Public Key Cryptography

Unlike the symmetric ciphers, Public Key Cryptographic systems obviate the need for utilizing a secure channel prior to initiating secure communication.

In principle, public-key encryption systems can be based on one-way trapdoor functions which can be defined informally as:

\[ f_t(x) : \mathbb{D} \rightarrow \mathbb{R} \]

which is easy to evaluate \( \forall x \in \mathbb{D} \) and difficult to invert for almost all values of \( \mathbb{R} \). Nevertheless if one is in possession of the trapdoor \( t \) then it is easy to invert \( f_t(x) \).

These cryptosystems based on the one-way trapdoor functions are also called as asymmetric cryptosystems.

A trapdoor function is not necessary though for building a public-key cryptosystem. The first public-key method was discovered by Diffie and Hellman in 1977. Their concrete cryptosystem design allows two users to exchange a key without the employment of a secure channel.

Diffie-Hellman Key Exchange Protocol

Symmetric cryptosystems always had the problem of safe transfer of secret key, which often meant a physical delivery of the key to the intended parties. Public Key Cryptography on the other hand, transfers the secret key via insecure channels. This protocol was proposed by Diffie and Hellman known as Diffie-Hellman exponential key exchange protocol.

**Diffie-Hellman exponential key exchange protocol**

- **Common Public Input** \((p, g)\), \(p\) is a large prime number
  - \(g\) is a generator in \(\mathbb{F}_p^*\) of order \(m\)

- **Output** Element in \(\mathbb{F}_p^*\) shared between Alice and Bob

**Step 1:**

- Alice selects \(a \in U [1,m]\);
- Computes \(g_a = g^a \text{ mod } p\);
- Sends \(g^a\) to Bob.
Step 2:
- Bob selects \( b \in U[1,m) \);
- computes \( g_b = g^b \mod p \);
- sends \( g^b \) to Alice.

Step 3:
- Alice computes \( K_a = g^{a \cdot b} \mod p \);

Step 4:
- Bob computes \( K_b = g^{b \cdot a} \mod p \);

Observe that, \( K = K_a = K_b \in \mathbb{Z}_p^* \)

From the protocol it is clear that for Alice,
\[
K_a = g^{ba} \mod p
\]
and for Bob
\[
K_b = g^{ab} \mod p
\]

Since, \( ab \equiv ba \mod (p - 1) \), both Alice and Bob compute the same value.

**Intuition For Security**

a) Modular exponentiation is "easy"
\[
x \rightarrow g^x \mod p
\]
and
b) Computing the Discrete Logarithm is “hard”
given \( h \in \langle g \rangle \)
to compute \( \log_g h \).

**Computational Diffie-Hellman Problem (CDH)**

The secrecy of the shared key is similar to the problem of computing \( g^{ab} \mod p \) given \( g_a \) and \( g_b \). This is called the CDH problem.

CDH problem is formally stated as:

Given
\[
\langle g,p \rangle \leftarrow \text{Desc}(1^k)
\]
and
\[
g^a, g^b \in \langle g \rangle
\]
with,
\[
0 \leq a, b < \text{order}(g),
\]
find
\[
g^{a \cdot b} = g^{a \cdot b} \mod \text{order}(g).
\]

The CDH problem in turn depends on the difficulty of the Discrete Logarithm Problem.
**Discrete Logarithm Problem (DL)**

The DL problem is formulated as follows:

Given \( A \) and \( \langle g, p \rangle \leftarrow \text{Desc}(1^k) \)

find \( a \) such that,

\[ A = g^a \quad 0 < a < \text{order}(g) \]

It is clear that the security of the Diffie Hellman Key-Exchange protocol is related to somehow to the hardness of the above problems. We investigate this more formally below.

**The CDH Assumption** For any probabilistic polytime (ppt) algorithm \( A \), it holds that,

\[ \text{Prob}[g^{a \cdot b} \leftarrow A(g, p, g^a, g^b)] \text{ is negligible in } k \]

where \( \langle g, p \rangle \leftarrow \text{Desc}(1^k) \) and \( 0 < a, b < \text{order}(g) \).

**Theorem** (Informally stated)

“Any adversary that is poly-time bounded in \( k \) and is given the communication transcript between Alice and Bob in the DH-key exchange protocol with security parameter \( k \) is incapable of computing the key unless the CDH assumption fails.”

Note that the above theorem is rather weak. It merely says that the adversary cannot extract the whole key from the communication transcript. We will revisit the above later on.

**The Discrete Logarithm Assumption**

For any ppt \( A \), it holds that

\[ \text{Prob}[a \leftarrow A(g, p, g^a)] \text{ is negligible.} \]

where \( \langle g, p \rangle \leftarrow \text{Desc}(1^k) \) and \( 0 < a < \text{order}(g) \).

**Proposition.** The CDH assumption implies the DL assumption.

**Proof.** Let’s make a counterpositive statement:

Assume that the DL assumption fails

This implies,

\[ \exists A : \text{Prob}[a \leftarrow A(g, p, g^a)] \geq \alpha(k) \]

\( \alpha(k) \) is a “substantial” function (i.e., non-negligible).

Design an algorithm \( A' \) that operates as follows:

Input : \( \langle g, p, g_a, g_b \rangle = \langle g, p, g^a, g^b \rangle \)

\( A' \) applies the ppt \( A \) on input \( \langle g, p, A \rangle \) obtains the output \( a \); then it returns \( g_b^a \)

It is very easy to see that \( A \) violates the CDH assumption, as it will solve the CDH problem with probability exactly \( \alpha(k) \). This completes the proof.

**Open problem**

Does DL assumption \( \Rightarrow \) CDH assumption?
What about $\text{Desc}(1^k)$?

Bad choices of $g, p$ can make it very easy to compute the secret keys. We will demonstrate such a bad choice in this section. We will assume that $g, p$ are chosen such that

$$Z_p^* = < g >$$

from which it follows

$$\text{order}(g)b = p - 1$$

(i.e., the order of $g$ is maximum, so this may look like a good choice). Finally also assume,

$$(p - 1) = q_1.q_2....q_l$$

i.e., $p - 1$ can be factored into small primes, and all $q_i$’s are small numbers. As $p$ is public and all $q_i$’s are small, they can be found easily.

Consider,

$$G_i = \langle g^{p-1}/q_i \rangle$$

a subgroup of $Z_p^*$ of order $q_i$

Let,

$$h \in \langle g \rangle = Z_p^*$$

such that,

$$h = g^a$$

with $a$ unknown

Project $h$ into $G_i$

$$\therefore \ h^{\frac{p-1}{q_i}} \in G_i$$

$$\Rightarrow h^{\frac{p-1}{q_i}} = (g^{\frac{p-1}{q_i}})^a$$

$\therefore$ it holds that,

$$h_i = (g^{\frac{p-1}{q_i}})^{a_i}$$

such that

$$a_i \in [0, q_i)$$

and

$$a_i = a \mod q_i.$$

As $q_i$ is really small, $a_i$ can be found out very quickly (e.g., trial and error).

By doing this, all $a_i$’s can be easily obtained.

It can be observed that,

$$a = a_1 \mod q_1$$
$$a = a_2 \mod q_2$$
$$a = a_3 \mod q_3$$
$$\vdots$$
$$a = a_l \mod q_l$$

These equations can be solved to obtain a unique value for $a$ by the Chinese Remainder Theorem. Hence the Discrete Logarithm $a = \log_g h$ is found.

$\therefore$ It is a bad choice to choose $p$, such that $p - 1$ can be factored into small primes.

A good choice for $\text{Desc}(1^k)$ is to output a prime number, so that $\frac{p-1}{2}$ contains a large prime $q$ and have $g$ generate a subgroup of size $q$ inside $Z_p^*$. 