Congratulations! You passed!
TO PASS 80% or higher

Keep Learning

GRADE
90.9%

Week 6 - Problem Set

LATEST SUBMISSION GRADE
90.9%

1. Recall that with symmetric ciphers it is possible to encrypt a 32-bit message and obtain a 32-bit ciphertext (e.g. with the one time pad or with a nonce-based system). Can the same be done with a public-key system?

1 / 1 point

- [x] No, public-key systems with short ciphertexts can never be secure.
- [ ] Yes, when encrypting a short plaintext the output of the public-key encryption algorithm can be truncated to the length of the plaintext.
- [ ] It is not possible with the ElGamal system, but may be possible with other systems.
- [ ] Yes, the RSA-OAEP system can produce 32-bit ciphertexts.

> **Correct**
> An attacker can use the public key to build a dictionary of all 2^{32} ciphertexts of length 32 bits along with their decryption and use the dictionary to decrypt any captured ciphertext.

2. Let (Gen, E, D) be a semantically secure public-key encryption system. Can algorithm E be deterministic?

1 / 1 point

- [ ] Yes, some public-key encryption schemes are deterministic.
- [x] No, semantically secure public-key encryption must be randomized.
- [ ] No, but chosen-ciphertext secure encryption can be deterministic.
- [ ] Yes, RSA encryption is deterministic.

> **Correct**
> That's correct since otherwise an attacker can easily break semantic security.

3. Let (Gen, E, D) be a chosen ciphertext secure public-key encryption system with message space $\{0,1\}^{128}$. Which of the following is also chosen ciphertext secure?

0 / 1 point

- [ ] (Gen, E', D') where
  $E'(\mathrm{pk}, m) = (E(\mathrm{pk}, m),\ E(\mathrm{pk}, m))$
  and $D'(\mathrm{sk}, (c_1, c_2)) = \begin{cases} D(\mathrm{sk}, c_1) & \text{if } D(\mathrm{sk}, c_1) = D(\mathrm{sk}, c_2) \\ \perp & \text{otherwise} \end{cases}$.
- [ ] (Gen, E', D') where
  $E'(\mathrm{pk}, m) = (E(\mathrm{pk}, m),\ 0^{128})$
  and $D'(\mathrm{sk}, (c_1, c_2)) = \begin{cases} D(\mathrm{sk}, c_1) & \text{if } c_2 = 0^{128} \\ \perp & \text{otherwise} \end{cases}$.
- [x] (Gen, E', D') where
  $E'(\mathrm{pk}, m) = \left[c \leftarrow E(\mathrm{pk}, m),\ \text{output } (c, c)\right]$
  and $D'(\mathrm{sk}, (c_1, c_2)) = \begin{cases} D(\mathrm{sk}, c_1) & \text{if } c_1 = c_2 \\ \perp & \text{otherwise} \end{cases}$.

> **Correct**
> This construction is chosen-ciphertext secure.
> An attack on (Gen, E', D) gives an attack on (Gen, E, D).

- [x] (Gen, E', D') where
  $E'(\mathrm{pk}, m) = (E(\mathrm{pk}, m),\ E(\mathrm{pk}, 0^{128}))$
  and $D'(\mathrm{sk}, (c_1, c_2)) = D(\mathrm{sk}, c_1)$.

> **! This should not be selected**
> This construction is not chosen-ciphertext secure.
> An attacker can output two messages $m_0 = 0^{128}$ and $m_1 = 1^{128}$ and be given back a challenge ciphertext (c_1, c_2). The attacker would then ask for the decryption of $(c_1, E(pk, 1^{128})$ and be given in response m_0 or m_1 thereby letting the attacker win the game. Note that the decryption query is valid since it is different from the challenger ciphertext (c_1, c_2).

4. Recall that an RSA public key consists of an RSA modulus N and an exponent e. One might be tempted to use the same RSA modulus in different public keys. For example, Alice might use $(N,3)$ as her public key while Bob may use $(N,5)$ as his public key. Alice's secret key is $d_a = 3^{-1} \bmod \varphi(N)$ and Bob's secret key is $d_b = 5^{-1} \bmod \varphi(N)$.
In this question and the next we will show that it is insecure for Alice and Bob to use the same modulus N. In particular, we show that either user can use their secret key to factor N. Alice can use the factorization to compute $\varphi(N)$ and then compute Bob's secret key.
As a first step, show that Alice can use her public key $(N,3)$ and private key d_a to construct an integer multiple of $\varphi(N)$.
Which of the following is an integer multiple of $\varphi(N)$?

1 / 1 point

- [ ] $5d_a - 1$
- [ ] $d_a + 1$
- [x] $3d_a - 1$
- [ ] $3d_a + 1$

> **Correct**
> Since $d_a = 3^{-1} \bmod \varphi(N)$ we know that $3d_a = 1 \bmod \varphi(N)$ and therefore $3d_a - 1$ is divisibly by $\varphi(N)$.

5. Now that Alice has a multiple of $\varphi(N)$ let's see how she can factor $N = pq$. Let x be the given muliple of $\varphi(N)$.
Then for any g in $\mathbb{Z}_N^*$ we have $g^x = 1$ in $\mathbb{Z}_N$. Alice chooses a random g in $\mathbb{Z}_N^*$ and computes the sequence
$g^x, g^{x/2}, g^{x/4}, g^{x/8} \ldots$ **in $\mathbb{Z}_N$**
and stops as soon as she reaches the first element $y = g^{x/2^i}$ such that $y \neq 1$ (if she gets stuck because the exponent becomes odd, she picks a new random g and tries again). It can be shown that with probability $1/2$ this y satisfies

$$\begin{cases} y = 1 \bmod p, \text{ and} \\ y = -1 \bmod q \end{cases} \quad \text{or} \quad \begin{cases} y = -1 \bmod p, \text{ and} \\ y = 1 \bmod q \end{cases}$$

How can Alice use this y to factor N?

1 / 1 point

- [ ] compute $gcd(N+1,\ y)$
- [ ] compute $gcd(N,\ y)$
- [x] compute $gcd(N,\ y-1)$

> **Correct**
> We know that $y-1$ is divisible by p or q, but not divisible by the other. Therefore, $gcd(N,\ y-1)$ will output a non-trivial factor of N.

- [ ] compute $gcd(N,\ y^2-1)$
- [ ] compute $gcd(N,\ 2y-1)$

6. In standard RSA the modulus N is a product of two distinct primes. Suppose we choose the modulus so that it is a product of three distinct primes, namely $N = pqr$. Given an exponent e relatively prime to $\varphi(N)$ we can derive the secret key as $d = e^{-1} \bmod \varphi(N)$. The public key (N,e) and secret key (N,d) work as before. What is $\varphi(N)$ when N is a product of three distinct primes?

1 / 1 point

- [x] $\varphi(N) = (p-1)(q-1)(r-1)$
- [ ] $\varphi(N) = (p-1)(q-1)(r+1)$
- [ ] $\varphi(N) = (p+1)(q+1)(r+1)$
- [ ] $\varphi(N) = (p-1)(q-1)$

> **Correct**
> When is a product of distinct primes then $|\mathbb{Z}_N^*|$ satisfies $|\mathbb{Z}_N^*| = |\mathbb{Z}_p^*| \cdot |\mathbb{Z}_q^*| \cdot |\mathbb{Z}_r^*| = (p-1)(q-1)(r-1)$.

7. An administrator comes up with the following key management scheme: he generates an RSA modulus N and an element s in $\mathbb{Z}_N^*$. He then gives user number i the secret key $s_i = s^{r_i}$ in $\mathbb{Z}_N$ where r_i is the i'th prime (i.e. 2 is the first prime, 3 is the second, and so on).
Now, the administrator encrypts a file that is accssible to users i, j and t with the key $k = s^{r_i r_j r_t}$ in $\mathbb{Z}_N$.
It is easy to see that each of the three users can compute k. For example, user i computes k as $k = (s_i)^{r_j r_t}$. The administrator hopes that other than users i, j and t, no other user can compute k and access the file.
Unfortunately, this system is terribly insecure. Any two colluding users can combine their secret keys to recover the master secret s and then access all files on the system. Let's see how. Suppose users 1 and 2 collude. Because r_1 and r_2 are distinct primes there are integers a and b such that $ar_1 + br_2 = 1$.
Now, users 1 and 2 can compute s from the secret keys s_1 and s_2 as follows:

1 / 1 point

- [ ] $s = s_1^b \cdot s_2^a$ in $\mathbb{Z}_N$.
- [ ] $s = s_1^b / s_2^a$ in $\mathbb{Z}_N$.
- [ ] $s = s_2^a$ in $\mathbb{Z}_N$.
- [x] $s = s_1^a \cdot s_2^b$ in $\mathbb{Z}_N$.

> **Correct**
> $s = s_1^a \cdot s_2^b = s^{r_1 a} \cdot s^{r_2 b} = s^{r_1 a + r_2 b} = s$ in $\mathbb{Z}_N$.

8. Let G be a finite cyclic group of order n and consider the following variant of ElGamal encryption in G:

- **Gen: choose a random generator g in G and a random x in $\mathbb{Z}_n$. Output $\mathrm{pk} = (g, h = g^x)$ and $\mathrm{sk} = (g, x)$.**
- **$E(\mathrm{pk}, m \in G)$: choose a random r in $\mathbb{Z}_n$ and output $(g^r,\ m \cdot h^r)$.**
- **$D(\mathrm{sk}, (c_0, c_1))$: output c_1 / c_0^x.**

This variant, called plain ElGamal, can be shown to be semantically secure under an appropriate assumption about G. It is however not chosen-ciphertext secure because it is easy to compute on ciphertexts. That is, let (c_0, c_1) be the output of $E(\mathrm{pk}, m_0)$ and let (c_2, c_3) be the output of $E(\mathrm{pk}, m_1)$. Then just given these two ciphertexts it is easy to construct the encryption of $m_0 \cdot m_1$ as follows:

1 / 1 point

- [x] $(c_0 c_2,\ c_1 c_3)$ is an encryption of of $m_0 \cdot m_1$.
- [ ] $(c_0 + c_2,\ c_1 + c_3)$ is an encryption of of $m_0 \cdot m_1$.
- [ ] $(c_0 / c_3,\ c_1 / c_2)$ is an encryption of of $m_0 \cdot m_1$.
- [ ] $(c_0 c_3,\ c_1 c_2)$ is an encryption of of $m_0 \cdot m_1$.

> **Correct**
> Indeed, $(c_0 c_2,\ c_1 c_3) = (g^{r_0 + r_1},\ m_0 m_1 h^{r_0 + r_1})$, which is a valid encryption of $m_0 m_1$.

9. Let G be a finite cyclic group of order n and let $\mathrm{pk} = (g, h = g^a)$ and $\mathrm{sk} = (g, a)$ be an ElGamal public/secret key pair in G as described in [Segment 12.1](). Suppose we want to distribute the secret key to two parties so that both parties are needed to decrypt. Moreover, during decryption the secret key is never re-constructed in a single location. A simple way to do so it to choose random numbers a_1, a_2 in $\mathbb{Z}_n$ such that $a_1 + a_2 = a$. One party is given a_1 and the other party is given a_2. Now, to decrypt an ElGamal ciphertext (u, c) we send u to both parties. What do the two parties return and how do we use these values to decrypt?

1 / 1 point

- [ ] party 1 returns $u_1 \leftarrow u^{(a_1^2)}$, party 2 returns $u_2 \leftarrow u^{(a_2^2)}$ and the results are combined by computing $v \leftarrow u_1 \cdot u_2$.
- [ ] party 1 returns $u_1 \leftarrow u^{a_1}$, party 2 returns $u_2 \leftarrow u^{a_2}$ and the results are combined by computing $v \leftarrow u_1 + u_2$.
- [x] party 1 returns $u_1 \leftarrow u^{a_1}$, party 2 returns $u_2 \leftarrow u^{a_2}$ and the results are combined by computing $v \leftarrow u_1 \cdot u_2$.
- [ ] party 1 returns $u_1 \leftarrow u^{a_1}$, party 2 returns $u_2 \leftarrow u^{a_2}$ and the results are combined by computing $v \leftarrow u_1 / u_2$.

> **Correct**
> Indeed, $v = u_1 \cdot u_2 = g^{a_1 + a_2} = g^a$ as needed for decryption. Note that the secret key was never re-constructed for this distributed decryption to work.

10. Suppose Alice and Bob live in a country with 50 states. Alice is currently in state $a \in \{1, \ldots, 50\}$ and Bob is currently in state $b \in \{1, \ldots, 50\}$. They can communicate with one another and Alice wants to test if she is currently in the same state as Bob. If they are in the same state, Alice should learn that fact and otherwise she should learn nothing else about Bob's location. Bob should learn nothing about Alice's location.
They agree on the following scheme:

- **They fix a group G of prime order p and generator g of G**
- **Alice chooses random x and y in $\mathbb{Z}_p$ and sends to Bob $(A_0, A_1, A_2) = (g^x,\ g^y,\ g^{xy+a})$**
- **Bob choose random r and s in $\mathbb{Z}_p$ and sends back to Alice $(B_1, B_2) = (A_1^r g^s,\ (A_2/g^b)^r A_0^s)$**

What should Alice do now to test if they are in the same state (i.e. to test if $a = b$) ?
Note that Bob learns nothing from this protocol because he simply recieved a plain ElGamal encryption of g^a under the public key g^x. One can show that if $a \neq b$ then Alice learns nothing else from this protocol because she recieves the encryption of a random value.

1 / 1 point

- [ ] Alice tests if $a = b$ by checking if $B_2^x B_1 = 1$.
- [ ] Alice tests if $a = b$ by checking if $B_1 / B_2^x = 1$.
- [x] Alice tests if $a = b$ by checking if $B_2 / B_1^x = 1$.
- [ ] Alice tests if $a = b$ by checking if $B_2 B_1^x = 1$.

> **Correct**
> The pair (B_1, B_2) from Bob satisfies $B_1 = g^{yr+s}$ and $B_2 = (g^x)^{yr+s} g^{r(a-b)}$. Therefore, it is a plain ElGamal encryption of the plaintext $g^{r(a-b)}$ under the public key (g, g^x). This plaintext happens to be 1 when $a = b$.
> The term B_2 / B_1^x computes the ElGamal plaintext and compares it to 1.
> Note that when $a \neq b$ the $r(a-b)$ term ensures that Alice learns nothing about b other than the fact that $a \neq b$.
> Indeed, when $a \neq b$ then $r(a-b)$ is a uniform non-zero element of $\mathbb{Z}_p$.

11. What is the bound on d for Wiener's attack when N is a product of three equal size distinct primes?

1 / 1 point

- [ ] $d < N^{1/5}/c$ for some constant c.
- [ ] $d < N^{1/3}/c$ for some constant c.
- [x] $d < N^{1/6}/c$ for some constant c.
- [ ] $d < N^{1/4}/c$ for some constant c.

> **Correct**
> The only change to the analysis is that $N - \varphi(N)$ is now on the order of $N^{2/3}$. Everything else stays the same. Plugging in this bound gives the answer. Note that the bound is weaker in this case compared to when N is a product of two primes making the attack less effective.