Problem 1

Consider the following public-key encryption scheme. The public key is (G, q, g, h) and the private key is x, generated exactly as in the ElGamal encryption scheme. In order to encrypt a bit b, the sender does the following: If $b = 0$ then choose a random $y \in \mathbb{Z}_q$ and compute $c_1 = g^y$ and $c_2 = h^y$. The ciphertext is (c_1, c_2). If $b = 1$ then choose independent random $y, z \in \mathbb{Z}_q$, compute $c_1 = g^y$ and $c_2 = g^z$, and set the ciphertext equal to (c_1, c_2).

(a) Show that it is possible to decrypt efficiently (with some negligible error probability) given knowledge of the secret-key x.

Solution: Based on assumptions x is random and $h = g^x$. Assume we have received a ciphertext (c_1, c_2) and we know the secret-key x. We shall compute $\mathsf{Dec}(c_1) := c_1^x = (g^y)^x = (g^x)^y = h^y$.

Thus, if $\mathsf{Dec}(c_1) = c_2$, then $c_2 = h^y$ and we decrypt to $m = 0$. In this situation, we know that either $b = 0$ was encrypted (and the decryption was correct), or $b = 1$ was encrypted and z was chosen such that $g^z = h^y$, i.e. $z = xy$. In this case, the decryption was incorrect. But this will only happen with a probability $1/q$, which is negligible in n.

If $\mathsf{Dec}(c_1) \neq c_2$ we decrypt to $m = 1$. This decryption is always correct.

□

(b) Prove that this encryption scheme is CPA-secure if the Decisional Diffie-Hellman problem is hard.

Solution: Let Π denote the presented encryption scheme. We prove that Π has indistinguishable encryption in the presence of eavesdropper. Then is also CPA-secure.

Assume that there exists adversary A that can execute IND-EAV attack with non-negligible probability. Now consider the following PPT algorithm D that attempts to solve DDH problem relative to $G = (\mathbb{G}, q, g, g_1 = g^x, g_2 = g^y, g_3)$ (the notation is as in Th. 10.22 for ElGamal encryption).

1. Set $pk = (\mathbb{G}, q, g, g_1)$ and run $A(pk)$ with messages $m_0 = 0$, $m_1 = 1$.

2. Set $c_1 := g_2 = g^y$ and $c_2 := g_3$; i.e. $g_3 = g^{xy}$ or g^z for random z.
3. Run A and obtain an output bit b. Output whatever b is.

In case g_3 was a arbitrary value $g^z \neq g^{xy}$ the algorithm outputs 1 as shown in part (i). In case g_3 was g^{xy}, it satisfies that $c_1^x = (g_2)^x = g^{xy} = c_2$ and the algorithm thus outputs 0. We see, that D solves DDH with non-negligible error probability (the negligible probability of error was discussed in part (i)).

□

Problem 2

Consider the following language consisting of pairs of integers:

$$L = \{(N, x) | \text{ there exists } y, \text{ such that } y^2 = x \bmod N \text{ and } \gcd(N, x) = 1\}. \quad (1)$$

We will consider a zero-knowledge proof for L, i.e. the prover shows the verifier integers N, x and claims that x is quadratic residue modulo N. (An x for which a y exists such that $y^2 = x \bmod N$ is called a quadratic residue.) In the protocol below, $\quad_N = \{0, 1, ..., N-1\}$, and $\quad^*_N = \{x \in Z_N | gcd(x, N) = 1\}$.

1. V checks that $\gcd(x, N) = 1$ and rejects if this is not the case.
2. P chooses r at random in $\quad^*_N$ and sends $a = r^2 \bmod N$ to V .
3. V chooses a random bit b and sends it to P.
4. P sends $z = ry^b \bmod N$ to V, who checks that $z^2 = ax^b \bmod N$ and that $gcd(z, N) = 1$. V rejects if this is not the case and accepts otherwise.

Show that this protocol is a perfect zero-knowledge proof system for L. For this, you must show that the above protocol is:

Complete: If $(N, x) \in L$ then an honest verifier V will always accept if interacting with an honest prover P (who knows y s.t. $y^2 = x \bmod N$.) Also, convince yourself that P and V are efficient.

Sound: If $(N, x) \notin L$, then V will reject when interacting with any cheating prover P^* with "high" probability. Give a lower bound on this probability. Hint: If x is not a quadratic residue modulo N, then for any $a \in {}^{*}_{N}$, either a and/or $xa \bmod N$ is not a quadratic residue modulo N.

Perfect Zero-Knowledge: For any efficient verifier V^*, there exits an efficient simulator S such that $S(x, N)$ samples exactly the same distribution as the transcript of $V^*(x, N)$ interacting with $P(x, N, y)$. (Note that S does not get y.) Express the running time of your simulator S in terms of the running time of V^*.

Solution:

It is clear that the verifier and prover runs in polynomial time, i.e. are efficient, since all they need is square and multiply algorithms and GCD which can be done easily in, for instance, $O(n^2)$ time.

Completness: Let $(N, x) \in L$. Clearly holds that if both parties follow the protocol, then the verifier accepts with probability equal to 1. If $b = 0$ then P sends $z = r$. V checks $z^2 = r^2 (= a)$. Because $z = r \in \mathbb{Z}^*_{\mathbb{N}}$ we have that $gcd(r, N) = 1$. If $b = 1$ then P sends $z = ry$. V checks $z^2 = r^2 y^2 (= ax \bmod N)$. Also $gcd(ry, N) = 1$.

Soundness: Let P' be a prover strategy that makes the verifier accept with probability $> 1/2$. Then one of the possible first messages y sent by the prover P' must be such that V accepts for both choices $b = 0$ and $b = 1$. Let z_0, z_1 be the third round messages sent by P' in such cases. Then we have $y \equiv z_0^2$ and $xa \equiv z_1^2$, so that $x \equiv (z_0^{-1} z_1)^2$ and so x is a quadratic residue.

In other words, if x is not a quadratic residue, then P can answer only one of two possible challenges (only if $b = 0$), because in such a case y is a quadratic residue if and only if xy is not a quadratic residue. This means that P will be caught in any given round of the protocol with probability $1/2$. The overall probability that P decieves V is therefore $2^{-\log n} = 1/n$.

If there exists $v_0^2 = y^0 z$, $v_1^2 = yz$, then v_1/v_0 is a square root of y.

The required lower bound for the probability is $1/2$ as shown above.

Simulator: Let V' be an arbitrary verifier strategy. Given (N, x), the simulator algorithm for V' does the following

1. Pick uniformly at random $b \in \{0, 1\}$ and $r \in \mathbb{Z}_N^*$; pick randomness R for V'.
2. Set $a \equiv r^2 x^{-b} \pmod{N}$.
3. Infoke V' on the message a to obtain a bit B. If V', using randomness R, given a as first message, outputs b, then halt and output transcript: "V' selects randomness R, P sends a at first round, V' sends b at second round, P sends r at third round, V' acceps."
4. Go to 1.

Regardless of the choice of b, the simulator chooses y as a uniformly distributed quadratic residue in $\mathbb{Z}_N^*$. This means that a and b, as random variables, are statistically independent, and so that the second message of V' given a is also statistically indepenndent of b. No matter what the V' algorithm is, then, the simulator has probability $1/2$ of outputting a simulation in each attempt, and so the average number of attempts is just 2. Conditioned on a transcrippt being given in output, the distribution of the transcript is identical to the distribution of actual transcripts of the interaction between V' and P.

If the running time of V' is t then the simulator is polynomial in t. More specificaly, it's determined by the time required for the multiplication in $\mathbb{Z}_N^*$ which could be done by $O(n \log n)$ time.

□

Problem 3

Let f be a one-way permutation (as in Definition 6.2 of the textbook). Consider the following signature scheme for messages in the set $\{1, ..., n\}$:

- To generate keys, choose random $x \leftarrow \{0, 1\}^n$ and set $y := f^n(x)$. (Here $f^n(x)$ is defined as $f^n(x) := f(f(\ldots(f(x))))$.) The public key is y and the private key is x.

- To sign message $m \in \{1, ..., n\}$, output $\sigma = f^{n-m}(x)$ (where $f^0(x) := x$).
- To verify signature σ on message $m \in \{1, ..., n\}$ with respect to public key y, check whether $y = f^m(\sigma)$.

(a) Show that the above is not a one-time signature scheme. Given a signature on a message m, for what messages m' can an adversary output a forgery? (2 pt.)

Solution: Assume that the adversary knows m and its signature $\sigma = f^{n-m}(x)$. He can compute $\sigma_1 = f(\sigma) = f(f^{n-m}(x)) = f^{n-(m-1)}(x)$. He obtained a signature of $m_1 = m - 1$. Using this iteratively, he can compute a signature σ_i of any $m_i = m - i$ for $i = 1, \ldots, (m-1)$, i.e. he can output a forgery for messages $\{1, \ldots, m-1\}$. □

(b) Prove that no PPT adversary given a signature on m can output a forgery on any message $m' > m$ except with negligible probability. (3 pt.)

Solution: Idea: The approach is opposite as in 3(a). For computing a forgery on $m' < m$ we need to be able to compute f which is by definition easy. On the other hand, for computing a forgery on $m' > m$ we need to be able to compute f^{-1} which is by definition hard. If we can output a forgery, we can invert f. A contradiction.

Claim: *If f is a OWP, then so is f^k (f applied k times).* **Proof:** Suppose that we have an algorithm A that inverts f^k with non-negligible probability $\varepsilon(n)$. We will build a PPT algorithm B that inverts f also with probability $\varepsilon(n)$. Let B be as follows: on input $y \in \{0,1\}^n$, run $A(f^{k-1}(y))$. With probability $\varepsilon(n)$, obtain x such that $f^k(x) = f^{k-1}(y)$. Since f is a permutation f^{-1} is well-defined, and so $y = f^{1-k}(f^{k-1}(y)) = f^{1-k}(f^k(x)) = f(x)$, and so x is what we are looking for.

Now assume that C given a message m and its signature s is able to compute a signature s' of a message $m' > m$. Define d such that $m' = m + d$. Then $s' = f^{n-m'}(x) = f^{(n-m)-d}(x)$. Define $g := f^d$. By claim, g is a OWP. But $s' = g^{-1}(f^{n-m}(x)) = g^{-1}(s(x))$. We have computed a forgery with the same probability we are able to compute

an inversion of OWP g, but this is negligible as follows from the claim. In other words, if PPT C is able to compute forgery with non-negligible probability, he knows the inverse of g with non-negligible probabilitty. A contradiction.

□

(c) Suggest how to modify the scheme to obtain a one-time signature scheme. Prove its security. (3 pt.)
Hint: Include two values y, y' in the public key.

Solution: Consider the following modification:

- To generate keys, choose random $x_1, x_2 \leftarrow \{0,1\}^n$ and set $y_1 := f^n(x_1)$, $y_2 := f^n(x_2)$. The public key is (y_1, y_2) and the private key is (x_1, x_2).
- To sign message $m \in \{1, \ldots, n\}$, output $\sigma_1 = f^{n-m}(x_1)$, $\sigma_2 = f^m(x_2)$.
- To verify signature (σ_1, σ_2) on message $m \in \{1, \ldots, n\}$ with respect to public key (y_1, y_2), check whether $y_1 = f^m(\sigma_1)$ and $y_2 = f^{n-m}(\sigma_2)$.

The verification works: $f^m(\sigma_1) = f^m(f^{n-m}(x_1)) = f^n(x_1) = y_1$, $f^{n-m}(\sigma_2) = f^{n-m}(f^m(x_2)) = f^n(x_2) = y_2$.

The proof of security is twice applied 2b. Assume that an adversary C is given a messsage m and its signature (s_1, s_2). Assume that C is able to compute a signature (s_1', s_2') of a message $m' \neq m$ with non-negligible probability. We will treat separately two cases, first $m' > m$, later $m' < m$. If $m' > m$ then there exist d such that $m' = m + d$. Denote g as f^d. We have that

$$s_1' = f^{n-(m+d)}(x_1) = f^{(n-m)-d}(x_1) = g^{-1}(f^{n-m}(x_1)) = g^{-1}(s_1).$$

We have shown in (b) that g is a OWP. But C is PPT algorithm able to compute inverse of g^{-1}. A contradiction.

If $m' < m$ then there exist e such that $m' = m - e$. Denote h as f^e. We have that

$$s_2' = f^{m-e}(x_2) = h^{-1}(f^m(x_2)) = h^{-1}(s_2).$$

Using the same argument, there exist a PPT algorithm, that is able to invert OWP h. A contradiction.

Since x_1, x_2 are IID variables with uniform distribution and f is a OWP, f^n is a OWP, it follows that $f^n(x_1)$, $f^n(x_2)$ are indistinguishable and no information about x_1 can be obtained from y_2 and vise versa. This completes the proof.

□