**CSE 5014: Cryptography and Network Security**
**2023 Spring Semester Written Assignment # 4**
**Due: May 30th, 2023, please submit at the beginning of class**

# Sample Solutions

**Q.1** Let $(Gen_1, H_1)$ and $(Gen_2, H_2)$ be hash functions from which *at least one* is collision-resistant. Decide for the following constructions whether the resulting hash function is *necessarily* collision-resistant and prove your answer (we assume that $Gen$ runs $Gen_1$ and $Gen_2$ to obtain a key $(s_1, s_2)$).

1. $H_a^{(s_1,s_2)}(x) := H_1^{s_1}(x)||H_2^{s_2}(x)$
2. $H_b^{(s_1,s_2)}(x) := H_1^{s_1}(H_2^{s_2}(x))||H_2^{s_2}(H_1^{s_1}(x))$
3. $H_c^{(s_1,s_2)}(x) := H_1^{s_1}(H_2^{s_2}(x)||x)||H_2^{s_2}(H_1^{s_1}(x)||x)$

**Solution:** We do not write the key explicitly for reasons of notations, since it is fixed and known by the adversary.

1. $H_a$ is collision-resistant: As $H_a(x) = H_a(y)$ implies $H_1(x) = H_1(y)$ and $H_2(x) = H_2(y)$, any adversary that finds a collision for $H_a$ can be used to construct an adversary that finds a collision for both $H_1$ and $H_2$.

2. $H_b$ is not collision-resistant: Assuming that $H_1$ is the constant zero function (for all keys), it follows that $H_b(x) = 0||H_2(0)$ for any $x$.

3. $H_c$ is collision-resistant: To see this we assume that there is a polynomial-time algorithm $A$ that finds a collision $x, y$ with non-negligible probability. We have $H_c(x) = H_c(y)$ which implies that

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

and

$$H_2(H_1(x)||x) = H_2(H_1(y)||y).$$

Since $(H_2(x)||x) \neq (H_2(x)||x)$ and $(H_1(x)||x \neq (H_1(x)||x)$, we found collisions for both $H_1$ and $H_2$. Therefore, the attacker $A$ can be used to construct an efficient adversary that finds a collision for both hash functions which contradicts the fact that at least one of $H_1$ and $H_2$ is collision-resistant.

□

**Q.2** Let $(Gen, H)$ be a collision-resistant hash function with inputs of arbitrary size. We define a MAC for arbitrary-length message by

$$Mac_{s,k}(m) = H^s(k||m).$$

Show that this is not a secure MAC if $H$ is constructed by the Merkle-Damgard transform from an arbitrary collision-resistant hash function $h$. (Assume that $s$ is known to the attacker)

**Solution:**

Let $h$ be the collision-resistant hash function from which $H$ is constructed by applying the Merkle-Damgard transform. We show that the MAC is not secure by constructing an adversary: we first query an arbitrary message $m$ of length $n$ and obtain

$$t = Mac_k(m) = H(k||m) = h(h(0^n||k)||m).$$

The adversary outputs $m' = m||t$ and $t' = h(t||t)$. Now it holds that

$$\begin{aligned} Mac_k m' &= H(k||m') \\ &= H(k||m||t) \\ &= h(h(h(0^n||k)||m)||t) \\ &= h(t||t) \\ &= t'. \end{aligned}$$

It follows that the adversary wins with probability 1, which is certainly not negligible.

□

**Q.3**

1. We say that a number $y \in \mathbb{Z}_n^*$ is a quadratic residue (QR) if $y = x^2$ for some $x \in \mathbb{Z}_n^*$. Prove that the set of QRs is a subgroup of $\mathbb{Z}_n^*$.

2. Let $p > 1$ be a prime. It can be shown that $\mathbb{Z}_p^*$ is a cyclic group, that is, there exists a generator $g \in \mathbb{Z}_p^*$ such that $\mathbb{Z}_p^* = \{g^1, g^2, \ldots, g^{p-1}\}$. For $y \in \mathbb{Z}_p^*$, let $\log_g(y)$ denote the smallest nonnegative integer $i$ for which $g^i = y$. For example $\log_g(1) = 0$, and $\log_g(g) = 1$. Show that $y$ is a QR in $\mathbb{Z}_p^*$ if and only if $\log_g(y)$ is an even number.

**Solution:**

1. Denote the set of QRs by $S$. For two elements $a, b \in S$, i.e., $a = x^2$ and $b = y^2$ for some $x, y \in \mathbb{Z}_n^*$, we then have $a \cdot b = x^2 y^2 = (xy)^2 \in S$. The closure property is proved.

   Clearly, for $a, b, c \in S$, we have $(a \cdot b) \cdot c = a \cdot (b \cdot c)$. And, $1 = 1^2 \in S$ is the identity elements.

   It then suffices to prove the existence of the inverse element for each element in $S$. For $a = x^2 \in S$, we have $x^2 \cdot x^{-2} = (x \cdot x^{-1})^2 = 1 \in S$, which means $a^{-1} = x^{-2} \in S$.

2. (Note: There are various ways to prove that $\mathbb{Z}_p^*$ is a cyclic group, which is equivalent to prove that there is an element in $\mathbb{Z}_p^*$ whose order is exactly $p - 1$) Suppose that $y \in S$, the set of QRs. This means $y = x^2$ for some element $x \in \mathbb{Z}_p^*$. For a fixed generator $g \in \mathbb{Z}_p^*$, suppose that $x = g^i$ for a certain $i$ with $0 \leq i \leq p - 2$. Then we have $y = x^2 = g^{2i}$, and equivalently, $\log_g(y) = 2i$, which is an even number.

   It remains to prove the "if" part. If $\log_g(y) = 2k$, i.e., $y = g^{2k}$. Then $y = (g^k)^2 = x^2 \in S$, where $x = g^k$.

□

**Q.4**

The discrete logarithm problem is easy in $\mathbb{Z}_N$ for any integer $N$ and for any generator. Explain this.

**Solution:**

The discrete logarithm problem in $\mathbb{Z}_N$ is that: given a generator $g$ and $y = xg \bmod N$, find $x$. Since $g$ is a generator, we have $\gcd(g, N) = 1$. Thus, $g$ has an inverse modulo $N$, and we can use the extended Euclidean algorithm to get the inverse $g^{-1}$ of $g$. The linear congruential equation can be thereby solved to get $x$.

$\square$

**Q.5**

Consider the cyclic group $\mathbb{Z}_{17}^* = \{1, 2, \ldots, 16\}$ and the mapping $f$ defined by $f(x) = x^2 \bmod 17$ for all $x$ in the group.

1. What is the size of the image set of $f$, i.e., the set $S = \{f(x) : x \in \mathbb{Z}_{17}^*\}$?
2. How many generators are there in $\mathbb{Z}_{17}^*$?
3. Pick a generator $g$. What is the probability that, for a randomly chosen $a, b \in \{0, 1, \ldots, 15\}$, the value of $g^{ab}$ is in $S$?

**Solution:**

1. $|S| = 8$, i.e., squaring is a 2-to-1 mapping over $\mathbb{Z}_{17}^*$.
2. This is equivalent to count the number of elements in $\mathbb{Z}_{17}^*$ whose order is 15. The number is $\phi(\phi(17)) = \phi(16) = 8$.
3. The probability is 3/4.

$\square$

**Q.6**

When $p$ and $q$ are distinct odd primes and $N = pq$, the elements in $\mathbb{Z}_N^*$ have either 0 or 4 square roots. A quarter (1/4) of the elements have 4 square roots; the rest have no square root. The four square roots of $x \in \mathbb{Z}_N^*$ look like $\pm a, \pm b$ (of course, $-a$ means $N - a$ since we always work modulo $N$). Suppose that you are given an efficient deterministic algorithm $A$ that, on input $x$ that has square roots, finds some square root. (If $x$ does not have a square root, it returns $\perp$.)

Use $A$ to make an efficient *probabilistic* algorithm $A'$ that factors $N$. (Hint: If you can find two square roots of a number, call them $a$ and $b$, which are not of the form $a = \pm b \bmod N$, then you can factor $N$. Show how.] **Note**: you only get to call $A$ as a black-box, so you don't know a *priori* which of the square roots it will find.

**Solution:**

Consider the following algorithm $A'$. On input $N$, it picks $x \leftarrow_R \mathbb{Z}_N^*$, and computes $y \leftarrow x^2 \bmod N$. (Note that sampling from $\mathbb{Z}_N^*$ is effectively done by sampling from $\mathbb{Z}_N$, because if you manged to find an $x$ that wasn't in $\mathbb{Z}_N^*$, then you could factor $N$ immediately) It then runs $z \leftarrow A(y)$. If $z = \pm x$ then it samples a new $x$ and repeats the process; it does this until $z \neq \pm x$. Once this loop is broken, we know that $z^2 = x^2 \bmod N$, or $z^2 - x^2 = 0 \bmod N$. By simple factoring this gives $(z - x)(z + x) = 0 \bmod N$ and since $z \neq \pm x$ we know that neither factor is zero. But then it must be the case that $\gcd((z - x) \bmod N, N)$ is one of the two factors of $N$, and the other is found immediately. Thus, $A'$ can factor $N$ in this way.

As for efficiency, we know that $A(y)$ is some fixed values, but we don't know which a priori. Since two of the four square roots of $y$ "work" for us, the probability that $A(y)$ returns one of these is $1/2$. Thus, $A'$ requires only two samples on average.

□

**Q.7** Show that the regular RSA signature scheme is *arbitrarily forgeable* (forging the signature of any challenge message $m$) if the attacker is allowed to ask the signing oracle. Note that the challenge message $m$ cannot be queried to the signing oracle. (Recall that the RSA signature is $m^d \bmod N$, where $d$ is the private key and $N = pq$)

**Solution:**

We forge the RSA signature $\sigma$ of any challenge message $m$ by querying the signing oracle the message $m' = m \cdot r^e \bmod N$, where $r \in_R \mathbb{Z}_N^*$ is chosen randomly. The signing oracle will return the signature $\sigma' = m'^d = m^d \cdot r \bmod N$. Then, we can compute the signature $\sigma = \sigma'/r = m^d \bmod N$.

□

**Q.8** Describe the discrete logarithm problem, Computational Diffie-Hellman (CDH) problem, and Decisional Diffie-Hellman (DDH) problem, respectively. State also the relation of the three assumptions of these three problems, i.e., which one is stronger than another.

**Solution:** The Dlog problem: Given a cyclic group $G$ and its generator $g$, for an element $h \in G$, compute $x$ such that $g^x = h$.

The CDH problem: Given $g, h_1, h_2$, compute $DH_g(h_1, h_2) = g^{xy}$, where $h_1 = g^x$ and $h_2 = g^y$.

The DDH problem: Given $g, h_1, h_2$, distinguish $DH_g(h_1, h_2)$ form a *uniform* element of $G$.

The DDH assumption is *stronger* than the CDH assumption, and the CDH assumption is *stronger* than the Dlog assumption.

$\square$

**Q.9** Recall the El Gamal encryption scheme: the public key is $(p, g, h)$, where $g$ is a generator of $\mathbb{Z}_p^*$ and $h = g^x$, and the private key is $x$; the encryption scheme is $Enc(m) = (g^y, h^y \cdot m)$, where $y \leftarrow_R \mathbb{Z}_p^*$; the decryption scheme is $Dec(c_1, c_2) = c_2/c_1^x$. The El Gamal signature scheme is: To sign on a message $m$, $k \leftarrow_R \mathbb{Z}_p^*$ with $\gcd(k, p-1) = 1$,

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

To verify a signature $\sigma = (r, s)$, it is accepted if $h^r r^s = g^m$.

(1) Show that El Gamal encryption scheme is *not* secure against the chosen ciphertext attack.

(2) Is El Gamal signature scheme secure against the chosen message attack (allowing to ask the signing oracle) if the *hash-and-sign paradigm* is used.

(3) Assume that the hash-and-sign paradigm is *not* used. Can we forge a signature for any given message $m$ by asking the signing oracle. Note that you cannot ask the oracle about the signature of $m$.

**Solution:**

(1) If such a oracle exists, then Eve who wants to decrypt the ciphertext $c = (c_1, c_2)$, with $c_1 = g^y$ and $c_2 = h^y \cdot m$, chooses random elements $k'$ and $m'$ and gets oracle to decrypt $c' = (c_1 \cdot g^{y'}, m \cdot m' \cdot h^{y+y'})$. Oracle send $mm'$, the plaintext of $c' = (g^{y+y'}, mm'h^{y+y'})$ to Eve. Eve simply divides by $m'$ and obtains the plaintext $m$ of $c$.

(2) When El Gamal signature used without a hash function ,it is existential forgeable as discussed in the slides. El Gamal signature scheme is secure against the chosen message attack if a hash function $h$ is applied to the original message, and it is the hash value that is signed. Thus, to forge the signature of a real message is not easy. Adversary Eve has to find some meaningful message $m'$ which $h(m') = m$. If $h$ is collision-resistant hash function, her probability of success is negligible.

(3) We can query the oracle for any message except $m$. Therefore, we design a forger algorithm as follows.

(i) Query the oracle for message $m'$, where $m/m' = u \bmod (p-1)$. (Oracle returns $(r = g^k \bmod p, s = k^{-1}(m' - rx) \bmod (p-1)$.

(ii) Compute $s' = su \bmod (p-1)$ and $r'$ such that $r' \equiv ru \bmod (p-1)$ and $r' \equiv r \bmod p$.

(iii) Now we check the verification step:

$$h^{r'} r^{s'} = h^{ru} r^{su} = (h^r r^s)^u = (g^{m'})^u = g^m.$$

(iv) Return $(m, r', s')$.

$\square$