UNIT 3 TRANSPORT LAYER PROTOCOLS

Structure **Page Nos.**

3.0 INTRODUCTION

The internet work environment consists of hosts connected to networks that are in turn interconnected via gateways. Such networks are based on packet switching technology. The active elements that produce and consume messages are the *processes.* Therefore, such communication can be viewed as an inter-process communication.

As the computer communication systems are playing an important role in military, government and civilian environments, it has become essential to provide some means of interconnecting the communication networks and to provide standard inter-process communication protocols that can support a broad range of applications. In anticipation of the need for such standards, the Department of Defence (DoD), USA, decided to employ Transmission Control Protocol (TCP) as the basis of inter-process communication protocol.

3.1 OBJECTIVES

Our objective is to introduce you to the basic concepts of Transport layer Protocols. On successful completion of this unit, you should be able to:

- have a reasonable understanding of the Transmission Control Protocol architecture;
- understand the role and use of port numbers;
- describe the operation of Transmission Control Protocol and its header format;
- describe the role and meaning of User Datagram Protocol (UDP), and
- describe the role and meaning of Internet Control Message Protocol (ICMP).

Ports

When a process starts up, it registers a port number with the protocol stack. The port numbers are specified by a 16-bit number i.e., the overall available set of port numbers are 2^{16}, ranging from 0 to 65535. The various categories of ports are as under:

1) **Well Known Ports:** The Port numbers in the range 0-1023 are called 'Well Known Ports'. These port numbers are assigned to the server side of an application and are already reserved for specific applications by IANA (Internet Assigned Number Authority). Some of these port numbers and their associated application are given in ***Table*** I.

Table 1: Port number and associated application

Port Nos.	Protocol	Application	Port No	Protocol	Application
23	TCP	TELNET	110	TCP	POP3
25	TCP	SMTP	161	UDP	SNMP
37	UDP	TIME	179	TCP	BGP
43	TCP	WHOIS	443	TCP	HTTPS
53	TCP/UDP	DNS	68	UDP	BOOTPC
67	UDP	BOOTPS	69	UDP	TFTP

2) **Registered Ports:** Port numbers in the range 1024-49151 are called Registered Ports. These port numbers have been publicly defined as a convenient service for the Internet community to help them avoid vendor conflicts.

3) **Dynamic and/or Private Ports:** The remaining port numbers in the range 49152-65535, are called Dynamic and/or Private Ports and can be used freely by any client or server application.

Remember, only one process per protocol can listen on a given port i.e., two different processes, one using UDP and another TCP can both listen on port number XXX. However, two processes using the same transport protocol cannot listen on the same port number.

In order to establish a connection between two end machines i.e., providing the process-to-process delivery of messages, we need an identifier called socket. It contains the combination of IP address of the machine along with the port number of the process running on it. For example, suppose IP address of the machine is 171.10.20.1 and a process running on it is assigned a port number of 4534, then the socket contains (171.10.20.1, 4544). Thus a socket uniquely identifies the process from the set of processes running on a machine.

In the client server model, a pair of sockets is required i.e., one for the client ***(client*** IP ***address, client port number)*** and another for the server ***(server*** IP ***address, server port number).***

3.2 OVERVIEW OF TCP

In the previous discussion, we have discussed that in data link layer, frames are delivered node-to-node (intermediate nodes) and in the network layer, datagrams are delivered host-to-host. But the real communication takes place between two processes running on two different machines. Therefore, the basic function of the transport layer is process-to-process delivery of messages. However, it is also necessary to identify the various kinds of processes running on the machine (e.g., e-mail, ftp, telnet). Thus, we need some kind of addressing scheme for transfer of data between the processes.

In order to deliver some information to a specific destination, we need an address for correspondence. Similarly, at the data link layer, medium access control addresses (MAC) are employed for addressing. Correspondingly, at the network layer, Internet Protocol IP addresses are required. In the same way, the transport layer uses the port numbers for addressing between two processes. These port numbers are just a way of labelling the packets. When the transport layer receives a message from an application, first of all, it registers a temporary port number for the application process and subsequently assigns it. Thereafter, it sends the data to the designated destination application program that would be running on another port number. In ***Figure*** 1, the client server model has been illustrated, wherein, whenever a client process starts up, a port number is assigned, say 3000 and the server process is assigned port number of

100. These assigned port numbers are put on each packet transmitted between these processes.

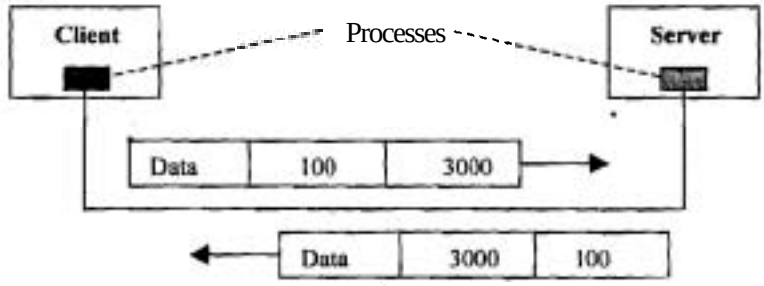

Figure: 1 Client Server Model

It may be noted that IP addresses and port numbers play different roles in communication networks. The IP address simply identifies the destination machine from the network, while the port number identifies the particular process running on that destination machine.

There are two types of protocols at the transport layer as shown in *Figure.2*.

1) Transmission Control Protocol (TCP)
2) User Datagram Protocol (UDP).

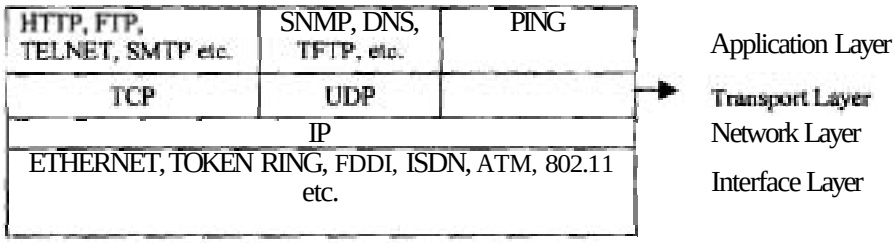

Figure 2: Protocol Stack

3.3 TRANSMISSION CONTROL PROTOCOL (TCP)

Transmission control protocol (TCP) of the transport layer provides end-to-end reliable communication. TCP is a connection oriented communication service. As the IP layer of TCP/IP protocol stack is connectionless, the TCP layer adds the reliability feature along with connection-oriented service. Basically, TCP establishes a virtual circuit between the source and the destination and the stream of bytes are transferred through that circuit such that the order of the segments remains integral. The acknowledgement and retransmission of lost and damaged segments is guaranteed. In TCP, two processes running on two different hosts communicate with each other as shown in *Figure3*.

The various services being offered by TCP to the processes running on application layer are as under:

1) **Stream data transfer:** With stream data transfer, TCP delivers an unstructured stream of bytes identified by sequence numbers. This service benefits applications because they do not have to chop data into blocks before handing it off to TCP. Instead, TCP groups bytes into segments and passes them to IP for delivery.

2) **Reliability:** TCP offers reliability by providing connection-oriented, end-to-end reliable packet delivery through an internetwork. It does this by sequencing bytes with a forwarding acknowledgement number that indicates to the destination, the next byte the source expects to receive. Bytes not acknowledged within a specified time period are retransmitted. The reliability mechanism of TCP allows devices to deal with lost, delayed, duplicate or misread packets. A

time-out mechanism allows devices to detect lost packets and request retransmission.

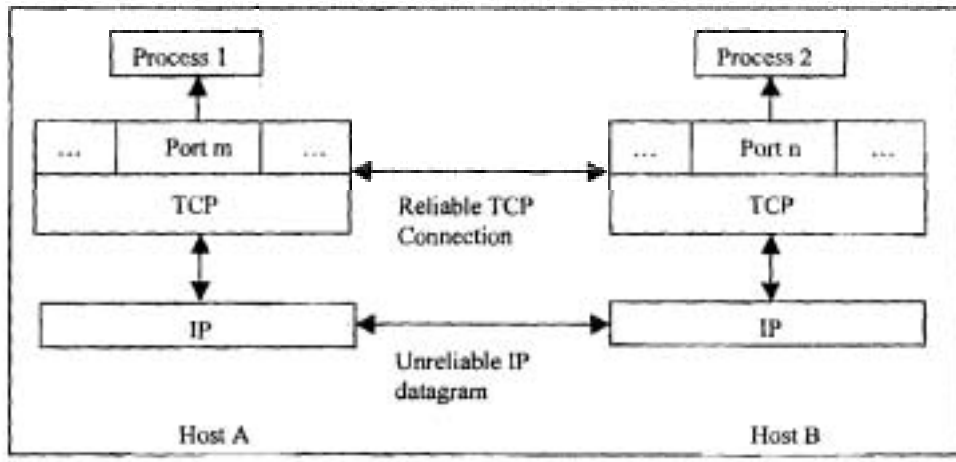

Figure 3: TCP Connection between two processes running on different machines

3) **Efficient flow control:** TCP offers efficient flow control, which means that, when sending acknowledgements back to the source, the receiving TCP process indicates the highest sequence number it can receive without overflowing its internal buffers.

4) **Full-duplex operation:** TCP offers full-duplex operation, wherein TCP processes can both send and receive data at the same time.

5) **Multiplexing: TCP's** multiplexing provides numerous simultaneous upper-layer conversations multiplexed over a single connection.

3.3.1 TCP Header

The TCP data unit is called a segment. The format of the segment is shown in Figure 4.

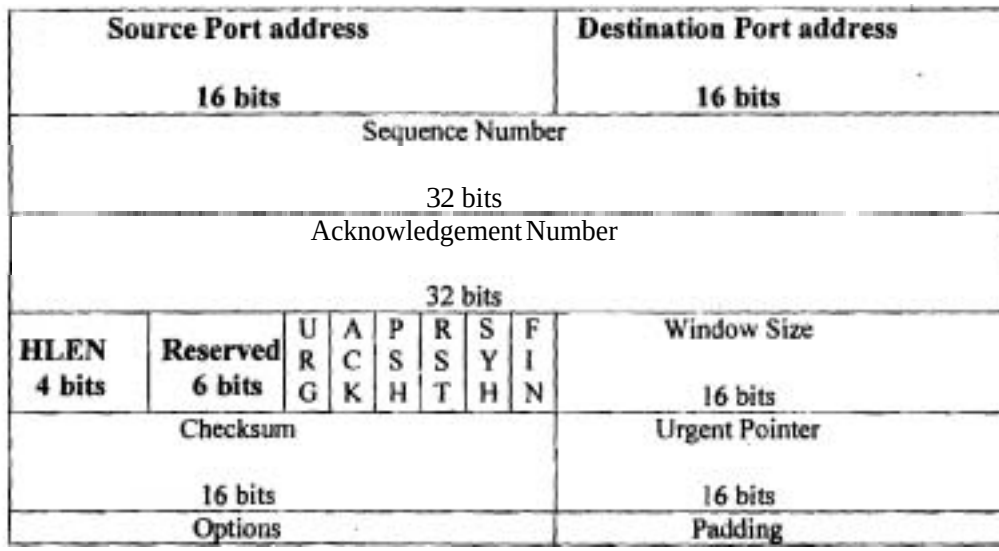

Figure 4: TCP Header

1) **Source Port Number:** This is a 16-bit number which defines the source port number for a particular application program that is sending the TCP segments.

2) **Destination Port Number:** This is a 16-bit number which defines the destination port number for a particular application program that is receiving the TCP segments.

3) Sequence Number: As the unit of data transfer in TCP is termed as segment, each segment's first data byte number denotes the 32-bit sequence number. Since the sequence number refers to a byte count rather than a segment count, sequence numbers in contiguous TCP segments are not numbered sequentially. For example, if a file of 5000 bytes is transferred using TCP connection and the first byte is numbered as 20002 and data divided into 5 segments each of 1000 bytes, the sequence numbers assigned to various segments are as under:

Segment 1 → sequence number: 20002

Segment 2 → sequence number: 21002

Segment 3 → sequence number: 22002

Segment 4 → sequence number: 23002

Segment 5 → sequence number: 24002

4) Acknowledgement Number: This is used by the sender to acknowledge the received data. This 36-bit field indicates the sequence number of the next byte expected from the receiver. For example, if host A has sent a segment having sequence number 2000 to host B, host B would send an acknowledgement with acknowledgement number field set to 2001 (one plus the sequence number of last received segment).

5) Header Length: The HLEN field consists of 4 bits. It indicates the length of the TCP header. The length of the TCP header can be between 20 bytes to 60 bytes i.e. HLEN field can have binary values ranging from 0101-1111 (5 to 15, 32 bit words) (5 × 4 =20, 15 x 4 =60).

6) Reserved: This 6 bit field is reserved for future use. The value set in this field must be zero.

7) Control Flags: This field contains six different control flags that can control certain aspects of the TCP connection such as connection establishment, connection termination and flow control. The flags include:

a) Urgent Pointer URG: When set, the ACK indicates that the current segment contains urgent (or high-priority) data and that the Urgent Pointer field value is valid.

b) Acknowledgement (ACK): When set, indicates that the value contained in the Acknowledgement Number field is valid. This bit is usually set, except during the first message during connection establishment.

c) Push (PSH): PSH is used when the transmitting application wants to force TCP to immediately transmit the data that is currently buffered to the application without waiting for the buffer to fill. It is useful for transmitting small units of data.

d) Reset (RST): When set, RST immediately terminates the end-to-end TCP connection.

e) Synchronize (SYN): SYN is set in the initial segments used to establish a connection, indicating that the segments carry the initial sequence number.

f) Finish (FIN): FIN is set to request normal termination of the TCP connection in the direction this segment is travelling. Complete closure of the connection requires one FIN segment in each direction.

8) Window Size: The window size 16 bits field is used for flow control. It contains in bytes, the size of the window that the receiver has to maintain i.e., the value of the receive *window size.* It is basically the number of transmitted bytes that the sender of this segment is willing to accept from the receiver.

9) Checksum: This is a 16-bit field that provides bit error detection for the segment (including the header and data).

10) Urgent Pointer: Urgent data is information that has been marked as high-priority by a higher layer application. The data sent under high-priority usually bypasses the normal TCP buffering and is placed in a segment between the header and normal data. When the URG flag of control flag is set, then the urgent pointer 16-bit number indicates the position of the first octet of non-priority data in the segment.

11) Options: The option field contains 40 bytes of optional information about connection establishment. The maximum segment size (MSS) is the most commonly used option and if absent, defaults to an MSS of 536. Another option is Selective Acknowledgement (SACK), which allows out-of-sequence segments to be accepted by a receiver. The further discussion about options is beyond the scope of this book.

Characteristics of TCP

The basic characteristics of TCP are as follows:

1) It employs a connection-oriented service for communication.

2) It is a reliable source of communication i.e. guarantees delivery of messages.

3) It splits the messages into segments and keeps track of the order (sequence) of segments.

4) It employs the checksums for detecting any errors in data as well as the TCP header.

3.3.2 TCP Connection Establishment and Termination

A connection is a requirement of a reliable data delivery service. It is set up before the actual data exchange takes place. The connection is used to acknowledge the receipt of packets and retransmit those that are lost.

TCP provides a connection-oriented service over packet switched networks, 'Connection-oriented' implies that there is a virtual connection between two endpoints. There are three phases in any virtual connection. These are the connection establishment, data transfer and connection termination phases.

3.3.3 TCP Connection Establishment

In order to connect two machines, first of all both machines must initialize the communication and get a formal approval from each other before the start of data transfer. The establishment of such a connection by exchange of control messages is known as the three-way handshake. The process of the three-way handshake is as under:

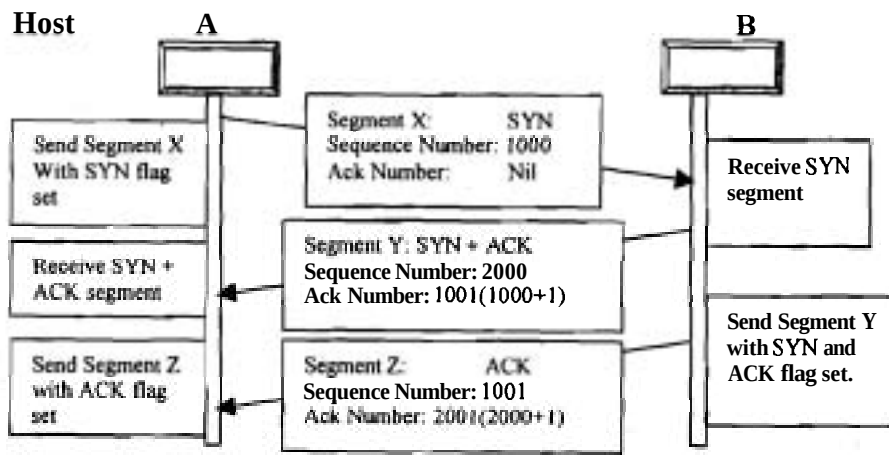

Figure 5: TCP Connection Establishment

1) The initiator of the session (source machine) sends a segment with SYN flag set to the destination node along with the proposed initial sequence number (ISN) in the sequence number field of the segment. Remember, the segment includes

the source and destination port numbers. For example, in ***Figure*** 5, Host A sends a segment named X, with SYN flag set, initial sequence number is 1000 and acknowledgement number is Nil.

2) Upon receipt of the segment, the recipient sends a segment with SYN flag set to the initiator with the acknowledgement number set to the sequence number (ISN) + 1 and the ACK flag is also set. The sent segment is assigned a new sequence number (ISN) for its own segment. For example, in ***Figure*** 5, Host B, alter receiving the segment, sends a segment named Y with SYN + ACK flag set, initial sequence number assigned to the segment is 2000 and acknowledgement number is 1001(sequence number of segment sent by host A plus 1).

3) The initiator after receiving the segment sends a segment with ACK flag set in response to the recipient's SYN, with the acknowledgement number set to the recipient's sequence number (ISN) + 1. For example, in ***Figure*** 5, Host A, after receiving the segment, sends a segment named Z, with ACK flag set, sequence number assigned to the segment is 1001(the first segment sent by A had a sequence number of 1000) and acknowledgement number is 2001(sequence number of segment sent by host B plus 1).

3.3.4 TCP Connection Termination

There is a saying that, "all good things must come to an end" and so it is with TCP connections. After the exchange of data between the source and destination nodes, either of them can close the connection. Thus, after the user has sent all its data and wishes to close the TCP connection, four segments are required to completely close a connection gracefully from both directions. The connection termination phases are discussed below:

1) The initiator (source) sends a segment with the FIN flag set. For example, in ***Figure 6,*** Host A sends a segment named X, with FIN flag set, initial sequence number is 2000 and acknowledgement number is Nil.

2) Upon receipt (destination) of the segment, the recipient issues an ACK segment, in order to confirm the receipt of the initiator. The acknowledgement number is 1 plus the sequence number received from the initiator i.e. from the FIN segment. For example, in ***Figure*** 6, Host ***B,*** after receiving the segment, sends a segment named Y, ACK flag set, initial sequence number assigned to the segment is 4000 and ack no. is 2001(sequence number of segment sent by host A plus 1).

3) If the recipient (destination) still has some data to send to the initiator (source), it sends the data. Otherwise, the recipient (destination) sends a request for termination of connection from its side by sending a segment with FIN flag set.

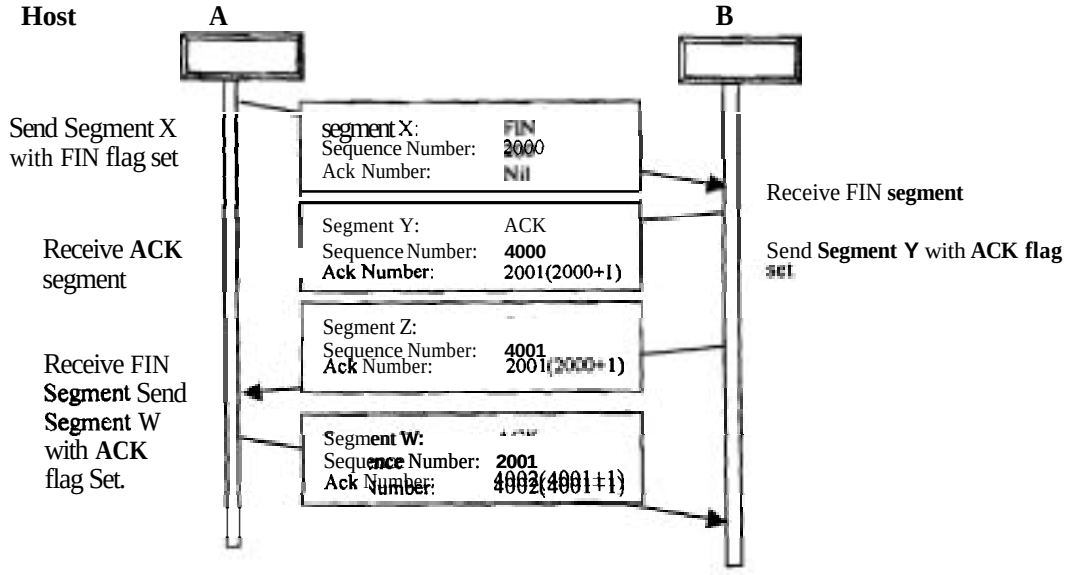

Figure 6: TCP Connection Termination

For example, in ***Figure*** 6, in case Host B too wishes to disconnect, it sends another segment named Z, with FIN flag set, sequence number assigned to the segment is 4001(sequence number of the last segment sent by B plus 1) and acknowledgement number is 2001(sequence number of segment sent by host A plus 1).

4) On receipt of this segment, the initiator sends an ACK segment, to confirm the FIN segment from the recipient(destination). For example, in ***Figure*** 6, Host A, after receiving the segment, sends a segment named W, with ACK flag set, sequence number assigned to the segment is 2001(the first segment sent by A had a sequence number of 1000) and acknowledgement number is 4002(the last sequence number of segment sent by host B plus 1).

Another method of closing the connection can be done by requesting the resetting of connection i.e., a host machine can send a segment with the **RST** bit set which will convey the other machine to immediately terminate the connection. Apart from connection establishment and connection termination, flow control, error control and congestion control are few other responsibilities being performed by TCP.

Flow control in TCP

The flow control coordinates the amount of data that can be sent by the source between the terminals before receiving acknowledgement from the destination.

In order to speed up the communication, the source can send the complete message at a given instance without waiting for acknowledgement to arrive from the receiver. However, such a situation can cause adverse effects at the receiver side. In such situations, TCP sends data in accordance with sliding window protocol – a Flow Control Mechanism.

Sliding Window Protocol

In the sliding window protocol, a window is maintained for each connection. The window defines the size of the buffer i.e. the total number of bytes that can be sent by a terminal at a given time shown in ***Figure*** 7, this also shows the total number of blocks signifies the total size of the window. The sliding window also keeps track of bytes which have been sent but are unacknowledged, bytes still stored in buffer but haven't been sent etc.

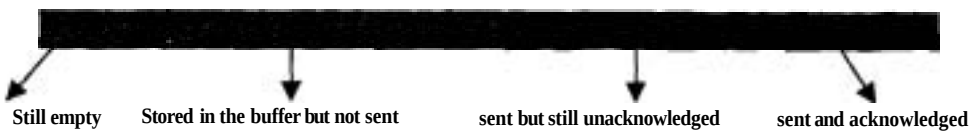

Figure 7: Sliding Window of sender node

Initially a window size is negotiated between the end terminals especially from the receiving side, while establishing a connection. Since TCP provides a byte-stream connection, sequence numbers are assigned to each byte in the stream. TCP divides this contiguous byte stream into TCP segments to transmit them. Therefore, the window principle is used at the byte level, that is, the segments sent and acknowledgements received will carry byte-sequence numbers and the window size is expressed as a number of bytes.

There are mainly two windows that are maintained i.e., one by the sender and another one by the receiver. The receiver window stores information about the various bytes received by its buffer but are still unconsumed and the other empty locations of the buffer.

With the help of these windows, a maximum limit on the number of bytes that can be sent by the sender to the receiver is decided, depending upon the total number of bytes that can be stored in the receiver's buffer at that particular instance. Thus, the mechanism of sliding window helps in controlling the flow of packets.

Error Control in TCP

The major tasks of error control in TCP include detection of out-of-order segments, lost segments, duplicate segments and corrupted segments. The above-mentioned

problems are deciphered using the aid of acknowledgements, checksum and time-out period.

Out-of-Order segment

As IP is a connectionless service, IP datagrams might arrive out of order. The TCP in turn assures that the receiver does not acknowledge the received out of order segment until and unless it receives all those expected segments which precede it as shown in *Figure 8.*

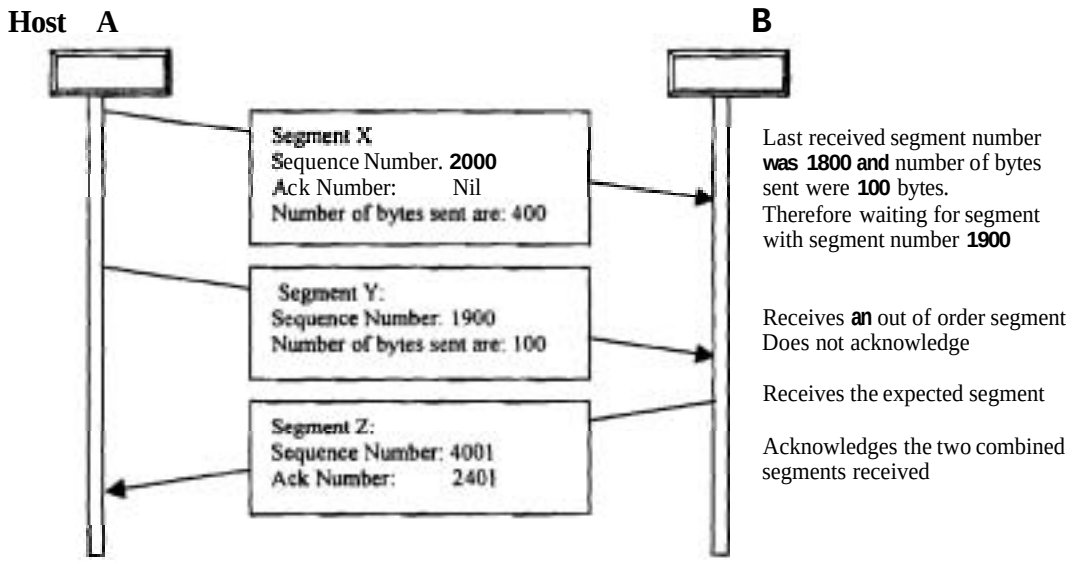

Figure 8: Out of order segments

Duplicate Segment

If the receiver receives a duplicate segment, it simply discards that segment, as a TCP segment with the same sequence number has already arrived.

Corrupted Segments

Whenever a segment gets corrupted, it is simply discarded by the destination and has to be retransmitted by the source as shown in Figure 9.

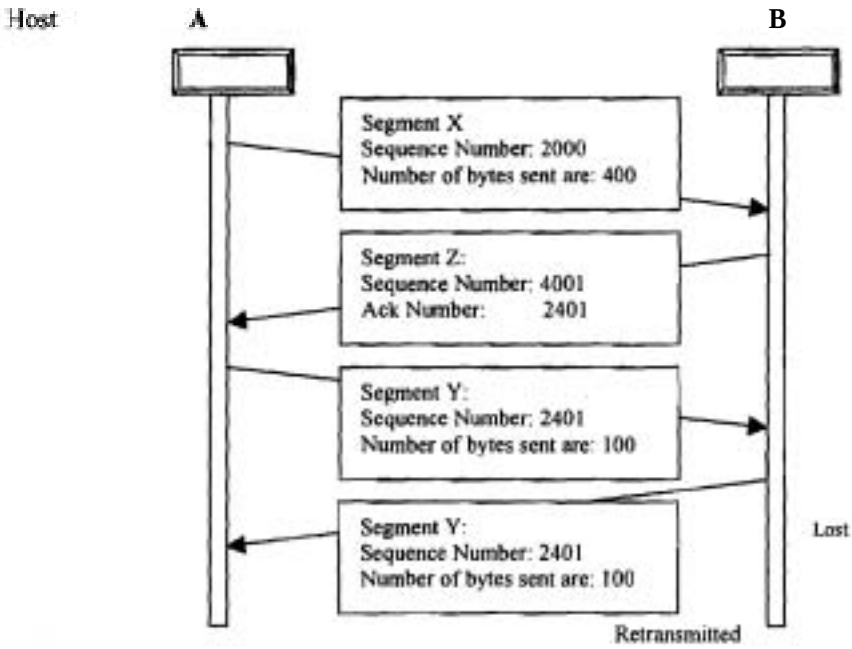

Figure 9: Corrupted Segment

Lost Acknowledgements

An acknowledgement for a segment can be lost in between. However, in TCP, the functioning of acknowledgement is as follows: An acknowledgement is a confirmation that everything up to the bytes specified by the acknowledgement number in TCP header has been received. Thus, a latest acknowledgement overrides the previous reached /lost acknowledgements as shown in Figure 10.

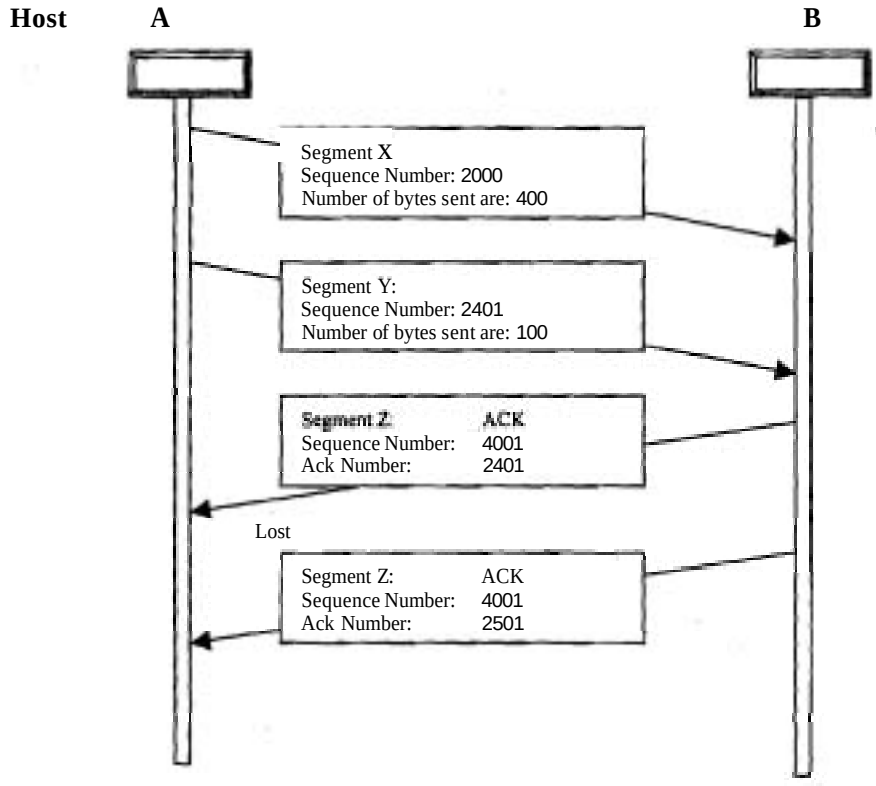

Figure 10: Lost Acknowledgement

Check Your Progress 1

1) Which of the following are transport layer protocols?

a) TCP
b) UDP
c) IP
d) Both a and b

.....

2) The combination of IP address and port numbers is known as-

a) Socket Address
b) Passive Address
c) Transport Address
d) Network Address

.....

3) Which of the following function is performed by the TCP.

a) Process-to process communication
b) End to end delivery of data
c) Host to host communication
d) None of the above

.....

.....

4) How many types of port numbers are used?
   a) One
   b) Two
   c) Three
   d) None of the above

.....

5) Which of the following are parts of TCP header?
   a) Sequence Number
   b) Acknowledgement Number
   c) Checksum
   d) All of the above.

.....

3.4 USER DATAGRAM PROTOCOL (UDP)

UDP is an unreliable transport layer based protocol. It uses the connectionless service for providing the communication between the nodes of similar and/or different networks. The IP layer provides host-to-host communication. However, UDP provides process-to-process communication.

As UDP is connectionless, it means that no overheads will be incurred in terms of connection establishment, connection termination, acknowledgement of packets, sequence number assigned to each packet etc. Similar to the services offered by IP, UDP does not provide flow control and error control. In UPD, whenever the message is ready for transmission, it is simply transferred to the lower layers without bothering about establishment of connection. Thus, in situations where a small message has to be transmitted, UDP should be employed. On the other hand, the UDP is an unreliable source of communication. It does not guarantee accuracy, reachability and timeliness. Sequence numbers are not assigned to the packets, therefore, order of packets is not maintained and packets are unacknowledged.

IJDP is a convenient protocol for multimedia and multicasting applications. It is suitable for processes that require simple request – reply communication with little interest in flow and error control.

UDP Header

The UDP packets are termed as User Datagrams and the header part of the user datagram has a fixed size of 8 bytes as shown in Figure 11.

1) **Source Port Number:** A 16-bit number which defines the source port number for a particular application program that is sending the UDP datagrams.

2) **Destination Port Number:** A 16-bit number that defines the destination port number for a particular application program that is receiving the UDP datagrams.

Source port number 16 bits	Destination port number 16 bits
Length 16 bits	Checksum 16 bits
Data	

Figure11: UDP Header

3) **Length:** The 16-bit field denotes the size of UDP header combined with payload data. It can range between 0 to 65,535 bytes.

4) **Checksum:** This is 16-bit field that provides detection of errors over the entire user datagram.

Characteristics of UDP

The basic characteristics of UDP are as follows:

1. UDP is a connectionless service.
2. It adds no non-reliable flow control to IP.
3. It serves as a multiplex/demultiplexer for sending and receiving datagrams.
4. The communication ends (end terminals) need not be synchronized.
5. There is no provision for acknowledgement of datagrams.

Applications of UDP

The standard applications using UDP include:

- Trivial File Transfer Protocol (TFTP)
- Domain Name System (DNS) name server
- Remote Procedure Call (RPC) used by the Network File System (NFS)
- Simple Network Management Protocol (SNMP).

Internet Control Message Protocol (ICMP)

The Internet Control Message Protocol notifies the sender of IP datagrams about abnormal events. ICMP is important in the connectionless environment. ICMP messages are carried in IP packets. The most commonly employed ICMP message types include:

- **Destination Unreachable:** This message type indicates that a packet cannot be delivered because the destination host cannot be reached. The reason for the non-delivery may be that the host or network is unreachable or unknown, the protocol or port is unknown or unusable.
- **Echo and Echo Reply:** These two messages are used to check whether hosts are reachable on the network. One host sends an Echo message to the other, optionally containing some data and the receiving host responds with an Echo Reply containing the same data. These messages are the basis for the Ping command.
- **Source Quench:** Sent by a router to indicate that it is experiencing congestion and is discarding datagrams.
- **TTL Exceeded:** This message indicates that a datagram has been discarded because the TTL field reached 0 or because the entire packet was not received before the fragmentation timer expired.
- **Timestamp and Timestamp Reply:** These messages are similar to the Echo messages, but place a timestamp (with millisecond granularity) in the message, yielding a measure of how long remote systems spend buffering and processing datagrams and providing a mechanism so that hosts can synchronize their clocks.

Check Your Progress 2

1) The UDP performs which of the following function:

   a) Process-to process communication

   b) End to end delivery of data

   c) Host to host communication

   d) None of the above

.....

.....

2) Which of the following are parts of **UDP** header?
   a) Sequence Number
   b) Acknowledgement Number
   c) Checksum
   d) All of the above

.....

3) Which of the following field (s) is/are required for ordering the packets?
   a) Sequence Number
   b) Acknowledgement Number
   c) Checksum
   d) All of the above

.....

4) What is the purpose of sliding window protocol?
   a) Congestion Control
   b) Flow Control
   c) Error Control
   d) All of the above

.....

5) How is TCP different from UDP?

.....

3.5 SUMMARY

This unit provides a complete overview of TCP. We have learnt that the data link layer provides node-to-node (intermediate), network layer provides host-to-host and transport layer provides process-to-processcommunication. The transport layer uses TCP and UDP protocols. TCP is a reliable. connection-oriented service. while UDP is an unreliablk and connectionless service. As real communication takes place between two application programs, each process should be assigned a unique port number for identifying the various processes running on the same machine. There are three types of port numbers: well known, registered and dynamic ports. The combination of IP address and port number is called a socket address. The size of UDP header is 8 bytes and TCP header requires 20-60 bytes. The TCP employs error control, flow control and congestion control mechanisms. A three-way handshake is required for TCP connection establishment. The TCP connection termination requires four steps. The next unit of this course covers the application layer protocols and their role in TCP/IP.

3.6 SOLUTIONS/ANSWERS

Check Your Progress 1

1) D
2) A
3) A
4) C
5) D

Check Your Progress 2

1) A
2) C
3) A
4) B
5) TCP is a connection-orientedservice and UDP is a connectionlessservice.

3.7 FURTHER READINGS

1) Andrew S. Tanenbaum, *Computer Networks*, Third edition.

2) Behrouz A. Forouzan, *Data Communications and Networking*, Third edition.

3) Douglas E. Comer, *Internetworking with TCP/IP Vol.1*: Principles, Protocols, and Architecture (4th Edition).

4) James F. Kurose, *Computer Networking: A Top-Down Approach Featuring the Internet* (3rd Edition).

5) Larry L. Peterson, *Computer Networks: A Systems Approach*, 3rd Edition (The Morgan Kaufmann Series in Networking).

6) W. Richard Stevens, *The Protocols (TCP/IP Illustrated, Volume 1)*.

7) William Stallings, *Data and Computer Communications*, Seventh Edition.