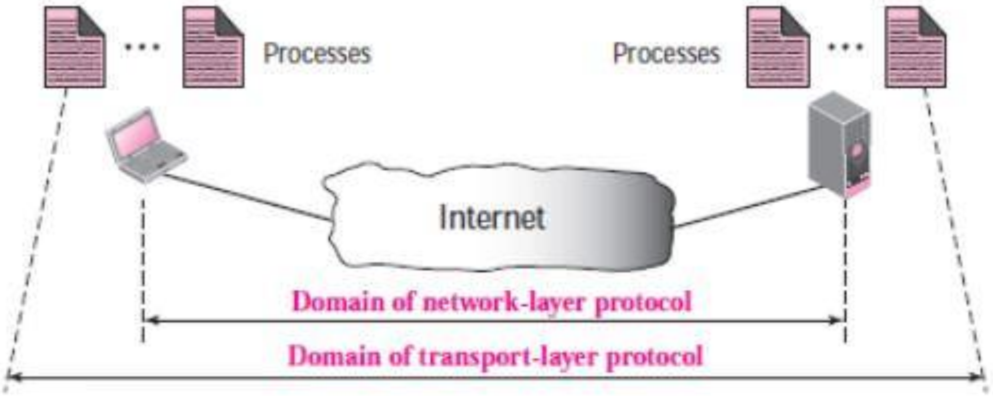
UNIT IV

TRANSPORT LAYER

Process to Process Communication, User Datagram Protocol (UDP), Transmission Control Protocol (TCP), SCTP Congestion Control; QoS Improving Techniques (Traffic Shaping, Admission Control And Resource Reservation).

4.1 Process to Process Communication

The transport layer is responsible for **process-to- process** delivery—the delivery of a packet, part of a message, from one **process** to another. A transport-layer protocol provides process-to-process communication. A process is an application-layer entity (running program) that uses the services of the transport layer.


Processes
Processes
Internet
Domain of network-layer protocol
Domain of transport-layer protocol

The network layer is responsible for communication at the computer level (host-to-host communication). A network layer protocol can deliver the message only to the destination computer.

A transport layer protocol is responsible for delivery of the message to the appropriate process. It is shown in figure.

The relationship between the communicating processes is the client-server relationship.

Process to Process Delivery:

The data link layer is responsible for delivery of frames between two neighboring nodes over a link. This is called **node-to-node** delivery. The network layer is responsible for delivery of datagrams between two hosts. This is called **host-to-host** delivery. Real communication takes place between two processes (application programs). We need **process-to-process** delivery. The transport layer is responsible for process-to-process delivery-the delivery of a packet, part of a message, from one process to another. Figure 4.1 shows these three types of deliveries and their domains

4.2 Transmission Control Protocol

When you connect to the Internet, you establish a connection between a router and a computer or mobile device in a few simple steps, whether you’re using wired or wireless technology. Nothing else is required because the system automatically logs in to the network and obtains the unique Internet address that you need to receive and send data. This is made possible by a set of protocols known as the **Internet protocol suite**. One of the oldest and most important protocols in the suite is the Transmission Control Protocol (TCP). It determines how network devices exchange data.

TCP allows for **transmission of information in both directions**. This means that computer systems that communicate over TCP can send and receive data at the same time, similar to a telephone conversation. The protocol uses segments (packets) as the basic units of data transmission. In addition to the payload, segments can also contain control information and are limited to 1,500 bytes. The **TCP software** in the network protocol stack of the operating system is responsible for establishing and terminating the end-to-end connections as well as transferring data.

The TCP software is controlled by the various network applications, such as web browsers or servers, via **specific interfaces**. Each connection must always be identified by two clearly defined endpoints (client and server). It doesn’t matter which side assumes the client role and which assumes the server role. All that matters is that the TCP software is provided with a unique, **ordered pair** consisting of IP address and port (also referred to as "2-tuple" or "socket") for each endpoint.

The three-way handshake: How a TCP connection is established in detail

Prerequisites for establishing a valid TCP connection: Both endpoints must already have a **unique IP address** (IPv4 or IPv6) and have assigned and enabled the **desired port** for data transfer. The IP address serves as an identifier, whereas the port allows the operating system to assign connections to the specific client and server applications.

Source port (16 bits): Identifies the port number of the sender.

Destination port (16 bits): Identifies the port number of receiver.

Sequence number (32 bits): The sequence number specifies the first byte of attached payload data or is sent when the connection is established or terminated. It is also used for validating and sorting the segments after transmission.

Acknowledgment number (32 bits): This field contains the next sequence number that the sender is expecting. An ACK flag (in the “Flags” field) is a precondition for validity.

Offset (4 bits): The “Offset” field specifies the length of the TCP header in 32-bit words to highlight the starting point of the payload data. This starting point varies from segment to segment due to the variable “Options” field.

Reserved (6 bits): Reserved for future use according to RFC 793 and not yet in use. This field must always be set to 0.

Flags (6 bits): The six possible single bits in the “Flags” field enable various TCP actions for organizing communication and data processing. The following flags are either set or not set for these actions:

- **URG**: The "Urgent" flag signals to the TCP application that the payload data must be processed immediately up to the set Urgent pointer (see above).
- **ACK**: In combination with the acknowledgment number, the ACK flag acknowledges the receipt of TCP packets. If the flag is not set, the confirmation number is also invalid.
- **PSH**: The "Push" flag ensures that a TCP segment is immediately pushed through without first being sent to the buffer of the sender and receiver.

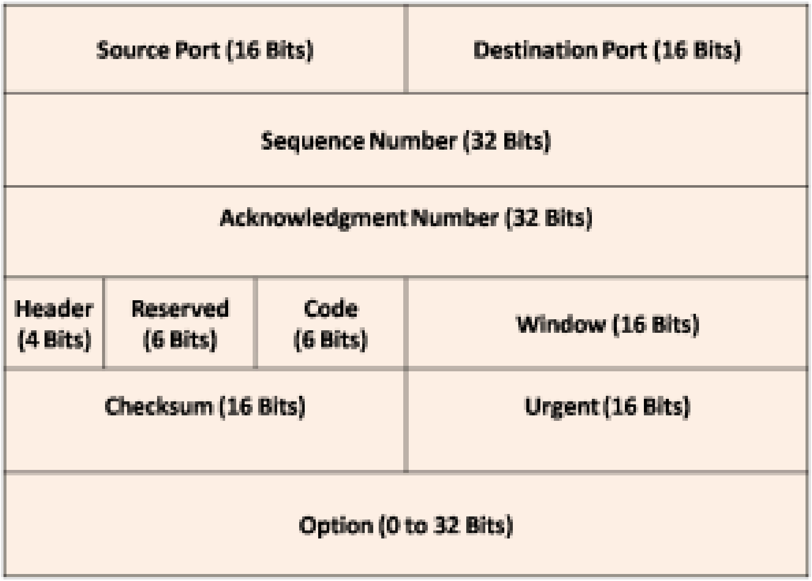
- **RST**: If there is an error during transmission, a TCP packet with the RST flag set can be used to reset the connection.
- **SYN**: Messages that have SYN flag set represent the first step of the three-way handshake, meaning they initiate the connection.
- **FIN**: The "Finish" flag signals to the other party that a sender is ending the transmission.

Window size (16 bits): This field specifies the number of bytes that the sender is willing to receive.

Checksum (16 bits): The Transmission Control Protocol can reliably detect transmission errors. The checksum calculated from the header, the payload data and the pseudo-header is used for this purpose.

Urgent pointer (16 bits): The urgent pointer indicates the position of the first byte after the payload data that is to be processed urgently. As a result, this field is only valid and relevant if the URG flag is set.

Options (0 - 32bits): Use the Options field if you want to include TCP functions that don’t belong in the general header, for example if you want to define the maximum segment size. The length of the options must always be a multiple of 32, otherwise zero-bit padding is required.



<table>
<tr><td colspan="3">Source Port (16 Bits)</td><td>Destination Port (16 Bits)</td></tr>
<tr><td colspan="4">Sequence Number (32 Bits)</td></tr>
<tr><td colspan="4">Acknowledgment Number (32 Bits)</td></tr>
<tr><td>Header
(4 Bits)</td><td>Reserved
(6 Bits)</td><td>Code
(6 Bits)</td><td>Window (16 Bits)</td></tr>
<tr><td colspan="3">Checksum (16 Bits)</td><td>Urgent (16 Bits)</td></tr>
<tr><td colspan="4">Option (0 to 32 Bits)</td></tr>
</table>

The Transmission Control Protocol (TCP) is a transport protocol that is used on

top of IP to ensure reliable transmission of packets. TCP includes mechanisms to solve many of the problems that arise from packet-based messaging, such as lost packets, out of order packets, duplicate packets, and corrupted packets.

Summary of key facts about the Transmission Control Protocol

The TCP protocol has shaped the history and development of computer networks for nearly a half a century. TCP can be easily combined with Internet protocol (IP), which also has a long history, and it has many advantages over other alternatives such as UDP and SCTP. The most important features can be summarized as follows:

- TCP is **connection-oriented** and enables two-way communication between two endpoints after the **three-way handshake**.
- TCP is **reliable** because the protocol ensures that all data is fully transmitted and can be assembled by the receiver in the correct order.
- TCP allows data to be sent in individual segments of up to **1,500 bytes** (including headers) in size.
- TCP is positioned at the **transport layer** (layer 4) of the OSI model.
- TCP is usually used in conjunction with the **Internet Protocol** (IP) and is commonly known as the TCP/IP protocol stack.
- The **TCP header** has a default size of 20 bytes. Up to 40 bytes of additional options can be added.

4.3 User Datagram Protocol

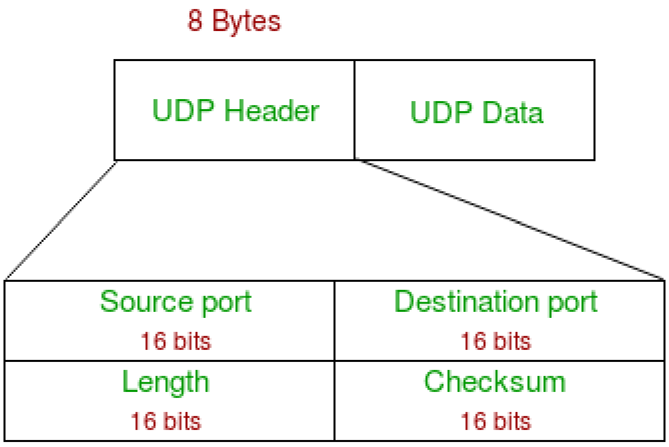
UDP (User Datagram Protocol) is a communications protocol that is primarily used for establishing low-latency and loss-tolerating connections between applications on the internet. It speeds up transmissions by enabling the transfer of data before an agreement is provided by the receiving party. As a result, UDP is beneficial in time-sensitive communications, including voice over Internet Protocol (VoIP), domain name system (DNS) lookup, and video or audio playback. UDP is an alternative to Transmission Control Protocol (TCP).

Both UDP and TCP run on top of IP and are sometimes referred to as UDP/IP or TCP/IP; however, there are important differences between the two. For example, UDP enables process-to-process communication, while TCP supports host-to-host communication. Furthermore, TCP sends individual packets and is considered a reliable transport medium. On the other hand, UDP sends messages, called *datagrams*, and is considered a best-effort mode of communications meaning the service does not provide any guarantees that the data will be delivered or offer special features to retransmit lost or corrupted messages.

UDP provides two services not provided by the IP layer. It provides port numbers to help distinguish different user requests and, optionally, a checksum capability to verify that the data arrived intact.

TCP has emerged as the dominant protocol used for the bulk of internet connectivity due to its ability to break large data sets into individual packets, check for and resend lost packets, and reassemble packets in the correct sequence. But these additional services come at a cost in terms of additional data overhead and latency.

In contrast, UDP is considered a connectionless protocol because it doesn't require a virtual circuit to be established before any data transfer occurs. The communication protocol just sends the packets, which means that it has much lower bandwidth overhead and latency. With UDP, packets may take different paths between sender and receiver, and as a result, some packets may be lost or received out of order.



1. **Source Port:** Source Port is 2 Byte long field used to identify port number of sources.
2. **Destination Port:** It is 2 Byte long field, used to identify the port of destined packet.
3. **Length:** Length is the length of UDP including header and the data. It is 16-bits field.
4. **Checksum:** Checksum is 2 Bytes long field. It is the 16-bit one's complement of the one's complement sum of the UDP header, pseudo header of information from the IP header and the data, padded with zero octets at the end (if necessary) to make a multiple of two octets.

- The transmission of Real-time packets, mainly in multimedia applications

4.4 SCTP Congestion Control

SCTP stands for Stream Control Transmission Protocol**.**
It is a connection- oriented protocol in computer networks which provides a full-duplex association i.e., transmitting multiple streams of data between two end points at the same time that have established a connection in network. It is sometimes referred to as next generation TCP**, SCTP makes it easier to support telephonic conversation on Interne**t. A telephonic conversation requires transmitting of voice along with other data at the same time on both ends, SCTP protocol makes it easier to establish reliable connection.

Stream Control Transmission Protocol (**SCTP**) is a transport-layer protocol that ensures reliable, in-sequence transport of data. **SCTP** provides multihoming support where one or both endpoints of a connection can consist of more than one IP address. This enables transparent failover between redundant **network** paths.

Like TCP, SCTP manages "reliable transport" (ensuring the complete arrival of data units that are sent over the network) over the Internet's basically connectionless Internet Protocol (IP), the protocol responsible for moving the data but not for managing whether all the data arrives.

Congestion control tries avoiding overload situations in network components like routers. Congestion in network components can lead to packet loss which is handled by the error control function of SCTP . The goal of congestion control is to **avoid packet loss in** the first place.

4.5 QoS Improving Techniques

QoS or Quality of Service in networking is the process of managing network resources to reduce packet loss as well as lower network jitter and latency. **QoS** is usually applied on networks that cater to traffic that carry resource-intensive data like: Video-on-demand.

Quality-of-Service (QoS) refers to traffic control mechanisms that seek to either differentiate performance based on application or network-operator requirements or provide predictable or guaranteed performance to applications, sessions or traffic aggregates. Basic phenomenon for QoS means in terms of packet delay and losses of various kinds.

Need for QoS –

- Video and audio-conferencing leads to bounded delay and loss rate.
- Video and audio streaming require bounded packet loss rate, it may not be so sensitive to delay.
- Time-critical applications (real-time control) in which bounded delay is considered to be an important factor.
- Valuable applications should be provided better services than less valuable applications.

When the host and IP phone transmit data and voice packets destined for the host and IP phone on the other side, it is likely that we get congestion on the serial link. The router will queue packets that are waiting to be transmitted but the queue is not unlimited. What should the router do when the queue is full? drop the data packets? the voice packets? When you drop voice packets, the user on the other side will complain about poor voice quality. When you drop data packets, a user might complain that transfer speeds are poor.

QoS is about using tools to change how the router or switch deals with different packets. For example, we can configure the router so that voice traffic is prioritized before data traffic.

Characteristics of network traffic

There are four characteristics of network traffic that we must deal with:

- **Bandwidth**
- **Delay**
- **Jitter**
- **Loss**

Bandwidth is the speed of the link, in bits per second (bps). With QoS, we can tell the router how to use this bandwidth. With FIFO, packets are served on a first come first served basis. One of the things we can do with QoS is create different queues and put certain traffic types in different queues. We can then configure the router so that queue one gets 50% of the bandwidth, queue two gets 20% of the bandwidth and queue three gets the remaining 30% of the bandwidth.

Delay is the time it takes for a packet to get from the source to a destination, this is called the **one-way delay**. The time it takes to get from a source to the destination and back is called the **round-trip delay**. There are different types of delay; without going into too much detail, let me give you a quick overview:

- Processing delay: this is the time it takes for a device to perform all tasks required to forward the packet. For example, a router must do a lookup in the routing table, check its ARP table, outgoing access-lists and more. Depending on the router model, CPU, and switching method this affects the processing delay.
- Queuing delay: the amount of time a packet is waiting in a queue. When an interface is congested, the packet will have to wait in the queue before it is transmitted.
- Serialization delay: the time it takes to send all bits of a frame to the physical interface for transmission.
- Propagation delay: the time it takes for bits to cross a physical medium. For example, the time it takes for bits to travel through a 10-mile fiber optic link is much lower than the time it takes for bits to travel using satellite links.

Some of these delays, like the propagation delay, is something we can’t change. What we can do with QoS however, is influence the queuing delay. For example, you could create a priority queue that is always served before other queues. You could add voice packets to the priority queue so they don’t have to wait long in the queue, reducing the queuing delay.

Jitter is the variation of one-way delay in a stream of packets. For example, let’s say an IP phone sends a steady stream of voice packets. Because of congestion in the network, some packets are delayed. The delay between packet 1 and 2 is 20 ms, the delay between packet 2 and 3 is 40 ms, the delay between packet 3 and 4 is 5 ms, etc. The receiver of these voice packets must deal with jitter, making sure the packets have a steady delay or you will experience poor voice quality.

Loss is the amount of lost data, usually shown as a percentage of lost packets sent. If you send 100 packets and only 95 make it to the destination, you have 5% packet loss. Packet loss is always possible. For example, when there is congestion, packets will be queued but once the queue is full…packets will be dropped. With QoS, we can at least decide which packets get dropped when this happens.

4.5.1 **Traffic Shaping**

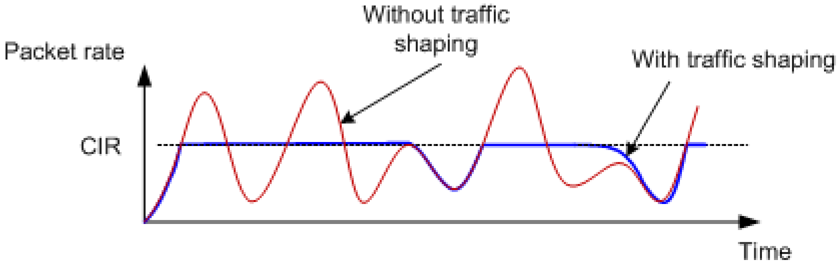
Traffic shaping, also known as **packet shaping**, is a congestion management method that regulates network data transfer by delaying the flow of less important or less desired packets.

Traffic shaping is used to control bandwidth of the network to ensure quality of service to business-critical applications.

This technique uses three parameters to shape the flow of network traffic :

1. Burst size
2. Average bandwidth
3. Peak bandwidth

These are explained as following below.



4.5.2 Admission control

It is a validation process in communication systems where a check is performed before a connection is established to see **if current resources are sufficient for the proposed connection**

4.5.3 Resource Reservation

The **Resource Reservation Protocol** (**RSVP**) is a transport layer protocol designed to reserve resources across a network using the integrated services model. RSVP operates over an IPv4 or IPv6 and provides receiver-initiated setup of resource reservations for multicast or unicast data flows.

RSVP Messages

There are two types of RSVP messages −

- **Path Messages (path):** A path message is sent by the sender to all receivers by multicasting storing the path state at each node in its path. It stores the necessary information so that the receivers can make the reservation.
- **Reservation messages (resv):** The resv message is sent by the receiver to the sender along the reverse path of the path message. It identifies the resources that is requires by the data flow.