COMP5631: Cryptography and Security
2025 Spring – Written Assignment Number 3
Handed out: on March 13, 2025
Due: on March 30 by 11:30pm.

Please upload your solution paper into Canvas by 23:30 on March 30. No email submission will be accepted. You should work out a solution to each question with your own hands. This is the best way to learn these topics.

Q1. Recall the RSA public-key cipher introduced in Lecture 8. Prove the correctness of the decryption process: $M = C^d \bmod n$. (Hint: you may use Euler's theorem and Fermat's theorem proved in Q4 of Assignment 1.) 20 marks

Solution. To prove the correctness of the RSA decryption process, we need to demonstrate that $M = C^d \bmod n$ where $C = M^e \bmod n$ for any message $M \in \mathbb{Z}_n$.

Let $n = pq$ where p and q are distinct primes. In the RSA cryptosystem, the key pair (e, d) satisfies $ed \equiv 1 \pmod{\phi(n)}$, where $\phi(n) = (p-1)(q-1)$ is Euler's totient function.

This congruence relation implies the existence of an integer k such that:

$$ed = 1 + k\phi(n) \tag{1}$$

Substituting the encryption $C = M^e \bmod n$ into the decryption formula:

$$\begin{aligned} C^d \bmod n &= (M^e)^d \bmod n && (2)\\ &= M^{ed} \bmod n && (3)\\ &= M^{1+k\phi(n)} \bmod n && (4)\\ &= M \cdot M^{k\phi(n)} \bmod n && (5) \end{aligned}$$

We now need to show that $M \cdot M^{k\phi(n)} \equiv M \pmod{n}$. This is equivalent to proving $M^{k\phi(n)} \equiv 1 \pmod{n}$.

We can establish this by examining the congruence relations modulo p and modulo q separately, then applying the Chinese Remainder Theorem.

For any M with $0 \leq M < n$, we consider two cases:

Case 1: If $\gcd(M, p) = 1$, then by Fermat's Little Theorem, $M^{p-1} \equiv 1 \pmod{p}$. Therefore:

$$\begin{aligned} M^{k\phi(n)} &= M^{k(p-1)(q-1)} && (6)\\ &= (M^{p-1})^{k(q-1)} && (7)\\ &\equiv 1^{k(q-1)} && (8)\\ &\equiv 1 \pmod{p} && (9) \end{aligned}$$

Case 2: If $\gcd(M, p) \neq 1$, then since $M < n$, we must have $M \equiv 0 \pmod{p}$. In this case:

$$M^{ed} \equiv 0^{ed} \equiv 0 \equiv M \pmod{p} \tag{10}$$

By symmetric reasoning, we can establish that $M^{ed} \equiv M \pmod{q}$ as well.

By the Chinese Remainder Theorem, since $M^{ed} \equiv M \pmod{p}$ and $M^{ed} \equiv M \pmod{q}$, we have $M^{ed} \equiv M \pmod{n}$, which proves:

$$C^d \bmod n = M^{ed} \bmod n = M \tag{11}$$

Thus, the RSA decryption process correctly recovers the original message.

Q2. Let p be a prime and α be a primitive root modulo p. The ElGamal public-key cipher $(\mathcal{M}, \mathcal{C}, \mathcal{K}_e, \mathcal{K}_d, E_{k_e}, D_{k_d})$ is defined as follows:

- $\mathcal{M} = \mathbf{Z}_p^\times = \{1, 2, 3, \cdots, p-1\},\ \mathcal{C} = \mathbf{Z}_p^\times \times \mathbf{Z}_p^\times,\ \mathcal{K}_e = \{p\} \times \{\alpha\} \times \mathbf{Z}_p^\times,\ \mathcal{K}_d = \mathbf{Z}_{p-1}.$

A user first chooses a random number u in $\mathbf{Z}_{p-1}$ as his private key $k_d := u$, then publicizes his public key $k_e = (p, \alpha, \beta)$, where $\beta = \alpha^u \bmod p$.

To encrypt a message x with a public key $k_e = (p, \alpha, \beta)$, one picks up a (secret) random number $v \in \mathbf{Z}_{p-1}$, and then does the encryption as follows:

$$E_{k_e}(x, v) = (y_1, y_2),$$

where $y_1 = \alpha^v \bmod p,$ and $y_2 = x\beta^v \bmod p.$

When the receiver receives the ciphertext $(y_1, y_2) \in \mathbf{Z}_p^\times \times \mathbf{Z}_p^\times$, he does the decryption as follows:

$$D_{k_d}(y_1, y_2) = y_2 \left(y_1^{k_d}\right)^{-1} \bmod p,$$

where $\left(y_1^{k_d}\right)^{-1}$ denotes the multiplicative inverse of $y_1^{k_d}$ modulo p. Prove that the decryption process above is correct. 20 marks

Solution. To prove the correctness of the ElGamal decryption process, we need to demonstrate that $D_{k_d}(y_1, y_2) = x$ for any ciphertext $(y_1, y_2) \in \mathbb{Z}_p^\times \times \mathbb{Z}_p^\times$.

Given the ciphertext (y_1, y_2), we have:

$$\begin{aligned} D_{k_d}(y_1, y_2) &= y_2 \left(y_1^{k_d}\right)^{-1} \bmod p && (12)\\ &= (x\beta^v)\left(\alpha^{uv}\right)^{-1} \bmod p && (13)\\ &= x\alpha^{uv}\alpha^{-uv} \bmod p && (14)\\ &= x \bmod p && (15) \end{aligned}$$

Q3. The domain and range of the encryption function E_{k_e} of the ElGamal cipher are $\mathbf{Z}_p^\times$ and $\mathbf{Z}_p^\times \times \mathbf{Z}_p^\times$. The binary operation associated to $\mathbf{Z}_p^\times$ is $\otimes_p$ and the binary operation associated to $\mathbf{Z}_p^\times \times \mathbf{Z}_p^\times$ is

$$(x_1, y_1) \otimes_p (x_2, y_2) := (x_1 \otimes_p x_2, y_1 \otimes_p y_2).$$

Show that the ElGamal cipher is multiplicatively homomorphic. [Hint: You may need to use Fermat's theorem, see Assignment 1.] (20 marks)

Solution. To show that the ElGamal cipher is multiplicatively homomorphic, we need to demonstrate that for any two plaintexts $x_1, x_2 \in \mathbb{Z}_p^\times$ and their corresponding ciphertexts $(y_1, y_2) = E_{k_e}(x_1, v_1)$ and $(y_3, y_4) = E_{k_e}(x_2, v_2)$, the product of the ciphertexts $(y_1, y_2) \otimes_p (y_3, y_4)$ is equal to the encryption of the product of the plaintexts $x_1 \otimes_p x_2$.

Given the ciphertexts $(y_1, y_2) = E_{k_e}(x_1, v_1)$ and $(y_3, y_4) = E_{k_e}(x_2, v_2)$, we have:

$$y_1 = \alpha^{v_1} \bmod p \tag{16}$$
$$y_2 = x_1\beta^{v_1} \bmod p \tag{17}$$
$$y_3 = \alpha^{v_2} \bmod p \tag{18}$$
$$y_4 = x_2\beta^{v_2} \bmod p \tag{19}$$

The product of the ciphertexts is:

$$(y_1, y_2) \otimes_p (y_3, y_4) = (\alpha^{v_1}\alpha^{v_2} \bmod p, x_1\beta^{v_1}x_2\beta^{v_2} \bmod p) \tag{20}$$
$$= (\alpha^{v_1+v_2} \bmod p, x_1x_2\beta^{v_1+v_2} \bmod p) \tag{21}$$

The decryption of the product of the ciphertexts is:

$$D_{k_d}((\alpha^{v_1+v_2} \bmod p, x_1x_2\beta^{v_1+v_2} \bmod p)) = x_1x_2\beta^{v_1+v_2}\left(\alpha^{v_1+v_2}\right)^{-1} \bmod p \tag{22}$$
$$= x_1x_2\beta^{v_1+v_2}\alpha^{-(v_1+v_2)} \bmod p \tag{23}$$
$$= x_1x_2\beta^{v_1+v_2}\alpha^{-v_1}\alpha^{-v_2} \bmod p \tag{24}$$
$$= x_1x_2\beta^{v_1}\beta^{v_2}\alpha^{-v_1}\alpha^{-v_2} \bmod p \tag{25}$$
$$= x_1x_2 \bmod p \tag{26}$$

Therefore, the product of the ciphertexts is equal to the encryption of the product of the plaintexts, which demonstrates that the ElGamal cipher is multiplicatively homomorphic.

Q4. Consider the digital signature scheme $m||D_{k_d^{(A)}}(h(m))$ introduced in Lecture 7 and answer the following two questions:

- Why should the underlying public-key cipher satisfy Condition C1? 5 marks
- Consider the following forgery attack on the digital signature scheme. Carol finds out a pair of messages m_1 and m_2 such that
  1. m_2 has the same length as the digital signature; and
  2. $h(m_1) = E_{k_e^{(A)}}(m_2)$.

  If this is computationally feasible, Carol can claim that m_2 is Alice's digital signature on m_1. How should the underlying public-key cipher and hash function h be designed with respect to this forgery attack? 15 marks

Solution.

1. Condition C1 requires that the underlying public-key cipher be secure against chosen-plaintext attacks. This is necessary to prevent an attacker from generating a valid digital signature for a message that they did not sign. If the public-key cipher is not secure against chosen-plaintext attacks, an attacker could forge a digital signature by encrypting a message of their choice and appending the decryption of the hash of the message. This would allow the attacker to claim that the message is a valid digital signature.

2. To prevent the forgery attack described, the underlying public-key cipher and hash function h should be designed such that the hash of a message is not equal to the encryption of another message. This can be achieved by using a secure hash function that produces unique outputs for different inputs. Additionally, the public-key cipher should be secure against chosen-plaintext attacks to prevent an attacker from generating a valid digital signature for a message that they did not sign. By ensuring that the hash function and public-key cipher are designed to prevent the forgery attack, the security of the digital signature scheme can be maintained.

Q5. Given a one-key block cipher $(\mathcal{M}, \mathcal{C}, \mathcal{K}, E_k, D_k)$, where E_k maps a block of n bits into a block of n bits, and each key k has l bits, one can construct a hash function as follows.

For a given binary string m, pad m with 0...01 on the left side so that the length of the modified message m' is a multiple of ℓ. Even if the length of m is a multiple of ℓ, padding is also done.

Then divide m' into blocks of length ℓ: $m' = m_t \cdots m_2 m_1$. The hash value H is computed as follows:

$$\begin{aligned} H_0 &= \text{fixed initial value of } n \text{ bits} \\ H_i &= E_{m_i}(H_{i-1}) \\ H &= H_t \end{aligned}$$

- Two distinct keys k and k' are called **equivalent keys**, if $E_k(m) = E_{k'}(m)$ for all plaintexts m. If it is easy to find equivalent keys of the given cipher, is it possible to find strong collisions of this hash function? Justify your answer. 10 marks
- If you are given two distinct keys k and k' such that $E_{k'}(E_k(m)) = E_k(E_{k'}(m))$ for all plaintexts m, could you find strong collisions of this hash function? Justify your answer. 10 marks

Solution.

- If it is easy to find equivalent keys of the given cipher, it is possible to find strong collisions of the hash function. Since equivalent keys produce the same ciphertext for all plaintexts, an attacker can construct a collision by finding two distinct messages that produce the same hash value. By using equivalent keys to encrypt the messages, the attacker can generate a collision by constructing two messages that produce the same hash value under the equivalent keys. Therefore, if equivalent keys can be easily found, it is possible to find strong collisions of the hash function.
- If two distinct keys k and k' satisfy $E_{k'}(E_k(m)) = E_k(E_{k'}(m))$ for all plaintexts m, it is possible to find strong collisions of the hash function. By using the keys k and k' to encrypt the messages, an attacker can construct a collision by finding two distinct messages that produce the same hash value. Since the keys satisfy the condition $E_{k'}(E_k(m)) = E_k(E_{k'}(m))$, the attacker can generate a collision by constructing two messages that produce the same hash value under the keys k and k'. Therefore, if the keys satisfy the given condition, it is possible to find strong collisions of the hash function.