Cryptography II

Question 1 ***Diffie–Hellman key exchange*** **(15 min)**

Recall that in a Diffie-Hellman key exchange, there are values a, b, g and p. Alice computes $g^a \bmod p$ and Bob computes $g^b \bmod p$.

(a) Which of these values (a, b, g, and p) are publicly known and which must be kept private?

Solution:

g and p are publicly known. Implementations of Diffie-Hellman often have carefully picked values of g and p which are known to everyone. Alice and Bob must keep a and b secret respectively.

(b) Mallory can eavesdrop, intercept, and modify everything sent between Alice and Bob. Alice and Bob perform Diffie-Hellman to agree on a shared symmetric key K. After the exchange, Bob gets the feeling something went wrong and calls Alice. He compares his value of K to Alice's and realizes that they are different. Explain what Mallory has done.

Solution:

Mallory is performing a **man-in-the-middle attack**. Mallory pretends to be Bob when she talks to Alice, and Mallory also pretends to be Alice when she talks to Bob. In this way, both Alice and Bob are unknowingly talking to Mallory. Mallory can then decrypt/re-encrypt the traffic in both directions and modify it however she wishes to.

More technically, when Alice sends $A = g^a \bmod p$ to Bob, Mallory intercepts this (preventing it from going to Bob), and sends back to Alice: $M = g^c \bmod p$. Now when Alice sends a message to Bob, she uses $K_{bad} = M^a \bmod p$ which Mallory knows as $K_{bad} = A^c \bmod p$. Mallory can then decrypt all messages sent from Alice. She can also send messages to Alice which Alice thinks are from Bob. Mallory then does the same trick to Bob.

(c) Alice and Bob want to prevent Mallory from tampering with their keys by attaching a hash to each message (ie. Alice sends $(g^a, H(g^a))$ and Bob sends $(g^b, H(g^b))$). Does this successfully stop Mallory?

Solution: No, this does not stop Mallory. Since a hash function is public, Mallory can do the same attack as before by additionally attaching a hash of her spoofed key.

(d) Alice and Bob want to prevent Mallory from tampering with their keys by using a MAC. Assume they hold a shared symmetric key K and attach a MAC to each message (ie. Alice sends $(g^a, MAC_k(g^a))$ and Bob sends $(g^b, MAC_k(g^b))$). Does this successfully stop Mallory? Assume Mallory can observe multiple, unique key exchanges between Alice and Bob before attempting an attack.

Solution: Yes, this does stop Mallory. Since she doesn't know the key for the MAC she can't ever successfully spoof a valid message for an exponent that she knows. Replay attacks won't work since she needs to figure out the correct exponent.

As an aside, DH is unnecessary at this point since they already share a symmetric key and can just use a block cipher to generate fresh symmetric keys.

Question 2 ***Perfect Forward Secrecy*** **(15 min)**

Alice (A) and Bob (B) want to communicate using some shared symmetric key encryption scheme. Consider the following key exchange protocols which can be used by A and B to agree upon a shared key, K.

ElGamal-Based Key Exchange

Message 1	$A \to B$:	$\{K\}_{PK_B}$
	Key exchanged	
Message 2	$A \leftarrow B$:	$\{secret1\}_K$
Message 3	$A \to B$:	$\{secret2\}_K$

Diffie-Hellman Key Exchange

Message 1	$A \to B$:	$g^a \mod p$
Message 2	$A \leftarrow B$:	$g^b \mod p$
	Key exchanged $K = g^{ab} \mod p$	
Message 3	$A \leftarrow B$:	$\{secret1\}_K$
Message 4	$A \to B$:	$\{secret2\}_K$

Some additional details:

- PK_B is Bob's long-lived public key.
- K, the Diffie-Hellman exponents a and b, and the messages themselves are destroyed once all messages are sent. That is, these values are not stored on Alice and Bob's devices after they are done communicating.

Eavesdropper Eve records all communications between Alice and Bob, but is unable to decrypt them. At some point in the future, Eve is lucky and manages to compromise Bob's computer.

(a) Is the confidentiality of Alice and Bob's prior ElGamal-based communication in jeopardy?

Solution: Yes. The compromise of Bob's computer gives Eve access to Bob's private key, allowing Eve to decrypt the traffic she previously recorded that was encrypted using Bob's public key. Once decrypted, she obtains K, and can then apply it to decrypt the traffic encrypted using symmetric key encryption.

(b) What about Alice and Bob's Diffie-Hellman-based communication?

Solution: No. Since Alice and Bob destroy the DH exponents a and b after use, and since the key computed from them itself is never transmitted, there is no information present on Bob's computer that Eve can leverage to recover K. This means that with Diffie-Hellman key exchanges, later compromises in no way harm the confidentiality of previous communication, even if the ciphertext for that communication was recorded in full. This property is called *Perfect Forward Secrecy*.

Question 3 ***Confidentiality and integrity*** ()

Alice and Bob want to communicate with confidentiality and integrity. They have:

- Symmetric encryption.
  - Encryption: $\mathsf{Enc}(\mathsf{K}, \mathsf{m})$.
  - Decryption: $\mathsf{Dec}(\mathsf{K}, \mathsf{c})$.
- Cryptographic hash function: $\mathsf{Hash}(\mathsf{m})$.
- MAC: $\mathsf{MAC}(\mathsf{K}, \mathsf{m})$.
- Signature: $\mathsf{Sign}_{\mathsf{sk}}(\mathsf{m})$.

They share a symmetric key K and know each other's public key.

We assume these cryptographic tools do not interfere with each other when used in combination; *i.e.*, we can safely use the same key for encryption and MAC.

Alice sends to Bob
1. $\mathsf{c} = \mathsf{Hash}(\mathsf{Enc}(\mathsf{K}, \mathsf{m}))$
2. $\mathsf{c} = \mathsf{c}_1, \mathsf{c}_2$: where $\mathsf{c}_1 = \mathsf{Enc}(\mathsf{K}, \mathsf{m})$ and $\mathsf{c}_2 = \mathsf{Hash}(\mathsf{Enc}(\mathsf{K}, \mathsf{m}))$
3. $\mathsf{c} = \mathsf{c}_1, \mathsf{c}_2$: where $\mathsf{c}_1 = \mathsf{Enc}(\mathsf{K}, \mathsf{m})$ and $\mathsf{c}_2 = \mathsf{MAC}(\mathsf{K}, \mathsf{m})$
4. $\mathsf{c} = \mathsf{c}_1, \mathsf{c}_2$: where $\mathsf{c}_1 = \mathsf{Enc}(\mathsf{K}, \mathsf{m})$ and $\mathsf{c}_2 = \mathsf{MAC}(\mathsf{K}, \mathsf{Enc}(\mathsf{K}, \mathsf{m}))$
5. $\mathsf{c} = \mathsf{Sign}_{\mathsf{sk}}(\mathsf{Enc}(\mathsf{K}, \mathsf{m}))$
6. $\mathsf{c} = \mathsf{c}_1, \mathsf{c}_2$: where $\mathsf{c}_1 = \mathsf{Enc}(\mathsf{K}, \mathsf{m})$ and $\mathsf{c}_2 = \mathsf{Enc}(\mathsf{K}, \mathsf{Sign}_{\mathsf{sk}}(\mathsf{m}))$

(a) Which ones of them can Bob decrypt?

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6

Solution: Bob cannot decrypt Scheme 1 because he cannot invert Hash. Similarly, he cannot extract the original message from the signature sent in Scheme 5.

The signature does not include the message.

In sum: 2-4, 6.

(b) Consider an eavesdropper Eve, who can see the communication between Alice and Bob.

Which schemes, of those decryptable in (a), also provide *confidentiality* against Eve?

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6

Solution: Scheme 3 does not provide confidentiality because the MAC is sent in plaintext. For the same message, the MAC is the same, thus leaky.

In sum: 2, 4, 6.

(c) Consider a man-in-the-middle Mallory, who can eavesdrop and modify the communication between Alice and Bob.

Which schemes, of those decryptable in (a), provide *integrity* against Mallory? *i.e.*, Bob can detect any tampering with the message?

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6

Solution: Scheme 2 does not provide integrity as Mallory can forge a message by sending Bob $(\mathsf{c}', \mathsf{Hash}(\mathsf{c}'))$.

In sum: 3, 4, 6.

(d) Many of the schemes above are insecure against a *replay attack*.

If Alice and Bob use these schemes to send many messages, and Mallory remembers an encrypted message that Alice sent to Bob, some time later, Mallory can send the exact same encrypted message to Bob, and Bob will believe that Alice sent the message *again*.

How to modify those schemes with confidentiality & integrity to prevent replay attack?

◇ The first scheme providing confidentiality & integrity is Scheme ☐.

The modification is:

◇ The second scheme providing confidentiality & integrity is Scheme ☐.

The modification is:

Solution: Add a non-repeating nonce or timestamp in the MAC (Scheme 4) or in the signature (Scheme 6).

In sum: In both 4 and 6, we replace message m with $\mathsf{timestamp} \parallel \mathsf{m}$.