CS 161 Spring 2024 — Introduction to Computer Security — Exam Prep 6

Q1 *Bob's Birthday* (11 points)

It's Bob's birthday! Alice wants to send an encrypted birthday message to Bob using ElGamal.

Recall the definition of ElGamal encryption:

- b is the private key, and $B = g^b \bmod p$ is the public key.
- $\mathsf{Enc}(B, M) = (C_1, C_2)$, where $C_1 = g^r \bmod p$ and $C_2 = M \times B^r \bmod p$
- $\mathsf{Dec}(b, C_1, C_2) = C_1^{-b} \times C_2 \bmod p$

Q1.1 (2 points) Mallory wants to tamper with Alice's message to Bob. In response, Alice decides to sign her message with an RSA digital signature. Bob receives the signed message and verifies the signature successfully. Can he be sure the message is from Alice?

- ◯ Yes, because RSA digital signatures are unforgeable.
- ◯ Yes, because RSA encryption is IND-CPA secure.
- ◯ No, because Mallory could have blocked Alice's message and replaced it with a different one.
- ◯ No, because Mallory could find a different message with the same hash as Alice's original message.

As we discussed in class, ElGamal is malleable, meaning that a man-in-the-middle can change a message in a *predictable* manner, such as producing the ciphertext of the message $2 \times M$ given the ciphertext of M.

Q1.2 (3 points) Consider the following modification to ElGamal: Encrypt as normal, but further encrypt portions of the ciphertext with a block cipher E, which has a block size equal to the number of bits in p. In this scheme, Alice and Bob share a symmetric key K_{sym} known to no one else.

Under this modified scheme, C_1 is computed as $E_{K_{\mathsf{sym}}}(g^r \bmod p)$ and C_2 is computed as $E_{K_{\mathsf{sym}}}(M \times B^r \bmod p)$. Is this scheme still malleable?

- ◯ Yes, because block ciphers are not IND-CPA secure encryption schemes
- ◯ Yes, because the adversary can still forge $k \times C_2$ to produce $k \times M$
- ◯ No, because block ciphers are a pseudorandom permutation
- ◯ No, because the adversary isn't able to learn anything about the message M

The remaining parts are independent of the previous part.

For Bob's birthday, Mallory hacks into Bob's computer, which stores Bob's private key b. She isn't able to read b or overwrite b with an arbitrary value, but she can multiply the stored value of b by a random value z known to Mallory.

Mallory wants to send a message to Bob that appears to decrypt as normal, but **using the modified key** $b \cdot z$. Give a new encryption formula for C_1 and C_2 that Mallory should use. Make sure you only use values known to Mallory!

Clarification during exam: For subparts 3 and 4, assume that the value of B is unchanged.

Q1.3 (3 points) Give a formula to produce C_1, encrypting M.

Q1.4 (3 points) Give a formula to produce C_2, encrypting M.

Q2 *Cryptography: EvanBot Signature Scheme* **(12 points)**

EvanBot decides to make a signature scheme!

To initialize the system, a Diffie-Hellman generator g and prime p are generated and shared to all parties. The private key is some $x \bmod p$ chosen randomly, and the public key is $y = g^x \bmod p$.

To sign a message m such that $2 \le m \le p - 2$:

1. Choose a random integer k between 2 and $p - 2$.
2. Set $r = g^k \bmod p$.
3. Set $s = (\mathsf{H}(m) - xr)k^{-1} \bmod (p - 1)$. If $s = 0$, restart from Step 1.
4. Output (r, s) as the signature.

Clarification after exam: k is chosen to be coprime to $p - 1$.

To verify, check that $g^{H(m)} \equiv$ ______ $\bmod p$. We will fill in this blank in the next few subparts.

Q2.1 (3 points) Select the correct expression for $H(m)$ in terms of x, r, k, s and $p - 1$.
HINT: Use Step 3 of the signature algorithm.

- ○ $k(xr)^{-1} + s \bmod (p - 1)$
- ○ $k^{-1} + xr \bmod (p - 1)$
- ○ $ks - xr \bmod (p - 1)$
- ○ $ks + xr \bmod (p - 1)$

Q2.2 (4 points) Using the previous result, select the correct value for the blank in the verification step.
HINT: Replace the $H(m)$ in $g^{H(m)}$ with your results from the previous subpart.

- ○ $y^s r^2 \bmod p$
- ○ $r^y r^s \bmod p$
- ○ $y^r r^s \bmod p$
- ○ $rg^{yr} \bmod p$

Q2.3 (5 points) Show how to recover the private key x if a signature is generated such that $s = 0$ (i.e. the check on Step 3 is ignored).

Q3 *The Lorenzo Von Matterhorn* **(0 points)**

Barney needs to make sure that no attackers can access his highly sensitive, top secret playbook tricks!

For each password scheme, select all true statements. Assume that:

- Each user has a unique username, but not necessarily a unique password.
- All information is stored in a read-only database that both the server and the attacker can access.
- The server has a symmetric key K not known to anyone else. The server also has a secret key SK not known to anyone else, and a corresponding public key PK that everyone knows.
- An *operation* is defined as one of the following actions: hash, encryption, decryption, and HMAC.
- The attacker does not have access to a client UI; therefore, online attacks are not possible.

Q3.1 For each user, the database contains $\mathsf{username}$ and $\mathsf{H}(\mathsf{password})$, where H is a cryptographic hash function.

☐ If a user inputs a username and password, the server can verify whether the password is correct

☐ Given the information in the database, the attacker can verify that a given username and password pairing is correct.

☐ The server can list all plaintext passwords by computing at most one operation per user

☐ An attacker can list all passwords by computing at most one operation per possible password

☐ None of the above

Q3.2 For each user, the database contains $\mathsf{username}$ and $\mathsf{HMAC}(K, \mathsf{password})$.

☐ If a user inputs a username and password, the server can verify whether the password is correct

☐ Given the information in the database, the attacker can verify that a given username and password pairing is correct.

☐ The server can list all plaintext passwords by computing at most one operation per user

☐ An attacker can list all passwords by computing at most one operation per possible password

☐ None of the above

Q3.3 For this subpart, Enc denotes an IND-CPA secure symmetric encryption function.

For each user, the database contains username and $\mathsf{Enc}(K, \mathsf{password})$.

- ☐ If a user inputs a username and password, the server can verify whether the password is correct
- ☐ Given the information in the database, the attacker can verify that a given username and password pairing is correct.
- ☐ The server can list all plaintext passwords by computing at most one operation per user
- ☐ An attacker can list all passwords by computing at most one operation per possible password
- ☐ None of the above

Q3.4 For this subpart, RSA denotes RSA encryption without OAEP padding.

For each user, the database contains username and $\mathsf{RSA}(\mathsf{PK}, \mathsf{password})$.

- ☐ If a user inputs a username and password, the server can verify whether the password is correct
- ☐ Given the information in the database, the attacker can verify that a given username and password pairing is correct.
- ☐ The server can list all plaintext passwords by computing at most one operation per user
- ☐ An attacker can list all passwords by computing at most one operation per possible password
- ☐ None of the above

Q3.5 Consider a modification to the scheme in the first subpart: Instead of storing $\mathsf{H}(\mathsf{password})$ per user, we now store $\mathsf{H}(\mathsf{password}||\mathsf{salt})$ per user.

Assume that concatenation does not count as an operation. Compared to the original scheme, which of the following algorithms for generating salts would force the attacker to compute more operations to list all passwords? Select all that apply.

- ☐ A 128-bit value, randomly generated per user
- ☐ A 128-bit counter, starting at 0 and incremented per user
- ☐ A 128-bit counter, starting at a random number and incremented per user
- ☐ None of the above

Q3.6 Which of these hash algorithms makes the scheme in the first subpart most secure against offline brute-force attacks? Briefly explain (10 words or fewer).

- ○ MD5
- ○ SHA2-256
- ○ Argon2Key (PBKDF2)