SCIS 2006 The 2006 Symposium on Cryptography and Information Security
Hiroshima, Japan, Jan. 17-20, 2006
The Institute of Electronics, Information and Communication Engineers

An Improved Certificateless Public Key Encryption

Routo Terada * Denise H. Goya †

1. Introduction

The concept of Identity Based Encryption – IBE – system was proposed by [Sh4] for which the public key can be the identity itself. [BoFr1] presented an IBE system based on bilinear pairing functions, that requires a Public Key Generator – PKG. The PKG needs to be trusted in the sense that it can generate any of the private keys, i.e., it can exercise the so-called *key escrow*, which is undesirable in many applications. On the other hand this system does not require the so-called Public Key Infrastructure – PKI – with its complex and costly management of Digital Certificates. [AlPa3] proposed a Certificateless Public Key Encryption – CL-PKE – scheme, i.e., a cryptographic scheme which does not require either a Digital Certificate to certify the public key or a PKI. It is also based on bilinear pairing functions. In CL-PKE an adversary $\mathcal{A}$ may replace the victm's public key with another one, say X, so that $\mathcal{A}$ knows the private key corresponding to X; but still $\mathcal{A}$ is *not* able to decrypt the message encrypted with the original *published* public key. This important property is accomplished by the fact that only the PKG can *bind* the key pair for any other entity with that entity. For a secure CL-PKE scheme the public key of an entity can be bound to an identity of the entity without any security measure. Furthermore, it is key escrow free, which is not achieved in the framework proposed in [Sh4].

In this paper we construct a CL-PKE scheme based on bilinear pairing functions which: (1) does not allow key escrow by the PKG; (2) does not require Digital Certificates; (3) is more efficient on computation than previously published IBE or CL-PKE schemes ([BoFr1], [Ge3], [AlPa3], [AlPa5], [ChCo5], [Ga5]); (4) and is secure in the sense that it is strong against IND-CCA2 attack [1] [Be8a], based on the Random Oracle Model [Be8a] and the difficulty of the BDH Problem[2] [ChLe2]. For the security proof we reduce (in polynomial time) the problem of solving the BDH Problem to the IND-CCA2 attack against our CL-PKE. The BDH Problem is as follows: (1) Let G_1 and G_2 be two groups of prime order q and let $\hat{e} : G_1 \times G_1 \rightarrow G_2$ be a bilinear pairing function; (2) Given $P \in G_1^*$, $a, b, c \in Z_q^*$ and P, aP, bP, cP compute $\hat{e}(P, P)^{abc}$.

2. Proposed CL-PKE scheme

Our CL-PKE sheme is defined as the following set of algorithms.

$\boxed{\texttt{setup}}$ Given a security parameter k the Public Key Generator - PKG: (1) generates two cyclic groups G_1 and G_2 of prime order q and a bilinear pairing $\hat{e} : G_1 \times G_1 \rightarrow G_2$. Choose randomly a generator $P \in G_1^*$. (2) Chooses randomly $s \in Z_q^*$ and compute $P_{pub} = sP$. (3) Chooses three hash functions: (a) $H_1 : \{0,1\}^* \rightarrow G_1^*$ (b) $H_2 : G_1 \times G_2 \times G_1 \rightarrow \{0,1\}^n$ integer $n > 0$ (c) $H_3 : \{0,1\}^{n-k_0} \times \{0,1\}^{k_0} \rightarrow Z_q^*$, for integers n and k_0, $0 < k_0 < n$, with k_0 polynomial on n. The message space is $\mathcal{M} = \{0,1\}^{n-k_0}$. The ciphertext space is $\mathcal{C} = G_1^* \times \{0,1\}^n$. The system `master-key` is s. The system parameters are `params` $= \langle q, G_1, G_2, \hat{e}, n, k_0, P, P_{pub}, H_1, H_2, H_3\rangle$

$\boxed{\texttt{extract}}$ Given an identifier $ID_A \in \{0,1\}^*$, `params` and a `master-key` s the PKG: (1) computes $Q_A = H_1(ID_A)$. (2) returns the partial secret key $d_A = sQ_A$.

$\boxed{\texttt{publish}}$ Given `params`, an entity A selects at random a secret information $t_A \in Z_q^*$ and computes its public key $N_A = t_A P$. A keeps t_A safely and publishes N_A.

$\boxed{\texttt{encrypt}}$ Given a plaintext $m \in \mathcal{M}$, an identity ID_A, `params` and a public key N_A any message sender: (1) chooses randomly $\sigma \in \{0,1\}^{k_0}$. (2) computes $r = H_3(m, \sigma)$, $Q_A = H_1(ID_A)$, $g^r = \hat{e}(P_{pub}, Q_A)^r$, $f = rN_A$. (3) returns the ciphertext $C = \langle rP, (m||\sigma) \oplus H_2(rP, g^r, f)\rangle$.

$\boxed{\texttt{decrypt}}$ Given $C = \langle U, V\rangle \in \mathcal{C}$ and the secret values d_A and t_A, the entity A: (1) computes $g' = \hat{e}(U, d_A)$, $f' = t_A U, V \oplus H_2(U, g', f') = (m||\sigma)$ (2) splits $(m||\sigma)$ and computes $r = H_3(m, \sigma)$ (3) if $U = rP$, returns the plaintext m, else return $\perp$.

2.1. Decryption of the proposed CL-PKE

To prove the decryption is correct, it is enough to remember the pairing $\hat{e}()$ is bilinear, as shown next: $g' = \hat{e}(U, d_A) = \hat{e}(rP, sQ_A) = \hat{e}(P, Q_A)^{rs} = \hat{e}(sP, Q_A)^r = \hat{e}(P_{pub}, Q_A)^r = g^r$, $f' = t_A U = t_A rP = rN_A = f$. Hence $H_2(U, g', f') = H_2(rP, g^r, f)$. Since $V = (m||\sigma) \oplus H_2(rP, g^r, f)$, $V \oplus H_2(rP, g^r, f) = (m||\sigma)$. Therefore `decrypt` recovers correctly the plaintext from the ciphertext by applying `encrypt`.

*rt@ime.usp.br, University of Sao Paulo
†University of Sao Paulo

[1] Indistinguishability Adaptive Chosen-Ciphertext Attack
[2] Bilinear Diffie-Hellman Problem

In summary, our CL-PKE may be described as a composition of previous works, as follows. The usage of rP in H_2 was suggested by [CrSh4]. The usage of rN_A and a complete formulation of H_2 was found in [ChCo5], where they try to optimize the CL-PKE in [AlPa5] and [AlPa3]. The choices of the message and ciphertext spaces, besides H_3, was inspired by [Ga5], which in turn adopted the tranformation in [FuOk0] to strengthen the public key encryption scheme using fewer hash functions. [Ga5] involved three elliptic groups, it is not CL-PKE (but is IBE), and presented an improvement on the work in [BoFr1], which in turn allowed the realization of CL-PKE.

We briefly show previous PKE schemes based on bilinear pairing below:

Scheme	Ciphertext C
[BoFr1]	$\langle rP, \sigma \oplus H_2(g^r), m \oplus H_4(\sigma)\rangle$
[AlPa3]	$\langle rP, \sigma \oplus H_2(g^r_{(N_A)}), m \oplus H_4(\sigma)\rangle$
[AlPa5]	$\langle rP, \sigma \oplus H_2(g^r) \oplus H_5(rN_A),$ $m \oplus H_4(\sigma)\rangle$
[ChCo5]	$\langle rP, \sigma \oplus H_2(rP, g^r, rN_A),$ $m \oplus H_4(\sigma)\rangle$
[Ga5]	$\langle tP, (m\|\|\sigma) \oplus H_2(g^t)\rangle$
Our CL-PKE	$\langle tP, (m\|\|\sigma) \oplus H_2(tP, g^t, tN_A)\rangle$

Regarding this table, we observe that: (1) In [FuOk9] $r = H_3(m, \sigma)$ where $m, \sigma \in \{0,1\}^n$. (2) In [FuOk0] $t = H_3(m, \sigma)$ where $m \in \{0,1\}^{n-k_0}$, $\sigma \in \{0,1\}^{k_0}$.

3. Complexity of the proposed CL-PKE

Let the basic operations be denoted as follows: P for bilinear pairing, M for scalar multiplication, E for exponentiation, and H for hash function.

Then we have the following tables with the number of computations in each of the published similar schemes of public key encryption based on bilinear pairing (where g is the number of bits to represent one point in G_1):

	crypt				decrypt			
PKE Scheme	P	M	E	H	P	M	E	H
[AlPa3]	3	1	1	4	1	1	0	3
[AlPa5]	1	2	1	5	1	2	0	4
[ChCo5]	1	2	1	4	1	2	0	3
Our CL-PKE	1	2	1	3	1	2	0	2

	size (bits)		
PKE Scheme	pub.k.	msg	ciph.
[AlPa3]	$2g$	n	$g+n+n$
[AlPa5]	g	n	$g+n+n$
[ChCo5]	g	n	$g+n+n$
Our CL-PKE	g	$m'-k_0$	$g+m'$

For some choices of m' and k_0, k_0 being of polynomial size on m', our CL-PKE computes smaller ciphertexts, as shown below, for the case $n = m' - k_0$: (1) for $k_0 > n, m' > 2n$, the ciphertext size is greater than in previous schemes; for $k_0 = n, m' = 2n$ it is equal; and for $k_0 < n, m' < 2n$, it is smaller. The case $k_0 < n$ is the best choice, as long as $k_0(n) = O(n^{1/c})$, where $c > 1$ is a constant.

4. Type-I IND-CCA2 adversary

This Section defines a type of adversary that will be used in Section 7.

The Type-I adversary against CL-PKE does not know the `master-key` and plays the Game 1 against a challenger, as follows:

`Setup` The challenger uses a security parameter k and executes the algorithm `setup`. It gives the adversary the system parameters `params` and keeps `master-key` safely.

Phase 1 The adversary issues queries $q_1, ..., q_n$ of one of follows, to the challenger: (1) `Extract the partial secret key` $PrivKeyL$ of ID_i. The challenger answers executing the algorithm `extract` and gives the result d_{ID_i} to the adversary. (2) `Publish the public key` of ID_i. The challenger answers executing the algorithm `publish` and provides N_{ID_i} to the adversary, but it keeps a list of key pairs (N_{ID_i}, t_{ID_i}) already generated. (3) `Replace.` Replace the public key of ID_i by a new value N'_{ID_i}. The challenger records the new value N'_{ID_i} for ID_i. (4) `Extract the secret key` $PrivKeyR$ of ID_i. If the public key of ID_i has not been replaced, the challenger answers returning the corresponding value t_{ID_i}; otherwise, the game is aborted. (5) `Decrypt` $< ID_i, C_i, N_i >$. The challenger decrypts the ciphertext, after getting the secret values d_{ID_i} (by executing `extract`, if necessary) and t_{ID_i} (by looking up the list of key pairs, whose size is bounded by a polynomial on the amount of publications). If t_{ID_i} is not found, the challenger returns $\perp$.

`Challenge` Once the adversary decides the Phase 1 is finished, it returns to the challenger two equal length messages $m_0, m_1 \in \mathcal{M}$, an identity ID_{ch} and a public key N_{ch} upon which the challenge is to be applied. The challenger randomly chooses $b \in \{0,1\}$ and returns to the adversary the ciphertext $C^* =$`encrypt`$(\texttt{params}, ID_{ch}, m_b, N_{ch})$ as the challenge, under the condition that ID_{ch} was not used in any previous extraction of the secret key $PrivKeyL$, in Phase 1 (so, N_{ch} could be replaced, if the $PrivKeyR$ is not asked).

Phase 2 The adversary issues new queries $q_{n+1}, ..., q_l$ of one of follows, to the challenger: (1) `Extract the partial secret key` of ID_i, such that $ID_i \neq ID_{ch}$. The challenger answers as in Phase 1. (2) `Publish the public key` ID_i. The challenger answers as in Phase 1. (3) `Replace the public key` of ID_i by a new public key value N'_{ID_i}. The challenger answers as in Phase 1. (4) `Extract the secret key` $PrivKeyR$ of ID_i. The challenger answers as in Phase 1. (5) `Decrypt` $< ID_i, C_i, N_i > \neq < ID_{ch}, C^*, N_{ch} >$. The challenger answers as in Phase 1.

`Guess` The adversary generates a guess $b' \in \{0,1\}$ and wins the game if $b' = b$

In summary, a Type-I adversary $\mathcal{A}$ against CL-PKE does not know the `master-key`, but can replace public key values and extract secret keys $PrivKeyL$ and $PrivKeyR$, ask public keys and decryptions, for chosen identities, under the following restrictions: (1) $\mathcal{A}$ cannot extract the secret key $PrivKeyL$ for ID_{ch}. (2) $\mathcal{A}$ cannot extract the secret key $PrivKeyR$ for identifiers whose public keys were replaced. (3) In Phase 2, $\mathcal{A}$ cannot issue decryption queries on the challenge ciphertext C^*, for the corresponding ID_{ch} and N_{ch}, which were used to encrypt m_b.

Such an adversary is called Type-I IND-CCA2. We say the advantage of an adversary $\mathcal{A}$ of Type-I IND-CCA2 against a scheme $\mathcal{S}$ is a function of the security parameter k, defined by: $Adv^{I}_{\mathcal{S},\mathcal{A}}(k) = |\Pr[b = b'] - 1/2|$

5. Type-II IND-CCA2 adversary

The Type-II adversary against CL-PKE knows the `master-key` (and thus it knows $PrivKeyL$ for any entity) and plays Game 2 against a challenger, as follows:

$\boxed{\texttt{Setup}}$ The challenger uses a security parameter k and executes the algorithm `setup`. It gives to the adversary the system parameters `params` and the `master-key`. The adversary chooses a victim with identity ID_{ch}.

$\boxed{\textbf{Phase 1}}$ The adversary issues queries $q_1, ..., q_n$ of one of follows, to the challenger: (1) `Publish the public key` of ID_i. As in Phase 1 of Game 1, the challenger answers executing the algorithm `publish` and returns N_{ID_i} to the adversary, but it keeps a list of key pairs $< N_{ID_i}, t_{ID_i} >$ previously generated. (2) `Decrypt` $< ID_{ch}, C_i, N_{ch} >$. The challenger answers with a decryption, as in Phase 1 of Game 1.

$\boxed{\texttt{Challenge}}$ Once the adversary decides Phase 1 is finished, it gives the challenger two equal length messages $m_0, m_1 \in \mathcal{M}$, over which the challenge is to be applied. The challenger randomly chooses a challenge $b \in \{0, 1\}$ and returns to the adversary, as a challenge, the ciphertext $C^* = \texttt{encrypt}(\texttt{params}, ID_{ch}, m_b, N_{ch})$

$\boxed{\textbf{Phase 2}}$ The adversary issues new queries $q_{n+1}, ..., q_l$ of one of follows, to the challenger: (1) `Publish the public key` of ID_i. The challenger answers as in Phase 1. (2) `Decrypt` $< ID_{ch}, C^*, N_{ch} >$ where $C_i \neq C^*$. The challenger answers as in Phase 1.

$\boxed{\texttt{Guess}}$ The adversary generates a $b' \in \{0, 1\}$ and wins the game if $b' = b$

In summary, an adversary $\mathcal{A}$ of Type-II against CL-PKE knows the `master-key`, and consequently it can compute the secret partial key $PrivKeyL$. Furthermore, it may ask for public keys and for decryptions, for an identity of its choice, under the following restrictions: (1) $\mathcal{A}$ cannot replace the public key values. (2) $\mathcal{A}$ cannot extract secret keys $PrivKeyR$. (3) In Phase 2, $\mathcal{A}$ cannot issue decryption queries on the challenge ciphertext C^*, for the corresponding ID_{ch} and N_{ch}, which were used to encrypt m_b.

Such an adversary is called Type-II IND-CCA2. We say the advantage of an adversary $\mathcal{A}$ of Type-II IND-CCA2 against a scheme $\mathcal{S}$ is a function of the security parameter k, defined by: $Adv^{II}_{\mathcal{S},\mathcal{A}}(k) = |\Pr[b = b'] - 1/2|$

Given the descriptions of adversaries against CL-PKE, we define the security notions as follows.

Definition 1. *A CL-PKE scheme $\mathcal{S}$ satisfies the IND-CCA2 notion of security for any Type-I and Type-II IND-CCA2 adversary $\mathcal{A}$, in polynomial time on k, if the following advantages are negligible $Adv^{I}_{\mathcal{S},\mathcal{A}}(k)$ and $Adv^{II}_{\mathcal{S},\mathcal{A}}(k)$*

6. Auxiliary Schemes

In this Section we define the set of algorithms **Basic** and **Basic Hyb**, which constitute Public Key Encryption schemes, to be used in the proofs in Section 7 as challengers.

For Type I adversaries we will have a sequence of reductions envolving **Basic** and **Basic Hyb** such that if there is a non-negligible advantage for an adversary against our CL-PKE then there is a polynomial time algorithm to solve the BDH (Bilinear Diffie-Hellman) Problem. Similarly for Type II adversaries the GDH (Gap Diffie-Hellman) Problem is reduced to a polynomial time attack of our CL-PKE.

6.1. Basic

Basic is defined by three algorithms as follows:

$\boxed{\texttt{generatekeysb}}$ Given a security parameter k: (1) Generate two cyclic groups G_1 and G_2 of prime order q and a bilinear pairing $\hat{e} : G_1 \times G_1 \rightarrow G_2$. Choose randomly a generator $P \in G_1^*$. (2) Choose randomly $s \in Z_q^*$ and compute $P_{pub} = sP$. (3) Choose randomly $Q_A \in G_1^*$ and $t_A \in Z_q^*$ and compute $N_A = t_A P$. (4) Choose hash function $H_2 : G_1 \times G_2 \times G_1 \rightarrow \{0, 1\}^n$ for integer $n > 0$. The message space is $\mathcal{M} = \{0, 1\}^n$. The ciphertext space is $\mathcal{C} = G_1^* \times \{0, 1\}^n$. The secret key is $K_{sec} = d_A$. The public key is $K_{pub} =< q, G_1, G_2, \hat{e}, n, P, P_{pub}, H_2, Q_A, N_A >=< \texttt{paramsb}, Q_A, N_A >$.

$\boxed{\texttt{encryptb}}$ Given a message $m \in \mathcal{M}$ and a public key K_{pub}: (1) choose randomly $r \in Z_q^*$. (2) compute $g^r = \hat{e}(P_{pub}, Q_A)^r$. (3) return the ciphertext $C =< rP, m \oplus H_2(rP, g^r, rN_A) >$

$\boxed{\texttt{decryptb}}$ Given $C =< U, V > \in \mathcal{C}$, `paramsb`, The secret key K_{sec} and a value $t_A \in Z_q^*$: (1) compute $g' = \hat{e}(U, d_A)$. (2) compute and return $V \oplus H_2(U, g', t_A U) = m$.

6.2. Basic Hyb

Basic Hyb is defined by the following algorithms:

$\boxed{\texttt{generatekeysh}}$ Given a security parameter k: (1) Generate two cyclic groups G_1 and G_2 of prime order q and a bilinear pairing $\hat{e} : G_1 \times G_1 \rightarrow G_2$ choose randomly a generator $P \in G_1^*$. (2) choose randomly $s \in Z_q^*$ and compute $P_{pub} = sP$. (3) choose randomly $Q_A \in G_1^*$ and $t_A \in Z_q^*$; compute $N_A = t_A P$ and $d_A = sQ_A$. (4) choose two hash functions $H_2 : G_1 \times G_2 \times G_1 \rightarrow \{0, 1\}^n$, $H_3 : \{0, 1\}^{n-k_0} \times \{0, 1\}^{k_0} \rightarrow Z_q^*$ for integers n and k_0, $0 < k_0 < n$, with k_0 polynomial on n. The message space is $\mathcal{M} = \{0, 1\}^{n-k_0}$. The ciphertext space is $\mathcal{C} = G_1^* \times \{0, 1\}^n$. The secret key is $K_{sec} = d_A$. The public key is $K_{pub} =< q, G_1, G_2, \hat{e}, n, k_0, P, P_{pub}, H_2, H_3, Q_A, N_A >=< \texttt{paramsh}, Q_A, N_A >$.

$\boxed{\texttt{encrypth}}$ Given a message $m \in \mathcal{M}$ and a public key K_{pub}: (1) choose randomly $\sigma \in \{0, 1\}^{k_0}$. (2) compute $r = H_3\ (m, \sigma), g^r = \hat{e}(P_{pub}, Q_A)^r$. (3) return the ciphertext $C =< rP, (m||\sigma) \oplus H_2(rP, g^r, rN_A) >$

$\boxed{\texttt{decrypth}}$ Given $C =< U, V > \in \mathcal{C}$, `paramsh` and the secret key K_{sec} and a value $t_A \in Z_q^*$: (1) compute $g' = \hat{e}(U, d_A)$. (2) compute $V \oplus H_2(U, g', t_A U) = (m||\sigma)$. (3) split $(m||\sigma)$ and compute $r = H_3(m, \sigma)$. (4) if $U = rP$, return the message m, otherwise return $\perp$.

7. Type I adversary and a sequence of reductions

With the auxiliary schemes **Basic** and **Basic Hyb**, we will prove that our CL-PKE scheme is secure against

Type-I IND-CCA2 adversaries. The sequence of reductions is illustrated below with the corresponding Lemmas, where $\rightarrow$ means "reduced to", and $\mathcal{A}_j$ means "an adversary":

$\mathcal{A}_1 \rightarrow\rightarrow\rightarrow\rightarrow\rightarrow\rightarrow$ Lemma 1 $\rightarrow \mathcal{A}_2$

Type I IND-CCA2
$\downarrow$
against
$\downarrow$
Our CL-PKE

$\mathcal{A}_2 \rightarrow\rightarrow\rightarrow\rightarrow\rightarrow\rightarrow$ Lemma 3 $\rightarrow \mathcal{A}_3$

Type I IND-CCA2
$\downarrow$
against
$\downarrow$
PKE-**Basic Hyb**

$\mathcal{A}_3 \rightarrow\rightarrow\rightarrow\rightarrow\rightarrow\rightarrow$ Lemma 4 $\rightarrow \mathcal{B}$

IND-CPA $\mathcal{B}$ solves BDH
$\downarrow$
against
$\downarrow$
PKE-**Basic**

Lemma 1. *Assume H_1 is a random oracle. Let $\mathcal{A}_1$ be a Type-I IND-CCA2 adversary against our CL-PKE scheme, with advantage ε_1, execution time t_1, which issues at most q_1 queries to H_1, q_e key* `extract` *queries and q_d* `decrypt` *queries. Then there is a Type I IND-CCA2 adversary $\mathcal{A}_2$ against **Basic Hyb** with advantage $\varepsilon_2 \geq \varepsilon_1/q_1$ and execution time $t_2 \leq t_1 + c_{G_1}(q_1+q_d+q_e)$, where c_{G_1} denotes the multiplication time of a point in G_1 by a scalar.*

Proof: We construct a Type-I IND-CCA2 adversary $\mathcal{A}_2$ that uses $\mathcal{A}_1$ to obtain an advantage against **Basic Hyb**. This proof is similar to the proof of Lemma 1 in [ChCo5].

The game between the challenger $\mathcal{C}_h$ and the adversary $\mathcal{A}_2$ begins with the challenger generating the public parameters, with the execution of algorithm `generatekeysh`, of **Basic Hyb**. As result, the public key is $K_{pub} = < q, G_1, G_2, \hat{e}, n,\ k_0, P, P_{pub}, H_2, H_3, Q_A, N_A >=< \texttt{paramsh}, Q_A, N_A >$ and the secret key $K_{\sec} = d_A$ are obtained. The challenger returns K_{pub} to $\mathcal{A}_2$ and keeps $K_{\sec}$.

The adversary $\mathcal{A}_2$ mounts an IND-CCA2 attack against **Basic Hyb** using K_{pub} and $\mathcal{A}_1$ as follows.

$\mathcal{A}_2$ chooses at random an index $I, 1 \leq I \leq q_1$, that will be associated to an identity upon which $\mathcal{A}_2$ will try to apply the challenge. $\mathcal{A}_2$ simulates the algorithm `setup` of our CL-PKE, returning to $\mathcal{A}_1$: $\texttt{params} =< q, G_1, G_2, \hat{e}, n, k_0, P, P_{pub}, H_1, H_2, H_3 >=< \texttt{paramsh}, H_1 >$ where H_1 is a random oracle controlled by $\mathcal{A}_2$.

The adversary $\mathcal{A}_1$ can issue queries to H_1 at any time. These queries are processed by the algorithm `queries to` H_1, as follows.

`queries to` H_1 When $\mathcal{A}_1$ queries H_1 for ID_i, $\mathcal{A}_2$ answers by looking up a list of triples $< ID_j, Q_j, h_j >$. The list, denoted by H_1^{list}, is initially empty. $\mathcal{A}_2$ may give one out of the following three answers: (1) If ID_i already occurs in H_1^{list}, say a triple $< ID_i, Q_i, h_i >$, then $\mathcal{A}_2$ answers with $Q_i \in G_1^*$. (2) Otherwise, if the query is on the I-th distinct identifier, then $\mathcal{A}_2$ stores $< ID_I, Q_A, \perp >$ in the list and answers $H_1(ID_I) = Q_A$. (3) Otherwise, $\mathcal{A}_2$ selects at random an integer $h_i \in Z_q^*$, computes $Q_i = h_i P \in G_1^*$, stores $< ID_i, Q_i, h_i >$ in the list and answers Q_i.

The game between the challenger $\mathcal{C}_h$ and the adversary $\mathcal{A}_2$ proceeds in the following Phases.

Phase 1 The adversary $\mathcal{A}_1$ launches this Phase 1 of its attack as a series of queries (of `extract` of partial secret key $PrivKeyL$, `publish` of public key, `replace`, `extract` of $PrivKeyR$ or `decrypt`). We can consider that $\mathcal{A}_1$ will always issue a query to H_1, for the same identity on which it will try to issue the next queries. $\mathcal{A}_2$ simulates the challenge of $\mathcal{A}_1$ and answers these queries as follows: (1) `Extract` the $PrivKeyL$ of ID_i. If $ID_i = ID_I$, then $\mathcal{A}_2$ aborts the game (**Event 1**); otherwise answers with $h_i sP$, where h_i comes from the triple corresponding to ID_i in H_1^{list} and $sP = P_{pub}$. (2) `Publish` for ID_i. To answer this query, $\mathcal{A}_2$ keeps another list, K^{list}, with quadruples of the form $< ID_i, t_i, t_i P, R_i >$, indexed by ID_i. For a new ID_i, $\mathcal{A}_2$ selects randomly an integer $t_i \in Z_q^*$ and inserts a quadruple $< ID_{i_i}, t_i, t_i P, t_i P >$ in the list, otherwise $\mathcal{A}_2$ answers with R_i. (3) `Replacement` for ID_i by N_i. $\mathcal{A}_2$ replaces R_i by N_i, in the quadruple indexed by ID_i in the K^{list}. (4) `Extract` the $PrivKeyR$ for ID_i. $\mathcal{A}_2$ verifies if $R_i = t_i P$ in the quadruple indexed by ID_i in K^{list}. If so, $\mathcal{A}_2$ returns t_i, otherwise aborts the game (**Event 2**). (5) `Decrypt` $< ID_i, C_i, N_i >$. $\mathcal{A}_2$ looks up K^{list} for a quadruple such that $N_i = t_i P$. If such quadruple does not exist, then $\mathcal{A}_2$ aborts the game (**Event 3**). If the query for decryption was for $C_i =< U, V >$ for ID_I (i.e., $ID_i = ID_I$), then $\mathcal{A}_2$ requests decryption of C_i and t_i, and gives the answer from the challenger $\mathcal{C}_h$ to $\mathcal{A}_1$. Otherwise, $\mathcal{A}_2$ tries to decrypt by asking to H_2 (which is controlled by $\mathcal{C}_h$) to obtain $H_2(U, g^r, t_i U)$, where $g^r = \hat{e}(U, h_i sP)^r$ and $h_i sP$ is obtained thru the extraction of $PrivKeyL$. By splitting $V \oplus H_2(U, g^r, t_i U)$, $\mathcal{A}_2$ gets a result which is given to $\mathcal{A}_1$.

`Challenge` At some point in time, $\mathcal{A}_1$ finishes Phase 1, chooses ID_{ch}, N_{ch} and two messages m_0, m_1 on which it wants to challenge. If $ID_{ch} \neq ID_I$, then $\mathcal{A}_2$ aborts the game (**Event 4**); otherwise, m_0, m_1 and N_{ch} are given to $\mathcal{C}_h$, that answers with the ciphertext $C_{ch} =< U', V' >$. $\mathcal{A}_2$ gives to $\mathcal{A}_1$ the ciphertext C_{ch}.

Phase 2 $\mathcal{A}_2$ keeps answering the requests the same way as in Phase 1, but it aborts the game if any decryption request on $< ID_i, C_{ch}, N_{ch} >$ is done (**Event 5**).

`Guess` If $\mathcal{A}_1$ issues a guess b', $\mathcal{A}_2$ returns the same guess b'.

The proof of Lemma 1 will continue after Proposition 2, whose proof is given in the full paper.

Proposition 2. *If the adversary $\mathcal{A}_2$ does not abort during the simulation, then the algorithm $\mathcal{A}_1$'s view is the same as its view in the real attack.*

Now it suffices to compute the probability of $\mathcal{A}_2$ not aborting during the simulation. There are five events which cause an abortion: (1) Event 1, denoted $\mathcal{H}_1$: $\mathcal{A}_1$ asked $PrivKeyL$ for ID_I; (2) Event 2, denoted $\mathcal{H}_2$: $\mathcal{A}_1$ replaced a public key of a particular identity and later asked its $PrivKeyR$; (3) Event 3, denoted $\mathcal{H}_3$: $\mathcal{A}_1$ asked to decrypt with an unknown public key N_i; (4) Event 4, denoted $\mathcal{H}_4$: $\mathcal{A}_1$ did not choose ID_I as ID_{ch}; (5) Event 5, denoted $\mathcal{H}_5$: $\mathcal{A}_2$ asked to decrypt C_{ch} in Phase 2;

Any attempt to extract $PrivKeyR$ for an entity that had its public key replaced is not allowed for $\mathcal{A}_1$. Since $\mathcal{A}_2$ only simulates the requests by $\mathcal{A}_1$, $\mathcal{H}_2$ does not happen without $\mathcal{A}_1$ aborting. A decryption request with an unknown public key does not produce a result (since $\mathcal{A}_1$ would receive $\perp$ as answer) and it is possible for $\mathcal{A}_1$ avoid this situation in its implementation. Thus, we consider $\mathcal{A}_2$ does not imply $\mathcal{H}_3$. Further, $\mathcal{H}_5$ only happens when, in the challenge, $\mathcal{A}_1$ chooses $ID_I = ID_{ch}$, but $\mathcal{A}_1$ would be aborted if it requests a decryption on $< ID_{ch}, C_{ch}, N_{ch} >$, the same way $\mathcal{A}_2$ would do. Further, the fact that $\mathcal{H}_4$ did not occur in the challenge phase (i.e.,, $ID_I = ID_{ch}$) implies that $\mathcal{H}_1$ did not occur (since if it occured, $\mathcal{A}_1$ would have asked $PrivKeyL$ of ID_{ch} and would be aborted before the challenge). Hence, we have $Pr[\mathcal{A}_2$ not aborted$] = Pr[\neg\mathcal{H}_1 \wedge \neg\mathcal{H}_2 \wedge \neg\mathcal{H}_3 \wedge \neg\mathcal{H}_4 \wedge \neg\mathcal{H}_5] = Pr[\neg\mathcal{H}_1 \wedge \neg\mathcal{H}_4 \wedge \neg\mathcal{H}_5] = Pr[\neg\mathcal{H}_4]$ The choice of I by $\mathcal{A}_2$ is independent of the choice of ID_I by $\mathcal{A}_1$, thus, $Pr[\mathcal{A}_2$ not aborted$] = 1/q_1$. $\mathcal{A}_2$ uses the guess of $\mathcal{A}_1$, whose definition says $|Pr[b = b'] - 1/2| \geq \varepsilon_1$. If $\mathcal{A}_2$ does not abort and its guess is successful, it wins the game. Combining these elements, we have the advantage: $\varepsilon_2 \geq \varepsilon_1/q_1$

For the analysis of the time complexity of $\mathcal{A}_2$, observe the following: (1) Since $\mathcal{A}_2$ basically simulates the challenge by $\mathcal{A}_1$, the execution time of $\mathcal{A}_1$, denoted t_1, is the principal component. (2) In the simulation, each extraction request of $PrivKeyL$, decrypt request and query to H_1 involves a scalar multiplication in G_1^*, that $\mathcal{A}_1$ would not do in the real attack. Considering c_{G_1} denotes the multiplication time of a point in G_1 by a scalar (random), then $c_{G_1}(q_1 + q_d + q_e)$ is another component in the computation of t_2. Then $t_2 \leq t_1 + c_{G_1}(q_1 + q_d + q_e)$. This ends the proof of Lemma 1 □

Lemma 3. *Assume H_3 is a random oracle. Let $\mathcal{A}_2$ be an IND-CCA2 adversary against the* ***Basic Hyb*** *scheme, with advantage ε_2, execution time t_2, which issues at most q_3 queries to H_3 and q_d decryption queries. Then there is an IND-CPA (defined in the Appendix) adversary $\mathcal{A}_3$ against* ***Basic*** *with advantage $\varepsilon_3 \geq \left(\varepsilon_2 - \frac{q_3}{2^{k_0-1}}\right)\left(1 - \frac{1}{q}\right)^{q_d}$ and execution time $t_3 \leq t_2 + q_3(T_{encryptb} + c.n)$ where c is a constant, $(n - k_0)$ is the plaintext message length in bits, q is the order of G_1 and $T_{encryptb}$ is the execution time of* `encryptb`.

Proof: Consequence of Theorem 5.4. in [FuOk0].□

Lemma 4. *Assume H_2 is a random oracle. Let $\mathcal{A}_3$ be an IND-CPA adversary against the* ***Basic*** *scheme, with advantage ε, execution time t_3 and it issues at most q_2 queries to H_2. Then there is an algorithm $\mathcal{B}$ that solves BDH (Bilinear Diffie Hellman) Problem in G_1 with advantage $\varepsilon \geq 2\varepsilon_3/q_2$, with time complexity $O(t_3)$.*

Proof: We will construct an algorithm $\mathcal{B}$ to solve BDH, while interacting with the adversary $\mathcal{A}_3$. Our proof is similar to Lemma 4.3 in [BoFr1], which was also used in the proof of Lemma 3 in [ChCo5].

Initially, $\mathcal{B}$ receives as input the BDH parameters $< q, G_1, G_2, P, \hat{e} >$, produced from a security parameter k. It receives an instance $< P, aP, bP, cP >$, where a, b, c are values selected at random from Z_q^*. The algorithm $\mathcal{B}$ finds the value $\hat{e}(P,P)^{abc}$, interacting with $\mathcal{A}_3$ as follows: (1) `setup`. $\mathcal{B}$ simulates the algorithm `generatekeysb` to create the public key $K_{pub} =< q, G_1, G_2, \hat{e}, n, P, P_{pub}, H_2, Q_A, N_A >$ with $P_{pub} = aP$ (i.e., $s = a$), $Q_A = bP$ and $N_A = t_A P$, where t_A is chosen randomly from Z_q^*. The secret key $K_{\sec} = d_A = sQ_A$, that $\mathcal{B}$ does not know, is $d_A = abP$. H_2 is a random oracle controlled by $\mathcal{B}$. K_{pub} is given to $\mathcal{A}_3$ and the game proceeds in the following phases. (2) `queries`. $\mathcal{B}$ answers the queries from $\mathcal{A}_3$ to the oracle H_2 (to encrypt the plaintexts) as follows: `queries` $H_2(X_i, Y_i, Z_i)$: At any time $\mathcal{A}_3$ can query H_2. $\mathcal{B}$ answers to these queries with the help of a list of quadruples $< X_i, Y_i, Z_i, H_i >$, indexed by the first three terms. The list, denoted H_2^{list}, is initially empty. $\mathcal{B}$ can give one out the two answers: (a) If (X_i, Y_i, Z_i) is an index of a quadruple in H_2^{list}, then $\mathcal{B}$ answers with the corresponding H_i. (2) otherwise, $\mathcal{B}$ selects at random a string $H_i \in \{0,1\}^n$, inserts a quadruple $< X_i, Y_i, Z_i, H_i >$ in the list and answers with H_i. (2) `challenge`. $\mathcal{A}_3$ finishes the query phase and gives two equal length messages m_0 and m_1. $\mathcal{B}$ chooses at random a string $R \in \{0,1\}^n$, it defines the ciphertext $C_{ch} =< U', V' >=< cP, R >$, which is given to $\mathcal{A}_3$. Observe that the decrypted C_{ch} is: $V' \oplus H_2(U', \hat{e}(U', d_A), cN_A) = R \oplus H_2(cP, \hat{e}(cP, abP), cN_A)$. (3) `guess`. $\mathcal{A}_3$ generates a guess $b \in \{0,1\}$ (that will not be used). Now, $\mathcal{B}$ chooses at random a quadruple $< X_i, Y_i, Z_i, H_i >$ from H_2^{list}. $\mathcal{B}$ assumes $X_i = cP$ and answers Y_i, as being the solution of $\hat{e}(cP, abP) = \hat{e}(P,P)^{abc}$.

Let $\mathcal{H}$ be the event of the algorithm $\mathcal{A}_3$ issuing a query about $H_2(cP, \hat{e}(cP, abP), cN_A) = H_2(X, Y, Z)$ at some point in time during the simulation.

The proof of Lemma 4 will continue after Proposition 5 and 6, whose proofs are given in the full paper.

Proposition 5. *$Pr[\mathcal{H}]$ in the simulation is equal to $Pr[\mathcal{H}]$ in the real attack by $\mathcal{A}_3$.*

Proposition 6. *In the real attack $Pr[\mathcal{H}] \geq 2\varepsilon_3$.*

From Proposition 5 and Proposition 6 it follows that $Pr[\mathcal{H}] \geq 2\varepsilon_3$. Hence, after the simulation, (X, Y, Z) occurs in some quadruple in H_2^{list}, with probability at least $2\varepsilon_3$. Considering that $\mathcal{A}_3$ issues q_2 distinct queries, it follows that $\mathcal{B}$ answers correctly the computation of $\hat{e}(P,P)^{abc}$ with probability at least $2\varepsilon_3/q2$.

For the time complexity analysis of $\mathcal{B}$, we need to observe that, during the queries, q_2 distinct operations are done with the list H_2^{list}. Then the time complexity $t = q_2 O(\log q_2) + O(1)$. On the other hand, $t_3 = q_2 T_{encryptb} + T_{guess}$, where $T_{encryptb}$ is the execution time of `encryptb` and T_{guess} is the time for $\mathcal{A}_3$ to produce the guess. By the definition of $\mathcal{A}_3$, its execution time is polynomial on k, as well as the time for `encryptb`. fwConsider the integers

$0 \leq x, y, z = O(k)$, with $q_2 = O(k^x)$, $T_{encryptb} = O(k^y)$, $T_{guess} = O(k^z)$. Then $t_3 = O(k^x)O(k^y) + O(k^z)$ and $t = O(k^x)\ O(\log k^x) = O(k^x)\ O(\log k) = O(t_3)$ since $y > 0$, because $T_{encryptb}$ involves computation of the pairing $\hat{e}()$. This ends the proof of Lemma 4. $\Box$

Theorem 7. *If the BDH Problem is difficult over G_1 and the hash functions H_1, H_2, and H_3 are random oracles, then the proposed CL-PKE is secure againts Type-I IND-CCA2.*

Proof: Assume there is an Type-I IND-CCA2 adversary against our CL-PKE, with a non-negligible advantage ε_1, that issues at most q_d queries for decryption, q_e queries for extraction of the partial secret key and q_1, q_2, q_3 queries to H_1, H_2, H_3, respectively, and with execution time t_1.

It follows from the three previous lemmas that there is an algorithm which solves BDH with non-negligible advantage $\varepsilon \geq \frac{2}{q_2}\left(\frac{\varepsilon_1}{q_1} - \frac{q_3}{2^{k_0-1}}\right)\left(1 - \frac{1}{q}\right)^{q_d}$ and execution time $t \leq t_1 + cG_1(q_1 + q_d + q_e) + q_3(T_{encryptb} + c_0.n) + c_1$ where c_0, c_1 are constants, $(n - k_0)$ is the size in bits of messages and $T_{encryptb}$ is the execution time of `encryptb`. Since t_1 is polynomial on k (as well as t_2 and t_3), t is also polynomial on k. If we rename all $q_i, i \in \{1, 2, 3\}$, by q_H, we obtain the approximation $\varepsilon \simeq \frac{2\varepsilon_1}{q_H^2}$. Hence our CL-PKE is secure against a Type-I IND-CCA2 adversary.$\Box$

8. Type-II adversary and a sequence of reductions

The Gap Diffie-Hellman - GDH - Problem is considered in the following:

Theorem 1. *If the GDH Problem is difficult in G_1 and H_1, H_2, H_3 are random oracles, then our CL-PKE scheme is secure against Type-II IND-CCA2 adversaries.*

Proof: It is given in the full paper.

9. Conclusions

We have constructed a CL-PKE scheme that does not allow key escrow, which is undesirable in many applications. We analyzed both its efficiency and security. It is more efficient than previously published CL-PKE schemes, and we proved it is strong against IND-CCA2 attack, under the Random Oracle Model.

References

[AlPa3] AL-RIYAMI, S. S.; PATERSON, K. G. Certificateless Public Key Cryptography. 2003. Cryptology ePrint Archive, Report 2003/126. http://eprint.iacr.org/.

[AlPa5] AL-RIYAMI, S. S.; PATERSON, K. G. C: CBE from CL-PKE: A generic construction and efficient schemes. In: Public Key Cryptography - PKC 2005. [S.l.: s.n.], 2005. pp. 398-415.

[Be8a] BELLARE, M. et al. Relations among notions of security for public-key encryption schemes. In: CRYPTO 98: Proceedings of the 18th Annual IACR Conference on Advances in Cryptology. Springer-Verlag, 1998. pp. 26-45. ISBN 3-540-64892-5.

[Be8] BELLARE, M.; Practice-oriented provable-security. In: ISW '97: Proceedings of the First International Workshop on Information Security. London, UK: Springer-Verlag, 1998. pp 221-231. ISBN 3-540-64382-6.

[BoFr1] BONEH, D.; FRANKLIN, M. K. Identity-based encryption from the Weil pairing. In: CRYPTO 01: Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology. London, UK: Springer-Verlag, 2001. pp. 213-229. ISBN 3-540-42456-3. http://eprint.iacr.org/2001/090/.

[ChCo5] CHENG, Z.; COMLEY, R. Efficient Certificateless Public Key Encryption. 2005. Cryptology ePrint Archive, Report 2005/012. http://eprint.iacr.org/.

[ChLe2] CHEON, J. H.; LEE, D. H. Diffie-Hellman Problems and Bilinear Maps. 2002. Cryptology ePrint Archive, Report 2002/117. http://eprint.iacr.org/.

[CrSh4] CRAMER, R.; SHOUP, V. Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adptative Chosen Ciphertext Attack, SIAM Journal of Computing 33:167-226, 2003.

[FuOk9] FUJISAKI, E.; OKAMOTO, T. Secure Integration os Asymetric and Symmetric Encryption Schemes. In: CRYPTO 99: Proceedings of the Annual IACR Conference on Advances in Cryptology. Springer-Verlag LNCS #1666, 1999. pp. 535-544.

[FuOk0] FUJISAKI, E.; OKAMOTO, T. How to Enhance the Security of Public-Key Encryption at Minimum Cost, IEICE Transactions on Fundamentals, vol. E83-A, number 1, 2000, ISBN 3-540-65644-8, pp.24–32.

[Ga5] GALINDO, D. Boneh-Franklin Identity Based Encryption Revisited. 2005. Cryptology ePrint Archive, Report 2005/117. http://eprint.iacr.org/.

[Ge3] GENTRY, C. Certificate-Based Encryption and the Certificate Revocation Problem. 2003 http://eprint.iacr.org/2003/183/.

[Sh4] SHAMIR, A. Identity-based cryptosystems and signature schemes. In: Proceedings of CRYPTO 84 on Advances in cryptology. New York, NY, USA: Springer-Verlag New York, Inc., 1984. pp. 47-53. ISBN 0-387-15658-5.