Peyrin Kao
Fall 2023

CS 161 Computer Security

Exam Prep 11

Q1 *I (T)C(P) You (su20-final-q7)* **(26 points)**

EvanBot builds a new course feature that sends announcements to students over TCP. To receive announcements, a student initiates a TCP connection with the server. The server sends the announcements and terminates the connection.

Q1.1 (3 points) Assuming that no adversaries are present, which of the following does communication over a TCP connection guarantee? Select all that apply.

■ (A) That both the server and client can detect if a particular announcement needs to be resent

■ (B) That different announcements are delivered in the same order they were sent in

☐ (C) That announcements are delivered using the most efficient path through the internet

☐ (D) None of the above

☐ (E) ——

☐ (F) ——

Solution: TCP guarantees that messages will be retransmitted until they are successfully delivered, and that messages will be delivered in the correct order. TCP makes no guarantees about what path a packet takes through the Internet.

Q1.2 (3 points) When only an on-path adversary is present, which of the following does communication over a TCP connection guarantee? Select all that apply.

☐ (G) That both the server and client can detect if a particular announcement needs to be resent

☐ (H) That different announcements are delivered in the same order they were sent in

☐ (I) That announcements are delivered using the most efficient path through the internet

■ (J) None of the above

☐ (K) ——

☐ (L) ——

Solution: An on-path attacker has access to the TCP sequence numbers, so they can inject arbitrary messages. Since the attacker can interfere with all messages, TCP no longer has any guarantees about message delivery. TCP still makes no guarantees about what path a packet takes through the Internet.

Q1.3 (3 points) Suppose that EvanBot instead sends announcements over UDP. Assuming that no adversaries are present, which of the following might happen? Select all that apply.

■ (A) Students might not receive some announcements

■ (B) Students might receive the announcements more quickly

■ (C) The server might not detect some errors which it would have had it been using TCP

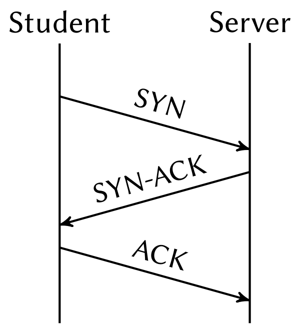
☐ (D) None of the above

☐ (E) ——

☐ (F) ——

Solution: UDP no longer guarantees delivery, so some announcements might not be delivered. However, UDP does not require a handshake at the beginning, so announcements can be delivered more quickly. UDP has no guarantees about what order announcements arrive in, so the server will no longer detect if packets arrive out of order.

EvanBot realizes that the server is sending messages to the student, but the student only responds with ACKs and never sends any messages after the initial handshake. They design a *Half TCP* protocol which provides TCP's properties for communications from the server to the student, but not for communications from the student to the server. This is accomplished using a modified version of the standard three step handshake pictured below.



Q1.4 (5 points) Some sequence numbers are no longer necessary in *Half TCP*. Which fields **do not** need to be transmitted? Select all that apply.

■ (G) The sequence number in the SYN packet

□ (H) The sequence number in the SYN-ACK packet

■ (I) The ACK number in the SYN-ACK packet

■ (J) The sequence number in the ACK packet

□ (K) The ACK number in the ACK packet

□ (L) None of the above

Solution: The key insight here is that because the student isn't sending messages to the server, the student's sequence numbers are no longer necessary. The SYN and ACK packets are sent from the student to the server, so their sequence numbers are no longer necessary. The SYN-ACK packet is sent from the server to the student, so its ACK number is no longer necessary.

An earlier version of the solutions incorrectly marked H, K as the set of correct answers. When revising the exam, we changed the question to be "which fields **do not** need to be transmitted," which caused the set of correct answers to be inverted.

Q1.5 (3 points) Which of these are consequences of moving from TCP to *Half TCP* for this application? Select all that apply.

□ (A) The student will no longer receive announcements in the correct order

■ (B) The server will not have to keep track of as much state

■ (C) The student will not have to keep track of as much state

□ (D) None of the above

□ (E) ——

□ (F) ——

Solution: Announcements are sent from the server to the student. We are still using sequence numbers in this direction, so the announcements are still received in the correct order. Because the server and student each only need to keep track of one sequence number instead of two, they both do not need to keep track of as much state.

The 161 staff likes security and decides to use TLS over *Half TCP*. Assume that the staff server has a valid certificate for their public key.

For each different adversary below, select all attacks which become ***easier*** when running TLS over *Half TCP* compared to normal TCP.

Q1.6 (3 points) Off-path adversary

- ■ (G) RST Injection Attack
- ☐ (H) Interfere with a TLS handshake to learn the master key
- ☐ (I) Replay an encrypted command from a previous TLS connection
- ☐ (J) None of the above
- ☐ (K) ——
- ☐ (L) ——

Q1.7 (3 points) On-path adversary

- ☐ (A) RST Injection Attack
- ☐ (B) Interfere with a TLS handshake to learn the master key
- ☐ (C) Replay an encrypted command from a previous TLS connection
- ■ (D) None of the above
- ☐ (E) ——
- ☐ (F) ——

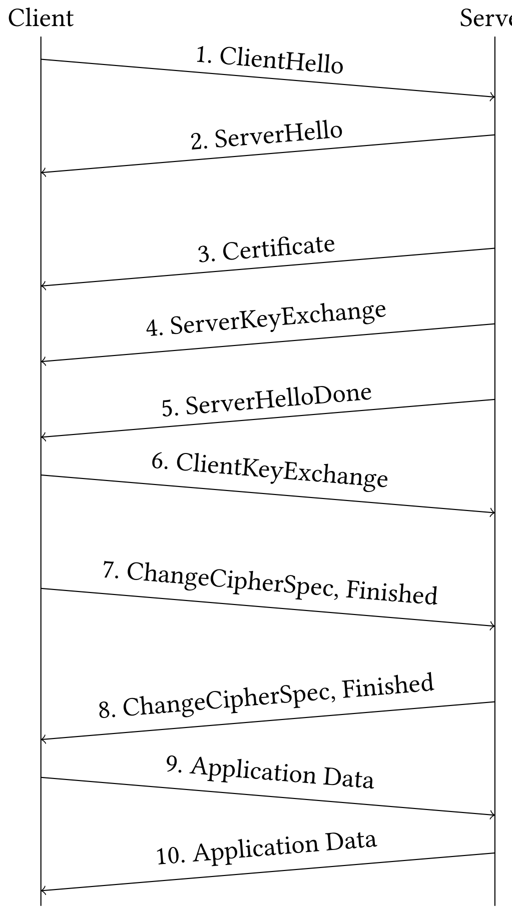
Q1.8 (3 points) Man-in-the-middle adversary

- ☐ (G) RST Injection Attack
- ☐ (H) Interfere with a TLS handshake to learn the master key
- ☐ (I) Replay an encrypted command from a previous TLS connection
- ■ (J) None of the above
- ☐ (K) ——
- ☐ (L) ——

Solution: The key insight here is that attacks on the TLS protocol are not made any easier by using half-TCP, because the cryptographic messages sent between the student and the server are unchanged. The only attack that becomes easier is the RST injection attack for an off-path attacker, since the attacker doesn't need to guess sequence numbers when injecting a RST packet from the student to the server. On-path and MITM attackers can see all sequence numbers, so RST injection is not any easier for them.

Q2 *Mutuality (SP21 Final Q9)* **(18 points)**

Recall the TLS handshake:



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 $\{M_2, \text{MAC}(M_2, I_s)\}_{C_s}$

In TLS, we verify the identity of the server, but not the client. How would we modify TLS to also verify the identity of the client?

Clarification during exam: All parts of this question refer to a modified TLS scheme designed to verify the identity of the client.

Q2.1 (3 points) Which of these additional values should the client send to the server?

○ (A) A certificate with the client's public key, signed by the client's private key

○ (B) A certificate with the client's public key, signed by the server's private key

○ (C) A certificate with the client's private key, signed by a certificate authority's private key

[✓ (D) A certificate with the client's public key, signed by a certificate authority's private key]

○ (E) ——

○ (F) ——

Solution: This is analogous to the server sending its certificate, which has the server's public key, signed by the certificate authority's private key.

Q2.2 (3 points) How should the client send the premaster secret in RSA TLS?

[✓ (G) Encrypted with the server's public key, signed by the client's private key]

○ (H) Encrypted with the client's public key, signed by the server's private key

○ (I) Encrypted with the server's public key, signed by a certificate authority's private key

○ (J) Encrypted with the client's public key, signed by a certificate authority's private key

○ (K) ——

○ (L) ——

Solution: The client should encrypt the premaster secret with the server's public key so that the server can decrypt it (just like in regular TLS).

However, the client should additionally sign the premaster secret, so that the server can validate the signature and confirm that the server is talking to the correct client. The client should sign the premaster secret with their own private key. (The client doesn't know the certificate authority's private key, and the CA's private key is only used to sign certificates anyway.)

Q2.3 (3 points) EvanBot argues that the key exchange protocol in Diffie-Hellman TLS doesn't need to be changed to support client validation. Is EvanBot right?

○ (A) Yes, because only the client knows the secret a, so the server can be sure it's talking to the legitimate client

○ (B) Yes, because the server has already received and verified the client's certificate

[✓ (C) No, the client must additionally sign their part of the Diffie-Hellman exchange with the client's private key]

○ (D) No, the client must additionally sign their part of the Diffie-Hellman exchange with the certificate authority's private key

○ (E) ——

○ (F) ——

Solution: Diffie-Hellman on its own doesn't provide any authenticity. We also need the client to sign their Diffie-Hellman message.

Q2.4 (2 points) True or False: The server can be sure that they're talking to the client (and not an attacker impersonating the client) immediately after the client and server exchange certificates.

○ (G) True [✓ (H) False] ○ (I) —— ○ (J) —— ○ (K) —— ○ (L) ——

Solution: False. Remember that certificates are public, and attackers can present a certificate for anyone. The ClientHello and ServerHello messages only contain random nonces and an agreement on what algorithms to use, so they also do not give the client and server any guarantees about who they're talking to.

The client and the server need to wait at least until the signatures are exchanged to verify that they're talking to the correct person. If an attacker tampers with the handshake, the client and the server may even have to wait until the MACs are exchanged.

Q2.5 (3 points) At what step in the TLS handshake can both the client and server be sure that they have derived the same symmetric keys?

○ (A) Immediately after the TCP handshake, before the TLS handshake starts

○ (B) Immediately after the ClientHello and ServerHello are sent

○ (C) Immediately after the client and server exchange certificates

○ (D) Immediately after the client and server verify signatures

[✓ ● (E) Immediately after the MACs are exchanged and verified]

○ (F) ——

Solution: The reasoning here is the same as in regular TLS. A MITM could tamper with messages, and the client and server will only detect this once they verify the MAC on the entire handshake.

Q2.6 (4 points) Which of these keys, if stolen individually, would allow the attacker to impersonate the client? Select all that apply.

[✓ ■ (G) Private key of a certificate authority]

[✓ ■ (H) Private key of the client]

☐ (I) Private key of the server

☐ (J) Public key of a certificate authority

☐ (K) None of the above

☐ (L) ——

Solution: If the attacker steals the private key of a trusted CA, they can sign a fake certificate claiming that the attacker's public key belongs to the client.

If the attacker steals the private key of the client, they can sign messages as the client.

Stealing the public key of the server doesn't help the attacker impersonate the client.

Q3 *I Love You 3000 (sp22-final-q11)* **(18 points)**

Tony wants to send a message, M, to his daughter, Morgan. The message is split across 3 packets, M_1, M_2, and M_3. Assume that both Tony and Morgan will use the modified version of TCP specified in each subpart. Each subpart is independent.

Q3.1 (3 points) Consider a modified version of TCP where **Morgan** no longer sends an ACK to Tony. If Tony sends M using this modified version of TCP and M_2 was dropped during delivery, then which of the following are true?

- ☐ M_2 will be resent until it is received by Morgan.
- ■ Morgan will be able to notice that M_2 is lost.
- ☐ Morgan will be able to reconstruct M even if M_2 is not resent.
- ☐ None of the above

Solution: If Morgan no longer sends an ACK to Tony, then Tony will never actually know that M_2 got dropped since Morgan will never "tell" Tony that she didn't get the packet. As such, even though Morgan knows that she didn't receive M_2, she won't be able to tell Tony this.

Q3.2 (3 points) Consider a modified version of TCP where **Tony** no longer sends an ACK to Morgan. If Tony sends M using this modified version of TCP and M_2 was dropped during delivery, then which of the following are true?

- ■ M_2 will be resent until it is received by Morgan.
- ■ Morgan will be able to notice that M_2 is lost.
- ☐ Morgan will be able to reconstruct M even if M_2 is not resent.
- ☐ None of the above

Solution: Here, since Tony no longer sends the ACK to Morgan, but M_2 is dropped when Tony sends the message to Morgan, not only will Morgan know that the packet got dropped, but she will be able to "notify" Tony about this through her ACK packet. If M_2 is not recent, Morgan has no way to reconstruct M.

Q3.3 (6 points) Consider a modified version of TCP where Tony and Morgan have the same ISN (Initial Sequence Number). Assume all adversaries can spoof packets. Which of the following is true about the resulting connection?

- ■ It is possible for an adversary who can see only packets sent by Tony to spoof more than one message from Tony to Morgan without being detected by either party.

- ■ It is possible for an adversary who can see only packets sent by Morgan to spoof more than one message from Tony to Morgan without being detected by either party.

- ☐ It is possible for an adversary who can see only packets sent by Tony to spoof only one message from Tony to Morgan without being detected by either party.

- ☐ It is possible for an adversary who can see only packets sent by Morgan to spoof only one message from Tony to Morgan without being detected by either party.

- ■ An in-path attacker can spoof more than one message from Tony to Morgan without being detected by either party.

- ■ An on-path attacker can spoof more than one message from Tony to Morgan without being detected by either party only if their message arrives before Tony's message.

- ☐ None of the above

Solution: If Tony and Morgan have the same ISN, then regardless of whether the adversary can only see packets sent by Tony or by Morgan, they will be able to spoof more than one message from Tony to Morgan without being detected by either party since the adversary can "start" spoofing from the very beginning of the conversation.

Furthermore, since an in-path attacker can block and modify packets, they can simply block any messages from Tony and replace it with their own. An on-path attacker cannot block messages, so they are subject to a race condition, meaning that the spoofed message would have to arrive before the actual message to be accepted by the receiver.

For the following subparts, for each modification to TLS, select all true statements. Each subpart is independent.

Q3.4 (3 points) The digital signature algorithm used to create the certificate is forgeable.

- ■ A MITM attacker can impersonate the server to the client.
- ☐ A MITM attacker can inject messages.
- ☐ An on-path attacker can read messages.
- ☐ None of the above

Solution: The attacker can send a forged certificate claiming that the attacker's public key is the server's public key. The certificate doesn't help the attacker learn the symmetric keys.

Q3.5 (3 points) In RSA TLS, the RSA encryption algorithm has a backdoor that lets anyone decrypt the ciphertext without the private key.

- ■ A MITM attacker can impersonate the server to the client.
- ■ A MITM attacker can inject messages.
- ■ An on-path attacker can read messages.
- ☐ None of the above

Solution: The attacker can decrypt the premaster secret and derive the symmetric keys to inject messages in both directions. Because decrypting the premaster secret is also how the server proves its identity to the client, a MITM could impersonate the server to the client with the RSA backdoor.