TCP and UDP in Transport Layer

Layer 3 or the Network layer uses IP or Internet Protocol which being a connection less protocol treats every packet individually and separately leading to lack of reliability during a transmission. For example, when data is sent from one host to another, each packet may take a different path even if it belongs to the same session. This means the packets may/may not arrive in the right order. Therefore, IP relies on the higher layer protocols to provide reliability.

TCP (Transmission Control Protocol):

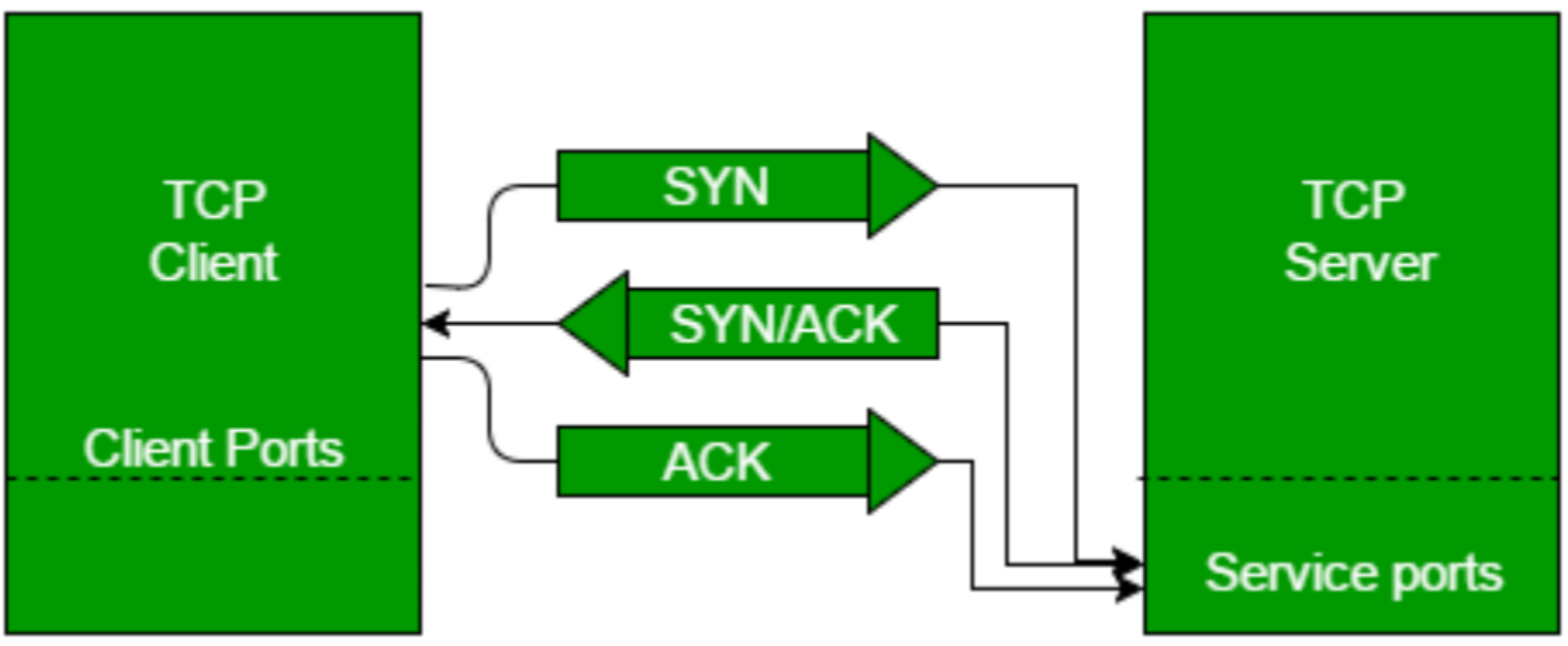
TCP is a layer 4 protocol which provides acknowledgement of the received packets and is also reliable as it resends the lost packets. It is better than UDP but due to these features it has an additional overhead. It is used by application protocols like HTTP and FTP.

UDP (User Datagram Protocol):

UDP is also a layer 4 protocol but unlike TCP it doesn't provide acknowledgement of the sent packets. Therefore, it isn't reliable and depends on the higher layer protocols for the same. But on the other hand it is simple, scalable and comes with lesser overhead as compared to TCP. It is used in video and voice streaming.

TCP is connection oriented protocol, in order to transmit segments from the sender to the receiver, a TCP connection needs to be established between them.

TCP connections go through a complete life cycle, roughly, the stages of a TCP connection are — establish connection, transfer data, terminate connection.



TCP provides reliable communication with something called **Positive Acknowledgement with Re-transmission(PAR)**. The Protocol Data Unit(PDU) of the transport layer is called segment. Now a device using PAR resend the data unit until it receives an acknowledgement. If the data unit received at the receiver's end is damaged(It checks the data with checksum functionality of the transport layer that is used for Error Detection), then receiver discards the segment. So

the sender has to resend the data unit for which positive acknowledgement is not received. You can realize from above mechanism that three segments are exchanged between sender(client) and receiver(server) for a reliable TCP connection to get established. Let us delve how this mechanism works :

- **Step 1 (SYN) :** In the first step, client wants to establish a connection with server, so it sends a segment with SYN(Synchronize Sequence Number) which informs server that client is likely to start communication and with what sequence number it starts segments with
- **Step 2 (SYN + ACK):** Server responds to the client request with SYN-ACK signal bits set. Acknowledgement(ACK) signifies the response of segment it received and SYN signifies with what sequence number it is likely to start the segments with

- **Step 3 (ACK) :** In the final part client acknowledges the response of server and they both establish a reliable connection with which they will start the actual data transfer

The steps 1, 2 establish the connection parameter (sequence number) for one direction and it is acknowledged. The steps 2, 3 establish the connection parameter (sequence number) for the other direction and it is acknowledged. With these, a full-duplex communication is established.

TCP Header-

The following diagram represents the TCP header format-

TCP Header

1. Source Port-

- Source Port is a 16 bit field.
- It identifies the port of the sending application.

2. Destination Port-

- Destination Port is a 16 bit field.
- It identifies the port of the receiving application.

3. Sequence Number-

- Sequence number is a 32 bit field.
- TCP assigns a unique sequence number to each byte of data contained in the TCP segment.
- This field contains the sequence number of the first data byte.

4. Acknowledgement Number-

- Acknowledgment number is a 32 bit field.
- It contains sequence number of the data byte that receiver expects to receive next from the sender.
- It is always sequence number of the last received data byte incremented by 1.

5. Header Length-

- Header length is a 4 bit field.
- It contains the length of TCP header.
- It helps in knowing from where the actual data begins.

Minimum and Maximum Header length-

The length of TCP header always lies in the range-

[20 bytes , 60 bytes]

- The initial 5 rows of the TCP header are always used.
- So, minimum length of TCP header = 5 x 4 bytes = 20 bytes.
- The size of the 6th row representing the Options field vary.
- The size of Options field can go up to 40 bytes.
- So, maximum length of TCP header = 20 bytes + 40 bytes = 60 bytes.

Concept of Scaling Factor-

- Header length is a 4 bit field.
- So, the range of decimal values that can be represented is [0, 15].
- But the range of header length is [20, 60].
- So, to represent the header length, we use a scaling factor of 4.

In general,

Header length = Header length field value x 4 bytes

Examples-

- If header length field contains decimal value 5 (represented as 0101), then-

Header length = 5 x 4 = 20 bytes

- If header length field contains decimal value 10 (represented as 1010), then-

Header length = 10 x 4 = 40 bytes

- If header length field contains decimal value 15 (represented as 1111), then-

Header length = 15 x 4 = 60 bytes

6. Reserved Bits-

- The 6 bits are reserved.
- These bits are not used.

7. URG Bit-

URG bit is used to treat certain data on an urgent basis.

When URG bit is set to 1,

- It indicates the receiver that certain amount of data within the current segment is urgent.
- Urgent data is pointed out by evaluating the urgent pointer field.
- The urgent data has be prioritized.
- Receiver forwards urgent data to the receiving application on a separate channel.

8. ACK Bit-

ACK bit indicates whether acknowledgement number field is valid or not.

- When ACK bit is set to 1, it indicates that acknowledgement number contained in the TCP header is valid.
- For all TCP segments except request segment, ACK bit is set to 1.
- Request segment is sent for connection establishment during **Three Way Handshake**.

9. PSH Bit-

PSH bit is used to push the entire buffer immediately to the receiving application.

When PSH bit is set to 1,

- All the segments in the buffer are immediately pushed to the receiving application.
- No wait is done for filling the entire buffer.
- This makes the entire buffer to free up immediately.

10. RST Bit-

RST bit is used to reset the TCP connection.

When RST bit is set to 1,

- It indicates the receiver to terminate the connection immediately.
- It causes both the sides to release the connection and all its resources abnormally.
- The transfer of data ceases in both the directions.
- It may result in the loss of data that is in transit.

This is used only when-

- There are unrecoverable errors.
- There is no chance of terminating the TCP connection normally.

11. SYN Bit-

SYN bit is used to synchronize the sequence numbers.

12. FIN Bit-

FIN bit is used to terminate the TCP connection.

When FIN bit is set to 1,

- It indicates the receiver that the sender wants to terminate the connection.
- FIN segment sent for **TCP Connection Termination** contains FIN bit set to 1.

13. Window Size-

- Window size is a 16 bit field.
- It contains the size of the receiving window of the sender.
- It advertises how much data (in bytes) the sender can receive without acknowledgement.
- Thus, window size is used for **Flow Control**.

14. Checksum-

- Checksum is a 16 bit field used for error control.
- It verifies the integrity of data in the TCP payload.
- Sender adds CRC checksum to the checksum field before sending the data.
- Receiver rejects the data that fails the CRC check.

15. Urgent Pointer-

- Urgent pointer is a 16 bit field.
- It indicates how much data in the current segment counting from the first data byte is urgent.
- Urgent pointer added to the sequence number indicates the end of urgent data byte.
- This field is considered valid and evaluated only if the URG bit is set to 1.

16. Options-

- Options field is used for several purposes.
- The size of options field vary from 0 bytes to 40 bytes.

UDP Protocol-

- UDP is short for **User Datagram Protocol**.

  It is the simplest transport layer protocol.
- It has been designed to send data packets over the Internet.
- It simply takes the datagram from the network layer, attaches its header and sends it to the user.

Characteristics of UDP-

- It is a connectionless protocol.
- It is a stateless protocol.
- It is an unreliable protocol.
- It is a fast protocol.
- It offers the minimal transport service.
- It is almost a null protocol.
- It does not guarantee in order delivery.
- It does not provide congestion control mechanism.
- It is a good protocol for data flowing in one direction.

Need of UDP-

- TCP proves to be an overhead for certain kinds of applications.
- The **Connection Establishment** Phase, **Connection Termination** Phase etc of TCP are time consuming.
- To avoid this overhead, certain applications which require fast speed and less overhead use UDP.

UDP Header-

The following diagram represents the UDP Header Format-

Source Port (2 bytes)	Destination Port (2 bytes)
Length (2 bytes)	Checksum (2 bytes)

UDP Header

1. Source Port-

- Source Port is a 16 bit field.
- It identifies the port of the sending application.

2. Destination Port-

- Destination Port is a 16 bit field.
- It identifies the port of the receiving application.

3. Length-

- Length is a 16 bit field.
- It identifies the combined length of UDP Header and Encapsulated data.

Length = Length of UDP Header + Length of encapsulated data

4. Checksum-

- **Checksum** is a 16 bit field used for error control.
- It is calculated on UDP Header, encapsulated data and IP pseudo header.
- Checksum calculation is not mandatory in UDP.

Note-01:

Size of UDP Header= 8 bytes

- Unlike TCP header, the size of UDP header is fixed.
- This is because in UDP header, all the fields are of definite size.
- Size of UDP Header = Sum of the size of all the fields = 8 bytes.

UDP is an unreliable protocol.

This is because-

- UDP does not guarantee the delivery of datagram to its respective user (application).
- The lost datagrams are not retransmitted by UDP.

Checksum calculation is not mandatory in UDP.

This is because-

- UDP is already an unreliable protocol and error checking does not make much sense.
- Also, time is saved and transmission becomes faster by avoiding to calculate it.

UDP does not guarantee in order delivery.

This is because-

- UDP allows out of order delivery to ensure better performance.
- If some data is lost on the way, it does not call for retransmission and keeps transmitting data.