a FIN segment and goes to FIN_WAI state.
    - * It recieves an ACK and go to FIN_WAIT_2 state.

* It recieves a FIN segment, send an ACK and go to TIME-WAIT state. and starts a 2MSL Timer.
* It remain in this state for 2MSL time.
* When 2MSL timer expire, it goes to CLOSED state
* MSL - Maximum segment life → Given in form of time in second. (30s to 2min)
* MSL is the amount of time any segment can exist in the network.
* 2MSL time allow TCP to resend the final ACK in case ACK is lost.

- Events at the server side.
  * Server issue a passive open.
  * " TCP goes to Listen.
  * " recieve SYN segment, sends SYN+ACK segment and goes to SYN-RCVD state waiting for ACK.
  * Server recieves ACK segment and goes to ESTABLISHED state.
  * Data transfer takes place.
  * Server recieves FIN and sends a ACK and goes to CLOSE_WAIT state.
  * After transmitting data, server issue passive close
  * Server send FIN segment and goes to LAST_ACK state waiting for ACK.
  * Server recieves ACK segment and goes to **CLOSED** State.

Windows in TCP:

- TCP uses two windows
  - Send window
  - Recieve window

(1) Send window in TCP:

- Fig. 3.54 (Slide no
- window size here is 100 Bytes.
- Send window can open, close or shrink.
- If left wall of window moves to right, then window closes
- If right wall moves right then window opens
- " " " " left " " shrinks.
- Send window in TCP is similar to SR protocol but have difference
  a. window size in SR is number of packet but in TCP is number of bytes.
  b. In some TCP, the segment of data is delivered as soon as it recieved.
  c. TCP protocol uses only one timer but SR uses one timer for each packet.

(2) Recieve Window

- Fig. 3.55 (Slide no.
- Recieve window opens and closes.
- The difference bet^n recieve window of TCP and SR protocol
  a. TCP allow recieve procss to pull data at its own pace.
  rwnd = buffer size − number of bytes waiting to be pulled.
  b. ACK is SR is selective while TCP uses both selective and cummulative ACKs.

Flow Control in TCP:

- Flow control balances the rate ~~of~~ producer creates data with rate consumer use data.
- Fig. 3.56 (slide no
- Flow control is the feedback from recieving TCP to sending TCP.
- To achaive flow control, sender and reciever adjust there window size.
- Recieve window closes when more bytes arrive from the sender
- Recieve window opens when bytes are pulled by application layer.
- Recieve window does not shrink.
- Opening, closing and shrinking of send window is controlled by reciever.
- new ACK + new Window (rwnd) ≥ Last ACK + last rwnd.
- Fig. 3.57 (slide no.
- Example of flow control

Q-1 - Host A and B are directly connected with a 100 Mbps link. There is one TCP connection between the two host. Host A is sending to Host B file over this connection. Host A sends its application data into its TCP socket as a rate as high as 120 Mbps. But host B can read out its TCP recieve buffer at a maximum rate of 60 Mbps. Describe the effect of TCP flow control.

Sol^n:

- Link rate is 100Mbps ⇒ Host A sending rate can be at most 100Mbps
- Host A sends data to Host B faster than host B capacity to remove data from buffer.

- Host B has maximum rate of 60Mbps.
- When buffer of host B will be full, it will send signal to host A to stop sending data by setting the recieve window to 0.
- Host A will repeatedly stop and start transmission based on recieve window it obtain from host B.
- If this TCP communication continues, maximum data rate host A can send to host B is 60Mbps.

Window shutdown

- Shrinking of the send window is discouraged.
- Reciever can temporarily shutdown the window by sending a rwnd of 0.
- This happen when reciever do not want to recieve data.
- Sender does not actually shrink the size of ~~data~~ window but stop sending data until new window arrive.
- But sender send a segment with 1 byte of data called as probing.
- Used to prevent deadlock.

- Silly window Syndrome
  - Problem occur when sending applicatn creates data slowly and/or recieving application consume slowly.
  - Data send in very small segment which reduces efficiency.
  - 41 byte datagram (20byte TCP header, 20 byte IP header and 1 byte data) contain 1 byte data which is inefficient.
  - This is called silly window syndrome.
- Syndrome created by sender
  - Silly window syndrome occur at sender if application layer creates data slowly.

- 1byte data segments will be created more frequently and travel through internet.
- Soln is provided by Nagle's algorithm
  * Sending TCP sends first piece of data even it it is 1byte
  * Then it accumulates the data to fill maximum segment size and it sends.
  * Segment 3 is sent it ACK recieved or it have enough data.
  * It takes into account speed of applicatn program to create data and speed of network to transport data.
  * Applicatn program faster segment larger
    " " slower " smaller.

- Syndrome created by reciever.
  * Silly window syndrome will occur at reciever it it consume data slowly.
  * It it consume very slowly, then buffer will be full
  * Then reciever will send advertise window of size 0.
  * Sender stops sending.
  * Then 1 byte consumed and reciever send window to sender
  * Sender send 1 byte data and again silly window syndrome.
  * Two solutions propound

(1) Clark's solution
- Send an ACK as soon as the data arrive but announce a window size 0 until enough space to accomodate a segment of maximum size.

(2) Delay the ACK.
- Reciever wait until there is space in buffer. Then it will send ACK for previous segment.
- It also reduce traffic, but may result in retransmission
- Delay should not be more than 500 ms.

Error Control:

- TCP is a reliable transport layer protocol means it deliver data without error.
- TCP provids error control to ensure reliability.
- Error control is a mechanism for
  - detecting and resending corrupted segment.
  - resending lost segment
  - Storing out-of-order segment
  - Detecting and discarding duplicate segment
- Error control in TCP achieved through
  - checksum
  - Acknowledgement
  - Time-out.
- Checksum
  - Used to check for corrupted segment.
  - Segment has invalid checksum, then it is discarded.
  - 16-bit checksum is used in TCP.
- Acknowledgement
  - TCP use ACK for confirmation of reciept of data.
  - Control segment that carry no data also ACKed.
  - ACKs are never acknowledged.
  - Two types of ACK in TCP

    (1) Cummulative ACKs:

    * It advertise the next byte expeted by the reciever.
    * 32 bit is used for ACK in TCP header.
    * Its valid when ACK flag is 1.

    (2) Selective ACKs:

    * It reports the bytes that are recieved which may be out-of-order.
    * Implemented in option field of TCP header.

- Generating ACK

1. When segment is sent, an Ack is piggybacked which gives the next sequence number expected.
2. When reciever has nothing to send, and it recieve an segment, reciever delay sending Ack. (500 ms)
3. When a segment arrive which is expected and previous segment has not been Acked, an Ack should be send immediately.
4. When out-of-order segment arrive, reciever send Ack with next expected segment.
5. When missing segment arrive, an Ack with next expected sequence number is send.
6. If duplicate segment arrive, it discarded but an Ack with next expected sequence number sent.

- Time Out:

- Time out leads to retransmission of segment.
- When segment sent, it is stored in the queue until Acked.
- When timer expires and Ack not came retransmition occur.
- When sender recieves 3 duplicate Ack, retransmission occur.

(1) Retransmission after RTO:

* Sending TCP maintain retransmission time out (RTO) for each connect^{n}.

* When timer time out, TCP resend the segment in front of queue and restart timer.

* RTO is based on RTT.

(2) Retransmission after 3 duplicate ACK.

* Sender can retransmit without waiting for time out.
* When sender recieves 3 duplicate ACK, it immediately retransmit the segment.
* It is called FAST RETRANSMISSION.
* Three duplicate ACK → One original and 3 identical copies.

- TCP does not discard out-of-order segment
- " stores them and deliver when all missing segment arrives.

Normal operation in TCP:

- Fig. 3.61. (Slide no.
- client TCP sends one segment and server TCP send 3 segments.
- Client send ACK after a delay of 500 ms.

Lost ACK Scenario:

- Fig. 3.64.

Lost Segment Scenario:

- fig. 3.62.

Fast Retransmission scenario:

- Fig. 3.63.

Lost ACK corrected by resending:

- Fig. 3.65.

Q-1 – Host A and B are communicating over a TCP connection. Host B has recieved all bytes up to 120. Suppose host A then sends two segments to Host B back-to-back. The first and second segment contain 80 and 50 Bytes of data. Host B send Ack when it recieve segment from Host A.

i. If first segment arrive before 2nd segment, what is the sequence number of arriving segment? What should be the Ack number that should be sent for 1st segment?

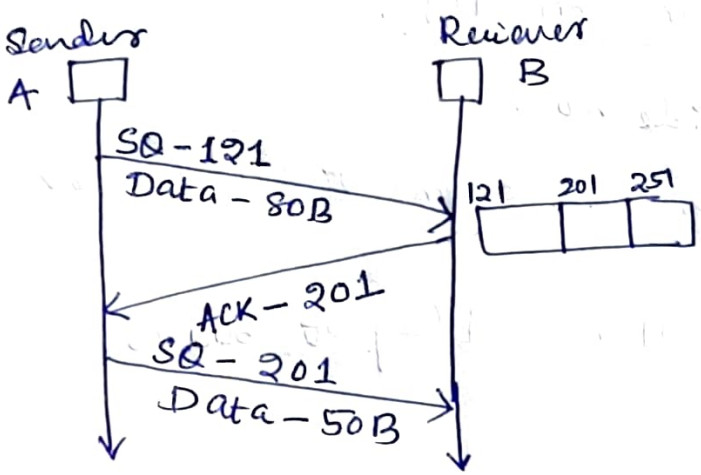
ii. If the second segment arrive before 1st segment, what is the Ack that should be sent?

Sol^n – All byts upto 120Byte recieved

Segment 1 contain – 80 Byte

" 2 " – 50 "

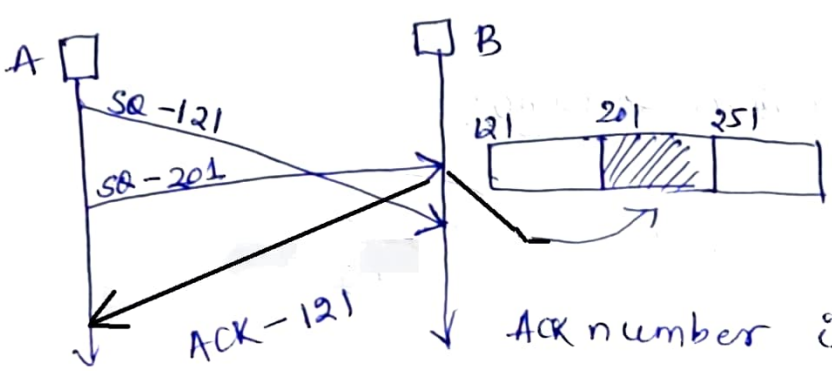
(i)



Sequence number of first segment – 121

Ack " Sent is 201

(ii)



Ack number is 121.