**CSCI 367 - Computer Networks I**

TCP/IP Sockets – TCP – SEQ/ACK Field

**Sequence and Acknowledgment Numbers**

<https://packetlife.net/blog/2010/jun/7/understanding-tcp-sequence-acknowledgment-numbers/>

The client on either side of a TCP session maintains a 32-bit *sequence number* it uses to keep track of how much data it has sent. This sequence number is included on each transmitted packet and acknowledged by the opposite host as an *acknowledgement number* to inform the sending host that the transmitted data was received successfully.

When a host initiates a TCP session, its initial sequence number is effectively random; it may be any value between 0 and 4,294,967,295, inclusive. However, protocol analyzers like Wireshark will typically display *relative* sequence and acknowledgement numbers in place of the actual values. These numbers are relative to the initial sequence number of that stream. This is handy, as it is much easier to keep track of relatively small, predictable numbers rather than the actual numbers sent on the wire.

<https://madpackets.com/2018/04/25/tcp-sequence-and-acknowledgement-numbers-explained/>

TCP Sequence (seq) and Acknowledgement (ack) numbers help enable ordered reliable data transfer for TCP streams.  The seq number is sent by the TCP client, indicating how much data has been sent for the session (also known as the byte-order number). The ack number is sent by the TCP server, indicating that is has received cumulated data and is ready for the next segment.

The TCP seq and ack numbers are coordinated with one another and are key values during the [TCP handshake](https://madpackets.com/2018/04/10/tcp_handshake/), [TCP close](https://madpackets.com/2018/04/17/tcp-connection-close-what-does-it-look-like-in-wireshark/), and, of course, while data is transferred between the client and server.

[By default, Wireshark converts all sequence and acknowledgement numbers into [relative numbers](https://wiki.wireshark.org/TCP_Relative_Sequence_Numbers). This means that all SEQ and ACK numbers always start at 0 for the first packet seen in each conversation.]

**References**

https://packetlife.net/blog/2010/jun/7/understanding-tcp-sequence-acknowledgment-numbers/

https://madpackets.com/2018/04/25/tcp-sequence-and-acknowledgement-numbers-explained/

**Packet Captures**: TCP\_SEQ\_ACK\_A.pcapng

**Source Code Listing:** 11\_A\_TCP\_SEQ\_ACK

**Compilation:** gcc -g -I../../Libraries -o Program \*.c ../../Libraries/Utilities.c

Server Sends First Byte: Server Sequence Number: 1

Graphical user interface, text, application, email

Description automatically generated

Client Receives First Byte and Acknowledges 1 Byte: Client Acknowledgement Number: 2

The client acknowledgement number represents the number of the next byte expected from the server.

Graphical user interface, text, application, email

Description automatically generated

Server Sends Second Byte: Server Sequence Number: 2

Graphical user interface, text, application, email

Description automatically generated

Client Receives Second Byte and Acknowledges 2 Byte: Client Acknowledgement Number: 3

The client acknowledgement number represents the number of the next byte expected from the server.

Graphical user interface, text, application, email

Description automatically generated