UNIT-5

Transport Layer

The transport layer is a 4th layer from the top. It works for the transmission of data from one host to the other located in different networks. It also takes care of selection of shortest path to transmit the packet from the number of routes available. Segment in network layer is referred as packet. The transport layer protocols are implemented in the end systems but not in the network routers. A computer network provides more than one protocol to the network applications. For example, TCP and UDP are two transport layer protocols that provide a different set of services to the network layer.

Functions of Transport Layer

i) Segmentation and Reassembly → This layer accepts the message from the (session) layer, breaks into smaller units. The transport layer at the destination station resembles the message.

ii) Service Point Addressing → This layer includes service point address which makes sure that the message is delivered to the correct process.

iii) Flow Control → In this layer, flow control is performed end to end.

iv) Error Control → Error control is performed end to end in this layer to ensure that the complete message arrives at the receiving transport layer without any error. Error correction is done through retransmission.

v) Connection Control → It includes Connectionless Transport layer and Connection Oriented Transport layer. In connectionless transport layer each segment is considered as an independent packet and delivered to the transport layer at the destination machine. In connection oriented transport layer before delivering packets, connection is made with transport layer at the destination machine.

Services/Responsibilites of Transport Layer:

i) **Process to Process Delivery** → Transport layer requires a port number to correctly deliver the segments of data to the correct process amongst the multiple processes running on a particular host. A port number is a 16 bit address used to identify any client-server program uniquely.

ii) **End-to-end Connection between Hosts** → Transport layer is also responsible for creating the end-to-end connection between hosts for which it mainly uses TCP and UDP.

iii) **Multiplexing and De-multiplexing** → Multiplexing allows simultaneous use of different applications over networks which are running on a host. De-multiplexing is required at the receiver side to obtain the data comming from various processes.

iv) **Congestion Control** → Congestion is a situation in which too many sources over a network attempt to send data and the router buffers start overflowing due to which lots of packets occur. In this situation Transport Layer provides Congestion Control in different ways like open loop, closed loop etc to prevent congestion.

v) **Data Integrity and Error Correction** → Transport layer checks for errors in the messages comming from application layer by using error delection codes and uses the ACK and NACK services to inform the sender if the data is arrived or not and checks for the integrity of data.

⑧. Transport Protocols:

The transport protocols provide services to their upper layers at well-defined interface points, which are also referred as ports. The IP address and the port are an important combination to set up a transport connection. TCP and UDP are the main transport layer protocols that provide different set of services to the network layer.

Transmission control protocol (TCP)	User datagram protocol (UDP)
i) TCP is a connection-oriented protocol.	i) UDP is the Datagram oriented protocol.
ii) TCP is reliable as it guarantees the delivery of data to the destination router.	ii) The delivery of data to the destination cannot be guaranteed in UDP.
iii) TCP provides extensive error checking mechanisms. It is because it provides flow control and acknowledgement of data.	iii) UDP has only the basic error checking mechanism using checksums.
iv) TCP doesn't support Broadcasting.	iv) UDP supports Broadcasting.
v) TCP is comparatively slower than UDP.	v) UDP is faster, simpler and more efficient than TCP.
vi) TCP is used by HTTP, HTTPs, FTP, SMTP and Telnet.	vi) UDP is used by DNS, DHCP, TFTP, SNMP, RIP and VoIP.

⊛. Connection Oriented and Connectionless Services:

Comparision Parameter	Connection-oriented service	Connectionless service
Definition	The service used to create an end to end connection betwn the senders to the receiver before transmitting the data over the network is called Connection-oriented service.	The service used to transfer the data packets betwn senders to the receiver without creating any connection is called connectionless service.
Virtual path	It creates a virtual path between the sender and receiver.	It does not create any virtual path between the sender and receiver.
Authentication	It requires authentication before transmitting the data packets to the receiver.	It does not require authentication before transferring data packets.

Data Packets Path.	All data packets are received in the same order as those sent by the sender.	Not all data packets are received in the same order as those sent by the sender.
Bandwidth Requirement	It requires a higher bandwidth to transfer the data packets.	It requires low bandwidth to transfer the data packets.
Examples	TCP is an example of a connection-oriented service.	UDP is an example of connectionless service.

⊛ Congestion Control:

An important issue in a packet-switched network is congestion. Congestion in a network may occur if the load on the network is greater than the capacity of network. Congestion control refers to the mechanisms and techniques to control the congestion and keep the load below the capacity. Congestion happens in any system that involves waiting. In general, we can divide congestion control mechanisms into two broad categories: open-loop congestion control and closed-loop congestion control.

1) **Open-Loop Congestion Control:** In open-loop congestion control, policies are applied to prevent congestion before it happens. Following are the policies used to prevent congestion in open-loop.

Retransmission Policy → If the sender feels that a sent packet is lost or corrupted, the packet needs to be retransmitted. Retransmission in general may increase congestion in network. However, a good retransmission policy can prevent congestion.

Window Policy → The type of window at the sender may also affect congestion. The Selective Repeat window is better than the Go-Back-N window for congestion control. The Selective Repeat window tries to send the specific packets that have been lost or corrupted.

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. Sending fewer acknowledgments means imposing less load on the network.

Discarding Policy → A good discarding policy by the routers may prevent congestion and at the same time may not harm the integrity of the transmission.

Admission Policy → It also can prevent congestion in virtual-circuit networks. 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.

2. Closed-Loop Congestion Control: In closed-loop congestion control, polices are applied to prevent congestion after it happens. Following are the policies used to prevent congestion in closed-loop.

Backpressure → It is a mechanism in which a congested node stops receiving data from the immediate upstream node or nodes. This technique can be only applied to virtual circuit networks.

Choke Packet → It is a packet sent by a node to the source to inform it congestion. In backpressure the warning is from one node to its upstream node but in choke packet method, the warning is from the router, which has encountered congestion.

Implicit Signaling → In implicit signaling, there is no communication between the congested node(s) and the source. The source guesses that there is a congestion somewhere in the network from other symptoms.

Explicit Signaling → In explicit signaling, the signal is included in the packets that carry data.

Backward Signaling → A bit can be set in a packet moving in the direction opposite to the congestion. This bit can warn the source that there is congestion.

Forward Signaling → A bit can be set in a packet moving in the direction to the congestion. This bit can warn the destination that there is congestion.

TCP Congestion Control:

TCP's general policy for handling congestion consists of following three phases:

i) **Slow Start Phase** → It helps to avoid sending more data than the network is capable of forwarding. Initially sender sets Congestion window size = Maximum Segment Size (1MSS). After receiving each acknowledgment, sender increases the congestion window size by 1MSS. In this phase, the size of congestion window increases exponentially. The Formula is;

Congestion Window Size = Congestion Window Size + Maximum Segment Size

In this phase after every RTT the congestion window size increments exponentially.

Initially cwnd = 1 (congestion window)

After,
1RTT, cwnd = $2^1 = 2$
2RTT, cwnd = $2^2 = 4$
3RTT, cwnd = $2^3 = 8$ (Round-Trip Time)

ii) **Congestion Avoidance Phase** → It is also called additive increment. This phase starts after the threshold value also denoted as ssthresh. The size of congestion window increases additive. After each RTT cwnd = cwnd+1.

Initially cwnd = i
After,
1RTT, cwnd = i+1
2RTT, cwnd = i+2
3RTT, cwnd = i+3.

iii) **Congestion detection** → It is also called multiplicative decrement. Retransmission is needed to recover a missing packet which is assumed to have been dropped by a router due to congestion. Retransmission can occur in one of two cases: when the RTO timer times out or when three duplicate ACKs are received.

⊛. Traffic Shaping Algorithms:

Traffic Shaping is a mechanism to control the amount and the rate of the traffic sent to the network. Approach of congestion management is called Traffic Shaping. Traffic Shaping helps to regulate rate of data transmission and reduces congestion. There are two types of traffic shaping algorithms: Leaky Bucket and Token Buket.

Leaky bucket → The leaky Bucket algorithm is used to control rate in a network. It is implemented as a single server queue with constant service time. If the bucket overflows then packets are discarded. In this algorithm the input rate can vary but the output rate remains constant. This algorithm saves busty traffic into fixed rate traffic by averaging the data rate.

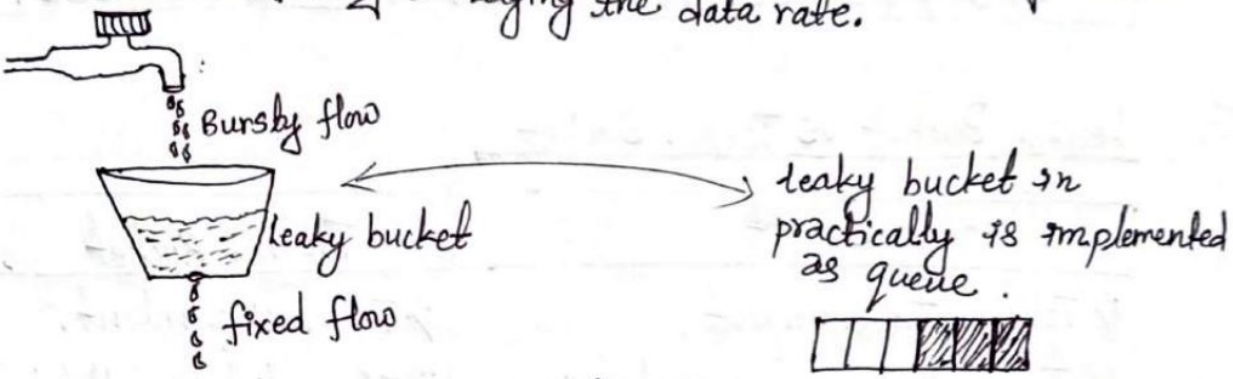


Fig. Demonstrating leaky bucket concept.

Algorithm:

Step1: Initialize the counter to 'n' at every tick of clock.
Step2: If n is greater than the size of packet in the front of queue send the packet into the network and decrement the counter by size of packet. Repeat the step until n is less than the size of packet.
Step3: Reset the counter and go to step-1.

Token bucket → The token bucket algorithm allow to vary the output rate depending on the size of brust. In this algorithm the bucket holds token to transmit a packet, the host must capture and destroy one token. Token are generated by a clock at the rate of one token every second.

Algorithm:

Step1: A token is added at every Δt time.
Step2: The bucket can hold b-token. If a token arrive when bucket is full, it is discarded.

Step3: When a packet of m bytes arrived m tokens are removed from the bucket and the packet is sent to the network.

Step4: If less than m tokens are available no tokens are removed from the buckets and the packet is considered to be non conformant.

Note: May be these formulas important if numericals asked:

i) Brust Length = $\frac{\text{Capacity of bucket (In kb)}}{(\text{Output rate} - \text{Arrival rate}) * 1000}$ = ... msec (unit)

(Output rate — In mbps; Capacity of bucket considered 500 kb (let))

ii) For another 500 kb the time taken will be,

$\frac{\text{Capacity of bucket}}{\text{Arrival rate} * 1000}$ = let we get 250 msec.

∴ Output time = Brust length + 250 = ... msec.

Leaky Bucket vs Token Bucket

Leaky Bucket	Token Bucket
i) Token Independent	i) Token Dependent.
ii) If bucket is full packet is discarded.	ii) If bucket is full token are discarded but not the packet.
iii) Bucket leaks at constant rate.	iii) Bucket has maximum capacity.
iv) Packets are transmitted continiously.	iv) Packets can only be transmitted when there are enough token.
v) It does not save token	v) It saves token to send large bursts.

⊛. Techniques to Improve QOS:

1) Scheduling → A good scheduling technique treats the different flows of packets in a fair and appropriate manner. Several scheduling techniques are designed to improve the quality of service. FIFO queuing, priority queuing and weighted fair queuing are some of those techniques.

i) FIFO Queuing → In FIFO queuing, packets wait in a queue until the node 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.

ii) 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.

iii) Weighted Fair Queuing → It is a better scheduling technique. In which the packets are assigned to different classes and admitted to different queues. The queues are weighted based on the priority of queues; higher priority means a higher weight. The system processes packets in each queue in a round-robin fashion.

2) Traffic Shaping → Already Discussed.

3) 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. This section consist of QoS model called Integrated Services, which depends heavily on resource reservation to improve the quality of service.

4) 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 and its previous commitments to other flows can handle the new flow.

⊛ Queuing Techniques for Scheduling:-

FIFO queuing, Priority queuing & Weighted fair queuing discussed above are queuing techinques for scheduling.

[already discussed]

⊛ Introduction to Ports and Sockets:

Port → A port is a logical construct assigned to network processes so that they can be identified within the system. The word "Port" is the number used by the particular software. The same port number can be used in different computer running on same software. A port is a communication endpoint.

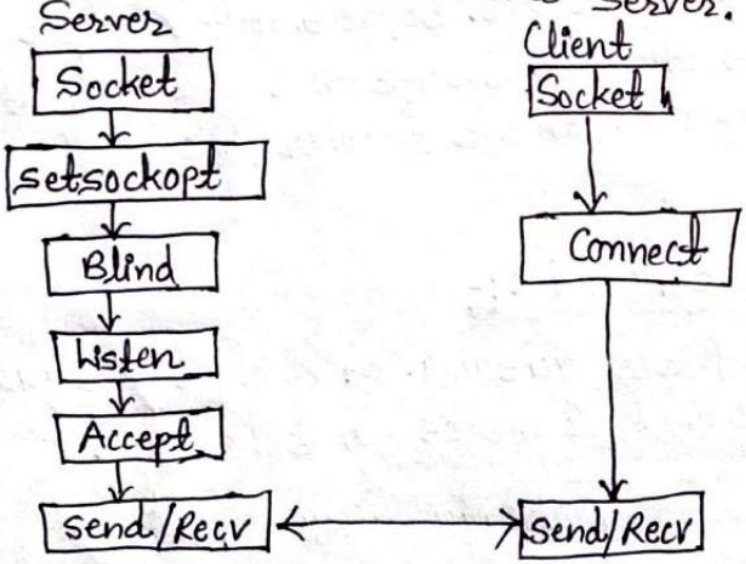
A port number is a 16-bit unsigned integer thus ranging from 0 to 65535. For TCP, port number 0 is reserved and cannot be used while the source port is optional and a value of zero means no port for UDP.

Socket → A socket is a combination of port and IP address. The word "socket" is the combination of port and IP address. It is used to identify both a machine and a service within the machine. A socket is one endpoint of a two-way communication link between two programs running on the network.

In networking, a socket is used to allow many processes within a single or different host to use TCP communication simultaneously. The socket is formed by including the IP address with the port number to uniquely identify seperate data stream.

⊛. Socket Programming:

Socket Programming is a way of connecting two nodes on a network to communicate with eachother. One socket (node) listens on a particular port at an IP, while other socket reaches out to the other to form a connection. Server forms the listener socket while client reaches out to the server.



This is a state diagram for server and client model.