

Intro

TCP (Transmission Control Protocol) - Transport layer protocol. It is a reliable, binary, stream based protocol.

It is connection oriented - it has official “start” (3-way handshake) and “end” (4-way handshake).

It is reliable - has confirmation of data delivery (ACK packets). Data bytes arrive in order they were sent. TCP also has packet retransmission.

It has flow control - TCP adjusts transmission rate (receiver can tell sender to slow down sending data or to speed up)

Sequence and Acknowledgement numbers

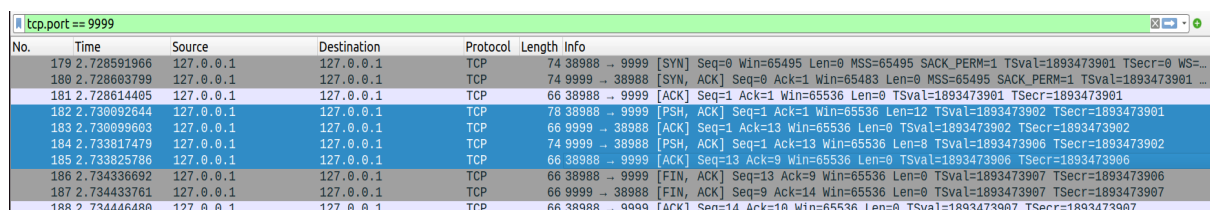
In short, sequence numbers (SEQ) track what has been sent and acknowledgement numbers (ACK) track what has been received. Sequence/Acknowledgement numbers are a measure of **bytes** sent/received.

Initial sequence numbers (ISN) are chosen semi-randomly by both parties and are exchanged during connection establishment (3-way handshake).

$$\begin{aligned}\text{ACK}(n) &= \text{SEQ}(n) + L(n) \\ \text{SEQ}(n+1) &= \text{ACK}(n)\end{aligned}$$

Here n is the n -th sent packet number. $L(n)$ is the length of the n -th sent TCP packet. These equations are true during the data transmission stage, when the connection was successfully established and before connection is terminated.

SEQ number is also a label/number of the first byte in current packet's payload and ACK is a label of the first byte in next packet's payload (Which means $\text{ACK} - 1$ is the label of the last byte inside current payload).



No.	Time	Source	Destination	Protocol	Length	Info
179	2.728591966	127.0.0.1	127.0.0.1	TCP	74	38988 → 9999 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=1893473901 TSecr=0 WS=...
180	2.728603799	127.0.0.1	127.0.0.1	TCP	74	9999 → 38988 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=1893473901 ...
181	2.728614405	127.0.0.1	127.0.0.1	TCP	66	38988 → 9999 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1893473901 TSecr=1893473901
182	2.730092644	127.0.0.1	127.0.0.1	TCP	78	38988 → 9999 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=12 TSval=1893473902 TSecr=1893473901
183	2.730099603	127.0.0.1	127.0.0.1	TCP	66	9999 → 38988 [ACK] Seq=1 Ack=13 Win=65536 Len=0 TSval=1893473902 TSecr=1893473902
184	2.733817479	127.0.0.1	127.0.0.1	TCP	74	9999 → 38988 [PSH, ACK] Seq=1 Ack=13 Win=65536 Len=8 TSval=1893473906 TSecr=1893473902
185	2.733825786	127.0.0.1	127.0.0.1	TCP	66	38988 → 9999 [ACK] Seq=13 Ack=9 Win=65536 Len=0 TSval=1893473906 TSecr=1893473906
186	2.734336692	127.0.0.1	127.0.0.1	TCP	66	38988 → 9999 [FIN, ACK] Seq=13 Ack=9 Win=65536 Len=0 TSval=1893473907 TSecr=1893473906
187	2.734433761	127.0.0.1	127.0.0.1	TCP	66	9999 → 38988 [FIN, ACK] Seq=9 Ack=14 Win=65536 Len=0 TSval=1893473907 TSecr=1893473907
188	2.734446488	127.0.0.1	127.0.0.1	TCP	66	38988 → 9999 [ACK] Seq=14 Ack=10 Win=65536 Len=0 TSval=1893473907 TSecr=1893473907

An example of TCP communication (fig. 1)

In fig.1 There are actual examples of both TCP communication “start” and “end” as well as data transmission (blue lines). Port 38988 is a client that has sent 12 bytes of data and port

9999 is a server that received the 12 bytes and responded with sending 8 bytes of data back to the client.

ACK number is just an index of the next expected byte in the incoming byte stream/array. It is the index of byte that the receiver expects from the sender to send first in the next stream chunk/packet.

SEQ number is just an index of the first byte in the stream/array chunk that is being sent. It has to match the last received ACK number from the receiver, because the receiver expects exactly that chunk.

So, basically, sender must send the chunk that the receiver expects

Window size

Limits how much unacknowledged data can be sent. The window size is sent in each packet (there is a field for window size in TCP header). Can be dynamically updated (Flow Control)

Questions

Q: What's the point of [PSH,ACK] ?

A: PSH flag - The push flag is set by the sending host and indicates to the receiving host that the received data should be passed to the receiving application immediately. TCP stack implementation controls whether to set this flag or not

Sometimes, different TCP implementations will buffer several packets together before sending them up to the application layer, so PSH flag sends them to the application

Q: Why is ACK always set?

References:

- <https://networkengineering.stackexchange.com/questions/29823/why-is-tcp-acknowledging-all-the-time#:~:text=The%20ACK%20bit%20is%20continuously.is%20used%20to%20do%20so>.
- <https://serverfault.com/questions/928642/all-tcp-packets-have-the-psh-flag-set-who-would-be-responsible-for-that>
- <https://serverfault.com/questions/124517/what-is-the-difference-between-unix-sockets-and-tcp-ip-sockets?rq=1>