Constructing Efficient Certificate-based Encryption with Paring

Yang Lu* and Jiguo Li
College of Computer and Information Engineering, Hohai University, Nanjing, China
Email: {luyangnsd, ljg1688}@163.com

Junmo Xiao
Institute of Communication Engineering, PLA University of Science and Technology, Nanjing, China
Email: jmxiao753@vip.sina.com

Abstract—**The certificate-based encryption (CBE) is a new PKC paradigm which combines traditional public-key encryption (PKE) and identity based encryption (IBE) while preserving their features. CBE provides an efficient implicit certificate mechanism to eliminate third-party queries for the certificate status and to simply the certificate revocation problem. Therefore, CBE can be used to construct an efficient PKI requiring fewer infrastructures. In addition, it also solves the key escrow problem and key distribution problem inherent in IBE. In this paper, we construct an efficient CBE scheme with paring and prove it to be CCA-secure in the random oracle model based on the hardness of the Bilinear Diffie-Hellman Inversion assumption. When compared with other existing CBE schemes, our scheme has obvious advantage in the computation performance.**

Index Terms—**certificate-based encryption, pairing, *p*-BDHI assumption, random oracle model**

I. INTRODUCTION

In Eurocrypt 2003, Gentry [1] introduced the notion of certificate-based encryption (CBE), which combines identity-based encryption (IBE) and traditional PKI-supported public key encryption (PKE) while preserving their features. CBE provides an implicit certification mechanism for a traditional PKI and allows a periodical update of certificate status. As traditional PKIs, each user in CBE generates his own public/private key pair and requests a long-lived certificate from the CA. This long-lived certificate has all the functionalities of a traditional PKI certificate. But, CA generates the long-lived certificate as well as short-lived certificates (i.e., certificate status). A short-lived certificate can be pushed only to the owner of the public/private key pair and acts as a partial decryption key. This additional functionality provides an implicit certification so that the sender is not required to obtain fresh information on certificate status and the recipient can only decrypt the ciphertext using his private key along with an up-to-date short-lived certificate from its CA. The feature of implicit certification allows us to eliminate third-party queries for the certificate status and simply the public key revocation problem so that CBE does not need infrastructures like CRL [29] and OCSP [30]. Therefore, CBE can be used to construct an efficient PKI [21] requiring fewer infrastructures than previous proposals [32-35]. Furthermore, there is no key escrow problem (since the CA does not know the private keys of users) and key distribution problem (since the certificates need not be kept secret) in CBE.

Since the introduction of CBE in [1], in which Gentry proposed the first concrete CBE scheme based on the BF-IBE scheme [23] and proved its security in the random oracle model [18,19], there are different variants or improvements proposed in the literature later on. Yum and Lee [4] provided a formal equivalence theorem among IBE, certificateless public key encryption (CL-PKE) [2] and CBE. They showed that IBE implies both CBE and CL-PKE by giving a generic construction from IBE to those primitives. They [5] also proposed a method to generically construct of CL-PKE from IBE and PKE, which can also be adapted to generically construct CBE. However, Galindo et al. [6] pointed out that a dishonest authority could break the security of the three generic constructions given in [4, 5]. These constructions were inherently flawed due to a naive use of double encryption without further treatments. We solved this problem by providing two security-enhancing conversions in [9] and achieving two generic CBE constructions from PKE and IBE in [10, 11], which are provably CCA-secure in the random oracle model. Al-Riyami and Paterson [3] gave an analysis of Gentry's CBE concept and repaired a number of problems with the original definition and security model for CBE. They also presented a generic conversion of CBE from CL-PKE and claimed that a secure CBE scheme could be constructed from any secure CL-PKE scheme using this conversion. Kang and Park [12] pointed out that their conversion was incorrect due to the flaw in their security proof. In [14], Dodis and Katz gave generic techniques to build CCA-secure multiple-encryption schemes from PKE schemes which are individually CCA-secure. They showed that their method could be applied to an IBE and a PKE (instead of two

*Corresponding author.

PKEs) and to build CBE schemes without resorting to the random oracle model. Recently, Wang et al. [15] proposed a certificate-based proxy cryptosystem based on Gentry's CBE scheme. Moreover, Morillo and Ràfols [8] proposed the first construction of CBE scheme secure in the standard model. Galindo et al. [7] proposed an improved CBE scheme based on the CBE scheme in [8]. In parallel to CBE, Kang et al. [13] proposed the security notion of certificate-based signature (CBS) that follows the idea of Gentry's CBE scheme and provided a concrete CBS scheme in the random oracle model. However, Li et al. [16] pointed out that this signature scheme was insecure against the key replacement attack. They refined the security model of CBS given in [13] and constructed a new CBS scheme which is secure in the random oracle model. Moreover, Au et al. [17] propose a new notion called certificate-based ring signature, which is the ring signature in certificate based cryptography setting.

In this paper, we propose a new efficient CBE scheme with pairing by combining the SK-IBE scheme [20,21] with a traditional ElGamal-like cryptosystem [22], and prove it to be CCA-secure in the random oracle model. Due to the advantage of SK-IBE in the computation performance, our CBE scheme requires computing no parings in the encryption algorithm and computing only one pairing in the encryption algorithm. When compared with other existing CBE schemes [1,7,8], our scheme has obvious advantage in the computation performance.

II. Background Definitions

Throughout the paper, G_1 and G_2 denote two additive cyclic groups of prime order q and G_T a multiplicative cyclic group of the same order. P_1 denotes a generator of G_1 and P_2 denotes a generator of G_2. φ is an isomorphism from G_2 to G_1 with $\varphi(P_2) = P_1$. Note that from [25], we can either assume that φ is efficiently computable or make our security proof relative to some oracle which computes φ. For us, a bilinear paring is a map e: $G_1 \times G_2 \rightarrow G_T$ with following properties:

Bilinear: For all $P \in G_1$, all $Q \in G_2$ and all $a, b \in Z$ we have $e(aP, bQ) = e(P, Q)^{ab}$.

Non-degenerate: $e(P_1, P_2) \neq 1_{G_T}$.

Computable: For all $P \in G_1$ and $Q \in G_2$, $e(P, Q)$ can be efficiently computed.

A bilinear paring satisfying the three properties above is said to be an *admissible* bilinear map. Typically, the map e can be derived from either the Weil or Tate paring on an elliptic curve over a finite field.

The security of the CBE scheme in this paper is proved based on the difficulty of the following problem which is introduced in [21,24].

***p*-Bilinear Diffie-Hellman Inversion (*p*-BDHI) Assumption:** For an integer p and a random element $x \in Z_q^*$, $P_2 \in G_2^*$, $P_1 = \varphi(P_2)$, e: $G_1 \times G_2 \rightarrow G_T$, given $(P_1, P_2, xP_2, x^2P_2, \ldots x^pP_2,)$, computing $e(P_1, P_2)^{1/x}$ is hard.

In [21], Chen and Cheng have proved that the p-BDHI assumption and the standard Bilinear Diffie-Hellman (BDH) assumption are polynomial time equivalent, i.e., if there exists a polynomial time algorithm to solve BDH, then there exists a polynomial time algorithm for 1-BDHI, and vice versa.

III. Certificate-based Encryption

In this section, we briefly review the definition and security model for CBE. These definitions are taken from [3, 12], where the original definitions given in [1] were reconsidered.

Definition 1. A certificate-based encryption (CBE) scheme is a tuple of five PPT algorithms (*Setup*, *SetKeyPair*, *Certify*, *Enc*, *Dec*) such that:

- *Setup* is a probabilistic algorithm taking as input a security parameter k. It returns the certifier's master-key sk_{CA} and public parameters *params* that include the descriptions of a finite message (plaintext) space *MSPC*, a finite ciphertext space *CSPC*, and two string spaces T and Λ. We consider *params* to be an implicit input to the rest of the algorithms.
- *SetKeyPair* is a probabilistic algorithm that outputs a public/private key pair $<pk, sk>$.
- *Certify* is a deterministic certification algorithm that takes as input $<sk_{CA}, \tau \in T, id \in \Lambda, pk>$. It returns a short-live certification $Cert_{id,\tau}$, which is sent to the user. Here τ is a string identifying a time period, while *id* contains other information needed to certify the user, and *pk* is the user's public key.
- *Enc* is a probabilistic algorithm taking as inputs $<\tau, id, pk, M>$. It returns a ciphertext C for message M.
- *Dec* is a deterministic algorithm taking $<Cert_{id,\tau}, sk, C>$ as input in time period τ. It returns either a message M or the special symbol $\perp$ indicating a decryption failure.

Naturally, it is required that these algorithms must satisfy the standard consistency constraint, that is for all $M \in MSPC$, $Dec(Cert_{id,\tau}, sk, C)=M$ where $C=Enc(\tau, id, pk, M; \sigma)$, $Cert_{id,\tau} = Certify(sk_{CA}, \tau, id, pk)$ and $<pk, sk>$ is a valid public/private key pair.

The adaptive chosen-ciphertext security for CBE is defined against two different types of adversaries. The Type I adversary $\mathcal{A}_1$ has no access to the master-key and models an uncertified user. The Type II adversary $\mathcal{A}_2$ models an honest-but-curious certifier who possesses the master-key sk_{CA} attacking a fixed user's public key.

Game 1: The challenger $\mathcal{C}$ runs *Setup* algorithm, gives *params* to the adversary $\mathcal{A}_1$ and keeps sk_{CA} to itself.

Phase 1. The adversary $\mathcal{A}_1$ interleaves certification and decryption queries, $\mathcal{C}$ handles these queries as follows:

On certification query $<\tau, id, pk, sk>$, $\mathcal{C}$ checks that $id \in \Lambda$ and $<pk, sk>$ is a valid key-pair. If so, it runs *Certify* on input $<sk_{CA}, \tau, id, pk>$ to generate $Cert_{id,\tau}$; else it returns $\perp$.

On decryption query $<\tau, id, pk, sk, C>$, $\mathcal{C}$ checks that $id \in \Lambda$ and $<pk, sk>$ is a valid key-pair. If so, it runs *Certify* on input $<sk_{CA}, \tau, id, pk>$ to obtain $Cert_{id,\tau}$ and outputs $Dec(Cert_{id,\tau}, sk, C)$; else it returns $\perp$.

These queries may be asked adaptively, that is, they may depend on the answers to previous queries.

Challenge. On challenge query $<\tau_{ch}, id_{ch}, pk_{ch}, sk_{ch}, M_0, M_1>$, where $M_0, M_1 \in MSPC$ are of equal length, $\mathcal{C}$ checks that $id_{ch} \in \Lambda$, $<pk_{ch}, sk_{ch}>$ is a valid key-pair and $<\tau_{ch}, id_{ch}, pk_{ch}, sk_{ch}>$ was not the subject of a certification query in phase 1. If so, it picks a random bit $b \in \{0, 1\}$, encrypts M_b under the challenge public key pk_{ch} and sends the resulting ciphertext C^* to $\mathcal{A}_1$; else it returns $\perp$.

Phase 2. As in phase 1, with the restriction that $<\tau_{ch}, id_{ch}, pk_{ch}, sk_{ch}, C^*>$ is not the subject of a decryption query and $<\tau_{ch}, id_{ch}, pk_{ch}, sk_{ch}>$ is not the subject of a certification query.

Guess. The adversary $\mathcal{A}_1$ outputs a guess $b' \in \{0, 1\}$ and wins the game if $b = b'$. $\mathcal{A}_1$'s advantage in this game is defined to be $Adv(\mathcal{A}_1) := |2\Pr[b = b']-1|$.

Game 2: $\mathcal{C}$ runs *Setup* algorithm, gives *params* and sk_{CA} to the adversary $\mathcal{A}_2$. $\mathcal{C}$ then runs *SetKeyPair* to obtain a key-pair $<pk_{ch}, sk_{ch}>$ and gives pk_{ch} to $\mathcal{A}_2$.

Phase 1. $\mathcal{A}_2$ issues a series of decryption queries of the form $<\tau, id, C>$. On this query, $\mathcal{C}$ checks that $id \in \Lambda$. If so, it runs *Certify* on input $<sk_{CA}, \tau, id, pk_{ch}>$ to obtain $Cert_{id,\tau}$ and outputs $Dec(Cert_{id,\tau}, sk_{ch}, C)$; else it returns $\perp$. These queries may be asked adaptively.

Challenge. On challenge query $<\tau_{ch}, id_{ch}, M_0, M_1>$, where $M_0, M_1 \in MSPC$ are of equal length, $\mathcal{C}$ checks that $id_{ch} \in \Lambda$. If so, it picks a random bit $b \in \{0, 1\}$, encrypt M_b under the challenge public key pk_{ch} and sends the resulting ciphertext C^* to $\mathcal{A}_2$; else it returns $\perp$.

Phase 2. As in phase 1, with the restriction that $<\tau_{ch}, id_{ch}, C^*>$ is not the subject of a decryption query.

Guess. $\mathcal{A}_2$ outputs a guess $b' \in \{0, 1\}$ and wins the game if $b = b'$. $\mathcal{A}_2$'s advantage in this game is defined to be $Adv(\mathcal{A}_2) := |2\Pr[b = b']-1|$.

Definition 2. A CBE scheme is secure against adaptive chosen ciphertext attacks (or IND-CBE-CCA) if no polynomial-time adversary has non-negligible advantage in either Game 1 or Game 2.

Similarly, we can define the weak security notion IND-CBE-CPA for CBE schemes, in which the adversaries are disallowed to issue any decryption queries.

IV. An Efficient CBE Scheme

In this section, we first build a basic CBE scheme called BasicCBE which is IND-CBE-CPA secure. Then we extend BasicCBE to an IND-CBE-CCA secure CBE scheme called FullCBE by using a security enhancing transformation introduced in [11].

A. BasicCBE

The scheme BasicCBE is consisted of the following five polynomial time algorithms:

Setup: Given a security parameter $k \in Z^+$, the parameter generator takes the following steps:

1. Generate three cyclic groups G_1, G_2 and G_T of prime order q, an isomorphism φ from G_2 to G_1, and a bilinear pairing map e: $G_1 \times G_2 \rightarrow G_T$. Pick a random generator $P_2 \in G_2^*$ and set $P_1 = \varphi(P_2)$.
2. Pick a random $s \in Z_q^*$ and compute $P_{pub} = sP_1$.
3. Compute $g = e(P_1, P_2)$.
4. Pick two hash functions $H_1 : \{0,1\}^* \rightarrow Z_q^*$ and $H_2 : G_T \times G_T \rightarrow \{0,1\}^n$ for some integer $n > 0$.

The message space is $\{0,1\}^n$. The ciphertext space is $G_1^* \times \{0,1\}^n$. The system parameters are *params* = $\{q, G_1, G_2, G_T, \varphi, e, n, P_1, P_2, g, P_{pub}, H_1, H_2\}$. The certifier's master key is s.

SetKeyPair: This algorithm picks a random $x \in Z_q^*$ as a user's private key SK and sets the corresponding public key as $PK = g^x$.

Certify: On input $<s, \tau, id, PK>$, this algorithm outputs $Cert_{id,\tau} = \frac{1}{H_1(\tau \| id \| PK) + s} P_2$.

Enc: On input $<\tau, id, PK, M>$, this algorithm performs the following steps:

1. Check that PK is in G_2^*, if not output $\perp$. This checks the validity of the public key.
2. Compute $Q_{id} = H_1(\tau \| id \| PK)P_1 + P_{pub}$.
3. Pick a random $r \in Z_q^*$ and compute the ciphertext $C = <U, V> = < rQ_{id}, M \oplus H_2(g^r, PK^r) >$.

Dec: On input $<Cert_{id,\tau}, SK, C = <U, V>>$, this algorithm computes the plaintext $M = V \oplus H_2(e(U, Cert_{id,\tau}), (e(U, Cert_{id,\tau}))^{SK})$.

In the above construction, the certificates are short signatures computed using a signature scheme considered in [28]. As proven in Theorem 3 of [28], this signature scheme is existentially unforgeable under chosen-message attack [27] in the random oracle model, provided that the k-sCCA1 assumption is sound in G_2.

The consistency of the construction is easy to check as we have

$$e(U, Cert_{id,\tau}) = e(r(H_1(\tau \| id \| PK)P_1 + P_{pub}), \frac{1}{H_1(\tau \| id \| PK) + s} P_2) = e(P_1, P_2)^r = g^r.$$

B. FullCBE

The scheme FullCBE is consisted of the following five polynomial time algorithms:

Setup: As in the BasicCBE. In addition, we select two hash functions $H_3 : \{0,1\}^* \rightarrow Z_q^*$ and $H_4 : \{0,1\}^n \rightarrow \{0,1\}^n$. Now, the message space is $\{0,1\}^n$ and the ciphertext space is $G_1^* \times \{0,1\}^{n+k_0}$ for some integer $k_0 > 0$.

SetKeyPair: As in the scheme BasicCBE.

Certify: As in the scheme BasicCBE.

Enc: On input $<\tau, id, PK, M>$, this algorithm performs the following steps:

1. Check that PK is in G_2^*, if not output $\perp$. This checks the validity of the public key.
2. Compute $Q_{id} = H_1(\tau \| id \| PK)P_1 + P_{pub}$.
3. Pick a random $\sigma \in Z_q^*$ and compute $r = H_3(M \| \sigma \| \tau \| id \| PK)$.
4. Compute and output the ciphertext $C = <U, V, W> = < rQ_{id}, \sigma \oplus H_2(g^r, PK^r), M \oplus H_4(\sigma) >$.

Dec: On input $<Cert_{id,\tau}, SK, C = <U, V, W>>$, this algorithm computes $\sigma = V \oplus H_2(e(U, Cert_{id,\tau}), (e(U, Cert_{id,\tau}))^{SK})$, and then $M = W \oplus H_4(\sigma)$. The message is accepted iff $U = r(H_1(< \tau \| id \| PK >)P_1 + P_{pub})$ with $r = H_3(M \| \sigma \| \tau \| id \| PK)$.

V. Security Proof

To prove the security of FullCBE, we define the following two public key encryption schemes called BasicPub-I and BasicPub-II.

The public key encryption scheme BasicPub-I is specified by following three algorithms:

Keygen: Given a security parameter $k \in Z^+$, the parameter generator takes the following steps:

1. Generate the parameters $\{q, G_1, G_2, G_T, \varphi, e, n, P_1, P_2, g\}$ which are identical to the ones of NewBasicCBE.
2. Pick a random $s \in Z_q^*$ and compute $P_{pub} = sP_1$. Randomly choose different elements $h_i \in Z_q^*$ and compute $\frac{1}{h_i + s} P_2$ for $0 \le i < q_1$.
3. Pick a random $x \in Z_q^*$ and compute $PK = g^x$.
4. Pick a hash function $H_2 : G_T \times G_T \to \{0,1\}^n$.

The message space is $\{0,1\}^n$. The ciphertext space is $G_1^* \times \{0,1\}^n$. The public key is $K_{pub} = \{q, G_1, G_2, G_T, \varphi, e, n, P_1, P_2, g, P_{pub}, x, PK, h_0, (h_1, \frac{1}{h_1 + s} P_2), \ldots, (h_{q_1-1}, \frac{1}{h_{q_1-1} + s} P_2), H_2\}$ and the private key is $K_{pri} = \frac{1}{h_0 + s} P_2$.

Encrypt: To encrypt M, this algorithm picks a random $r \in Z_q^*$ and computes the ciphertext

$C = <U, V> = < r(h_0 P_1 + P_{pub}), M \oplus H_2(g^r, PK^r) >$.

Decrypt: Given the ciphertext $C = <U, V>$, this algorithm computes the plaintext

$M = V \oplus H_2(e(U, K_{pri}), (e(U, K_{pri}))^x)$.

The public key encryption scheme BasicPub-II is specified by following three algorithms:

Keygen: Given a security parameter $k \in Z^+$, the parameter generator takes the following steps:

1. Generate the parameters $\{q, G_1, G_2, G_T, \varphi, e, n, P_1, P_2, g\}$ which are identical to the ones of BasicCBE.
2. Pick a random $s \in Z_q^*$ and compute $P_{pub} = sP_1$. Randomly choose an element $h_0 \in Z_q^*$.
3. Pick a random $x \in Z_q^*$ and compute $PK = g^x$.
4. Pick a hash function $H_2 : G_T \times G_T \to \{0,1\}^n$.

The message space is $\{0,1\}^n$. The ciphertext space is $G_1^* \times \{0,1\}^n$. The public key is $K_{pub} = \{q, G_1, G_2, G_T, \varphi, e, n, P_1, P_2, g, s, P_{pub}, PK, h_0, H_2\}$ and the private key is $K_{pri} = x$.

Encrypt: To encrypt $M \in \mathbb{M}$, this algorithm picks a random $r \in Z_q^*$ and computes the ciphertext

$C = <U, V> = < r(h_0 P_1 + P_{pub}), M \oplus H_2(g^r, PK^r) >$.

Decrypt: Given the ciphertext $C = <U, V>$, this algorithm computes the plaintext

$M = V \oplus H_2(e(U, \frac{1}{h_0 + s} P_2), (e(U, \frac{1}{h_0 + s} P_2))^{K_{pri}})$.

In the following, we first prove the security of the scheme BasicCBE based on the security of the PKE schemes defined above.

Lemma 1. *Suppose that H_1 is a random oracle and that there exists a Type I IND-CBE-CPA adversary $\mathcal{A}_1$ against BasicCBE with advantage $\varepsilon(k)$ which makes at most q_1 distinct queries to H_1. Then there exists an IND-CPA adversary $\mathcal{B}$ against BasicPub-I with advantage at least $\varepsilon(k)/q_1$.*

Proof. We show how to make use of the adversary $\mathcal{A}_1$ to construct an IND-CPA adversary $\mathcal{B}$ against BasicPub-I. The IND-CPA challenger $\mathcal{C}$ starts an IND-CPA Game by running the algorithm of BasicPub-I.Keygen to generate a public key $K_{pub} = \{q, G_1, G_2, G_T, \varphi, e, n, P_1, P_2, g, P_{pub}, x, PK, h_0, (h_1, \frac{1}{h_1 + s} P_2), \ldots, (h_{q_1-1}, \frac{1}{h_{q_1-1} + s} P_2), H_2\}$ and the corresponding private key $K_{pri} = \frac{1}{h_0 + s} P_2$. The challenger $\mathcal{C}$ passes K_{pub} to $\mathcal{B}$. Now, $\mathcal{B}$ interacts with $\mathcal{A}_1$ as follows:

Setup: $\mathcal{B}$ chooses an index I with $1 \le I \le q_1$. Then $\mathcal{B}$ gives $\mathcal{A}_1$ the BasicCBE public parameters $params = \{q, G_1, G_2, G_T, \varphi, e, n, P_1, P_2, g, P_{pub}, H_1, H_2\}$ where H_1 is a random oracle controlled by $\mathcal{B}$ as described below.

***H_1*-queries:** At any time $\mathcal{A}_1$ can query the random oracle H_1. To response to these queries $\mathcal{B}$ maintains a list of tuples $<(\tau \| id \| PK)_i, h_i, Cert_i>$ indexed by $(\tau \| id \| PK)_i$ as explained below. We refer to this list as the H_1^{list}. The list is initially empty. When $\mathcal{A}_1$ queries the oracle H_1 at a point $(\tau \| id \| PK)_i$, $\mathcal{B}$ responds as follows:

1. If the query $(\tau \| id \| PK)_i$ already appears in the H_1^{list} in a tuple $<(\tau \| id \| PK)_i, h_i, Cert_i>$, then $\mathcal{B}$ responds

with $H_1((\tau||id||PK)_i) = h_i$.

2. Otherwise, if the query is on the *Ith* distinct $\tau||id||PK$, then $\mathcal{B}$ returns $H_1((\tau||id||PK)_I) = h_0$ and adds the tuple $<(\tau||id||PK)_I, h_0, \perp>$ into the H_1^{list}.

3. Otherwise, $\mathcal{B}$ selects a random integer h_i ($i > 0$) from K_{pub} which has not been chosen by $\mathcal{B}$ and adds the tuple $<(\tau||id||PK)_i, h_i, \frac{1}{h_i + s} P_2>$ into the H_1^{list}. $\mathcal{B}$ responds with $H_1((\tau||id||PK)_i) = h_i$.

Phase 1: $\mathcal{A}_1$ launches Phase 1 of its attack by making a series of certification queries. Let $<\tau, id, PK, SK>$ be a certification query issued by $\mathcal{A}_1$. On this query, $\mathcal{B}$ first searches list H_1^{list}. If $\tau||id||PK$ is not on the list, $\mathcal{B}$ queries $H_1(\tau||id||PK)$. Then $\mathcal{B}$ checks the value *Cert* in the tuple $<(\tau||id||PK), h, Cert>$: if $Cert \neq \perp$, $\mathcal{B}$ responds with *Cert*; otherwise, $\mathcal{B}$ aborts the game (**Event 1**).

Challenge: At some point, $\mathcal{A}_1$ decides to end Phase 1 and outputs $<\tau_{ch}, id_{ch}, PK_{ch}, SK_{ch}, M_0, M_1>$ on which it wants to be challenged. $\mathcal{B}$ responds as follows:

1. If the *I*th query on H_1 has not been issued, $\mathcal{B}$ inserts the tuple $<(\tau_{ch}||id_{ch}||PK_{ch}), h_0, \perp>$ into H_1^{list}.

2. Otherwise, if $(\tau_{ch}||id_{ch}||PK_{ch}) \neq (\tau||id||PK)_I$, then $\mathcal{B}$ aborts the game (**Event 2**).

3. Otherwise, $\mathcal{B}$ gives $\mathcal{C}$ the pair $<M_0, M_1>$ as its challenge. $\mathcal{C}$ chooses a random bit $b \in \{0,1\}$, encrypts M_b and responds with the ciphertext C_{ch}. Then $\mathcal{B}$ forwards C_{ch} to $\mathcal{A}_1$.

Phase 2: As in phase 1, with the restriction that $<\tau_{ch}, id_{ch}, PK_{ch}, SK_{ch}>$ is not the subject of a certification query.

Guess: Finally, $\mathcal{A}_1$ outputs its guess b' for b, $\mathcal{B}$ outputs the same b' as its own guess.

Obviously, if $\mathcal{B}$ does not abort during the simulation then $\mathcal{A}_1$'s view is identical to its view in the real attack. As shown above, $\mathcal{B}$ could abort when one of the following events happens: (1) **Event 1**, denoted by $\mathcal{H}_1$: $\mathcal{A}_1$ queries the certification for $\tau_{ch}||id_{ch}||PK_{ch}$ at some point; (2) **Event 2**, denoted by $\mathcal{H}_2$: $\mathcal{A}_1$ wants to be challenged on $<\tau_{ch}, id_{ch}, PK_{ch}>$ and $(\tau_{ch}||id_{ch}||PK_{ch}) \neq (\tau||id||PK)_I$. Notice that $\neg\mathcal{H}_2$ implies that $\neg\mathcal{H}_1$. Hence, the probability that $\mathcal{B}$ does not abort during the simulation is $\Pr[\neg\mathcal{H}_1 \wedge \neg\mathcal{H}_2] = \Pr[\neg\mathcal{H}_2] = 1/q_1$. This shows that $\mathcal{B}$'s advantage is at least $\varepsilon(k)/q_1$. □

Lemma 2. *Suppose that H_2 is a random oracle and that there exists an IND-CPA adversary $\mathcal{A}$ against BasicPub-I with advantage $\varepsilon(k)$ which makes at most q_2 distinct queries to H_2. Then there exists an algorithm $\mathcal{B}$ to solve the q_1-BDHI problem with advantage at least $\varepsilon(k)/q_2$.*

Proof. Algorithm $\mathcal{B}$ is given as input a random q_1-BDHI instance $<q, G_1, G_2, G_T, \varphi, e, P_1, P_2, xP_2, x^2P_2, \ldots, x^{q_1}P_2>$ where x is a random element from Z_q^*. $\mathcal{B}$ finds $e(P_1, P_2)^{1/x}$ with advantage at least $\varepsilon(k)/q_2$ by interacting with $\mathcal{A}$ as follows:

Setup: Algorithm $\mathcal{B}$ first simulates the algorithm Keygen of BasicPub-I to create the public key as below:

1. Randomly choose different $h_0, \ldots, h_{q_1-1} \in Z_q^*$ and let $f(z)$ be the polynomial $f(z) = \prod_{i=1}^{q_1-1}(z + h_i)$. Reformulate $f(z)$ to get $f(z) = \sum_{i=0}^{q_1-1} c_i z^i \in Z_q[z]$. The constant term c_0 is non-zero because $h_i \neq 0$ and c_i are computable from h_i.

2. Compute $Q_2 = \sum_{i=0}^{q_1-1} c_i x^i P_2 = f(x)P_2$, $Q_1 = \varphi(Q_2) = f(x)P_1$ and $g = e(Q_1, Q_2)$.

3. Compute $P_{pub} = \varphi(xf(x)P_2) - h_0 f(x)P_1 = (x - h_0)f(x)P_1$.

4. Compute $f_i(z) = f(z)/(z + h_i) = \sum_{j=0}^{q_1-2} d_j z^j$ and $\frac{1}{x + h_i} Q_2 = f_i(x)P_2 = \sum_{j=0}^{q_1-2} d_j x^j P_2$ for $1 \le i < q_1$.

5. Compute $T^* = e(\varphi(Q_2 - c_0 P_2), Q_2 + c_0 P_2) = e(P_1, P_2)^{f^2(x) - c_0^2}$.

6. Pick a random $y \in Z_q^*$ and compute $PK = g^y$.

Now, $\mathcal{B}$ passes $\mathcal{A}$ the public key $K_{pub} = \{q, G_1, G_2, G_T, \varphi, e, n, Q_1, Q_2, g, P_{pub}, y, PK, h_0, (h_1 + h_0, \frac{1}{h_1 + x} Q_2), \ldots, (h_{q_1-1} + h_0, \frac{1}{h_{q_1-1} + x} Q_2), H_2\}$ and the private key is $K_{pri} = \frac{1}{x} Q_2$ which $\mathcal{B}$ does not know. H_2 is a random oracle controlled by $\mathcal{B}$. Note that $e((h_i + h_0)Q_1 + P_{pub}, \frac{1}{h_i + x} Q_2) = e(Q_1, Q_2)$ for $1 \le i \le q_1 - 1$ and $e(h_0 Q_1 + P_{pub}, K_{pri}) = e(Q_1, Q_2)$. Hence K_{pub} is a valid public key of BasicPub-I.

***H_2*-queries:** At any time $\mathcal{A}$ can query the random oracle H_2. To response to these queries $\mathcal{B}$ maintains a list of tuples $<\gamma_1, \gamma_2, \zeta>$ where $\gamma_2 = \gamma_1^y$. We refer to this list as H_2^{list}. The list is initially empty. When $\mathcal{A}$ queries the oracle H_2 at a point $<\gamma_1, \gamma_2>$, $\mathcal{B}$ responds as follows:

1. If the query $<\gamma_1, \gamma_2>$ already appears in the H_2^{list} in a tuple $<\gamma_1, \gamma_2, \zeta>$, then $\mathcal{B}$ responds with $H_2(\gamma_1, \gamma_2) = \zeta$.

2. Otherwise, $\mathcal{B}$ chooses a random $\zeta \in \{0,1\}^n$, returns $H_2(\gamma_1, \gamma_2) = \zeta$ and inserts a new tuple $<\gamma_1, \gamma_2, \zeta>$ to H_2^{list}.

Challenge: Adversary $\mathcal{A}$ outputs two messages M_0 and M_1 on which it wants to be challenged. $\mathcal{B}$ chooses a random $R \in \{0,1\}^n$ and a random $r \in Z_q^*$, and then defines the challenge ciphertext to be $C_{ch} = <U, V> = <rQ_1, R>$. Observe that the decryption of C_{ch} is $V \oplus H_2(\gamma, \gamma^y)$ where $\gamma = e(U, K_{pri}) = e(Q_1, Q_2)^{\frac{r}{x}}$.

Guess: After $\mathcal{A}$ outputs its guess, $\mathcal{B}$ picks a random tuple $<\gamma_1, \gamma_2, \zeta>$ from H_2^{list}. $\mathcal{B}$ first computes $T = \gamma_1^{\frac{1}{r}}$, and then returns $(T / T^*)^{\frac{1}{c_0^2}}$. Note that $(T / T^*)^{\frac{1}{c_0^2}} = e(P_1, P_2)^{\frac{1}{x}}$ if $T = e(Q_1, Q_2)^{\frac{1}{x}}$.

Next, we analyze $\mathcal{B}$'s probability of outputting the correct answer.

Let $\mathcal{H}$ be the event that $\mathcal{A}$ issues a query for $H_2(e(Q_1, Q_2)^{\frac{r}{x}}, e(Q_1, Q_2)^{\frac{ry}{x}})$ at some point during the simulation above. We first show that $\Pr[\mathcal{H}] \geq \varepsilon(k)$. As H_2 are random oracles, $\Pr[\mathcal{A} \text{ wins} | \neg\mathcal{H}] = 1/2$.

Then we have that

$$\Pr[\mathcal{A} \text{ wins}] \leq \Pr[\mathcal{H}] + \frac{1}{2}(1 - \Pr[\mathcal{H}]) = \frac{1}{2} + \frac{1}{2}\Pr[\mathcal{H}],$$

$$\Pr[\mathcal{A} \text{ wins}] \geq \Pr[\mathcal{A} \text{ wins} | \neg\mathcal{H}] \Pr[\neg\mathcal{H}]$$

$$= \frac{1}{2}(1 - \Pr[\mathcal{H}]) = \frac{1}{2} - \frac{1}{2}\Pr[\mathcal{H}].$$

It follows that $\varepsilon(k) = |2\Pr[\mathcal{A} \text{ wins}] - 1| \leq \Pr[\mathcal{H}]$.

Assume that $\mathcal{A}$ has issued q_2 distinct queries to H_2. Hence, we know that $\mathcal{B}$ produces the correct answer with probability at least $\varepsilon(k)/q_2$. □

Lemma 3. *Suppose that H_1 is a random oracle and that there exists a Type II IND-CBE-CPA adversary $\mathcal{A}_2$ against BasicCBE with advantage $\varepsilon(k)$ which makes at most q_1 distinct queries to H_1. Then there exists an IND-CPA adversary $\mathcal{B}$ against BasicPub-II with advantage at least $\varepsilon(k)/q_1$.*

Proof. We show how to make use of the adversary $\mathcal{A}_2$ to construct an IND-CPA adversary $\mathcal{B}$ against BasicPub-II. The challenger $\mathcal{C}$ starts an IND-CPA Game by running algorithm Keygen of BasicPub-II to generate a public key $K_{pub} = \{q, G_1, G_2, G_T, \varphi, e, n, P_1, P_2, g, s, P_{pub}, PK_{ch}, h_0, H_2\}$ and the corresponding private key $K_{pri} = x_{ch}$, and passing K_{pub} to $\mathcal{B}$. Now, $\mathcal{B}$ interacts with $\mathcal{A}_2$ as follows:

Setup: $\mathcal{B}$ chooses an index I with $1 \leq I \leq q_1$. Then $\mathcal{B}$ gives $\mathcal{A}_2$ the BasicCBE public parameters *params* = $\{q, G_1, G_2, G_T, \varphi, e, n, P_1, P_2, g, P_{pub}, H_1, H_2\}$ where H_1 is a random oracle controlled by $\mathcal{B}$ as described below, s as the certifier's master key, and PK_{ch} as the challenge public key.

***H_1*-queries:** At any time $\mathcal{A}_2$ can query the random oracle H_1. To response to these queries $\mathcal{B}$ maintains a list of tuples $<(\tau||id||PK_{ch})_i, h_i, Cert_i>$ indexed by $(\tau||id||PK_{ch})_i$ as explained below. We refer to this list as the H_1^{list}. The list is initially empty. When $\mathcal{A}_2$ queries the oracle H_1 at a point $(\tau||id||PK_{ch})_i$, $\mathcal{B}$ responds as follows:

1. If the query $(\tau||id||PK_{ch})_i$ already appears in the H_1^{list} in a tuple $<(\tau||id||PK_{ch})_i, h_i>$, then $\mathcal{B}$ responds with $H_1((\tau||id||PK_{ch})_i) = h_i$.
2. Otherwise, if the query is on the *Ith* distinct $\tau||id||PK_{ch}$, then $\mathcal{B}$ returns $H_1((\tau||id||PK_{ch})_I) = h_0$ and adds the tuple $<(\tau||id||PK_{ch})_I, h_0>$ into the H_1^{list}.
3. Otherwise, $\mathcal{B}$ selects a random integer $h_i \in Z_q^*$ and adds the tuple $<(\tau||id||PK_{ch})_i, h_i>$ into the H_1^{list}. $\mathcal{B}$ responds with $H_1((\tau||id||PK_{ch})_i) = h_i$.

Challenge: At some point, $\mathcal{A}_2$ outputs $<\tau_{ch}, id_{ch}, M_0, M_1>$ on which it wants to be challenged. $\mathcal{B}$ responds as follows:

1. If the *I*th query on H_1 has not been issued, $\mathcal{B}$ inserts the tuple $<(\tau_{ch}||id_{ch}||PK_{ch}), h_0>$ into H_1^{list}.
2. Otherwise, if $(\tau_{ch}||id_{ch}||PK_{ch}) \neq (\tau||id||PK_{ch})_I$, then $\mathcal{B}$ aborts the game (**Event 1**).
3. Otherwise, $\mathcal{B}$ forwards $<M_0, M_1>$ to $\mathcal{C}$ as its challenge. $\mathcal{C}$ chooses a random bit $b \in \{0,1\}$, encrypts M_b and responds with the ciphertext C_{ch}. Then $\mathcal{B}$ forwards C_{ch} to $\mathcal{A}_2$.

Guess: Finally, $\mathcal{A}_2$ outputs its guess b' for b, $\mathcal{B}$ outputs the same b' as its own guess.

Obviously, if $\mathcal{B}$ does not abort during the simulation then $\mathcal{A}_2$'s view is identical to its view in the real attack. As shown above, $\mathcal{B}$ could abort when the following event happens: **Event 1**, denoted by $\mathcal{H}$: $\mathcal{A}_2$ wants to be challenged on $<\tau_{ch}, id_{ch}, PK_{ch}>$ and $(\tau_{ch}||id_{ch}||PK_{ch}) \neq (\tau||id||PK_{ch})_I$. Hence, the probability that $\mathcal{B}$ does not abort during the simulation is $\Pr[\neg\mathcal{H}] = 1/q_1$. This shows that $\mathcal{B}$'s advantage is at least $\varepsilon(k)/q_1$. □

Lemma 4. *Suppose that H_2 is a random oracle and that there exists an IND-CPA adversary $\mathcal{A}$ against BasicPub-II with advantage $\varepsilon(k)$ which makes at most q_2 distinct queries to H_2. Then there exists an algorithm $\mathcal{B}$ to solve the 1-BDHI problem with advantage at least $\varepsilon(k)/q_2$.*

Proof. Algorithm $\mathcal{B}$ is given as input a random 1-BDHI instance $<q, G_1, G_2, G_T, \varphi, e, P_1, P_2, xP_2>$ where x is a random element from Z_q^*. $\mathcal{B}$ finds $e(P_1, P_2)^{1/x}$ by interacting with $\mathcal{A}$ as follows:

Setup: Algorithm $\mathcal{B}$ first simulates algorithm Keygen of BasicPub-II to create the public key as below.

1. Set $Q_2 = xP_2$, $Q_1 = \varphi(Q_2) = xP_1$ and $g = e(Q_1, Q_2)$.

2. Pick a random $s \in Z_q^*$ and compute $P_{pub} = sQ_1$. Randomly choose an element $h_0 \in Z_q^*$.

3. Pick a random $y \in Z_q^*$ and compute $PK = e(P_1, P_2)^y = e(Q_1, Q_2)^{\frac{y}{x^2}}$.

Now, $\mathcal{B}$ passes $\mathcal{A}$ the public key $K_{pub} = \{q, G_1, G_2, G_T, \varphi, e, n, Q_1, Q_2, g, s, P_{pub}, PK, h_0, H_2\}$ where H_2 is a random oracle controlled by $\mathcal{B}$. The corresponding private key is $K_{pri} = \frac{y}{x^2}$ which $\mathcal{B}$ does not know. Note that $e(h_0 Q_1 + P_{pub}, \frac{1}{h_0 + s} Q_2) = e(Q_1, Q_2)$. Hence K_{pub} is a valid public key of BasicPub-II.

***H_2*-queries:** At any time $\mathcal{A}$ can query the random oracle H_2. To response to these queries $\mathcal{B}$ maintains a list of tuples $<\gamma_1, \gamma_2, \zeta>$. We refer to this list as H_2^{list}. The list is initially empty. When $\mathcal{A}$ queries the oracle H_2 at a point $<\gamma_1, \gamma_2>$, $\mathcal{B}$ responds as follows:

1. If the query $<\gamma_1, \gamma_2>$ already appears in the H_2^{list} in a tuple $<\gamma_1, \gamma_2, \zeta>$, then $\mathcal{B}$ responds with $H_2(\gamma_1, \gamma_2) = \zeta$.

2. Otherwise, $\mathcal{B}$ chooses a random $\zeta \in \{0,1\}^n$, returns $H_2(\gamma_1, \gamma_2) = \zeta$ and inserts a new tuple $<\gamma_1, \gamma_2, \zeta>$ to H_2^{list}.

Challenge: Adversary $\mathcal{A}$ outputs two messages M_0 and M_1 on which it wants to be challenged. $\mathcal{B}$ chooses a random $R \in \{0,1\}^n$ and a random $r \in Z_q^*$, and then defines the challenge ciphertext to be $C_{ch} = <U, V> = < r(h_0 P_1 + sP_1), R > = < \frac{r}{x}(h_0 Q_1 + P_{pub}), R >$. Observe that the decryption of the ciphertext C_{ch} is $V \oplus H_2(e(U, \frac{1}{h_0 + s} P_2), e(Q_1, Q_2)^{\frac{y}{x^2} \cdot \frac{r}{x}})$.

Guess: After $\mathcal{A}$ outputs its guess, $\mathcal{B}$ picks a random tuple $<\gamma_1, \gamma_2, \zeta>$ from H_2^{list} and returns $\gamma_2^{\frac{1}{yr}}$. Note that $\gamma_2^{\frac{1}{yr}} = e(P_1, P_2)^{\frac{1}{x}}$ if $\gamma_2 = e(Q_1, Q_2)^{\frac{y}{x^2} \cdot \frac{r}{x}}$.

Let $\mathcal{H}$ be the event that $\mathcal{A}$ issues a query for $H_2(*, e(Q_1, Q_2)^{\frac{y}{x^2} \cdot \frac{r}{x}})$ at some point during the simulation above. Using the same methods in Lemma 2, we can prove that $\Pr[\mathcal{H}] \geq \varepsilon(k)$. Assume that $\mathcal{A}$ has issued q_2 distinct queries to H_2. Hence, we know that $\mathcal{B}$ produces the correct answer with probability at least $\varepsilon(k)/q_2$. □

By combining the above four lemmas, we have the following theorem directly:

Theorem 1. *If hash functions H_1 and H_2 are modeled as random oracles, then BasicCBE is IND-CBE-CPA secure under the p-BDHI assumption.*

The scheme FullCBE is modified from the scheme BasicCBE by using a security enhancing transformation introduced in [11]. It has been proved that this transformation can generically upgrade any IND-CBE-CPA secure CBE scheme into an IND-CBE-CCA secure one in the random oracle model. Therefore, we have

Theorem 2. *If hash function H_3 is modeled as a random oracle and BasicCBE is IND-CBE-CPA secure, then FullCBE is IND-CBE-CCA secure.*

VI. Performance Comparison

In this section, we will make a computation performance comparison of the scheme FullCBE and other existing CBE schemes [1,7,8]. Here, we consider four major operations in all the CBE schemes, i.e., pairing (*p*), multi-exponentiation (*m*), exponentiation (*e*) and hash (*h*). Among these operations, the pairing operation is considered as the heaviest time-consuming one in spite of the recent advances in the implementation technique [26]. The ciphertext size is expressed by the length *n* for the plaintext, the length k_0 for the random string used in the encryption algorithm, and the length *l* for an element in G_1(or G_2), while *com* and *tag* refer to the size of the commitment string for the commitment scheme and the message authentication code used in the scheme proposed by Morillo and Ràfols [8], *vk* and *s* refer to the size of the verification key and the signature for the one-time signature scheme used in the scheme presented by Galindo, Morillo and Ràfols [7].

Considering the pre-computation, the detailed computation performance of all the CBE schemes is listed in Table I. We can see that our CBE scheme has better computation performance than other CBE schemes, especially in the encryption algorithm. Although, Scheme 2 and Scheme 3 do not require computing any pairings in the encryption algorithm too, they require additional building blocks to guarantee the security of the resulting schemes. Compared to Scheme 1, our scheme requires one more multiplication operation and exponentiation operation in the encryption algorithm. However, our scheme is still more efficient because two pairing operations in the encryption algorithm of Scheme 1 are much more time-consuming. Moreover, Scheme 1 and Scheme 3 both require the maptopoint operation [23] to map a string to an element in G_1 (or G_2) which is inefficient and slower than the general hash function used in our scheme which maps a string to an element in Z_q^*.

TABLE I. Performance Comparing of the CBE schemes

Scheme	Encryption	Decryption	Ciphertext size
Scheme 1 [1]	2*p*+1*m*+1*e*+4*h*	1*p*+1*m*+3*h*	$n+k_0+l$
Scheme 2 [8]	3*m*+2*e*+2*h*+*S* +*MAC*	3*p*+2m+2*h*+*R* +*MAC*	n+3l+*com*+*tag*
Scheme 3 [7]	3*m*+2*e*+3*h*+*sig*	3*p*+2m+1*h*+*vfy*	n+3l+ *vk*+*s*
Our scheme	2*m*+2*e*+4*h*	1*p*+1*m*+1*e*+3*h*	$n+k_0+l$

VII. Conclusion

In this paper, we present an efficient pairing-based CBE scheme and prove it to be secure in the sense of IND-CBE-CCA in the random oracle model based on the hardness of the *p*-BDHI problem. Regarding the computation performance, our scheme does not require computing any pairings in the encryption algorithm and need to compute only one pairing in the decryption algorithm. When compared with other existing CBE schemes, our scheme has obvious advantage in the computation performance.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (No.60673070) and the National High Technology Research and Development Program of China (No. 2007AA01Z409).

References

[1] C. Gentry, "Certificate-based encryption and the certificate revocation problem," In *Advances in Cryptology - EUROCRYPT'03*, Ploand, LNCS 2656, pp. 272-293, 2003.

[2] S. S. Al-Riyami and K.G. Paterson, "Certificateless public key cryptography," In *Advances in Cryptology-ASIACRYPT 2003*, Taiwan, LNCS 2894, pp. 452–473, 2003.

[3] S. S. Al-Riyami and K.G. Paterson, "CBE from CL-PKE: a generic construction and efficient schemes," In *Public Key Cryptography – PKC'05*, Switzerland, LNCS 3386, pp. 398-415, 2005.

[4] D.H. Yum and P.J. Lee, "Identity-based cryptography in public key management," In *EuroPKI 2004*, Greece, LNCS 3093, pp. 71-84, 2004.

[5] D.H. Yum and P.J. Lee, "Generic construction of certificateless encryption," In *ICCSE'04*, Italy, LNCS 3040, pp. 802-811, 2004.

[6] D. Galindo, P. Morillo, and C. Ràfols, "Breaking Yum and Lee generic constructions of certificateless and certificate-based encryption schemes," In *EuroPKI 2006*, Italy, LNCS 4043, pp. 81-91, 2006.

[7] D. Galindo, P. Morillo, and C. Ràfols, "Improved certificate-based encryption in the standard model," *Journal of Systems and Software*, vol. 81(7), pp. 1218-1226, 2008.

[8] P. Morillo and C. Ràfols, "Certificate-based encryption without random oracles," Cryptology ePrint Archive, Report 2006/12.

[9] Y. Lu, J. Li, and J. Xiao, "Applying the Fujisaki-Okamoto conversion to certificate-based encryption," In *2008 International Symposium on Electronic Commerce and Security*, China, pp. 296-300, 2008.

[10] Y. Lu and J. Li, "A general and secure certificate-based encryption construction," In *3rd ChinaGrid Annual Conference*, China, pp. 182-189, 2008.

[11] Y. Lu, J. Li, and J. Xiao, "Generic construction of certificate-based encryption," In *9th International Conference for Young Computer Scientists*, China, 2008, in press.

[12] B.G. Kang and J.H. Park, "Is it possible to have CBE from CL-PKE?" Cryptology ePrint Archive, Report 2005/431.

[13] B.G. Kang, J.H. Park, and S.G. Hahn, "A certificate-based signature scheme," In *CT-RSA 2004*, USA, LNCS 2964, pp. 99-111, 2004.

[14] Y. Dodis and J. Katz, "Chosen-ciphertext security of multiple encryption," In *TCC 2005*, USA, LNCS 3378, pp. 188-209, 2005.

[15] L. Wang, J. Shao, Z. Cao, M. Mambo, and A. Yamamura, "A certificate-based proxy cryptosystem with revocable proxy decryption power," In *Indocrypt 2007*, India, LNCS 4859, pp. 297-311, 2007.

[16] J. Li, X. Huang, Y. Mu, W. Susilo, and Q. Wu, "Certificate-based signature: security model and efficient construction," In *EuroPKI 2007*, Spain, LNCS 4582, pp. 110-125, 2007.

[17] M.H. Au, J.K Liu, W. Susilo, and T.H. Yuen, "Certificate based (linkable) ring signature," In *ISPEC 2007*, HongKong, China, LNCS 4464, pp.79-92, 2007.

[18] M. Bellare and P. Rogaway, "Random oracles are practical: a paradigm for designing efficient protocols," In *1st ACM Conference on Communications and Computer Security*, USA, pp. 62-73, 1993.

[19] R. Canetti, O. Goldreich, and S. Halevi, "The random oracle methodology, revisited," In *STOC'98*, USA, pp. 209-218, 1998.

[20] R. Sakai and M. Kasahara, "ID based cryptosystems with pairing on elliptic curve," Cryptology ePrint Archive, Report 2003/054.

[21] L.Q. Chen and Z.H. Cheng, "Security proof of Sakai-Kasahara's identity-based encryption scheme," Cryptology ePrint Archive, Report 2005/226.

[22] T.E. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," In *Advances in Cryptology- CRYPTO'84*, USA, LNCS 196, pp.10–18, 1985.

[23] D. Boneh. and M. Franklin, "Identity-based encryption from the Weil pairing," In *Advances in Crptology - CRYPTO'01*, USA, LNCS 2139, pp.213-229, 2001.

[24] D. Boneh and X. Boyen, "Efficient Selective-ID Secure Identity Based Encryption without Random Oracles," In *Advances in Cryptology - EUROCRYPT 2004*, Switzerland, LNCS 3027, pp.223–238, 2004.

[25] N. Smart and F. Vercauteren, "On computable isomorphisms in efficient pairing based systems," Cryptology ePrint Archive, Report 2005/116.

[26] P.S.L.M. Barreto, H.Y. Kim, B. Lynn, and M. Scott, "Efficient algorithms for pairing-based cryptosystems," In *Advances in Cryptology - CRYPTO 2002*, USA, LNCS 2442, pp.354-368, 2002.

[27] S. Goldwasser, S. Micali, and R. Rivest, "A digital signature scheme secure against adaptive chosen-message attack," *SIAM J. Computing*, vol.17(2), pp. 281–308, April, 1988.

[28] F. Zhang, R. Safavi-Naini, and W. Susilo, "An efficient signature scheme from bilinear pairings and its applications," In *Public Key Cryptography – PKC'04*, Singapore, LNCS 2947, pp.277–290, 2004.

[29] R. Housley, W. Polk, W. Ford, and D. Solo, "Internet X.509 Public Key Infrastructure certificate and certificate revocation list (CRL) profile," RFC 3280, IETF, 2002.

[30] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, "X.509 Internet Public Key Infrastructure online certificate status protocol – OCSP," RFC 2560, IETF, 1999.

[31] P. Guttman, "PKI: It's not dead, just resting," *IEEE Computer*, vol. 35, pp. 41–49, 2002.

[32] S. Micali, "Novomodo: scalable certificate validation and simplified PKI management," In *1st Annual PKI Research Workshop*, USA, pp. 15-25, 2002.

[33] S. Micali, "Efficient certificate revocation," Technical Report TM-542b, MIT Laboratory for Computer Science, 1996.

[34] M. Naor and K. Nissim, "Certificate revocation and certificate update," In *7th Annual USENIX Security Symposium*, USA, pp. 217-228, 1998.

[35] W. Aiello, S. Lodha, and R. Ostrovsky, "Fast digital identity revocation," In *Advances in Cryptology - CRYPTO 1998*, USA, LNCS 1462, pp. 137-152, 1998.