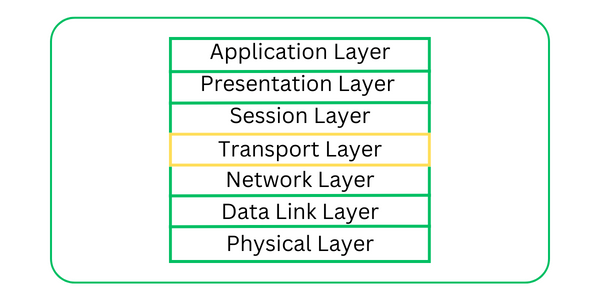
Chapter 4: Transport Layer

* The transport layer is a 4th layer from the top.
* The main role of the transport layer is to provide the communication services directly to the application processes running on different hosts.
* The transport layer provides a logical communication between application processes running on different hosts.
* The unit of data encapsulation in the Transport Layer is a **segment.**

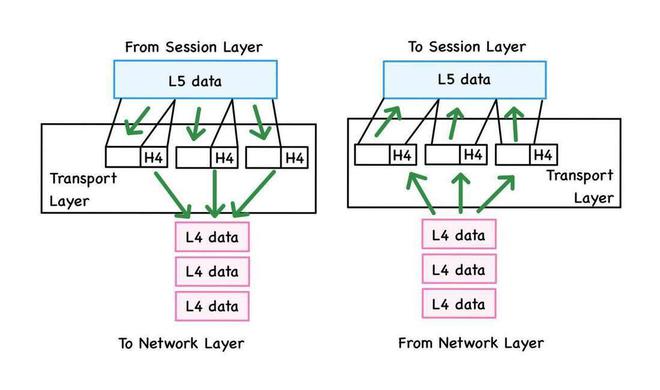


**Working of Transport Layer**

The transport layer takes services from the [Application layer](https://www.geeksforgeeks.org/application-layer-in-osi-model/) and provides services to the [Network layer](https://www.geeksforgeeks.org/network-layer-services-packetizing-routing-and-forwarding/).

**At the sender’s side:** The transport layer receives data (message) from the Application layer and then performs Segmentation, divides the actual message into segments, adds the source and destination’s port numbers into the header of the segment, and transfers the message to the Network layer.

**At the receiver’s side:** The transport layer receives data from the Network layer, reassembles the segmented data, reads its header, identifies the port number, and forwards the message to the appropriate port in the Application layer.

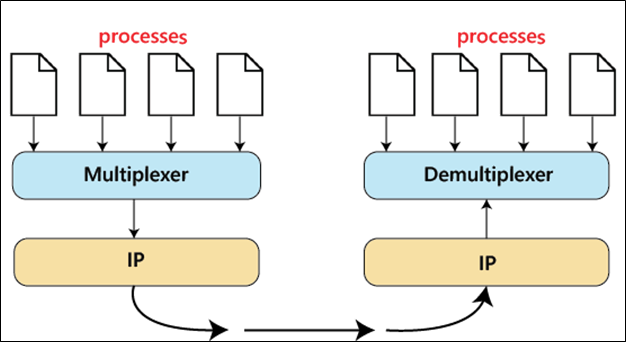


## Responsibilities of a Transport Layer

* The Process to Process Delivery
* End-to-End Connection between Hosts
* Multiplexing and Demultiplexing
* Congestion Control
* Data integrity and Error correction
* Flow control

### **1. The Process to Process Delivery**

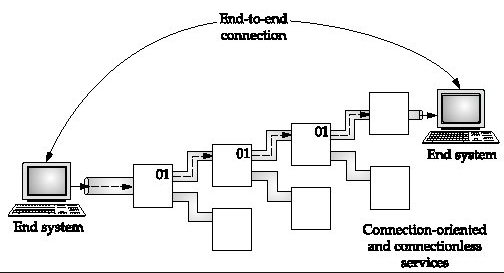
While Data Link Layer requires the MAC address (48 bits address contained inside the Network Interface Card of every host machine) of source-destination hosts to correctly deliver a frame and the Network layer requires the IP address for appropriate routing of packets, in a similar way Transport Layer 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.



Process to Process Delivery

### **2. End-to-end Connection between Hosts**

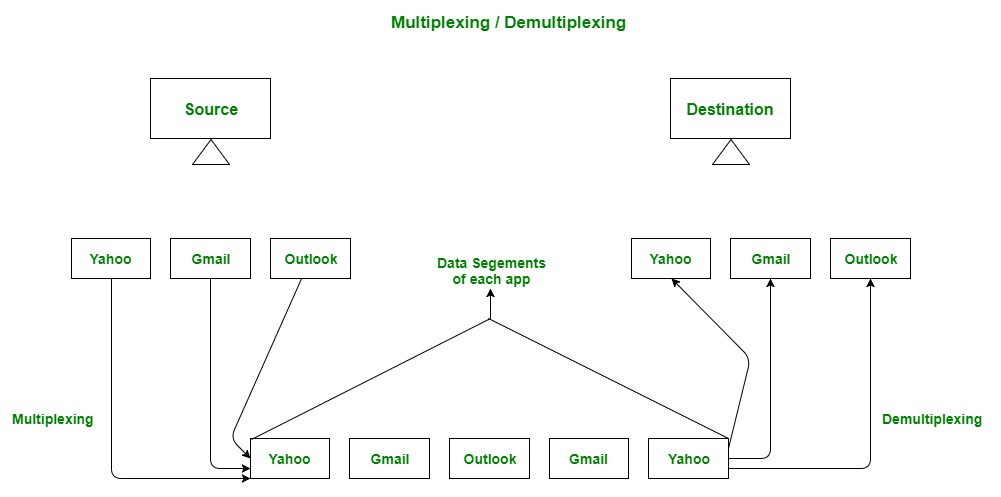
The transport layer is also responsible for creating the end-to-end Connection between hosts for which it mainly uses TCP and UDP. TCP is a secure, connection-orientated protocol that uses a handshake protocol to establish a robust connection between two end hosts. TCP ensures the reliable delivery of messages and is used in various applications. UDP, on the other hand, is a stateless and unreliable protocol that ensures best-effort delivery. It is suitable for applications that have little concern with flow or error control and requires sending the bulk of data like video conferencing. It is often used in multicasting protocols.



End to End Connection.

### **3.Multiplexing and Demultiplexing**

Multiplexing(many to one) is when data is acquired from several processes from the sender and merged into one packet along with headers and sent as a single packet. Multiplexing allows the simultaneous use of different processes over a network that is running on a host.  The processes are differentiated by their port numbers. Similarly, Demultiplexing(one to many) is required at the receiver side when the message is distributed into different processes. Transport receives the segments of data from the network layer distributes and delivers it to the appropriate process running on the receiver’s machine.



Multiplexing and Demultiplexing

### **4. Congestion Control**

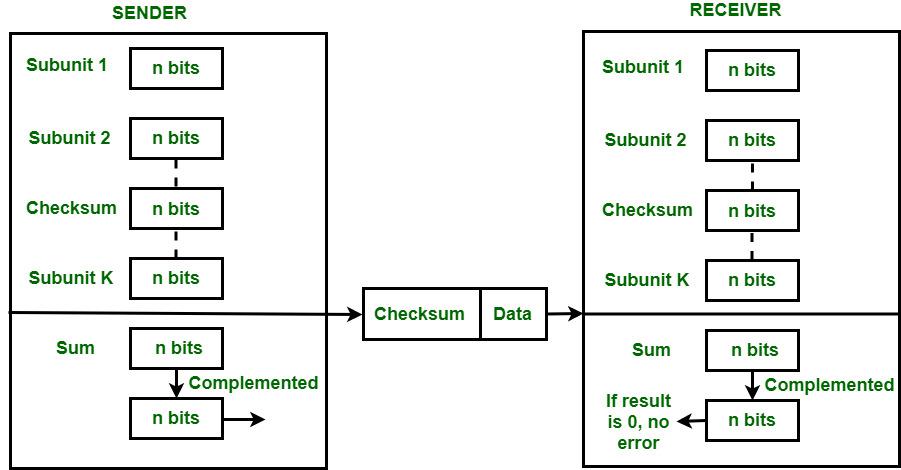
Congestion is a situation in which too many sources over a network attempt to send data and the router buffers start overflowing due to which loss of packets occurs. As a result, the retransmission of packets from the sources increases the congestion further. In this situation, the Transport layer provides [Congestion Control](https://www.geeksforgeeks.org/congestion-control-in-computer-networks/) in different ways. It uses open-loop congestion control to prevent congestion and closed-loop congestion control to remove the congestion in a network once it occurred. TCP provides AIMD – additive increases multiplicative decrease and [leaky bucket technique](https://www.geeksforgeeks.org/leaky-bucket-algorithm/) for congestion control.



Leaky Bucket Congestion Control Technique

### **5. Data integrity and Error Correction**

The transport layer checks for errors in the messages coming from the application layer by using error detection codes, and computing checksums, it checks whether the received data is not corrupted and uses the ACK and NACK services to inform the sender if the data has arrived or not and checks for the integrity of data.



Error Correction using Checksum

### **6. Flow Control**

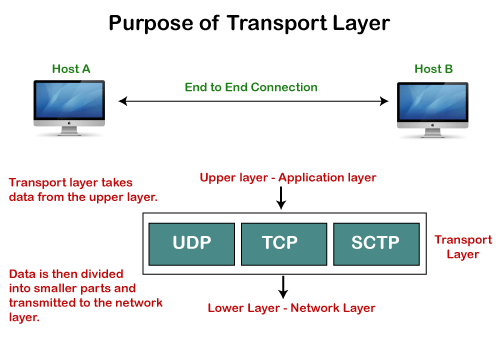
The transport layer provides a flow control mechanism between the adjacent layers of the TCP/IP model. TCP also prevents data loss due to a fast sender and slow receiver by imposing some flow control techniques. It uses the method of sliding window protocol which is accomplished by the receiver by sending a window back to the sender informing the size of data it can receive.

**TOPIC: TCP**

The transmission Control Protocol (TCP) is one of the most important protocols of Internet Protocols suite. It is most widely used protocol for data transmission in communication network such as internet.

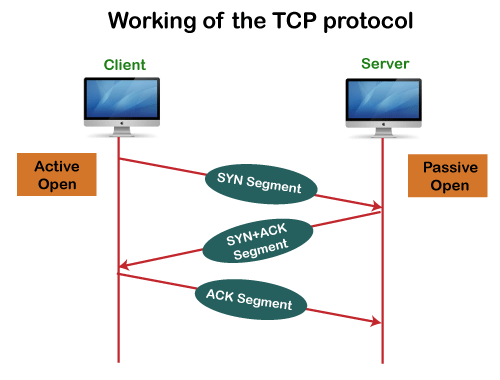
## Features

* TCP is reliable protocol. That is, the receiver always sends either positive or negative acknowledgement about the data packet to the sender, so that the sender always has bright clue about whether the data packet is reached the destination or it needs to resend it.
* TCP ensures that the data reaches intended destination in the same order it was sent.
* TCP is connection oriented. TCP requires that connection between two remote points be established before sending actual data.
* TCP provides error-checking and recovery mechanism.
* TCP provides end-to-end communication.
* TCP provides flow control and quality of service.
* TCP operates in Client/Server point-to-point mode.
* TCP provides full duplex server, i.e. it can perform roles of both receiver and sender.



**Working of TCP**

In TCP, the connection is established by using three-way handshaking. The client sends the segment with its sequence number. The server, in return, sends its segment with its own sequence number as well as the acknowledgement sequence, which is one more than the client sequence number. When the client receives the acknowledgment of its segment, then it sends the acknowledgment to the server. In this way, the connection is established between the client and the server.



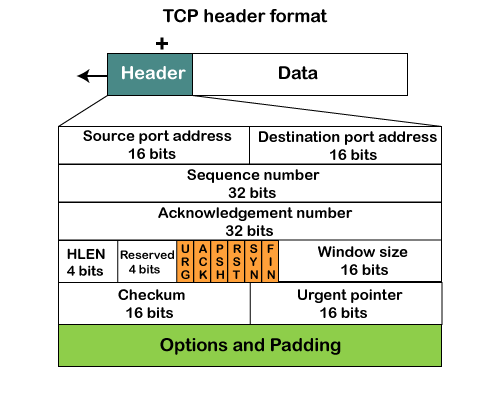
**Advantages of TCP**

* It provides a connection-oriented reliable service, which means that it guarantees the delivery of data packets. If the data packet is lost across the network, then the TCP will resend the lost packets.
* It provides a flow control mechanism using a sliding window protocol.
* It provides error detection by using checksum and error control by using Go Back or ARP protocol.
* It eliminates the congestion by using a network congestion avoidance algorithm that includes various schemes such as additive increase/multiplicative decrease (AIMD), slow start, and congestion window.

**Disadvantage of TCP**

It increases a large amount of overhead as each segment gets its own TCP header, so fragmentation by the router increases the overhead.

**TCP Header Format**



* **Source port:** It defines the port of the application, which is sending the data. So, this field contains the source port address, which is 16 bits.
* **Destination port:** It defines the port of the application on the receiving side. So, this field contains the destination port address, which is 16 bits.
* **Sequence number:** This field contains the sequence number of data bytes in a particular session.
* **Acknowledgment number:** When the ACK flag is set, then this contains the next sequence number of the data byte and works as an acknowledgment for the previous data received. For example, if the receiver receives the segment number 'x', then it responds 'x+1' as an acknowledgment number.
* **HLEN:** It specifies the length of the header indicated by the 4-byte words in the header. The size of the header lies between 20 and 60 bytes. Therefore, the value of this field would lie between 5 and 15.
* **Reserved:** It is a 4-bit field reserved for future use, and by default, all are set to zero.
* **Flags**  
  **There are six control bits or flags:**
  1. **URG:** It represents an urgent pointer. If it is set, then the data is processed urgently.
  2. **ACK:** If the ACK is set to 0, then it means that the data packet does not contain an acknowledgment.
  3. **PSH:** If this field is set, then it requests the receiving device to push the data to the receiving application without buffering it.
  4. **RST:** If it is set, then it requests to restart a connection.
  5. **SYN:** It is used to establish a connection between the hosts.
  6. **FIN:** It is used to release a connection, and no further data exchange will happen.
* **Window size**  
  It is a 16-bit field. It contains the size of data that the receiver can accept. This field is used for the flow control between the sender and receiver and also determines the amount of buffer allocated by the receiver for a segment. The value of this field is determined by the receiver.
* **Checksum**  
  It is a 16-bit field. This field is optional in UDP, but in the case of TCP/IP, this field is mandatory.
* **Urgent pointer**  
  It is a pointer that points to the urgent data byte if the URG flag is set to 1. It defines a value that will be added to the sequence number to get the sequence number of the last urgent byte.
* **Options**  
  It provides additional options. The optional field is represented in 32-bits. If this field contains the data less than 32-bit, then padding is required to obtain the remaining bits.

**TOPIC: UDP**

The User Datagram Protocol (UDP) is simplest Transport Layer communication protocol available of the TCP/IP protocol suite. It involves minimum amount of communication mechanism. UDP is said to be an unreliable transport protocol but it uses IP services which provides best effort delivery mechanism.

In UDP, the receiver does not generate an acknowledgement of packet received and in turn, the sender does not wait for any acknowledgement of packet sent. This shortcoming makes this protocol unreliable as well as easier on processing.

**Requirement of UDP**

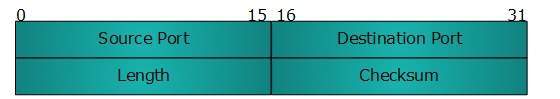
A question may arise, why do we need an unreliable protocol to transport the data? We deploy UDP where the acknowledgement packets share significant amount of bandwidth along with the actual data. For example, in case of video streaming, thousands of packets are forwarded towards its users. Acknowledging all the packets is troublesome and may contain huge amount of bandwidth wastage. The best delivery mechanism of underlying IP protocol ensures best efforts to deliver its packets, but even if some packets in video streaming get lost, the impact is not calamitous and can be ignored easily. Loss of few packets in video and voice traffic sometimes goes unnoticed.

**Features**

* UDP is used when acknowledgement of data does not hold any significance.
* UDP is good protocol for data flowing in one direction.
* UDP is simple and suitable for query based communications.
* UDP is not connection oriented.
* UDP does not provide congestion control mechanism.
* UDP does not guarantee ordered delivery of data.
* UDP is stateless.
* UDP is suitable protocol for streaming applications such as VoIP, multimedia streaming.

**UDP Header**

UDP header is as simple as its function.



UDP header contains four main parameters:

* **Source Port**  - This 16 bits information is used to identify the source port of the packet.
* **Destination Port**  - This 16 bits information, is used identify application level service on destination machine.
* **Length**  - Length field specifies the entire length of UDP packet (including header). It is 16-bits field and minimum value is 8-byte, i.e. the size of UDP header itself.
* **Checksum**  - This field stores the checksum value generated by the sender before sending. IPv4 has this field as optional so when checksum field does not contain any value it is made 0 and all its bits are set to zero.

**UDP application**

Here are few applications where UDP is used to transmit data:

* Domain Name Services
* Simple Network Management Protocol
* Trivial File Transfer Protocol
* Routing Information Protocol
* Kerberos

### Limitations

* It provides an unreliable connection delivery service. It does not provide any services of IP except that it provides process-to-process communication.
* The UDP message can be lost, delayed, duplicated, or can be out of order.
* It does not provide a reliable transport delivery service. It does not provide any acknowledgment or flow control mechanism. However, it does provide error control to some extent.

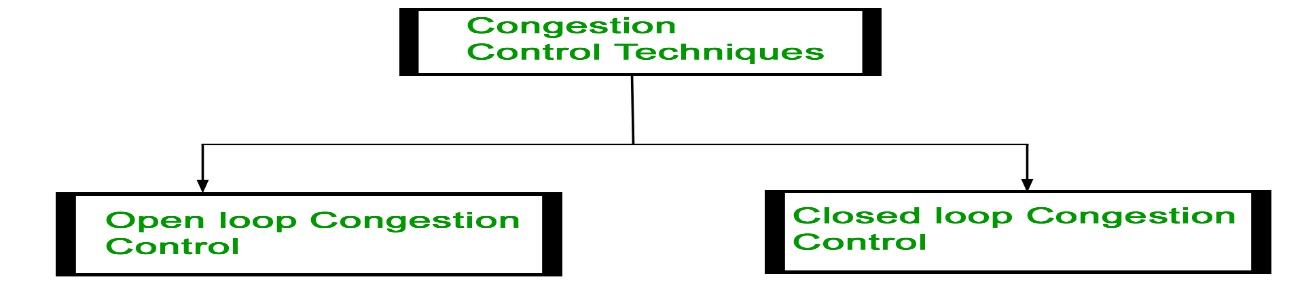
### Advantages

* It produces a minimal number of overheads.
* **Let's look at the differences between the TCP and UDP in a tabular form.**

|  |  |  |
| --- | --- | --- |
|  | **TCP** | **UDP** |
| **Full form** | It stands for **Transmission Control Protocol**. | It stands for **User Datagram Protocol**. |
| **Type of connection** | It is a connection-oriented protocol, which means that the connection needs to be established before the data is transmitted over the network. | It is a connectionless protocol, which means that it sends the data without checking whether the system is ready to receive or not. |
| **Reliable** | TCP is a reliable protocol as it provides assurance for the delivery of data packets. | UDP is an unreliable protocol as it does not take the guarantee for the delivery of packets. |
| **Speed** | TCP is slower than UDP as it performs error checking, flow control, and provides assurance for the delivery of | UDP is faster than TCP as it does not guarantee the delivery of data packets. |
| **Header size** | The size of TCP is 20 bytes. | The size of the UDP is 8 bytes. |
| **Acknowledgment** | TCP uses the three-way-handshake concept. In this concept, if the sender receives the ACK, then the sender will send the data. TCP also has the ability to resend the lost data. | UDP does not wait for any acknowledgment; it just sends the data. |
| **Flow control mechanism** | It follows the flow control mechanism in which too many packets cannot be sent to the receiver at the same time. | This protocol follows no such mechanism. |
| **Error checking** | TCP performs error checking by using a checksum. When the data is corrected, then the data is retransmitted to the receiver. | It does not perform any error checking, and also does not resend the lost data packets. |
| **Applications** | This protocol is mainly used where a secure and reliable communication process is required, like military services, web browsing, and e-mail. | This protocol is used where fast communication is required and does not care about the reliability like VoIP, game streaming, video and music streaming, etc. |

**Topic: Congestion Control**

Congestion control refers to the techniques used to control or prevent congestion. Congestion control techniques can be broadly classified into two categories: 



### **Open Loop Congestion Control**

Open loop congestion control policies are applied to prevent congestion before it happens. The congestion control is handled either by the source or the destination.

**Policies adopted by open loop congestion control –** 

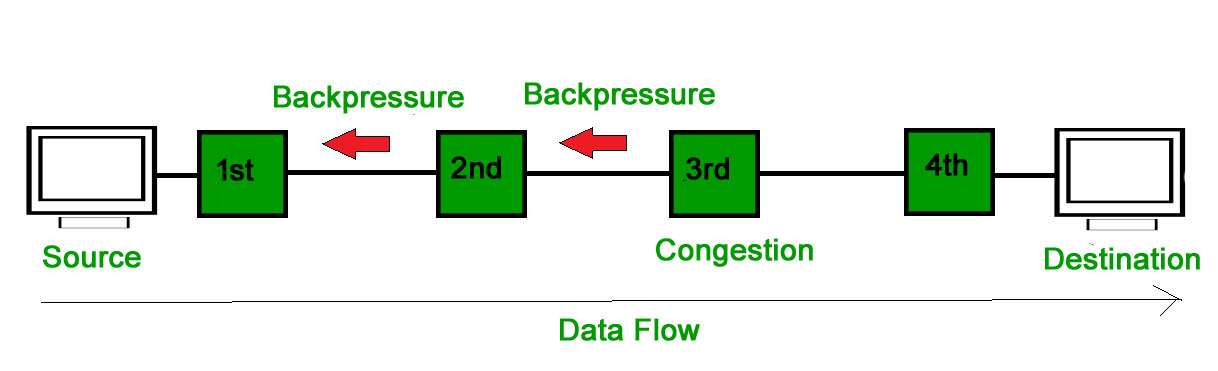
1. **Retransmission Policy :**   
   It is the policy in which retransmission of the packets are taken care of. If the sender feels that a sent packet is lost or corrupted, the packet needs to be retransmitted. This transmission may increase the congestion in the network.   
   To prevent congestion, retransmission timers must be designed to prevent congestion and also able to optimize efficiency.
2. **Window Policy :**   
   The type of window at the sender’s side may also affect the congestion. Several packets in the Go-back-n window are re-sent, although some packets may be received successfully at the receiver side. This duplication may increase the congestion in the network and make it worse.   
   Therefore, Selective repeat window should be adopted as it sends the specific packet that may have been lost.
3. **Discarding Policy :**   
   A good discarding policy adopted by the routers is that the routers may prevent congestion and at the same time partially discard the corrupted or less sensitive packages and also be able to maintain the quality of a message.   
   In case of audio file transmission, routers can discard less sensitive packets to prevent congestion and also maintain the quality of the audio file.
4. **Acknowledgment Policy :**   
   Since acknowledgements are also the part of the load in the network, the acknowledgment policy imposed by the receiver may also affect congestion. Several approaches can be used to prevent congestion related to acknowledgment.   
   The receiver should send acknowledgement for N packets rather than sending acknowledgement for a single packet. The receiver should send an acknowledgment only if it has to send a packet or a timer expires.
5. **Admission Policy :**   
   In admission policy a mechanism should be used to prevent congestion. Switches in a flow should first check the resource requirement of a network flow before transmitting it further. If there is a chance of a congestion or there is a congestion in the network, router should deny establishing a virtual network connection to prevent further congestion.

All the above policies are adopted to prevent congestion before it happens in the network. 

### **Closed Loop Congestion Control**

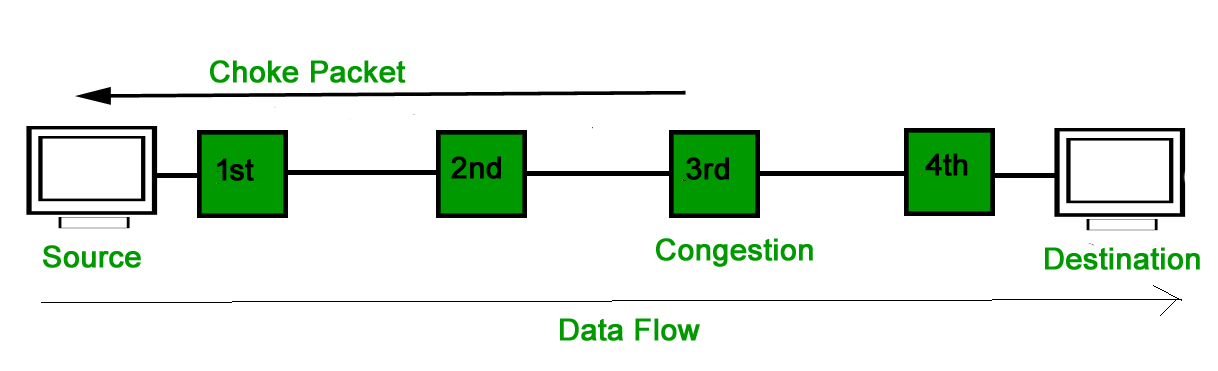
Closed loop congestion control techniques are used to treat or alleviate congestion after it happens. Several techniques are used by different protocols; some of them are: 

**1. Backpressure :**   
Backpressure is a technique in which a congested node stops receiving packets from upstream node. This may cause the upstream node or nodes to become congested and reject receiving data from above nodes. Backpressure is a node-to-node congestion control technique that propagate in the opposite direction of data flow. The backpressure technique can be applied only to virtual circuit where each node has information of its above upstream node.



          In above diagram the 3rd node is congested and stops receiving packets as a result 2nd node may be get congested due to slowing down of the output data flow. Similarly 1st node may get congested and inform the source to slow down. 

**2. Choke Packet Technique :**   
Choke packet technique is applicable to both virtual networks as well as datagram subnets. A choke packet is a packet sent by a node to the source to inform it of congestion. Each router monitors its resources and the utilization at each of its output lines. Whenever the resource utilization exceeds the threshold value which is set by the administrator, the router directly sends a choke packet to the source giving it a feedback to reduce the traffic. The intermediate nodes through which the packets has traveled are not warned about congestion. 



**3. Implicit Signaling :**   
In implicit signaling, there is no communication between the congested nodes and the source. The source guesses that there is congestion in a network. For example when sender sends several packets and there is no acknowledgment for a while, one assumption is that there is a congestion. 

**4. Explicit Signaling :**   
In explicit signaling, if a node experiences congestion it can explicitly sends a packet to the source or destination to inform about congestion. The difference between choke packet and explicit signaling is that the signal is included in the packets that carry data rather than creating a different packet as in case of choke packet technique.   
Explicit signaling can occur in either forward or backward direction.

* **Forward Signaling :** In forward signaling, a signal is sent in the direction of the congestion. The destination is warned about congestion. The receiver in this case adopt policies to prevent further congestion.
* **Backward Signaling :** In backward signaling, a signal is sent in the opposite direction of the congestion. The source is warned about congestion and it needs to slow down.

**Topic: congestion control techniques**

What is **congestion**?

A state occurring in network layer when the message traffic is so heavy that it slows down network response time.

**Effects** of Congestion

* As delay increases, performance decreases.
* If delay increases, retransmission occurs, making situation worse.

**Congestion control algorithms**

* Congestion Control is a mechanism that controls the entry of data packets into the network, enabling a better use of a shared network infrastructure and avoiding congestive collapse.
* Congestive-Avoidance Algorithms (CAA) are implemented at the TCP layer as the mechanism to avoid congestive collapse in a network.
* There are two congestion control algorithm which are as follows:
* **Leaky Bucket Algorithm**
* The leaky bucket algorithm discovers its use in the context of network traffic shaping or rate-limiting.
* A leaky bucket execution and a token bucket execution are predominantly used for traffic shaping algorithms.
* This algorithm is used to control the rate at which traffic is sent to the network and shape the burst traffic to a steady traffic stream.
* The disadvantages compared with the leaky-bucket algorithm are the inefficient use of available network resources.
* The large area of network resources such as bandwidth is not being used effectively.

Let us consider an example to understand

Imagine a bucket with a small hole in the bottom.No matter at what rate water enters the bucket, the outflow is at constant rate.When the bucket is full with water additional water entering spills over the sides and is lost.

[](https://media.geeksforgeeks.org/wp-content/uploads/leaky.jpg)

Similarly, each network interface contains a leaky bucket and the following **steps** are involved in leaky bucket algorithm:

1. When host wants to send packet, packet is thrown into the bucket.
2. The bucket leaks at a constant rate, meaning the network interface transmits packets at a constant rate.
3. Bursty traffic is converted to a uniform traffic by the leaky bucket.
4. In practice the bucket is a finite queue that outputs at a finite rate.

**Token bucket Algorithm**

* The leaky bucket algorithm has a rigid output design at an average rate independent of the bursty traffic.
* In some applications, when large bursts arrive, the output is allowed to speed up. This calls for a more flexible algorithm, preferably one that never loses information. Therefore, a token bucket algorithm finds its uses in network traffic shaping or rate-limiting.
* It is a control algorithm that indicates when traffic should be sent. This order comes based on the display of tokens in the bucket.
* The bucket contains tokens. Each of the tokens defines a packet of predetermined size. Tokens in the bucket are deleted for the ability to share a packet.
* When tokens are shown, a flow to transmit traffic appears in the display of tokens.
* No token means no flow sends its packets. Hence, a flow transfers traffic up to its peak burst rate in good tokens in the bucket.

**Need** of token bucket Algorithm:-

The leaky bucket algorithm enforces output pattern at the average rate, no matter how bursty the traffic is. So in order to deal with the bursty traffic we need a flexible algorithm so that the data is not lost. One such algorithm is token bucket algorithm.

**Steps** of this algorithm can be described as follows:

1. In regular intervals tokens are thrown into the bucket. ƒ
2. The bucket has a maximum capacity. ƒ
3. If there is a ready packet, a token is removed from the bucket, and the packet is sent.
4. If there is no token in the bucket, the packet cannot be sent.