CS161 Summer 2025 — Introduction to Computer Security — Exam Prep 9

Q1 *Networking: New Phone Who This* (6 points)

EvanBot joins a new textbf{broadcast} local network with many users. CodaBot is on the local network, but EvanBot doesn't know CodaBot's phone number. EvanBot wants to learn CodaBot's phone number, using the following protocol:

1) EvanBot broadcasts a request asking what CodaBot's phone number is.
2) CodaBot sends a response to EvanBot with their phone number.
3) EvanBot caches the phone number.

Q1.1 (1 point) Which networking protocol is this most similar to?

● ARP ○ WPA2 ○ BGP ○ TCP

Solution: In ARP, the user broadcasts a request asking for an IP address to MAC mapping (in this protocol, it's a user to phone number mapping). Then, the user with that IP address responds with their MAC address (here, their phone number instead).

ARP and this modified protocol will both then cache the resulting answer.

Q1.2 (2 points) Eve is an on-path attacker in the local network. Select all attacks that Eve can carry out.

☐ Perform an online brute-force attack to learn CodaBot's phone number, by sending back every possible phone number to EvanBot.

■ Learn CodaBot's phone number by reading message(s) Eve was not supposed to read.

■ Learn CodaBot's phone number without reading message(s) Eve was not supposed to read.

■ Convince EvanBot that CodaBot's phone number is some malicious value chosen by Eve.

☐ None of the above

Solution: (A): False. EvanBot sends no sort of signal as to whether the phone number is correct, so Eve sending every possible phone number to EvanBot is not helpful for learning CodaBot's phone number.

(B): True. CodaBot's phone number is supposed to be sent directly to EvanBot. However, this is a broadcast local network, so to send this message, CodaBot will broadcast the message to everybody, and expect that only EvanBot will read that message (and everyone else will discard it). Eve receives, but is not supposed to read the message with CodaBot's phone number, but can maliciously choose to read it.

(C): The intended answer was false – Eve needs to be able to read CodaBot's reply (which she's not supposed to read) in order to learn CodaBot's phone number. The only message Eve is allowed to read in this protocol is the initial request, which does not contain CodaBot's phone number.

However, we did not specify whether Eve could broadcast her own legitimate request for CodaBot's phone number, so everyone will receive credit for this subpart.

(D): True. As in ARP spoofing, Eve can send a malicious response to EvanBot claiming that she is CodaBot and her phone number is some malicious value. If Eve's answer arrives before CodaBot's answer, then EvanBot will be convinced that CodaBot's phone number is Eve's malicious value.

In the next three subparts, consider this modification to the protocol: Instead of sending just the phone number, CodaBot sends their public key, and a signature on their phone number.

When EvanBot receives this data, EvanBot uses the public key to verify the signature on the phone number.

Eve wants to trick EvanBot into thinking CodaBot's phone number is a malicious value chosen by Eve. What values does Eve include in the packet she sends to EvanBot?

Q1.3 (1 point) For the public key, Eve sends:

[✓ Eve's public key] ○ EvanBot's public key

○ CodaBot's public key ○ The router's public key

Solution: The vulnerability here is that CodaBot's public key is not being verified. Therefore, Eve can send her own public key, and EvanBot has no way to distinguish between Eve's public key and CodaBot's public key.

Q1.4 (1 point) For the signature over the phone number, Eve signs using:

[✓ Eve's private key] ○ EvanBot's private key

○ CodaBot's private key ○ The router's private key

Solution: In the previous part, Eve sends her public key, so now EvanBot is convinced that Eve's public key corresponds to the user CodaBot.

Now, Eve can use her own corresponding private key to sign the phone number. EvanBot will use Eve's public key (which Bot thinks belongs to CodaBot) to verify the phone number, and the signature will check out (since it was made with Eve's private key and verified with Eve's public key).

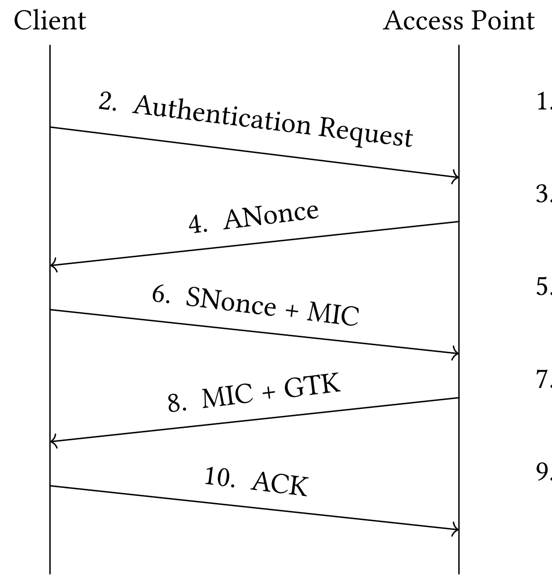
Q1.5 (1 point) How often will this attack succeed?

○ 100% of the time ○ Only when CodaBot's packet arrives first

[✓ Only when Eve's packet arrives first] ○ Never

Solution: This attack involves a race condition, because EvanBot is not expecting two answers, and Eve's answer must arrive before CodaBot's answer in order to be accepted.

Q2 *I am Inevitable (SP22 Final Q10)* **(20 points)**



1. Client and AP derive the PSK from SSID and password.

3. AP randomly chooses ANonce.

5. Client randomly chooses SNonce and derives PTK.

7. AP derives PTK and verifies the MIC.

9. Client verifies the MIC.

For each method of client-AP authentication, select all things that the given adversary would be able to do. Assume that:

- The attacker does not know the WPA-PSK password but that they know that client's and AP's MAC addresses.
- For rogue AP attacks, there exists a client that knows the password that attempts to connect to the rogue AP attacker.
- The AMAC is the Access Point's MAC address and the SMAC is the Client's MAC address.

Q2.1 (5 points) The client and AP perform the WPA 4-way handshake with the following modifications:

- $\text{PTK} = F(\text{ANonce}, \text{SNonce}, \text{AMAC}, \text{SMAC}, \text{PSK})$, here F is a secure key derivation function
- $\text{MIC} = \text{PTK}$

■ An on-path attacker that observes a successful handshake can decrypt subsequent WPA messages without learning the value of the PSK.

☐ An on-path attacker that observes a successful handshake can trick the AP into completing a new handshake without learning the value of the PSK.

☐ An on-path attacker that observes a successful handshake can learn the PSK without brute force.

☐ A rogue AP attacker can learn the PSK without brute force.

■ A rogue AP attacker can only learn the PSK if they use brute force.

☐ None of the above

Solution: Because the MIC is the value of the PTK, it is trivial to decrypt subsequent communications. However, replay attacks are not possible since the ANonce is chosen by the AP, so the attacker can't trick the AP into completing a new handshake.

Additionally, because all the information needed to brute-force the PSK is sent in the clear (ANonce, SNonce, and MICs), brute-force attacks are possible by the rogue AP. However, there is no way of learning the PSK given the PTK with any method other than brute-force.

Q2.2 (5 points) The client and AP perform the WPA 4-way handshake with the following modifications:

- $\text{PTK} = F(\text{ANonce}, \text{SNonce}, \text{AMAC}, \text{SMAC})$, here F is a secure key derivation function
- $\text{MIC} = \text{HMAC}(\text{PTK}, \text{Dialogue})$

■ An on-path attacker that observes a successful handshake can decrypt subsequent WPA messages without learning the value of the PSK.

■ An on-path attacker that observes a successful handshake can trick the AP into completing a new handshake without learning the value of the PSK.

☐ An on-path attacker that observes a successful handshake can learn the PSK without brute force.

☐ A rogue AP attacker can learn the PSK without brute force.

☐ A rogue AP attacker can only learn the PSK if they use brute force.

☐ None of the above

Solution: Because the PSK isn't actually incorporated into this handshake, it is trivial for an attacker to derive the PTK to decrypt subsequent messages, and it is easy for them to form a new handshake with the AP.

Q2.3 (5 points) The client and AP perform the WPA 4-way handshake with the following modifications:

- Authentication: Client sends $H(\text{PSK})$ to AP, where H is a secure cryptographic hash.
- Verification: AP compares $H(\text{PSK})$ and to the value it received.
- AP sends: $\text{Enc}(\text{PSK}, \text{PTK})$ to client, where Enc is an IND-CPA secure encryption algorithm.

☐ An on-path attacker that observes a successful handshake can decrypt subsequent WPA messages without learning the value of the PSK.

[✓] An on-path attacker that observes a successful handshake can trick the AP into completing a new handshake without learning the value of the PSK.

☐ An on-path attacker that observes a successful handshake can learn the PSK without brute force.

☐ A rogue AP attacker can learn the PSK without brute force.

[✓] A rogue AP attacker can only learn the PSK if they use brute force.

☐ None of the above

Solution: Assuming that an on-path attacker doesn't know the PSK, they can't brute-force the PTK since it's encrypted using the PSK and thus can't decrypt subsequent communications without learning the PSK. However, there are no nonces involved in the handshake, so it is possible to replay $H(\text{PSK})$ to trick the AP into completing a new handshake.

Because the PSK is hashed, it is not possible to learn the PSK as the attacker without brute force. However, if brute force is allowed, it is easy to guess a value of PSK and check if its hash equals the sent $H(\text{PSK})$.

Q2.4 (5 points) The client and AP perform the WPA 4-way handshake with the following modifications:

- Authentication: Client conducts a Diffie-Hellman exchange with the AP to derive a shared key K.
- Client sends: $\text{Enc}(K, \text{PSK})$ to the AP.
- Verification: Check if $\text{Dec}(K, \text{Ciphertext})$ equals the PSK
- Upon verification, AP sends: $\text{Enc}(K, \text{PTK})$, where PTK is a random value, and sends it to the client.
- Assume that Enc is an IND-CPA secure encryption algorithm.

☐ An on-path attacker that observes a successful handshake can decrypt subsequent WPA messages without learning the value of the PSK.

☐ An on-path attacker that observes a successful handshake can trick the AP into completing a new handshake without learning the value of the PSK.

☐ An on-path attacker that observes a successful handshake can learn the PSK without brute force.

■ A rogue AP attacker can learn the PSK without brute force.

☐ A rogue AP attacker can only learn the PSK if they use offline brute force.

☐ None of the above

Solution: Unlike the previous question, Diffie-Hellman defends against replay attacks since the AP would choose a new private Diffie-Hellman component for each handshake. However, a rogue AP learns the value of K, and is thus able to learn the value of the PSK by decrypting $\text{Enc}(K, \text{PSK})$ using K.