TCP Analysis

Elements of Packet Sniffing

Informationssicherheit & Datenschutz

1. COMMON TCP OPTIONS

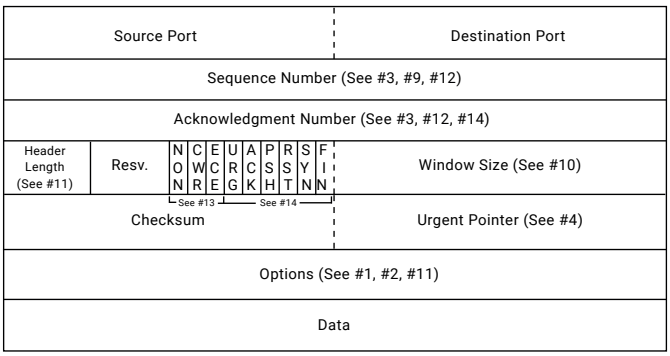
KIND	LEN.	OPTION
1	-	NO-OPERATION [RFC793] (SEE **#11**)
2	4	MAXIMUM SEGMENT SIZE [RFC793] (SEE #2)
3	3	WINDOW SCALE [RFC7323] (SEE #10)
4	2	SACK PERMITTED [RFC2018] (SEE #14)
5	N	SACK [RFC2018](SEE #13) (SEE #14)
8	10	TIMESTAMPS [RFC7323]

2. MSS OPTION

The maximum segment size (MSS) is the most commonly seen option in the TCP handshake. The MSS is the number of BYTES that a host can receive in a TCP data segment (which does not include the TCP header).
The MSS option is not a negotiation - it is a state - meant of what each host can receive - each side can support different values. If the MSS option is missing, the MSS value 536 is used.

3. RELATIVE SEQ. NUMBERING

Wireshark uses relativ sequence numbers by default. Rather than starting with the true 4-BYTE sequence number value (assigned by TCP peers in SYN and SYN/ACK packets).
Wireshark assigns sequence number 0 to the SYN and SYN/ACK packets (SEE #6 AND #71) Wireshark uses a relative acknowledgment number field value (for all packets after the SYN packet) (SEE #7 AND #81) relative numbering begins at 1 if the SYN-SYN/ACK aren´t seen.



4. TCP FLAGS

URGENT (URG) tcp.flags.urg==1
Urgend pointer field is in ise: Look at the urgent Data. (rare)

ACKNOWLEDGE (ACK) tcp.flags.ack==1
Acknowledgment of incoming DATA/TCP connection control frames (SYN, FIN, RESET). Indicates the acknowledgment number field is significant.

PUSH (PSH) tcp.flags.push==1
Set by an application, the TCP transmit buffer must flush all buffered data onto the network. RFC 793 states that incoming data with the push bit set should be sent directly to the receiving application (rarely used).

RESET (RST) top.flags.reset==1
Used to refuse a TCP connection or terminate an active connection.

SYNCHRONIZE (SYN) tcp.flags.syn==1
Sender is providing its initial sequence number (ISN) for a new connection.

FINISH (FIN) tcp.flags.fin==1
Indicates a sender will send no more data, but it may still receive data. When a fin is received, the connection can close.

5. HOT WIRESHARK FILTERS

tcp.flags.syn==1	SYNS AND SYN/ACKS	tcp.analysis.flags	ALL TCP FLAG ITEMS
tcp.flags.reset==1	TCP RESETS	tcp.analysis.retransmission	ALL TRANSMISSIONS
tcp.flags.urg==1	URGENT BIT SET	tcp.port==x	TCP TO/FROM PORT X
tcp.window_size<x	CALC. WINSIZE < THAN X	tcp.analysis.ack.rtt>x	ROUNDTRIPS > X
tcp.stream==x	TCP CONVERSATION X	tcp.window_size_scalefactor==-2	NO WINDOW SCALING

6. TCP SYN HEADER

tcp.flags.syn==1 && tcp.flags.ack==0
Source port: 60223
Destination port: 80
[STREAM INDEX: 1] (SEE #5)
[TCP SEGMENT LEN: 0]
Sequence number: 0 (Relative SEQ#) (SEE #3, #12)
(NEXT SEQUENCE NUM.: 0 (Relative SEQ#)(SEE #3, #12)
Acknowledgment number: 0 (SEE #3, #12)
1000 .... = Header length: 32 BYTES (8) (SEE #11)
Flags: 0X002 (SYN) (SEE #4)
000. .... = RESERVED: not set
...0 .... = NONCE: not set (SEE #13)
.... 0... .... = CWR: not set (SEE #13)
.... .0.. .... = ECN-ECHO: not set (SEE #13)
.... ..0. .... = URGENT: not set (SEE #4)
....0 .... = ACKNOWLEDGMENT:not set (SEE #4)
.... 0... = PUSH: not set (SEE #4)
.... .0.. = RESET: not set (SEE #4)
....1. = SYN: Set (SEE #4)
....0 = FIN: not set (SEE #4)
[TCP flags: ---------S-] (SEE #4)

Window size value: 8192 (SEE #10)
[Calculated window size: 8192] (SEE #10)
Checksum: 0X64FC
Urgent pointer: 0 (SEE #4)

Options: (SEE #1)
Max. seg. size: 1460 BYTES (SEE #2)
No-operation (NOP) (SEE #11)
Window scale: 8 (Multiply by 256) (SEE #10)
No-operation (NOP) (SEE #11)
No-operation (NOP) (SEE #11)

7. TCP SYN/ACK HEADER

tcp.flags.syn==1 && tcp.flags.ack==0
Source port: 80
Destination port: 60223
[STREAM INDEX: 1] (SEE #5)
[TCP SEGMENT LEN: 0]
Sequence number: 0 (Relative SEQ#) (SEE #3, #12)
[NEXT SEQUENCE NUM.: 0 (Relative SEQ#)] (SEE #3, #12)
Acknowledgment number: 1 (Relative ACK#) (SEE #3, #12)
1000 .... = Header length: 32 BYTES (8) (SEE #11)
Flags: 0X012 (SYN) (SEE #4)
000. .... = RESERVED: not set
...0 .... = NONCE: not set (SEE #13)
.... 0... .... = CWR: not set (SEE #13)
.... .0.. .... = ECN-ECHO: not set (SEE #13)
.... ..0. .... = URGENT: not set (SEE #4)
....0 .... = ACKNOWLEDGMENT: not set (SEE #4)
.... 0... = PUSH: not set (SEE #4)
.... .0.. = RESET: not set (SEE #4)
....1. = SYN: Set (SEE #4)
....0 = FIN: not set (SEE #4)
[TCP flags: -------A--S-] (SEE #4)

Window size value: 65535 (SEE #10)
[Calculated window size: 65535] (SEE #10)

Checksum: 0X8686
Urgent pointer: 0 (SEE #4)

Options: (SEE #1)
Max. seg. size: 1460 BYTES (SEE #2)
No-operation (NOP) (SEE #11)
No-operation (NOP) (SEE #11)
SACK permitted (SEE #14)
No-operation (NOP) (SEE #11)
Window scale: 9 (Multiply by 512) (SEE #10)

[SEQ/ACK Analysis]
[This is an ACK to the segment in frame: 59](SEE #12)
[The RTT to ACK the segment was: 0.010912 SECS.]
[IRTT: 0.011071 seconds] (SEE #15)

8. TCP ACK HEADER

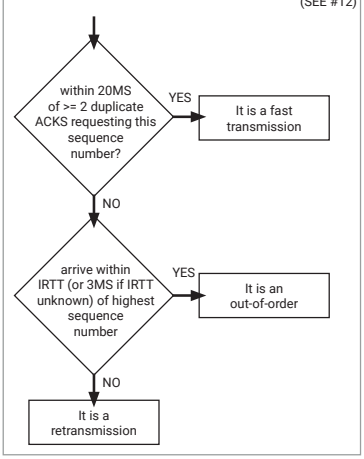
Source port: 60223
Destination port: 80
[STREAM INDEX: 1] (SEE #5)
[TCP SEGMENT LEN: 0]
Sequence number: 0 (Relative SEQ#) (SEE #3, #12)
[NEXT SEQUENCE NUM.: 1 (Relative SEQ#)] (SEE #3, #12)
Acknowledgment number: 1 (Relative ACK#)(SEE #3, #12)
0101 .... = Header length: 20 BYTES (5) (SEE #11)
Flags: 0X010 (SYN) (SEE #4)
000. .... = RESERVED: not set
...0 .... = NONCE: not set (SEE #13)
.... 0... .... = CWR: not set (SEE #13)
.... .0.. .... = ECN-ECHO: not set (SEE #13)
.... ..0. .... = URGENT: not set (SEE #4)
....1 .... = ACKNOWLEDGMENT: set (SEE #4)
.... 0... = PUSH: not set (SEE #4)
.... .0.. = RESET: not set (SEE #4)
....0. = SYN: not set (SEE #4)
....0 = FIN: not set (SEE #4)
[TCP flags: -------A----] (SEE #4)

Window size value: 256 (SEE #10)
[Calculated window size: 65536] (SEE #10)
[Window size scaling factor: 256] (Multiplier) (SEE #10)
Checksum: 0XF68A
Urgent pointer: 0 (SEE #4)

[SEQ/ACK Analysis]
[This is an ACK to the segment in frame: 60] (SEE #12)
[The RTT to ACK the segment was: 0.000159 SECS.]
[IRTT: 0.011071 seconds] (SEE #15)

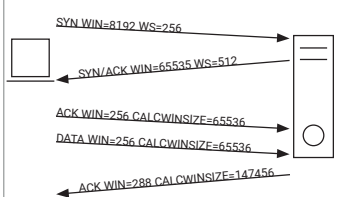
9. RETRANSMISSIONS VS. OUT-OF-ORDERS

Frame contains data (or is a SYN or FIN) and doesn´t advance the sequence number (SEE #12)



10. WINDOW SCALING

To advertise larger receive buffer sizes, the window size value (reveice buffer size) is multiplied by the scaling factor to create a calculated (true) window size. Both TCP peers must support window scaling to use this



11. PADDING OPTIONS

The NO-OPERATION (NOP) option can be used to pad TCP options to ensure they start on a word (2-BYTE) boundary.

TCP headers must end on a 4-BYTE boundary. The minimum TCP headers is 20 BYTES (without any options).

This 4-BYTE boundary is required because the TCP header length field value is only 4 BITS long. The TCP header lenght field value is multiplied by 4 to obtain the final TCP header length.

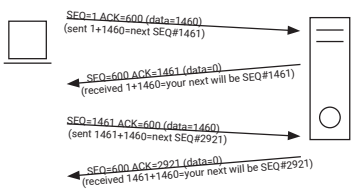
12. SEQUENCE AND ACKNOWLEDGMENT NUMBERING

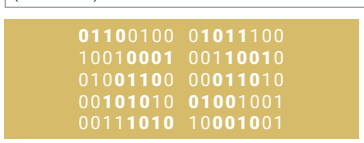
The sequence and acknoledgment number fields are used to detect and recover from packet loss and identify out-of-order (000) packets (SEE #9).

The sequence number increments by the number of data BYTES sent. The acknowledgment number indicates the next expected sequence number from the TCP Peer.

At right, the client is sending data, so only the client´s sequence number field increments.

The server´s acknowledgment number field increments to indicate the next expected sequence number from the client.
(SEE also #14)





13. NONCE, CWR, ECN-ECHO

NONCE: Explicit congestion notification (ECN) concealment protection SEE RFC 3540

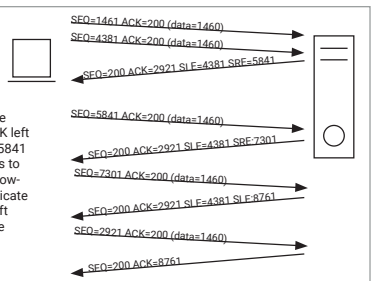
CWR: Indicates the sender has reduced the congestion window (congestion window reduced) SEE RFC 3168

ECE: ECN-ECHO (ECE) set in ACKS to congestion experienced (CE) packets. SEE RFC 3168.

14. SELECTIVE ACKNOWLEDGMENTS

Selective acknowledgment (SACK) ensures that only the missing packets are retransmitted when packet loss occurs. Both TCP peers must indicate they support SACK in the TCP handshake for it to be used.

At right, sequence numbers 2921-4380 went missing. The server indicates it received sequence number 4381 (SACK left edge, or SLE) up to, but not including, sequence number 5841 (SACK right edge, or SRE). The SACK right edge increases to acknowledge additional data packets received. The acknowledgment number from the server remains at 2921 to indicate the start of the missing sequence numbers. The SACK left edge/right option is removed when the missing sequence numbers are received.

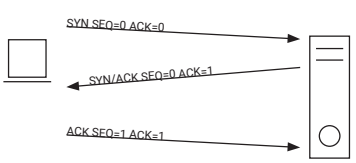


15. IRTT CALCULATION

The inital round trip time (IRTT) calculation is based on the time from the first (SYN) to the third (ACK) packet of TCP handshakes. This provides for a full round trip time measurement.

The IRTT value provides the base path latency between peers.

If Wireshark sees the full TCP handshake, it uses this value to differentiate between retransmissions and out-of-order packets. (SEE #9)



S-COP GmbH
+49 (0)8022 7058 185
Rathausplatz 5
datenschutz@s-cop.bayern
83684 Tegernsee
https://www.s-cop.bayern

Inspired by Chappell University