TCP (Transmission Control Protocol) provides a reliable end to end service that delivers packets over the Internet. Packets are delivered in sequence without loss or duplication.

This sequence diagram was generated with EventStudio System Designer (http://www.EventHelix.com/EventStudio).

This sequence diagram explores following: (1) The three-way handshake to establish a TCP (2) Data transfer using the byte oriented sequence numbers (3) Release of a TCP connection.

The TCP socket creation and deletion on the server and client is also covered.

---

**Client initiated three way handshake to establish a TCP connection**

**SYN**

*src = Client_Port, dst = Server_Port, seq_num = 0*

**SYN**

*src = Client_Port, dst = Server_Port, seq_num = 0*

**SYN+ACK**

*src = Server_Port, dst = Client_Port, seq_num = 100, ack_num = 1, window = 65535*

**ACK**

*src = Client_Port, dst = Server_Port, ack_num = 101, window = 5000*

**ACK**

*src = Client_Port, dst = Server_Port, ack_num = 101, window = 5000*

---

**Data transfer phase:** Here a short data transfer takes place, thus TCP slow start has little impact.
The first TCP segment is sent with a sequence number of 1. This is the sequence number for the first byte in the segment. (Note that unlike other protocols, TCP maintains sequence numbers at byte level. The sequence number field in the TCP header corresponds to the first byte in the segment.)

Bytes in the first TCP segment correspond to 1 to 512 sequence numbers. Thus, the second TCP segment contains data starting with 513 sequence number.

Server receives both the segments

Server acknowledges the data segments with the next expected sequence number as 1025 (TCP typically sends an acknowledgement every two received segments)

Client has received both the TCP segments

Client sends a TCP ACK with the next expected sequence number set to 701

Client closes TCP connection
**Client to server TCP connection release**

<table>
<thead>
<tr>
<th>Client</th>
<th>Net</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FIN</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>ACK</strong></td>
<td></td>
</tr>
</tbody>
</table>

- **Client sends a TCP segment with the FIN bit set in the TCP header**
- **Server receives the FIN**
- **Server responds back with ACK to acknowledge the FIN**
- **Client receives the ACK**

**Server to client TCP connection release**

<table>
<thead>
<tr>
<th>Client</th>
<th>Net</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FIN</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ACK</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **FIN is sent out to the client to close the connection**
- **Client receives FIN**
- **Client sends ACK**
- **Server receives the ACK**

This sequence diagram was generated with EventStudio System Designer (http://www.EventHelix.com/EventStudio).