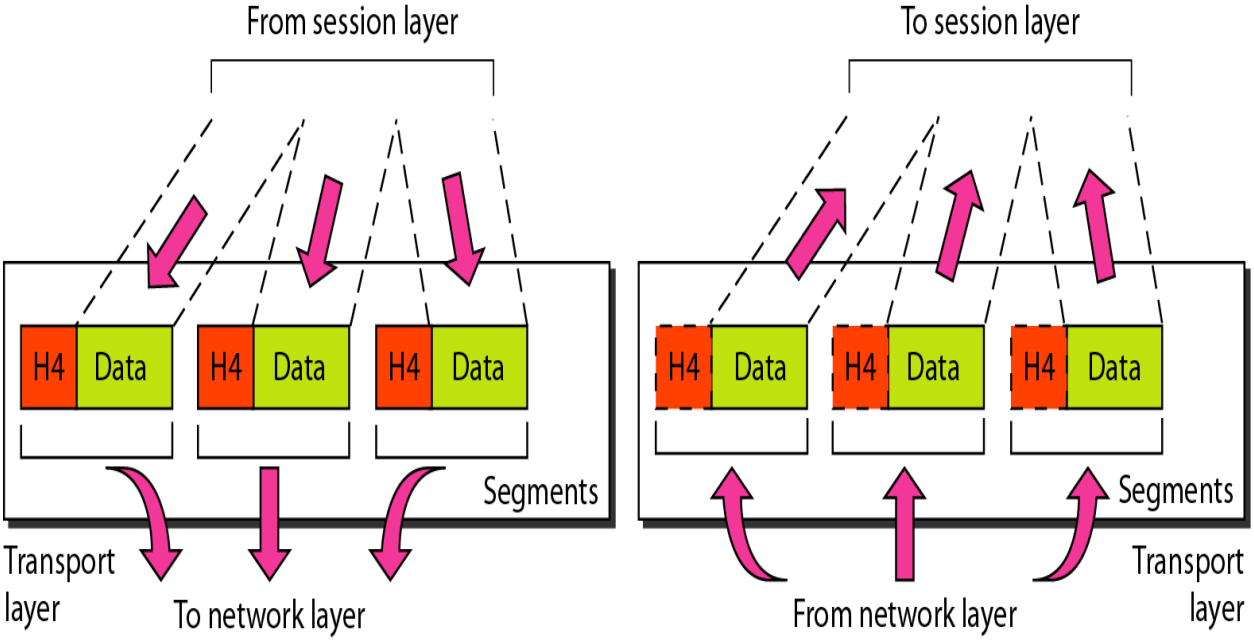
Chapter Three

Transport Layer

Next Layer in OSI Model is recognized as Transport Layer (Layer-4). All modules and procedures pertaining to transportation of data or data stream are categorized into this layer. As all other layers, this layer communicates with its peer Transport layer of the remote host.

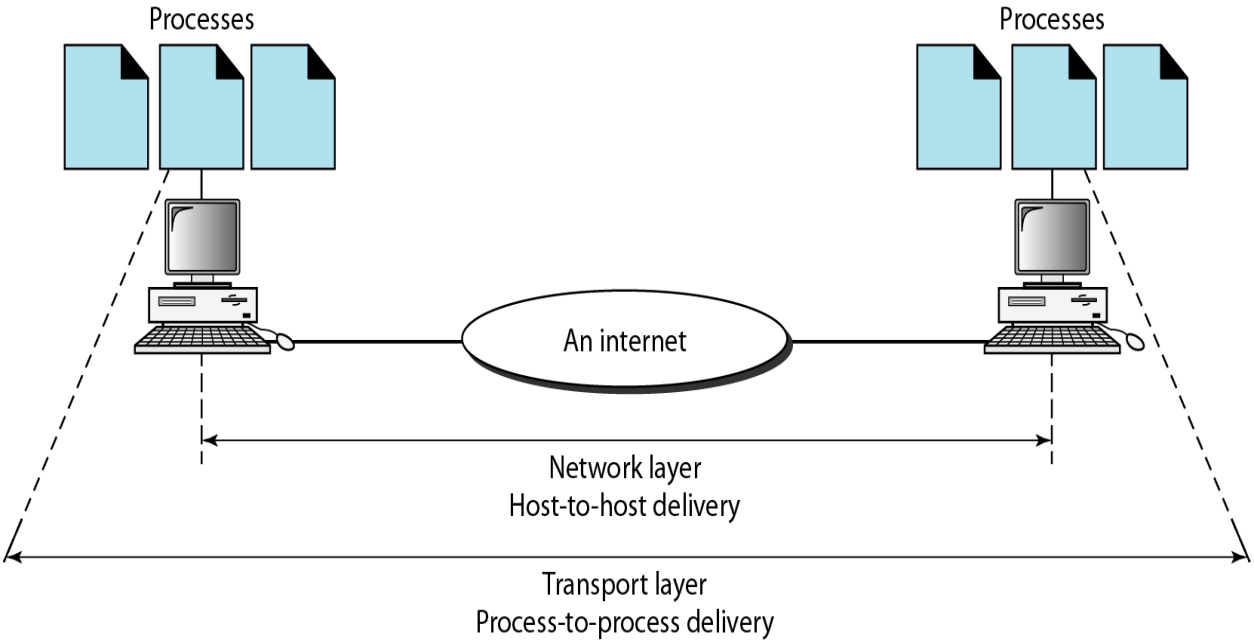
Transport layer offers peer-to-peer and end-to-end connection between two processes on remote hosts. Transport layer takes data from upper layer (i.e. Application layer) and then breaks it into smaller size segments, numbers each byte, and hands over to lower layer (Network Layer) for delivery.



- The transport layer is responsible for process-to-process delivery of the entire message.
- A process is an application program running on a host.
- Ensures that the whole message arrives intact and in order, overseeing both error control and flow control at the source-to-destination level.
- The transport layer is responsible for the delivery of a message from one process to another.
- The transport layer header must therefore include a type of address called a *service-point address* in the OSI model and port number or port addresses in the Internet and TCP/IP protocol suite.
- Transport layer offers peer-to-peer and end-to-end connection between two

processes on remote hosts.

- This Layer breaks the information data, supplied by Application layer in to smaller units called segments. It numbers each byte, and hands over to lower layer (Network Layer) for delivery.
- All modules and procedures pertaining to transportation of data or data stream are categorized into this layer.
- All server processes intend to communicate over the network are equipped with well-known Transport Service Access Points (TSAPs) also known as port numbers.



Functions

- This Layer is the first one which breaks the information data, supplied by Application layer in to smaller units called segments. It numbers every byte in the segment and maintains their accounting.
- This layer ensures that data must be received in the same sequence in which it was sent.
- This layer provides end-to-end delivery of data between hosts which may or may not belong to the same subnet.
- All server processes intend to communicate over the network are equipped with well-known Transport Service Access Points (TSAPs) also known as port numbers.

The two main Transport layer protocols are:

- **Transmission Control Protocol**
  It provides reliable communication between two hosts.

- **User Datagram Protocol**
  It provides unreliable communication between two hosts.

Transmission Control Protocol

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

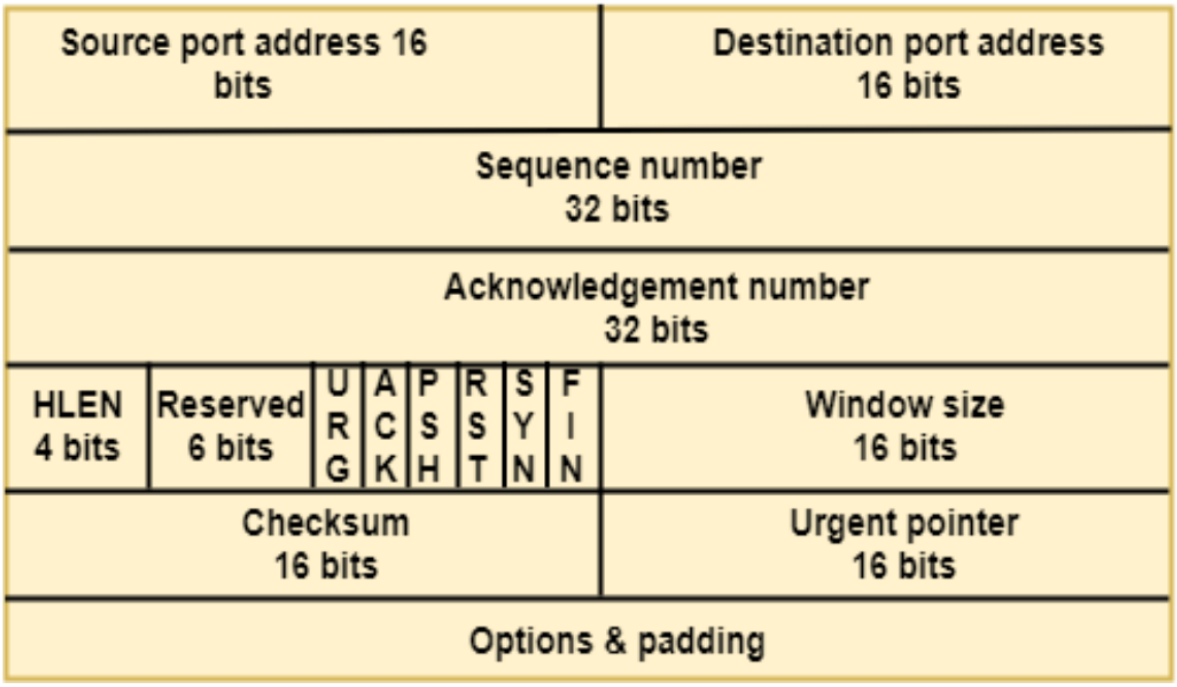
- TCP is reliable protocol. That is, the receiver always sends either positive or negative acknowledgement about the data packet to the sender, so that the sender always has bright clue about whether the data packet is reached the destination or it needs to resend it.
- TCP ensures that the data reaches intended destination in the same order it was sent.
- Stream data transfer**:** TCP protocol transfers the data in the form of contiguous stream of bytes
- 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.
- Logical Connections**:** The combination of sockets, sequence numbers, and window sizes, is called a logical connection.
- 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.

Suppose the process A wants to send and receive the data from process B. The following steps occur:

- ✓ Establish a connection between two TCPs.
- ✓ Data is exchanged in both the directions.
- ✓ The Connection is terminated

Header

The length of TCP header is minimum 20 bytes long and maximum 60 bytes.



- **Source Port (16-bits)** - It identifies source port of the application process on the sending device.
- **Destination Port (16-bits)** - It identifies destination port of the application process on the receiving device.
- **Sequence Number (32-bits)** - Sequence number of data bytes of a segment in a session.
- **Acknowledgement Number (32-bits)** - When ACK flag is set, this number contains the next sequence number of the data byte expected and works as acknowledgement of the previous data received.
- **Data Offset (4-bits)** - This field implies both, the size of TCP header (32-bit words) and the offset of data in current packet in the whole TCP segment.
- **Reserved (3-bits)** - Reserved for future use and all are set zero by default.
- **Windows Size** - This field is used for flow control between two stations and indicates the amount of buffer (in bytes) the receiver has allocated for a segment, i.e. how much data is the receiver expecting.
- **Checksum** - This field contains the checksum of Header, Data and Pseudo Headers.
- **Urgent Pointer** - It points to the urgent data byte if URG flag is set to 1.
- **Options** - It facilitates additional options which are not covered by the regular header. Option field is always described in 32-bit words. If this field contains data less than 32-bit, padding is used to cover the remaining bits to reach 32-bit boundary.

Addressing

TCP communication between two remote hosts is done by means of port numbers (TSAPs). Ports numbers can range from 0 – 65535 which are divided as:

- System Ports (0 – 1023)
- User Ports (1024 – 49151)
- Private/Dynamic Ports (49152 – 65535)

Whenever we need to deliver something to one specific destination among many, we need an address. At the transport layer, we need a transport layer address, called a port number, to choose among multiple processes running on the destination host. The destination port number is needed for delivery; the source port number is needed for the reply.

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

- The client program defines itself with a port number, chosen randomly by the transport layer software running on the client host. This is the ephemeral port number
- The server process must also define itself with a port number. This port number, however, cannot be chosen randomly.
- If the computer at the server site runs a server process and assigns a random number as the port number, the process at the client site that wants to access that server and use its services will not know the port number.
- one solution would be to send a special packet and request the port number of a specific server, but this requires more overhead.
- The Internet has decided to use universal port numbers for servers; these are called well-known port numbers
- It should be clear by now that the IP addresses and port numbers play different roles in selecting the final destination of data. The destination IP address defines the host among the different hosts in the world. After the host has been selected, the port number defines one of the processes on this particular host.

IANA Ranges

The IANA (Internet Assigned Number Authority) has divided the port numbers into three ranges: well known, registered, and dynamic (or private),

Well-known ports- The ports ranging from 0 to 1023 are assigned and controlled by IANA.

Registered ports- The ports ranging from 1024 to 49,151 are not assigned or controlled by IANA. They can only be registered with IANA to preventduplication.

Dynamic ports-The ports ranging from 49,152 to 65,535 are neither controlled nor registered. They can be used by any process. These are the ephemeral ports

Socket Addresses

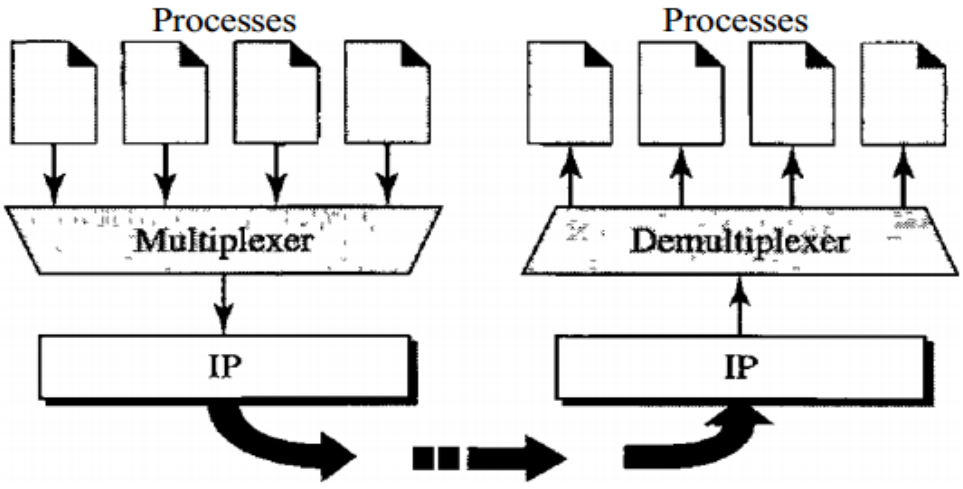
- Process-to-process delivery needs two identifiers, IP address and the port number, at each end to make a connection. The combination of an IP address and a port number is called a socket address.
- The client socket address defines the client process uniquely just as the server socket address defines the server process uniquely .
- A transport layer protocol needs a pair of socket addresses: the client socket address and the server socket address.
- These four pieces of information are part of the IP header and the transport layer protocol header.

Multiplexing and Demultiplixing

The technique to combine two or more data streams in one session is called Multiplexing. When a TCP client initializes a connection with Server, it always refers to a well-defined port number which indicates the application process. The client itself uses a randomly generated port number from private port number pools.

Using TCP Multiplexing, a client can communicate with a number of different application process in a single session. For example, a client requests a web page which in turn contains different types of data (HTTP, SMTP, FTP etc.) the TCP session timeout is increased and the session is kept open for longer time so that the three-way handshake overhead can be avoided.

This enables the client system to receive multiple connection over single virtual connection. These virtual connections are not good for Servers if the timeout is too long.



Multiplexing

- At the sender site, there may be several processes that need to send packets. However, there is only one transport layer protocol at any time. This is a many-to-one relationship and requires multiplexing.
- The protocol accepts messages from different processes, differentiated by their assigned port numbers. After adding the header, the transport layer passes the packet to the network layer.

DE multiplexing

At the receiver site, the relationship is one-to-many and requires demultiplexing. The transport layer receives datagrams from the network layer. After error checking and dropping of the header, the transport layer delivers each message to the appropriate process based on the port number.

Connectionless Versus Connection-Oriented Service

- A transport layer protocol can either be connectionless or connection-oriented.

Connectionless Service

- In a connectionless service, the packets are sent from one party to another with no need for connection establishment or connection release.
- The packets are not numbered; they may be delayed or lost or may arrive out of sequence.
- There is no acknowledgment either.

Exam: UDP is connectionless.

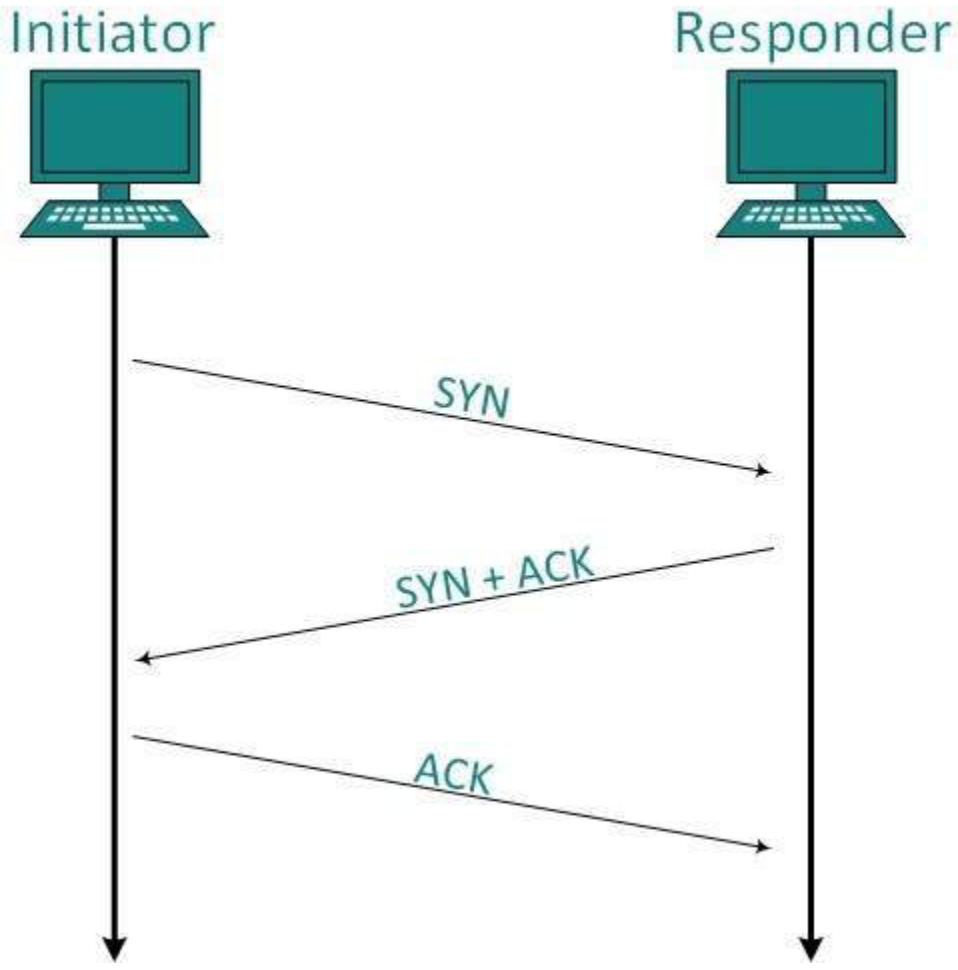
Connection-Oriented *Service*

- In a connection-oriented service, a connection is first established between the sender and the receiver. Data are transferred. At the end, the connection is released .
- It creates a relationship between the segments using sequence numbers.

Exam: TCP, SCTP

Connection Management

TCP communication works in Server/Client model. The client initiates the connection and the server either accepts or rejects it. Three-way handshaking is used for connection management.



Establishment

Client initiates the connection and sends the segment with a Sequence number. Server acknowledges it back with its own Sequence number and ACK of client's segment which is one more than client's Sequence number. Client after receiving ACK of its segment sends an acknowledgement of Server's response.

Release

Either of server and client can send TCP segment with FIN flag set to 1. When the receiving end responds it back by ACKnowledging FIN, that direction of TCP communication is closed and connection is released.

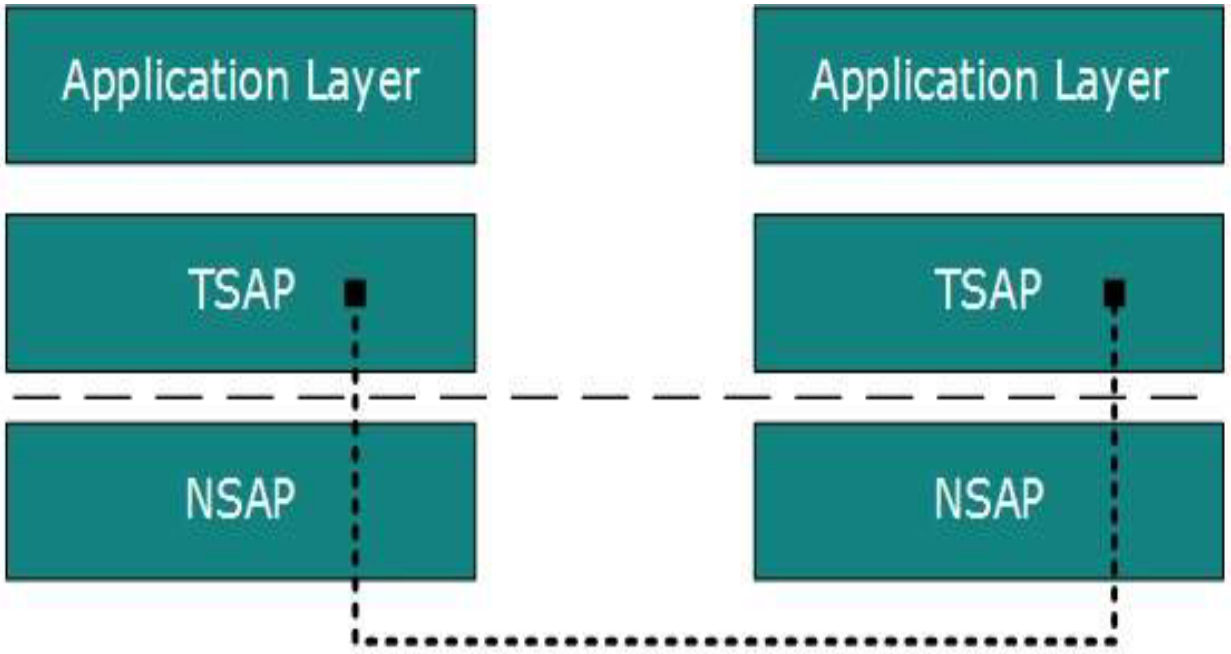
Error Control and Flow Control

TCP uses port numbers to know what application process it needs to handover the data segment. Along with that, it uses sequence numbers to synchronize itself with the remote host. All data segments are sent and received with sequence numbers. The Sender knows which last data segment was received by the Receiver when it gets ACK. The Receiver knows about the last segment sent by the Sender by referring to the sequence number of recently received packet.

If the sequence number of a segment recently received does not match with the sequence number the receiver was expecting, then it is discarded and NACK is sent back. If two segments arrive with the same sequence number, the TCP timestamp value is compared to make a decision.

End-to-End Communication

- A process on one host identifies its peer host on remote network by means of TSAPs, also known as Port numbers.
- TSAPs are very well defined and a process which is trying to communicate with its peer knows this in advance.



- For example, when a DHCP client wants to communicate with remote DHCP server, it always requests on port number 67. When a DNS client wants to communicate with remote DNS server, it always requests on port number 53.

Congestion Control

When large amount of data is fed to system which is not capable of handling it, congestion occurs. TCP controls congestion by means of Window mechanism. TCP sets a window size telling the other end how much data segment to send. TCP may use three algorithms for congestion control:

- Additive increase, Multiplicative Decrease
- Slow Start
- Timeout React

User Data Protocol

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.

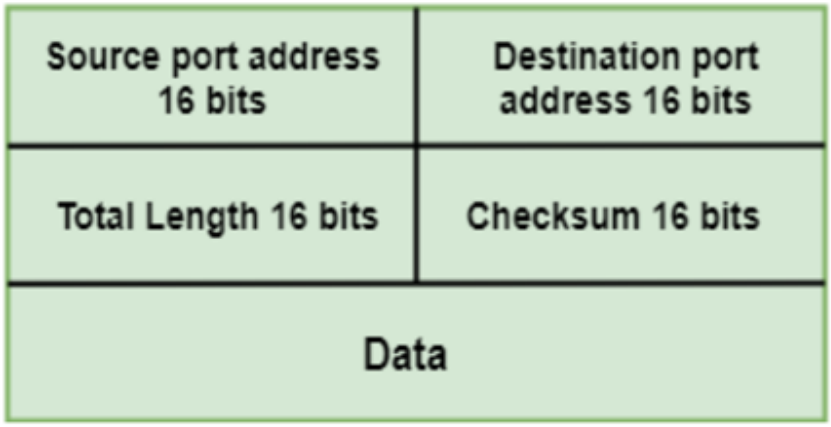
UDP Header

UDP header is as simple as its function.

UDP header contains four main parameters:

User Datagram Format

- The user datagram has a 16-byte header which is shown below:



- **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.

Use of UDP

The following lists some uses of the UDP protocol:

- ✓ UDP is suitable for a process that requires simple request-response communication with little concern for flow and error control.
- ✓ UDP is suitable for a process with internal flow and error control mechanisms. For example, the Trivial File Transfer Protocol (TFTP) process includes flow and error control.
- ✓ UDP is a suitable transport protocol for multicasting. Multicasting capability is embedded in the UDP software but not in the TCP software.
- ✓ UDP is used for management processes such as SNMP.
- ✓ UDP is used for some route updating protocols such as Routing Information Protocol(RIP).

Disadvantages of UDP protocol

- ✓ It does not provide any sequencing or reordering functions and does not specify the damaged packet when reporting an error.
- ✓ UDP can discover that an error has occurred, but it does not specify which packet

has been lost as it does not contain an ID or sequencing number of a particular data segment.

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

Difference between TCP and UDP

TCP	UPD
TCP establishes a virtual circuit before transmitting the data.	UDP transmits the data directly to the destination computer without verifying whether the receiver is ready to receive or not.
It is a Connection Oriented protocol	It is a Connectionless protocol
slow	high
It is a reliable protocol.	It is an unreliable protocol.
20 bytes	8 bytes
It waits for the acknowledgement of data and has the ability to resend the lost packets.	It neither takes the acknowledgement, nor it retransmits the damaged frame.