15. Twin-Diffie-Hellman

The goal of this exercise is to study variants of ElGamal key encapsulation, based on the **Twin Diffie-Hellman**.

15.1 Diffie-Hellman Problems

Let us consider $\mathbb{G} = \langle g \rangle$, a cyclic group of prime order q. For any $X = g^x, Y = g^y \in \mathbb{G}$, we denote $\mathsf{DH}_g(X, Y) = g^{xy}$, the Diffie-Hellman value of X and Y in basis g. The Diffie-Hellman problems in basis g are defined by:

- The Computational Diffie-Hellman problem (CDH_g): Given $X, Y \stackrel{R}{\leftarrow} \mathbb{G}$, compute $Z = \mathsf{DH}_g(X, Y)$;
- The Decisional Diffie-Hellman problem (DDH_g): For $X, Y, Z \stackrel{R}{\leftarrow} \mathbb{G}$, decide between the two tuples $(g, X, Y, \mathsf{DH}_g(X, Y))$ —a Diffie-Hellman tuple— and (g, X, Y, Z) —a random tuple—.

We can even define more restricted versions of the Diffie-Hellman problems, for a fixed $X \in \mathbb{G}$ too:

- The Computational Diffie-Hellman problem ($\mathsf{CDH}_{g,X}$): Given $Y \stackrel{R}{\leftarrow} \mathbb{G}$, compute $Z = \mathsf{DH}_g(X, Y)$;
- The Decisional Diffie-Hellman problem ($\mathsf{DDH}_{g,X}$): For $Y, Z \stackrel{R}{\leftarrow} \mathbb{G}$, decide between the two tuples $(g, X, Y, \mathsf{DH}_g(X, Y))$ —a Diffie-Hellman tuple— and (g, X, Y, Z) —a random tuple—.

Let us consider an adversary that solves the ℓ-Set Restricted Diffie-Hellman problem ($\mathsf{SCDH}_{g,X,\ell}$), with success ε within time t: for any fixed $g, X \in \mathbb{G}$, but given a random $Y \stackrel{R}{\leftarrow} \mathbb{G}$, $\mathcal{A}$ outputs, within time t, a set S of size at most ℓ, such that $Z = \mathsf{DH}_g(X, Y) \in S$ with probability greater than ε: $\mathsf{Succ}^{\mathsf{scdh}_{g,X,\ell}}(\mathcal{A}) \geq \varepsilon$.

Q-1. Let us consider the algorithm $\mathcal{B}$ that runs $\mathcal{A}$, and then randomly chooses a candidate in S as a solution to the $\mathsf{CDH}_{g,X}$ problem. What is its success probability (a lower bound)?

Q-2. Give a relation between $\mathsf{Succ}^{\mathsf{cdh}_{g,X}}(t)$ and $\mathsf{Succ}^{\mathsf{scdh}_{g,X,\ell}}(t)$, ignoring all the additional computations that $\mathcal{B}$ does beyond $\mathcal{A}$.

Q-3. Let us now consider the algorithm $\mathcal{B}'$ that runs $\mathcal{A}$ on $Y^\alpha g^\beta$, for scalars α and β that it has randomly chosen, to get S':

1. How to convert the set S' of candidates for $\mathsf{DH}_g(X, Y^\alpha g^\beta)$ into a set S'' of candidates for $\mathsf{DH}_g(X, Y)$?

Let us now complete the algorithm $\mathcal{B}'$ by running $\mathcal{A}$ again, but on Y, to get S. Then, it computes $I = S \cap S''$: if $I = \emptyset$, it outputs "Failure", if I contains 2 elements or more, it outputs "Error", in the last case of one element, it outputs this value as the solution.

2. Explain why, if $\mathcal{A}$ succeeds in the 2 calls (on $Y^\alpha g^\beta$ and on Y), our algorithm $\mathcal{B}'$ outputs either the correct solution (success) or "Failure".
3. What is the probability that I contains a wrong solution (possibly additionally with the right one)?
4. What is the success probability of $\mathcal{B}'$ (a lower bound)?

Q-4. Give a new relation between $\mathsf{Succ}^{\mathsf{cdh}_{g,X}}(t)$ and $\mathsf{Succ}^{\mathsf{scdh}_{g,X,\ell}}(t)$, ignoring all the additional computations that $\mathcal{B}$ does beyond $\mathcal{A}$, and namely for computing the intersection I.
When is it better than the previous one (from Q-2)?

15.2 Key Encapsulation and Indistinguishability of Keys

A key encapsulation mechanism aims at generating a session key with a partner, in a non-interactive way. Such a scheme $\mathcal{S} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encaps}, \mathsf{Decaps})$, is defined by 4 algorithms:

- $\mathsf{Setup}(1^k)$ generates the public parameters params;
- $\mathsf{KeyGen}(\mathsf{params})$ generates the pair of private and public keys $(\mathsf{dk}, \mathsf{ek})$;
- $\mathsf{Encaps}(\mathsf{ek})$ outputs a session key $K \in \{0,1\}^k$ and an encapsulation c of this key, under the public key ek;
- $\mathsf{Decaps}(\mathsf{dk}, c)$ outputs the session key K encapsulated in c under ek, if dk is the private key associated to the public key ek.

Such an encapsulated key should be indistinguishable from a random key to any third party, hence the *indistinguishability* security game: the challenger runs the setup algorithm Setup and the key generation algorithm KeyGen to generate the encapsulation key ek and the associated decapsulation key dk. It runs the encapsulation algorithm on ek, and gets the pair (K, c). The challenger flips a bit $b \stackrel{R}{\leftarrow} \{0,1\}$ and sets $K_b \leftarrow K$, while $K_{1-b} \stackrel{R}{\leftarrow} \{0,1\}^k$. It provides the triple (K_0, K_1, c) to $\mathcal{A}$. Eventually, $\mathcal{A}$ has to guess b. To this aim, it outputs b'.

$\mathsf{Exp}^{\mathsf{ind}-b}_{\mathcal{S},\mathcal{A}}(k)$
1. $\mathsf{params} \leftarrow \mathsf{Setup}(1^k)$
2. $(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{KeyGen}(\mathsf{params})$
3. $(K, c) \leftarrow \mathsf{Encaps}(\mathsf{ek})$
4. $K_b \leftarrow K$, $K_{1-b} \stackrel{R}{\leftarrow} \{0,1\}^k$
5. $b' \leftarrow \mathcal{A}(\mathsf{ek}, K_0, K_1, c)$
6. `RETURN` b'

The quality of the adversary $\mathcal{A}$ is measured by its advantage

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{S}}(\mathcal{A}) = \Pr[1 \leftarrow \mathcal{A} \mid b = 1] - \Pr[1 \leftarrow \mathcal{A} \mid b = 0] = \Pr[\mathsf{Exp}^{\mathsf{ind}-1}_{\mathcal{S},\mathcal{A}}(k) = 1] - \Pr[\mathsf{Exp}^{\mathsf{ind}-0}_{\mathcal{S},\mathcal{A}}(k) = 1].$$

The security of the key encapsulation scheme $\mathcal{S}$ is measured by the advantage of the best adversary within time t:

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{S}}(t) = \max_{\mathcal{A} \le t}\{\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{S}}(\mathcal{A})\}.$$

15.3 Hashed ElGamal

Let us consider $\mathbb{G} = \langle g \rangle$, a cyclic group of prime order q of length $2k$, together with a hash function $\mathcal{H}$ onto $\{0,1\}^k$, where k is the security parameter. The setup algorithm outputs the triple $\mathsf{params} = (g, q, \mathcal{H})$. Then,

- $\mathsf{KeyGen}(\mathsf{params})$: the private key is a scalar, $\mathsf{dk} = x \in \mathbb{Z}_q^\star$, and the public key is the associated group element $\mathsf{ek} = X = g^x$;
- $\mathsf{Encaps}(\mathsf{ek})$: in order to generate an encapsulated key under $\mathsf{ek} = X$, one chooses a random scalar $y \in \mathbb{Z}_q^\star$, and computes $c = Y = g^y$, while $K = \mathcal{H}(Y, Z)$, where $Z = X^y = \mathsf{DH}_g(X, Y)$;
- $\mathsf{Decaps}(\mathsf{dk}, c)$: in order to decapsulate c, with $\mathsf{dk} = x$, one computes $Z = c_1^x$ and then $K = \mathcal{H}(Y, Z)$.

Q-5. Show that this scheme provides *indistinguishability* under a SCDH problem, in the random oracle model. Detail the proof by successive alterations of the security game.

Q-6. Show that a decapsulation oracle provides a kind of DDH-oracle. Precise this oracle available to an adversary when it can ask decapsulation queries.

Then indistinguishability with decapsulation-oracle access relies on the Gap-Diffie-Hellman problem $\mathsf{GDH}_{g,X}$: Solve $\mathsf{CDH}_{g,X}$ given access to a $\mathsf{DDH}_{g,X}$ oracle.

15.4 Twin Diffie-Hellman Problems

Let us consider $\mathbb{G} = \langle g \rangle$, a cyclic group of prime order q. The Twin Diffie-Hellman problems are defined by:

- The Computational Twin Diffie-Hellman problem (CTDH_g): Given $X_1, X_2, Y \stackrel{R}{\leftarrow} \mathbb{G}$, compute $(Z_1, Z_2) = (\mathsf{DH}_g(X_1, Y), \mathsf{DH}_g(X_2, Y))$;
- The Decisional Twin Diffie-Hellman problem (DTDH_g): For $X_1, X_2, Y, Z_1, Z_2 \stackrel{R}{\leftarrow} \mathbb{G}$, decide between the tuples $(X_1, X_2, Y, \mathsf{DH}_g(X_1, Y), \mathsf{DH}_g(X_2, Y))$ and (X_1, X_2, Y, Z_1, Z_2).

We can also define the restricted versions $\mathsf{CTDH}_{g,X_1,X_2}$ and $\mathsf{DTDH}_{g,X_1,X_2}$, when X_1 and X_2 are fixed too, and thus instances are respectively a group element Y, or a triple (Y, Z_1, Z_2).

Q-7. For fixed $g, X_1 \in \mathbb{G}$, but any chosen $X_2 \in \mathbb{G}$ (any way one wants), prove that the $\mathsf{CTDH}_{g,X_1,X_2}$ problem is at least as hard as the CDH_{g,X_1} problem. Give the relation between the success probabilities.

Q-8. For fixed $g, X_1 \in \mathbb{G}$, let us choose random scalars $r, s \stackrel{R}{\leftarrow} \mathbb{Z}_q^\star$, and set $X_2 = g^s / X_1^r$.

1. Explain why no information leaks about s from X_1 and X_2.
2. When we receive a $\mathsf{DTDH}_{g,X_1,X_2}$ instance (Y, Z_1, Z_2), prove that $Z_1^r Z_2 = Y^s$ if and only if both $Z_1 = \mathsf{DH}_g(X_1, Y)$ and $Z_2 = \mathsf{DH}_g(X_2, Y)$, but with negligible probability, which provides a $\mathsf{DTDH}_{g,X_1,X_2}$ distinguisher.

15.5 Hashed Twin Diffie-Hellman

Let us consider $\mathbb{G} = \langle g \rangle$, a cyclic group of prime order q of length $2k$, together with a hash function $\mathcal{H}$ onto $\{0,1\}^k$. The setup algorithm outputs the triple $\mathsf{params} = (g, q, \mathcal{H})$. Then,

- $\mathsf{KeyGen}(\mathsf{params})$: the private key is a scalar, $\mathsf{dk} = (x_1, x_2) \in \mathbb{Z}_q^{\star 2}$, and the public key is the associated group elements $\mathsf{ek} = (X_1 = g^{x_1}, X_2 = g^{x_2})$;
- $\mathsf{Encaps}(\mathsf{ek})$: in order to encapsulate a key under $\mathsf{ek} = (X_1, X_2)$, one chooses a random scalar $y \in \mathbb{Z}_q^\star$, and computes $c = Y = g^y$, while $K = \mathcal{H}(Y, Z_1, Z_2)$, where $Z_1 = X_1^y$ and $Z_2 = X_2^y$;
- $\mathsf{Decaps}(\mathsf{dk}, c)$: in order to decapsulate c, with $\mathsf{dk} = x$, one computes $Z_1 = c^{x_1}$ and $Z_2 = c^{x_2}$ and eventually $K = \mathcal{H}(Y, Z_1, Z_2)$.

Q-9. For any fixed $X_1 \leftarrow \mathbb{G}$, and two random scalars $r, s \xleftarrow{R} \mathbb{Z}_q^\star$. If we define $\mathsf{ek} = (X_1, X_2 = g^s / X_1^r)$, show that the knowledge of (r, s) allows to simulate the decapsulation oracle, in the random oracle model, but with negligible probability.

Q-10. Show that this scheme provides *indistinguishability* even with *decapsulation-oracle access* under the CDH_g assumption, in the random oracle model.