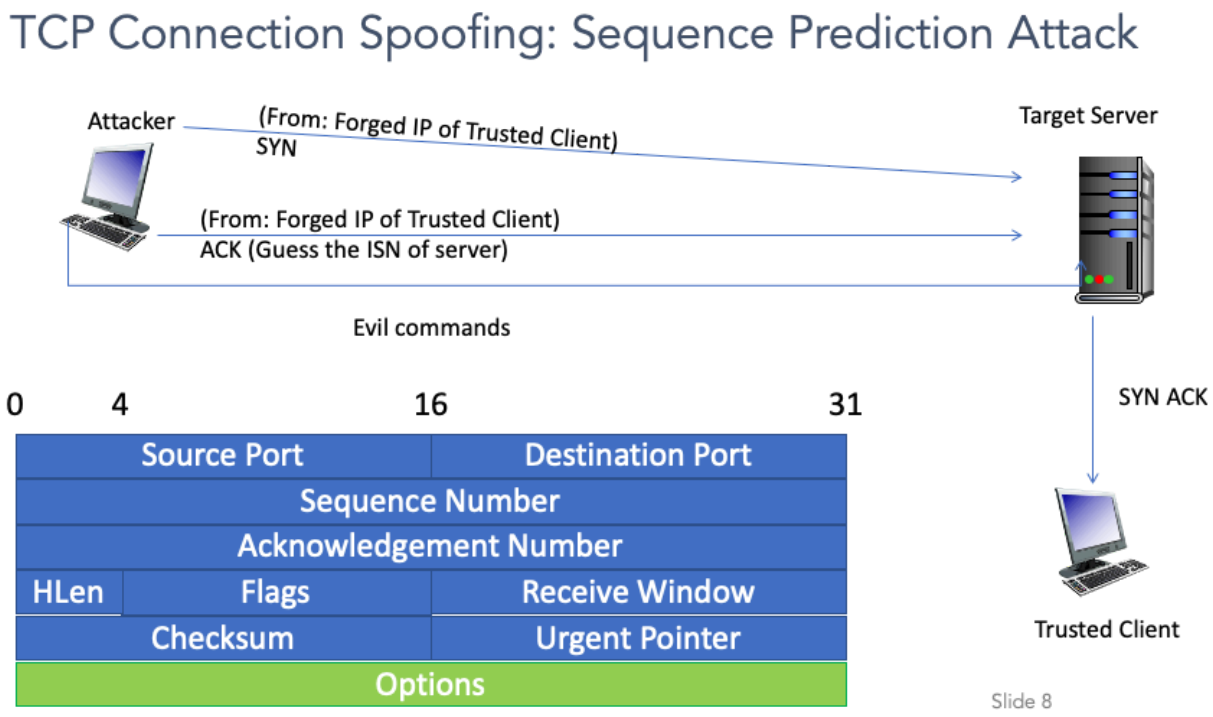
Week 10: Transport Layer Security

TCP Security

Q1. How should we choose the initial sequence number? (Hint: What can go wrong with choosing a particular sequence number to start from every time? How can this information be used for malicious reasons?)

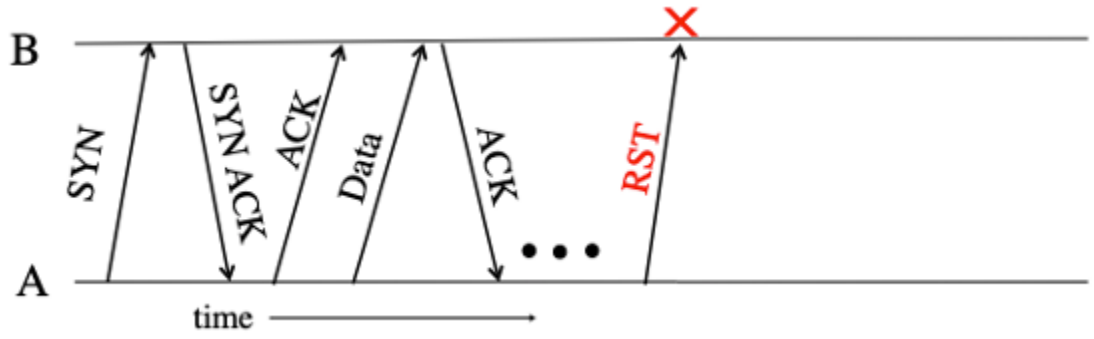
A) Start from zero
B) Start from one
C) Start from a random number
D) Start from some other value (such as…?)



Q2. What is the probability of success for an off-path attacker to guess the right sequence number given the TCP header information above?

A. 1
B. 0.5
C. 1 in 2^32
D. 1 in 2^16

Q3. TCP Resets: Part A: To **abort** a connection, one side sends a packet with the RST (reset) flag set: This means "I will no longer be sending nor receiving packets on this connection" and importantly, RST packets are not acknowledged (!) since they usually mean that something went wrong.



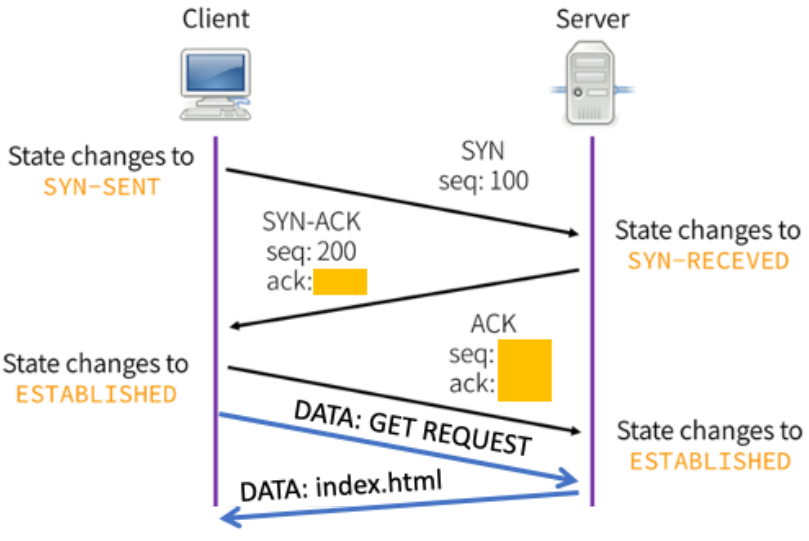
What harm can an attacker do with the TCP RST Injection attack?

A. The attacker can stop an on-going TCP connection
B. The attacker can divert an on-going TCP connection
C. The attacker can launch a denial of service attack on a TCP connection.
D. The attacker can launch a replay attack on a TCP connection

Part B: Who can launch a RST injection attack?

A. An off-path attacker
B. An on-path attacker
C. MiTM

Q4. What other forms of TCP attacks could you launch (choose from the list below) and who can launch these attacks (On-path, off-path, MiTM)

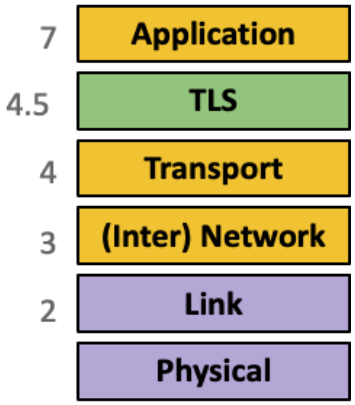


A. TCP Data Injection: Tampering with an existing session to modify or inject data into a connection
B. TCP Spoofing.
C. TCP SYN Flooding

Q5. What guarantees does TCP provide?

A. Confidentiality
B. Availability
C. Integrity
D. Reliable delivery of packets
E. Ordered delivery of packets
F. None of the above

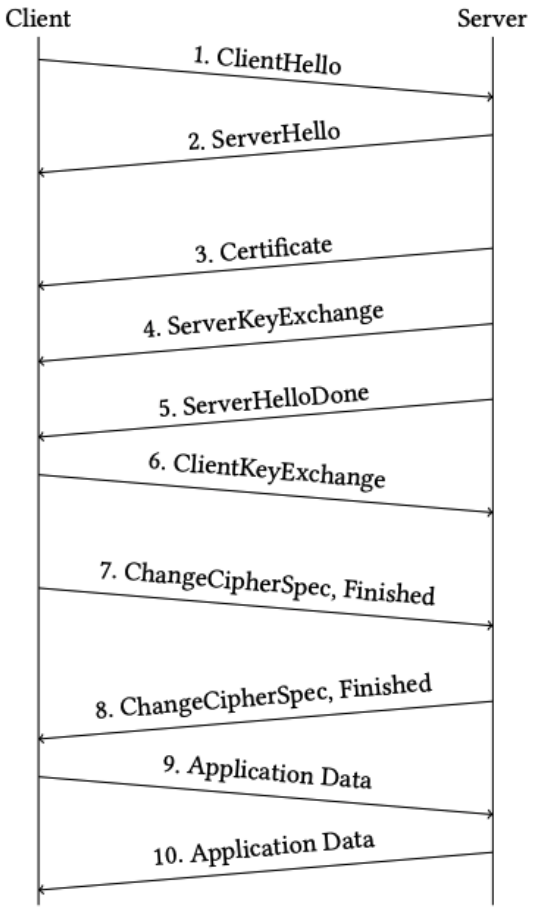
Transport Layer Security: TLS



- TLS is a protocol for creating a secure communication channel over the Internet.
- It is built on top of TCP and **relies** on TCP's byte-stream abstraction.

Goals of TLS:

A. **Confidentiality**: Ensure that attackers cannot read your traffic
B. **Integrity**: Ensure that attackers cannot tamper with your traffic
    a. Prevent **replay attacks**
C. **Authenticity**: Make sure you're talking to the legitimate server
    a. Defend against an attacker impersonating the server



1. Client sends 256-bit random number R_b and supported ciphers
2. Server sends 256-bit random number R_s and chosen cipher
3. Server sends certificate
4. DH: Server sends $\{g, p, g^a \bmod p\}_{K_{\text{server}}^{-1}}$
5. Server signals end of handshake
6. DH: Client sends $g^b \bmod p$
   RSA: Client sends $\{PS\}_{K_{\text{server}}}$
   Client and server derive cipher keys C_b, C_s and integrity keys I_b, I_s from R_b, R_s, PS
7. Client sends MAC(dialog, I_b)
8. Server sends MAC(dialog, I_s)
9. Client data takes the form $\{M_1, \text{MAC}(M_1, I_b)\}_{C_b}$
10. Server data takes the form $\{M_2, \text{MAC}(M_2, I_s)\}_{C_s}$

Q1. How can we be sure we are talking to the legitimate server?

A. Server sent their Diffie-Helman Exchange
B. Server sent its public key
C. Server proved that it owns the private key

Q2. What is the purpose of the client random and server random fields?

A. No purpose
B. They act as nonces to prevent replay attacks
C. They ensure validation of the two endpoints.

Q3. ClientHello and ServerHello are not encrypted or authenticated. Explain why a man-in-the-middle cannot exploit this. (Consider both the Diffie-Hellman and RSA case.)