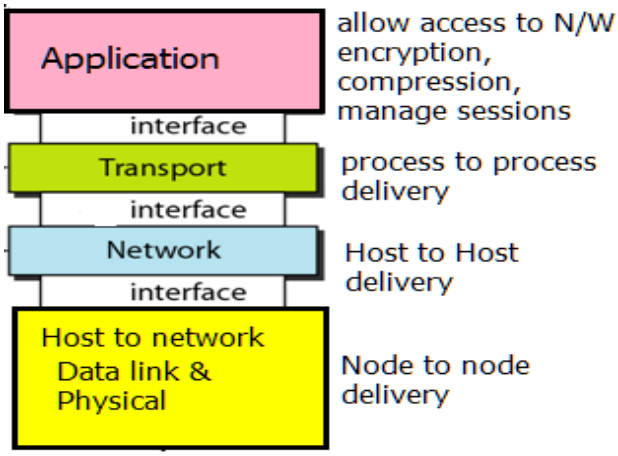
CNS (2019) Model Answers
Unit 4 : Transport layer

Explain Transport layer function

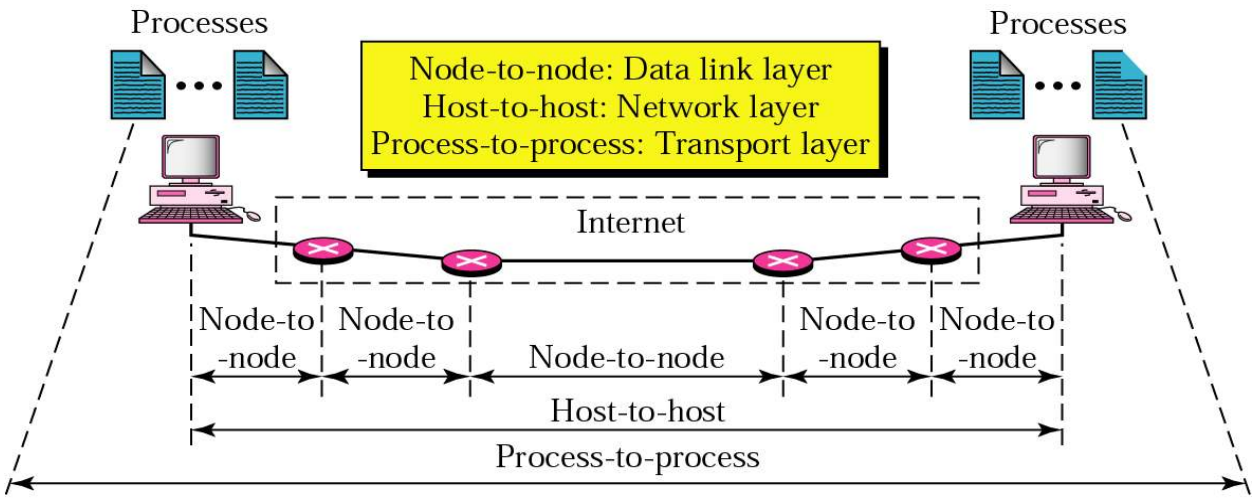
Main Function of Transport layer : Process-to-Process Communication
Transport layer provides services to upper layer i.e. application layer
Transport layer takes services from network layer



Functions of Transport layer

- ✓ Process-to-Process Communication
- ✓ Addressing: Port Numbers
- ✓ Encapsulation and Decapsulation (Segmentation and reassembly)
- ✓ Multiplexing and Demultiplexing
- ✓ Flow Control
- ✓ Error Control
- ✓ Congestion Control
- ✓ Connectionless and Connection-Oriented Services

What is the difference between function of data link layer, network layer & Transport Layer?



Analogy

DLL: Node to node : from sender house to post office in that city to post office in city of receiver to house of receiver

NL: Host to host : from sender house to receiver house

TL: process to process : A letter to one particular person , out of many persons living in same house

Explain port number

port numbers are assigned to processes (applications/ programs) which are communicating to each other.

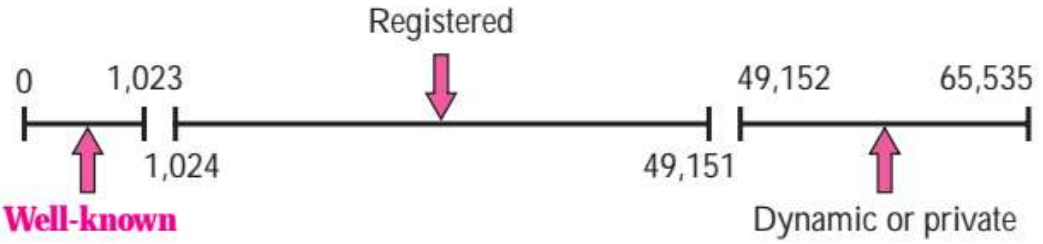
In the TCP/IP protocol suite, the port numbers are between 0 and 65,535.

The client program is assigned a port number, called the **ephemeral port number.** The word ephemeral means *short lived* because the life of a client is normally short. An ephemeral port number is recommended to be greater than 1,023 .The server processes are assigned universal port numbers; these are called **well-known port** numbers.

ICANN Ranges

ICANN has divided the port numbers into three ranges: well-known, registered, and dynamic (or private),

- **Well-known ports.** The ports ranging from 0 to 1,023 are assigned and controlled by ICANN. These are the well-known ports.
- **Registered ports.** The ports ranging from 1,024 to 49,151 are not assigned by ICANN. But They need to be registered with ICANN to prevent duplication.
- **Dynamic ports.** The ports ranging from 49,152 to 65,535 are neither assigned nor registered. They can be used as temporary or private port numbers

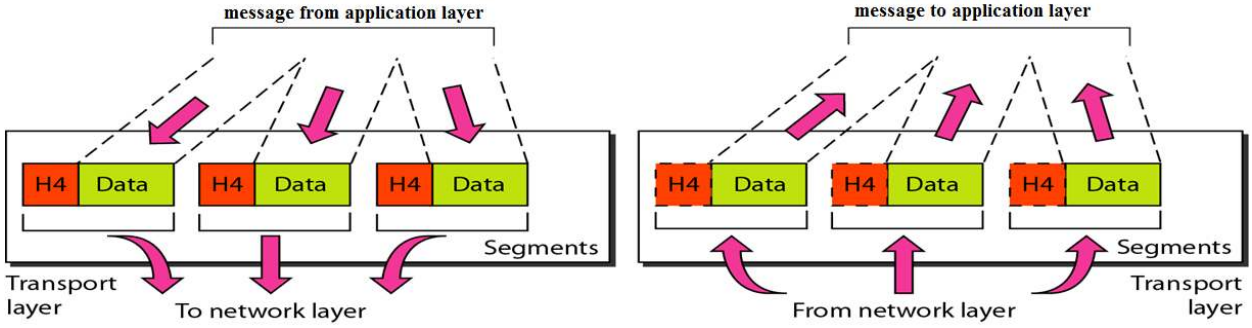


List & explain in brief the Services provided by Transport Layer

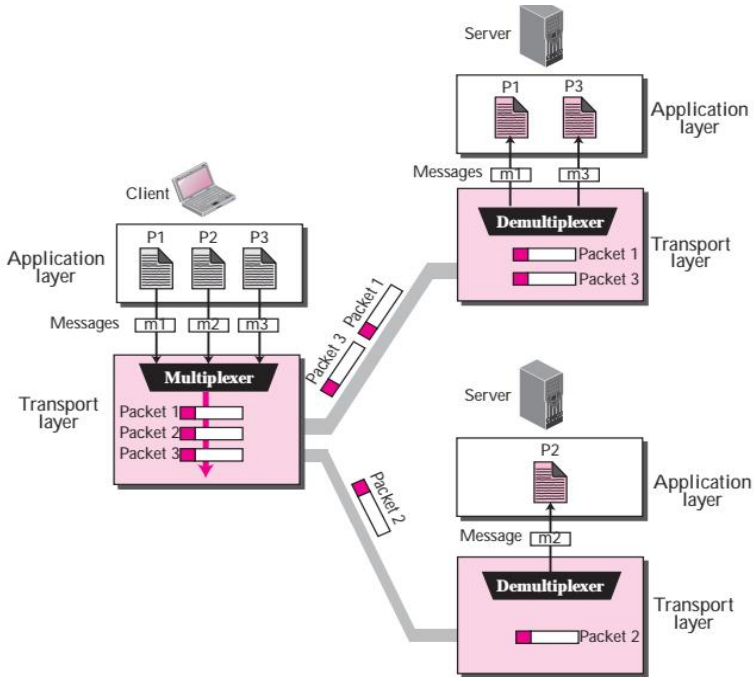
- ✓ Process-to-Process Communication
- ✓ Addressing: Port Numbers
- ✓ Encapsulation and Decapsulation (Segmentation and reassembly)
- ✓ Multiplexing and Demultiplexing
- ✓ Flow Control
- ✓ Error Control
- ✓ Congestion Control
- ✓ Connectionless and Connection-Oriented Services

Process-to-Process Communication: the transport layer is responsible for communication between process in source node to a process in destination node.
Addressing using Port Numbers : each communicating process is assigned an address called port number.
Encapsulation and Decapsulation (Segmentation and reassembly):

- transport layer in sender it puts message coming from application layer into segments (also called packets) (Encapsulation)
- At receiver it decapsulates segments & sends message to application layer

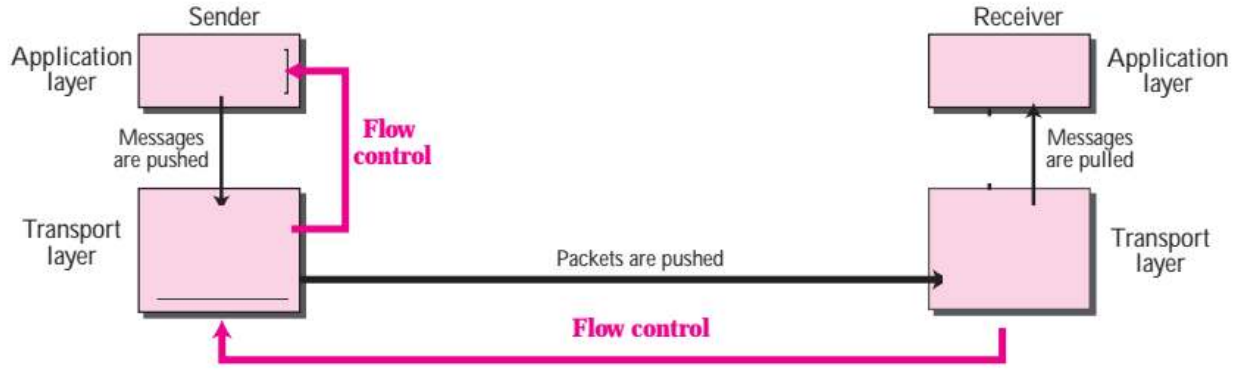


Multiplexing and Demultiplexing
Since there can be many process in the same sender which are communicating with processes in different destinations . the messages are multiplexed by transport layer in the sender.
In the destination the messages are demultiplexed



Flow control ;

When application layer in the sender sends messages faster than its transport layer can accept , or when sender transport layer sends segments faster than receiver transport layer can accept , then there is segment loss. So Flow control is required between application layer & transport layer of the sender & between transport layer of the sender & transport layer of the receiver.



Error Control

at the transport layer is responsible to

1. Detect and discard corrupted packets (segments).

2. Keep track of lost and discarded packets and resend them.

3. Recognize duplicate packets and discard them.

4. store out-of-order packets until the missing packets arrive

connectionless and connection-oriented

In a connection-oriented service, the client and the server first need to establish a connection between themselves. The data exchange can only happen after the connection establishment. After data exchange, the connection needs to be teared down.

Congestion Control (traffic jam)

Congestion in a network may occur if 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).

Analogy : when there is accident on a road , there is traffic jam

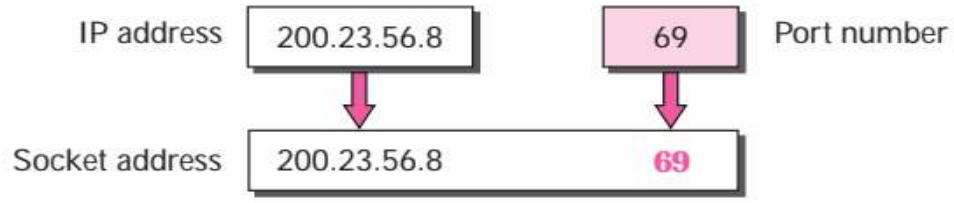
Congestion control handles this.

It can be preventive (before it happens) (also called Open-Loop Congestion Control) as well as corrective (after it happens) (also called Closed-loop congestion control)

Socket Programming.

Socket is a pair of IP address & port address .

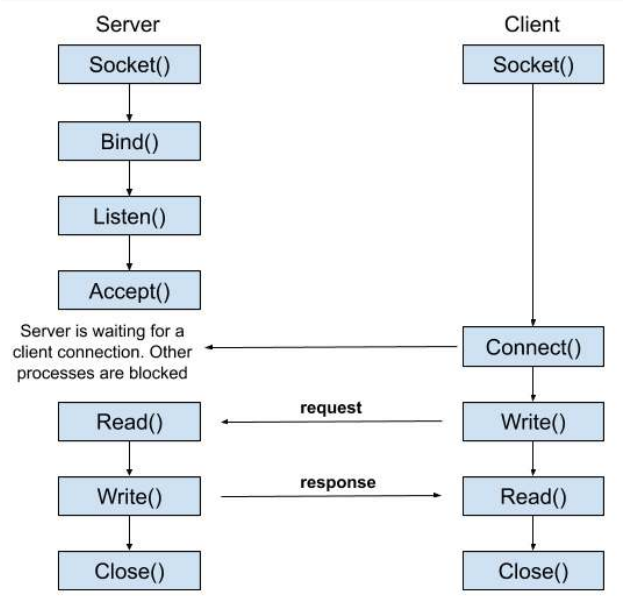
Both client & server have their own socket address : client socket address and the server socket address



Socket programming is used for communication between the applications running on client & server machines

Socket and ServerSocket classes are used for TCP which is connection-oriented and Datagram Socket and Datagram Packet classes are used for UDP which is connection-less.

Socket programming structure



Socket class

The Socket class is used to create a socket.

Important methods

Method	Description
1) public InputStream getInputStream()	returns the InputStream attached with this socket.
2) public OutputStream getOutputStream()	returns the OutputStream attached with this socket.
3) public synchronized void close()	closes this socket

ServerSocket class

The ServerSocket class can be used to create a server socket. This object is used to establish communication with the clients.

Important methods

Method	Description
1) public Socket accept()	returns the socket and establish a connection between server and client.
2) public synchronized void close()	closes the server socket.

Write a socket program for communication between client & server process

File: MyServer.java

```
1. import java.io.*;
```

```
2.  import java.net.*;
3.  public class MyServer {
4.  public static void main(String[] args){
5.  try{
6.  ServerSocket ss=new ServerSocket(6666);
7.  Socket s=ss.accept();//establishes connection
8.  DataInputStream dis=new DataInputStream(s.getInputStream());
9.  String  str=(String)dis.readUTF();
10. System.out.println("message= "+str);
11. ss.close();
12. }catch(Exception e){System.out.println(e);}
13. }
14. }
```

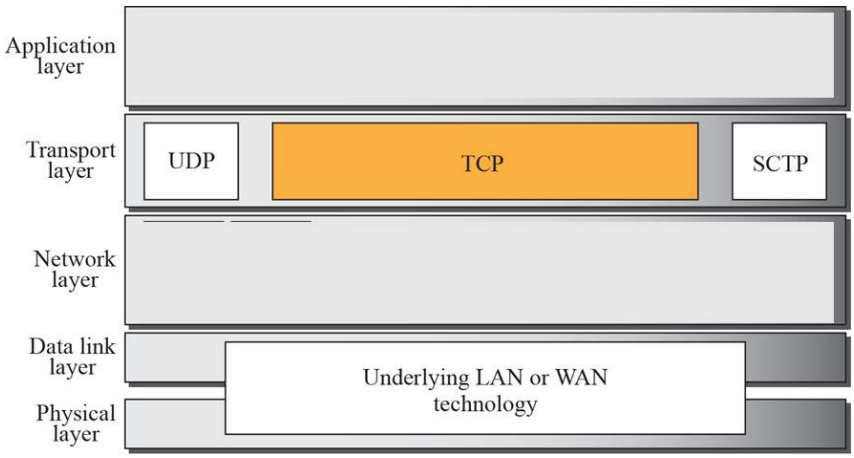
File: MyClient.java

```
1.  import java.io.*;
2.  import java.net.*;
3.  public class MyClient {
4.  public static void main(String[] args) {
5.  try{
6.  Socket s=new Socket("localhost",6666);
7.  DataOutputStream dout=new DataOutputStream(s.getOutputStream());
8.  dout.writeUTF("Hello Server");
9.  dout.flush();
10. dout.close();
11. s.close();
12. }catch(Exception e){System.out.println(e);}
13. }
14. }
```

Explain TCP

Transmission Control Protocol

It is a protocol used at transport layer
It is used for process to process communication
It is connection-oriented & reliable & full duplex



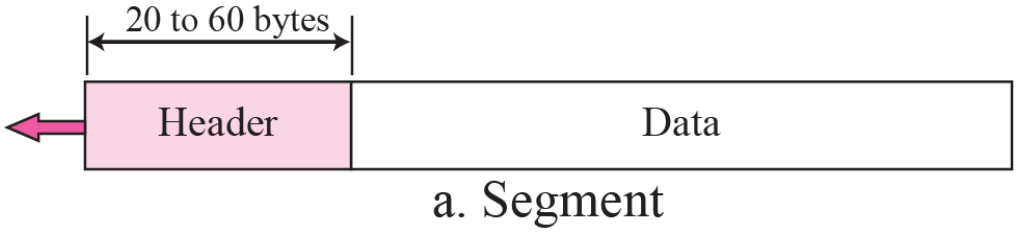
Connection-Oriented Service : means client TCP & server TCP establish connection before data transfers

Reliable Service **:** (error control) means it uses an acknowledgment mechanism to guarantee correct data transfer

full-duplex service, means segments can flow in both directions at the same time

SEGMENT

A packet in TCP is called a **segment.**



a. Segment

Header format

Source port address 16 bits								Destination port address 16 bits
Sequence number 32 bits								
Acknowledgement number 32 bits								
HLEN 4 bits	Reserved 6 bits	URG	ACK	PSH	RST	SYN	FIN	Window size 16 bits
Checksum 16 bits								Urgent pointer 16 bits
Options & padding								

Source port address**.** This is a 16-bit port number of the application program in the host that is sending the segment

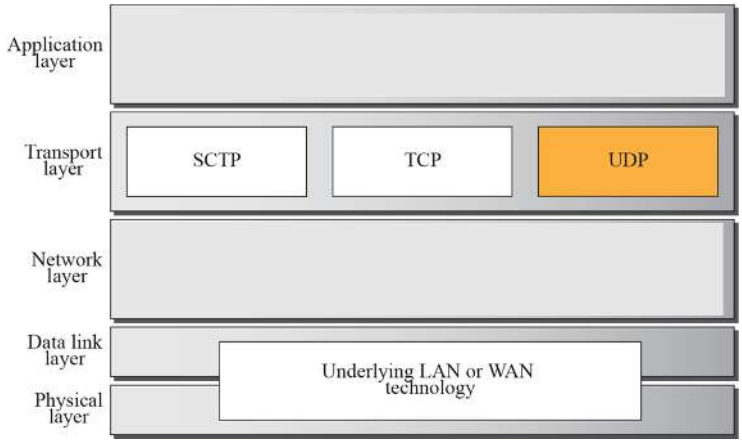
Destination port address**.** This is a 16-bit port number of the application program in the host that is receiving the segment.
Sequence number**.** This 32-bit field defines the number assigned to the first byte of data contained in this segment
Acknowledgment number..If the receiver of the segment has successfully received byte number *x* from the other party, it returns *x* + 1 as the acknowledgment number.
Header length. indicates the length of the TCP header.
Control.. These bits enable flow control, connection establishment and termination, connection abortion, and the mode of data transfer in TCP.
Window size**.** Receiver tells the sender what should be the window size , used for flow control
Checksum**.** Calculated & send by sender , is used by receiver transport layer to check errors in header of received segment.
Urgent pointer. is used when the segment contains urgent data

TCP at sender & receiver establish connection before actual data transfer by using three way handshake and also use three way handshake after data transfer .

Applications of TCP
Used by application layer protocols HTTPS & HTTP, (for web sites) , SMTP& POP (for emails) , FTP (for file transfers), etc

UDP

User datagram protocol
It is a protocol used at transport layer



UDP is a connectionless, unreliable protocol
In UDP segments are called datagrams
Connectionless : means sender does not establish connection with the receiver , it just starts sending segments.
unreliable protocol : means no flow and error control used . So no guarantee of data transfer. It is used where errors can be accepted eg audio & video streaming
UDP header

Is 8 byte size

Header format

Source port number 16 bits	Destination port number 16 bits
Total length 16 bits	Checksum 16 bits

Source port number : This is the port number of the process running on the source host which has created this datagram.
Destination port number This is the port number of the process running on the destination host to which this datagram is being sent.
Length. This is a 16-bit field that defines the total length of the user datagram, header plus data. The 16 bits can define a total length of 0 to 65,535 bytes
However, the total length needs to be much less because a UDP user datagram is put in an IP datagram by network layer with the total length of 65,535 bytes.
Checksum. This field is used to detect errors over the entire user datagram (header plus data). Unlike TCP, the Checksum calculation is not mandatory in UDP. No Error control or flow control is provided by UDP. Hence UDP depends on IP and ICMP for error reporting.

UDP services

- ✓ Process-to-Process Communication
- ✓ Connectionless Service
- ✓ Congestion Control
- ✓ Encapsulation and Decapsulation
- ✓ Queuing
- ✓ Multiplexing and Demultiplexing

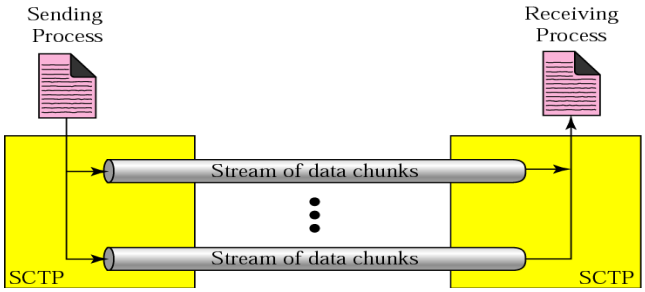
Applications of UDP:
Video conferencing, gaming , streaming, DNS, VoIP, etc

SCTP

Stream Control Transmission Protocol (SCTP)

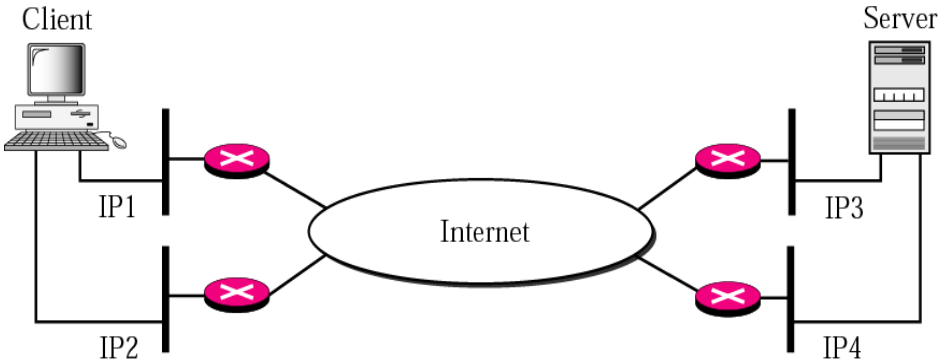
- is a new transport layer protocol.
- Full duplex, connection oriented , reliable, message-oriented
- SCTP Combines the best features of UDP and TCP

Multiple Streams between sender & receiver



Multihoming

A TCP connection involves only one source and one destination IP address, But SCTP can allow multiple connections between sender & receiver which are having multiple IP addresses since they are connected to multiple networks simultaneously.



Header format

Source port address 16 bits	Destination port address 16 bits
Verification tag 32 bits	
Checksum 32 bits	

Flow control :

in SCTP is similar to that in TCP. In SCTP, we need to handle two units of data, the byte and the chunk.

Error control :

SCTP uses a SACK (selective ACK) chunk to report the state of the receiver buffer to the sender.

congestion control :

SCTP uses the same strategies for congestion control as TCP. SCTP uses slow start, congestion avoidance, and congestion detection phases. SCTP also uses fast retransmission and fast recovery.

Applications of SCTP :

SCTP, designed for new Internet applications which need a more sophisticated service than TCP can provide.

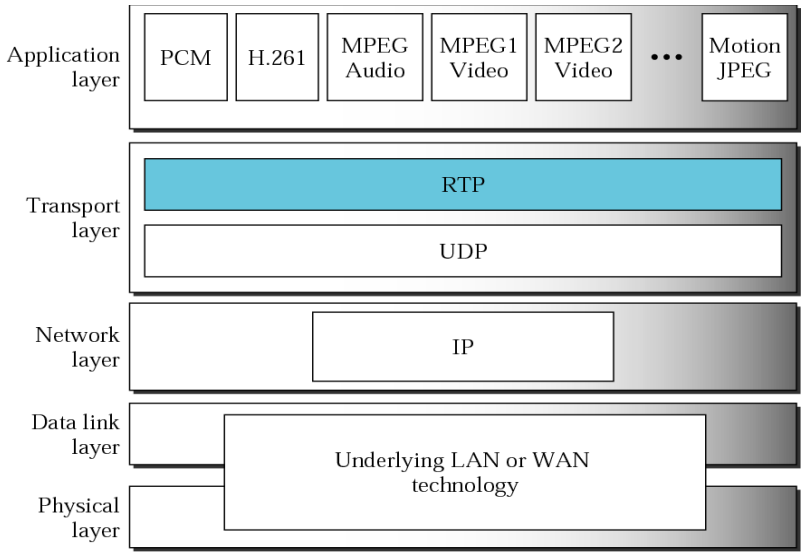
List of Applications using SCTP

Protocol	*Description*
IUA	ISDN over IP
M2UA	SS7 telephony signalling
M3UA	SS7 telephony signalling
H.248	Media gateway control
H.323	IP telephony
SIP	IP telephony

RTP

Real-time Transport Protocol (RTP)

is the protocol designed to handle real-time traffic on the Internet. Since RTP does not have a delivery mechanism (multicasting, port numbers, and so on); it must be used with UDP. RTP stands between UDP and the application program. The main contributions of RTP are time stamping, sequencing, and mixing facilities.



RTP is carried inside UDP Although RTP is itself a transport layer protocol, Since RTP does not have a delivery mechanism (multicasting, port numbers, and so on);

No well-known port is assigned to RTP. The port can be selected on demand with only one restriction: The port number must be an even number.

List of applications which use RTP

PCMμ Audio	LPC audio	G728 audio
1016	PCMA audio	Motion JPEG
G721 audio	G722 audio	H.261
GSM audio	L16 audio	MPEG1 video
DV14 audio	MPEG audio	MPEG2 video

Compare TCP UDP & SCTP

	UDP	TCP	SCTP
Data unit	message	Byte (Stream)	message
Connection type	Connection less	Connection oriented	Connection oriented
Casting	Multicasting & broadcasting	Unicasting	Multicasting & broadcasting
Reliability	unreliable	reliable	reliable
congestion control	No	Yes	Yes
flow control	No	Yes	Yes
streaming	single	Single	Multiple
homing	single	single	multiple
security	NO	NO	Yes
applications	Video conferencing, streaming, DNS, VoIP, etc	HTTPS, HTTP, SMTP, POP, FTP, etc	IP telephony ,SS7

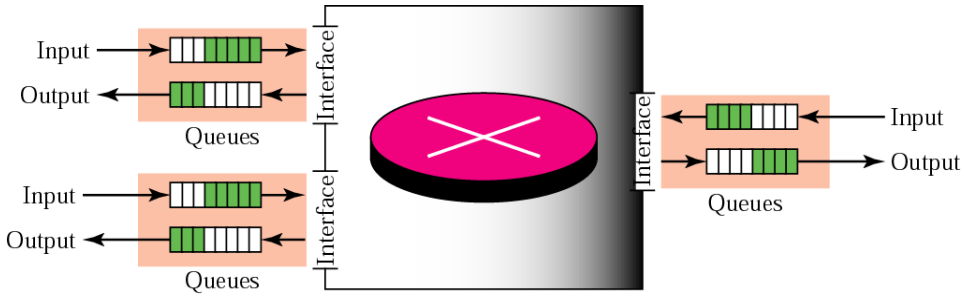
What is Congestion ? why it occurs?

Congestion in a network may occur if the **load** on the network(the number of packets sent to the network)is greater than the ***capacity*** of the network(the number of packets a network can handle).

Congestion = **load** on the network > ***capacity*** of the network

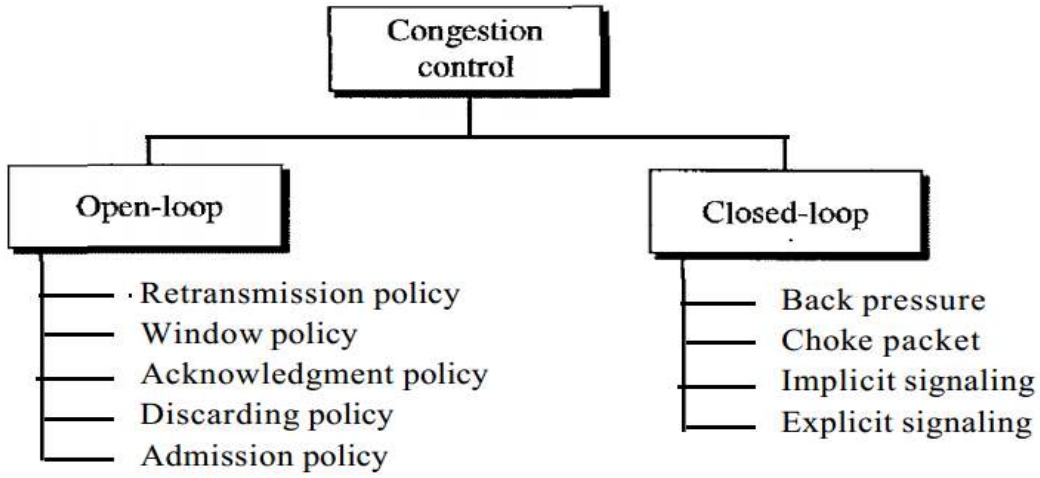
So Congestion control refers to the mechanisms and techniques to control the congestion and keep the load below the capacity.
Analogy : Toll booth processes less vehicles per minute than vehicle arriving per minute.

Congestion in a network occurs because routers and switches have queues (buffers that hold the packets before and after processing)These queues are finite, so it is possible for more packets come at router than the router can buffer
If the rate of packet arrival is higher than the Packet processing rate, the input queues become longer and longer. Second, if the packet departure rate is less than the packet processing rate, the output queues become longer and longer.



Congestion control techniques.

Congestion control refers to the mechanisms and techniques to control the congestion and keep the load below the capacity.
Classification of Congestion control techniques



Open-Loop Congestion Control

In this, policies are applied to prevent congestion **before** it happens. In these mechanisms, congestion control is handled by either the source or the destination.

1.Retransmission Policy Retransmission of packets in case of errors can increase congestion in the network. So a good retransmission policy can prevent congestion

2.Window Policy The type of window at the sender can also affect congestion. the *Selective Repeat* window is better than the *GoBack-N* window for congestion control.

3.Acknowledgment Policy. acknowledgments are sent from receivers to senders, which is also a load on network . So a good acknowledgments policy can prevent congestion

4.Discarding Policy, in this router drops less critical packets . eg packets carrying audio but quality of sound is still preserved.

5. Admission policy, used to 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 .

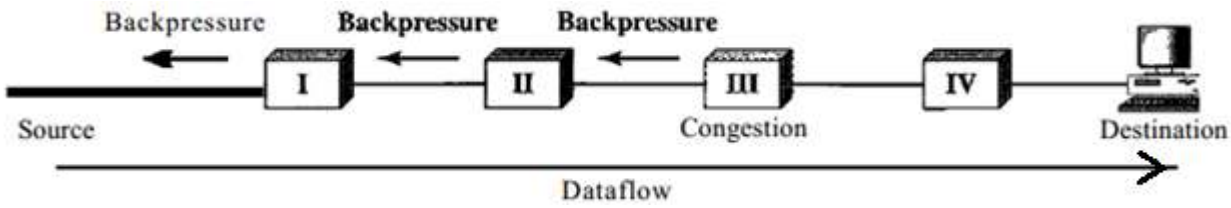
Closed-loop congestion control

try to alleviate congestion **after** it happens.

The size of the window at the sender size can be changed based on the congestion in the Internet. The sending transport layer can decrease the window size if the congestion is increasing and vice versa.

1.Backpressure

in which a Congested node stops receiving data from the immediate upstream node or nodes. This may cause the upstream node or nodes to become congested and they, in turn, reject Data from their upstream nodes or nodes.



2. Choke Packet

is a packet sent by a node to the source to inform it of congestion. In between nodes are not informed of congestion.

3. Implicit Signaling

In this, there is no communication between the congested node or nodes and the source. when source does not get ACKs for long time, the source guesses that there is a congestion somewhere in the network. Then it slows down the rate of sending packets .

4. Explicit Signaling

The node that experiences congestion can explicitly send a signal to the **source or destination**. This is different from the choke packet method. In the choke packet method , a separate packet is used for this purpose;

in the Explicit signaling method, the signal is included in the data packets itself

Congestion control in TCP

Is based on both open-loop and closed-loop mechanisms.
TCP uses a congestion window and a congestion policy for both : to avoid congestion (open loop) and also to detect and alleviate congestion after it has occurred (closed loop).

Congestion Window (cwnd)

In flow control TCP uses two windows :send & receive window.
In congestion control , the network (routers) also decide size of send window by setting congestion window size .
So The sender has two pieces of information: the receiver-advertised window size and the congestion window size. The actual size of the send window is the minimum of these two.

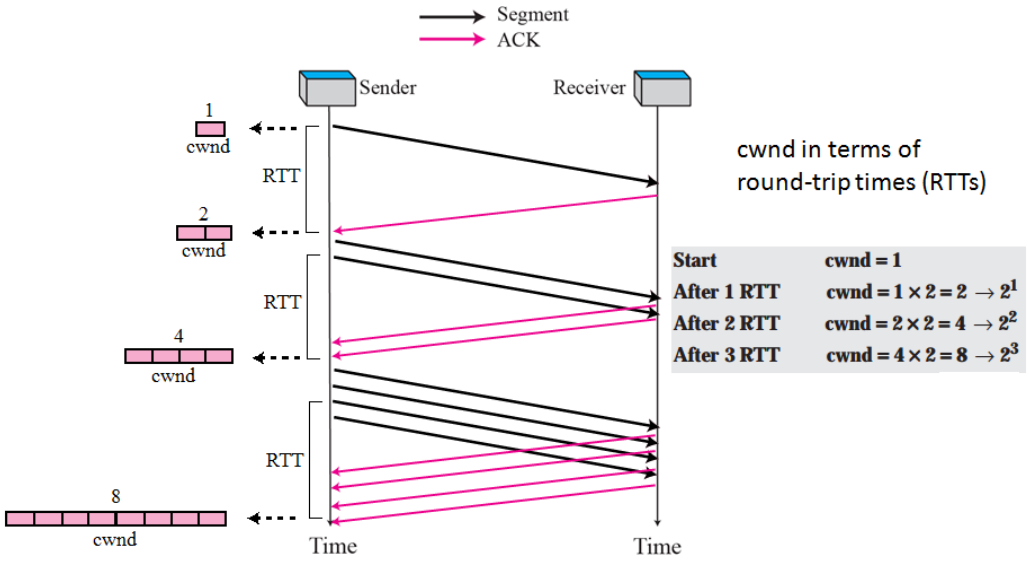
Actual send window size == minimum (rwnd, cwnd)

Congestion Policy

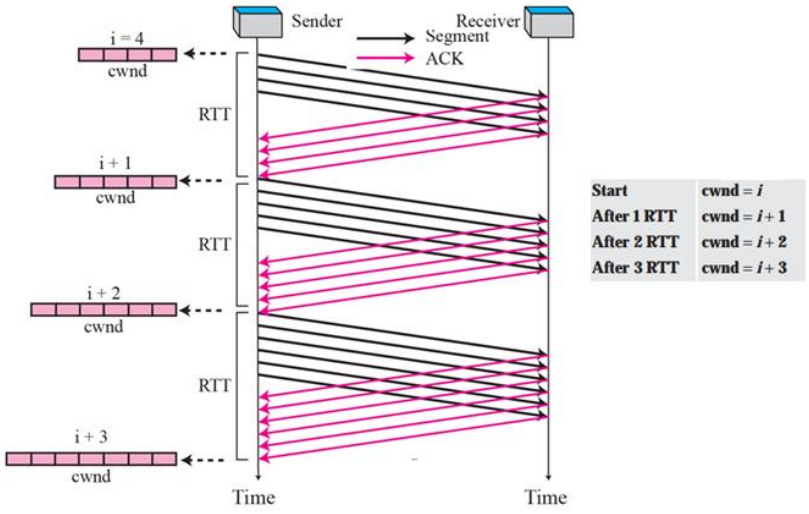
TCP's general policy for handling congestion is based on three phases:

1. slow start,
2. congestion avoidance, and
3. congestion detection.

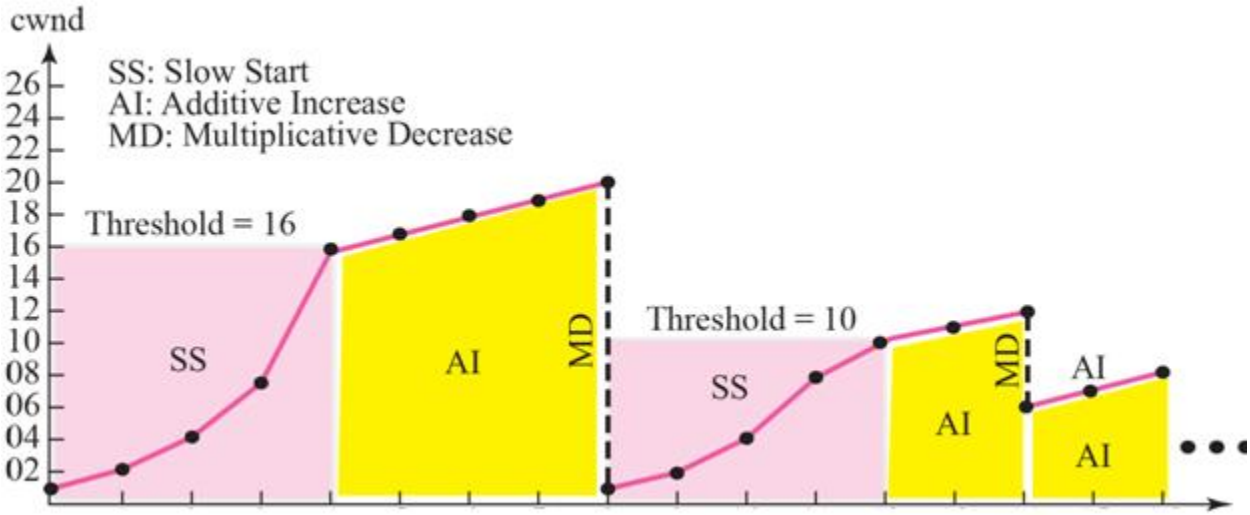
1. In slow start phase, the sender starts with a slow rate of transmission, but increases the rate geometric fashion. eg. $2^0 = 1$, $2^1 = 2$, $2^2 = 4$ so on



2. congestion avoidance : When the threshold is reached, the rate of increase is arithmetic. Eg i+1 . i+2 . i+3 & so on

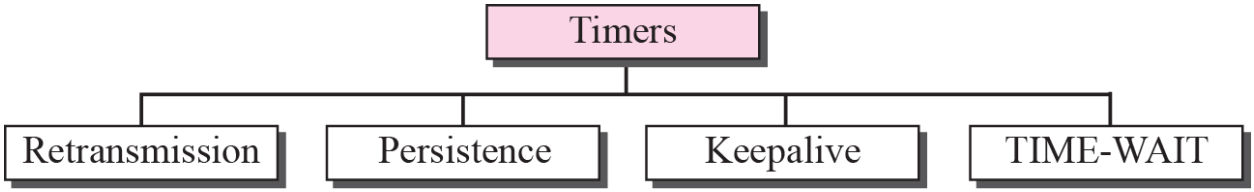


3. Finally if ever congestion is detected, the sender goes back to the slow start or congestion avoidance phase, based on how the congestion is detected.



Timers used in TCP

TCP uses four timers as shown



1.Retransmission Timer

TCP employs one retransmission timer (for the whole connection period) that is set equal to the retransmission time-out (RTO)

RTO is the time required by ACK (sent by receiver) to reach the sender
When Retransmission Timer is over (i.e. ACK does not come within RTO time) the sender retransmits the segment

2.Persistence Timer

If the receiving TCP tells sender that keep window size of zero, the sending TCP stops transmitting segments.
Sending TCP can start transmitting segments only after the receiving TCP sends an ACK segment announcing a nonzero window size.
But This ACK segment can be lost on the way to sender.
Since sender does not get it , it will keep on waiting indefinitely.
So avoid this , when sender receives window size of zero from receiver , it (sender) starts Persistence Timer . The value of the persistence timer is set to the value of the retransmission time.
When it is over , sender sends special segment called probe. Upon receiving this probe segment , the receiver resends the ACK with nonzero windows size.
if a response is not received from the receiver, sender sends another probe segment and the value of the persistence timer is doubled and reset ,
It is repeated till it becomes 60 seconds, then probe is after every 60s.

3.Keepalive Timer

A keepalive timer is used 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.
Eg the client has crashed. In this case, the connection remains open forever.
To remedy this situation, a server has a keepalive 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.

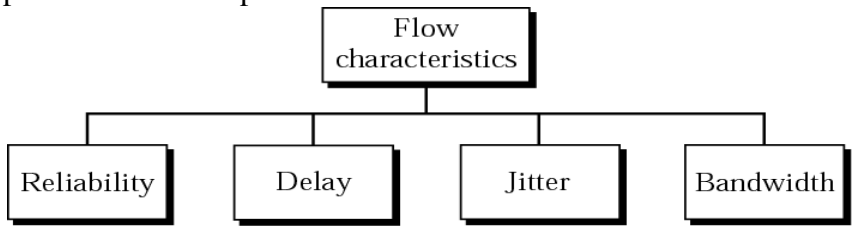
4. TIME-WAIT Timer

The TIME-WAIT timer is used during connection termination

Quality of Service (QoS)

Quality of service (QoS)

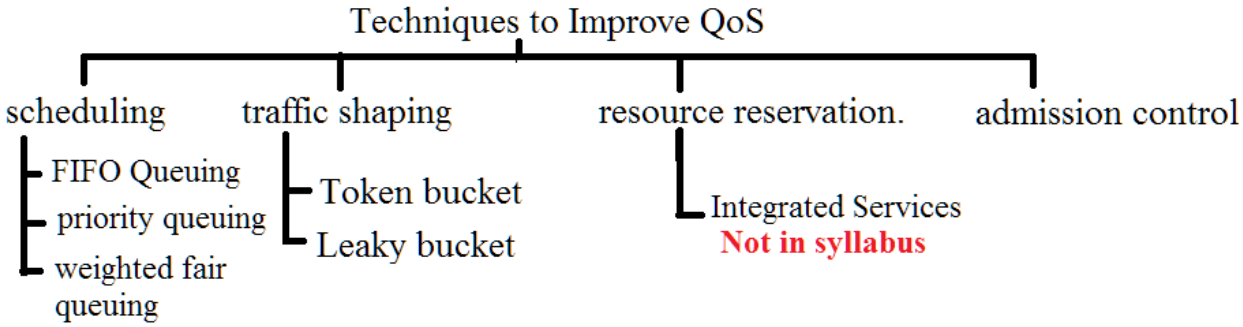
Is the measurement of the overall performance of a service.
Overall performace is can be measured from flow characteristics.
Some parameters of QoS is more important for multimedia (audio , video ,VOIP etc) and some parameters are important for text data.



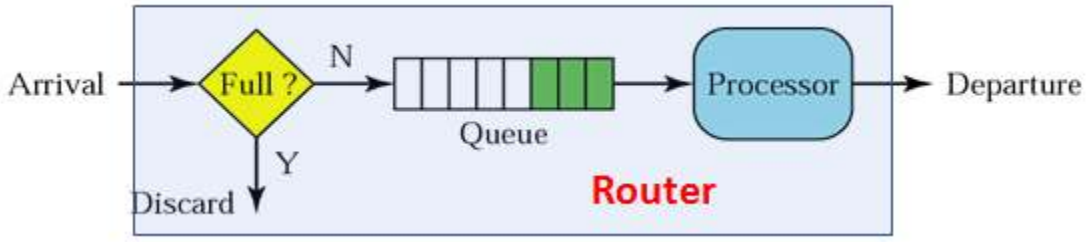
Reliability (Accuracy)
means network should not loose a packet or acknowledgment, it makes it possible by ACK & retransmission of lost packet or ACK.
Reliability is more important for electronic mail, file transfer, etc than telephony or audio conferencing.
Delay (Timeliness) Source-to-destination delay
It is time for packets to reach destination.
Applications such as telephony, audio conferencing, video conferencing, and remote log-in need minimum delay, while delay in file transfer or e-mail is less important.
Jitter
It is variation in delay for packets belonging to the same flow.
For example, if four packets depart at times 0, 1, 2, 3 and arrive at 20, 21, 22, 23, all have the same delay of 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: 21, 22, 19, and 24.
For applications such as audio and video, second case is not suitable.
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 e-mail may be less than a million.

Techniques to Improve QoS

four common methods:

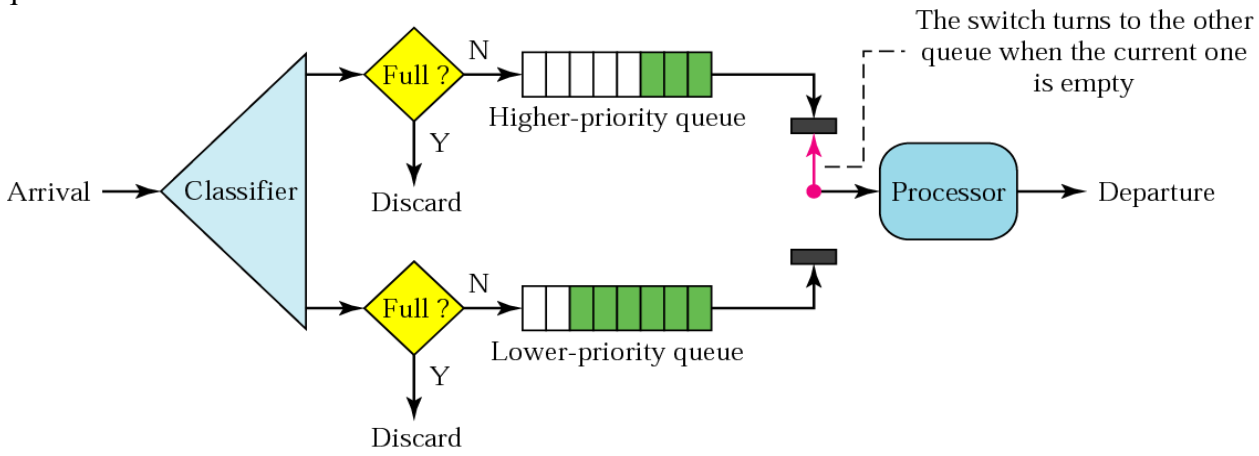


first-in, first-out (FIFO) queuing
When input queue of a router is full , the new packets are discarded.



Priority queuing
packets are assigned a priority class by sender.
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. Figure shows priority queuing with two priority levels (for simplicity).

Note that the system stops serving a queue only when it becomes empty, then it serves another queue

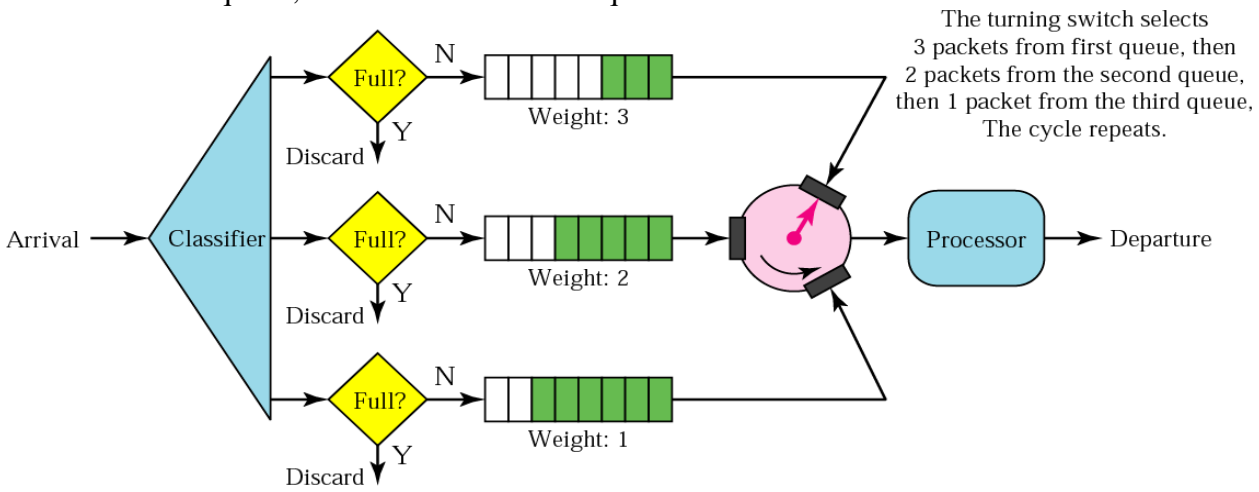


A priority queue can provide better QoS than the FIFO queue , but If there is a continuous flow in a high-priority queue, the packets in the lower-priority queues will never have a chance to be processed. This is a condition called starvation.

Weighted fair queuing

Here also packets are assigned different priorities. But difference is , in this system does not wait for a Q to be empty to swtich to lower proirity Q.

Instead it processes different number of packets from all Qs , in proportion to the priorities . 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



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

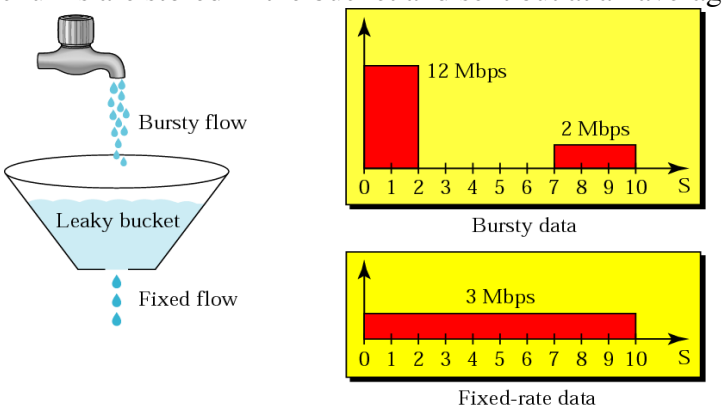
Explain Leaky Bucket

It is a congestion control technique used for QoS.

Analogy :

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

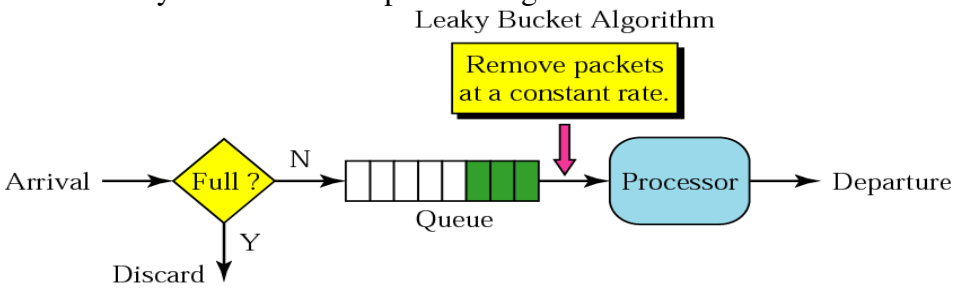
at which the water is input to the bucket unless the bucket is empty. The input rate can vary, but the output rate remains constant.
Similarly, in networking, a technique called leaky bucket can smooth out bursty traffic. Bursty chunks are stored in the bucket and sent out at an average rate



In Figure 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 same rate of 3 Mbps during the same 10 s.

Leaky bucket implementation

A FIFO queue is used to hold the packets. If the traffic consists of fixed-size packets (e.g., cells in ATM networks), the process sends 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 is based on the number of bytes or bits in the packets together.

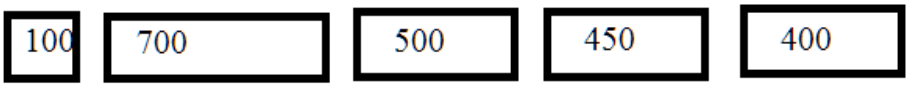


Eg . Packets with differnet number of bytes

Assume : Ticks after every 1ms

1. set N = 1000

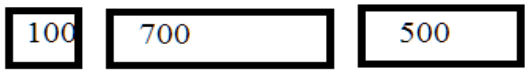
N(1000) is greater than 200 so send packet(200) ,



Now N= 1000 -200 =800 , which is greater than 400 so send next packet(400)

Now N= 800-400=400 , which is less than 450 , packet can't be sent. **so wait for next tick**.
When next tick comes , go to step 1.
N is again set to 1000

1000> 450 , so send packet(450)

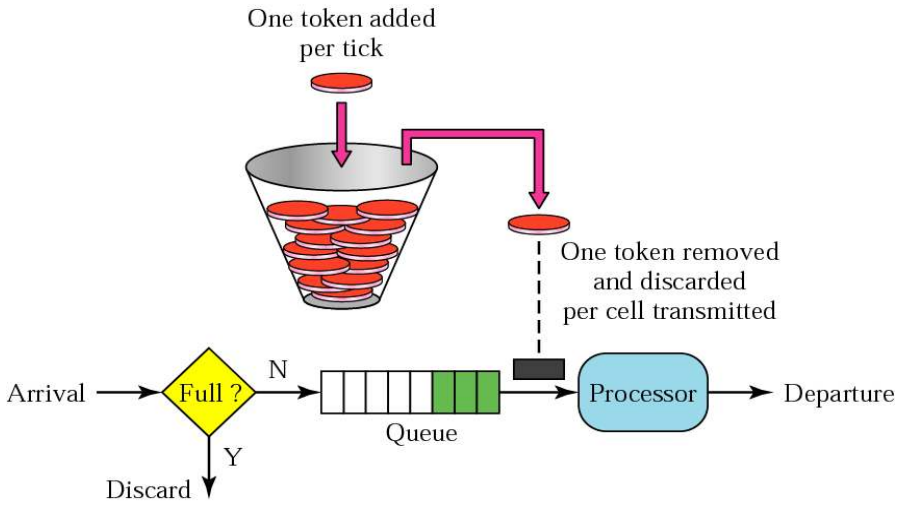


N= 1000-450 = 550 , which is greater than 500 , so send next packet (500)

N= 550-500 =50 , packet(700) can't be send , . **so wait for next tick**.
When next tick comes , go to step 1.
& so on.
Thus in each milli second , at the most 1000 bytes are send

Explain Token Bucket

It is a congestion control technique used for QoS.
It is an improvement over leaky bucket . Since The leaky bucket does not give credit an idle host. For example, if a host is not sending for a while, its bucket becomes empty. Now if the host has bursty data, the leaky bucket allows only an average rate. The time when the host was idle is not taken into account by leaky bucket.
On the other hand, the token bucket algorithm allows idle hosts to accumulate credit for the future in the form of tokens.
For each tick of the clock, the system sends n tokens to the bucket. 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
The token bucket can easily be implemented with a token counter. The token is initialized to zero. When host does not send packets , in each tick , it gets a token credit . the token counter is incremented by 1. Each time a unit of data is sent, the token counter is decremented by 1. When the token counter is zero, the host cannot send data.
i.e. host can send packets as long as , it has tokens



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 (i.e. for each host , these resources are reserved before they start to transmit)

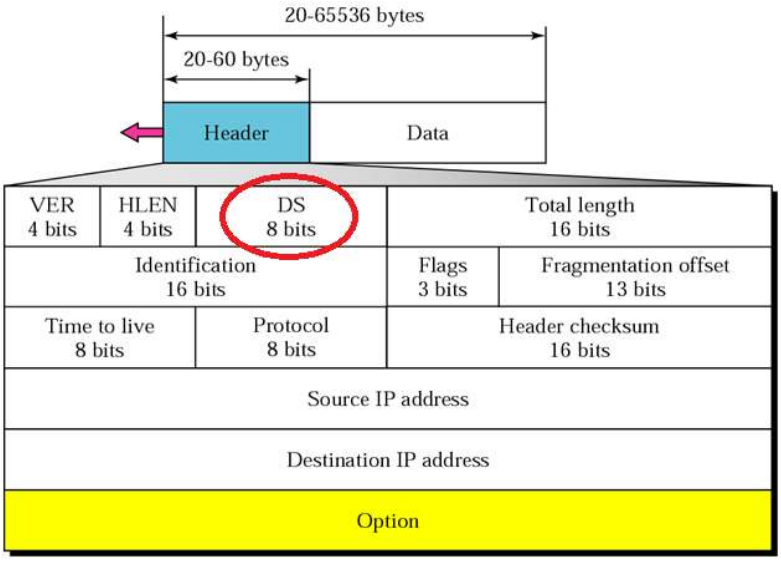
Admission Control
A mechanism used by a router, or a switch,
to accept or reject a flow based on predefined flow specifications. (such as bandwidth, buffer size, CPU speed, etc)
Before a router accepts a flow for processing, it checks its previous commitments to other flows & requirement of new flow . If it is within its capacity , it is admitted else it is denied.

Differentiated services

Two models have been designed to provide quality of service in the Internet:
1. Integrated Services
2. Differentiated Services

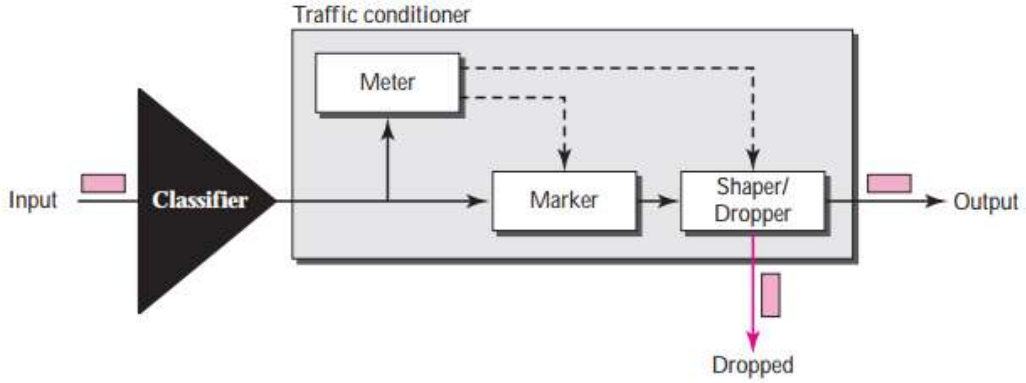
Integrated Services is a flow-based QoS model designed for IP
Differentiated Services (DS or Diffserv) was introduced to overcome the shortcomings of Integrated Services.
Differentiated Services is a class-based QoS model designed for IP.

In Diffserv, each IP packet contains a field called the DS field.



The value of this field is set at the boundary of the network by the host or the first router designated as the boundary router.

To implement Diffserv, the DS node uses traffic conditioners such as meters, markers, shapers, and droppers, as shown in Figure



Meters The meter checks to see if the incoming flow matches the negotiated traffic profile. The meter can use several tools such as a token bucket to check the profile.
Marker A marker can remark a packet that is using best-effort delivery based on information received from the meter. Downmarking (lowering the class of the flow) occurs if the flow does not match the profile. A marker does not up-mark a packet (promote the class).
Shaper A shaper uses the information received from the meter to reshape the traffic if it is not compliant with the negotiated profile
Dropper A dropper, which works as a shaper with no buffer, discards packets if the flow severely violates the negotiated profile.

TCP for Wireless networks

Transport protocols was initially designed for Fixed end-systems & Fixed, wired networks
WTCP ("Wireless Transmission Control Protocol") is used in wireless networks to improve the performance of TCP.

In this the access point (foreign agent) splits the TCP connection in to two. Which both end hosts do not notice