Unit - 4
Transport Layer

- Transport Layer is the fourth layer on the sender side of the OSI reference model.
- It lies b/w network layer and session layer.
- The (5) functions/services of the Transport Layer are:
    1. Segmentation and Reassembly of data: At the sender-side, the transport layer breaks down the data from session layer(upper layer) into multiple variable sized partitions (known as segments/datagrams), and then provides it to the network layer (lower layer), and vice-versa (re-assemble) at the receiver-side.
    2. Flow Control: Transport Layer is responsible for managing the data transfer rate b/w sender and receiver, so that the receiver may not be overwhelmed with the receiving data. This ensures that there is no data loss. It is similar to that of the data-link layer, but the only difference is that, in the data-link layer , flow control is performed across a single link whereas in the transport layer, end-to-end flow control is performed.
    3. Error Detection: The checksum value(a unique value string of numbers and letters) is assigned to each segment, to detect whether the data is lost or not. If lost, then only that segment will be re-transferred.
    4. Connection Control:
        + Transport Layer provides two methods to transfer data(i.e, connection and connectionless). A data can be sent without first establishing a connection with the receiving device, and without.
        + The Method which first establishes a connection before sending, is known as Connection-Oriented method, and the protocol used is TCP.
        + The Method which sends the data to the destination address, without first establishing a connection, is known as Connectionless method, and the protocol used is UDP.
        + Hence, if the Connection-oriented Method is used, the Transport layer is responsible for Establishing, Maintaining, and Terminating the TCP connection.
    5. Addressing: The data to be sent is encapsulated with the IP address of the destination device and with the Port number of the destination application, and then sent over a medium. Hence, the transport layer is responsible for encapsulating the addresses of the destination device and its application so that the data may reach the correct end point.

Transport Services (Services offered by the Transport Layer):

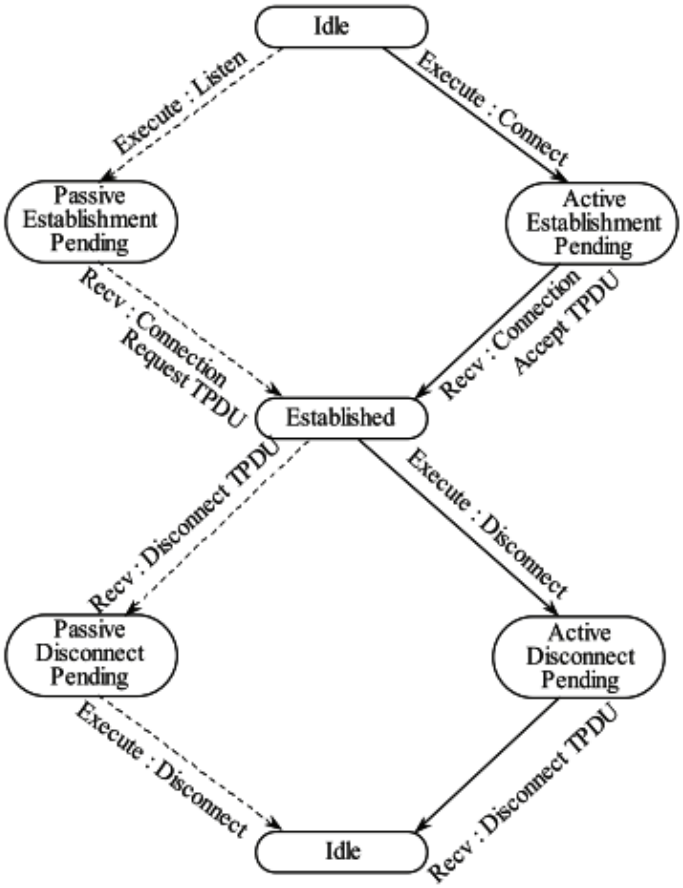
- Transport Layer offers two services/methods to transfer data, viz:
    1. Connection-Oriented Service
    2. Connectionless Service
- These are used based on the application requirements.

1. Connection-Oriented Service:
    - In this Service, before transmitting data, a connection is established with the receiving device.
    - Then, the segments of data are transmitted sequentially, through the I/O stream.
    - After all the segments are transmitted, the connection is removed.
    - With this Service, there is guarantee that the data will be transmitted successfully.
    - The protocol that offers this Service is TCP(Transfer Control Protocol).
    - This service is used for web-browsing, emailing, etc., applications
2. Connectionless Service:
    - In this Service, the destination(receiver) address is encapsulated with the data packet and sent independently (without any connection).
    - No connection is established and not ensured whether the receiver is listening on the I/O Stream or not.
    - Instead of a sequential set of segments, a single data packet/datagram is transferred at a time.

- With this Service, there is no guarantee that the data packet will be received at the destination or not.
- Hence, the data packet must be sent only when it is assured that the receiver is listening to the I/O stream and is ready to receive the data packet, to avoid data loss.
- The protocol that offers this Service is UDP(User Datagram Protocol)
- This service is used for video and audio streaming, online video games, etc., applications.

Transport Service Primitives:

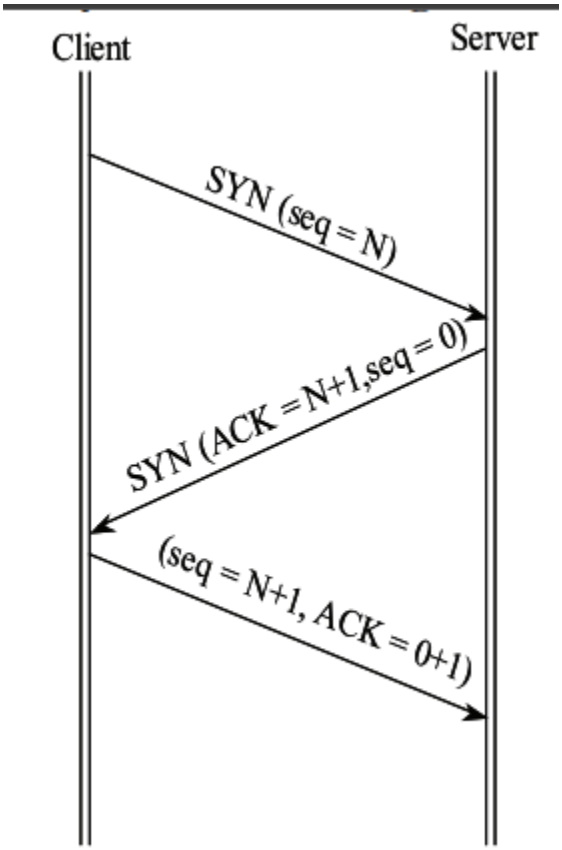
- Not every application establishes and manages a connection when the application starts. Some Applications are user-input driven.
- That is, in some applications a connection is established and managed based on the user input event.
- For such user-event driven applications, the Transport Layer offers some of its services, in the form of interfaces, known as the Service Primitives.
- With these Service Primitives, the upper layers of the transport layer (including application layer) can have control over some of the services of the transport layer.
- The various Service Primitives offered by the Transport Layer are:
    - LISTEN: Server/Receiver is waiting for the connection request.
    - CONNECT: Client/Sender sends the connection request.
    - SEND: Transfer the data.
    - RECEIVE: Receive the data.
    - DISCONNECT: Terminate the connection.
- LISTEN, CONNECT, RECEIVE are known as Blocking Call Primitives, because when an application invokes these operations, it typically waits until the requested action is completed before continuing its execution.
- SEND, DISCONNECT are known as Non-Blocking Call Primitives, because when an application invokes these operations, they typically return control to the application immediately, allowing it to continue its execution.



Elements of Transport Protocols:

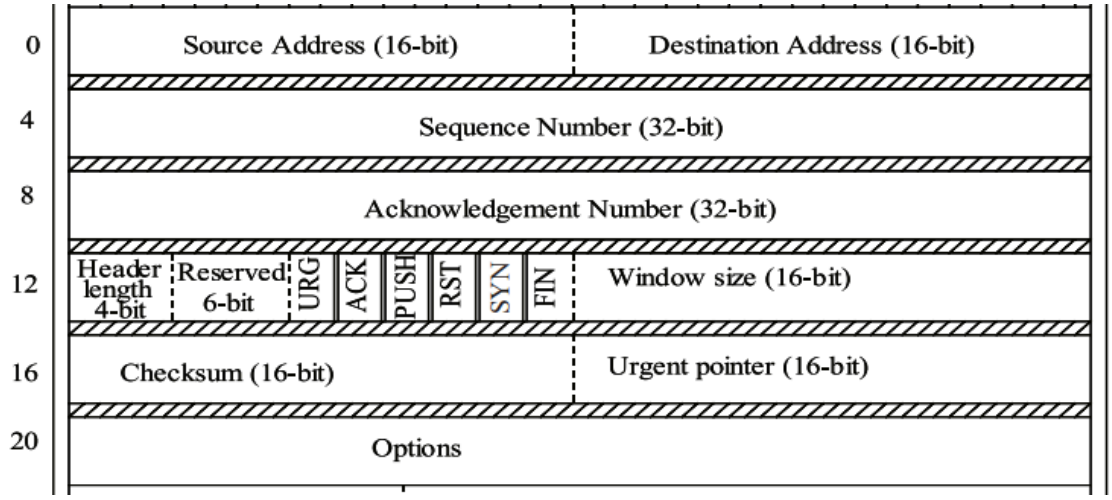
Connection Management:

- Transport Layer provides two methods to transfer data(i.e, connection and connectionless). A data can be sent without first establishing a connection with the receiving device, and without.
- The Method which first establishes a connection before sending, is known as Connection-Oriented method, and the protocol used is TCP.
- The Method which sends the data to the destination address, without first establishing a connection, is known as Connectionless method, and the protocol used is UDP.
- Hence, if the Connection-oriented Method is used, the Transport layer is responsible for Establishing, Maintaining, and Terminating the TCP connection.
- Hence, Connection Management is only when the connection-oriented service(TCP) is used.
- TCP uses a three-way handshaking technique to open a connection. (i.e, SYC, SYC-ACK, ACK)
    1. SYN (Synchronize): The client sends a SYN packet to the server, indicating its intention to establish a connection and specifying an initial sequence number.
    2. SYN-ACK (Synchronize-Acknowledgment): The server responds with a SYN-ACK packet, acknowledging the client's SYN and indicating its own initial sequence number.
    3. ACK (Acknowledgment): Finally, the client sends an ACK packet back to the server, acknowledging the receipt of the server's SYN-ACK. At this point, the connection is established, and data can be exchanged bidirectionally.



TCP (Transfer Control Protocol)

- It is a Connection-Oriented protocol used for reliable, ordered, and error-cheeked transmission of data.
- It operates in the Transport Layer of OSI Model and provides full-duplex and byte-stream communication b/w two hosts.
- In this protocol, a connection is established first, before sending the sequence of segments to destination. This guarantees the successful transmission of data.
- Every Segment of a data is encapsulated with multiple fields including the “checksum”, at the sender-side.
- “Checksum” fields in the headers of segments are used by the receiver to check whether the received data is correct/error-free or not.
- It is used in applications that require a reliable and ordered data exchange, such as messaging apps, web-browsing, Email transfer, File Transfer, Database Communication, etc.
- Features of TCP Connection are:
    - Full-duplex: Data can be transmitted in both directions(sender-receiver), simultaneously.
    - Reliability: TCP ensures that data is reliably delivered to the destination by acknowledging received segments, detecting and retransmitting lost segments, and sequencing segments to ensure correct order delivery.
    - Connection Establishment and Termination: TCP uses a three-way handshake for connection establishment and a four-way handshake for connection termination to ensure orderly setup and teardown of connections.
    - Error Detection and Correction: TCP includes mechanisms for error detection through checksums and, in some cases, error correction through selective retransmission of lost segments.
    - Congestion Control: TCP dynamically adjusts its transmission rate based on network conditions to avoid congestion and optimize throughput.
- When a connection is established, the Receiver sends its IP address and Port number of the requesting application.
- This IP address and Port number are encapsulated (as a field in header part) with the segments and sent by the receiver.
- TCP Header:
    - A part of the segment where the additional details such as source address, destination address, checksum etc, are encapsulated by the Transport Layer, before transmitting it, is known as the header.
    - The Header of a segment basically directs the segment to the correct destination and provides some error control information.
    - The Header of every segment is of fixed 20-bytes (20 * 8 = 160 bits)
    - Fields of Header:

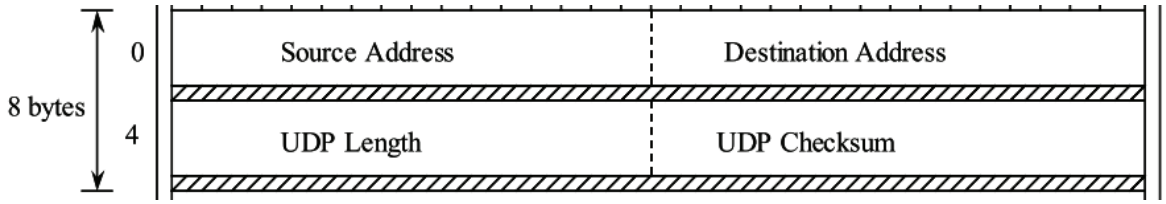


- Source Address(16 bit): It specifies the IP address of the Source machine and Port number of the responding application in the Source machine.
- Destination Address(16bit): It specifies the IP address of the destination machine and Port number of the requesting application in the destination machine.
- Sequence Number(32-bit): Every Segment of a data is assigned a number which indicates the position in a particular sequence, known as sequence number.The segments are transmitted in the sequence number order. It is used by the receiver to identify whether the currently received segment's sequence number matches with the next sequence number or not.
  (Ex: 1, 2, 3, 4, 5, …..)
- Acknowledgement Number(32-bit): It is sent by the receiver when a segment is successfully received with no errors. If (x) is the sequence number, then (x + 1) is the ACK number.
- Header Length(4-bit): It specifies the length of the header, which is fixed for every segment.
- Reserved Length(6-bit): a reserved field for any additional uses. It can be empty.
- Control Fields: These are 1 bit flags, used inconnection management.
  - URG(Urgent): It specifies if a part of segment data is to be processed urgently or not.
  - ACK(Acknowledgement): It is (1) if the transmission type is for Acknowledgement, from receiver to sender.
  - PSH(Push): The received segment data can be stored into a buffer in the transport layer or can be pushed to the upper layers. It is (1) if the segment data needs to be pushed to the application layer.
  - RST(Reset): If any segment is lost or got with error, then the receiver sets it to (1), indicating to reset the transmission. All segments will be re-transmitted from the first sequence number.
  - SYC(Synchronize): It is used in the initial phase, while requesting and accepting a connection.
  - FIN(Finish): It is used in the final phase, while terminating a connection.
- Window Size(16-bit): Specifies the size of the sender's receive window for flow control.
- Checksum(16-bit): An alpha-numeric sequence used for error detection.
- Urgent Pointer(16-bit): It is a pointer which points to a part of a segment, which has to be processed quickly/urgently.
- Options: Additional control and configuration options, such as maximum segment size (MSS) and timestamp.

UDP (User-Datagram-Protocol):

- It is a Connectionless and unreliable transport layer protocol, used for transmitting data.
- The Data is encapsulated with the destination IP address and its application's Port Number, and sent independently.
- That is, the data is sent without establishing a connection and without ensuring if the destination address is awake and listening or not. Hence, there is no guarantee of successful transmission of data.
- Unlike TCP, UDP does not provide features such as reliability and error control.
- It is suitable for real time communication applications where speed and efficiency are prioritized over reliability, such as audio and video streaming, online games, etc.
- Ulike TCP, where the data is divided into segments and transmitted sequentially, in UDP, the data is encapsulated with the header and sent as a whole, known as data packet/datagram.

- Features of UDP Connection:
  - Simple: Absence of the need for error control and other fields, makes it simple to be implemented.
  - Fast: UDP is specifically designed to provide Realtime and Fast communication.
- UDP Header:
  - Header of the data packet is of 8 bytes (8 * 8 = 64 bits)
  - Fields of Header:



    - Source Address(16 bit): It specifies the IP address of the Source machine and Port number of the responding application in the Source machine.
    - Destination Address(16bit): It specifies the IP address of the destination machine and Port number of the requesting application in the destination machine.
    - UDP Length(16 bit): It specifies the total length of the data packet.
      I.e, UDP Length = Payload Length + Header Length
    - UDP Checksum(16 bit): An optional field which is used by the receiver to check whether the data packet received is correct and error free or not.