CSE5014 · Cryptography and Network Security

Assignment 4

Liu Leqi / 12011327

May 27, 2023

1 Q1

1. Assume H_a is not collision-resistant, then there must be a pair of collision (x, x') such that $H_a(x) = H_a(x')$, i.e., $H_1(x)||H_2(x) = H_1(x')||H_2(x')$, which means that both H_1 and H_2 are not collision-resistant, contradicting that H_1 and H_2 at least one is collision-resistant.
2. Assume H_a is not collision-resistant, then there must be a pair of collision (x, x') such that $H_a(x) = H_a(x')$, i.e.,

$$H_1(H_2(x)) = H_1(H_2(x'))$$
$$H_2(H_1(x)) = H_2(H_1(x'))$$

   If both H_1 and H_2 are collision-resistant, then from the above we can get

$$H_2(x) = H_2(x')$$
$$H_1(x) = H_1(x')$$

   which contradicts. If H_1 is collision-resistant while H_2 is not (the same when reversed), then from the above we can get

$$H_2(x) = H_2(x')$$
$$H_1(x) \neq H_1(x') \; or \; H_1(x) = H_1(x')$$

   which satisfies the assumption. However, H_a is not necessarily collision-resistant.
3. Assume H_a is not collision-resistant, then there must be a pair of collision (x, x') such that $H_a(x) = H_a(x')$, i.e.,

$$H_1(H_2(x)||x) = H_1(H_2(x')||x')$$
$$H_2(H_1(x)||x) = H_2(H_1(x')||x')$$

   If both H_1 and H_2 are collision-resistant, then from the above we can get

$$H_2(x)||x = H_2(x')||x'$$
$$H_1(x)||x = H_1(x')||x'$$

   Since $x \neq x'$, the equations cannot be established. If H_1 is collision-resistant while H_2 is not (the same when reversed), then from the above we can get

$$H_2(x)||x = H_2(x')||x'$$
$$H_1(x)||x \neq H_1(x')||x' \; or \; H_1(x)||x = H_1(x')||x'$$

   Since $x \neq x'$, the equations cannot be established. Therefore, H_a is not necessarily collision-resistant.

2 Q2

The attacker can query the MAC oracle with the messages $m_1 = x||y$ and $m_2 = x$, and get the corresponding tags t_1 and t_2. By the construction of Merkle-Damgard, we can know that $t_1 = h(t_2, y)$. So the attacker can output $(m', t') = (t_2||y, t_1)$ which passes the verification.

3 Q3

1. To prove that the set of QRs is a subgroup of $\mathbb{Z}_n^*$, we need to show that 4 properties hold: closure, identity, inverse and associativity.

   - Closure:
     Let y_1 and y_2 be two quadratic residues in $\mathbb{Z}_n^*$, then there exists $x_1, x_2 \in \mathbb{Z}_n^*$ such that $y_1 \equiv x_1^2 (mod\ n)$ and $y_2 \equiv x_2^2 (mod\ n)$. By the definition of quadratic residue, we have $y_1 y_2 \equiv x_1^2 x_2^2 (mod\ n) \equiv (x_1 x_2)^2 (mod\ n)$. Thus, $y_1 y_2$ is also a quadratic residue in $\mathbb{Z}_n^*$.

   - Identity:
     The identity element e in $\mathbb{Z}_n^*$ is equal to 1. Since $1 \equiv 1^2 (mod\ n)$, we know that 1 is indeed a quadratic residue in $\mathbb{Z}_n^*$.

   - Inverse:
     Let y be a quadratic residue in $\mathbb{Z}_n^*$. Then $\exists x \in \mathbb{Z}_n^*, y \equiv x^2 (mod\ n)$. We want to show that the inverse of y, denoted as y^{-1} with $yy^{-1} \equiv 1(mod\ n)$, is also a quadratic residue in $\mathbb{Z}_n^*$.
     If $\gcd(y, n) \neq 1$, then y does not have an inverse in $\mathbb{Z}_n^*$. Otherwise, by Bezout's identity, there exist $a, b \in \mathbb{Z}_n^*$ such that $ay + bn = 1$, yielding $ay = 1 - bn$, which means that $ay \equiv 1(mod\ n) \equiv 1^2(mod\ n)$ and thus ay is also a quadratic residue in $\mathbb{Z}_n^*$ (since $1 \in \mathbb{Z}_n^*$). Therefore, $y^{-1} \equiv a^2 (mod\ n)$ is also a quadratic residue in $\mathbb{Z}_n^*$.

   - Associativity:
     Let y_1, y_2 and y_3 be three quadratic residues in $\mathbb{Z}_n^*$, then there exists $x_1, x_2, x_3 \in \mathbb{Z}_n^*$ such that $y_1 \equiv x_1^2 (mod\ n)$, $y_2 \equiv x_2^2 (mod\ n)$, $y_3 \equiv x_3^2 (mod\ n)$. Then we have $(y_1 y_2) y_3 \equiv (x_1 x_2)^2 x_3^2 (mod\ n) \equiv x_1^2 (x_2 x_3)^2 (mod\ n) \equiv y_1 (y_2 y_3) (mod\ n)$.

2. - "only if".
     Suppose y is a QR in $\mathbb{Z}_p^*$, then $y = x^2$ for some $x \in \mathbb{Z}_p^*$. If $x = 1$, then $y = 1$ and thus $\log_g(y) = 0$ is even. If $x > 1$, then x is a generator of $\mathbb{Z}_p^*$ since p is a prime, and thus $\exists i \geq 0, x^i \equiv y \equiv x^2 (mod\ p)$, i.e., $(x^{i/2} + x)(x^{i/2} - x) \equiv 0(mod\ p)$. Since p is a prime, it is either $p | (x^{i/2} + x)$ or $p | (x^{i/2} - x)$. Therefore, i must be even so that $i/2$ is an integer. That is, $\log_g(y)$ is even.

   - "if".
     Suppose $\log_g(y)$ is even, i.e., $\exists k \in \mathbb{Z}, \log_g(y) = 2k \implies y = g^{2k}$. Let $x = g^k \in \mathbb{Z}_p^*$, then $y = x^2$ is a QR in $\mathbb{Z}_p^*$.

4 Q4

Suppose the generator of $\mathbb{Z}_N$ is g, then for any $h \in \mathbb{Z}_N$, its discrete logarithm is x, i.e., $gx \equiv h(mod\ N)$. By the extended Euclidean algorithm, we can find the inverse of g module N, and the inverse is actually the discrete logarithm we need. Since the extended Euclidean algorithm is efficient, the discrete logarithm problem is easy under this situation.

5 Q5

1.
$$
\begin{aligned}
&f(1) = 1 \quad f(2) = 4 \quad f(3) = 9 \quad f(4) = 16 \\
&f(5) = 8 \quad f(6) = 2 \quad f(7) = 15 \quad f(8) = 13 \\
&f(9) = 13 \quad f(10) = 15 \quad f(11) = 2 \quad f(12) = 8 \\
&f(13) = 16 \quad f(14) = 9 \quad f(15) = 4 \quad f(16) = 1
\end{aligned}
$$

   Hence $S = \{1, 2, 4, 8, 9, 13, 15, 16\}$ and its size is 8.

2. Since 17 is a prime, there are 16 generators in $\mathbb{Z}_{17}^*$.

3. For any generator g of $\mathbb{Z}_{17}^*$, $g^i (i \in \{0, \dots, 16\})$ generates all elements in $\mathbb{Z}_{17}^*$ and thus it can generate all elements in S since $S \subset \mathbb{Z}_{17}^*$. Since a, b are randomly chosen from $\{0, \dots, 15\}$ (i.e., the multiplication ab has totally $16 + 15 + \dots + 1 = 136$ choices) and only $g^{ab} \in S$ satisfies the requirement (i.e., the multiplication ab has 8 choices), the probability is $\frac{8}{136} = \frac{1}{17}$.

6 Q6

We can construct A' as below:

1. Queries A for the square roots of the element $x \leftarrow_R \mathbb{Z}_N^*$. Continuing querying until no error returned.
2. Retrieves 4 square roots $\pm a, \pm b$. Suppose a, b are 2 non-trivial square roots, i.e., $a \not\equiv \pm b (mod\ N)$.
3. Computes $p = \gcd(N, a + b)$ and $q = \gcd(N, a - b)$ by using Euclidean algorithm.
4. Outputs p, q as the factors of N.

We can validate it as below:
Since

$$
\begin{aligned}
&a^2 \equiv b^2 (mod\ N) \\
&a \not\equiv b (mod\ N) \\
&a \not\equiv -b (mod\ N)
\end{aligned}
$$

we have

$$
\begin{aligned}
&pq \mid (a + b)(a - b) \\
&pq \nmid (a - b) \\
&pq \nmid (a + b)
\end{aligned}
$$

Since both p and q are primes, $(a + b)(a - b)$ has factors p and q. But neither $a + b$ nor $a - b$ contain the factors of both p and q. Hence $a + b$ and $a - b$ must each contain factors of exactly one of $\{p, q\}$. Thus, $\{\gcd(pq, a + b), \gcd(pq, a - b)\} = \{p, q\}$

7 Q7

The attacker can query the sign oracle with distinct messages m_1 and m_2 such that $m = (m_1 \cdot m_2) \mod N$, and then retrieve the signatures $\sigma_1 = m_1^d \mod N$ and $\sigma_2 = m_2^d \mod N$. Then the attacker outputs (m, σ), where $\sigma = (\sigma_1 \cdot \sigma_2) \mod N$, which can pass the verification since

$$
\sigma^e = (\sigma_1 \cdot \sigma_2)^e = m_1^{ed} \cdot m_2^{ed} = m_1 \cdot m_2 = m \mod N
$$

8 Q8

For a fixed cyclic group G and its generator g, define $DH_g(h_1, h_2) = DH_g(g^x, g^y) = g^{xy}$.

- Discrete logarithm (DLog) problem: Given g, h, compute $\log_g h$.
- Computational Diffie-Hellman (CDH) problem: Given g, h_1, h_2, compute $DH_g(h_1, h_2)$.
- Decisional Diffie-Hellman (DDH) problem: Given g, h_1, h_2, distinguish $DH_g(h_1, h_2)$ from a uniform element of G.

DDH is stronger than CDH, and CDH is stronger than DLog.

9 Q9

1. In the experiment, the attacker can query the encryption oracle with plaintexts m_1, m_2 and retrieve the corresponding ciphertexts c_1, c_2. Then the attacker outputs 2 messages $m_1' = m_1, m_2' = \alpha m_2$ for arbitrary α to the challenger and retrieves a ciphertext c. If $c = \alpha c_2$, then the attacker outputs 2; otherwise, outputs 1. At this time, the attacker will always succeed.

2. Suppose the hash function is H, then El Gamal signature on message m will be

$$\sigma = Sign_{sk}(m) = (r, s) = (g^k, k^{-1}(H(m) - rx) \mod (p-1))$$

   If the hash function H is collision-resistant, then $H(m)$ is indistinguishable with a random string r^*, and thus it is hard for the attacker to forge a message-signature pair to pass the verification.

3. We can select a message m' such that $m' \equiv m(\mod p-1)$, then query signing oracle for its signature $\sigma' = (r', s')$. Then the forged signature of m is actually $\sigma = \sigma'$.