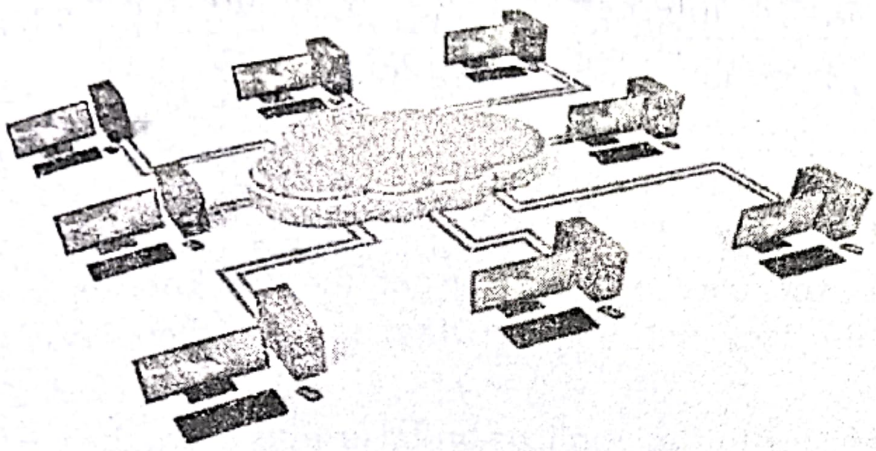


7

TRANSPORT LAYER

INTRODUCTION

- The transport layer provides an end to end data transfer service that shields upper layer protocols from the details of the intervening network or networks.
- A transport protocol can be either connection oriented, such as TCP, or connectionless, such as UDP.

(A) If the underlying network or inter-network service is unreliable, such as with the use of IP, then the connection oriented transport protocols becomes complex.

- The basic cause of this complexity is the need to deal with the relatively large and delays experienced between end systems.
- These large variable delays complicate the flow control and error control techniques.

(B) TCP uses credit – based flow and error control technique that is somewhat different from the sliding window flow control found in X.25 and HDLC.

- In essence, TCP separates acknowledgements from the management of size of sliding windows.

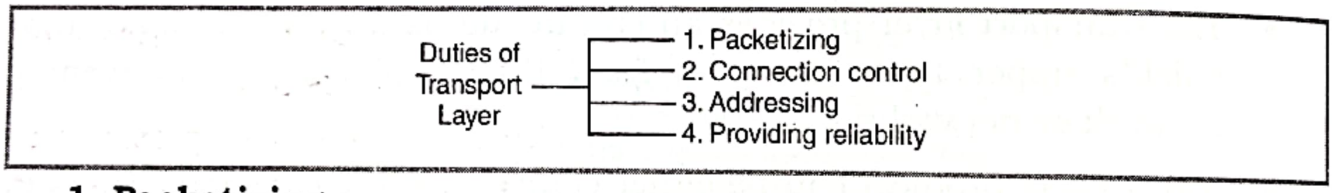
(C) The TCP credit- based mechanism was designed for end –to– end flow control, it is also used to assist in inter-network congestion control.

- When a TCP entity detects the presence of congestion in the internet, it reduces the flow of data in the internet until it detects an easing in congestion.

- The transport protocol provides services to transport service(TS) uses, FTP, SMTP, and TELNET.
- Transport layer is the core of the internet modes. The application layer programs interact with each other using the services of transport layer.
- It provides the services to the application layer and takes the services from network layer.

7.1 DUTIES OF TRANSPORT LAYER

- Transport layer is meant for the process to process delivery and it is achieved by performing a number of functions.
- The different functions of a transport layer



1. Packetizing :

- Transport layer creates packets out of the message received from the application layer.
- Packatizing the process of dividing a long message into smaller one.
- The packets are encapsulated into the data field of transport layer packet and header are added.
- Length of the message which is to be divided can vary from several lines to several pages.
- Header is added to each packet to allow the transport layer to perform its other function.

2. Connection Control :

- Transport layer protocols are divided into two categories :
  1. Connection oriented
  2. Connectionless

(i) Connection oriented delivery :

- This protocol establishes a connection *i.e.* a virtual path between sender and receiver.

- This is a virtual connection. The packet may travel out of order. The packets are numbered conecutively and communication is bi-directional.

(ii) Connectionless delivery :

- There is no fixed connection between sender and receiver.
- It treats each packet independently.
- Each packet can take its own different route.

3. Addressing :

- The client needs the address of the remote computers it wants to communicate with.
- Such a remote computer have unique address so that it can be distinguished from all the other computers.

4. Providing Reliability :

For the higher reliability the flow control should be incorporated.

- **Flow control :**

1. We know that flow control is handled by the data link layer in OSI reference model. In the transport layer the flow control is performed end to end rather than one to one across the single link.
2. **Error Control :** Error control at the transport layer can provide the error control as well. But error control at transport layer is performed end to end rather than across a single link. It is generally achieved through retransmission.

- **Congestion control and Quality of Service (QOS)**
- The congestion can take place at any of the layer *i.e.* data link, network or transport layer. But the effect of congestion is generally evident in the transport layer.
- QOS (Quality of Service) can be implemented in other layer but its actual effect in the transport layer.
- The transport layer enhances the QOS provided to the network layer.

Difference between Data Link Layer and Transport Layer

Data Link Layer	Transport Layer
1. Communication is through physical channel.	1. Communication is through subnet.
2. It is not necessary to specify the destination router.	2. Addressing of destination is essential.
3. Establishing a connection is simple.	3. Initial connection establishment is difficult or more complex.

4. No storage capacity.	4. There is some storage capacity in the subnet.
5. No additional delay.	5. Delay is introduced due to storing capacity of subnet.

7.2 DESIGN ISSUES

- Two basic types of transport services are possible : Connection oriented and connectionless or datagram service.
- A connection – oriented service provides for the establishment, maintenance, and termination of logical connection between Transport Service users.
- The **connection- oriented** service generally implies that the service is reliable.
- This section looks at the transport protocol mechanisms needed to support the connection oriented service.

In Connectionless Services The connections are not establishied in advance.

- All the time they adopt the different route to reach at the distination.
- A network service that makes life easy for transport protocol by guaranting the delivery of all transport data units in order and defining the required mechanisms.

7.2.1 RELIABLE SEQUENCING NETWORK SERVICE

- The assumption of a reliable sequencing networking services allows the use of a quite simple transport protocol. Three issues need to be addressed :
  - (a) Addressing
  - (b) Multiplexing
  - (c) Flow control.

(a) Addressing:

- The issue concerned with addressing is simple : A user of a given transport entity wishes either to establish a connection with or make a data transfer to a user of some other transport entity.
- The target user needs to be specified by all of the following:
  - User identification

- Transport entity identification
- Host address Network Number

- The user address is specified as (Host, Port).
- The Port variable represents particular TS users at the specified host.
- In TCP, the combination of Port and Host is referred to as socket.
- Because routing is not a concern with transport layer, it simply passes the Host portion of the address down to the network service.
- Port is included in a Header, to be used at the destination by the destination transport protocol.
- With respect to the interface between the transport protocol and higher- level protocols, the transport protocol performs a multiplexing/ de-multiplexing function.
- Multiple users employ the same transport protocols and are distinguished by port numbers or service access points.

(b) Multiplexing

- The multiplexing is the multiplexing of multiple connections on a single lower- level connection, and downward multiplexing as the splitting of a single connection among multiple lower –level connections.
- The downward multiplexing or splitting might be used to improve throughput.
- A larger sequence space might be needed for high speed, high- delay networks.
- Throughput can only be increased so far. If there is a single host-node link over which all virtual circuits are multiplexed, the throughput of transport connection cannot exceed the data rate of that link.

(c) Flow Control :

- Flow control is a relatively simple mechanism at the link layer; it is rather a complex mechanism at the transport layer for two main reasons:
  - The transmission delay between transport entities is generally long as compared to actual transmission time. This means that there is a considerable delay in the communication of flow control information.

- Because the transport layer operates over a network or internet, the amount of the transmission delay may be highly variable. This makes it difficult to effectively use a timeout mechanism for retransmission of lost data.

7.3 ELEMENTS OF TRANSPORT FROTOCOLS

1. Addressing
2. Establishing a connection
3. Releasing a connection
4. Flow control and buffering
5. Multiplexing
6. Crash Recovery.

1. Addressing

- If a remote process wants to be connected to another remote process, then it has to be specified the address of one to which to be connected.
- This is the address to which process can listen for connection request.
- In the internet these end points are IP address local port. Note that end points and addresses carry the same meaning.
- In the transport layer communication, let us use a general term TSAP (Transport Service Access Point) for the same.

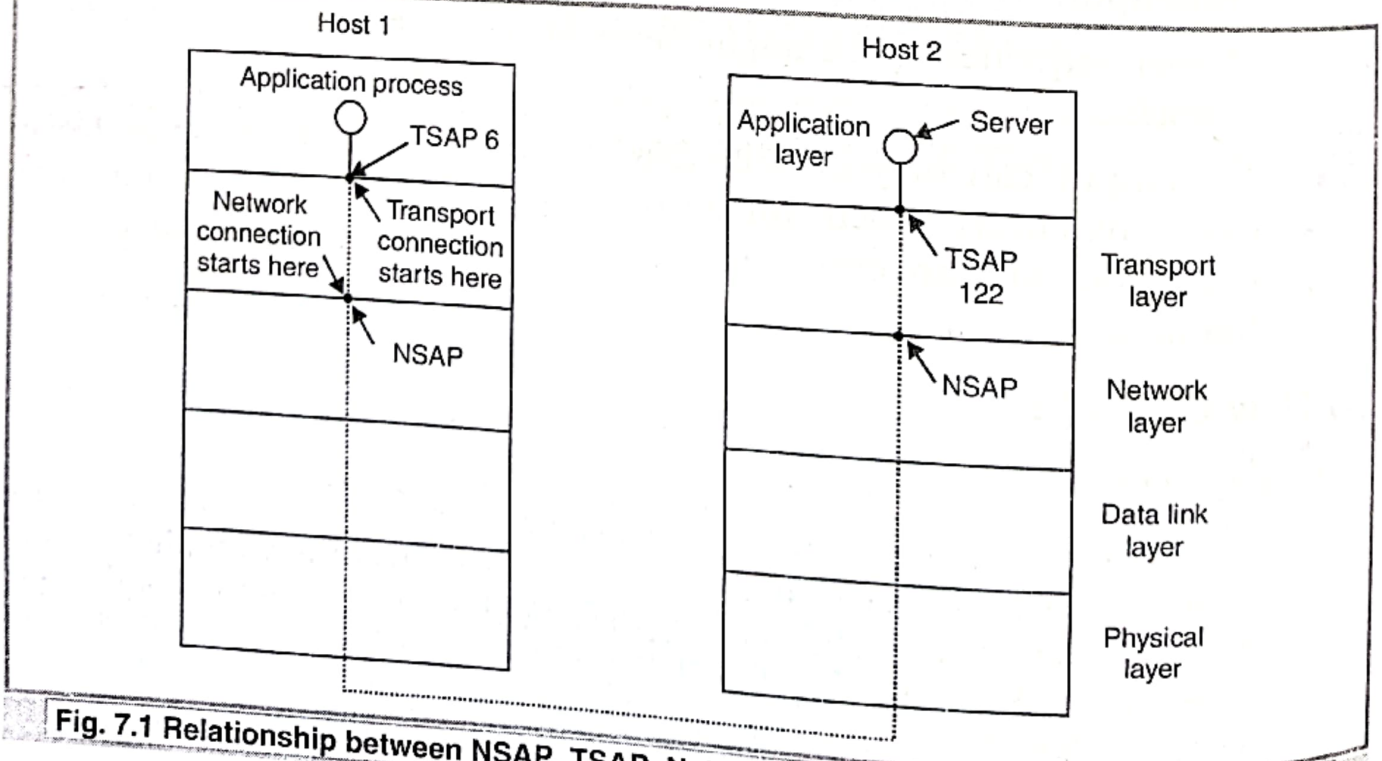


Fig. 7.1 Relationship between NSAP, TSAP, Network connection and transport connection

2. Establishing a Connection

- Refer Fig. 7.2 to understand the connection establishment :
- Following steps are involved in establishing a connection.

1. Host A sends a connection request packet to host B. This includes the initialization information about traffic from A to B.
2. Host B sends the packet of acknowledgement to confirm that it has received the request from A.
3. Host B sends a connection request to A along with the initialization information about traffic from B to A.
4. Host A sends a packet of acknowledgement to confirm the request of B.

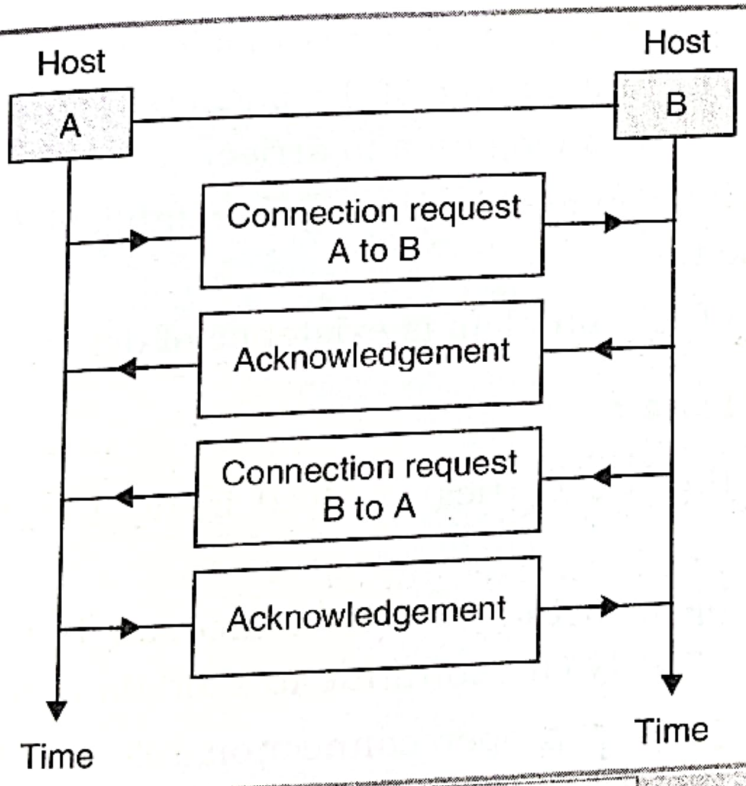


Fig. 7.2 Establishing a connection

- It is a possible to merge the steps 2 and 3.
- Note that each connection request must have a sequence number which is helpful in recovering from the loss or duplication of the packets.
- For the same reason, each acknowledgement also should have an acknowledgement number.
- The first sequence number in each direction should be random for each connection established. This is to ensure that a sender cannot create more than one connection which starts with the same sequence number (e.g. 2).
- This is to recognize the duplication for packets.

- Since a sequence number is required for each connection, the receiver has to keep the history of sequence numbers for each remote host for a specific time.

Problems :

- Establishing a connection seems to be very easy. But actually it is a very tricky job. The problem occurs when the network can lose, store and duplicate packets.
- The problems can be elaborated as follows :
  1. Due to congestion on a subnet, the acknowledgements do not get back in time. So retransmission of each packet takes place.
  2. If the subnet uses datagrams inside and every packet travels on a different route. Some of the packets might get stuck in a traffic jam and take a long time to arrive.
  3. The same connection getting re-established due to duplication of packet.
- So the crux of the problem is existence of delayed duplicate packets.

3. Connection Release :

- Any one of the two parties involved in data exchange can close the connection.
- But the problem is that, when connection is terminated from one end, the other party can continue to send data in the other direction.
- Hence, to ensure a proper connection release one has to follow the steps given below.

Procedure to release a connection :

Refer Fig. 7.3 to understand this procedure.

1. Host 1 sends a connection release request to host 2.
2. Host 2 sends an acknowledgement to confirm the request of host 1.
3. After this the connection is closed in one direction but host 2 can continue to send data to host 1.
4. When host 2 finishes sending his data, it sends a connection release request to host 1.
5. Host 1 acknowledges (confirms) the request made by host 2 and the connection is released from both ends.

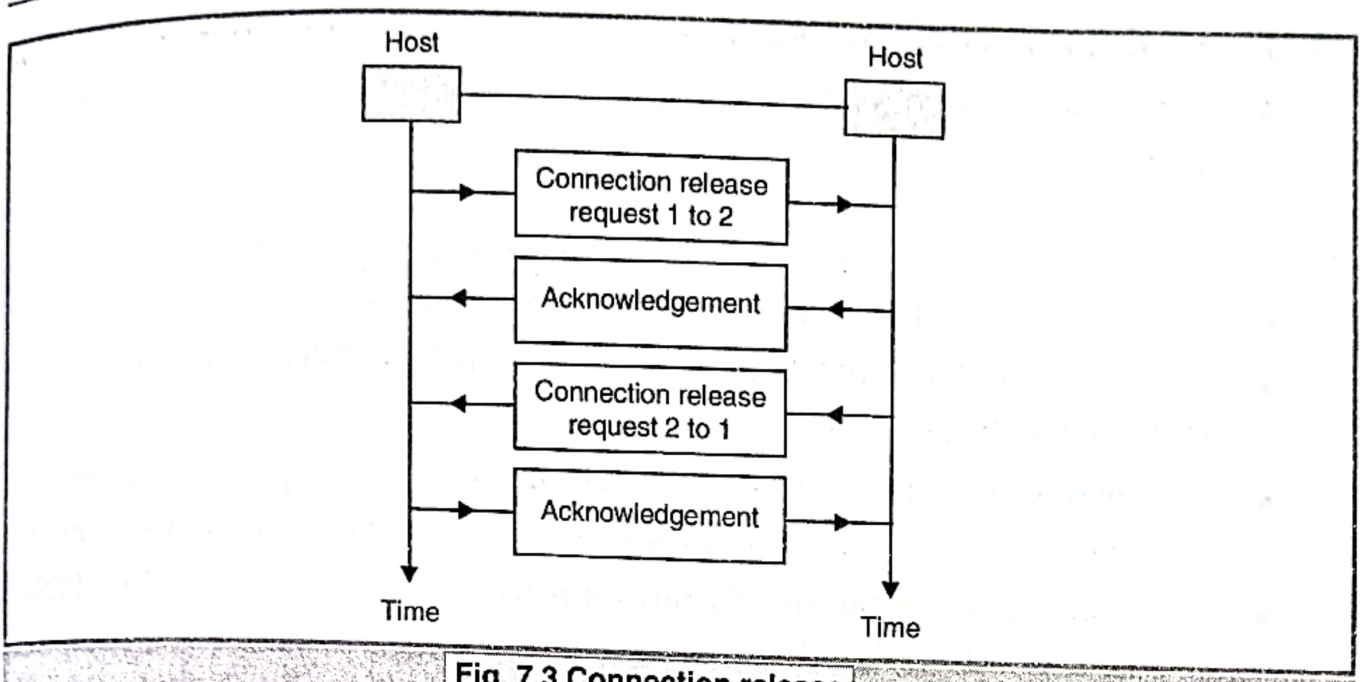


Fig. 7.3 Connection release

Releasing a connection is easier than establishing it. There are two styles of releasing a connection :

1. Asymmetrtic release and 2. Symmetric release.

1. Asymmetric Release :

In asymmetric release, when one party hangs up the connection is broken. It is an abrupt release and may result in loss of data.

2. Symmetric release :

- Symmetric release treats the connection as two separate unidirectional connections and requires each one to be released separately.
- Symmetric release is as shown in Fig. 7.3 and does not involve any loss of data.

Two Army Problem :

- Assume that a white army is encamped in a valley as shown in Fig. 7.3 and on both the surrounding hill sides are the blue armies.

Protocol 2 : Three way handshake

1. Now let us improve the two way handshake protocol by making it a three way handshake.
2. This protocol is shown in Fig. 7.3 Assuming no messages lost, blue army 2 will also get the acknowledgement.
3. But now the chief of blue army 1 will hesitate, because he does not know if the last message he has sent has got through or not.
4. So we can make a four way handshake. But it also does not help, because in every protocol, the uncertainty after last handshake message always remains.
5. In fact we can prove that, no protocol exists that works.

4. Flow Control and Buffering :

- For flow control, a sliding window is needed on each connection to keep a fast transmitter from overrunning a slow receiver (the same as the data link layer). Since a host may have numerous connections, it is impractical to implement the same data link buffering strategy (using dedicated buffers for each line).
- The sender should always buffer outgoing TPDUs until they are acknowledged.
- The receiver may not dedicate specific buffers to specific connections. Instead, a single buffer pool may be maintained for all connections.
- When a TPDU come in, if there is a free buffer available, the TPDU is accepted, otherwise it is discarded. However, for high–bandwidth traffic (e.g.: file transfers), it is better if the receiver dedicate a full window of buffers, to allow the data to flow at maximum speed.
- How large the buffer size should be ?
- As the traffic pattern changes, the transport protocol should allow a sending host to request buffer space at the other end. Alternatively, the receiver could tell the sender "I have reserved buffers for you".
- A general way to manage dynamic buffer allocation is to decouple the buffering from the acknowledgements.

Dynamic buffer management is a variable-sized window :

- Initially, the sender requests a certain number of buffers based on its perceived needs.
- The receiver then grants as many buffers as it can afford.
- Every time the sender transmits a TPDU, it must decrement its allocation, stopping altogether when the allocation reaches zero.
- The receiver then **separately** piggybacks both acknowledgements and buffer allocations onto reverse traffic.
- To prevent deadlock caused by loss of control TPDUs, each host should periodically send control TPDUs giving the acknowledgement and buffer status on each connection.
- The sender's window size could be dynamically adjusted not only by the availability of the buffers at the receiver side, but also by the capacity and traffic of the subnet.
- The bigger the subnet's carrying capacity and lighter the traffic, larger is the sender's window size.

5. Multiplexing and Demultiplexing

- The addressing mechanism allows multiplexing and demultiplexing by the transport layer as shown in Fig. 7.4.

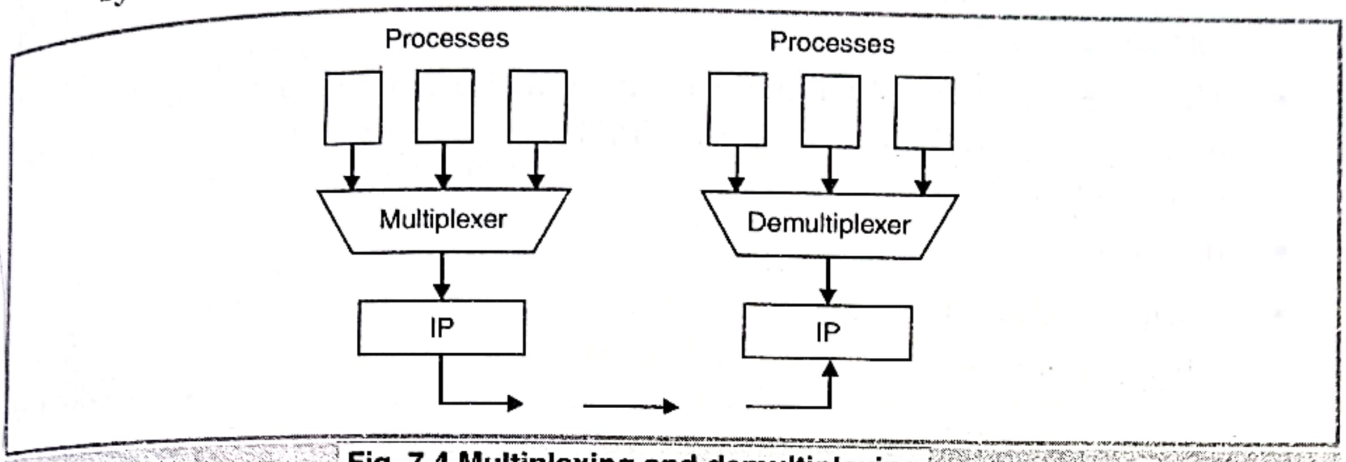


Fig. 7.4 Multiplexing and demultiplexing

Multiplexing :

- At the sending end, there are several processes that want to send packets. But there is only one transport layer protocol (UDP or TCP).
- This is a many-to-one relationship and hence requires multiplexing.
- The protocol accepts massages from different processes. These messages are separated from each other by their port numbers.
- Then the transport layer adds header and passes the packet to the network layer as shown in Fig. 7.4.

Demultiplexing :

- At the receiving end, the relationship is one as to many. So we need a demultiplexer.
- First the transport layer receives datagrams from the network layer.
- The transport layer then checks for errors and drops the header to obtain the messages and delivers them to appropriate process based on the port number.

6. Crash Recovery :

- The host and routes are subject to crashes, and the recovery from such crashes is essential. Such crashes will result in loss of data packets.
- If the transport entity is completely inside the hosts then the recovery from the network and router crashes is simple. This is as explained below.
- In case of a **router crash**, the two transport entities must exchange information after the crash to determine which TPDUs were received

and which were not. The crash can be recovered by retransmitting the lost ones.

- If the network layer provides connection oriented service, then the loss of a virtual circuit (crash can be handled by establishing a new virtual circuit.
- Then the remote transport entity can be asked about which TPDUs it has received and which TPDUs are lost. Those lost can be retransmitted over the newly established virtual circuit.
- It is very difficult to recover from the host crash.
- Suppose that the sender is sending a long file to the receiver using a simple stop-and-wait protocol. Part way through the transmission the receiver crashes. When the receiver comes back up, it might send a broadcast TPDU to all other hosts, requesting the other hosts to inform it of the status of all.

7.4 CONNECTION MANAGEMENT ADDRESSING

- In a transport layer end to end delivery can be done in either of two modes : connection oriented and connectionless.
- The connection –oriented modes is the more commonly used.
- A connection –oriented protocol setup a virtual circuit or pathway between the sender and receiver.
- All of the packet belonging to a message are then sent over the same path.
- Using a single pathway for the entire message facilities the acknowledgement process and retransmission of damaged or lost frames. Connection –oriented services, therefore are generally reliable.

Connection –oriented transmission has three stages :

1. Connection Establishment
2. Data Transfer
3. Connection Termination

1. Connection Establishment :

- Before a communicating device can send data to the other, the initiating device must first determine the availability of the other to exchange data and pathway has to be found through the network by which the data can be sent. This is called connection establishment.

- Connection establishment require three actions which is called a three way hand shake.

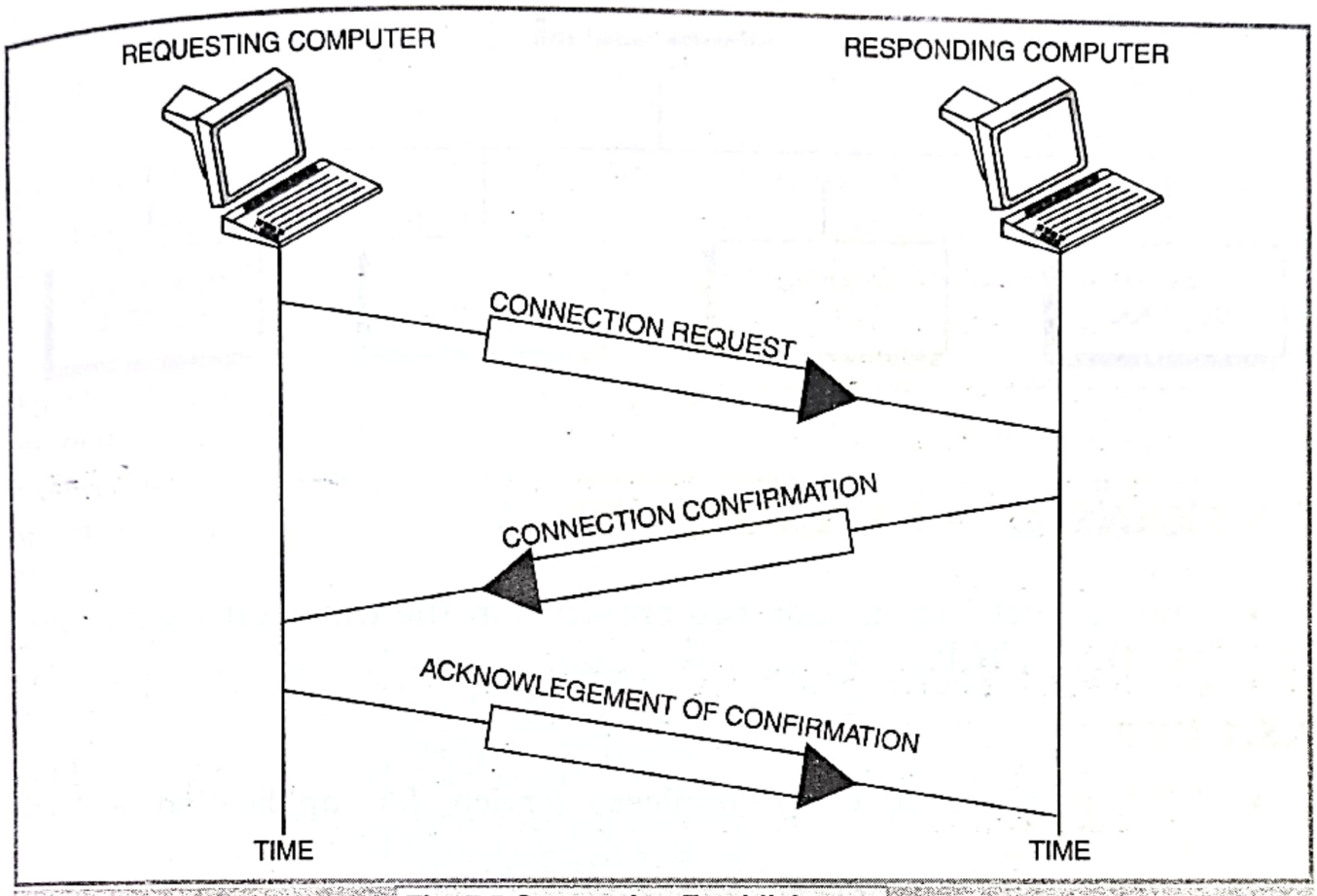


Fig. 7.5 Connection Establishment

(*i*) The computer requesting the connection send a connection request packet to the desired.

(*ii*) The responding computer returns a confirmation packet to the requesting computer.

(*iii*) The requesting computer returns a packet acknowledging the confirmation

2. Data Transfer :

After the establishment of the connection between the system the data transferring starts between the systems.

3. Connection Termination/Realising

- Once all of the data have been transferred, the connection has to be terminated.

(*i*) The requesting computer sends disconnection request packet.

(*ii*) The responding computer confirms the disconnection request.

(*iii*) The requesting computer acknowledges the confirmation.

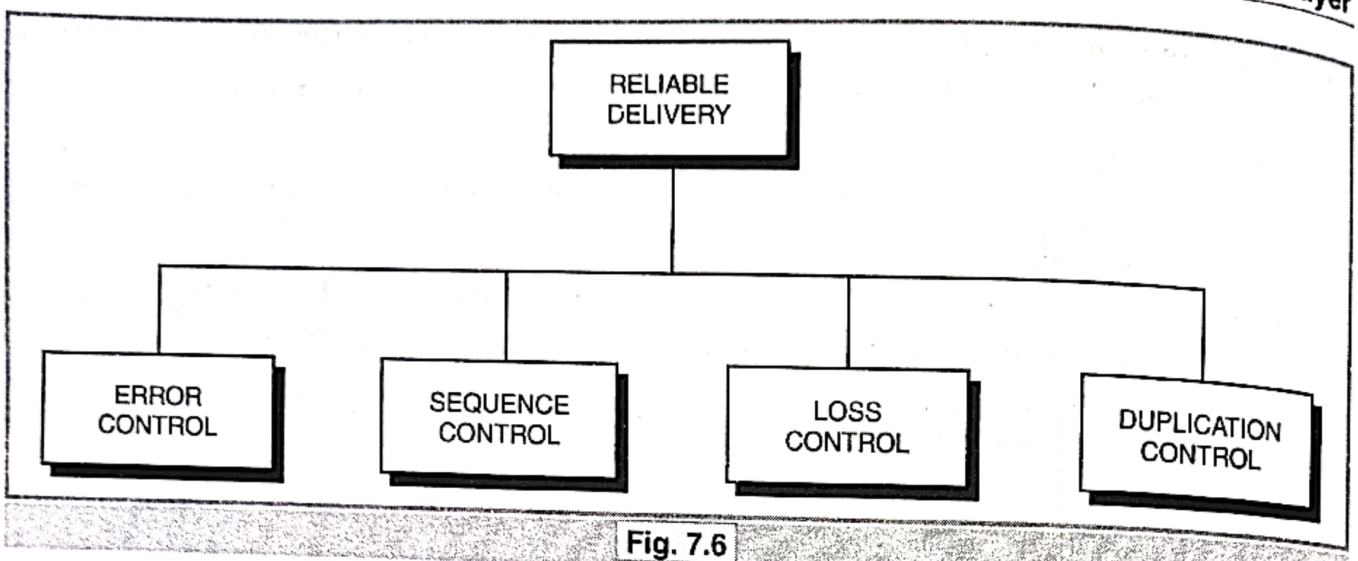


Fig. 7.6

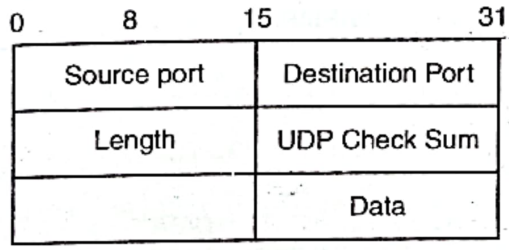
7.5 TRANSPORT LAYER PROTOCOL

- The internet has mainly two protocols in the transport layer protocl (i) UDP (ii) TCP.

7.5.1 UDP

- UDP provides a connectionless service for application – level procedures.
- UDP is basically an unreliable service: delivery and duplicate protection are not guaranteed. However, this does reduce the overhead of the protocol and may be adequate in many cases.
- The user datagram protocol (UDP) is unreliable, connectionless protocol.
- There is no need to establish a connection with a host before exchanging data with it using UDP, and there is no mechanism for ensuring that data sent is received.
- A unit of data sent using UDP is called a datagram UDP adds four 16-bit header fields (8 bytes) to whatever data is sent.
- These fields are : a length field, a checksum field, source port number and destination port number. Port number in this context, represents a software port, not a hardware port.
- The concept of port number is common to both TCP and UDP. The part number identify which protocol module sent (or is to receive) the data.
- Most protocol have standard port that are generally used for this. For example the Telnet protocol generally uses port 23. The simple and mail transfer protocol (SMTP).

- The use of standard port number makes it possible for client to communicate with a server without first having to establish which port to use.
- IP uses the protocol field to determine which data should be passed to the UDP or TCP module.
- UDP or TCP uses the port number to determine which application layer protocol should receive the data.
- Although UDP is not reliable, it is still an appropriate choice for many applications.
- It is used in real time application like Net audio and video where, if data is lost, it is better to do without it than send it again out of sequence. It is also used by protocol like Simple Network Management Protocol (SNMP).



UDP Datagram.

Fig. 7.7

1. Purpose of UDP

- UDP provides a connectionless packet service that offer unreliable "best effort delivery." This means that the arrival of packet is not guaranteed, nor is the correct sequencing of delivering packet.
- Applications that do not require any acknowledgement of data, for example, audio or video broadcasting uses UDP.
- It is also used by applications that typically transmit small amount of data at one time, the Simple Network Management Protocol (SNMP).
- UDP provides protocol port numbers used to be distinguished between the multiple program executing on a single device.

That is, in addition to the data sent, each UDP message contains destination port number and a source port number. This makes it possible for the UDP software at the destination to deliver the message to the correct application program, and for in application program to send a reply.

2. UDP Datagram format :

- The user datagram protocol is a simple transport.

- It is end to end transport level that adds only port address, checksum error control, and length information to the data from the upper layer. The packet produced by the UDP is called a user data gram. Its field are:

Source Port Address: It is the address of the application program that has created the message.

Destination Port Address : It is the address of the application program that has created the message.

Total Length : It defines the total length of the user datagram in bytes.

Checksum: It is a 16 bit field used in error detection.

Thus, there is a place at the transport level for both a connection-oriented and connectionless type of service.

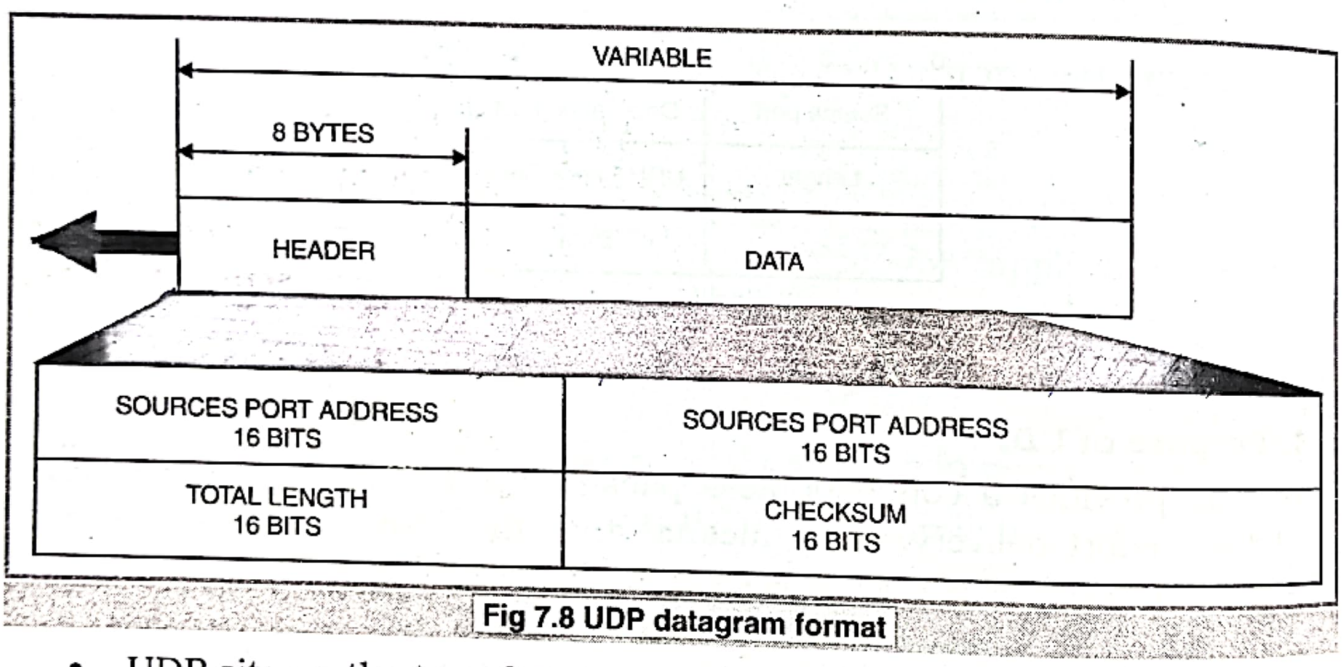


Fig 7.8 UDP datagram format

- UDP sits on the top of IP.
- Because it is connectionless UDP has very little to do. Essentially, it adds a port addressing capability to IP.
- This is best seen by examining the UDP header.
- The header includes a source port and destination port.
- The length field contains the length of the entire UDP segment, including header and data.
- The checksum is the same algorithm used for TCP and IP, For UDP, the checksum applies to the entire UDP segment plus a pseudo header prefixed to the UDP header at the time of calculation and which is same pseudo – header used for TCP.

- If an error is detected, the segment is discarded and no further action is taken.
- The checksum field in UDP is optional. If it is not used, it is set to zero.
- However, it should be pointed out that the IP checksum applies only to the IP header and not to the data field which in this case consists of UDP header and the user data.
- Thus, if no checksum calculation is performed by UDP, then no check is made on the user data.

3. Characteristics of UDP

1. Simple, high-speed, low-functionality "wrapper" that interfaces applications to the network layer and does little else. Connectionless;
2. Data is sent without setup.
3. Data is sent in discrete packages by the application.
4. Unreliable, best-effort delivery without acknowledgements.
5. Applications where data delivery speed matters more than completeness, where small amounts of data are sent; or where multicast/broadcast are used.

7.5.2 TCP

TCP (Transmission Control Protocol) is connection oriented and UDP (User's data protocol) is connectionless protocol.

- UDP is just IP address with an additional short header.
- The TCP provides the reliable transmission of data in IP environment. TCP correspond to the transport layer (Layer 4) of OSI model.
- TCP provides the stream of the data transfer, reliability, efficient flow control, full duplex operation and multiplex.
- TCP delivers unstructured stream of bytes unspecified by sequence number.
- This service benefits applications because they do not have to chope the data into the blocks before handing it off to TCP. TCP groups bytes into segment and passes them to IP for delivery.
- As it offers reliability by providing connection oriented, end to end reliable packet delivery through the internet.
- It does this by sequencing bytes with forwarding acknowledgement number that indicate to the destination the next byte the source expect to receive.

- Bytes not acknowledged within a specified time period is retransmitted.
- The reliability mechanism of TCP allows devices to deal with lost, delayed, duplicate or misread packet. A time out mechanism allows devices to detect lost packets and request retransmission.
- TCP offers efficient flow control, which means that when sending acknowledgements back to the source, the receiving TCP process indicates the highest sequence number that it can receive without overflowing its internal buffers.
- Full duplex operation means that TCP processes can both send and receive at the same time.
- Finally TCP multiplexing means that numerous simultaneous upper-layer conversion can be multiplexed over a single connection.

1. Characteristics of TCP

1. Full-featured protocol that allows applications to send data reliably without worrying about network layer issues.
2. Connection must be established prior to transmission.
3. Data is sent by the application with no particular structure.
4. Reliable delivery of messages; all data is acknowledged.
5. Delivery of all data is managed, and lost data is retransmitted automatically.
6. Flow control using sliding windows; window size adjustment heuristics; congestion avoidance algorithms.
7. Most protocols and applications sending data that must be received reliably, including most file and message transfer protocols.

2. TCP SERVICES :

- TCP is designed to provide a reliable communication between pairs of processes (TCP users) across a variety of reliable and unreliable networks and internets.
- TCP provides two useful facilities for labeling data: Data stream push and urgent data signalling.

1. Data stream push: Ordinarily, TCP decides when sufficient data have accumulated to form a segment for transmission. The TCP user can require a TCP to transmit all outstanding data up to and including that labeled with a push flag on the receiving end TCP will deliver this data to his user in the same manner. A user might request this if it has come to a logical break in the data.

2. Urgent Data Signaling: This provides a means of informing the destination TCP user that significant or "urgent" data in the upcoming data stream. It is up to the destination user to determine appropriate action.

3. TCP HEADER FORMAT:

- TCP use only a single type of protocol data unit, called a TCP segment. Because one header must serve to perform all protocol mechanism, it is rather large, with a minimum length of 20 octets. The fields are as follows:
  - Source port (16 bits): Source TCP user (originates from sending host).
  - Destination port (16 bits): Destination TCP user (Receiving HOST).
  - Sequence number (32 bit): Sequence number of the first data octet in this segment except when the SYN flag is set. If SYN is set, it is the initial sequence number. After reaching $2^{32}-1$, this number will wrap around too.
  - (ISN) and the first data octet is ISN+1.
  - Acknowledgement number (32 bits) a piggybacked acknowledgement, contains the sequence number of the next data octet that the TCP entity expects to receive.
  - Data offset (4 bit) words in the header.
  - Reserved (6 bits): reserved for future use .
  - Flags(6 bits): To Check in flag
  - URG: urgent pointer field significant.

ACK: acknowledgement field significant.
PSH: push function.
RST: reset the connection.
SYN: synchronize the sequence numbers.
FIN: no more data from sender.

* **Window (16 bits):** flow control credit allocation in octets. Contains the number of data octets beginning with the one indicated in the acknowledgement field that the sender is willing to accept.
* **Checksum (16 bits)**: A TCP sender computer a value based on the contents of the TCP Header and data fields. This 16-bit value will be coupared with the value the revciever generates using the same computation. If the value match, the receiver can be very much confident that segment arrived intact.

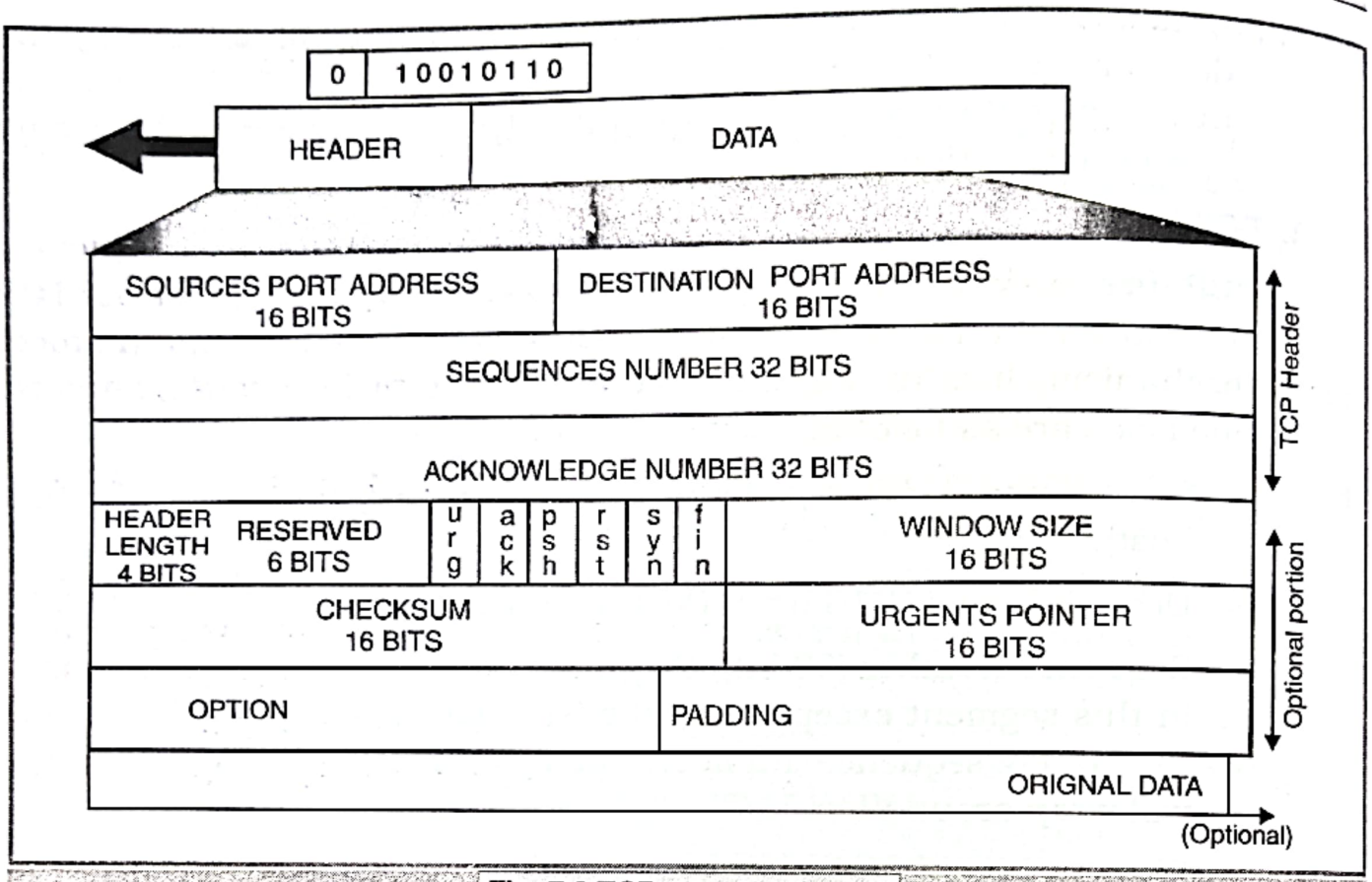


Fig. 7.9 TCP segment format

* **Urgent pointer (16- bits):** Points to the last octet in the sequence of urgent data. This allows the receiver to know how much urgent data are coming.
* **Options (variable):** It is the optional that specifies the maximum segment size that will be accepted.

The sequence number and acknowledgement number are bound to the octets rather than to the entire segments.

The checksum field applies to the entire segment plus a pseudo-header prefixed to the header at the time of calculation (at both transmission and reception). The pseudo-header includes the following fields from the IP header: source and destination internet address and protocol plus a segment length field.

4. Transport Control Protocol Operation (TCP Operation)

1. TCP incorporates many of the features of HDLC (High Level Data Link Control) protocol.
2. It supports duplex message transfer and follow the go-back-N error control procedure with a type of sliding window protocol.
3. All the PDUs that the TCP uses to set up the connection, transfer data and clear a connection have a standard format are known as segments.

4. Segments are transferred between two TCP entities in the user data field of UDP.
5. TCP has two addresses indicates the two end points of the logical connection between the two application protocol.
6. The sequence number has the same role as the send sequence number in HDLC protocol, and the acknowledgment number have the same role as receive sequence number. Thus the former relates to the flow of the data in the direction of sending TCP entity and the later to the flow in the reverse direction.
7. The presence of the option field in the segment header means that the header can be of the variable length.
8. All the segments have the same header format and the validity of selected fields in the segment header is indicated by setting of bits in the 6-bit code field, if bit is a set (= 1), the corresponding field is valid. Note that multiple bit can be set in a single segment.
9. The window field relate to the sliding window flow control mechanism.
10. The check sum relate to the complete segment, header plus contents. The check sum is the complement of the sum of all then 16 bit words in the segment added together using 1's-compliment arithmetic.
11. The URG flag is set to the code field. The urgent pointer indicates that the amount of urgent data in the segment. Normally this is delivered by TCP entity immediately it is received.
12. The default maximum number of octect in the data field segment is 536. This has been choosen on the assumptions that WANs are present in the route, in general, they have an inferior bit error rate probability.

5. TCP IMPLEMENTATION POLICY OPTIONS:

- The TCP standard provides a precise specification of the protocol to be used between TCP entities.
- However, certain aspects of the protocol admit several possible implementation options. Although, two implementation that choose alternative options will be interoperable, they may be performance implications.
- The design areas for which options are specified are the following:
  (a) Send policy
  (b) Deliver policy
  (c) Accept policy

(d) Retransmit policy
(e) Acknowledgement policy

(a) Send policy:

- A sending TCP entity is free to transmit data at its own convenience.
- As data are issued by the user, they are buffered in transmit buffer.
- TCP may construct a segment for echo batch of data provided by its user or it may wait until a certain amount of data accumulates before constructing and sending a segment.
- The actual policy will depend on performance considerations.
- If transmission is infrequent and large, there is overhead in terms of segment generation and processing.
- On the other hand if transmissions are frequent and small, then the system is providing quick response.

(b) Deliver Policy

- A receiving TCP entity is free to deliver data to the user as its own convenience.
- It may deliver data as each in order segment is received, or it may buffer data from a number of segments in the receive buffer before delivery.
- The actual policy will depend on performance considerations.
- If delivers are infrequent and large, the user is not receiving data as promptly as may be desirable.
- On the other hand, if deliveries are frequent and small there may be unnecessary processing both in TCP and in the user software, as well as necessary number of operating system interrupts.

(c) Accept policy:

- When all data segments arrive in order over a TCP connection, TCP places the data in a receive buffer for delivery to the user.
- The receiving TCP entity has two options:
  - **In order:** Accept only segment that arrive in order; any segment that arrives out of order is discarded.
  - **In window:** Accept all segments that are within the receive window.

(d) Retransmit policy:

- TCP maintains a queue of segments that have been sent out but not yet acknowledged.
- The TCP specification states that TCP will retransmit a segment if it fails to receive an acknowledgement within a given time.
- A TCP implementation may employ one of three retransmission strategies:
  - **First- only:** Maintain one retransmission for the entire queue. If an acknowledgement is received, remove the appropriate segment or segments from the queue and destroy the corresponding timer or timers. If any timer expires, retransmit the corresponding segment individually and reset its timer.

(e) Acknowledge Policy:

- When a data segment arrives that is in sequence, the receiving TCP entity has two options concerning the timing of acknowledgement:
  - **Immediate:** When data are accepted, immediately transmit an empty (no data) segment containing the appropriate acknowledgement number.
  - **Cumulative:** When data are accepted, record the need for acknowledgement, but wait for an outbound segment with data on which to piggyback, the acknowledgement. To avoid long delay, set a window timer if the timer expires before an acknowledgement is sent, transmit an empty segment containing the appropriate acknowledgement number.

7.6 TCP/IP

The transmission control protocol/ internet working protocol (TCP/IP) is a set of protocol, that defines all the transmissions exchanges across the Internet. TCP/IP is in active use for almost 20 years and has proved its effectiveness on a global scale.

TCP/ IP and the Internet: TCP/IP and the concept of internetworking have developed together, each the cause of growth of the other.

An internet under TCP/IP operates like a single network connecting many computers of any size and type. Internally, an internet is an interconnection of the independent physical networks (such as LANs) linked together by internetworking devices.

To TCP/IP, the same internet appears quite differently. TCP/ IP, considers all interconnected physical networks to be one huge network. It considers all the hosts to be connected to this larger logical network rather than to their individual physical networks.

7.7 COMPARISON OF TCP AND UDP

- The transport layer is responsible in TCP/IP by two protocols : TCP and UDP.
- User Data gram protocol UDP is simple. It provides non sequential transport functionality with speed at any size.
- The TCP offers reliability and security, but at the cost of the speed.
- Both make end to end delivery. Since reliable and safe transmission is more important. TCP is more commonly used. UDP is simpler than TCP.UDP is faster than TCP as no connection establishment or termination is required. Also, as it header is shorter less overhead are involved. UDP is used when speed is more important than reliability.
- TCP is a reliable stream transport Port to Port protocol. The term stream means connection –oriented i.e. a connection must be established between both ends of a transmission before any data transmission. By this TCP generates a virtual circuit between sender and receiver that remains active during transmission. The term reliable means the system under reference can perform sequence control, error control and loss control.TCP protocol includes all these control. Hence, it is considered as reliable.

Tabular Characteristics of TCP & UDP

Characteristic/ Description	UDP	TCP
General Description	Simple, high-speed, low-functionality "wrapper" that interfaces applications to the network layer and does little else.	Full-featured protocol that allows applications to send data reliably without worrying about network layer issues.
Protocol Connection Setup is	Connectionless; data is sent without setup.versions)	Connection-oriented; connection must be established prior to transmission.
Data Interface to Application	Message-based; data is sent in discrete packages by the application.	Stream-based; data is sent by the application with no particular structure.

Reliability and Acknowledgements	Unreliable, best-effort delivery without acknowledgements.	Reliable delivery of messages; all data is acknowledged.
Retransmissions	Not performed. Application must detect lost data and retransmit if needed.	Delivery of all data is managed, and lost data is retransmitted automatically.
Features Provided to Manage Flow of Data	None	Flow control using sliding windows; window size adjustment heuristics; congestion avoidance algorithms.
Overhead	Very low	Low, but higher than UDP
Transmission Speed	I Very high	High, but not as high as UDP
Data Quantity Suitability	Small to moderate amounts of data (up to a few hundred bytes)	Small to very large amounts of data (up to gigabytes)
Types of Applications That Use The Protocol	Applications where data delivery speed matters more than completeness, where small amounts of data are sent; or where multicast/broadcast are used.	Most protocols and applications sending data that must be received reliably, including most file and message transfer protocols.
Well-known Appli-cations and Protocols	Multimedia appli-cations, DNS, BOOTP, DHCP, TFTP, SNMP, RIP, NFS (early	FTP, Telnet, SMTP, DNS, HTTP, POP, NNTP, IMAP, BGP, IRC, NFS (later versions)

SUMMARY

- It is heart of the whole protocol hierarchy. Its task is to provide reliable, cost effective data transport from the source machine to destination machine ,independent of the physical networks or network current in use.
- The existence of the transport layer possible for the transport service to be more reliable than the underlying network service. Lost packets and mangled data can be detected and compensated for the transport layer.
- The bottom four layer can be seen as transport service provider, where as upper layers are the transport service user.

- The Transport Service primitives allow transport users to access the transport service. Each transport service has its own access primitives. The transport service is similar to network service, but there are also some important differences. The main difference is that the network service is intended to model the service offered by real networks. Real networks can lose packets, so network services are generally unreliable.
- Transport protocols must be able to do connection management over unreliable networks. Connection establishment is complicated by the existence of the delayed duplicate packets. To deal with them, three-way handshake are needed to establish connection. Releasing a connection is easier than establishing .
- The main Internet transport protocol is TCP. It uses 20-byte header on all segments can be fragmented by routers with in the Internet. A great deal of work has gone into optimizing TCP performance.
- ATM has four protocols in the AAL layer. All of them break messages into cells at he source and reassemble the cells into messages at destination.

EXERCISE

SHORT ANSWER QUESTIONS

1. What is connection oriented Transport Protocol ?
2. What is Multiplexing ?
3. What do you understand by Addressing ?
4. What is flow control ?
5. Write the TCP Header Format ?
6. What are the various TCP implimentation Policy ?
7. What is source port address in UDP diagram ?
8. What is a connection management in Transport Protocal ?
9. What is the difference between TCP and UDP ?
10. What is retransmit policy ?

DESCRIPTIVE QUESTION

1. Explain in detail the concept of Transport protcol.
2. Differentiate between TCP and UDP taking suitable Example ?
3. Explain TCP Implementation Policy ?
4. Explain Connection Management ?

□□□□□