Unit-5

1) Difference between TCP vs UDP? **

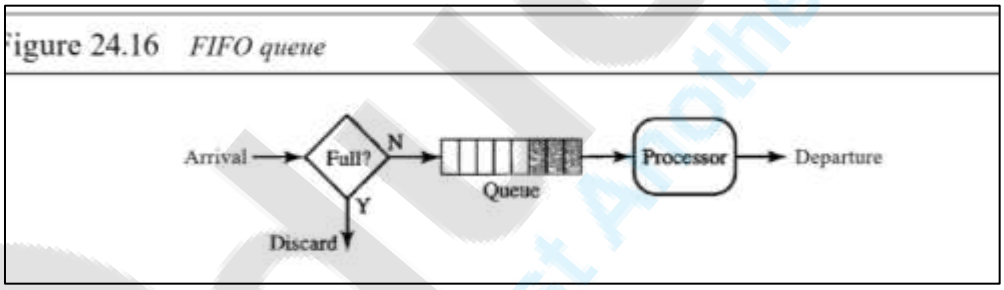
Answer: -

	TCP	**UDP**
Acronym for	Transmission Control Protocol	User Datagram Protocol or Universal Datagram Protocol
Connection	TCP is a connection-oriented protocol.	UDP is a connectionless protocol.
Function	As a message makes its way across the internet from one computer to another. This is connection based.	UDP is also a protocol used in message transport or transfer. This is not connection based which means that one program can send a load of packets to another and that would be the end of the relationship.
Usage	TCP is suited for applications that require high reliability, and transmission time is relatively less critical.	UDP is suitable for applications that need fast, efficient transmission, such as games. UDP's stateless nature is also useful for servers that answer small queries from huge numbers of clients.
Use by other protocols	HTTP, HTTPs, FTP, SMTP, Telnet	DNS, DHCP, TFTP, SNMP, RIP, VOIP.
Ordering of data packets	TCP rearranges data packets in the order specified.	UDP has no inherent order as all packets are independent of each other. If ordering is required, it has to be managed by the application layer.
Speed of transfer	The speed for TCP is slower than UDP.	UDP is faster because error recovery is not attempted. It is a "best effort" protocol.
Reliability	There is absolute guarantee that the data transferred remains intact and arrives in the same order in which it was sent.	There is no guarantee that the messages or packets sent would reach at all.
Header Size	TCP header size is 20 bytes	UDP Header size is 8

		bytes.
Weight	TCP is heavy-weight. TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control.	UDP is lightweight. There is no ordering of messages, no tracking connections, etc. It is a small transport layer designed on top of IP.
Acknowledgement	Acknowledgement segments	No Acknowledgment

2) Explain the different queue management algorithm used in routers? **
OR
Describe various queue management algorithms used in TCP? **
Answer: -

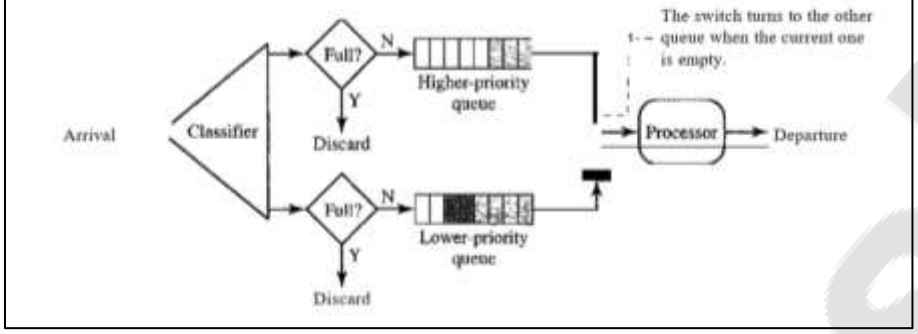
1) **Scheduling:** -Packets from different flows arrive at a switch or router for processing. A good scheduling technique treats the different flows in a fair and appropriate manner. Several scheduling techniques are designed to improve the quality of service. We discuss three of them here: **FIFO queuing, priority queuing, and weighted fair queuing.**
2) **FIFO Queuing: -**



A) In first-in, first-out (FIFO) queuing, packets wait in a buffer (queue) until 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 fill 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. Figure 24.16 shows a conceptual view of a FIFO queue.

3) **Priority Queuing: -**

Figure 24.17 *Priority queuing*

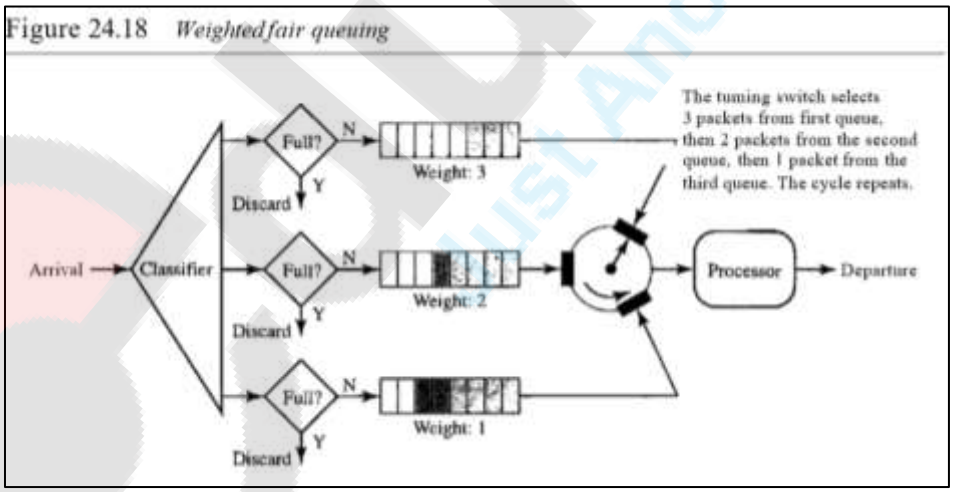


A) In priority queuing, packets are first assigned to a priority class. Each priority class has its own queue. The packets in the highest-priority queue are processed first. Packets in the lowest-priority queue are processed last. Note that the system does not stop serving a queue until it is empty.

B) Figure 24.17 shows priority queuing with two priority levels (for simplicity). A priority queue can provide better QoS than the FIFO queue because higher priority traffic, such as multimedia, can reach the destination with less delay. However, there is a potential drawback. If there is a continuous flow in a high-priority queue, the packets in the lower-priority queues will never have a chance to be processed. This is a condition called starvation.

4) **Weighted Fair Queuing: -**

Figure 24.18 *Weighted fair queuing*



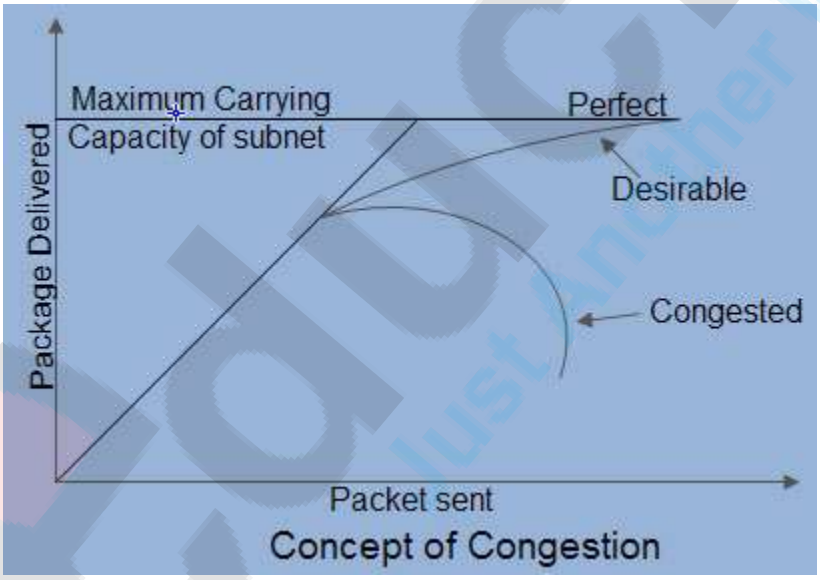
A) A better scheduling method is weighted fair queuing. In this technique, the packets are still assigned to different classes and admitted to different queues. The queues, however, are weighted based on the priority of the queues; higher priority means a higher weight.

B) The system processes packets in each queue in a round-robin fashion with the number of packets selected from each queue based on the corresponding weight.

For example, if the weights are 3, 2, and 1, three packets are processed from the first queue, two from the second queue, and one from the third queue. If the system does not impose priority on the classes, all weights can be equal In this way, we have fair queuing with priority.

3) What is congestion? Explain the congestion control in TCP? **
OR
What is Congestion control? How it is different form flow control mechanism. Explain leaky bucket algorithm to deal with congestion? **
Answer: -

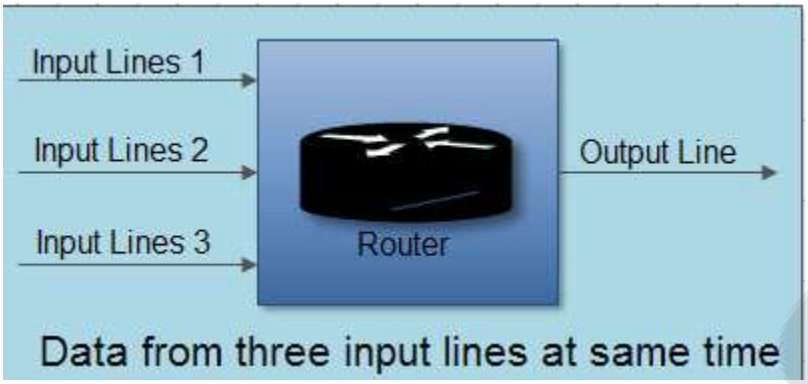
1) Congestion is an important issue that can arise in packet switched network. Congestion is a situation in Communication Networks in which too many packets are present in a part of the subnet, performance degrades. Congestion in a network may occur when the load on the network *(i.e.* the number of packets sent to the network) is greater than the capacity of the network *(i.e.* the number of packets a network can handle.)In other words when too much traffic is offered, congestion sets in and performance degrades sharply



2) **Causing of Congestion:** The various causes of congestion in a subnet are:

A) The input traffic rate exceeds the capacity of the output lines. If suddenly, a stream of packet start arriving on three or four input lines and all need the same output line. In this case, a queue will be built up. If there is insufficient memory to hold all the packets, the packet will be lost. Increasing the memory to unlimited size does not solve the problem. This is because, by the time packets reach front of the queue, they have already timed out (as they waited the queue). When timer goes off source transmits duplicate packet that are also added to the queue. Thus

same packets are added again and again, increasing the load all the way to the destination.



B) The routers are too slow to perform bookkeeping tasks (queuing buffers, updating tables, etc.).

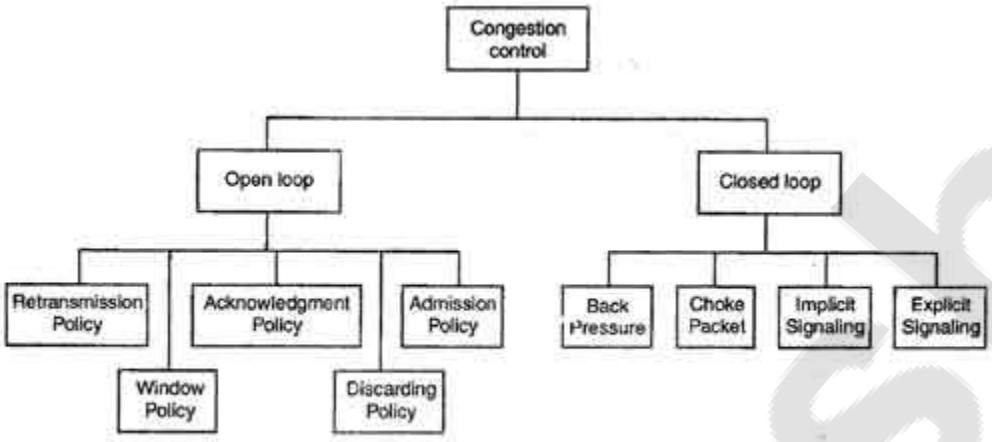
C) The routers' buffer is too limited.

D) Congestion in a subnet can occur if the processors are slow. Slow speed CPU at routers will perform the routine tasks such as queuing buffers, updating table etc slowly. As a result of this, queues are built up even though there is excess line capacity.

E) Congestion is also caused by slow links. This problem will be solved when high speed links are used. But it is not always the case. Sometimes increase in link bandwidth can further deteriorate the congestion problem as higher speed links may make the network more unbalanced. Congestion can make itself worse. If a route!" does not have free buffers, it starts ignoring/discarding the newly arriving packets. When these packets are discarded, the sender may retransmit them after the timer goes off. Such packets are transmitted by the sender again and again until the source gets the acknowledgement of these packets. Therefore multiple transmissions of packets will force the congestion to take place at the sending end.

3) **How to correct the Congestion Problem:**

A) Congestion Control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened. Congestion control mechanisms are divided into two categories, one category prevents the congestion from happening and the other category removes congestion after it has taken place.



Types of Congestion Control Methods

1) Open loop
2) Closed loop

A) **Open Loop Congestion Control**

a) In this method, policies are used to prevent the congestion before it happens.
b) Congestion control is handled either by the source or by the destination.
c) The various methods used for open loop congestion control are:

1) **Retransmission Policy**

- The sender retransmits a packet, if it feels that the packet it has sent is lost or corrupted.
- However retransmission in general may increase the congestion in the network. But we need to implement good retransmission policy to prevent congestion.
- The retransmission policy and the retransmission timers need to be designed to optimize efficiency and at the same time prevent the congestion.

2) **Window Policy**

- To implement window policy, selective reject window method is used for congestion control.
- Selective Reject method is preferred over Go-back-n window as in Go-back-n method, when timer for a packet times out, several packets are resent, although some may have arrived safely at the receiver. Thus, this duplication may make congestion worse.

- Selective reject method sends only the specific lost or damaged packets.

3) **Acknowledgement Policy**

- The acknowledgement policy imposed by the receiver may also affect congestion.
- If the receiver does not acknowledge every packet it receives it may slow down the sender and help prevent congestion.
- Acknowledgments also add to the traffic load on the network. Thus, by sending fewer acknowledgements we can reduce load on the network.
- To implement it, several approaches can be used:
- A receiver may send an acknowledgement only if it has a packet to be sent.
- A receiver may send an acknowledgement when a timer expires.
- A receiver may also decide to acknowledge only *N* packets at a time.

4) **Discarding Policy**

- A router may discard less sensitive packets when congestion is likely to happen.
- Such a discarding policy may prevent congestion and at the same time may not harm the integrity of the transmission.

5) **Admission Policy**

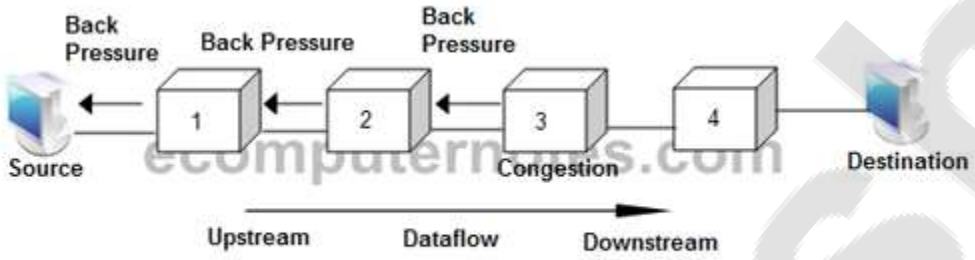
- An admission policy, which is a quality-of-service mechanism, can also prevent congestion in virtual circuit networks.
- Switches in a flow first check the resource requirement of a flow before admitting it to the network.
- A router can deny establishing a virtual circuit connection if there is congestion in the "network or if there is a possibility of future congestion.

B) **Closed Loop Congestion Control**

a) Closed loop congestion control mechanisms try to remove the congestion after it happens.
b) The various methods used for closed loop congestion control are:

1) **Backpressure**

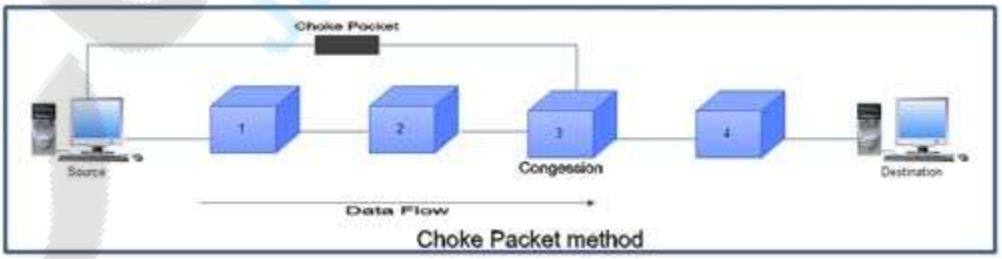
- Backpressure is a node-to-node congestion control that starts with a node and propagates, in the opposite direction of data flow.



Backpressure Method

- The backpressure technique can be applied only to virtual circuit networks. In such virtual circuit each node knows the upstream node from which a data flow is coming.
- In this method of congestion control, the congested node stops receiving data from the immediate upstream node or nodes.
- This may cause the upstream node on nodes to become congested, and they, in turn, reject data from their upstream node or nodes.
- As shown in fig node 3 is congested and it stops receiving packets and informs its upstream node 2 to slow down. Node 2 in turns may be congested and informs node 1 to slow down. Now node 1 may create congestion and informs the source node to slow down. In this way the congestion is alleviated. Thus, the pressure on node 3 is moved backward to the source to remove the congestion.

2) Choke Packet



- In this method of congestion control, congested router or node sends a special type of packet called choke packet to the source to inform it about the congestion.
- Here, congested node does not inform its upstream node about the congestion as in backpressure method.

- In choke packet method, congested node sends a warning directly to the source station *i.e.* the intermediate nodes through which the packet has traveled are not warned.

3) **Implicit Signaling**

- In implicit signaling, there is no communication between the congested node or nodes and the source.
- The source guesses that there is congestion somewhere in the network when it does not receive any acknowledgment. Therefore the delay in receiving an acknowledgment is interpreted as congestion in the network.
- On sensing this congestion, the source slows down.
- This type of congestion control policy is used by TCP.

4) **Explicit Signaling**

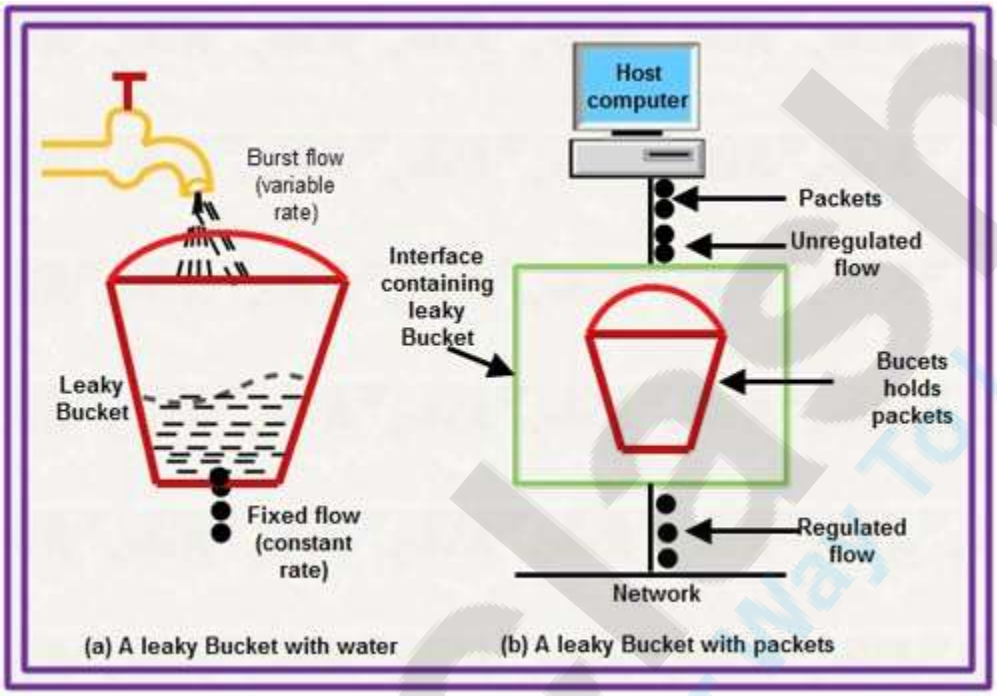
- In this method, the congested nodes explicitly send a signal to the source or destination to inform about the congestion.
- Explicit signaling is different from the choke packet method. In choke packed method, a separate packet is used for this purpose whereas in explicit signaling method, the signal is included in the packets that carry data.
- Explicit signaling can occur in either the forward direction or the backward direction.
- In backward signaling, a bit is set in a packet moving in the direction opposite to the congestion. This bit warns the source about the congestion and informs the source to slow down.
- In forward signaling, a bit is set in a packet moving in the direction of congestion. This bit warns the destination about the congestion. The receiver in this case uses policies such as slowing down the acknowledgements to remove the congestion.

3) Congestion control algorithms

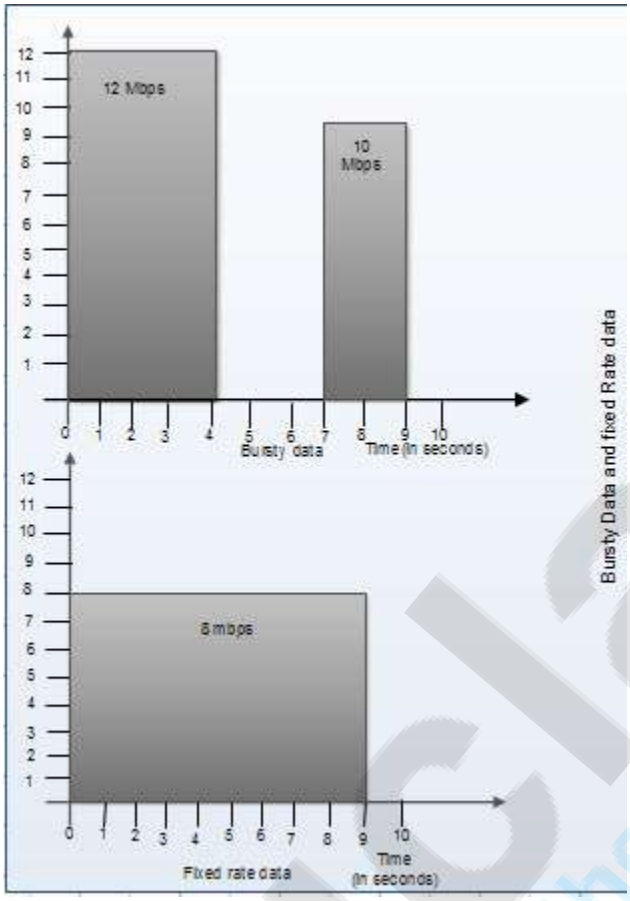
A) Leaky Bucket Algorithm

- It is a traffic shaping mechanism that controls the amount and the rate of the traffic sent to the network.
- A leaky bucket algorithm shapes bursty traffic into fixed rate traffic by averaging the data rate.
- Imagine a bucket with a small hole at the bottom.
- The rate at which the water is poured into the bucket is not fixed and can vary but it leaks from the bucket at a constant rate. Thus (as long as water is present in bucket), the rate at which the water

leaks does not depend on the rate at which the water is input to the bucket.



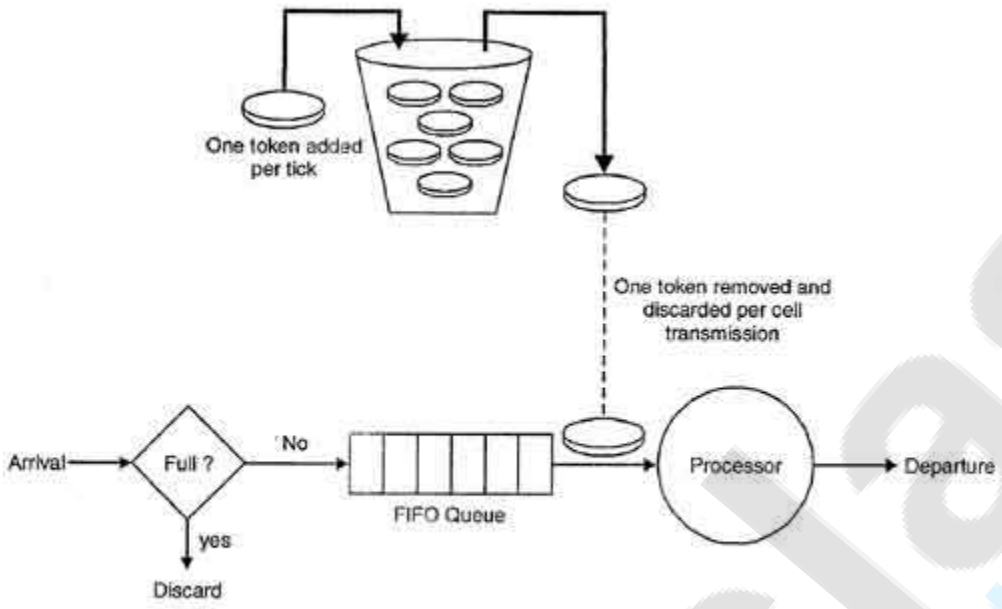
- Also, when the bucket is full, any additional water that enters into the bucket spills over the sides and is lost.
- The same concept can be applied to packets in the network. Consider that data is coming from the source at variable speeds. Suppose that a source sends data at 12 Mbps for 4 seconds. Then there is no data for 3 seconds. The source again transmits data at a rate of 10 Mbps for 2 seconds. Thus, in a time span of 9 seconds, 68 Mb data has been transmitted.
- If a leaky bucket algorithm is used, the data flow will be 8 Mbps for 9 seconds. Thus constant flow is maintained.



2) Token bucket Algorithm

- The leaky bucket algorithm allows only an average (constant) rate of data flow. Its major problem is that it cannot deal with bursty data.
- A leaky bucket algorithm does not consider the idle time of the host. For example, if the host was idle for 10 seconds and now it is willing to sent data at a very high speed for another 10 seconds, the total data transmission will be divided into 20 seconds and average data rate will be maintained. The host is having no advantage of sitting idle for 10 seconds.
- To overcome this problem, a token bucket algorithm is used. A token bucket algorithm allows bursty data transfers.
- A token bucket algorithm is a modification of leaky bucket in which leaky bucket contains tokens.
- In this algorithm, a token(s) are generated at every clock tick. For a packet to be transmitted, system must remove token(s) from the bucket.
- Thus, a token bucket algorithm allows idle hosts to accumulate credit for the future in form of tokens.
- For example, if a system generates 100 tokens in one clock tick and the host is idle for 100 ticks. The bucket will contain 10,000 tokens.
- Now, if the host wants to send bursty data, it can consume all 10,000 tokens at once for sending 10,000 cells or bytes.

- Thus a host can send bursty data as long as bucket is not empty.



Token bucket algorithm

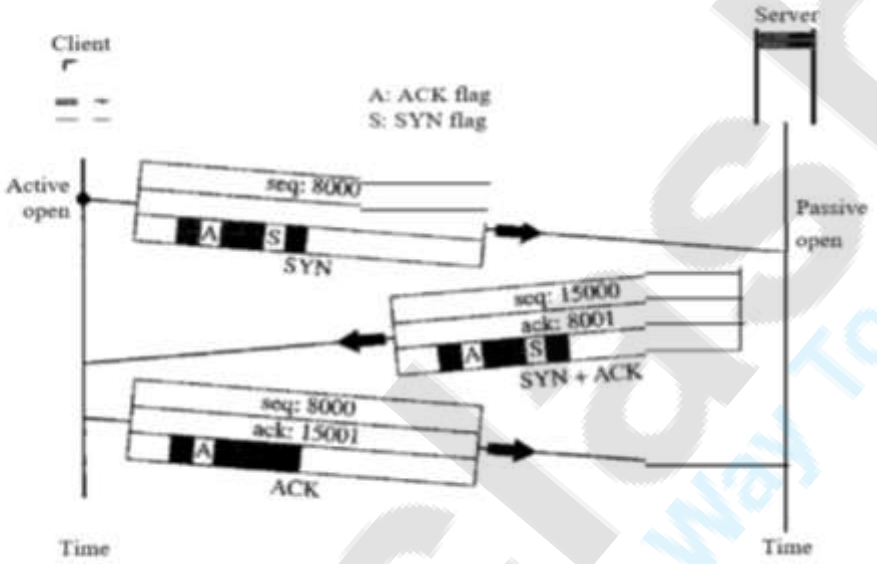
4) Explain the 4-way termination process of TCP connection termination? **

Answer: -

1) TCP transmits data in full-duplex mode. When two TCPs in two machines are connected, they are able to send segments to each other simultaneously. This implies that each party must initialize communication and get approval from the other party before any data are transferred.
2) The process starts with the server. The server program tells its TCP that it is ready to accept a connection. This is called a request for a passive open. Although the server TCP is ready to accept any connection from any machine in the world, it cannot make the connection itself.

3) The client program issues a request for an active open. A client that wishes to connect to an open server tells its TCP that it needs to be connected to that particular server.

Figure 23.18 *Connection establishment using three-way handshaking*



Step 1 : -The client sends the first segment, a SYN segment, in which only the SYN flag is set. This segment is for synchronization of sequence numbers. It consumes one sequence number. When the data transfer starts, the sequence number is incremented by 1. We can say that the SYN segment carries no real data, but we can think of it as containing 1 imaginary byte.

Step 2: -The server sends the second segment, a SYN +ACK segment, with 2 flag bits set: SYN and ACK. This segment has a dual purpose. It is a SYN segment for communication in the other direction and serves as the acknowledgment for the SYN segment. It consumes one sequence number.

Step 3: -The client sends the third segment. This is just an ACK segment. It acknowledges the receipt of the second segment with the ACK flag and acknowledgment number field. Note that the sequence number in this segment is the same as the one in the SYN segment; the ACK segment does not consume any sequence numbers.

5) **What is quality of service? What are the methods used in provide QoS? ****

Answer: -

1) The notion of quality of service, or QoS, concerns certain characteristics of a network connection under the sole of the network service provider liability.

2) A QoS value applies to the whole of a network connection. It must be identical at both ends of the connection, even if it is supported by several interconnected subnetworks each offering different services.
3) QoS is described by parameters. Defining a QoS parameter indicates how to measure or determine its value, mentioning if necessary the events specified by the network service primitives.
4) **Two types of QoS parameters have been defined:**
   A) Those whose values are transmitted peer users via the Network service during the establishment phase of the network connection. During this transmission, a tripartite negotiation can take place between users and the network service provider to define a value for the QoS parameters.
   B) Those whose values are transmitted or negotiated between users and network service provider. For these QoS parameters, it is possible to obtain, by local means, information on the value to the supplier and values to each user of the network service.
5) **The main QoS parameters are:**
   A) **Time of establishment of the network connection**. Is the time that elapses between a network connection request and confirmation of the connection? This QoS parameter indicates the maximum time acceptable to the user.
   B) **Probability of failure of the establishment of the network connection**. This probability is established from the applications which have not been met in the normal time limit for establishing the connection.
   C) **Flow data transfer**. The flow rate defines the number of bytes transported over a network connection in a reasonably long time (a few minutes, a few hours or days). The difficulty in determining the speed of a connection network comes from the asynchronous transport packets. To obtain a value acceptable, observe the network on a sequence of several packages and consider number of bytes of data transported taking into account the elapsed time since the application or the data transfer indication.
   D) **Transit time when transferring data**. The transit time corresponds to elapsed time between a data transfer request and indicating transfer of data. This transit time is difficult to calculate because of the geographical distribution ends. The satisfaction of a quality service on the transit time may moreover contradict flow control.
   E) **Residual error rate**. Is calculated from the number of packets that arrive erroneous, lost or duplicated on the total number of transmitted packets. It is a rate Error packet. Also denotes the probability that a packet does not arrive correctly to the receiver.
   F) **Transfer Probability incident**. Is obtained by the ratio of the number of incidents listed on the total number of transfer taken. To have a correct estimate

of this probability, just consider the number of network disconnection relative to the number of transfer taken.

G) **Probability of failure of the network connection**. Is calculated from the number of release and resetting of a network connection based on the number of transfer made.

H) **Release time the network connection**. This is the maximum acceptable delay between a disconnection request and the actual release.

I) Probability of failure upon release of the network connection. The number Liberation of failure required by the total number requested release.

6) **The following three additional parameters used to characterize the quality of Service:**

A) **Protection of the network connection**. Determines the probability that the network connection be in working order throughout the period when it is opened by the user. There is ways to protect a connection by duplicating or having a Backup connection ready to be opened in case of failure. The value for a telephone network is 99.999%, the so-called five nines, equivalent to a few minutes of downtime per year. The protection is much lower for an IP network, with a value of the order of 99.9%, three or nine. This value arises besides problem for IP telephony, which requires stronger protection telephone connections.

B) **Priority of the network connection**. Determines priority of access to a connection network, the holding priority of a network connection and priority of data connection.

C) **Maximum acceptable cost**. Determines if the network connection is tolerable or not. The definition of the cost is quite complex since it depends on the use of resources for the establishment, maintenance and release of the connection network.

D) **Flow Characteristics** Traditionally, four types of characteristics are attributed to a flow: reliability, delay, jitter and bandwidth.

a) Reliability

- Reliability is an important characteristic of flow.
- Lack of reliability means losing a packet or acknowledgement which then requires retransmission.
- However, the sensitivity of application programs to reliability is not the same. For example, it is more important that electronic mail, file transfer, and internet access have reliable transmissions than audio conferencing or telephony.

b) Delay

- Source to destination delay is another flow characteristic.

- Applications can tolerate delay in different degrees. In this case, telephony, audio conferencing, video conferencing and remote log in need minimum delay while delay in file transfer or e-mail is less important.

c) Jitter

- Jitter is defined as the variation in delay for packets belonging to the same flow.
- High Jitter means the difference between delays is large and low jitter means the variation is small.
- For example, if four packets depart at times 0, 1, 2, 3 and arrive at 20, 21,22, 23, all have same delay, 20 units of time. On the other hand, if the above four packets arrive at 21, 23, 21, and 28 they will have different delays of21, 22, 19 and 24.

d) Bandwidth

- Different applications need different bandwidths.
- In video conferencing we need to send millions of bits per second to refresh a color screen while the total number of bits in an email may not reach even a million.

7) **TECHNIQUES TO IMPROVE QoS**

A) In this section, we discuss some techniques that can be used to improve the quality of service. We briefly discuss four common methods: **scheduling, traffic shaping, admission control, and resource reservation.**

B) **Scheduling: - Refer Question No- 4**

C) **Traffic Shaping: - Refer Question No- 14**

D) **Resource Reservation:** -A flow of data needs resources such as a buffer, bandwidth, CPU time, and so on. The quality of service is improved if these resources are reserved beforehand. We discuss in this section one QoS model called Integrated Services, which depends heavily on resource reservation to improve the quality of service.

E) **Admission Control: -** Admission control refers to the mechanism used by a router, or a switch, to accept or reject a flow based on predefined parameters called flow specifications. Before a router accepts a flow for processing, it checks the flow specifications to see if its capacity (in terms of bandwidth, buffer size, CPU speed, etc.) and its previous commitments to other flows can handle the new flow.

6) Explain the timers used in TCP?

Answer: -

1) TCP implementations use at least four timers as shown in Figure 15.38.

2) **Retransmission Timer: -** To retransmit lost segments, TCP employs one retransmission timer (for the whole connection period) that handles the retransmission time-out (RTO), the waiting time for an acknowledgment of a segment. We can define the following rules for the retransmission timer:

A) When TCP sends the segment in front of the sending queue, it starts the timer.

B) When the timer expires, TCP resends the first segment in front of the queue, and restarts the timer.

C) When a segment (or segments) is cumulatively acknowledged, the segment (or segments) is purged from the queue.

D) If the queue is empty, TCP stops the timer; otherwise, TCP restarts the timer.

3) **Persistence Timer: -** To deal with a zero-window-size advertisement, TCP needs another timer. If the receiving TCP announces a window size of zero, the sending TCP stops transmitting segments until the receiving TCP sends an ACK segment announcing a nonzero window size. This ACK segment can be lost. Remember that ACK segments are not acknowledged nor retransmitted in TCP. If this acknowledgment is lost, the receiving TCP thinks that it has done its job and waits for the sending TCP to send more segments. There is no retransmission timer for a segment containing only an acknowledgment. The sending TCP has not received an acknowledgment and waits for the other TCP to send an acknowledgment advertising the size of the window. Both TCPs might continue to wait for each other forever (a deadlock). To correct this deadlock, TCP uses a persistence timer for each connection. When the sending TCP receives an acknowledgment with a window size of zero, it starts a persistence timer. When the persistence timer goes off, the sending TCP sends a special segment called a probe. This segment contains only 1 byte of new data. It has a sequence number, but its sequence number is never acknowledged; it is even ignored in calculating the sequence number for the rest of the data. The probe causes the receiving TCP to resend the acknowledgment.

4) **Keep-alive Timer: -** A keep-alive timer is used in some implementations to prevent a long idle connection between two TCPs. Suppose that a client opens a TCP connection to a server, transfers some data, and becomes silent. Perhaps the client has crashed. In this case, the connection remains open forever. To remedy this situation, most implementations equip a server with a keep-alive timer. Each time the server hears from a client, it resets this timer. The time-out is usually 2 hours. If the server does not hear from the client after 2 hours, it sends a probe segment. If there is no response after 10 probes, each of which is 75 s apart, it assumes that the client is down and terminates the connection.

5) **TIME-WAIT Timer: -** The TIME-WAIT (2MSL) timer is used during connection termination.

7) Explain the 3-way handshake method for TCP connection establishment?

Answer: -

1) TCP transmits data in full-duplex mode. When two TCPs in two machines are connected, they are able to send segments to each other simultaneously. This implies that each party must initialize communication and get approval from the other party before any data are transferred.

2) The connection establishment in TCP is called three-way handshaking. In our example, an application program, called the client, wants to make a connection with another application program, called the server, using TCP as the transport layer protocol.
3) The process starts with the server. The server program tells its TCP that it is ready to accept a connection. This request is called a **passive open**. Although the server TCP is ready to accept a connection from any machine in the world, it cannot make the connection itself.
4) The client program issues a request for an **active open.** A client that wishes to connect to an open server tells its TCP to connect to a particular server. TCP can now start the three-way handshaking process as shown in Figure 15.9.
5) The three steps in this phase are as follows.

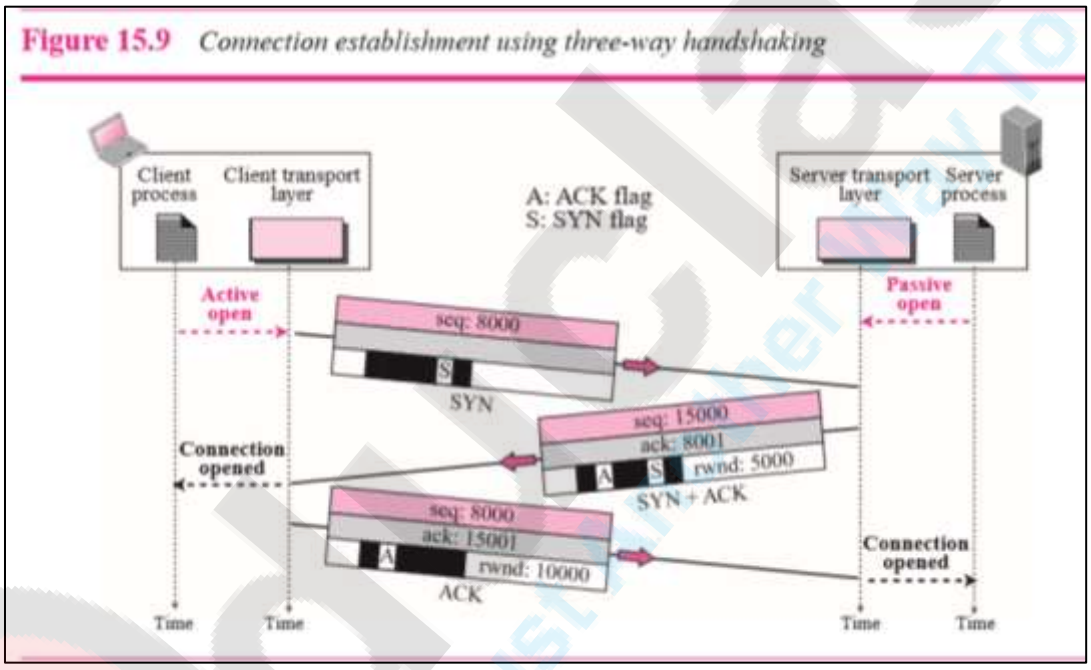


Figure 15.9 *Connection establishment using three-way handshaking*

6) **Step-1: -** The client sends the first segment, a SYN segment, in which only the SYN flag is set. This segment is for synchronization of sequence numbers. The client in our example chooses a random number as the first sequence number and sends this number to the server. This sequence number is called the initial sequence number (ISN). Note that this segment does not contain an acknowledgment number. It does not define the window size either; a window size definition makes sense only when a segment includes an acknowledgment.
7) **Step-2: -** The server sends the second segment, a SYN + ACK segment with two flag bits set: SYN and ACK. This segment has a dual purpose. First, it is a SYN segment for communication in the other direction. The server uses this segment to initialize a sequence number for numbering the bytes sent from the server to the client. The server also acknowledges the receipt of the SYN segment from the client by setting the ACK flag and displaying the next sequence number it expects to receive from the

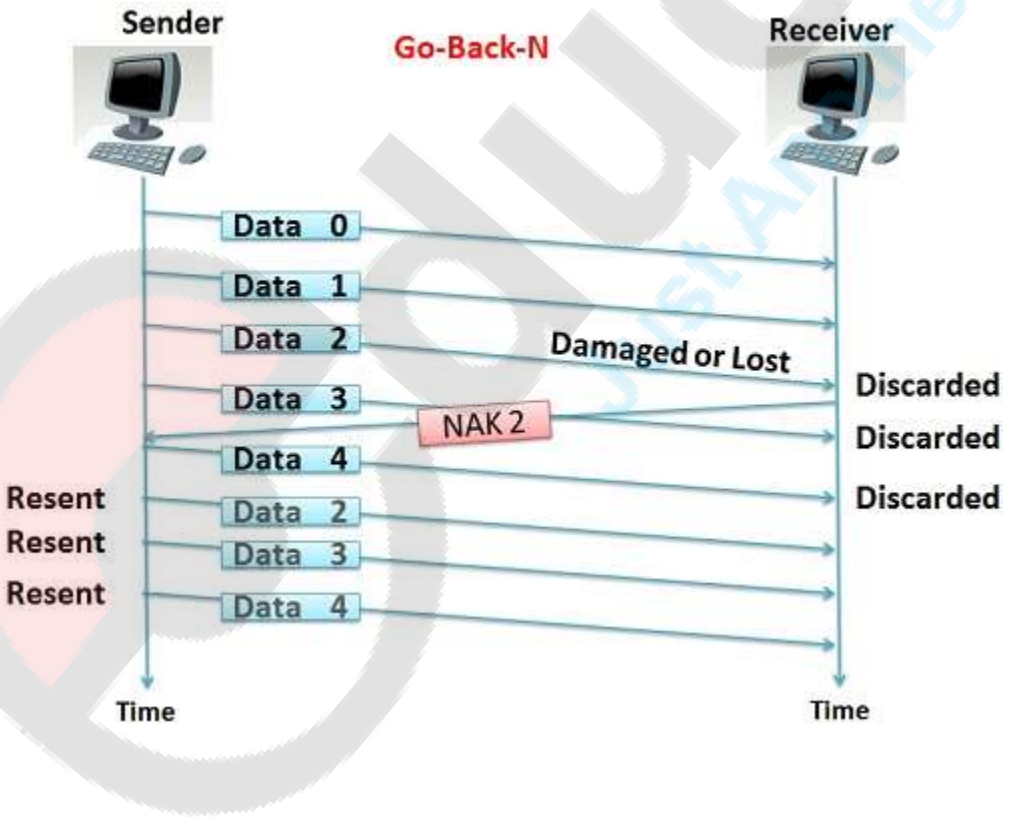
client. Because it contains an acknowledgment, it also needs to define the receive window

8) **Step-3: -** The client sends the third segment. This is just an ACK segment. It acknowledges the receipt of the second segment with the ACK flag and acknowledgment number field. Note that the sequence number in this segment is the same as the one in the SYN segment; the ACK segment does not consume any sequence numbers. The client must also define the server window size. Some implementations allow this third segment in the connection phase to carry the first chunk of data from the client. In this case, the third segment must have a new sequence number showing the byte number of the first byte in the data. In general, the third segment usually does not carry data and consumes no sequence numbers.

8) Explain Go Back N and Selective Repeat Protocol? Difference between Go Back N and Selective Repeat?

Answer: -

1) **Definition of Go-Back-N:** -Go-Back-N protocol is a sliding window protocol. It is a mechanism to detect and control the error in datalink layer. During transmission of frames between sender and receiver, if a frame is damaged, lost, or an acknowledgement is lost then the action performed by sender and receiver is explained in the following content.



2) **Damaged Frame**

   a) If a receiver receives a damaged frame or if an error occurs while receiving a frame then, the receiver sends the NAK (negative acknowledgement) for that frame along with that frame number, that it expects to be retransmitted. After sending NAK, the receiver discards all the frames that it receives, after a damaged frame. The receiver does not send any ACK (acknowledgement) for the discarded frames. After the sender receives the NAK for the damaged frame, it retransmits all the frames onwards the frame number referred by NAK.

3) **Lost frame**

   a) The receiver checks the number on each frame, it receives. If a frame number is skipped in a sequence, then the receiver easily detects the loss of a frame as the newly received frame is received out of sequence. The receiver sends the NAK for the lost frame and then the receiver discards all the frames received after a lost frame. The receiver does not send any ACK (acknowledgement) for that discarded frames. After the sender receives the NAK for the lost frame, it retransmits the lost frame referred by NAK and also retransmits all the frames which it has sent after the lost frame.

4) **Lost Acknowledgement**

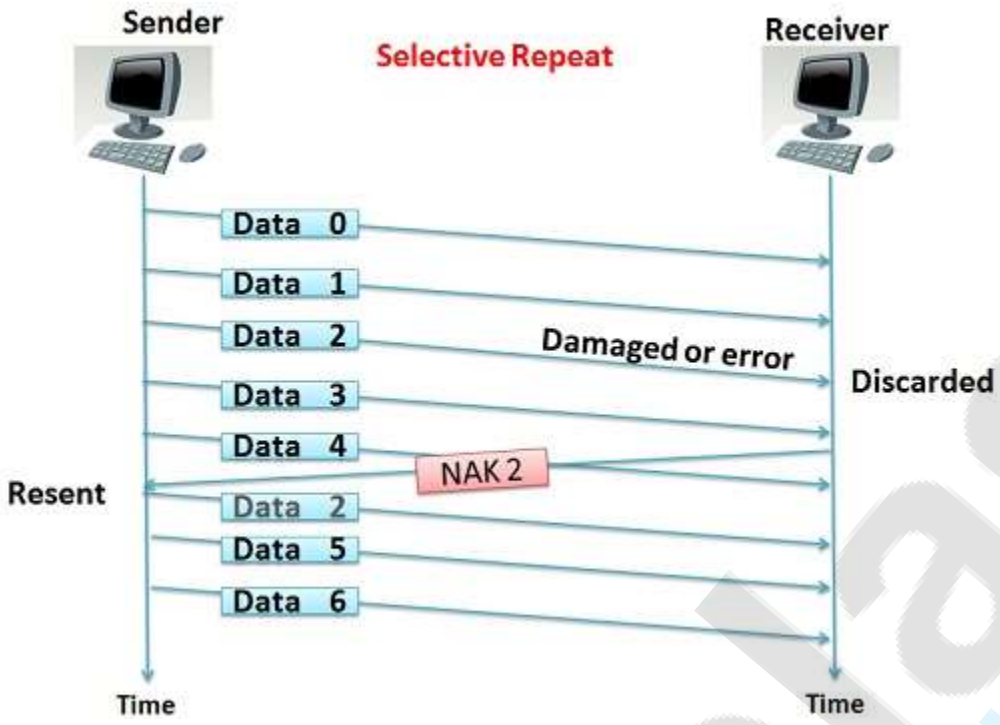
   a) If the sender does not receive any ACK or if the ACK is lost or damaged in between the transmission. The sender waits for the time to run out and as the time run outs, the sender retransmits all the frames for which it has not received the ACK. The sender identifies the loss of ACK with the help of a timer.

   b) The ACK number, like NAK (negative acknowledgement) number, shows the number of the frame, that receiver expects to be the next in sequence. The window size of the receiver is 1 as the data link layers only require the frame which it has to send next to the network layer. The sender window size is equal to 'w'. If the error rate is high, a lot of bandwidth is lost wasted.

5) **Definition of Selective Repeat: -**Selective repeat is also the sliding window protocol which detects or corrects the error occurred in datalink layer. The selective repeat protocol retransmits only that frame which is damaged or lost. In selective repeat protocol, the retransmitted framed is received out of sequence. The selective repeat protocol can perform following actions

   A) The receiver is capable of sorting the frame in a proper sequence, as it receives the retransmitted frame whose sequence is out of order of the receiving frame.
   B) The sender must be capable of searching the frame for which the NAK has been received.

C) The receiver must contain the buffer to store all the previously received frame on hold till the retransmitted frame is sorted and placed in a proper sequence.
D) The ACK number, like NAK number, refers to the frame which is lost or damaged.
E) It requires the less window size as compared to go-back-n protocol.



A) **Damaged frames**

1) If a receiver receives a damaged frame, it sends the NAK for the frame in which error or damage is detected. The NAK number, like in go-back-n also indicate the acknowledgement of the previously received frames and error in the current frame. The receiver keeps receiving the new frames while waiting for the damaged frame to be replaced. The frames that are received after the damaged frame are not be acknowledged until the damaged frame has been replaced.

B) **Lost Frame**

1) As in a selective repeat protocol, a frame can be received out of order and further they are sorted to maintain a proper sequence of the frames. While sorting, if a frame number is skipped, the receiver recognize that a frame is lost and it sends NAK for that frame to the sender. After receiving NAK for the lost frame the sender searches that frame in its window and retransmits that frame. If the last transmitted frame is lost then receiver does not respond and this silence is a negative acknowledgement for the sender.

C) **Lost Acknowledgement**

1) If the sender does not receive any ACK or the ACK is lost or damaged in between the transmission. The sender waits for the time to run out and as the time run outs, the sender retransmit all the frames for which it has not received the ACK. The sender identifies the loss of ACK with the help of a timer.

D) **Key Differences Between Go-Back-N and Selective Repeat**

1. Go-Back-N protocol is design to retransmit all the frames that are arrived after the damaged or a lost frame. On the other hand, Selective Repeat protocol retransmits only that frame that is damaged or lost.
2. If the error rate is high i.e. more frames are being damaged and then retransmitting all the frames that arrived after a damaged frame waste the lots of bandwidth. On the other hand, selective repeat protocol re-transmits only damaged frame hence, minimum bandwidth is wasted.
3. All the frames after the damaged frame are discarded and the retransmitted frames arrive in a sequence from a damaged frame onwards, so, there is less headache of sorting the frames hence it is less complex. On the other hand only damaged or suspected frame is retransmitted so, extra logic has to be applied for sorting hence, it is more complicated.
4. Go-Back-N has a window size of N-1 and selective repeat have a window size <=(N+1)/2.
5. Neither sender nor receiver need the sorting algorithm in Go-Back-N whereas, receiver must be able to sort the as it has to maintain the sequence.
6. In Go-Back-N receiver discards all the frames after the damaged frame hence, it don't need to store any frames. Selective repeat protocol does not discard the frames arrived after the damaged frame instead it stores those frames till the damaged frame arrives successfully and is sorted in a proper sequence.
7. In selective repeat NAK frame refers to the damaged frame number and in Go-Back-N, NAK frame refers to the next frame expected.
8. Generally the Go-Back-N is more is use due to its less complex nature instead of Selective Repeat protocol.

E) **Conclusion:**

The selective repeat is a more efficient protocol as it does not waste bandwidth for the frames which are properly received but, its complexity and expense favors the use of the go-back-n protocol.

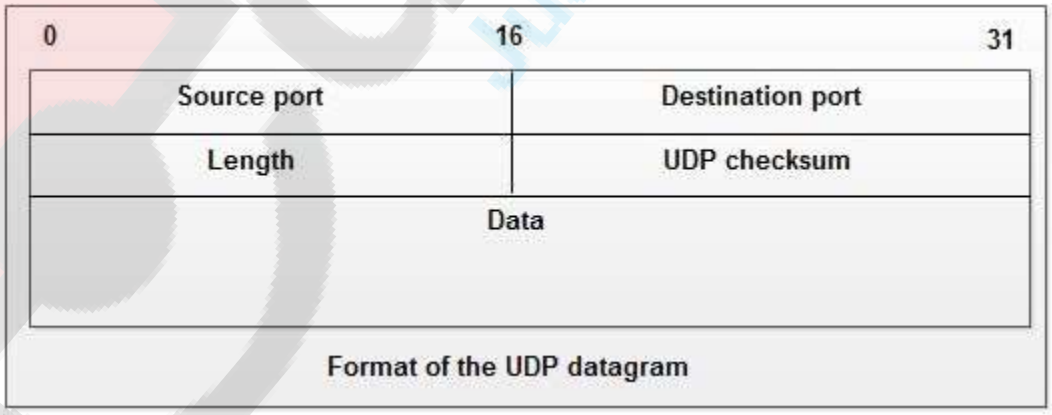
BASIS FOR COMPARISON	GO-BACK-N	SELECTIVE REPEAT
Basic	Retransmits all the frames that sent after the frame which suspects to be damaged or lost.	Retransmits only those frames that are suspected to lost or damaged.
Bandwidth Utilization	If error rate is high, it wastes a lot of bandwidth.	Comparatively less bandwidth is wasted in retransmitting.
Complexity	Less complicated.	More complex as it require to apply extra logic and sorting and storage, at sender and receiver.
Window size	N-1	<= (N+1)/2
Sorting	Sorting is neither required at sender side nor at receiver side.	Receiver must be able to sort as it has to maintain the sequence of the frames.
Storing	Receiver do not store the frames received after the damaged frame until the damaged frame is retransmitted.	Receiver stores the frames received after the damaged frame in the buffer until the damaged frame is replaced.
Searching	No searching of frame is required neither on sender side nor on receiver	The sender must be able to search and select only the requested frame.
ACK Numbers	NAK number refer to the next expected frame number.	NAK number refer to the frame lost.
Use	It more often used.	It is less in practice because of its complexity.

9) Short note on UDP?

Answer: -

1) In TCP communications, not only TCP but also UDP (User Datagram Protocol) can meet the functional requirements defined in the transport layer. Unlike TCP, UDP is a connectionless type protocol in which the sending terminal does not check whether data has been received by receiving terminal. In other words, it is a less reliable protocol. However, this protocol makes it possible to omit a variety of processes thus reducing the load on the CPU.
2) The UDP protocol allows applications to exchange datagrams. It uses that the concept of port, which allows to distinguish the different applications running on a machine. Besides the datagram and its data, a UDP message contains a source port number and destination port number.

   UDP provides a connectionless service and without error recovery. It uses no acknowledgment, not resequencing the messages and sets up no flow control. It is possible that the UDP messages that are lost are duplicated, or delivered out of sequence they arrive too early to be processed upon receipt. As previously explained, UDP is a very simple message protocol level architecture reference model. It has the advantage of a quick execution, taking into account real time constraints or place limitations on a processor. These constraints or limitations may not allow the use of heavier protocols, such as TCP. Applications that do not require a high level security transmission, and there are many, and the management software, which require quizzes resources prefer to use UDP. Search requests in directories pass through UDP, for example. To identify the different applications, UDP needed to place each fragment in a reference that the port plays role. Figure shows the UDP fragment. A reference identifies, much like the next header field in IPv6, which is carried in the body of the fragment. The most important applications that use UDP correspond to the following port numbers**:•7:echoservice;•9: Rejection of service;•53: DNS domain name server (Dynamic Name Server);• 67: DHCP configuration server;• 68: DHCP configuration client.**
3) The simplicity of the UDP header stems from the unsophisticated nature of the services it provides.



Format of the UDP datagram

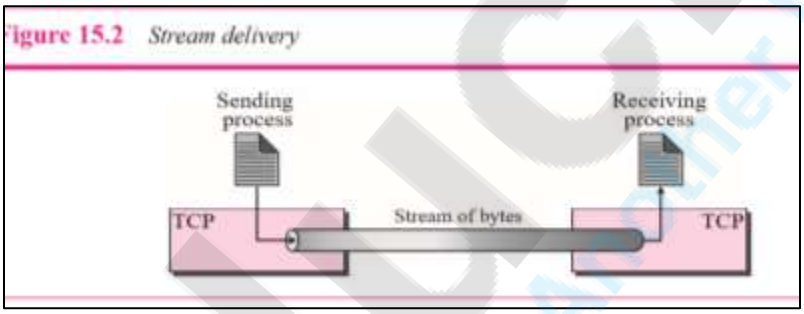
4) Following is a brief description of each field:

A) **Source Port** this is the port number of the application that is originating the user data.
B) **Destination Port** this is the port number pertaining to the destination application.
C) **Length** This field describes the total length of the UDP datagram, including both data and header Information.
D) **UDP Checksum** Integrity checking is optional under UDP. If turned on, both ends of the communications channel use this field for data integrity checks. At this point, it is important to understand the layering concept along with the need for headers.

10) Explain services provided by TCP?

Answer: -

1) **Stream Delivery Service: -**
   A) TCP, on the other hand, allows the sending process to deliver data as a stream of bytes and allows the receiving process to obtain data as a stream of bytes. TCP creates an environment in which the two processes seem to be connected by an imaginary "tube" that carries their bytes across the Internet. This imaginary environment is depicted in Figure 15.2. The sending process produces (writes to) the stream of bytes and the receiving process consumes (reads from) them.



2) **Full-Duplex Communication: -**
   A) TCP offers full-duplex service, where data can flow in both directions at the same time. Each TCP endpoint then has its own sending and receiving buffer, and segments move in both directions.
3) **Multiplexing and De-multiplexing: -**
   A) TCP performs multiplexing at the sender and de-multiplexing at the receiver. However, since TCP is a connection-oriented protocol, a connection needs to be established for each pair of processes.
4) **Connection-Oriented Service: -**
   A) TCP, unlike UDP, is a connection-oriented protocol. As shown in Chapter 13, when a process at site A wants to send to and receive data from another process at site B, the following three phases occur:
      a) The two TCPs establish a virtual connection between them.
      b) Data are exchanged in both directions.
      c) The connection is terminated.

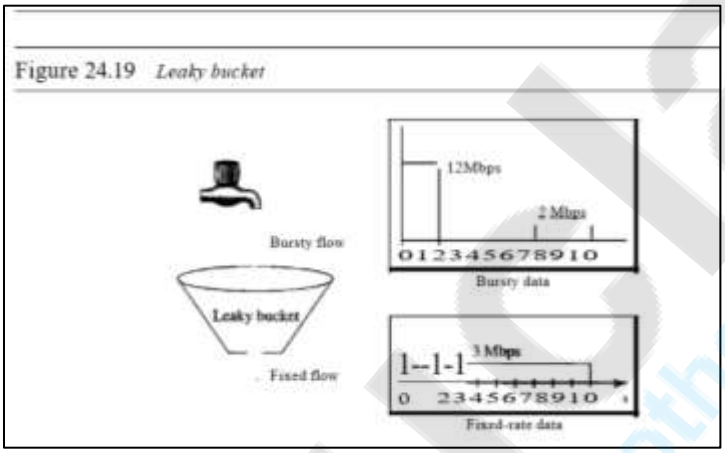
11) What is traffic shaping? What are the techniques used for traffic shaping? Explain any one technique in brief?

Answer: -

1) Traffic shaping is a mechanism to control the amount and the rate of the traffic sent to the network. Two techniques can shape traffic: leaky bucket and token bucket.

2) Leaky Bucket: -

A) If 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 leaks does not depend on 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 networking, a technique called leaky bucket can smooth out burst traffic. Busty chunks are stored in the bucket and sent out at an average rate.

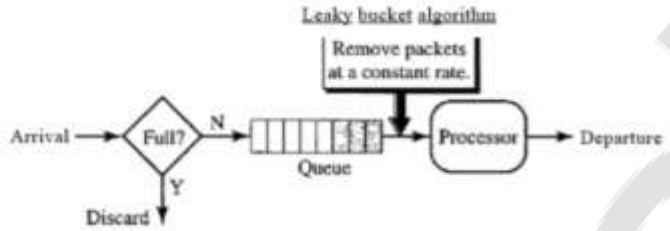


B) The network has committed a bandwidth of 3 Mbps for a host. The use of the leaky bucket shapes the input traffic to make it conform to this commitment. In Figure 24.19 the host sends a burst of data at a rate of 12 Mbps for 2 s, for a total of 24 Mb bits of data. The host is silent for 5 s and then sends data at a rate of 2 Mbps for 3 s, for a total of 6 Mb bits of data. In all, the host has sent 30 Mb its of data in 10s. The leaky bucket smooths the traffic by sending out data at a rate of3 Mbps during the same 10 s. Without the leaky bucket, the beginning burst may have hurt the network by consuming more bandwidth than is set aside for this host. We can also see that the leaky bucket may prevent congestion. As an analogy, consider the freeway during rush hour (bursty traffic). If, instead, commuters could stagger their working hours, congestion on our freeways could be avoided.

C) A simple leaky bucket implementation is shown in Figure 24.20. A FIFO queue holds the packets. If the traffic consists of fixed-size packets (e.g., cells in ATM networks), 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 number of bytes or bits. The following is an algorithm for variable-length packets:

Figure 24.20 *Leaky bucket implementation*



a) Initialize a counter to n at the tick of the clock.
b) If n is greater than the size of the packet, send the packet and decrement the counter by the packet size. Repeat this step until n is smaller than the packet size.
c) Reset the counter and go to step 1.

12) Difference between

A) **Flow Control and Congestion Control**

BASIS FOR COMPARISON	FLOW CONTROL	CONGESTION CONTROL
Basic	It controls the traffic from a particular sender to a receiver.	It controls the traffic entering the network.
Purpose	It prevents the receiver from being overwhelmed by the data.	It prevents the network from getting congested.
Responsibility	Flow control is the responsibility handled by data link layer and the transport layer.	Congestion Control is the responsibility handled by network layer and transport layer.
Responsible	The sender is responsible for transmitting extra traffic at receivers side.	The transport layer is responsible transmitting extra traffic into the network.
Preventive measures	The sender transmits the data slowly to the receiver.	Transport layer transmits the data into the network slowly.
Methods	Feedback-based flow control and Rate-based flow control	Provisioning, traffic-aware routing and admission control

B) **Flow Control and Error Control**

BASIS FOR COMPARISON	FLOW CONTROL	ERROR CONTROL
Basic	Flow control is meant for the proper transmission of the data from sender to the receiver.	Error control is meant for delivering the error-free data to the receiver.
Approach	Feedback-based flow control and rate-based flow control are the approaches to achieve the proper flow control.	Parity checking, Cyclic Redundancy Code (CRC) and checksum are the approaches to detect the error in data. Hamming code, Binary Convolution codes, Reed-Solomon code, Low-Density Parity Check codes are the approaches to correct the error in data.
Impact	avoid overrunning of receivers buffer and prevents the data loss.	Detects and correct the error occurred in the data.

13) Explain how the reliable data transfer is achieved using selective repeat protocol?
OR
Explain the principles of reliable data transfer in detail? **

14) Explain Go-Back-N Protocol?

Answer: -

1) In a Go-Back-N (GBN) protocol, the sender is allowed to transmit multiple packets (when available) without waiting for an acknowledgment, but is constrained to have no more than some maximum allowable number, N, of unacknowledged packets in the pipeline.
2) The sender's view of the range of sequence numbers in a GBN protocol. If we define base to be the sequence number of the oldest unacknowledged packet and next seq num to be the smallest unused sequence number (that is, the sequence number of the next packet to be sent), then four intervals in the range of sequence numbers can be identified. Sequence numbers in the interval [0,base-1] correspond to packets that have already been transmitted and acknowledged. The interval [base, next seq num-1] corresponds to packets that have been sent but not

yet acknowledged. Sequence numbers in the interval [nextseqnum,base+N-1] can be used for packets that can be sent immediately, should data arrive from the upper layer. Finally, sequence numbers greater than or equal to base+N cannot be used until an unacknowledged packet currently in the pipeline (specifically, the packet with sequence number base) has been acknowledged.

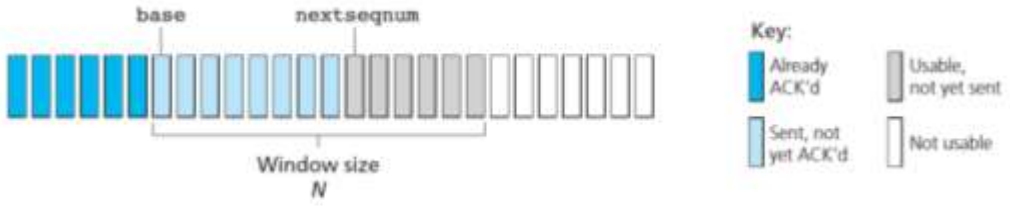


Figure 3.19 ♦ Sender's view of sequence numbers in Go-Back-N

3) The GBN sender must respond to three types of events:
4) **Invocation from above.** When rdt_send() is called from above, the sender first checks to see if the window is full, that is, whether there are N outstanding, unacknowledged packets. If the window is not full, a packet is created and sent, and variables are appropriately updated. If the window is full, the sender simply returns the data back to the upper layer, an implicit indication that the window is full. The upper layer would presumably then have to try again later. In a real implementation, the sender would more likely have either buffered (but not immediately sent) this data, or would have a synchronization mechanism (for example, a semaphore or a flag) that would allow the upper layer to call rdt_send() only when the window is not full.
5) **Receipt of an ACK.** In our GBN protocol, an acknowledgment for a packet with sequence number n will be taken to be a cumulative acknowledgment, indicating that all packets with a sequence number up to and including n have been correctly received at the receiver. We'll come back to this issue shortly when we examine the receiver side of GBN.
6) **A timeout event.** The protocol's name, "Go-Back-N," is derived from the sender's behavior in the presence of lost or overly delayed packets. As in the stop-and-wait protocol, a timer will again be used to recover from lost data or acknowledgment packets. If a timeout occurs, the sender resends all packets that have been previously sent but that have not yet been acknowledged. Our sender in Figure 3.20 uses only a single timer, which can be thought of as a timer for the oldest transmitted but not yet acknowledged packet. If an ACK is received but there are still additional transmitted but not yet acknowledged packets, the timer is restarted. If there are no outstanding, unacknowledged packets, the timer is stopped.
7) The receiver's actions in GBN are also simple. If a packet with sequence number n is received correctly and is in order (that is, the data last delivered to the upper layer came from a packet with sequence number n – 1), the receiver sends an

ACK for packet n and delivers the data portion of the packet to the upper layer. In all other cases, the receiver discards the packet and resends an ACK for the most recently received in-order packet. Note that since packets are delivered one at a time to the upper layer, if packet k has been received and delivered, then all packets with a sequence number lower than k have also been delivered. Thus, the use of cumulative acknowledgments is a natural choice for GBN. In our GBN protocol, the receiver discards out-of-order packets. Although it may seem silly and wasteful to discard a correctly received (but out-of-order) packet, there is some justification for doing so. Recall that the receiver must deliver data in order to the upper layer. Suppose now that packet n is expected, but packet n + 1 arrives. Because data must be delivered in order, the receiver could buffer (save) packet n + 1 and then deliver this packet to the upper layer after it had later received and delivered packet n. However, if packet n is lost, both it and packet n + 1 will eventually be retransmitted as a result of the GBN retransmission rule at the sender. Thus, the receiver can simply discard packet n + 1. The advantage of this approach is the simplicity of receiver buffering—the receiver need not buffer any out-of-order packets. Thus, while the sender must maintain the upper and lower bounds of its window and the position of next seq num within this window, the only piece of information the receiver need maintain is the sequence number of the next in-order packet. This value is held in the variable expected seq num, shown in the receiver FSM in Figure 3.21. Of course, the disadvantage of throwing away a correctly received packet is that the subsequent retransmission of that packet might be lost or garbled and thus even more retransmissions would be required.