TCP Connection

TCP is connection-oriented.

A connection-oriented transport protocol establishes a logical path between the source and destination. All of the segments belonging to a message are then sent over this logical path.

If a segment arrives out of order, TCP holds it until the missing segments arrive; IP is unaware of this reordering.

In TCP, connection-oriented transmission requires three phases: connection establishment,data transfer, and connection termination.

Connection Establishment

TCP transmits data in full-duplex mode. When two TCPs in two machines are connected,they are able to send segments to each other simultaneously. This implies that each party must initialize communication and get approval from the other party before any data are transferred.

Three-Way Handshaking

The connection establishment in TCP is called three-way handshaking.

For example,an application program, called the client, wants to make a connection with another application program, called the server, using TCP as the transport-layer protocol.

The process starts with the server. The server program tells its TCP that it is ready to accept a connection. This request is called a passive open.

The client program issues a request for an active open. A client that wishes to connect to an open server tells its TCP to connect to a particular server.

TCP function is enabled by three-way handshaking process, as shown in Figure 4.3.1 .

Three step process.

1.The client sends the first segment, a SYN segment, in which only the SYN flag is set.

This segment is for synchronization of sequence numbers.The client chooses a random number as the first sequence number and sends this number to the server. This sequence number is called the initial sequence number (ISN).

The SYN segment is a control segment and carries no data. It consumes one sequence number because it needs to be acknowledged. SYN segment carries one imaginary byte.

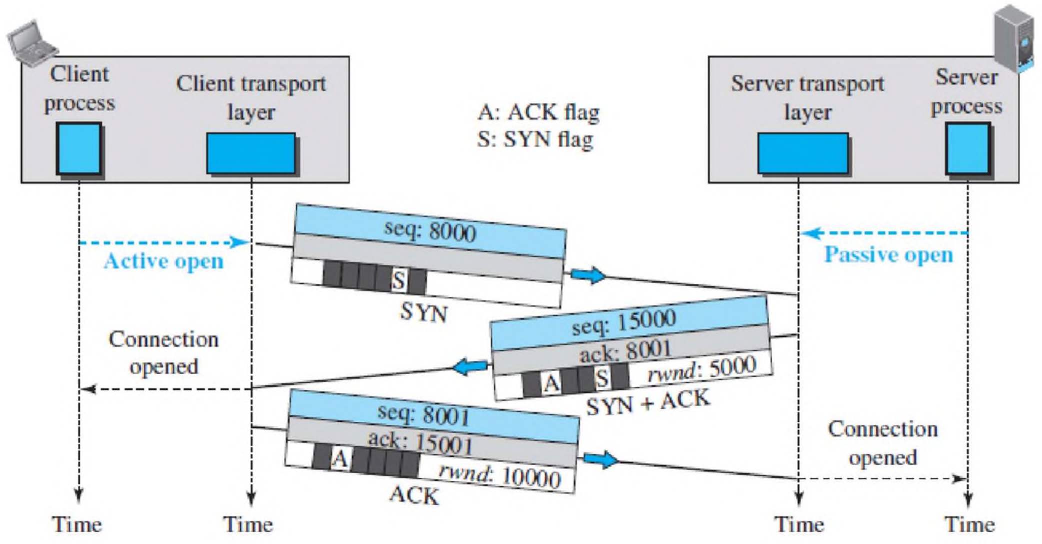

Fig4.3.1:TCP connection.

[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-751]

2.The server sends the second segment, a SYN + ACK segment with two flag bits set as: SYN and ACK. This segment has a dual purpose.

First, it is a SYN segment for communication in the other direction. The server uses this segment to initialize a sequence number for numbering the bytes sent from the server to the client.

The server also acknowledges the receipt of the SYN segment from the client by setting the ACK flag and displaying the next sequence number it expects to receive from the client.

3**.** The client sends the third ACK segment. It acknowledges the receipt of the second segment with the ACK flag and acknowledgment number field.

Note that the ACK segment does not consume any sequence numbers if it does not carry data.

Data Transfer

After connection is established, bidirectional data transfer take place. The client and server can send data and acknowledgments in both directions.The acknowledgment is piggybacked with the data. Figure 4.3.2 shows an example.

In this example, after a connection is established, the client sends 2,000 bytes ofdata in two segments. The server then sends 2,000 bytes in one segment.

The client sends one more segment. The first three segments carry both data and acknowledgment,but the last segment carries only an acknowledgment because there is no more data to be sent. Note the values of the sequence and acknowledgment numbers.

The data segments sent by the client have the PSH (push) flag set so that the server TCP knows to deliver data to the server process as soon as they are received.

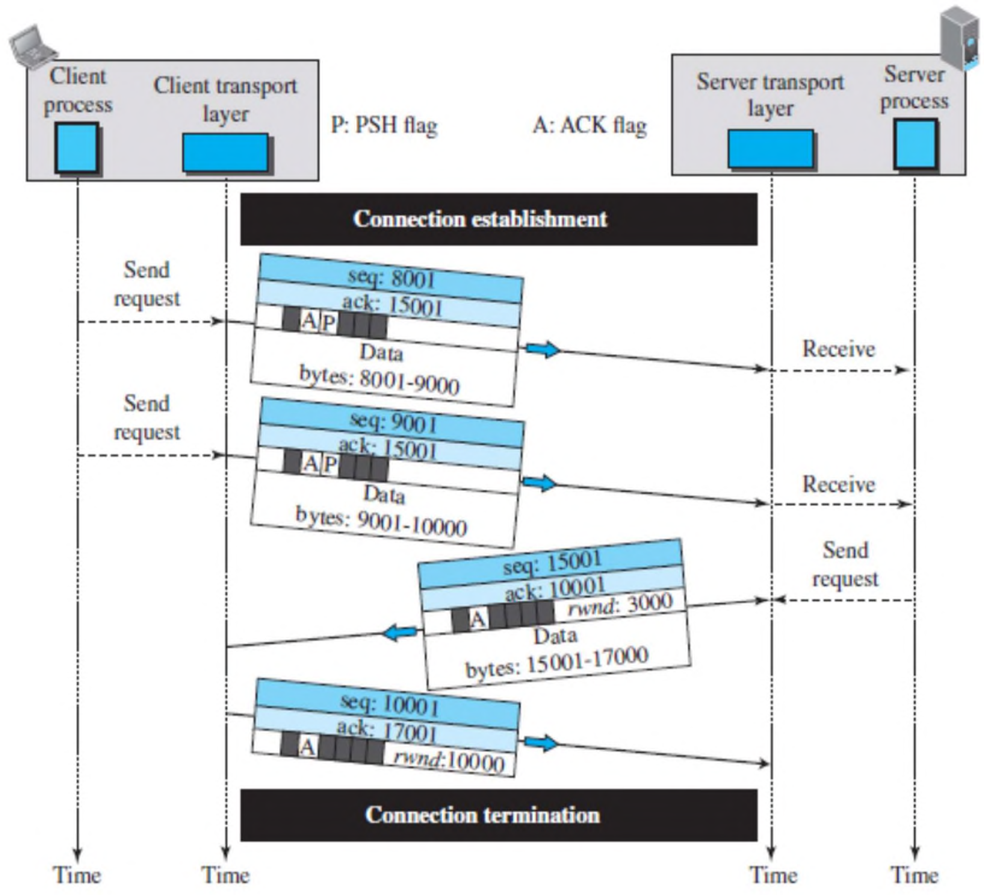

Fig4.3.2:Data Transfer.

[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-751]

TCP can handle push operation.

The application program at the sender can request a push operation. This means that the sending TCP must not wait for the window to be filled. It must create a segment and send it immediately.

The sending TCP set the push bit (PSH) to let the receiving TCP know that the segment includes data that must be delivered to the receiving application program as soon as possible and not to wait for more data to come.

Urgent Data

TCP is a stream-oriented protocol. TCP supports an application program which needs to send urgent bytes, some bytes that need to be treated in a special way by the application at the other end.

The sending application program tells the sending TCP that the piece of data is urgent. The sending TCP creates a segment and inserts the urgent data at the beginning of the segment. The rest of the segment can contain normal data from the buffer.

The urgent pointer field defines the end of the urgent data (the last byte of urgent data).

Connection Termination: Three-Way Handshaking

It has three phases as shown in figure 4.3.3.

1. In this situation, the client TCP, after receiving a close command from the client process, sends the first segment, a FIN segment in which the FIN flag is set. Note that a FIN segment can include the last chunk of data sent by the client or it can be just a control segment as shown in the figure. If it is only a control segment, it consumes only one sequence number because it needs to be acknowledged. The client TCP, after receiving a close command from the client process, sends the first segment called FIN segment.

The FIN segment has the last data sent by the client or it can be just a control segment.

2. The server TCP, after receiving the FIN segment, informs its process of the situation and sends the second segment, a FIN + ACK segment, to confirm the receipt of the FIN segment from the client and at the same time to announce the closing of the connection in the other direction. This segment can also contain the last chunk of data from the server. If it does not carry data, it consumes only one sequence number because it needs to be acknowledged.

3.The client TCP sends the last segment, an ACK segment, to confirm the receipt of the FIN segment from the TCP server. This segment contains the acknowledgment number, which is one plus the sequence number received in the FIN segment from the server. This segment cannot carry data and has no sequence numbers.

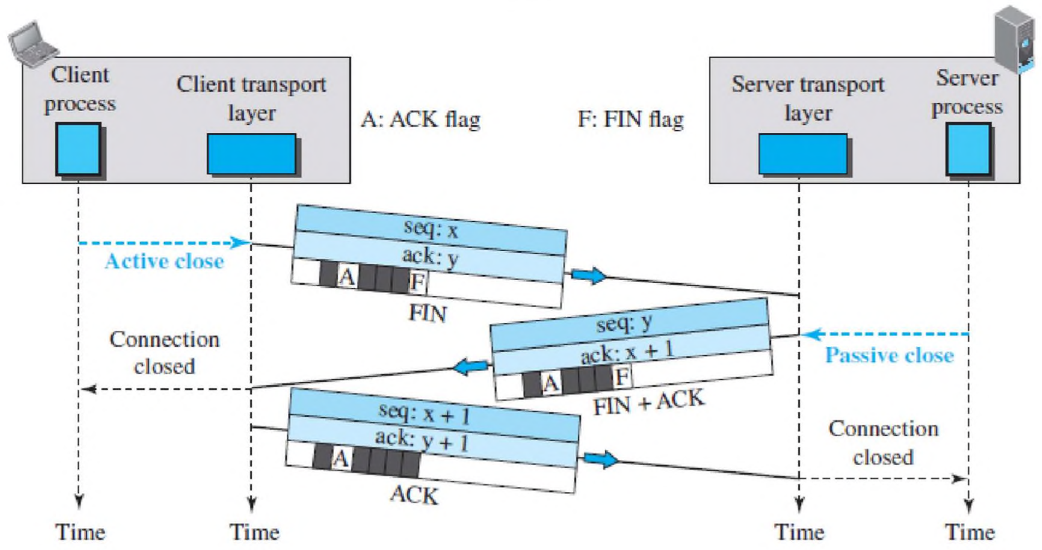

Fig4.3.3: Connection termination.
[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-713]

Half-Close

In TCP, one end can stop sending data while still receiving data. This is called a half- close. Either the server or the client can issue a half-close request. It can occur when the server needs all the data before processing can begin

Connection Reset

TCP at one end can oppose a connection request, or may terminate an idle connection. These are done with the RST (reset) flag.

4.4 STATE TRANSITION DIAGRAM

To observe the events happening during connection establishment, connection termination, and data transfer,TCP is specified as the finite state machine (FSM) as shown in Figure 4.4.1.

Here two FSMs used by the TCP client and server combined in one diagram.The rounded-corner rectangles represent the states. The transition from one state to another is shown using directed lines. Each line has two strings separated by a slash. The first string is the input, what TCP receives. The second is the output, what TCP sends.

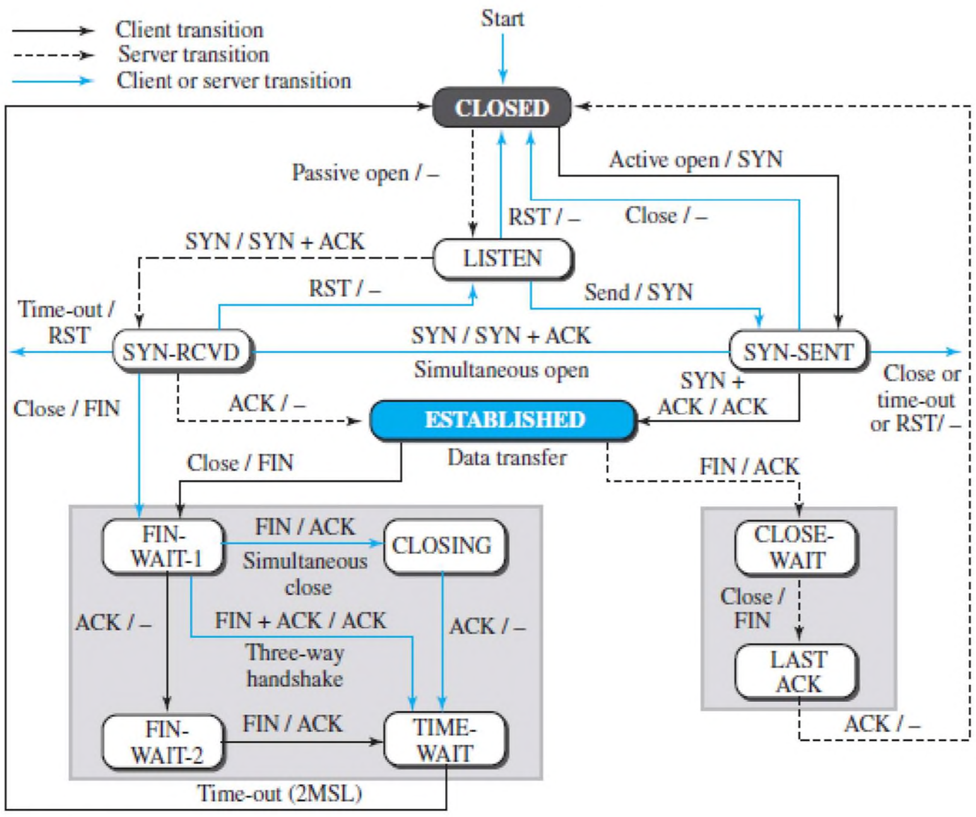

Fig4.4.1: State transition diagram.

[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-758]

.

The dotted black lines in the figure represent the transition that a server normally goes through; the solid black lines show the transitions that a client normally goes through.In some situations, a server transitions through a solid line or a client transitions through a dotted line. The colored lines show special situations. The rounded-corner rectangle marked ESTABLISHED has two sets of states, a set for the client and another for the server, that are used for flow and error control.

Consider the scenario. Figure 4.4.2 shows the state transition diagram for this scenario. The client process issues an active open command to its TCP to request a connection to a specific socket address.

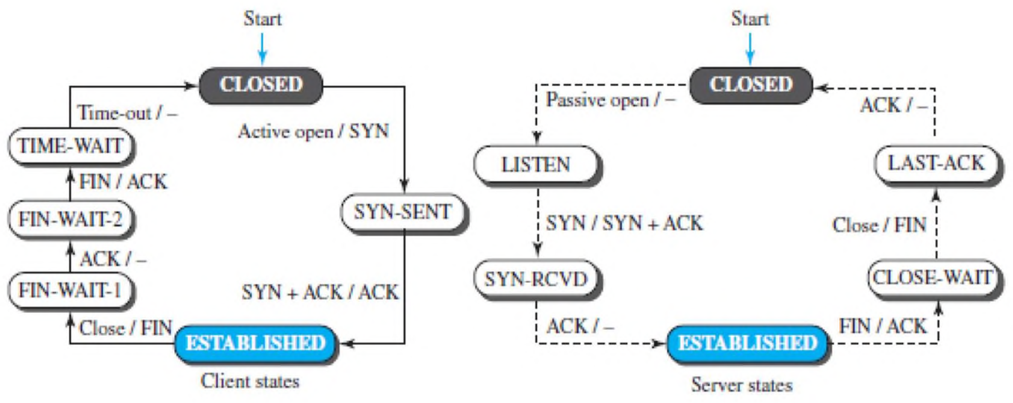

Fig4.4.2: State transition diagram.

[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-759]

TCP sends a SYN segment and moves to the SYN-SENT state. After receiving the SYN +ACK segment, TCP sends an ACK segment and goes to the ESTABLISHED state. Data are transferred, possibly in both directions, and acknowledged. When the client process has no more data to send, it issues a command called an active close. The TCP sends a FIN segment and goes to the FINWAIT-1 state. When it receives the ACK segment, it goes to the FIN-WAIT-2 state. When the client receives a FIN segment, it sends an ACK segment and goes to the TIME-WAIT state.The client remains in this state for 2 MSL .MSL is the maximum time a TCP segment is expected to live, or stay in the network.When the corresponding timer expires, the client goes to the CLOSED state.The server process issues a passive open command. The server TCP goes to the LISTEN state and remains there passively until it receives a SYN segment.

The TCP then sends a SYN +ACK segment and goes to the SYN-RCVD state, waiting for the client to send an ACK segment. After receiving the ACK segment, TCP goes to the ESTABLISHED state, where data transfer can take place. TCP remains in this state until it receives a FIN segment from the client signifying that there are no more data to be exchanged and the connection can be closed.The server, upon receiving the FIN segment, sends all queued data to the server with a virtual EOF marker, which means that the connection must be closed.

It sends an ACK segment and goes to the CLOSEWAIT state, but postpones acknowledging the FIN segment received from the client until it receives a passive close command from its process.

After receiving the passive close command, the server sends a FIN segment to the client and goes to the LASTACK state, waiting for the final ACK. When the ACK segment is received from the client,the server goes to the CLOSE state.