VE475 Intro to Cryptography Homework 7

Taoyue Xia, 518370910087

2021/07/07

Ex.1 — Cramer-Shoup cryptosystem

1. The Cramer–Shoup system is an asymmetric key encryption algorithm, which is the extension of the Elgamal cryptosystem. Its construction contains three procedures: **key generation**, **encryption** and **decryption**.

   (i) **Key generation**:

   - Alice generates a cyclic group G of order q with two distinct, random generators g_1, g_2.
   - Alice choose 5 random numbers (x_1, x_2, y_1, y_2, z) from $\{0, \cdots, q-1\}$.
   - Alice computes $c = g_1^{x_1} g_2^{x_2}$, $d = g_1^{y_1} g_2^{y_2}$, $e = g_1^z$.
   - Alice publishes (c, d, h), along with G, q, g_1, g_2 as her public key. Alice keeps x_1, x_2, y_1, y_2, z as her secret key.

   (ii) **Encryption**:

   To encrypt a message m to Alice under the public key Alice releases, Bob should do as following:

   - Bob converts m into an element of G.
   - Bob chooses a random k from $0, \cdots, q-1$, then calculates:
     - $u_1 = g_1^k$, $u_2 = g_2^k$.
     - $e = h^k m$.

- $\alpha = H(u_1, u_2, e)$, where H is a collision-resistant cryptographic hash function.
- $v = c^k d^{k\alpha}$.

- After that, Bob sends ciphertext (u_1, u_2, e, v) to Alice.

(iii) **Decryption**:

To decrypt a ciphertext (u_1, u_2, e, v), with her secret key (x_1, x_2, y_1, y_2, z), Alice does as follows:

- Alice computes $\alpha = H(u_1, u_2, e)$ and verifies that $u_1^{x_1} u_2^{x_2} (u_1^{y_1} u_2^{y_2})^{\alpha} = v$. If this test fails, further decryption is aborted as Bob doesn't pass the verification.
- Otherwise, Alice computes the plaintext as $m = e/(u_1^z)$.

The decryption stage correctly decrypts any properly-formed ciphertext, since:

$$u_1^z = g_1^{kz} = h^k, \text{ and } m = e/h^k$$

2. The adaptive chosen ciphertext attack can be applied on those systems that have the property of ciphertext malleability. It can change the ciphertext used to decrypt each time following a specific pattern, and gradually reveal information about the encrypted message or the decryption key. However, for the Cramer-Shoup cryptosystem, it first check the hash value of the ciphertext. While the attacker doesn't know the exact message, he cannot generate the corresponding $e = h^k m$. Since the hash function is collision-resistant, it is impossible for the attacker to pass the verification, thus he can never figure out the corresponding message. That's why it is secure against adaptive chosen ciphertext attack.

3. **Similarities**: Both the Cramer-Shoup cryptosystem and Elgamal are public cryptosystems, they both base the encryption in a cyclic group G, and uses private keys to decrypt others' ciphertexts. Their private key settings are both based on the difficulty of solving the DLP problem.

   Differences: The Cramer-Shoup cryptosystem uses a collision-resistant hash function to first check the identity of the sender, if passed, then going to decrypt the message, but the Elgamal doesn't have this process, just decrypting the ciphertext straightforward. Therefore, the Cramer-Shoup cryptosystem is more secure.

Ex2 — Simple questions

1. Since p is a prime and $p \nmid \alpha$, according to Euler's Theorem, We know that $\alpha^{p-1} \equiv 1 \ mod \ p$, so for the hash function $h(x) \equiv \alpha^x \ mod \ p$, we can easily find $x' = x + p - 1$ for a given x such that $h(x') = h(x)$, which is not second pre-image resistant. So it is not a good hash function.

2. First we can compute $2^{30}_{10} = 4 \cdot 16^7 = 40000000_{16}$, and $\sqrt{2}_{10} = 1.6A09E667F_{16}$. Therefore, we can express $\lfloor 2^{30}\sqrt{2} \rfloor$ as:

$$\lfloor 2^{30}\sqrt{2} \rfloor_{10} = 5A827999_{16}$$

Similarly, we can compute the rest three expressions in hexadecimal as:

$$\lfloor 2^{30}\sqrt{3} \rfloor_{10} = 6ED9EBA1_{16}$$

$$\lfloor 2^{30}\sqrt{5} \rfloor_{10} = 8F1BBCDC_{16}$$

$$\lfloor 2^{30}\sqrt{10} \rfloor_{10} = CA62C1D6_{16}$$

I find that the for $i = 2$, the expression is equal to $K_0, \cdots, K_{19}$. For $i = 3$, the expression is equal to $K_{20}, \cdots, K_{39}$. For $i = 5$, the expression is equal to $K_{40}, \cdots, K_{59}$. For $i = 10$, the expression is equal to $K_{60}, \cdots, K_{79}$.

Ex3 — Birthday paradox

1. For $f(x) = \ln(1 - x)$, we take its derivative as $f'(x) = -\frac{1}{1-x}$, then we can obviously find that $f'(x) < 0$, for $x \in [0, \frac{1}{2}]$. So we can conclude that $f(x)$ is strictly decreasing on $[0, \frac{1}{2}]$. For $g(x) = \ln(1 - x) + x + x^2$, we first take its derivative:

$$g'(x) = -\frac{1}{1-x} + 1 + 2x$$

We can easily find that $g'(0) = g'(\frac{1}{2}) = 0$, so we need to show the second-order derivative.

$$g''(x) = -\frac{1}{(1-x)^2} + 2$$

So we know that $g''(0) = 1$, which means $g(0)$ is a local minimum. And $g''(\frac{1}{2}) = -2$, which means $g(\frac{1}{2})$ is a local maximum. Therefore, for $x \in [0, \frac{1}{2}]$, $g(x) \geq g(0) = 0$, so we can find that:

$$g(x) \geq 0 \Rightarrow \ln(1-x) + x + x^2 \geq 0 \Rightarrow -x - x^2 \leq \ln(1-x)$$

Using the above approach, construct a new function $h(x) = \ln(1-x) + x$ to complete the proof. First take the derivative as:

$$h'(x) = -\frac{1}{1-x} + 1$$

Then we find that only $h'(0) = 0$, while for $x \in (0, \frac{1}{2}]$, $h'(x) < 0$. So we know that $h(0)$ is a global maximum, thus for $x \in [0, \frac{1}{2}]$, $h(x) \leq 0$, so we can find that:

$$h(x) \leq 0 \Rightarrow \ln(1-x) + x \leq 0 \Rightarrow \ln(1-x) \leq x$$

Therefore, combined with the above analysis on $g(x)$, we can conclude that:

$$-x - x^2 \leq \ln(1-x) \leq -x$$

And the equality holds when $x = 0$. Proof done.

2. Since $r \leq n/2$, for all $j \in \{1, \cdots, r-1\}$, we know that $\frac{j}{n} < \frac{1}{2}$. Thus for each j:

$$-\frac{j}{n} - (\frac{j}{n})^2 \leq \ln(1 - \frac{j}{n}) \leq \frac{j}{n}$$

Take the sum from $j = 1$ to $j = r-1$, we can get:

$$\sum_{j=1}^{r-1}(\frac{j}{n} - (\frac{j}{n})^2) \leq \sum_{j=1}^{r-1} \ln(1 - \frac{j}{n}) \leq \sum_{j=1}^{r-1} \frac{j}{n}$$

We can use the following two transformations:

$$\sum_{j=1}^{r-1} \frac{j}{n} = \frac{r(r-1)}{2} \cdot \frac{1}{n} = \frac{r(r-1)}{2n}$$

$$\sum_{j=1}^{r-1}(\frac{j}{n})^2 = \frac{r(r-1)(2r-1)}{6} \cdot \frac{1}{n} < \frac{r^3}{3n}, \ \forall r > 1$$

Therefore, we can conclude that:

$$-\frac{(r-1)r}{2n}-\frac{r^3}{3n}\leq\sum_{j=1}^{r-1}\ln(1-\frac{j}{n})\leq-\frac{(r-1)r}{2n}$$

Proof done.

3. For the inequality derived in problem 2, we can take the exponent of all sides to obtain:

$$e^{-\frac{(r-1)r}{2n}-\frac{r^3}{3n}}\leq\prod_{j=1}^{r-1}(1-\frac{j}{n})\leq e^{-\frac{(r-1)r}{2n}}$$

Take $\lambda=\frac{r^2}{2n}$, $c_1=\sqrt{\frac{\lambda}{2}}-\frac{(2\lambda)^{\frac{3}{2}}}{3}$, and $c_2=\sqrt{\frac{\lambda}{2}}$, we can have:

$$-\lambda+\frac{c_1}{\sqrt{n}}=-\frac{r^2}{2n}+\frac{r}{2n}-\frac{r^3}{3n}=-\frac{(r-1)r}{2n}-\frac{r^3}{3n}$$

$$-\lambda+\frac{c_2}{\sqrt{n}}=-\frac{r^2}{2n}+\frac{r}{2n}=-\frac{(r-1)r}{2n}$$

So we can conclude that:

$$e^{-\lambda}e^{\frac{c_1}{\sqrt{n}}}\leq\prod_{j=1}^{r-1}(1-\frac{j}{n})\leq e^{-\lambda}e^{\frac{c_2}{\sqrt{n}}}$$

Proof done.

4. If $\lambda<\frac{n}{8}$ is a constant, we can obtain that:

$$\lim_{n\to\infty}\frac{c_1}{\sqrt{n}}=0-0=0$$

$$\lim_{n\to\infty}\frac{c_2}{\sqrt{n}}=0$$

Thus we know that when $\lambda<\frac{n}{8}$ is a small constant, $e^{\frac{c_1}{\sqrt{n}}}\approx 0$ and $e^{\frac{c_2}{\sqrt{n}}}\approx 0$. Therefore, we can finally get:

$$e^{-\lambda}\leq\prod_{j=1}^{r-1}(1-\frac{j}{n})\leq e^{-\lambda}\Rightarrow\prod_{j=1}^{r-1}(1-\frac{j}{n})\approx e^{-\lambda}$$

Proof done.

Ex4 — Birthday attack

1. We can use 1 to subtract the possibility of no two plates have the same 3-digit number to get the answer:

$$P = 1 - \prod_{i=1}^{39} \frac{1000 - i}{1000} = 0.54637$$

   So the probability of seeing two plates with the same 3-digit number is 0.54637.

2. Given a plate with number 123, we can calculate the probability as:

$$P = 40 \cdot \frac{1}{1000} \cdot (\frac{999}{1000})^{39} = 0.03847$$

   So the probability is 0.03847.

3. From problem 1, we know that after trying for many times, it is getting gradually more likely that a collision would be found in a finite set. However, from problem2, we can also see that if one is given a value or message, it is very hard to find a collision. Therefore, in chapter 5, Alice can change her message or contract a little bit, like adding a comma or space, then the hash value would be completely different, and it is quite impossible for Eve to find the collision for the message.

Ex5 — Faster multiple modular exponentiation

1. For computing $\alpha^a \ mod \ n$, its time complexity is $O(\log(a))$. Similarly, for computing $\beta^b \ mod \ n$, its time complexity is $O(\log(b))$. Therefore, the time complexity for computing for both the two is $O(\log(a) + \log(b)) = O(\log(ab))$.

2. The algorithm is shown below on the next page.

3. For the common algorithm mentioned in the stem, We need $2l$ squaring and multiplications to compute $\alpha^a \beta^b \ mod \ n$. However, If we know the product of $\alpha\beta$, we just need l squaring and multiplications.

4. The implementation is in folder **ex5**, with a readme file inside.

Algorithm 1 Square and multiply

input: α, β two integers, and n, a, b three positive integers

output: $x = \alpha^a \beta^b \ mod \ n$

$d_a \leftarrow (d_{i-1}, \cdots, d_0)_2$
$d_b \leftarrow (d_{j-1}, \cdots, d_0)_2$
$k \leftarrow \text{MAX}(i, j)$
$flag \leftarrow 1$ **if** $k = i$, **else** $flag \leftarrow 0$
$power \leftarrow 1$
for $m = k - 1$ **to** 0 **do**
    $power \leftarrow power \cdot power$ **mod** n
    if $flag = 1$ **and** $m \geq j$ **then**
        if $d_a[m] = 1$ **then**
            $power \leftarrow (\alpha \cdot power)$ **mod** n
        end if
        continue
    end if
    if $flag = 0$ **and** $m \geq i$ **then**
        if $d_b[m] = 1$ **then**
            $power \leftarrow (\beta \cdot power)$ **mod** n
        end if
        continue
    end if
    if $d_a[m] = 1$ **and** $d_b[m] = 1$ **then**
        $power \leftarrow (\alpha\beta \cdot power)$ **mod** n
    else if $d_a[m] = 1$ **then**
        $power \leftarrow (\alpha \cdot power)$ **mod** n
    else if $d_b[m] = 1$ **then**
        $power \leftarrow (\beta \cdot power)$ **mod** n
    end if
end for
return $power$