Threshold Certificate-based Encryption

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

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

***Abstract*—Certificate-based encryption (CBE) is a new asymmetric encryption paradigm which combines traditional public-key encryption (PKE) and identity based encryption (IBE) while preserving some of their most attractive 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 introduce a new notion called Threshold Certificate-Based Encryption (TCBE) to overcome the limitations of CBE due to the using of sole master key in the system. It preserves the advantages of CBE such as implicit certificate and no private key escrow. At the same time it inherits the properties of threshold encryption. We first formalize the definition and security model for TCBE. Then we propose a concrete TCBE scheme and prove it to be CCA-secure under the Decisional Bilinear Diffie-Hellman assumption in the standard model.**

***Index Terms*—threshold certificate-based encryption, security model, CCA-secure, standard model**

I. INTRODUCTION

In traditional public key cryptography (PKC), a Public Key Infrastructure (PKI) is used to provide an assurance to the user about the relationship between a public key and the identity of the holder of the corresponding private key by certificates. However, the need for PKI supporting certificates is considered the main difficulty in the deployment and management of traditional PKC. To simplify the management, the certificates, Shamir [1] introduced identity-based cryptography (IBC) in which the public key of each user is derived directly from certain aspects of its identity, such as an IP address or an e-mail address, and the corresponding private key is generated by a trusted third party called Private Key Generator (PKG). For a long while it was an open problem to obtain a secure and efficient identity based encryption (IBE) scheme. Until 2001, Boneh and Franklin [3] presented the first practical and provably secure IBE scheme (BF-IBE) using the bilinear pairings on elliptic curves. The main practical benefit of IBC lies in greatly reduction of need for public key certificates. However, the PKG can generate the private keys of all its users, so private key escrow becomes an inherent problem in IBC. Moreover, private keys must be sent to the users over secure channels. It makes private key distribution a daunting task [7]. To fill the gap between traditional PKC and IBC, Al-Riyami and Paterson proposed a new paradigm called certificateless public key cryptography (CL-PKC) [4] in 2003. CL-PKC eliminates the key-escrow problem inherent in IBC. At the same time, it preserves the advantage of IBC which is the absence of digital certificates and their heavy management overhead. In CL-PKC, a trusted third party called Key Generating Center (KGC) is involved in the process of issuing a partial secret key for each user. The user independently generates its public/private key pair and combines the partial secret key from the KGC with its private key to generate the actual decryption key. By way of contrast to the PKG in IBC, the KGC does not have access to the user's decryption key. Therefore, CL-PKC solves the key escrow problem. However, due to the lack of public key authentication (certificate), CL-PKC is pointed out that it suffers from the Denial-of-Decryption (DOD) attack [6]. Moreover, CL-PKC suffers the same key distribution problem as IBC because partial secret keys must be sent to the users over secure channels.

In Eurocrypt 2003, Gentry [7] introduced the notion of certificate-based encryption (CBE), which combines identity-based encryption and traditional PKI-supported public key encryption (PKE) while preserving some of their most attractive 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. 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 certificate 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 [8] and OCSP [9]. Therefore, CBE can be used to construct an efficient PKI requiring fewer infrastructures. Although a CBE scheme is inefficient when a CA has a large number of users and performs frequent certificate updates, this problem can be overcome by using *subset covers* [7]. Furthermore, there is no key escrow and key distribution problem in CBE.

A. Related Work

Since the introduction of CBE [7], in which Gentry proposed a CBE scheme and proved the security in the random oracle model [22,23], there are different variants or improvements proposed in the literature later on. Yum and Lee [10] provided a formal equivalence theorem among IBE, CL-PKE and CBE. They showed that IBE implies both CBE and CL-PKE by giving a generic construction from IBE to those primitives. The same authors [11] 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. [18] pointed out that a dishonest authority could break the security of the three generic constructions given in [10,11]. These constructions were inherently flawed due to a naive use of double encryption without further treatments. We solved this problem by providing a security-enhancing conversion [14] and achieved two generic CBE constructions from PKE and IBE [15], which are provably CCA-secure in the random oracle model. Al-Riyami and Paterson [5] 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 [16] pointed out that their conversion was incorrect due to the flaw in their security proof. In [18], 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 PKEs) and to build CBE schemes without resorting to the random oracle model. Recently, Wang et al. [19] proposed a certificate-based proxy cryptosystem based on Gentry's CBE scheme. Moreover, Galindo et al. [13] proposed the first construction of CBE scheme secure in the standard model. In parallel to CBE, Kang et al. [17] 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. [20] pointed out that this signature scheme was insecure against the key replacement attack. They refined the security model of CBS given in [17] and constructed a new CBS scheme which is secure in the random oracle model. Moreover, Au et al. [21] propose a new notion called certificate-based ring signature, which is the ring signature in certificate based cryptography setting.

B. Our Contribution

We propose a new notion called Threshold Certificate-based Encryption (TCBE) to overcome the limitations of CBE due to the sole master key in the system. In TCBE, the system master key is not stored in a single location and held by a single trusted authority as in CBE, but shared by a set of trusted authorities, and each user needs to generate his certificate by combining adequate certificate shares from these trusted authorities. Therefore, it becomes more difficult to compromise the master key or to perform Denial-of-Service (DoS) attacks against the trusted authorities. To capture the collusive-attack scenarios in the threshold cryptography setting, we refine the security model of CBE by redefining the Type II adversary to model the collusive trusted authorities who have adequate master key shares to generate the certificates for any users in the system and to attack a fixed user's public key as in CBE. We also construct a concrete CCA-secure TCBE scheme which is proven under the Decisional Bilinear Diffie-Hellman (DBDH) assumption in the standard model.

II. Background Definitions

A. Certificate-based Encryption

In this section, we briefly review the definition and security model for CBE. These definitions are taken from [5], where the original definitions given in [7] were reconsidered.

Definition 1. Formally, a certificate-based encryption 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 Λ. It returns the certifier's master-key sk_{CA} and public parameters *params* that include the descriptions of a finite message space *MSPC* and a finite ciphertext space *CSPC*. We consider *params* to be an implicit input to the rest of the algorithms. Usually, this algorithm is run by a CA.
- *SetKeyPair* is a probabilistic algorithm that outputs a public/private key pair $<upk, usk>$.
- *Certify* is an algorithm that takes as input $<sk_{CA}, \tau, id, upk>$. It returns a short-lived certificate $Cert_{id,\tau}$, which is sent to the user *id*. Here τ is a string identifying a time period, while *id* contains other information needed to certify the user, and *upk* is the user's public key.
- *Enc* is a probabilistic algorithm taking as inputs $<\tau, id, upk, M>$, where $M \in MSPC$. It returns a ciphertext $C \in CSPC$ for message *M*.
- *Dec* is a deterministic algorithm taking $<Cert_{id,\tau}, usk, C>$ 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 for all $M \in MSPC$, $Dec(Cert_{id,\tau}, usk, Enc(\tau, id, upk, M)) = M$ where $Cert_{id,\tau} = Certify(sk_{CA}, \tau, id, upk)$ and $<upk, usk>$ is a valid key pair.

The security model for CBE is defined using two different games and the adversary chooses which game to play. In Game 1, the Type I adversary $\mathcal{A}_1$ has no access to the master-key and models an uncertified client and in

Game 2, the Type II adversary $\mathcal{A}_2$ models an honest-but-curious certifier who possesses the master-key sk_{CA} attacking a fixed use's public key. In [5], Game 2 requires that *params* and sk_{CA} are fixed at the beginning and are supplied to the adversary, while it in [7] is allowed to work with multiple values of *params*. Kang and Park [16] pointed out that the restriction in [5] is sufficiently reasonable because CA does not change its public parameters frequently. The strongest security notion of a CBE scheme is IND-CBE-CCA which is defined as follows:

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 the following CBE Game 1 and CBE Game 2:

CBE 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. $\mathcal{A}_1$ adaptively interleaves certificate and decryption queries, $\mathcal{C}$ handles these queries as follows:

On certificate query $<\tau, id, upk>$, $\mathcal{C}$ runs *Certify* on input $<sk_{CA}, \tau, id, upk>$ to generate $Cert_{id,\tau}$.

On decryption query $<\tau, id, upk, usk, C>$, $\mathcal{C}$ runs *Certify* on input $<sk_{CA}, \tau, id, upk>$ to obtain $Cert_{id,\tau}$ and outputs $Dec(Cert_{id,\tau}, usk, C)$.

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

Phase 2. As in phase 1, with the restriction that $<\tau_{ch}, id_{ch}, upk_{ch}, usk_{ch}, C_{ch}>$ is not the subject of a decryption query and $<\tau_{ch}, id_{ch}, upk_{ch}>$ was not the subject of a certificate query.

Guess. $\mathcal{A}_1$ 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 $Adv(\mathcal{A}_1) = 2|\Pr[b = b'] - 1/2|$.

CBE 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 $<upk_{ch}, usk_{ch}>$ and gives upk_{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, $\mathcal{C}$ runs *Certify* on input $<sk_{CA}, \tau, id, upk_{ch}>$ to obtain $Cert_{id,\tau}$ and outputs $Dec(Cert_{id,\tau}, usk_{ch}, C)$ to $\mathcal{A}_2$.

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

Phase 2. As in phase 1, with the restriction that $<\tau_{ch}, id_{ch}, C_{ch}>$ 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 $Adv(\mathcal{A}_2) = 2|\Pr[b = b'] - 1/2|$.

B. One Time Signature

Definition 3. A signature scheme is a triple of PPT algorithms (*KeyGen*, *Sign*, *Verify*) such that:

- *KeyGen* is a probabilistic algorithm taking as input a security parameter Λ. It outputs a signing key *sk* and a signature verification key *svk*.
- *Sign* is an algorithm taking as input a signing key *sk* a message *m*. It outputs a signature σ.
- *Verify* is a deterministic algorithm taking as input a verification key *svk*, a message *m* and a signature σ. It outputs 1 iff the signature is valid.

It is required that for all (*sk*, *svk*) output by *KeyGen*, all *m* in the message space, and all σ output by *Sign*(*sk*, *m*), we have *Verify*(*svk*, *m*, σ) = 1.

The standard notion of the security of a signature scheme is existential unforgeability under a chosen message attack [25]. A one-time signature (OTS) [24] is a signature scheme with strong existential unforgeability under a one-time chosen message attack, which is a security notion stronger than the standard one. Roughly speaking, an OTS scheme is required to be secure that an adversary should be unable to forge a new signature even on a previously signed message.

C. Mathematical Assumption

Throughout the paper, G_1 and G_2 denote two multiplicative cyclic groups of prime order p; g is a generator of G_1^*; e is a bilinear map $e: G_1 \times G_1 \to G_2$; $\mathcal{GG}(\Lambda)$ is a bilinear group generator. For us, a bilinear paring is a map $e: G_1 \times G_1 \to G_2$ with following three properties: *Bilinear*: For all $u, v \in G_1$ and $a, b \in Z_p^*$, $e(u^a, v^b) = e(u, v)^{ab}$; *Non-degenerate*: $e(g, g) \neq 1_{G_2}$; *Computable*: For all $u, v \in G_1$, $e(u, v)$ can be efficiently computed. A bilinear paring satisfying these three properties is said to be an admissible bilinear map. A bilinear group generator $\mathcal{GG}(\Lambda)$ takes a security parameter $\Lambda \in Z^+$ as input and outputs the description of groups G_1 and G_2 and a bilinear map $e: G_1 \times G_1 \to G_2$ where the group operation in G_1 and G_2 as well as e can be computed in polynomial time in Λ.

The bilinear Diffie-Hellman (BDH) problem in G_1 is as follows: given a tuple $g, g^a, g^b, g^c \in G_1$ as input where $a, b, c \in Z_p^*$, output $e(g, g)^{abc}$. An algorithm $\mathcal{A}$ has advantage ε in solving the BDH problem in G_1 if $\Pr[\mathcal{A}(g, g^a, g^b, g^c) = e(g, g)^{abc}] \geq \varepsilon$, where the probability is over the random choice of $g \in G_1^*$, the random choice of $a, b, c \in Z_p^*$, and the random bits used by $\mathcal{A}$.

Similarly, we say that an algorithm $\mathcal{B}$ that outputs $b \in \{0,1\}$ has advantage ε in solving the decisional BDH

(DBDH) problem in G_1 if $|\Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g, g)^{abc}) = 0] - \Pr[\mathcal{B}(g, g^a, g^b, g^c, T) = 0]| \geq \varepsilon$, where the probability is over the random choice of $g \in G_1^*$, the random choice of $T \in G_2$, and the random bits consumed by $\mathcal{B}$.

Definition 4. The DBDH assumption holds in G_1 if any polynomial time algorithm has negligible advantage in solving the DBDH problem in G_1.

III. Threshold Certificate-based Encryption

In this section, we give the formal definition and security model for threshold certificate-based encryption.

Definition 5. A threshold certificate-based encryption (TCBE) scheme consists of following seven PPT algorithms:

• *Setup* is a probabilistic algorithm that takes as input $<n, k, \Lambda>$ where n is the number of certificate servers, k ($1 \leq k \leq n$) is a threshold, and Λ is a security parameter. It outputs a triple (*params*, *vk*, ***msk***) where *params* is the system public parameters, *vk* is a public verification key, and ***msk*** = (msk_1, msk_2,…,msk_n) is a vector of master key shares. Certificate server i is given the master key share (i, msk_i). 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 <*upk*, *usk*>.

• *ShareCertify* is a probabilistic algorithm that takes as input <i, msk_i, τ, *id*, *upk*>. It outputs a short-live certificate share $Cert^i_{id,\tau}$. Here τ is a string identifying a time period, while *id* contains the information needed to certify the user, and *upk* is the user's public key.

• *ShareVerify* is a deterministic algorithm that takes as input $<\tau, id, upk, vk, Cert^i_{id,\tau}>$. It outputs ***valid*** or ***invalid***.

• *Combine* is a deterministic algorithm that takes as input $<\tau, id, upk, vk, \{ Cert^{i_1}_{id,\tau}, \ldots, Cert^{i_k}_{id,\tau} \}>$ where $i_1, \ldots, i_k \in \{1, 2, \ldots, n\}$. If one of $Cert^{i_1}_{id,\tau}, \ldots, Cert^{i_k}_{id,\tau}$ is ***invalid***, or if two certificate shares bear the same server index, it outputs $\perp$. Otherwise, it outputs a short-live certificate $Cert_{id,\tau}$ for the user *id* during the time period τ.

• *Enc* is a probabilistic algorithm that takes as input $<\tau$, *id*, *upk*, *M*> and outputs a ciphertext *C*.

• *Dec* is a deterministic algorithm taking $<Cert_{id,\tau}$, *usk*, *C*> as input in time period τ. It outputs either a message *M* or the special symbol $\perp$ indicating a decryption failure.

It is required that these algorithms must satisfy the following consistency requirements:

1. For any $<\tau$, *id*, *upk*>, if $Cert^i_{id,\tau} = ShareCertify(i, msk_i, \tau, id, upk)$ where msk_i is one of the master key share in ***msk***, then $ShareVerify(vk, \tau, id, upk, Cert^i_{id,\tau}) =$ ***valid***.

2. For any $<\tau$, *id*, *upk*, *usk*>, if $S = \{ Cert^{i_1}_{id,\tau}, \ldots, Cert^{i_k}_{id,\tau} \}$ are k certificate shares where $Cert^{i_j}_{id,\tau} = ShareCertify(i_j, msk_{i_j}, \tau, id, upk)$ and $Cert_{id,\tau}$ is the output of $Combine(vk, \tau, id, upk, S)$, then for any *M* we have $Dec(Cert_{id,\tau}, usk, Enc(\tau, id, upk, M)) = M$.

The security of a TCBE scheme is defined against two different types of adversaries: the Type I adversary $\mathcal{A}_1$ has access to at most k-1 master key shares and models an uncertified user; the Type II adversary $\mathcal{A}_2$ has access to at least k master key shares and models the collusive certificate servers to attack a fixed user's public key. The strongest security notion of a TCBE scheme is defined as follows:

Definition 6. A TCBE scheme is secure against adaptive chosen-ciphertext attacks (or IND-TCBE-CCA) if no polynomial-time adversary has non-negligible advantage in the following TCBE Game 1 and TCBE Game 2:

TCBE Game 1.

Init. The adversary $\mathcal{A}_1$ outputs a set of k-1 distinct certificate servers $S \subset \{1,\ldots,n\}$ that it wishes to corrupt.

Setup. The challenger $\mathcal{C}$ runs $Setup(n, k, \Lambda)$ to obtain a random instance (*params*, *vk*, ***msk***) where ***msk*** = (msk_1, msk_2,…, msk_n). It gives *params*, *vk* and all (j, msk_j) for $j \in S$ to the adversary $\mathcal{A}_1$.

Phase 1. The adversary $\mathcal{A}_1$ interleaves a series of certificate share and decryption queries, $\mathcal{C}$ handles these queries as follows:

• On certificate share query $<i, \tau, id, upk>$ where $i \notin S$, $\mathcal{C}$ responds $\mathcal{A}_1$ with $ShareCertify(i, msk_i, \tau, id, upk)$.

• On decryption query $<\tau, id, upk, usk, C>$, $\mathcal{C}$ first runs *ShareCertify* and *Combine* to generate $Cert_{id,\tau}$ for the user *id* during the time period τ, then responds $\mathcal{A}_1$ with $Dec(Cert_{id,\tau}, usk, C)$.

Challenge. On challenge query $<\tau_{ch}, id_{ch}, upk_{ch}, usk_{ch}, M_0, M_1>$, where M_0, M_1 are of equal length, $\mathcal{C}$ checks that $<\tau_{ch}, id_{ch}, upk_{ch}, usk_{ch}>$ was not the subject of a certificate share query in phase 1. If so, $\mathcal{C}$ picks a random bit $b \in \{0,1\}$ and sets the challenge ciphertext to $C_{ch} = Enc(\tau_{ch}, id_{ch}, upk_{ch}, M_b)$. It outputs C_{ch} to $\mathcal{A}_1$.

Phase 2. As in phase 1, with the restriction that $<\tau_{ch}, id_{ch}, upk_{ch}, usk_{ch}, C_{ch}>$ is not the subject of a decryption query and $<\tau_{ch}, id_{ch}, pk_{ch}>$ is not the subject of a certificate share 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 $Adv(\mathcal{A}_1) = 2\,|\Pr[b = b'] - 1/2|$.

TCBE Game 2.

Init. The adversary $\mathcal{A}_2$ outputs a set of k distinct master key shares $S \subset \{1,\ldots,n\}$ that it wishes to access.

Setup. The challenger $\mathcal{C}$ runs $Setup(n, k, \Lambda)$ to obtain a random instance (*params*, *vk*, ***msk***) where ***msk*** = (msk_1, msk_2,…, msk_n). It gives *params*, *vk* and all (j, msk_j) for $j \in S$ to the adversary $\mathcal{A}_2$. Then $\mathcal{C}$ runs *SetKeyPair* to obtain a key-pair $<upk_{ch}, usk_{ch}>$ and gives upk_{ch} to $\mathcal{A}_2$.

Phase 1. $\mathcal{A}_2$ issues a series of decryption queries of the form $<\tau, id, C>$. On such a query, $\mathcal{C}$ first runs *ShareCertify* and *Combine* to generate $Cert_{id,\tau}$ for the user *id* during the time period τ, then responds $\mathcal{A}_2$ with $Dec(Cert_{id,\tau}, usk_{ch}, C)$.

Challenge. On challenge query $<\tau_{ch}, id_{ch}, M_0, M_1>$, where M_0, M_1 are of equal length. $\mathcal{C}$ picks a random bit $b \in \{0,1\}$ and sets the challenge ciphertext to $C_{ch} = Enc(\tau_{ch}, id_{ch}, upk_{ch}, M_b)$. It outputs C_{ch} to $\mathcal{A}_2$.

Phase 2. As in phase 1, with the restriction that $<\tau_{ch}, id_{ch}, C_{ch}>$ 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 $Adv(\mathcal{A}_2) = 2|\Pr[b = b']-1/2|$.

IV. Building Block

In this section, we present an efficient CBE scheme presented by Galindo, Morillo and Ràfols [13], together with its security statements. We will use it as a building block and extend it to a TCBE scheme in the next section. This CBE scheme is described as follows:

Setup: Given a security parameter $\Lambda \in Z^+$, this algorithm performs as follows: Run $\mathcal{GG}(\Lambda)$ to obtain $<G_1, G_2, e>$, where G_1 and G_2 of prime order p. Randomly select $g, g_2, g_3 \in G_1^*$, $\alpha \in Z_q^*$ and set $g_1 = g^\alpha$. Randomly select $u' \in G_1$ and $u_i \in G_1$ for $i = 1,2,\ldots,l$. Set $U = (u_i)$ be an l-length vector. Choose two collision-resistant hash functions $H_1 : \{0,1\}^* \to \{0,1\}^l$ and $H_2 : \{0,1\}^* \to Z_p$. The public system parameters are $params = \{\Lambda, p, l, e, G_1, G_2, g, g_1, g_2, g_3, U, H_1, H_2\}$ and the system master-key is α.

SetKeyPair: User randomly chooses $x \in Z_q^*$ and $g' \in G_1^*$, then sets his private key as $usk = x$ and his public key as $upk = < upk_1, upk_2> = < g^x, g' >$.

Certify: On input $<\alpha, \tau, id, upk>$, this algorithm firstly computes $v = H_1(\tau||id||upk_1||upk_2)$. Let $v[i]$ be the i-th bit of v and $V \subseteq \{1,2,\ldots,l\}$ be the set of all i for which $v[i] = 1$. Then, it randomly select $r \in G_1^*$ and computes

$$Cert_{id,\tau} =< Cert_{id,\tau_1}, Cert_{id,\tau_2} >=< g_2^\alpha Q^r, g^r >,$$

where $Q = u' \prod_{i \in V} u_i$.

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

(1) Generate a one-time signature key-pair (svk, sk).

(2) Compute $Q = u' \prod_{i \in V} u_i$ and $h = H_2(svk)$.

(3) Randomly select $s \in Z_q^*$ and compute

$$C =< e(g_1, g_2)^s e(upk_1, upk_2)^s M, g^s, Q^s, (g_1^h upk_1^h g_3)^s >.$$

(4) Compute $\sigma = Sign(sk, C)$ and output $<C, svk, \sigma>$.

Dec: On input $<Cert_{id,\tau}, upk, usk, <C, svk, \sigma>>$, this algorithm performs the following steps:

(1) Check if $Verify(svk, C, \sigma) = 1$. If not, output $\perp$ and abort.

(2) Let $h = H_2(svk)$. Randomly select $t \in Z_q^*$ and set $(d_1, d_2, d_3) = (Cert_{id,\tau_1}(g_1^h upk_1^h g_3)^t upk_2^{usk}, Cert_{id,\tau_2}, g^t)$.

(3) Compute and output $M = \dfrac{C_1 e(d_2, C_3) e(d_3, C_4)}{e(d_1, C_2)}$.

Theorem 1. *The above CBE scheme is IND-CBE-CCA secure if H_1 and H_2 are two collision-resistant hash functions, OTS is a strongly unforgeable signature scheme and the DBDH assumption holds in G_1.*

Proof. See [13] for the security reduction and concrete security statement.

For simplification, we call the above CBE scheme as GMR-CBE scheme in the following text.

V. A Concrete TCBE Scheme

In this section, we build a concrete TCBE scheme which is IND-TCBE-CCA secure in the standard model. Our scheme can be viewed as an extension of the GMR-CBE scheme by applying Shamir's secret-sharing technology [2] to the system master key.

A. Construction

The scheme is described as follows:

Setup: On input $<n, k, \Lambda>$, this algorithm generates (*params*, *vk*, ***msk***) as follows: Run a bilinear group generator $\mathcal{GG}$ on input Λ and obtain $< G_1, G_2, e>$, where G_1 and G_2 of prime order p. Choose random generators $g, g_2, g_3 \in G_1^*$ and a random degree k-1 polynomial $f(x) = \sum_{i=0}^{k-1} a_i x^i \in Z_p[X]$. Set $\alpha = f(0) \in Z_p^*$ and $g_1 = g^\alpha$. Randomly select $u' \in G_1$ and $u_i \in G_1$ for $i = 1,2,\ldots,l$. Set $U = (u_i)$ be an l-length vector. Choose two collision-resistant hash functions $H_1 : \{0,1\}^* \to \{0,1\}^l$ and $H_2 : \{0,1\}^* \to Z_p$. The public system parameters are $params = \{n, k, \Lambda, p, l, e, G_1, G_2, g, g_1, g_2, g_3, U, H_1, H_2\}$. For $i = 1,2,\ldots,n$ the master key share msk_i of certificate server i is defined as $f(i)$. The public verification key $vk = (vk_1, vk_2,\ldots, vk_n) = (g^{f(1)}, g^{f(2)}, ..., g^{f(n)})$.

SetKeyPair: User randomly chooses $x \in Z_q^*$ and $g' \in G_1^*$, then sets his private key as $usk = x$ and his public key as $upk = < upk_1, upk_2> = < g^x, g' >$.

ShareCertify: On input $<i, msk_i, \tau, id, upk>$, this algorithm first computes $v = H_1(\tau||id||upk_1||upk_2)$. Let $v[i]$ be the i-th bit of v and $V \subseteq \{1,2,\ldots,l\}$ be the set of all i for which $v[i] = 1$. It then sets $Q = u' \prod_{i \in V} u_i$, selects a random $r \in G_1^*$ and computes the certification share $Cert_{id,\tau}^i$ as

$$Cert_{id,\tau}^i =< Cert_{id,\tau_1}^i, Cert_{id,\tau_2}^i >=< g_2^{msk_i} Q^r, g^r >.$$

ShareVerify: On input <*vk*, *τ*, *id*, *upk*, $Cert^{i}_{id,\tau}$>, this algorithm outputs ***valid*** or ***invalid*** according to the truth of the following condition:

$$e(vk_i, g_2) \cdot e(Q, Cert^{i}_{id,\tau_2}) = e(g, Cert^{i}_{id,\tau_1}).$$

Combine: On input <*vk*, *τ*, *id*, *upk*, { $Cert^{i_1}_{id,\tau}$,…, $Cert^{i_k}_{id,\tau}$ }> where $i_1,...,i_k \in \{1,2,...n\}$, this algorithm first checks whether one of { $Cert^{i_1}_{id,\tau}$,…, $Cert^{i_k}_{id,\tau}$ } is invalid, or two certification shares bear the same server index. If so, it outputs ⊥. Otherwise, it first computes the Lagrange coefficients $\lambda_1,...,\lambda_k \in Z_p$ such that $s = f(0) = \sum_{j=1}^{k} \lambda_j f(i_j)$, then computes the certification $Cert_{id,\tau}$ as

$$Cert_{id,\tau} = < \prod_{j=1}^{k} (Cert^{i_j}_{id,\tau_1})^{\lambda_j}, \prod_{j=1}^{k} (Cert^{i_j}_{id,\tau_2})^{\lambda_j} >.$$

Enc: On input <*τ*, *id*, *upk*, *M*>, this algorithm performs the following steps:

(1) Generate a one-time signature key-pair (*svk*, *sk*).

(2) Compute $Q = u' \prod_{i \in V} u_i$ and $h = H_2(svk)$.

(3) Randomly select $s \in Z_q^*$ and compute

$$C = < e(g_1, g_2)^s e(upk_1, upk_2)^s M, g^s, Q^s, (g_1^h upk_1^h g_3)^s >.$$

(4) Compute $\sigma = Sign(sk, C)$ and output <*C*, *svk*, *σ*>.

Dec: On input <$Cert_{id,\tau}$, *upk*, *usk*, <*C*, *svk*, *σ*>>, this algorithm performs the following steps:

(1) Check if $Verify(svk, C, \sigma) = 1$. If not, output ⊥ and abort.

(2) Let $h = H_2(svk)$. Randomly select $t \in Z_q^*$ and set (*d*1, *d*2, *d*3) = ($Cert_{id,\tau_1} (g_1^h upk_1^h g_3)^t upk_2^{usk}$, $Cert_{id,\tau_2}$, g^t).

(3) Compute and output $M = \frac{C_1 e(d_2, C_3) e(d_3, C_4)}{e(d_1, C_2)}$.

The consistency of above construction can be checked as follows:

(1) For any <*τ*, *id*, *upk*>, let $Cert^{i}_{id,\tau} = < g_2^{f(i)} Q^r, g^r >$ be a certification share generated by *ShareCertify*(*i*, msk_i, *τ*, *id*, *upk*), it is easy to check that the following equation $e(g^{f(i)}, g_2) \cdot e(Q, g^r) = e(g, g_2^{f(i)} Q^r)$ is hold. Therefore, *ShareVerify*(*vk*, *τ*, *id*, *upk*, $Cert^{i}_{id,\tau}$) = ***valid***.

(2) For any <*τ*, *id*, *upk*, *usk*>, if *S* = { $Cert^{i_1}_{id,\tau}$,…, $Cert^{i_k}_{id,\tau}$ } are *k* certificate shares where $Cert^{i_j}_{id,\tau}$ = *ShareCertify* ($i_j, msk_{i_j}, \tau, id, upk$), let the certificate generated by *Combine*(*vk*, *τ*, *id*, *upk*, *S*) be $Cert_{id,\tau}$ = $= < \prod_{j=1}^{k} (Cert^{i_j}_{id,\tau_1})^{\lambda_j}, \prod_{j=1}^{k} (Cert^{i_j}_{id,\tau_2})^{\lambda_j} >$ where $\lambda_1,...,\lambda_k \in Z_p$ are the Lagrange coefficients such that $s = f(0) = \sum_{j=1}^{k} \lambda_j f(i_j)$.

Then we have

$$\prod_{j=1}^{k} (Cert^{i_j}_{id,\tau_1})^{\lambda_j} = g_2^{\sum_{j=1}^{k} \lambda_j f(i_j)} Q^{\sum_{j=1}^{k} r_j \lambda_j} = g_2^{\alpha} Q^{\tilde{r}}, \quad \tilde{r} = \sum_{j=1}^{k} r_j \lambda_j$$

$$\prod_{j=1}^{k} (Cert^{i_j}_{id,\tau_2})^{\lambda_j} = g^{\sum_{j=1}^{k} r_j \lambda_j} = g^{\tilde{r}}.$$

Therefore, the certificate is $Cert_{id,\tau} = < g_2^{\alpha} Q^{\tilde{r}}, g^{\tilde{r}} >$. Then for any *M* and *C* = *Enc*(*τ*, *id*, *upk*, *M*), it is easy to check that *Dec*($Cert_{id,\tau}$, *usk*, *C*) = *M* as we have

$$\frac{C_1 e(Cert_{id,\tau_2}, C_3) e(g^t, C_4)}{e(Cert_{id,\tau_1} (g_1^h upk_1^h g_3)^t upk_2^{usk}, C_2)}$$

$$= \frac{e(g_1, g_2)^s e(g^x, g')^s M e(g^{\tilde{r}}, Q^s) e(g^t, (g_1^h (g^x)^h g_3)^s)}{e(g_2^{\alpha} Q^{\tilde{r}} (g_1^h (g^x)^h g_3)^t (g')^x, g^s)} = M.$$

B. Security Proof

Now, we prove the security of the above TCBE scheme based on the security of the GMR-CBE scheme.

Lemma 1. *The above TCBE scheme is secure against Type I IND-TCBE-CCA adversary if the GMR-CBE scheme is secure against Type I IND-CBE-CCA adversary.*

Proof. Suppose that there exists a Type I IND-TCBE-CCA adversary $\mathcal{A}_1$ against the above TCBE scheme with advantage *ε*, we show how to make use of the adversary $\mathcal{A}_1$ to construct a Type I IND-CBE-CCA adversary $\mathcal{B}_1$ against the GMR-CBE scheme with advantage *ε*.

The challenger $\mathcal{C}$ starts an IND-CBE-CCA Game by executing algorithm *Setup* of the GMR-CBE scheme to generate the system parameters *params* = {*Λ*, *p*, *l*, *e*, G_1, G_2, *g*, g_1, g_2, g_3, *U*, H_1, H_2} and the system master-key *α*. The challenger $\mathcal{C}$ passes *params* to $\mathcal{B}_1$. Now, $\mathcal{B}_1$ interacts with $\mathcal{A}_1$ as follows:

Init. $\mathcal{A}_1$ outputs a set of *k*-1 distinct certification servers $S \subset \{1,\ldots,n\}$ that it wishes to corrupt.

Setup. $\mathcal{B}_1$ first chooses *k*-1 random integers $a_1,...,a_{k-1} \in Z_p$ and sets the master key shares for the *k*-1 corrupt servers in *S* to be $msk_{i_j} = a_j$ for $i = 1,...,k-1$ and $i_j \in S$. Let $f(x) \in Z_p[X]$ be the degree *k*-1 polynomial satisfying $f(0) = \alpha$ and $f(i_j) = a_j$ for $i = 1,...,k-1$, hence these master key shares are consistent with this polynomial *f* since $msk_{i_j} = f(i_j)$ for $i = 1,...,k-1$. Note that $\mathcal{B}_1$ does not know *f* since it does not know *α*. Then, $\mathcal{B}_1$ constructs a public verification key *vk* which is a *n*-tuple (vk_1, vk_2,…, vk_n) such that $vk_i = g^{f(i)}$ for the polynomial *f* defined above as follows: for $i \in S$, $\mathcal{B}_1$ sets $vk_i = g^{a_i}$ since $f(i) = a_i$; for $i \notin S$, $\mathcal{B}_1$ first computes the Lagrange coefficients $\lambda_1,...,\lambda_{k-1} \in Z_p$ such

that $f(i) = \lambda_0 f(0) + \sum_{j=1}^{k-1} \lambda_j f(i_j)$. Note that these Lagrange coefficients are easily calculated since they do not depend on *f*. Then $\mathcal{B}_1$ sets $vk_i = g_1^{\lambda_0} \prod_{j=1}^{k-1} msk_{i_j}^{\lambda_j}$ which satisfies that $vk_i = g^{f(i)}$ as required. Finally, $\mathcal{B}_1$ gives $\mathcal{A}_1$ the system parameters *params* = {*n*, *k*, Λ, *p*, *l*, *e*, G_1, G_2, *g*, g_1, g_2, g_3, *U*, H_1, H_2}, the *k*-1 master key shares for the *k*-1 corrupt servers in *S*, and the public verification key *vk*.

Phase 1. $\mathcal{A}_1$ interleaves a series of certificate share and decryption queries, $\mathcal{B}_1$ handles these queries as follows:

On certificate share query <*i*, τ, *id*, *upk*, *usk*> where $i \notin S$, $\mathcal{B}_1$ first forwards <τ, *id*, *upk*> as the input to the certificate query to the challenger $\mathcal{C}$ and obtains the certificate $Cert_{id,\tau} = < Cert_{id,\tau_1}, Cert_{id,\tau_2} > = < g_2^{\alpha} Q^r, g^r >$. Then, it computes the Lagrange coefficients $\lambda_1, ..., \lambda_{k-1} \in Z_p$ such that $f(i) = \lambda_0 f(0) + \sum_{j=1}^{k-1} \lambda_j f(i_j)$, picks a random $r \in Z_p^*$, and sets

$$Cert_{id,\tau_1}^i = (Cert_{id,\tau_1})^{\lambda_0} \prod_{j=1}^{k-1} g_2^{\lambda_j a_j},\ Cert_{id,\tau_2}^i = (Cert_{id,\tau_2})^{\lambda_0}.$$

It is easy to check that $Cert_{id,\tau}^i = < Cert_{id,\tau_1}^i, Cert_{id,\tau_2}^i >$ is a valid response to this certificate share query. To see this, let $\tilde{r} = \lambda_0 r$, then we have that

$$Cert_{id,\tau_1}^i = (g_2^{\alpha} Q^r)^{\lambda_0} \prod_{j=1}^{k-1} g_2^{\lambda_j a_j} = g_2^{f(i)} Q^{\tilde{r}},$$

$$Cert_{id,\tau_2}^i = (g^r)^{\lambda_0} = g^{\tilde{r}}.$$

On decryption query <τ, *id*, *upk*, *usk*, <*C*, *svk*, σ>>, $\mathcal{B}_1$ forwards <τ, *id*, *upk*, *usk*, <*C*, *svk*, σ>> to the challenger $\mathcal{C}$ and outputs the result to $\mathcal{A}_1$.

Challenge. Once $\mathcal{A}_1$ outputs <τ_{ch}, id_{ch}, upk_{ch}, usk_{ch}, M_0, M_1> on which it wants to be challenged, $\mathcal{B}_1$ checks that <τ_{ch}, id_{ch}, upk_{ch}> is not the subject of a certificate share query in phase 1. If so, $\mathcal{B}_1$ forwards <τ_{ch}, id_{ch}, upk_{ch}, usk_{ch}, M_0, M_1> as its challenge to the challenger $\mathcal{C}$ and obtains the challenge ciphertext $C_{ch} = Enc(\tau_{ch}, id_{ch}, upk_{ch}, M_b)$ where $b \in \{0,1\}$ is a random bit. It outputs C_{ch} to $\mathcal{A}_1$.

Phase 2. As in phase 1, with the restriction that <τ_{ch}, id_{ch}, upk_{ch}, usk_{ch}, C_{ch}> is not the subject of a decryption query and <τ_{ch}, id_{ch}, upk_{ch}> is not the subject of a certification share query.

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

It is readily seen that $\mathcal{B}_1$ perfectly simulates all the oracles for $\mathcal{A}_1$. If $\mathcal{A}_1$ succeeds in guessing the bit *b*, then $\mathcal{B}_1$ also succeeds in outputting the correct bit. Therefore, the advantage that $\mathcal{B}_1$ breaks the IND-CBE-CCA security of the GMR-CBE scheme is ε. This completes the proof of Lemma 1.

Lemma 2. *The above TCBE scheme is secure against Type II IND-TCBE-CCA adversary if the GMR-CBE scheme is secure against Type II IND-CBE-CCA adversary.*

Proof. Suppose that there exists a Type II IND-TCBE-CCA adversary $\mathcal{A}_2$ against the above TCBE scheme with advantage ε, we show how to make use of the adversary $\mathcal{A}_2$ to construct a Type II IND-CBE-CCA adversary $\mathcal{B}_2$ against the GMR-CBE scheme with advantage ε.

The challenger $\mathcal{C}$ starts an IND-CBE-CCA Game by executing algorithm *Setup* of the GMR-CBE scheme to generate the system parameters *params* = {Λ, *p*, *l*, *e*, G_1, G_2, *g*, g_1, g_2, g_3, *U*, H_1, H_2} and the system master key α. It also runs the algorithm *SetKeyPair* to generate a key-pair <upk_{ch}, psk_{ch}>. The challenger $\mathcal{C}$ gives $\mathcal{B}_2$ the system parameters *params*, the master key α, and the challenge public key upk_{ch}. Now, $\mathcal{B}_2$ interacts with $\mathcal{A}_2$ as follows:

Init. $\mathcal{A}_2$ outputs a set of *k* distinct master key shares *S* $\subset$ {1,…,*n*} that it wishes to access.

Setup. $\mathcal{B}_2$ first choose a random degree *k*-1 polynomial $f(x) = \sum_{i=0}^{k-1} a_i x^i \in Z_p[X]$ such that $f(0) = \alpha$. Then it sets the master key share msk_i of certificate server *i* as *f*(*i*) for *i* = 1,2,…,*n*, and the public verification key *vk* as $(g^{f(1)}, g^{f(2)}, ..., g^{f(n)})$. Finally, $\mathcal{B}_2$ gives $\mathcal{A}_2$ the system parameters *params* = {*n*, *k*, Λ, *p*, *l*, *e*, G_1, G_2, *g*, g_1, g_2, g_3, *U*, H_1, H_2}, the *k* master key shares for the *k* servers in *S*, the public verification key *vk* and the challenge public key upk_{ch} to $\mathcal{A}_2$.

Phase 1. The adversary $\mathcal{A}_2$ issues a series of decryption queries. Let <τ, *id*, <*C*, *svk*, σ >> be a decryption query issued by $\mathcal{A}_2$, $\mathcal{B}_2$ forwards <τ, *id*, <*C*, *svk*, σ>> to the challenger $\mathcal{C}$ and outputs the result to $\mathcal{A}_2$.

Challenge. At some point, $\mathcal{A}_2$ decides to end Phase 1 and outputs <τ_{ch}, id_{ch}, M_0, M_1> on which it wants to be challenged. $\mathcal{B}_2$ forwards <τ_{ch}, id_{ch}, M_0, M_1> as its challenge to the challenger $\mathcal{C}$ and obtains the challenge ciphertext $C_{ch} = Enc(\tau_{ch}, id_{ch}, upk_{ch}, M_b)$ where $b \in \{0,1\}$ is a random bit. It outputs C_{ch} to $\mathcal{A}_2$.

Phase 2. As in phase 1, with the restriction that <τ_{ch}, id_{ch}, C_{ch}> is not the subject of a decryption query.

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

It is readily seen that $\mathcal{B}_2$ perfectly simulates all the oracles for $\mathcal{A}_2$. If $\mathcal{A}_2$ succeeds in guessing the bit *b*, then $\mathcal{B}_2$ also succeeds in outputting the correct bit. Therefore, the advantage that $\mathcal{B}_2$ breaks the IND-CBE-CCA security

of the GMR-CBE scheme is ε. This completes the proof of Lemma 2.

By combining Lemma 1 and Lemma 2, we have the following theorem:

Theorem 2. *The above TCBE scheme is secure in the sense of IND-TCBE-CCA if the GMR-CBE scheme is secure in the sense of IND-CBE-CCA*.

Furthermore, we have the following conclusion from Theorem 1 and Theorem 2:

Theorem 3. *The above TCBE scheme is secure in the sense of IND-TCBE-CCA if H_1 and H_2 are two collision-resistant hash functions, OTS is a strongly unforgeable one-time signature scheme and the DBDH assumption holds in G_1.*

VI. Conclusion

In this paper, we propose a new notion called Threshold Certificate-Based Encryption. This notion makes that it becomes more difficult to compromise the CA's master key or to perform DoS attacks against CA. We not only provide the formal definition and security model for TCBE, but also present a concrete construction which is proven to be IND-TCBE-CCA secure in the standard model. We believe that the notion of TCBE is of practical interest in some application, such as electronic commerce and certificate systems.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (No. 60842002), the National High Technology Research and Development Program of China (No. 2007AA01Z409), and the Natural Science Foundation of Hohai University (No. 2008428611).

References

[1] A. Shamir, " Identity-based cryptosystems and signature schemes," In Advances in Cryptology- CRYPTO'84, USA, LNCS 196, pp. 47-53, 1984.

[2] A. Shamir, "How to share a secret," Communications of the ACM, vol. 22, pp. 612-613, 1979.

[3] 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.

[4] 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.

[5] 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.

[6] J. K. Liu, M. H. Au, and W. Susilo, "Self-generated-certificate public key cryptography and certificateless signature/encryption scheme in the standard model," In ACM ASIACCS 2207, Singapore, pp. 273-283, 2007.

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

[8] 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.

[9] 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.

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

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

[12] 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.

[13] 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.

[14] 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.

[15] Y. Lu, J. Li, and J. Xiao, "Generic construction of certificate-based encryption," In 9th International Conference for Young Computer Scientists, China, pp. 1518-1594, 2008.

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

[17] 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.

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

[19] 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.

[20] 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.

[21] 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.

[22] 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.

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

[24] R. Canetti, S. Halevi, and J. Katz, "Chosen-ciphertext security from identity-based encryption," In Advances in Cryptology - EUROCRYPT 2004, Switzerland, LNCS 3027, pp. 207–22, 2004.

[25] 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.