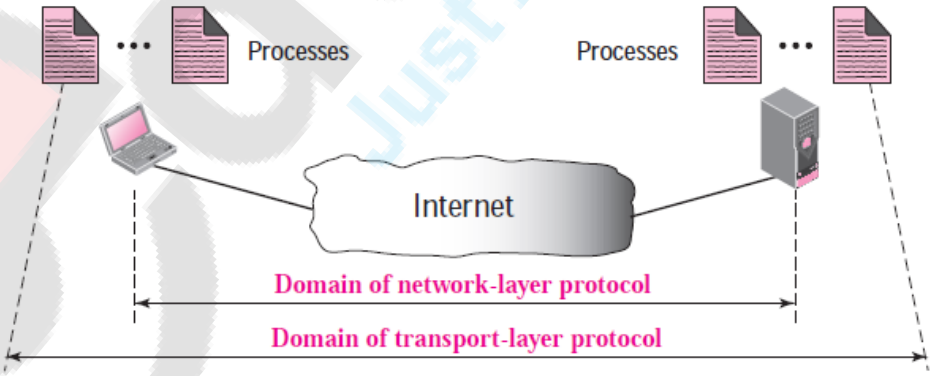
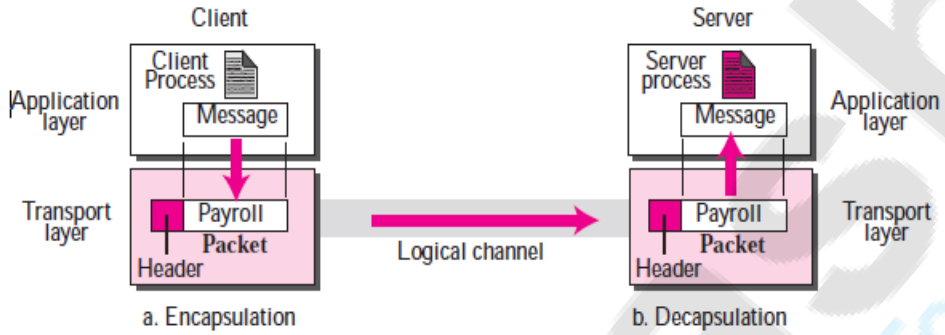
Q.1	Explain Transport layer services Or, Explain different services of Transport Layer.
Ans.	**TRANSPORT-LAYER SERVICES:** • The transport layer is located between the network layer andthe application layer. • The transport layer is responsible for providing services to theapplication layer; it receives services from the network layer. Services of Transport layer: 1. Process to process communication 2. Encapsulation and Decapsulation 3. Multiplexing Demultiplexing 4. Flow control 5. Error Control 6. Congestion Control **Process-to-Process Communication:** • The first duty of a transport-layer protocol is to provide **process-to-process communication.** • A process is an application-layer entity (running program) that uses the servicesof the transport layer. • The network layer is responsible for communication at the computer level (host-to-hostcommunication). • A network layer protocol can deliver the message only to thedestination computer. • However, this is an incomplete delivery. The message still needsto be handed to the correct process. This is where a transport layer protocol takes over. • A transport layer protocol is responsible for delivery of the message to the appropriateprocess.  **Encapsulation and Decapsulation:** • To send a message from one process to another, the transport layer protocol encapsulatesand decapsulates messages. • Encapsulation happens at the sender site. When a process has a message to send, itpasses the message to the transport layer along with a pair of socket addresses and some other pieces of information that depends on the transport layer protocol.

- Decapsulation happens at the receiver site. When the message arrives at the destinationtransport layer, the header is dropped and the transport layer delivers the messageto the process running at the application layer.
- The sender socket address is passedto the process in case it needs to respond to the message received.

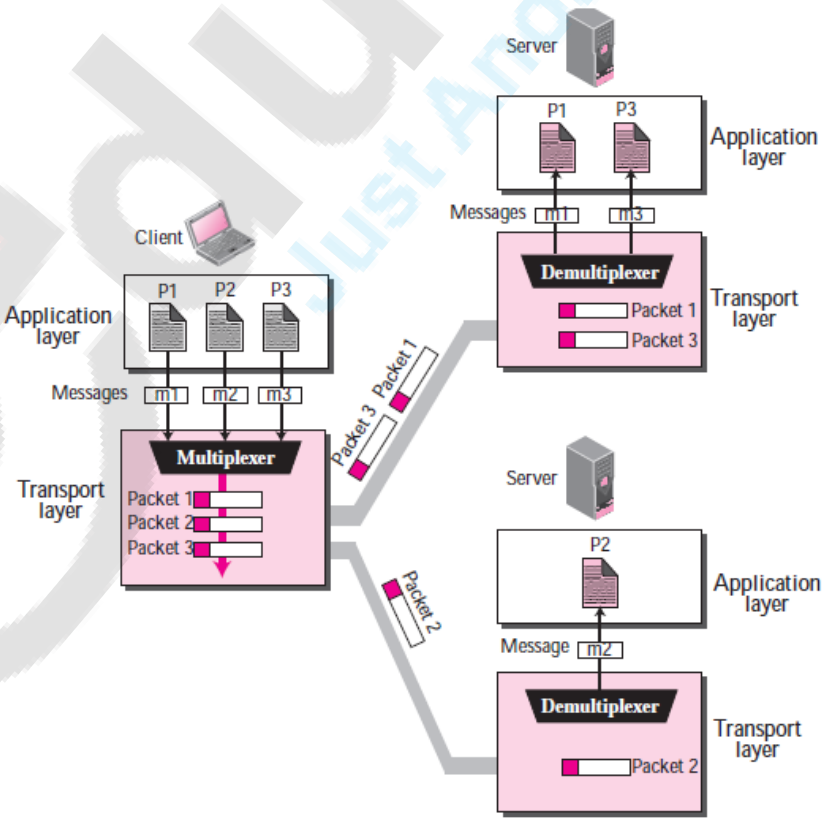
Figure 13.6 *Encapsulation and decapsulation*



Multiplexing and Demultiplexing:

- Whenever an entity accepts items from more than one source, it is referred to as **multiplexing**(many to one); whenever an entity delivers items to more than one source, it is referred to as **demultiplexing**(one to many).
- The transport layer at the source performs multiplexing; the transport layer at the destination performs demultiplexing.

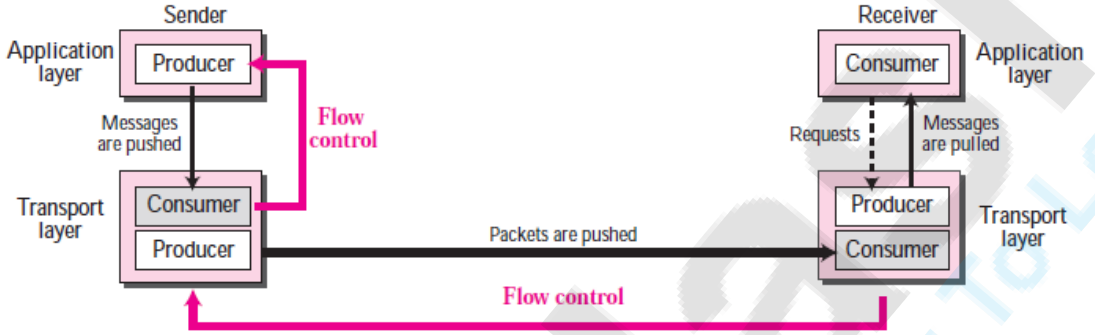
Figure 13.7 *Multiplexing and demultiplexing*



Flow Control:

- Whenever an entity produces items and another entity consumes them, there should bea balance between production and consumption rates.
- If the items are produced fasterthan they can be consumed, the consumer can be overwhelmed and needs to discardsome items.

Figure 13.9 *Flow control at the transport layer*

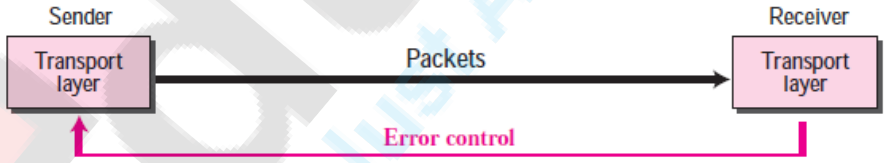


Error Control:

Error control at the transport layer is responsible to

1. Detect and discard corrupted packets.
2. Keep track of lost and discarded packets and resend them.
3. Recognize duplicate packets and discard them.
4. Buffer out-of-order packets until the missing packets arrive.

Figure 13.10 *Error control at the transport layer*



Congestion Control:

- An important issue in the Internet is **congestion.** Congestion in a network may occur ifthe **load** on the network—the number of packets sent to the network—is greater thanthe *capacity* of the network—the number of packets a network can handle.
- **Congestioncontrol** refers to the mechanisms and techniques to control the congestion and keep theload below the capacity.

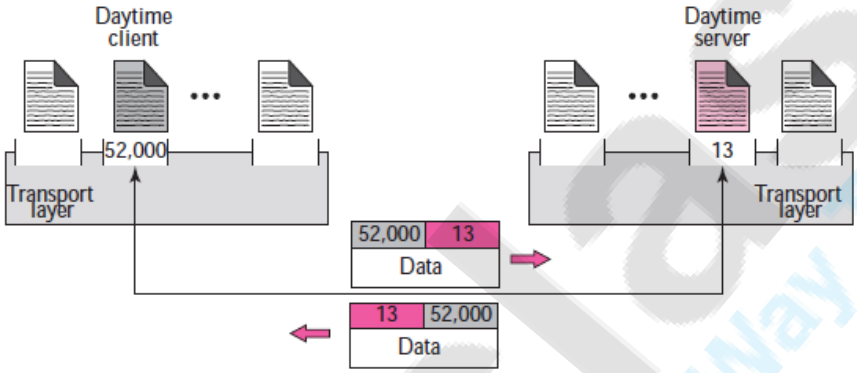
Q2.	Explain Port addressing.
Ans.	**Port Address:** • Port address is a feature of a network device that translates TCP or UDP communications made between a host and port on an outside network It allows a single IP address to be used for many internal hosts. • Port address can automatically modify the IP packets' destination or source host IP and port fields belonging to its internal hosts. **Figure 13.2** *Port numbers*  **Need more content…….**
Q.3	Explain Multiplexing anddemultiplexing.
Ans.	Multiplexing and De-multiplexing are the two very important functions that are performed by Transport Layer. **Multiplexing:** Transport layer at the sender side receives data from different Applications , encapsulates every packet with a Transport Layer header and pass it on to the underlying Network Layer. This job of transport layer is known as Multiplexing. **Demultiplexing:** At the receiver's side, the transport gathers the data, examines it socket and passes the data to the correct Application. This is known as De-Multiplexing.

Figure 13.7 *Multiplexing and demultiplexing*

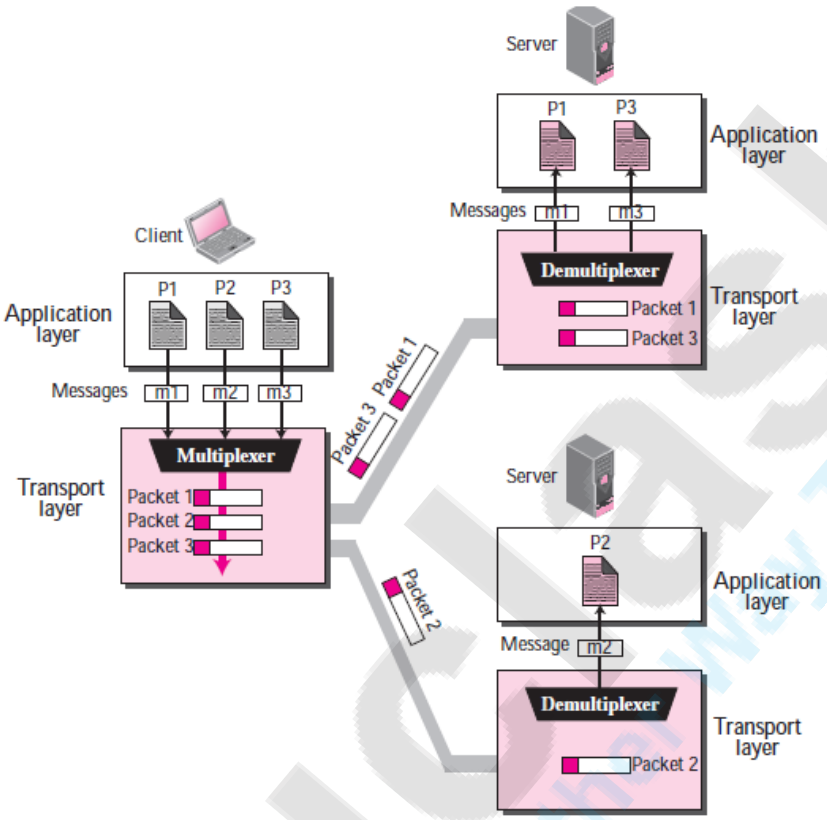


Diagram Explanation

Figure 13.7 shows communication between a client and two servers. Three client processes are running at the client site, P1, P2, and P3. The processes P1 and P3 need to send requests to the corresponding server process running in a server. The client process P2 needs to send a request to the corresponding server process running at another server. The transport layer at the client site accepts three messages from the three processes
and creates three packets. It acts as a *multiplexer*. The packets 1 and 3 use the same logical channel to reach the transport layer of the first server. When they arrive at the server, the transport layer does the job of a *multiplexer* and distributes the messages to two different processes. The transport layer at the second server receives packet 2 and delivers it to the corresponding process.

Q.5 Explain Congestion Control.

Ans.

Congestion Control:

- An important issue in the Internet is **congestion.** Congestion in a network may occur ifthe **load** on the network—the number of packets sent to the network—is greater thanthe *capacity* of the network—the number of packets a network can handle.
- **Congestioncontrol** refers to the mechanisms and techniques to control the congestion and keep theload below the capacity.

Traffic Shaping:

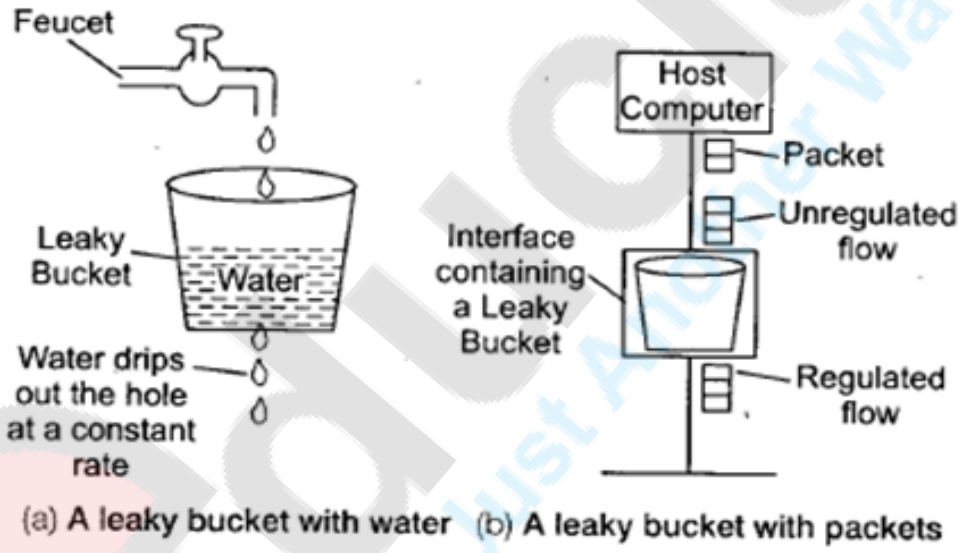
- Another method to congestion control is to shape the traffic before it enters the network.
- It controls the rate at which packets are sent (not just how many). Used in ATM and integrated services networks.
- At connections setup time, the sender and carrier negotiate a traffic pattern (shape).

Two traffic shaping algorithms are as follows:

1. Leaky Bucket
2. Token Bucket

The Leaky Bucket (LB) Algorithm

The Leaky Bucket algorithm used to control rate in the network. It is implemented as single-server queue with constant service time. If the buffer (bucket) overflows, then packets are discarded.

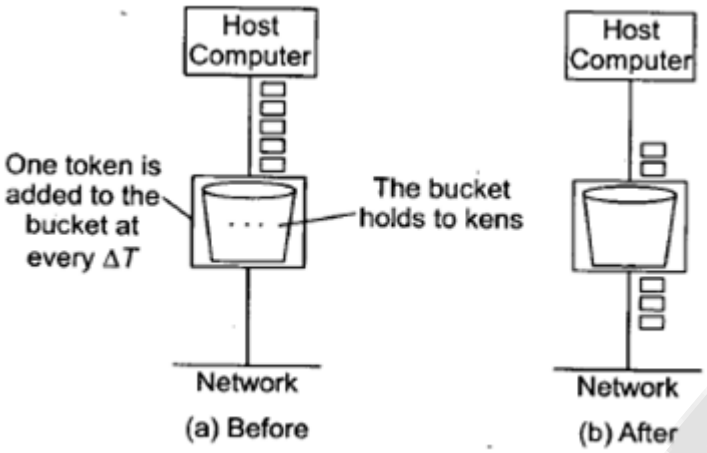


- The leaky bucket enforces a constant output rate (average rate) regardless of the burstiness of the input. Does nothing when input is idle.

- When packets are of the same size (as in ATM cells), the host should inject one packet per clock tick onto the network. But for variable length packets, it is better to allow a fixed number of bytes per tick.

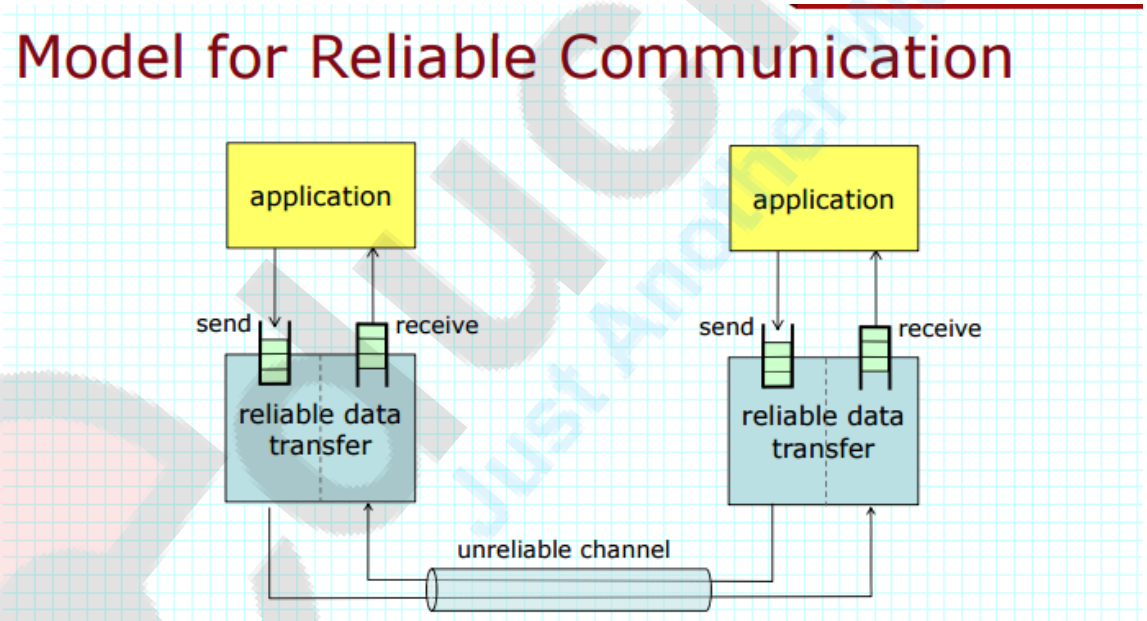
Token Bucket (TB) Algorithm

In contrast to LB, the token bucket algorithm, allows the output rate to vary depending on the size of the burst.



Q. 6 Principle of Reliable Data Transfer

Ans



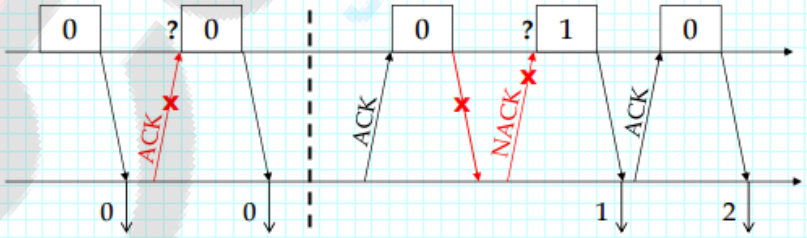
Basic Concepts

- focus separately on *sender* and *receiver* behavior
- receiver *acknowledges* packets from sender
  - » positive acks and/or negative acks (nack)
- sender *transmits* and *retransmits* based on information provided by the received
  - » time-out to trigger action if ack/nack is not received within a certain time-frame
  - » both packets and acks can get lost
- sequence numbers in each packet
  - » sequence number in ack identifies received packet(s)
  - » size of seq. num. field determines # packets that can be sent before acknowledgments must be received
    - simplest protocol uses 1 bit sequence number
      - –sends one packet and waits for ack before sending another

4

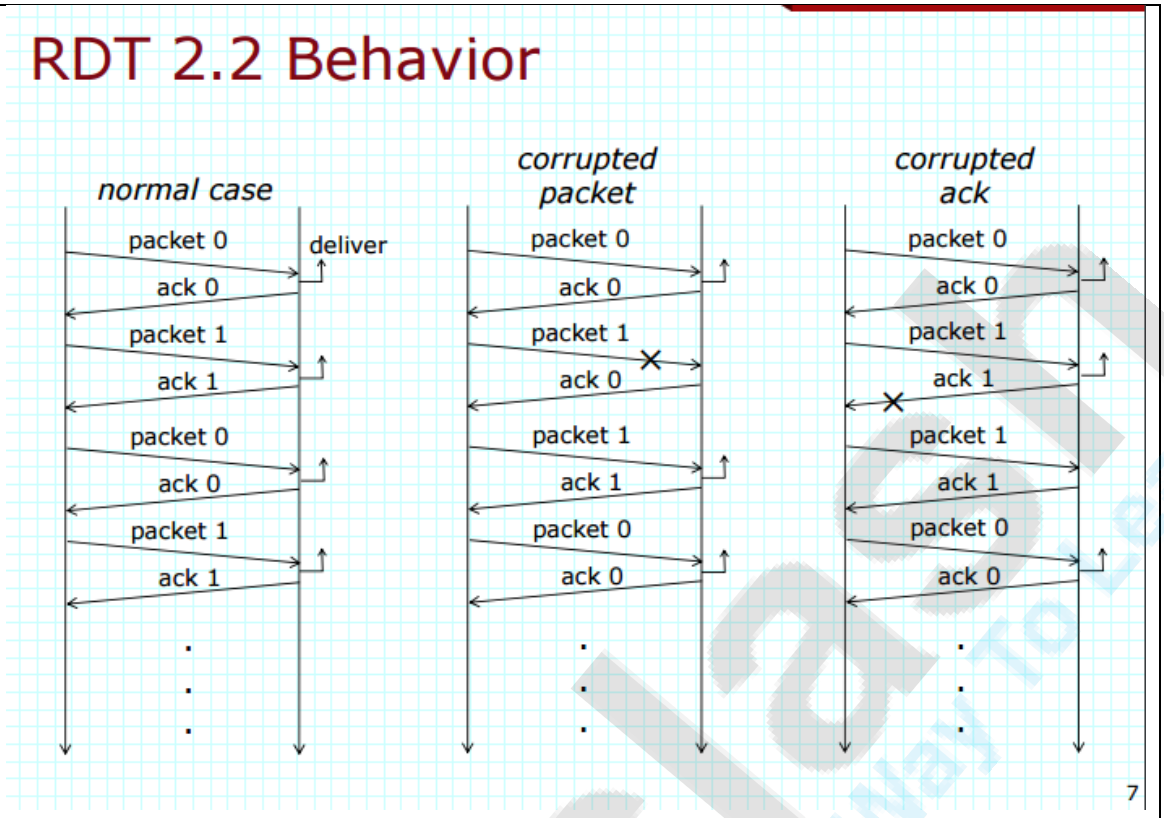
Basic Protocol – RDT 2.x

- many things can go wrong?
  - » Corruption of acks, nacks, or packets means that sender may not be able to tell which packet was received correctly or not
    - if sender retransmits the packet, the receiver could deliver the same packet to the application twice. If it transmits a new one, the receiver can fail to deliver a packet to the application.



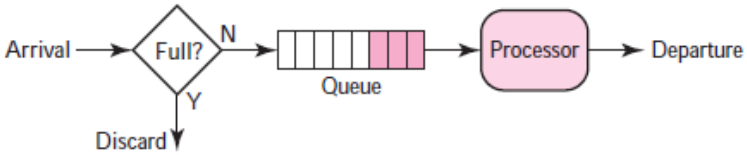
- solution is to add a 1 bit sequence number
  - » in this case, it is sufficient to use positive acks only

6



Q.7	Define FIFO queuing, priority queuing, and weighted fair queuing.(Scheduling) Or, Define several techniques to improve the quality of service
Ans.	FIFO queuing: 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 25.29 shows a conceptual view of a FIFO queue.

Figure 25.29 *FIFO queue*

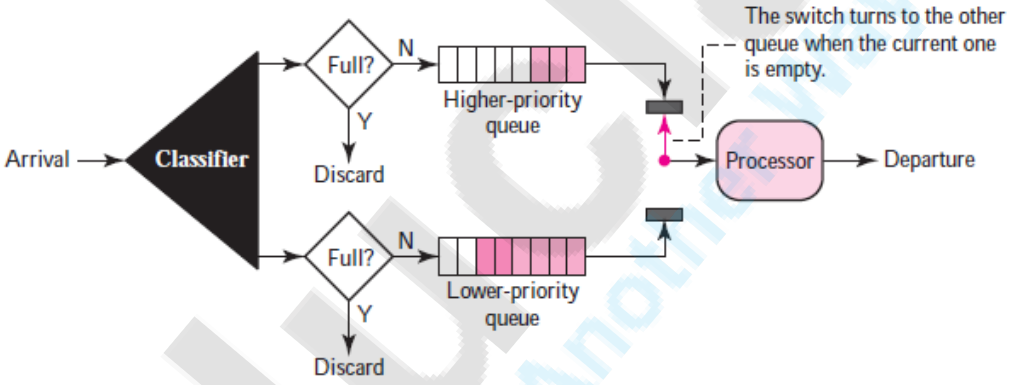


Priority Queuing:

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. Figure 25.30 shows priority queuing with two priority levels (for simplicity).

A priority queue can provide better QoS than the FIFO queue because higherpriority 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*.

Figure 25.30 *Priority queuing*

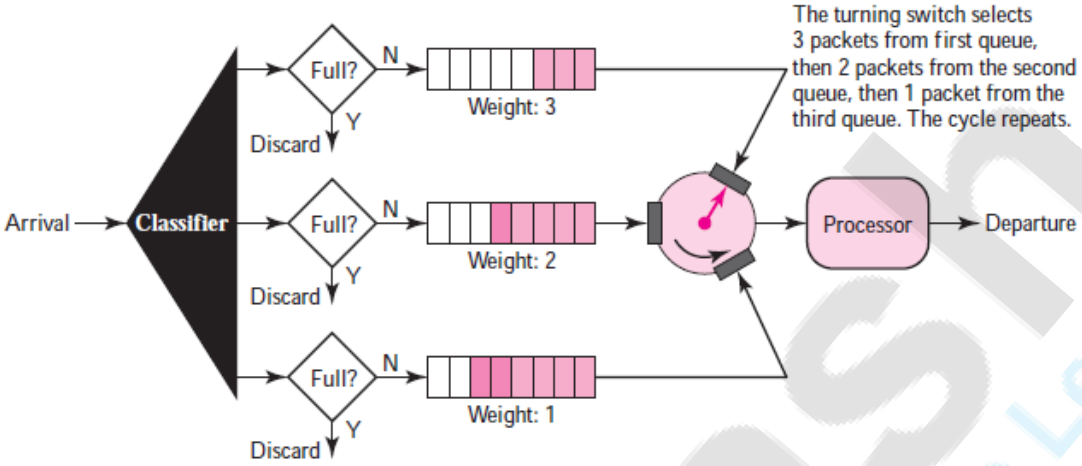


Weighted Fair Queuing:

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. 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. Figure 25.31 shows the technique with three classes.

Figure 25.31 *Weighted fair queuing*



Fair Queuing:

n important alternative to FIFO and priority is fair queuing. Where FIFO and its variants have a single input class and put all the incoming traffic into a single physical queue, fair queuing maintains a separate logical FIFO subqueue for each input class; we will refer to these as the per-class subqueues. Division into classes can be fine-grained – eg a separate class for each TCP connection – or coarse-grained – eg a separate class for each arrival interface, or a separate class for each designated internal subnet.

Q. 8 Explain traffic shaping: a. Leaky bucket b. Token bucket.
(this comes under technique to improve quality of services)

Ans.

Traffic Shaping

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.

Leaky Bucket:

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 bursty traffic. Bursty chunks are stored in the bucket and sent out at an average rate. Figure 25.32 shows a leaky bucket and its effects.

Figure 25.32 *Leaky bucket*

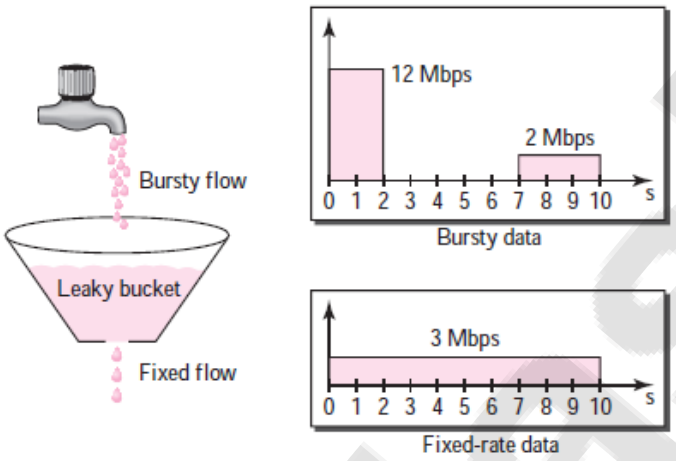


Diagram Explanation:

In the figure, we assume that 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 25.32 the host sends a burst of data at a rate of 12 Mbps for 2 s, for a total of 24 Mbits 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 Mbits of data. In all, the host has sent 30 Mbits of data in 10 s. The leaky bucket smooths the traffic by sending out data at a rate of 3 Mbps during the same 10 s.

Figure 25.33 *Leaky bucket implementation*

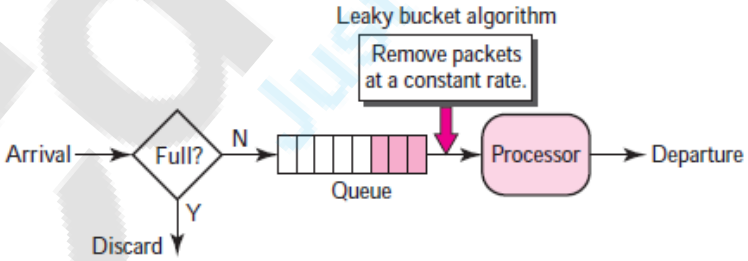


Diagram Explanation:

A simple leaky bucket implementation is shown in Figure 25.33. 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.

Token Bucket:

The leaky bucket is very restrictive. It does not credit an idle host. For example, if a host is not sending for a while, its bucket becomes empty. Now if the host hasbursty data, the leaky bucket allows only an average rate. The time when the host was idle is not taken into account. On the other hand, the **token bucket** algorithm allows idle hosts to accumulate credit for the future in the form of tokens

Figure 25.34 *Token bucket*

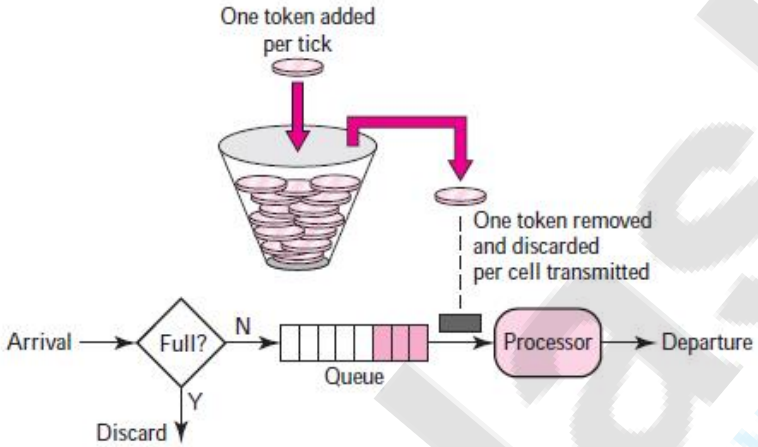


Diagram Explanation:

The system removes one token for every cell (or byte) of data sent. For example, if *n* is 100 and the host is idle for 100 ticks, the bucket collects 10,000 tokens. Now the host can consume all these tokens in one tick with 10,000 cells, or the host takes 1,000 ticks with 10 cells per tick. In other words, the host can send bursty data as long as the bucket is not empty. Figure 25.34 shows the idea

Q. 9 Explain Resource Reservation(RSVP)

OR,

Explain Integrated services: Resources and Reservation (RSVP)

Ans.

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.

Integrated Services:

one QoS model called Integrated Services, which depends heavily on resource reservation to improve the quality of service..

Integrated Services, sometimes called **IntServ,** is a *flow-based* QoS model, which means that a user needs to create a flow, a kind of virtual circuit, from the source to the destination and inform all routers of the resource requirement.

The Resource Reservation Protocol (RSVP) is a signaling protocol to help IP create a flow and consequently make a resource reservation.

The Resource Reservation Protocol (RSVP) message:

RSVP has several types of messages. However, for our purposes, we discuss only two of them: **Path** and **Resv.**

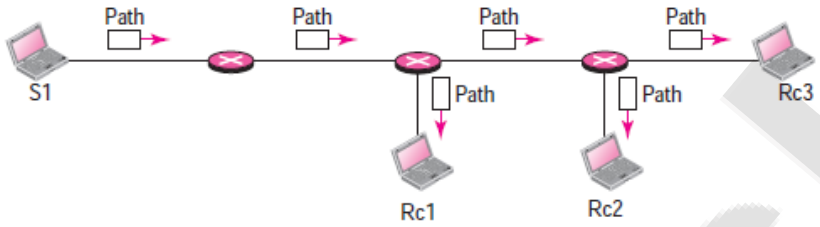
Path Messages

Recall that the receivers in a flow make the reservation in RSVP.

However, the receivers do not know the path traveled by packets before the reservation is made. The path is needed for the reservation. To solve the problem, RSVP uses *Path* messages. A Path message travels from the sender and reaches all receivers in the multicast path. On the way, a Path message stores the necessary information for the receivers.

A Path message is sent in a multicast environment; a new message is created when the path diverges. Figure 25.35 shows path messages.

Figure 25.35 *Path messages*



Resv Messages:

After a receiver has received a Path message, it sends a *Resv*message.
The Resv message travels toward the sender (upstream) and makes a resource reservation on the routers that support RSVP. If a router does not support RSVP on the path, it routes the packet based on the best-effort delivery methods we discussed before. Figure 25.36 shows the Resv messages.

Figure 25.36 *Resv messages*