UNIT-V

1. Describe how reliable and ordered delivery is achieved through TCP.

Transmission Control Protocol (TCP - RFC 793) is considered as a reliable protocol. Transmission Control Protocol (TCP) is responsible for breaking up the message (Data from application layer) into TCP Segments and reassembling them at the receiving side. It is not sure that the data reaching at the receiving device is in the same order as the sending side, because of the problems in network or different paths packets flow to the destination. TCP is responsible for keeping the unordered segments in the right order. TCP assures a reliable delivery by resending anything that gets lost while traveling the network.

Characteristics of Transmission Control Protocol (TCP).

Stream Data transfer: Applications working at the Application Layer transfers a contiguous stream of bytes to the bottom layers. It is the duty of TCP to pack this byte stream to packets, known as TCP segments, which are passed to the IP layer for transmission to the destination device. The application does not have to bother to chop the byte stream data packets.

Reliability: The most important feature of TCP is reliable data delivery. In order to provide reliability, TCP must recover from data that is damaged, lost, duplicated, or delivered out of order by the Network Layer. TCP assigns a sequence number to each byte transmitted, and expects a positive acknowledgment (ACK) from the receiving TCP layer. If the ACK is not received within a timeout interval, the data is retransmitted. The receiving TCP uses the sequence numbers to rearrange the TCP segments when they arrive out of order, and to eliminate duplicate TCP segments.

Flow control: Network devices operate at different data rates because of various factors like CPU and available bandwidth. It may happen a sending device to send data at a much faster rate than the receiver can handle. TCP uses a sliding window mechanism for implementing flow control. The number assigned to a segment is called the sequence number and this numbering is actually done at the byte level. The TCP at the receiving device, when sending an ACK back to the sender, also indicates to the TCP at the sending device, the number of bytes it can receive (beyond the last received TCP segment) without causing serious problems in its internal buffers.

Multiplexing: Multitasking achieved through the use of port numbers.

Connections: Before application processes can send data by using TCP, the devices must establish a connection. The connections are made between the port numbers of the sender and the receiver devices. A TCP connection identifies the end points involved in the connection.

A socket number is a combination of IP address and port number, which can uniquely identify a connection.
Full duplex: TCP provides for concurrent data streams in both directions

2. Why TCP does use an adaptive retransmission and describes its mechanism.

Answer:

TCP Adaptive Retransmission and Retransmission Timer Calculations

Whenever a TCP segment is transmitted, a copy of it is also placed on the retransmission queue. When the segment is placed on the queue, a retransmission timer is started for the segment, which starts from a particular value and counts down to zero. It is this timer that controls how long a segment can remain unacknowledged before the sender gives up, concludes that it is lost and sends it again.

The length of time we use for retransmission timer is thus very important. If it is set too low, we might start retransmitting a segment that was actually received, because we didn't wait long enough for the acknowledgment of that segment to arrive. Conversely, if we set the timer too long, we waste time waiting for an acknowledgment that will never arrive, reducing overall performance.

Difficulties in Choosing the Duration of the Retransmission Timer

Ideally, we would like to set the retransmission timer to a value just slightly larger than the *round-trip time (RTT)* between the two TCP devices, that is, the typical time it takes to send a segment from a client to a server and the server to send an acknowledgment back to the client (or the other way around, of course). The problem is that there ***is*** no such "typical" round-trip time. There are two main reasons for this:

- **Differences In Connection Distance:** Suppose you are at work in the United States, and during your lunch hour you are transferring a large file between your workstation and a local server connection using 100 Mbps Fast Ethernet, at the same time you are downloading a picture of your nephew from your sister's personal Web site—which is connected to the Internet using an analog modem to an ISP in a small town near Lima, Peru. Would you want both of these TCP connections to use the same retransmission timer value? I certainly hope not!
- **Transient Delays and Variability:** The amount of time it takes to send data between any two devices will vary over time due to various happenings on the internetwork: fluctuations in traffic, router loads and so on. To see an example of this for yourself, try typing "ping www.tcpipguide.com" from the command line of an Internet-connected PC and you'll see how the reported times can vary.

3. Illustrate the features of TCP that can be used but the sender to insert record boundaries into the byte stream. Also mention their original purpose.

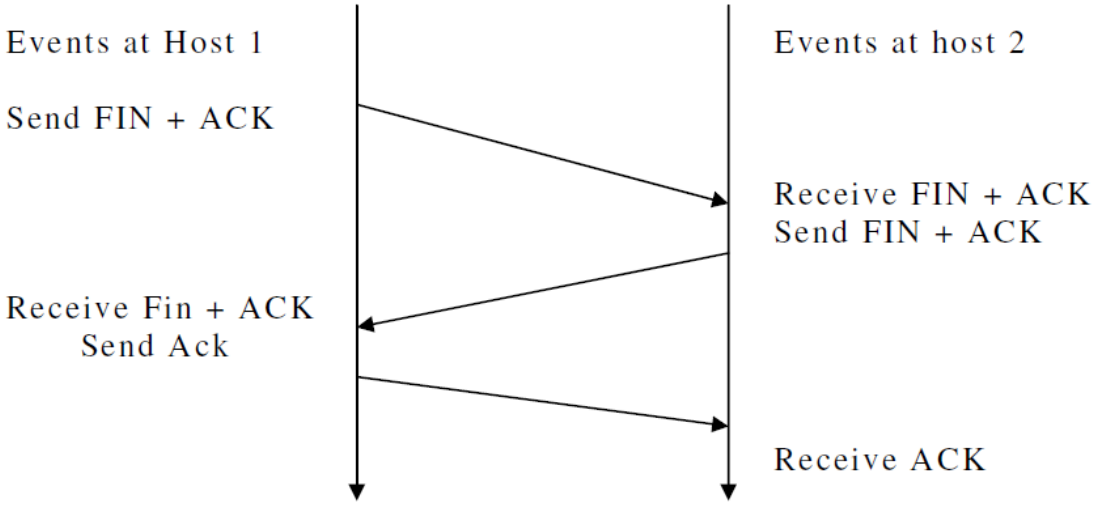
4. Highlight the features of UDP and briefly discuss the same.

5. Explain Three-Way Handshake Mechanism used by TCP to terminate a

Session reliably.

Answer:

To guarantee that connect ion are establ ished or terminated rel iably, TCP uses 3-way handshake in which three messages are exchanged. TCP uses the term synchronizat ion segment (SYN segment) to describe messages in a 3-way handshake used to create a connect ion , and the term FIN segment(short for finish) to describe messages in 3-way handshake to close a connect ion.



6. How congest ion is controlled in TCP?

Answer:

One of the most important aspects of TCP is a mechanism for congest ion control. In most modern internets, packet loss or extreme long delays are more likely to be caused by congest ion than a hardware failure. Interestingly, transport protocols that retransmit can exacerbate the problem of congest ion by injecting additional copies of a message.

To avoid such a problem, TCP always uses packet loss as a measure of congest ion and responds to congest ion by reducing the rate at which it retransmits data. TCP does not compute an exact transmission rate. Instead, TCP bases transmission on buffers. That is, the receiver advertises a window size and the sender can transmit data to fill the receiver 's window before an ACK is received. To cont rol the data rate, TCP imposes a restrict ion on the window size – by temporarily reducing the window size, the sending TCP effectively reduces the data rate.

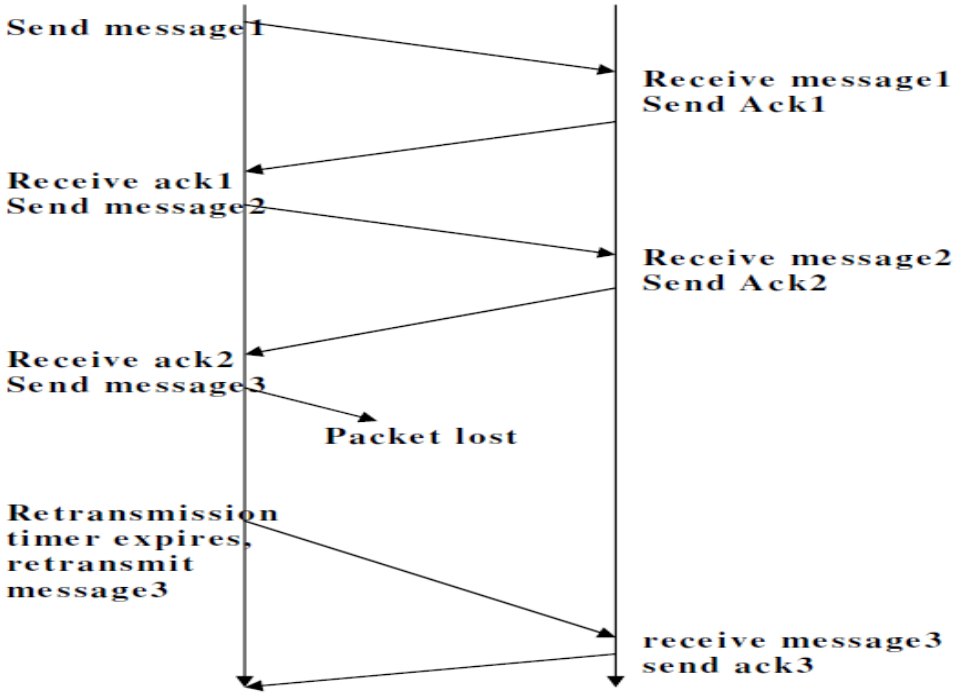
TCP congest ion control takes over when a message is lost . Instead of retransmitting enough data to fi l l the receiver 's buffer (the receiver's window size), TCP begins by sending a single message containing data. If the acknowledgement arrives without additional loss, TCP doubles the amount of data being sent and sends two additional messages. If acknowledgement arrive for those two , TCP sends four more and so on.

7. How does TCP achieve reliability?

Answer:

One of the most important technologies is retransmission. When TCP stands data the sender compensates for packet loss by implementing a retransmission scheme. Both sides of a

communication participate. When TCP receives data, it sends it acknowledgements back to the sender . Whenever it sends data, TCP starts a timer . if the timer expires before an acknowledgement arrives, the sender retransmits the data. The following figure illustrates retransmission.



8. Different Services of SCTP

Answer:

1. Process-to-Process Communication:

SCTP provides uses Process-to-Process Communication and also uses all well-known ports in the TCP space and also some extra port numbers.

2. Multiple Streams:

TCP is a stream-oriented protocol. Each connection between a TCP client and a TCP server involves one single stream. The problem with this approach is that a loss at any point in the stream blocks the delivery of the rest of the data. This can be acceptable when we are transferring text; it is not when we are sending real-time data such as audio or video. SCTP allows multi stream service in each connection, which is called association in SCTP terminology. If one of the streams is blocked, the other streams can still deliver their data.

2. **Multi homing:**

A TCP connection involves one source and one destination IP address. This means that even if the sender or receiver is a multihomed host (connected to more than one physical address with multiple IP addresses), only one of these IP addresses per end can be utilized during the connection. An SCTP association, on the other hand, supports multihoming service. The sending and receiving host can define multiple IP addresses in each end for an association. In this fault-tolerant approach, when one path fails, another interface can be used for data delivery without interruption. This fault-tolerant feature is very helpful when we are sending

and receiving a real-time payload such as Internet telephony.

4. Full-Duplex Communication:

Like TCP, SCTP offers full-duplex service, in which data can flow in both directions at the same time. Each SCTP then has a sending and receiving buffer, and packets are sent in both directions.

5. Connection-Oriented Service:

Like TCP, SCTP is a connection-oriented protocol. However, in SCTP, a connection is called an association. When a process at site A wants to send and receive data from another process at site B, the following occurs:
1. The two SCTPs establish an association between each other.
2. Data are exchanged in both directions.
3. The association is terminated.
4. Reliable Service: SCTP, like TCP, is a reliable transport protocol. It uses an acknowledgment mechanism to check the safe and sound arrival of data.

9. What are the different SCTP Features**?**
Answer**:**

Transmission Sequence Number (TSN):

The unit of data in TCP is a byte. Data transfer in TCP is controlled by numbering bytes by using a sequence number. On the other hand, the unit of data in SCTP is a DATA chunk which may or may not have a one-to-one relationship with the message coming from the process because of fragmentation, Data transfer in SCTP is controlled by numbering the data chunks. SCTP uses a transmission sequence number (TSN) to number the data chunks. TSNs are 32 bits long and randomly initialized between 0 and 232 - 1. Each data chunk must carry the corresponding TSN in its header.

Stream Identifier (SI):

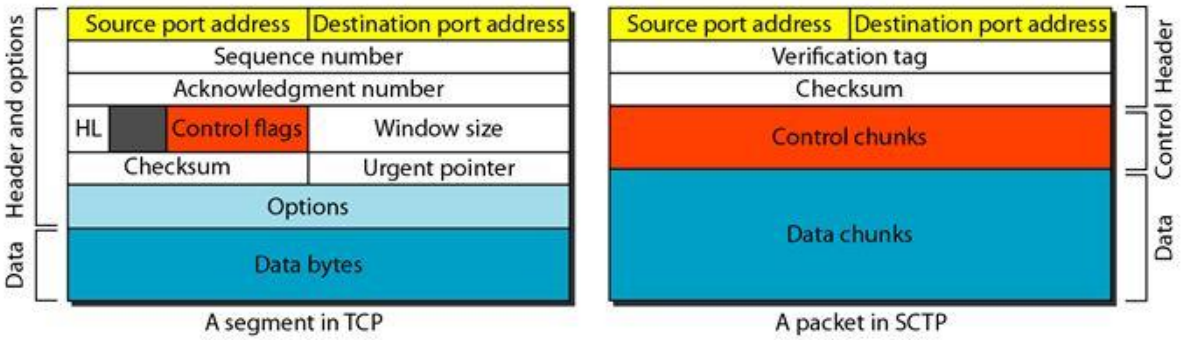
In TCP, there is only one stream in each connection. In SCTP, there may be several streams in each association. Each stream in SCTP needs to be identified by using a stream identifier (SI). Each data chunk must carry the SI in its header so that when it arrives at the destination, it can be properly placed in its stream. The 51 is a 16-bit number starting from 0.

Stream Sequence Number (SSI):

When a data chunk arrives at the destination SCTP, it is delivered to the appropriate stream and in the proper order. This means that, in addition to an SI, SCTP defines each data chunk in each stream with a stream sequence number (SSN).

Packets:

The design of SCTP packet is totally different: data are carried as data chunks, control information is carried as control chunks. Several control chunks and data chunks can be packed together in a packet. A packet in SCTP plays the same role as a segment in TCP. The following figure compares a segment in TCP and a packet in SCTP. Let us briefly list the differences between an SCTP packet and a TCP segment:



1. The control information in TCP is part of the header, the control information in SCTP is included in the control chunks. There are several types of control chunks each is used for a different purpose.
   2. The data in a TCP segment treated as one entity. But an SCTP packet can carry several data chunks and each can belong to a different stream.
   3. The options section, which can be part of a TCP segment, does not exist in an SCTP packet. Options in SCTP are handled by defining new chunk types.
   4. The mandatory part of the TCP header is 20 bytes, while the general header in SCTP is only 12 bytes. The SCTP header is shorter due to the following

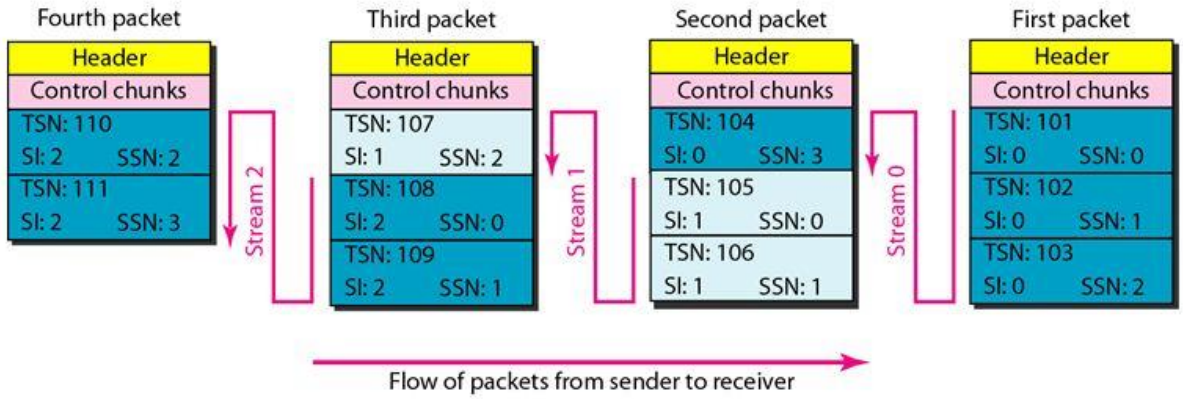
5. The checksum in TCP is 16 bits; in SCTP, it is 32 bits.

6. The verification tag in SCTP is an association identifier, which does not exist in TCP. In TCP, the combination of IP and port addresses defines a connection. In SCTP we may have multi homing using different IP addresses. A unique verification tag is needed to define each association.

7. TCP includes one sequence number in the header, which defines the number of the first byte in the data section. An SCTP packet can include several different data chunks. TSNs, SIs, and SSNs define each data chunk.

8. Some segments in TCP that carry control information (such as SYN and FIN) need to consume one sequence number. Control chunks in SCTP never use a TSN, SI, or SSN. These three identifiers belong only to data chunks, not to the whole packet.

In SCTP, An association may send many packets, a packet may contain several chunks, and chunks may belong to different streams. For example suppose that process A needs to send 11 messages to process B in three streams. The first four messages are in the first stream, the second three messages are in the second stream, and the last four messages are in the third stream.
Although the process could deliver one message from the first stream and then another from the second, we assume that it delivers all messages belonging to the first stream first, all messages belonging to the second stream next, and finally, all messages belonging to the last stream. We also assume that the network allows only three data chunks per packet, which means that we need four packets as shown in the following figure.



Data chunks in stream 0 are carried in the first packet and part of the second packet those in stream 1 are carried in the second and third packets; those in stream 2 are carried in the third and fourth packets.
Note that each data chunk needs three identifiers: TSN, SI, and SSN. TSN is a cumulative number and is used for flow control and error control. SI defines the stream to which the chunk belongs. SSN defines the chunk's order in a particular stream. In our example, SSN starts from 0 for each stream.

Acknowledgment Number:
• TCP acknowledgment numbers are byte-oriented and refer to the sequence numbers. SCTP acknowledgment numbers are chunk-oriented. They refer to the TSN.

• A second difference between TCP and SCTP acknowledgments is the control information. This information is part of the segment header in TCP. To acknowledge segments that carry only control information, TCP uses a sequence number and acknowledgment number (for example, a SYN segment needs to be acknowledged by an ACK segment).

• In SCTP, however, the control information is carried by control chunks, which do not need a TSN. These control chunks are acknowledged by another control chunk of the appropriate type (some need no acknowledgment).

Flow Control:

Like TCP, SCTP implements flow control to avoid overwhelming the receiver.
Error Control:

Like TCP, SCTP implements error control to provide reliability. TSN numbers and acknowledgment numbers are used for error control.
Congestion Control:

Like TCP, SCTP implements congestion control to determine how many data chunks can be injected into the network.