Strongly Secure Certificate-Based Encryption Scheme with Low Communication Bandwidth

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

Abstract: Certificate-based encryption (CBE) is a new asymmetric encryption paradigm which was introduced to solve the certificate management problem in traditional public-key encryption (PKE). It combines PKE and identity-based encryption (IBE) while preserving some of their most attractive features. CBE provides an efficient implicit certificate mechanism which eliminates the third-party queries and simplifies the certificate revocation problem in the traditional PKI. It also solves the key escrow problem and the key distribution problem inherent in IBE. In this paper, we first present a new security model for CBE, which defines the public key replacement attack and strengthens the power of adversaries against CBE. Our model is more elaborated and stronger compared with other existing ones. We then propose an efficient CBE scheme which is proved to be secure in the proposed security model under the random oracle model. When compared with other existing CBE schemes, our scheme enjoys better performance, especially on the communication bandwidth.

Keywords: certificate-based encryption; security model; public key replacement attack; random oracle model

1 Introduction

In Eurocrypt 2003, Gentry [1] introduced the notion of certificate-based encryption (CBE) to solve the certificate management problem in traditional public-key encryption (PKE). This new asymmetric encryption paradigm combines identity-based encryption (IBE) and PKE while preserving some of their most attractive features. As in the traditional PKI, each user in CBE generates his own public/private key pair and requests a certificate from a trusted third party, which is called as the certifier. The certifier generates a certificate as in a traditional PKI and is responsible for pushing a fresh certificate only to the holder of the public key periodically. A certificate in CBE has all the functionalities of a traditional PKI certificate, and also acts as a partial decryption key. This additional functionality provides an implicit certificate mechanism so that the sender is not required to obtain fresh information on certificate status and the receiver can only decrypt the ciphertext using his private key along with an up-to-date certificate from its certifier. The feature of implicit certificate allows us to eliminate third-party queries for the certificate status and to simplify the public key revocation problem so that CBE does not need infrastructures like CRL and OCSP. Therefore, CBE can be used to construct a more efficient PKI requiring fewer infrastructures. Furthermore, there is no key escrow problem (since the certifier does not know the private keys of users) and key distribution problem (since the certificates need

not be kept secret) in CBE.

In the original work [1], Gentry constructed the first CBE scheme in the random oracle [2] from the BF-IBE scheme [3]. A subsequent work by Yum and Lee [4] provided a formal equivalence theorem among IBE, certificateless public key encryption (CL-PKE) [5] and CBE, and showed that IBE implies both CBE and CL-PKE by giving a generic construction from IBE to those primitives. However, Galindo et al. [6] pointed out that a dishonest authority could break the security of their generic constructions. Actually, these generic constructions were inherently flawed due to a naive use of double encryption without further treatments. In [7], Lu et al. solved this problem by using the Fujisaki-Okamoto conversions [8, 9] and gave methods to achieve generic CCA-secure CBE constructions from PKE and IBE in the random oracle model. Moreover, Lu et al. [10] also proposed generic techniques to build CBE schemes from PKE and IBE in the standard model. In 2005, Al-Riyami and Paterson [11] gave an analysis of Gentry's CBE concept and repaired a number of problems in the original definition and security model for CBE. They also presented a generic conversion from CL-PKE to CBE 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. This implies that the derived CBE scheme by Al-Riyami and Paterson [11] is therefore invalid. In [13], Yum and Lee proposed a separable implicit certificate revocation system called status CBE to relieve the certifier's burden of certificate revocation, in which the authenticity of a public key is guaranteed by a long-lived certificate and the certificate revocation problem is resolved by a short-lived certificate. However, their status CBE scheme is insecure under the public key replacement attack [14]. In 2006, Morillo and Ràfols [15] proposed the first concrete CBE scheme in the standard model from the Waters-IBE scheme [16] and the BB-IBE scheme [17]. Subsequently, Galindo et al. [18] revised this CBE scheme and gave an improved scheme. In 2008, Liu and Zhou [19] proposed another CBE scheme in the standard model from the Gentry-IBE scheme [20]. Recently, Lu et al. [21] proposed a quite efficient CBE scheme in the random oracle model from the SK-IBE scheme [22, 23], which requires computing only one pairing in the encryption algorithm.

Our Contributions. The main contributions of this paper are twofold. We first present a new security model for CBE. As we know, a reasonable and elaborated security model is necessary for constructing provably secure cryptographic schemes. In [11], Al-Riyami and Paterson refined the original security model of CBE and proposed a revised one, but their model is still not elaborated and strong enough. Inspired by the recent improvements in the definition of security notions for certificate-based signature [24], we repair those problems still in the security model by Al-Riyami and Paterson, and strengthen the power of the adversaries against CBE, then propose a more elaborated and stronger security model for CBE. Our definition will provide a systematic approach for constructing CBE schemes secure against the public key replacement attack and proving their security on a stronger security level. The second contribution is that we construct an efficient CBE scheme with low communication bandwidth and prove it to be secure against adaptive chosen-ciphertext attack in the proposed security model. The proposed CBE scheme requires computing only one bilinear pairing in the decryption algorithm and introduces no redundancies in ciphertexts. When compared with other existing ones, our CBE scheme enjoys better performances both on the computation efficiency and the communication bandwidth, while its security is proved in a stronger security model.

2 Preliminaries

In this section, we review some basic concepts, including bilinear map, underlying hard problems and chosen-ciphertext secure symmetric encryption.

2.1 Bilinear Map and Underlying Hard Problems

Throughout the paper, G_1 denotes an additive cyclic group of prime order q and G_2 denotes a multiplicative cyclic group of the same order. For a group G, we use G^* to denote the set $G\backslash\{O\}$ where O is the identity element in the group G. We let P denote a generator of G_1. For us, a bilinear paring is a map e: $G_1 \times G_1 \rightarrow G_2$ with following properties:

– Bilinearity: $\forall\ P_1, P_2 \in G_1, \forall\ a, b \in Z_q^*$, we have $e(aP_1, bP_2) = e(P_1, P_2)^{ab}$.

– Non-degeneracy: $e(P, P) \neq 1$.

– Computability: $\forall\ P_1, P_2 \in G_1$, $e(P_1, P_2)$ can be efficiently computed.

The security of the CBE scheme in this paper is based on the hardness of the computational Diffie-Hellman (CDH) problem and the gap bilinear Diffie-Hellman (Gap-BDH) problem.

Definition 1. The CDH problem in G_1 is, given a tuple (P, aP, bP) where $a, b \in Z_q^*$, to compute $abP \in G_1$.

Let $\mathcal{B}$ be an adversary against the hardness of the CDH problem. $\mathcal{B}$'s advantage to solve the CDH problem is defined to be $Adv(\mathcal{B}) = \Pr[\mathcal{B}(P, aP, bP) = abP]$.

Definition 2 [25]. The Gap-BDH problem in (G_1, G_2) is, given a tuple (P, aP, bP, cP) where $a, b, c \in Z_q^*$, to compute $e(P, P)^{abc}$ with the help of a DBDH oracle $\mathcal{O}_{DBDH}(\cdot)$ that takes (P, aP, bP, cP, T) as input and outputs 1 if $T = e(P, P)^{abc}$ and 0 otherwise.

Let $\mathcal{B}$ be an adversary against the hardness of the Gap-BDH problem. $\mathcal{B}$'s advantage to solve the Gap-BDH problem is defined to be $Adv(\mathcal{B}) = \Pr[\mathcal{B}(P, aP, bP, cP, \mathcal{O}_{DBDH}(\cdot)) = e(P, P)^{abc}]$.

2.2 Chosen-Ciphertext Secure Symmetric Encryption

A symmetric encryption scheme is specified by an encryption algorithm E and a decryption algorithm D. Let λ be a security parameter. Algorithm E takes a symmetric key $K \in \{0,1\}^\lambda$ and a message M as input, and returns a ciphertext $C = E(K, M)$. Algorithm D takes $K \in \{0,1\}^\lambda$ and a ciphertext C as input, and returns either a message $M = D(K, C)$ or a special symbol $\perp$ denoting a decryption failure. Here, we restrict ourselves to the deterministic algorithms E and D.

Next, we briefly review the notion of chosen-ciphertext security for a symmetric encryption scheme described in [26].

Definition 3. A symmetric encryption scheme (E, D) is said to be IND-CCA secure if no PPT adversary $\mathcal{A}$ has non-negligible advantage in the following game:

Setup: The challenger randomly chooses a key $K \in \{0,1\}^\lambda$.

Phase 1: $\mathcal{A}$ issues a series of queries to the encryption oracle and the decryption oracle. The challenger responds these queries by using the key K.

Challenge: $\mathcal{A}$ outputs two plaintexts M_0 and M_1 which were not submitted to the encryption oracle or obtained from the decryption oracle. The challenger chooses a random bit $\beta \in \{0,1\}$ and encrypts M_β under the key K, then outputs the resulting ciphertext C^* to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ issues more queries as in Phase 1, but with the restrictions that C^* can not be submitted to the decryption oracle and M_0, M_1 can not be submitted to the encryption oracle.

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

3 Certificate-Based Encryption

In this section, we will first review the definition of CBE schemes. Then, we will redefine a new security model for CBE.

3.1 Definition of CBE

The following definition of CBE is essentially modified from [11], where the original definition given in [1] was reconsidered.

Definition 4. A certificate-base encryption (CBE) scheme consists of five algorithms:

- **Setup** is a probabilistic algorithm run by a certifier that takes as input a security parameter k to output a master key msk and a list of public parameters $params$ that include the descriptions of a finite plaintext space $MSPC$ and a finite ciphertext space $CSPC$. We consider $params$ to be an implicit input to the rest of the algorithms.
- **UserKeyGen** is a probabilistic algorithm run by a user that takes as input an identity id to output a public/private key pair (PK_{id}, SK_{id}).
- **Certify** is a deterministic or probabilistic algorithm run by a certifier that takes as input a master key msk, an index τ of the current time period, an identity id and a public key PK_{id}. It outputs a certificate $Cert_{id,\tau}$ which is sent to the user id through an open channel.
- **Encrypt** is a probabilistic algorithm that takes as input an index τ of the current time period, an identity id, a public key PK_{id} and a plaintext $M \in MSPC$ to output a ciphertext $C \in CSPC$.
- **Decrypt** is a deterministic algorithm that takes as input a private key SK_{id}, a certificate $Cert_{id,\tau}$ and a ciphertext C to output either a message $M \in MSPC$ or a special symbol $\perp$ indicating a decryption failure.

3.2 Security Model

In [1] and [11], the security models of CBE schemes are both defined by two different types of adversary: Type I and Type II. A Type I adversary models a malicious third party (i.e. anyone except the legitimate receiver or the certifier) who dose not have access to the master key, while a Type II adversary models an honest-but-curious certifier who has access to the master key. Although Al-Riyami and Paterson [11] have revised Gentry's security model, their model is still not elaborated enough. For example, the Type I adversary in their model is defined to have access to the target user's public/private key pair, but is not allowed to obtain the target user's certificate. However, in practice such an adversary may attack a CBE scheme by the target user's public key and certificate without knowing the corresponding private key. Moreover, this type adversary is required providing a private key along with the corresponding public key in all of decryption oracle queries. It enables the challenger to handle these decryption queries. This restriction is unnecessary and also restricts the ability of the adversary. Actually, the challenger can handle decryption queries using special purpose knowledge extractors without requiring the adversary to provide the private key. For another example, their security model dose not capture the public key replacement attack. It seems that the public key replacement attack does not exist in CBE due to the use of certificates. However, in CBE, only the owner needs to check the validity of its certificate and other users do not need. Therefore, such attack actually does exist. A concrete example is the status CBE scheme proposed by Yum and Lee [13]. In [14], this scheme is pointed out to be insecure under the public key replacement attack, although it was proved to be secure

without considering such attack.

In the following, we define a more elaborated and stronger security model for CBE. Our model also distinguishes two types of adversary. Below, we first define the oracles that an adversary against CBE may query and how each query should be responded by a challenger.

- **Public Key Request:** On input an identity *id*, the challenger responds with the public key PK_{id} for *id*. If the identity *id* has no associated public key, then the challenger generates a public key PK_{id} for *id* by running **UserKeyGen**.
- **Public Key Replacement:** In our security model, an adversary can repeatedly replace the public key of any entity with a key of its choice. On input an identity *id* and a key PK'_{id}, the challenger replaces the current public key PK_{id} for *id* with PK'_{id}. Note that the current value of an entity's public key is used by the challenger in any computations or responses to the adversary's requests. This oracle models the adversary's ability to convince a legitimate user to use an invalid public key, and then enables our security model to capture the public key replacement attack.
- **Certificate Request:** On input an index τ of a time period and an identity *id*, the challenger responds with the certificate $Cert_{id,\tau}$. If the identity *id* has no associated certificate in the time period τ, then the challenger generates a certificate $Cert_{id,\tau}$ by running **Certify** (after running **UserKeyGen** if necessary).
- **Private Key Extraction:** On input an identity *id*, the challenger responds with the private key SK_{id}. If the identity *id* has no associated private key, then the challenger generates a private key SK_{id} by running **UserKeyGen**. However, it is unreasonable to expect the challenger to respond to such a query if the public key for *id* has already been replaced.
- **Decryption:** On input an index τ of a time period, an identity *id*, and a ciphertext *C*, the challenger responds with the correct decryption of *C*, even if the public key for *id* has been replaced. This is a rather strong property for the security model of CBE. After all, the challenger may no longer know the correct corresponding private key. However, this capability may give the adversary more power in breaking the scheme. For further discussion of this feature (but in CL-PKE setting), see [5]. Note that the security models in [1, 11] only allow the adversaries to query a weak decryption oracle which additionally requires taking the public key and the private key for *id* as input.

Next, we provide a detailed description of two adversary types in our security model.

Type I Adversary: Such an adversary $\mathcal{A}_I$ dose not have access to the master key, but is allowed to query the following oracles: **Public Key Request**, **Certificate Request**, **Private Key Extraction**, **Public Key Replacement** and **Decryption**. The restrictions on this type of adversary are that:

- $\mathcal{A}_I$ can not request both the private key for the challenge identity id^* and the certificate for the challenge identity id^* in the challenge time period τ^* at any point.
- $\mathcal{A}_I$ can not extract the private key for any identity if the corresponding public key has been replaced.
- $\mathcal{A}_I$ can not request the certificate for the challenge identity id^* in the challenge time period τ^* if the public key for id^* has been replaced before the challenge was issued.
- $\mathcal{A}_I$ can not make a decryption query on the challenge ciphertext C^* for the combination (id^*, τ^*).

Type II Adversary: Such an adversary $\mathcal{A}_{II}$ has access to the master key, and is also allowed to query the following oracles: **Public Key Request**, **Private Key Extraction**, **Public Key Replacement** and **Decryption**. The restrictions on this type of adversary are that:

– $\mathcal{A}_{II}$ cannot extract the private key for the challenge identity id^* at any point.
– $\mathcal{A}_{II}$ cannot extract the private key for any identity if the corresponding public key has been replaced.
– $\mathcal{A}_{II}$ cannot output a challenge identity id^* for which it has replaced the public key.
– $\mathcal{A}_{II}$ cannot make a decryption query on the challenge ciphertext C^* for the combination (id^*, τ^*).

Definition 5. A CBE scheme is said to be IND-CBE-CCA2 secure if no polynomially bounded adversary $\mathcal{A}$ of Type I or Type II has a non-negligible advantage in the following game:

Setup: The challenger $\mathcal{C}$ runs $Setup(1^k)$ to generate a master key msk and a list of public system parameters $params$, then outputs $params$ to $\mathcal{A}$. If $\mathcal{A}$ is of Type II, $\mathcal{C}$ also outputs msk to $\mathcal{A}$. Otherwise, it keeps msk to himself.

Phase 1: In this phase, $\mathcal{A}$ may have access to certain oracles defined above. These oracle queries are asked adaptively, but are subject to the restrictions described above.

Challenge: Once $\mathcal{A}$ decides that Phase 1 is over, it outputs an index τ^* of a time period, an identity id^* and two equal length plaintexts M_0, M_1, on which it wants to be challenged. Again, the restrictions described above apply. $\mathcal{C}$ randomly chooses a bit $b \in \{0, 1\}$, computes the challenge ciphertext $C^* = \textbf{Encrypt}(params, \tau^*, id^*, PK_{id^*}, M_b)$, and then outputs C^* to $\mathcal{A}$.

Phase 2: As in Phase 1, but with the restrictions described above.

Guess: Finally, $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ and wins the game if $b = b'$. We define the advantage of the adversary $\mathcal{A}$ in this game to be $Adv(\mathcal{A}) := 2|\Pr[b = b'] - \frac{1}{2}|$.

4 Our CBE Scheme

In this section, we propose a new CBE scheme and prove it to be IND-CBE-CCA2 secure under the hardness of the CDH problem and Gap-BDH problem in our proposed security model. We also make a detailed comparison of our scheme and other existing CBE schemes on the computation efficiency and the communication bandwidth.

4.1 Concrete Construction

Our scheme is constructed based on a hybrid variant of the BF-IBE scheme proposed by Libert and Quisquater [26]. It is described as follows:

Setup: On input a security parameter k, this algorithm performs as follows:

(1) Generate a pair of groups (G_1, G_2) of prime order q and a bilinear map e: $G_1 \times G_1 \rightarrow G_2$.

(2) Choose a generator $P \in G_1$, randomly select $s \in Z_q^*$ and compute $P_{pub} = sP$.

(3) Choose two cryptographic hash functions H_1: $\{0,1\}^* \rightarrow G_1^*$ and H_2: $(G_1)^3 \times G_2 \rightarrow \{0,1\}^\lambda$, as well as an IND-CCA secure symmetric encryption scheme (E, D) of keylength λ, where λ is polynomial in k.

(4) Output $msk = s$ as the master key and $params = \{G_1, G_2, e, q, P, P_{pub}, H_1, H_2, \lambda, E, D, n\}$ as the public parameters, where n denotes a bound on the size of plaintexts. The plaintext space is $MSPC = \{0,1\}^n$ and the ciphertext space is $CSPC = G_1 \times \{0,1\}^n$.

UserKeyGen: On input <$params$, id>, this algorithm randomly chooses $x \in Z_q^*$ and outputs (PK_{id}, SK_{id}) = (xP, x) as the public/private key pair for the identity id.

Certify: On input <$params$, msk, τ, id, PK_{id}>, this algorithm computes $Q_{id} = H_1(\tau, id, PK_{id})$ and outputs $Cert_{id,\tau} = sQ_{id}$ as the certificate for the identity id in the time period τ.

Encrypt: On input <$params$, τ, id, PK_{id}, M>, this algorithm performs as follows:

(1) Randomly choose $r \in Z_q^*$ and compute $U = rP$.

(2) Compute $Q_{id} = H_1(\tau, id, PK_{id})$ and $SK_{sym} = H_2(Q_{id}, U, rPK_{id}, e(P_{pub}, Q_{id})^r)$.

(3) Compute $V = E(SK_{sym}, M)$ and output $C = (U, V)$ as the ciphertext.

Note that encryption needs to compute one pairing for a receiver in each time period. However, once $e(P_{pub}, Q_{id})$ has been pre-computed, it does not require any pairing computations to send messages for the identity *id* in the whole time period τ.

Decrypt: On input $<params, Cert_{id,\tau}, SK_{id}, C = (U, V)>$, this algorithm computes $SK_{sym} = H_2(Q_{id}, U, SK_{id} \cdot U, e(U, Cert_{id,\tau}))$ and outputs $M = D(SK_{sym}, V)$.

The consistency of our scheme is easy to check as we have

$$\begin{aligned} SK_{sym} &= H_2(Q_{id}, U, SK_{id} \cdot U, e(U, Cert_{id,\tau})) \\ &= H_2(Q_{id}, U, xrP, e(rP, sQ_{id})) \\ &= H_2(Q_{id}, U, rPK_{id}, e(P_{pub}, Q_{id})^r). \end{aligned}$$

4.2 Security Analysis

Theprem 1. Suppose that H_1, H_2 are random oracles and $\mathcal{A}_I$ is a Type I adversary against the IND-CBE-CCA2 security of our CBE scheme with advantage ε when running in time *t*, making q_{pub} public key requests, q_{pri} private key requests, q_{cer} certificate requests, q_{dec} decryption queries and q_i random oracle queries to H_i (i = 1,2). Then, for any $0 \le \nu \le \varepsilon$, there exists

– either an algorithm $\mathcal{B}$ to solve the Gap-BDH problem with advantage $\varepsilon' \ge \frac{\varepsilon}{q_1} - \nu$ and running in time $t' \le t + (q_1 + q_{pub} + q_{pri} + q_{cer})\tau_{mul} + q_2\phi + q_{dec}(\tau_{sym} + 2\tau_{pair})$, where τ_{mul} and τ_{pair} respectively denote the time for computing a multiplication in G_1 and the one for a pairing, while τ_{sym} and ϕ respectively denote the time for a symmetric decryption and the one for a call to the DBDH oracle.

– or an adversary that breaks the chosen-ciphertext security of the symmetric encryption scheme (*E*, *D*) with advantage ν within time *t*.

Proof. We show how to construct an algorithm $\mathcal{B}$ to solve the Gap-BDH problem from the Type I adversary $\mathcal{A}_I$. Assume that the algorithm $\mathcal{B}$ is given a random Gap-BDH instance (P, aP, bP, cP, $\mathcal{O}_{DBDH}$). $\mathcal{B}$ finds $e(P, P)^{abc}$ by interacting with $\mathcal{A}_I$ as follows:

Setup: $\mathcal{B}$ randomly chooses an index I with $1 \le I \le q_1$ and sets $P_{pub} = aP$. It then supplies $\mathcal{A}_I$ with $params = \{G_1, G_2, e, q, P, P_{pub}, H_1, H_2, \lambda, \mathcal{E}, \mathcal{D}, n\}$ as the public parameters, where H_1 and H_2 are random oracles controlled by $\mathcal{B}$. Note that the corresponding master key *msk* is *a* which is unknown to $\mathcal{B}$.

$\mathcal{A}_I$ may make queries to H_1 and H_2 at any time during its attack and $\mathcal{B}$ responds as follows:

***H_1* queries**: $\mathcal{B}$ maintains a list $\mathsf{H_1List}$ of tuples $< id_i, \tau_i, PK_{id_i}, \delta_i, Q_{id_i} >$ which is initially empty. On receiving such a query on $(id_i, \tau_i, PK_{id_i})$, $\mathcal{B}$ does the following:

– If $< id_i, \tau_i, PK_{id_i}, \delta_i, Q_{id_i} >$ exists in $\mathsf{H_1List}$, then $\mathcal{B}$ returns Q_{id_i}.

– Otherwise, if $i = I$, $\mathcal{B}$ adds $< id_i, \tau_i, PK_{id_i}, ?, cP >$ to $\mathsf{H_1List}$ and returns cP; else if $i \ne I$, $\mathcal{B}$ randomly chooses $\delta_i \in Z_q^*$, computes $Q_{id_i} = \delta_i P$, and then adds $< id_i, \tau_i, PK_{id_i}, \delta_i, Q_{id_i} >$ to $\mathsf{H_1List}$ and returns Q_{id_i}.

***H_2* queries**: $\mathcal{B}$ maintains three following lists which are initially empty.

– $\mathsf{H_2List\text{-}A}$ contains tuples $< Q_{id_i}, U_i, A_i, T_i, SK_i >$ for which $\mathcal{B}$ has assigned a value $SK_i \in \{0,1\}^{\lambda}$ to $H_2(Q_{id_i}, U_i, A_i, T_i)$.

– H_2List-B contains tuples $< Q_{id_i}, U_i, A_i, T_i >$ such that $< Q_{id_i}, U_i, A_i, T_i, SK_i >$ exists in H_2List-A for $SK_i \in \{0,1\}^{\lambda}$ and $\mathcal{O}_{DBDH}(P, P_{pub}, U_i, Q_{id_i}, T_i) = 1$.

– H_2List-C contains tuples $< Q_{id_i}, U_i, A_i, SK_i >$ for which $\mathcal{B}$ has implicitly assigned a value $SK_i \in \{0,1\}^{\lambda}$ to $H_2(Q_{id_i}, U_i, A_i, T_i)$ although the value T_i such that $\mathcal{O}_{DBDH}(P, P_{pub}, U_i, Q_{id_i}, T_i) = 1$ is unknown.

More concretely, when receiving such a query on $(Q_{id_i}, U_i, A_i, T_i)$, $\mathcal{B}$ does the following:

– If $< Q_{id_i}, U_i, A_i, T_i, SK_i >$ exists in H_2List-A for some value $SK_i \in \{0,1\}^{\lambda}$, $\mathcal{B}$ returns SK_i.

– Otherwise, $\mathcal{B}$ submits $(P, P_{pub}, U_i, Q_{id_i}, T_i)$ to the oracle $\mathcal{O}_{DBDH}(\cdot)$ to decide whether it is a valid BDH tuple.

- If it does, $\mathcal{B}$ adds $< Q_{id_i}, U_i, A_i, T_i >$ to H_2List-B and checks if $< Q_{id_i}, U_i, A_i, SK_i >$ exists in H_2List-C. If it does, $\mathcal{B}$ adds $< Q_{id_i}, U_i, A_i, T_i, SK_i >$ to H_2List-A and returns SK_i.
- Otherwise, $\mathcal{B}$ randomly chooses $SK_i \in \{0,1\}^{\lambda}$, adds $< Q_{id_i}, U_i, A_i, T_i, SK_i >$ to H_2List-A and returns SK_i.

Phase 1: $\mathcal{A}_I$ issues a series of oracle queries as defined in Definition 5, and $\mathcal{B}$ responds these queries as follows:

Public Key Request: $\mathcal{B}$ maintains a list KeyList of tuples $< id_i, PK_{id_i}, SK_{id_i}, flag_i >$ which is initially empty. On receiving such a query on id_i, $\mathcal{B}$ does the following:

– If $< id_i, PK_{id_i}, SK_{id_i}, flag_i >$ exists in KeyList, $\mathcal{B}$ returns PK_{id_i}.

– Otherwise, $\mathcal{B}$ randomly chooses $x_i \in Z_q^*$ as the private key for id_i and computes $x_i P$ as the corresponding public key. It then adds $< id_i, x_i P, x_i, 0 >$ to KeyList and returns $x_i P$.

Private Key Extraction: On receiving such a query on id_i, $\mathcal{B}$ does the following:

– If $< id_i, PK_{id_i}, SK_{id_i}, flag_i >$ exists in KeyList and $flag_i = 1$, namely that the public key for id_i has been replaced, then $\mathcal{B}$ rejects this query; else if $flag_i = 0$, $\mathcal{B}$ returns SK_{id_i}.

– Otherwise, $\mathcal{B}$ randomly chooses $x_i \in Z_q^*$, adds $< id_i, x_i P, x_i, 0 >$ to KeyList and returns x_i.

Public Key Replacement: On receiving such a query on (id_i, PK'_{id_i}), $\mathcal{B}$ searches id_i in the list KeyList and updates the resulting tuple $< id_i, PK_{id_i}, SK_{id_i}, flag_i >$ with $< id_i, PK'_{id_i}, SK_{id_i}, 1 >$.

Certificate Request: $\mathcal{B}$ maintains a list CertList of tuples $< id_i, \tau_i, Cert_{id_i, \tau_i} >$ which is initially empty. On receiving such a query on (id_i, τ_i), $\mathcal{B}$ does the following:

– If $i = I$, $\mathcal{B}$ aborts.

– Otherwise, $\mathcal{B}$ checks whether $< id_i, \tau_i, Cert_{id_i, \tau_i} >$ exists in CertList.

- If it does, $\mathcal{B}$ returns $Cert_{id_i, \tau_i}$.
- Otherwise, $\mathcal{B}$ searches H_1List for a tuple $< id_i, \tau_i, PK_{id_i}, \delta_i, Q_{id_i} >$ (after running the simulation algorithm for H_1 query if necessary) and computes $Cert_{id_i, \tau_i} = \delta_i P_{pub}$. It then adds $< id_i, \tau_i, Cert_{id_i, \tau_i} >$ to CertList and returns $Cert_{id_i, \tau_i}$.

Decryption: On receiving such a query on $(id_i, \tau_i, C = (U, V))$, $\mathcal{B}$ does the following:

– If $i \neq I$ and the public key for id_i has not been replaced, then $\mathcal{B}$ decrypts C in the normal way by using the corresponding private key SK_{id_i} and the certificate $Cert_{id_i, \tau_i}$.

– Otherwise, $\mathcal{B}$ checks if $< Q_{id_i}, U, A_i, T_i >$ exists in H_2List-B for some value $T_i \in G_2$ satisfying $e(U, PK_{id_i}) = e(A_i, P)$.

- If it does, $\mathcal{B}$ retrieves SK_i from the tuple $< Q_{id_i}, U, A_i, T_i, SK_i >$ in $\mathsf{H_2List\text{-}A}$ and returns $M = D(SK_i, V)$.
- Otherwise, $\mathcal{B}$ checks whether $\mathsf{H_2List\text{-}C}$ contains a tuple $< Q_{id_i}, U, A_i, SK_i >$ for some value $SK_i \in \{0,1\}^{\lambda}$ satisfying $e(U, PK_{id_i}) = e(A_i, P)$. If it does, $\mathcal{B}$ returns $M = D(SK_i, V)$; Otherwise, $\mathcal{B}$ randomly chooses $SK_i \in \{0,1\}^{\lambda}$, adds $< Q_{id_i}, U, T_i, SK_i >$ to $\mathsf{H_2List\text{-}C}$ and returns $M = D(SK_i, V)$.

Challenge Phase: $\mathcal{A}_I$ outputs id^*, τ^* and two messages M_0, M_1 on which it wants to be challenged. If $(id^*, \tau^*) \neq (id_I, \tau_I)$, $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ does the following:

– Set $U^* = bP$, retrieve SK_{id^*} from the tuple $<id^*, PK_{id^*}, SK_{id^*}, flag^*>$ in $\mathsf{KeyList}$, and compute $A^* = SK_{id^*} \cdot U^*$.

– Check whether $< Q_{id^*}, U^*, A^*, SK^*>$ exists in $\mathsf{H_2List\text{-}C}$ for some value $SK^* \in \{0,1\}^{\lambda}$ (satisfying $e(U^*, PK_{id^*}) = e(A^*, P)$ if the public key for id^* has been replaced).

- If it does, $\mathcal{B}$ computes $V^* = E(SK^*, M_\beta)$ for a random bit $\beta \in \{0,1\}$.
- Otherwise, $\mathcal{B}$ randomly chooses a value $SK^* \in \{0,1\}^{\lambda}$, adds $< Q_{id^*}, U^*, A^*, SK^*>$ to $\mathsf{H_2List\text{-}C}$, and computes $V^* = E(SK^*, M_\beta)$ for a random bit $\beta \in \{0,1\}$.

– Output $C^* = (U^*, V^*)$ as the challenge ciphertext.

Phase 2: As in Phase 1, with the restrictions specified in Definition 5.

Guess: Finally, $\mathcal{A}_I$ outputs a guess $\beta' \in \{0,1\}$ for β. $\mathcal{B}$ ignores the result and checks whether $< Q_{id^*}, U^*, A^*, T^*>$ exists in $\mathsf{H_2List\text{-}B}$. If it does, $\mathcal{B}$ outputs T^* as the solution to the Gap-BDH problem; otherwise, it loses the game. From the above simulation, it is easy to deduce that $T^* = e(P, P)^{abc}$ which is the right solution to the Gap-BDH problem.

Next, we estimate $\mathcal{B}$'s advantage in solving the Gap-BDH problem. Let $\mathsf{AskH_2}^*$ denote the event that $< Q_{id^*}, U^*, A^*, T^*>$ has been queried to the random oracle H_2, BreakV^* the event that V^* is broken as the challenge ciphertext in the chosen-ciphertext attack against the symmetric encryption scheme (E, D), and $\neg\mathsf{Abort}$ the event that $\mathcal{B}$ does not abort during the simulation.

Now, we define an event E to be $(\mathsf{AskH_2}^* \wedge \mathsf{BreakV}^*) | \neg\mathsf{Abort}$. It is clear that if E does not happen, then $\mathcal{A}_I$ does not gain any advantage greater than 1/2 to guess β. Namely, we have the probability $\Pr[\beta' = \beta | \neg\mathsf{E}] \leq 1/2$. Hence, by splitting $\Pr[\beta' = \beta]$, we obtain

$$\Pr[\beta' = \beta] = \Pr[\beta' = \beta | \neg\mathsf{E}] \cdot \Pr[\neg\mathsf{E}] + \Pr[\beta' = \beta | \mathsf{E}] \cdot \Pr[\mathsf{E}]$$

$$\leq \frac{1}{2} \cdot \Pr[\neg\mathsf{E}] + \Pr[\mathsf{E}] = \frac{1}{2} + \frac{1}{2} \Pr[\mathsf{E}].$$

By the definition of ε, we have

$$\varepsilon \leq 2 | \Pr[\beta' = \beta] - \frac{1}{2} | \leq \Pr[\mathsf{E}]$$

$$\leq \frac{1}{\Pr[\neg\mathsf{Abort}]} \cdot (\Pr[\mathsf{AskH_2}^*] + \Pr[\mathsf{BreakV}^*]).$$

Since $\Pr[\mathsf{BreakV}^*] \leq \nu$ and $\Pr[\neg\mathsf{Abort}] = \frac{1}{q_1}$, we obtain

$$\Pr[\mathsf{AskH_2}^*] \geq \frac{\varepsilon}{q_1} - \nu.$$

Hence, we have $\varepsilon' \geq \Pr[\mathsf{AskH_2}^*] \geq \frac{\varepsilon}{q_1} - \nu$.

This completes the proof of this theorem. □

Theorem 2. Suppose that H_1, H_2 are random oracles and $\mathcal{A}_{II}$ is a Type II adversary against

the IND-CBE-CCA2 security of our CBE scheme with advantage ε when running in time t, making q_{pub} public key requests, q_{pri} private key requests, q_{dec} strong decryption queries and q_i random oracle queries to H_i (i = 1,2). Then, for any $0 \le \nu \le \varepsilon$, there exists

– either an algorithm $\mathcal{B}$ to solve the CDH problem with advantage $\varepsilon' \ge \frac{\varepsilon}{q_1} - \nu$ and running in time $t' \le t + (q_{pub} + q_{pri})\tau_{mul} + 2q_2\tau_{pair} + q_{dec}(\tau_{sym} + \tau_{pair})$, where τ_{mul} and τ_{pair} respectively denote the time for computing a multiplication in G_1 and the one for a pairing, while τ_{sym} denotes the time for a symmetric decryption.

– or an adversary that breaks the chosen-ciphertext security of the symmetric encryption scheme (E, D) with advantage ν within time t.

Proof. We show how to construct an algorithm $\mathcal{B}$ to solve the CDH problem from the Type II adversary $\mathcal{A}_{II}$. Assume that $\mathcal{B}$ is given a random CDH instance (P, aP, bP). It finds abP by interacting with $\mathcal{A}_{II}$ as follows:

Setup: $\mathcal{B}$ first randomly chooses an index I with $1 \le I \le q_1$. It then randomly chooses $s \in Z_q^*$, computes $P_{pub} = sP$, and supplies $\mathcal{A}_{II}$ with the public parameters $params = \{G_1, G_2, e, q, P, P_{pub}, H_1, H_2, \lambda, E, D, n\}$ together with the master key s, where H_1 and H_2 are random oracles controlled by $\mathcal{B}$. $\mathcal{A}_{II}$ may make queries to H_1 and H_2 at any time during its attack and $\mathcal{B}$ responds as follows:

***H_1* queries**: $\mathcal{B}$ maintains a list $\mathsf{H_1List}$ of tuples $< id_i, \tau_i, PK_{id_i}, Q_{id_i} >$ which is initially empty. On receiving such a query on $(id_i, \tau_i, PK_{id_i})$, $\mathcal{B}$ does the following:

– If $< id_i, \tau_i, PK_{id_i}, Q_{id_i} >$ exists in $\mathsf{H_1List}$, $\mathcal{B}$ returns Q_{id_i}.

– Otherwise, $\mathcal{B}$ randomly chooses $Q_{id_i} \in G_1^*$, adds $< id_i, \tau_i, PK_{id_i}, Q_{id_i} >$ to $\mathsf{H_1List}$ and returns Q_{id_i}.

***H_2* queries**: $\mathcal{B}$ maintains three following lists which are initially empty.

– $\mathsf{H_2List\text{-}A}$ contains tuples $< Q_{id_i}, U_i, A_i, T_i, SK_i >$ for which $\mathcal{B}$ has assigned a value $SK_i \in \{0,1\}^\lambda$ to $H_2(Q_{id_i}, U_i, A_i, T_i)$.

– $\mathsf{H_2List\text{-}B}$ contains tuples $< Q_{id_i}, U_i, A_i, T_i >$ such that $< Q_{id_i}, U_i, A_i, T_i, SK_i >$ exists in $\mathsf{H_2List\text{-}A}$ for $SK_i \in \{0,1\}^\lambda$ and $e(U_i, PK_{id_i}) = e(A_i, P)$.

– $\mathsf{H_2List\text{-}C}$ contains tuples $< Q_{id_i}, U_i, T_i, SK_i >$ for which $\mathcal{B}$ has implicitly assigned a value $SK_i \in \{0,1\}^\lambda$ to $H_2(Q_{id_i}, U_i, A_i, T_i)$ although the value A_i such that $e(U_i, PK_{id_i}) = e(A_i, P)$ is unknown.

More concretely, when receiving such a query on $(Q_{id_i}, U_i, A_i, T_i)$, $\mathcal{B}$ does the following:

– If $< Q_{id_i}, U_i, A_i, T_i, SK_i >$ exists in $\mathsf{H_2List\text{-}A}$ for some value $SK_i \in \{0,1\}^\lambda$, $\mathcal{B}$ returns SK_i.

– Otherwise, $\mathcal{B}$ checks whether $e(U_i, PK_{id_i}) = e(A_i, P)$.

- If it does, $\mathcal{B}$ adds $< Q_{id_i}, U_i, A_i, T_i >$ to $\mathsf{H_2List\text{-}B}$ and checks whether $< Q_{id_i}, U_i, T_i, SK_i >$ exists in $\mathsf{H_2List\text{-}C}$. If it does, $\mathcal{B}$ adds $< Q_{id_i}, U_i, A_i, T_i, SK_i >$ to $\mathsf{H_2List\text{-}A}$ and returns SK_i.
- Otherwise, $\mathcal{B}$ randomly chooses $SK_i \in \{0,1\}^\lambda$, adds $< Q_{id_i}, U_i, A_i, T_i, SK_i >$ to $\mathsf{H_2List\text{-}A}$ and returns SK_i.

Phase 1: $\mathcal{A}_{II}$ issues a series of oracle queries as defined in Definition 5, and $\mathcal{B}$ responds these queries as follows:

Public Key Request: $\mathcal{B}$ maintains a list $\mathsf{KeyList}$ of tuples $< id_i, PK_{id_i}, SK_{id_i}, flag_i >$ which is initially empty. On receiving such a query on id_i, $\mathcal{B}$ does the following:

– If $< id_i, PK_{id_i}, SK_{id_i}, flag_i >$ exists in $\mathsf{KeyList}$, $\mathcal{B}$ returns PK_{id_i} to $\mathcal{A}_{II}$.

– Otherwise, if $i \neq I$, $\mathcal{B}$ randomly chooses $x_i \in Z_q^*$ as the private key for id_i and computes x_iP as the corresponding public key, then adds $<id_i, x_iP, x_i, 0>$ to KeyList and returns PK_{id_i}; else if $i = I$, $\mathcal{B}$ adds $< id_i, aP, ?, 0 >$ to KeyList and returns aP.

Private Key Extraction: On receiving such a query on id_i, $\mathcal{B}$ does the following:

– If $i = I$, $\mathcal{B}$ aborts.

– Otherwise, $\mathcal{B}$ checks whether $< id_i, PK_{id_i}, SK_{id_i}, flag_i >$ exists in KeyList.

- If it does, $\mathcal{B}$ rejects this query if $flag_i = 1$ or returns SK_{id_i} if $flag_i = 0$.
- Otherwise, $\mathcal{B}$ randomly chooses $x_i \in Z_q^*$, adds $< id_i, x_iP, x_i, 0 >$ in KeyList and returns x_i.

Public Key Replacement: On receiving such a query on $(id_i, PK_{id_i}^{'})$, $\mathcal{B}$ does the following:

– If $i = I$, $\mathcal{B}$ aborts.

– Otherwise, $\mathcal{B}$ searches id_i in KeyList and updates the resulting tuple $< id_i, PK_{id_i}, SK_{id_i}, flag_i >$ with $< id_i, PK_{id_i}^{'}, SK_{id_i}, 1 >$.

Decryption: On receiving such a query on $(id_i, \tau_i, C = (U, V))$, $\mathcal{B}$ does the following:

– If $i \neq I$ and the public key for id_i has not been replaced, then $\mathcal{B}$ decrypts C in the normal way by using the corresponding private key and the certificate $(SK_{id_i}, Cert_{id_i,\tau_i})$.

– Otherwise, $\mathcal{B}$ computes $T_i = e(U, Cert_{id_i,\tau_i})$ and checks whether $< Q_{id_i}, U, A_i, T_i >$ exists in H_2List-B for some value $A_i \in G_1$.

- If it does, $\mathcal{B}$ retrieves SK_i from the tuple $< Q_{id_i}, U, A_i, T_i, SK_i >$ in H_2List-A and returns $M = D(SK_i, V)$.
- Otherwise, $\mathcal{B}$ checks whether $< Q_{id_i}, U, A_i, SK_i >$ exists in H_2List-C for some value $SK_i \in \{0,1\}^{\lambda}$. If it does, $\mathcal{B}$ returns $M = D(SK_i, V)$. Otherwise, $\mathcal{B}$ randomly chooses $SK_i \in \{0,1\}^{\lambda}$, adds $< Q_{id_i}, U, T_i, SK_i >$ to H_2List-C and returns $M = D(SK_i, V)$.

Challenge Phase: $\mathcal{A}_{II}$ outputs id^*, τ^* and two messages M_0, M_1 on which it wants to be challenged. If $(id^*, \tau^*) \neq (id_I, \tau_I)$, $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ does the following:

– Set $U^* = bP$ and then check whether $< Q_{id^*}, U^*, T^*, SK^* >$ exists in H_2List-C for some value $SK^* \in \{0,1\}^{\lambda}$ satisfying $T^* = e(U^*, Cert_{id^*,\tau^*})$.

- If it does, $\mathcal{B}$ computes $V^* = E(SK^*, M_\beta)$ for a random bit $\beta \in \{0,1\}$.
- Otherwise, $\mathcal{B}$ randomly chooses $SK^* \in \{0,1\}^{\lambda}$, adds $< Q_{id^*}, U^*, T^*, SK^* >$ to H_2List-C, and then computes $V^* = E(SK^*, M_\beta)$ for a random bit $\beta \in \{0,1\}$.

– Output $C^* = (U^*, V^*)$ as the challenge ciphertext.

Phase 2: As in Phase 1, with the restrictions specified in Definition 5. Note that $\mathcal{A}_{II}$ can make the public key replacement query on the challenge identity id^* in this phase.

Guess: Finally, $\mathcal{A}_{II}$ outputs a guess $\beta^{'} \in \{0,1\}$ for β. $\mathcal{B}$ ignores the result and checks whether $< Q_{id^*}, U^*, A^*, T^* >$ exists in the list H_2List-B. If it does, it outputs A^* as the solution to the CDH problem. Otherwise, it will lose the game. From the above simulation, it is easy to deduce that $A^* = abP$ which is the right solution to the CDH problem.

Using the same method in the proof of Theorem 1, it is not difficult to deduce that $\mathcal{B}$'s advantage in solving the CDH problem is $\varepsilon^{'} \geq \frac{\varepsilon}{q_1} - \nu$, where ν denotes the advantage of the adversary against the chosen-ciphertext security of the symmetric encryption scheme (E, D).

This completes the proof of this theorem. □

By combining Theorem 1 and Theorem 2, we can deduce that:

Corollary 1. If the symmetric encryption scheme (E, D) is IND-CCA secure, then our CBE scheme is IND-CBE-CCA2 secure under the hardness of the CDH problem and the Gap-BDH problem in the random oracle model.

4.3 Efficiency Analysis and Comparison

In this section, we will make a comparison of our CBE scheme and other existing ones. In the computation cost comparison, we consider three major operations: Pairing (p), Multiplication (m), and Exponentiation (e). As usual, all symmetric operations are ignored. Some CBE schemes need a one-time signature scheme or a combination of an encapsulation scheme and a message authentication code to guarantee the IND-CBE-CCA2 security. We denote the signing algorithm and the verification algorithm in the signature scheme by (*Sign*, *Vfy*). In the communication cost comparison, ciphertext expansion represents the length difference between the ciphertext and the plaintext. The length of a string X is denoted by $|X|$. We denote the public commitment string and the de-commitment string by *com* and *dec* respectively of an encapsulation scheme, the message authentication code by *mac* of a message authentication code, and the verification key and the signature by *vk* and σ respectively of a one-time signature scheme. In [1] and [21], l should be at least 160 in order to obtain a reasonable security. Furthermore, “rom” means random oracle model while “standard” means standard model. Considering the pre-computation, the detailed performances of all the CBE schemes are listed in Table 1.

Table 1. Efficiency Comparison of the CBE Schemes

Scheme	Model	Computation cost		Communication cost		
		Encryption	Decryption	Ciphertext expansion	Public key	Certificate
[1]	rom	$2p+1m+1e$	$1p+1m$	$\|G_1\|+l$	$\|G_1\|$	$\|G_1\|$
[15]	standard	$4m+2e$	$3p+3m$	$3\|G_1\|+\|dec\|+\|com\|+\|mac\|$	$2\|G_1\|$	$2\|G_1\|$
[18]	standard	$5m+2e+Sign$	$3p+3m+Vfy$	$3\|G_1\|+\|vk\|+\|\sigma\|$	$2\|G_1\|$	$2\|G_1\|$
[19]	standard	$2m+7e$	$2p+1m+2e$	$2\|G_2\|+\|G_1\|$	$6\|G_1\|$	$3\|G_1\|+3\|Z_q\|$
[21]	rom	$2m+2e$	$1p+1m+1e$	$\|G_1\|+l$	$\|G_2\|$	$\|G_1\|$
Ours	rom	$2m+1e$	$1p+1m$	$\|G_1\|$	$\|G_1\|$	$\|G_1\|$

From the table, we can see that our scheme has better performances both on the computation efficiency and the communication bandwidth, while its security is proved in a stronger security model. What is worth mentioning is that our scheme introduces non redundancies in ciphertexts and has short public keys and short certificates. So, our scheme is more suitable for the bandwidth limited network.

5 Conclusion

In this paper, we define a new security model of CBE schemes which strengthens the power of adversaries against CBE and captures the public key replacement attack. Our model is more elaborated and stronger by comparison with other existing security models of CBE schemes. We also proposed a new CBE scheme and proved it to be IND-CBE-CCA2 secure in the proposed security model. When compared with the existing CBE schemes, our scheme enjoys better performances both on the computation efficiency and the communication bandwidth, while its security reaches a stronger level. However, the security of our scheme is only proved in the random oracle model. Therefore, how to design a strongly secure and efficient CBE scheme in the

standard model is an interesting open problem.

References

[1] C. Gentry, “Certificate-Based Encryption and the Certificate Revocation Problem”, Proc. Advances in Cryptology - Eurocrypt’03, LNCS 2656, pp. 272-293, 2003.

[2] M. Bellare, P. Rogaway, “Random Oracles are Practical: A Paradigm for Designing Efficient Protocols”, Proc. 1st ACM Conference on Communications and Computer Security, pp. 62-73, 1993.

[3] D. Boneh, M. Franklin, “Identity-Based Encryption from the Weil Pairing”, Proc. Advances in Crptology - Crypto’01, LNCS 2139, pp.213-229, 2001.

[4] D. H. Yum, P. J. Lee, “Identity-Based Cryptography in Public Key Management”, Proc. EuroPKI 2004, LNCS 3093, pp.71-84, 2004.

[5] S. S. Al-Riyami, K. G. Paterson, “Certificateless Public Key Cryptography”, Proc. Advances in Cryptology - Asiacrypt’03, LNCS 2894, pp.452-473, 2003.

[6] D. Galindo, P. Morillo, C. Ràfols, “Breaking Yum and Lee Generic Constructions of Certificateless and Certificate-Based Encryption Schemes”, Proc. EuroPKI 2006, LNCS 4043, pp.81-91, 2006.

[7] Y. Lu, J. Li, J. Xiao, “Generic Construction of Certificate-Based Encryption”, Proc. 9th International Conference for Young Computer Scientists, pp.1518-1594, 2008.

[8] E. Fujisaki, T. Okamoto, “Secure Integration of Asymmetric and Symmetric Encryption Schemes”, Proc. Advances in Cryptology - Crypto’99, LNCS 1666, pp. 537-554, 1999.

[9] E. Fujisaki, T. Okamoto, “How to Enhance the Security of Public-Key Encryption at Minimum Cost”, Proc. Public Key Cryptography - PKC’99, LNCS 1560, pp. 53-68, 1999.

[10] Y. Lu, J. Li, “Generic Construction of Certificate-based Encryption in the Standard Model”, Proc. 2nd International Symposium on Electronic Commerce and Security, pp.25-29, 2009.

[11] S. S. Al-Riyami, K. G. Paterson, “CBE from CL-PKE: A Generic Construction and Efficient Schemes”, Proc. Public Key Cryptography - PKC’05, LNCS 3386, pp. 398-415, 2005.

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

[13] D.H. Yum, P.J. Lee, “Separable Implicit Certificate Revocation”, Proc. 7th International Conference on Information Security and Cryptology, LNCS 3506, pp. 121-136, 2005.

[14] J. H. Park, D. H. Lee, “On the Security of Status Certificate-Based Encryption Scheme”, IEICE Trans. Fundamentals, E90-A(1): 303-304, 2007.

[15] P. Morillo, C. Ràfols, “Certificate-Based Encryption without Random Oracles”, Cryptology ePrint Archive, Report 2006/12.

[16] B. Waters, “Efficient Identity-Based Encryption without Random Oracles” , Proc. Advances in Cryptology - Eurocrypt’05, LNCS 3494, pp. 114-127, 2005.

[17] D. Boneh, X. Boyen, “Efficient Selective-ID Secure Identity Based Encryption without Random Oracles”, Proc. Advances in Cryptology - Eurocrypt’04, LNCS 3027, pp. 223-238, 2004.

[18] D. Galindo, P. Morillo, C. Ràfols, “Improved Certificate-Based Encryption in the Standard Model”, Journal of Systems and Software, vol.81(7):1218-1226, 2008.

[19] J. K. Liu, J. Zhou, “Efficient Certificate-Based Encryption in the Standard Model”, Proc. 6th International Conference on Security and Cryptography for Networks, LNCS 5229, pp.144-155, 2008.

[20] C. Gentry, “Practical Identity-Based Encryption without Random Oracles”, Proc. Advances in Cryptology - Eurocrypt’06, LNCS 4004, pp. 445-464, 2006.

[21] Y. Lu, J. Li, J. Xiao, “Constructing Efficient Certificate-Based Encryption with Paring”, Journal of Computers, vol.4(1): 19-26, 2009.

[22] L. Q. Chen, Z. H. Cheng, “Security Proof of Sakai-Kasahara’s Identity-Based Encryption Scheme”, Proc. Cryptography and Coding 2005, LNCS 3706, pp. 442-459, 2005.

[23] R. Sakai, M. Kasahara, “ID Based Cryptosystems with Pairing on Elliptic Curve”, Cryptology ePrint Archive, Report 2003/054.

[24] W. Wu, Y. Mu, W. Susilo, X. Huang, “Certificate-based Signatures Revisited”, Journal of Universal Computer Science, 15(8): 1659-1684, 2009.

[25] T. Okamoto, D. Pointcheval, “The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes”, Proc. Public Key Cryptography - PKC’01, LNCS 1992, pp. 104-118, 2001.

[26] B. Libert, J. J. Quisquater, “Identity Based Encryption without Redundancy”, Proc. 3rd International Conference on Applied Cryptography and Network Security, LNCS 3531, pp. 285-300, 2005.