Certificate-Based Signcryption Scheme without Pairing: Directly Verifying Signcrypted Messages Using a Public Key

Minh-Ha Le and Seong Oun Hwang

To achieve confidentiality, integrity, authentication, and non-repudiation simultaneously, the concept of signcryption was introduced by combining encryption and a signature in a single scheme. Certificate-based encryption schemes are designed to resolve the key escrow problem of identity-based encryption, as well as to simplify the certificate management problem in traditional public key cryptosystems. In this paper, we propose a new certificate-based signcryption scheme that has been proved to be secure against adaptive chosen ciphertext attacks and existentially unforgeable against chosen-message attacks in the random oracle model. Our scheme is not based on pairing and thus is efficient and practical. Furthermore, it allows a signcrypted message to be immediately verified by the public key of the sender. This means that verification and decryption of the signcrypted message are decoupled. To the best of our knowledge, this is the first signcryption scheme without pairing to have this feature.

Keywords: Signcryption, certificate-based signcryption, certificate-based public key cryptography.

Manuscript received Nov. 15, 2015; revised Mar. 3, 2016, accepted Apr. 19, 2016.

It was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2014R1A1A2054174). It was also supported by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the Global IT Talent support program (IITP-2016-H0905-15-1004) supervised by the IITP (Institute for Information and Communication Technology Promotion).

Minh-Ha Le (leminhha1989@gmail.com) is with the Department of Electronics and Computer Engineering, Hongik University, Sejong, Rep. of Korea.

Seong Oun Hwang (corresponding author, sohwang@hongik.ac.kr) is with the Department of Computer and Information Communications Engineering, Hongik University, Sejong, Rep. of Korea.

I. Introduction

Authentication is a fundamental block of a secure system. Basically, it is a process for verifying that the identity of an entity belongs to a human or a device. For example, in the authentication process, a certificate in traditional public key cryptography (PKC) is usually used to prove that a public key belongs to a specific user. However, a public key infrastructure (PKI) that supports a traditional PKC has issues, such as complex installation and maintenance processes, issuance, distribution, and a revocation of the certificates.

Although the authentication process seems to be irreplaceable, some public key cryptography models have been proposed in which the certificate is eliminated. In 1984, Shamir proposed the first concept of identity-based public key cryptography (ID-PKC) [1]. This scheme shows a great improvement, that is, it does not require PKI because the public key is an identity (for example, email, ID number, driver license number, and so on). In ID-PKC, a private key generator (PKG) uses a master secret key to generate all private keys for its users. The PKG requires secure channels to deliver the private keys to users securely. Although the improvement in ID-PKC is significant, some architectural issues still remain: (1) A secure channel to deliver the private keys is significantly costly to implement. (2) The PKG can impersonate any user at any time because it knows the private keys of all users, which is called the key escrow problem. This issue is unacceptable in certain cases such as legal applications because the PKG cannot guarantee non-repudiation. (3) Finally, the security of the whole system depends on the secrecy of the master secret

http://dx.doi.org/10.4218/etrij.16.0115.0983

key. If the PKG is compromised and the master key is revealed, the whole system is affected.

To overcome the drawbacks of the traditional PKC and ID-PKC, the first concept of certificateless public key cryptography (CL-PKC) was proposed by Al-Riyami and Paterson [2]. As the name implies, CL-PKC inherits the advantages of ID-PKC in the sense that it does not require a certificate for a public key. Furthermore, it also eliminates the key escrow problem owing to the fact that it allows users to create their own public key and private key pairs; the private key is kept secret so that even a trusted authority, called the key generation center (KGC), cannot decrypt the user messages. To decrypt a ciphertext, it requires both a partial private key generated by the KGC and a private key generated by the user. Unfortunately, there are no certificates protecting the public keys, and thus they can be replaced by an attacker who wants to prevent a receiver from decrypting a ciphertext. In CL-PKC, secure channels are still needed to distribute the partial private keys to users. In addition, if the KGC is compromised, we cannot prevent an attacker from changing the public key to impersonate any user in the system.

In 2003, Gentry proposed the notion of certificate-based cryptography (CB-PKC) [3], which uses the PKI in a more efficient manner. Compared with the previous models, CB-PKC seems to be a promising solution for the key escrow problem and enhances the PKI. As in PKC, each user generates their own public and private key pair, and requests a certificate to the CA. The crucial difference is that the CA uses an identity-based encryption (IBE) scheme to generate the certificate: The CA treats the user's public key as their identity, and generates its corresponding private key, called a certificate, which serves as a partial private key. Eventually, CB-PKC preserves all of the features of traditional PKCs, while simplifying the PKIs, and has none of the key escrow problem found in ID-PKC.

In 1997, Zheng [4] defined a new cryptographic concept of signcryption, which is a combination of both functions of encryption and signature simultaneously. This method is more efficient when compared to the sign-then-encrypt approach because the combination of encryption and signature reduces both computational cost and communication overhead. Following this method, we can achieve confidentiality, integrity, authentication, and non-repudiation concurrently.

1. Related Work

Since the concept of PKC was first proposed by Diffie and Hellman in 1976 [5], it has attracted the attention of many cryptographers, and has quickly became the main topic of modern cryptography. To improve the efficiency of traditional PKC, Shamir proposed the first concept of ID-PKC [1]. Boneh and Franklin proposed the first concrete construction of an IBE scheme [6]. Since then, a number of IBE schemes have been proposed [7]–[10]. By combining a public key encryption scheme and a public key signature scheme into a single scheme, Zheng proposed the first signcryption scheme in 1997 [4]. Bao proposed another signcryption scheme in which the signature is directly verifiable through a public key [11]. We note that, in this scheme, the signcrypted message still needs to be decrypted before it can be verified. In 2000, two more signcryption schemes were proposed with their own applications: one scheme, proposed by Seo and Kim [12], called a domain-verifiable signcryption scheme, is applied to electronic funds, and the other, proposed by Mu and Varadharajan [13], is a distributed signcryption scheme. The distributed signcryption scheme was improved by Kwak and Moon in 2003 [14]. In the same year, Boyen [15] proposed a multi-purpose signcryption scheme together with a comprehensive security model for a multi-purpose identity-based signcryption cryptosystem. After that, many identity-based signcryption schemes were proposed [16]–[21]. In 2008, Selvi and others [22] also proposed the first concept of certificateless signcryption.

Gentry proposed the first notion of CB-PKC [3]. It turned out that a CBE scheme can be constructed from certificateless public key encryption (CL-PKE) [23]. Wu and others proposed another generic construction of CBE from CL-PKE [24]. Many other CBE schemes have been proposed [25]–[29]. In 2006, Morillo and others proposed the first CBE scheme without random oracles [30]. After that, Liu and Zhou proposed an efficient CBE scheme in the standard model [31], which Galindo and others improved in [32]. In parallel with CB-PKC, Al-riyami and Paterson introduced the concept of CL-PKC, and proposed the first concrete scheme in 2003 [2]. Some other CL-PKC schemes have also been proposed [33]–[36].

Although the concept of CB-PKC was proposed in 2003, the first certificate-based signcryption (CBSC) was first introduced in 2008 by Li and others [25]. Lou and colleagues [37] proposed another CBSC scheme with a security proof, which turned out to be unsecure under two concrete attacks, described in [38] and [39]. In [39], the CBSC scheme was claimed to be secure against public key replacement and insider attacks. Recently, Lu and Li [40] proposed a new CBSC without pairing, and proved it to be secure using the random oracle model.

II. Preliminaries

1. Computational Diffie-Hellman Problem (CDH)

Let p_1 and p_2 be prime numbers such that $p_2 | p_1 - 1$. Let g be a generator of $\mathbb{Z}_{p_1}^*$. The CDH problem in $\mathbb{Z}_{p_1}^*$ is given (g, g^a, g^b) for a uniformly chosen $a, b \in \mathbb{Z}_{p_2}^*$ to compute g^{ab}. The advantage of any polynomial-time algorithm A_{CDH} in solving the CDH problem in $\mathbb{Z}_{p_1}^*$ is defined as

$$\text{Adv}(A_{\text{CDH}}) = Pr[A_{\text{CDH}}(\mathbb{Z}_{p_1}^*, p, g, g^a, g^b) = g^{ab} \mid a, b \in \mathbb{Z}_{p_2}].$$

The CDH assumption is that, for any polynomial-time algorithm A_{CDH}, the advantage A_{CDH} is negligible.

2. Discrete Logarithm (DL) Problem

Let p be a prime number, and g be a generator of $\mathbb{Z}_p$. The DL problem in $\mathbb{Z}_p$ is, given a tuple (g, g^a) for unknown $a \in \mathbb{Z}_p$, to compute a. The advantage of any polynomial-time algorithm A_{DL} in solving the DL problem in G is defined as

$$\text{Adv}(A_{\text{DL}}) = Pr[A_{\text{DL}}(\mathbb{Z}_p, p, g, g^a) = a \mid a \in \mathbb{Z}_p].$$

The DL assumption is that, for any polynomial-time algorithm A_{DL}, the advantage $\text{Adv}(A_{\text{DL}})$ is negligible.

3. Certificate-Based Signcryption Scheme

In this subsection, we provide an outline of the certificate-based signcryption scheme. The scheme is defined by five algorithms as follows:

- **Setup:** This algorithm is run at the CA side. Given security parameter 1^k, it returns the master secret key *msk* and system parameters *params* of the CA.
- **SetKeyPair:** This algorithm is run at the user side. Given *params*, it returns a public key *pk* and secret key *sk* for a user.
- **Certification:** This algorithm is run at the CA side. Given the user identity *ID*, the system parameters *params*, and the user public key *pk*, it returns a certificate *Cert*, which will be sent to the user over an open channel. In particular, in our scheme, *pk* will be updated with the help of CA after the certification step.
- **Signcryption:** This algorithm is run by a sender. Given a message *m*, the identities of the sender and receiver ID_{S} and ID_{R}, the certificate $Cert_{\text{S}}$ and secret key sk_{S} of the sender, public keys of the sender and receiver pk_{S} and pk_{R}, and the system parameters *params*, it returns a signcrypted message C = **Signcryption** $(M, ID_{\text{S}}, ID_{\text{R}}, Cert_{\text{S}}, sk_{\text{S}}, pk_{\text{S}}, pk_{\text{R}}, params)$.
- **Designcryption:** Given a signcrypted message C, the identities of the sender and receiver ID_{S} and ID_{R}, the certificate $Cert_{\text{R}}$ and secret key sk_{R} of the receiver, the public keys of the sender and receiver (pk_{S} and pk_{R}), and the system parameters *params*, it returns a message M' = **Designcryption** $(C, ID_{\text{S}}, ID_{\text{R}}, Cert_{\text{R}}, pk_{\text{S}}, pk_{\text{R}}, sk_{\text{R}}, params)$, which is equal to M, or the symbol $\perp$, indicating that C is an invalid signcryption between ID_{S} and ID_{R}.
- **Correctness:** If C is the result of applying the Signcryption algorithm with inputs $(M, ID_{\text{S}}, ID_{\text{R}}, Cert_{\text{S}}, sk_{\text{S}}, pk_{\text{S}}, pk_{\text{R}}, params)$, then M', which is the result of the designcryption algorithm, must be equal to M. We write this as **Designcryption** $(C, ID_{\text{S}}, ID_{\text{R}}, Cert_{\text{R}}, pk_{\text{S}}, pk_{\text{R}}, sk_{\text{R}}, params) = M$.

4. Security models of CBSC

CBSC schemes have to be secure in terms of both confidentiality and unforgeability.

A. Confidentiality:

As mentioned above, there are two kinds of adversary:

- A Type I adversary corresponds to indistinguishability under adaptive chosen ciphertext attacks, (IND-CBSC-CCA2) game I, from a normal client or an uncertified client who is not given the master secret key *msk* of the CA.
- A Type II adversary corresponds to the indistinguishability under adaptive chosen ciphertext attacks, (IND-CBSC-CCA2) game II, from a certified client who has the master secret key *msk* of the CA. Compared to IND-CBSC-CCA2 game I, the difference is that a Type II adversary is given the master secret key and the adversary does not have to query a $\mathbf{O}^{\mathbf{Certification}}$ because it can generate the certificate itself using the master secret key. Note that the simulation of an attack from a Type II adversary is necessary because a certificate-based cryptographic scheme is aimed at resolving the key escrow problem.

Because these two games have the same structure, we describe the models of both games as a single model and note the differences as below:

IND-CBSC-CCA2 games I and II:

A CBSC scheme is IND-CBSC-CCA2 secure against Types I and II adversaries if neither probabilistic polynomial-time adversary $\mathcal{A}_{\text{I}}$ or $\mathcal{A}_{\text{II}}$ has a non-negligible advantage in the following game:

Setup: Challenger $\mathcal{B}$ is given the security parameter 1^k. It runs the setup algorithm and returns public parameters *params* and master secret key *msk*. In IND-CBSC-CCA2 game I, the *params* are given to $\mathcal{A}_{\text{I}}$ and the challenger keeps the *msk* for itself. In IND-CBSC-CCA2 game II, both *params* and *msk* are given to $\mathcal{A}_{\text{II}}$.

Phase I: In phase I, adversary $\mathcal{A}_{\text{I}}$ ($\mathcal{A}_{\text{II}}$) makes queries and $\mathcal{B}$ answers them as follows:

$\mathbf{O}^{\mathbf{CreateUser}}$: Upon receiving an identity *ID*, the challenger generates a secret key *sk*, public key *pk*, and certificate *Cert*,

http://dx.doi.org/10.4218/etrij.16.0115.0983

and then responds to the ID with a public key pk.

$\mathbf{O}^{\text{RequestPrivateKey}}$: Upon receiving an identity ID, the challenger responds to the ID with a private key sk.

$\mathbf{O}^{\text{Certification}}$: Upon receiving a tuple (ID, pk), the challenger responds to the ID with $Cert$. Note that this oracle is used only by adversary $\mathcal{A}_{\text{I}}$. The adversary $\mathcal{A}_{\text{II}}$ does not have to query this oracle because it can generate the certificate by itself using the master secret key of the CA.

$\mathbf{O}^{\text{Signcryption}}$: Upn receiving a tuple (ID_{S}, ID_{R}, pk_{S}, sk_{S}, $Cert_{\text{S}}$, pk_{R}, M), the challenger responds with a corresponding signcrypted message C.

$\mathbf{O}^{\text{Designcryption}}$: Upon receiving a tuple (ID_{S}, ID_{R}, pk_{S}, pk_{R}, sk_{R}, $Cert_{\text{S}}$, C), the challenger responds with a corresponding plaintext message M.

Challenge: In this phase, $\mathcal{A}_{\text{I}}$ ($\mathcal{A}_{\text{II}}$) outputs two equal-length messages M_0 and M_1, and the identities of the sender and receiver $(ID_{\text{S}}^*, ID_{\text{R}}^*)$. The challenger chooses a bit $\gamma \in \{0,1\}$ at random and computes the signcrypted message C^* from $params$, $(ID_{\text{S}}^*, ID_{\text{R}}^*)$; the public keys of the sender and receiver $pk_{\text{S}}^*, pk_{\text{R}}^*$; the secret key of the sender sk_{S}^*; the certificate of the sender $Cert_{\text{S}}^*$; and M_γ.

Phase II: $\mathcal{A}_{\text{I}}$ ($\mathcal{A}_{\text{II}}$) continuously queries the oracles as in phase I, with two restrictions: (1) a query with $< ID_{\text{R}}^* >$ cannot be submitted to the $\mathbf{O}^{\text{Certification}}$ oracle, and (2) decryption query with $< C^*, ID_{\text{S}}^*, ID_{\text{R}}^* >$ cannot be submitted to the $\mathbf{O}^{\text{Designcryption}}$ oracle.

Guess: Finally, $\mathcal{A}_{\text{I}}$ ($\mathcal{A}_{\text{II}}$) terminates the game by outputting a guess γ' for γ. The advantage of $\mathcal{A}_{\text{I}}$ in the game is defined as follows:

$$\text{Adv}_{\mathcal{A}_{\text{I}}}^{\text{IND-CBSC-CCA2}} = 2\,|Pr[\gamma = \gamma'] - 1/2\,|.$$

In addition, the advantage of $\mathcal{A}_{\text{II}}$ is defined as below:

$$\text{Adv}_{\mathcal{A}_{\text{II}}}^{\text{IND-CBSC-CCA2}} = 2\,|Pr[\gamma = \gamma'] - 1/2\,|.$$

B. Unforgeability

Similarly to the *confidentiality* models, there are two kinds of adversaries: Types I and II adversaries .

EUF-CBSC-CMA games I and II:

The challenger $\mathcal{B}$ is given security parameter 1^k. It runs the setup algorithm and returns public parameters $params$ and master secret key msk. In EUF-CBSC-CMA game I, $params$ are given to $\mathcal{A}_{\text{I}}$ and the challenger keeps msk for itself. In EUF-CBSC-CMA game II, both $params$ and msk are given to $\mathcal{A}_{\text{II}}$. Adversary $\mathcal{A}_{\text{I}}$ ($\mathcal{A}_{\text{II}}$) makes queries, and $\mathcal{B}$ answers them as follows:

$\mathbf{O}^{\text{CreateUser}}$**:** Upon receiving the identity ID, the challenger generates secret key sk, public key pk, and certificate $Cert$, and then responds to the ID with public key pk.

$\mathbf{O}^{\text{RequestPrivateKey}}$**:** Upon receiving an identity ID, the challenger responds to the ID with private key sk.

$\mathbf{O}^{\text{Certification}}$**:** Upon receiving a tuple (ID, pk), the challenger responds to the ID with $Cert$. This oracle is used only by adversary $\mathcal{A}_{\text{I}}$. In EUF-CBSC-CMA game II, adversary $\mathcal{A}_{\text{II}}$ has the master secret key of the CA, and thus it can generate the certificate by itself.

$\mathbf{O}^{\text{Signcryption}}$: Upon receiving a tuple (ID_{S}, ID_{R}, pk_{S}, sk_{S}, $Cert_{\text{S}}$, pk_{R}, M), the challenger responds with the signcrypted message C.

$\mathbf{O}^{\text{Designcryption}}$: Upon receiving a tuple (ID_{S}, ID_{R}, pk_{S}, pk_{R}, sk_{R}, $Cert_{\text{R}}$, C), the challenger responds with a plaintext message M.

Forge: Finally, $\mathcal{A}_{\text{I}}$ ($\mathcal{A}_{\text{II}}$) outputs a forged signcrypted message $(C^*, ID_{\text{S}}^*, ID_{\text{R}}^*)$, which is not produced by the signcrypt query $\mathbf{O}^{\text{Signcryption}}$, and ID_{S}^* is not submitted to the certification query $\mathbf{O}^{\text{Certification}}$. Here, $\mathcal{A}_{\text{I}}$ ($\mathcal{A}_{\text{II}}$) wins if the result of the designcryption with $(C^*, ID_{\text{S}}^*, ID_{\text{R}}^*, pk_{\text{S}}^*, pk_{\text{R}}^*, sk_{\text{R}}^*, Cert_{\text{R}})$ is not a $\perp$ symbol.

Let $Pr[\mathcal{A}_{\text{I}}]$ ($Pr[\mathcal{A}_{\text{II}}]$) be the probability that adversary $\mathcal{A}_{\text{I}}$ ($\mathcal{A}_{\text{II}}$) successfully generates a forged message. We define the advantage of $\mathcal{A}_{\text{I}}$ in the above game as follows:

$$\text{Adv}_{\mathcal{A}_{\text{I}}}^{\text{EUF-CBSC-CMA}} = Pr[\mathcal{A}_{\text{I}}].$$

In addition, the advantage of $\mathcal{A}_{\text{II}}$ is defined as follows:

$$\text{Adv}_{\mathcal{A}_{\text{II}}}^{\text{EUF-CBSC-CMA}} = Pr[\mathcal{A}_{\text{II}}].$$

III. Proposed Scheme

1. Construction

Let p_1 and p_2 be two large prime integers such that $p_2|p_1 - 1$.

Setup: The CA picks a generator g of $\mathbb{Z}_{p_1}^*$ and random $\alpha \in \mathbb{Z}_{p_2}^*$. It sets $g_1 = g^{\alpha}(mod\ p_1)$. Four hash functions will be chosen: $H_1 : \{0, 1\}^* \times \mathbb{Z}_{p_1}^* \times \{0,1\}^n \times \mathbb{Z}_{p_1}^* \to \mathbb{Z}_{p_2}^*$; $H_2 : \{0,1\}^* \times \mathbb{Z}_{p_1}^* \times \mathbb{Z}_{p_1}^* \to \mathbb{Z}_{p_2}^*$; $H_3 : \mathbb{Z}_{p_1}^* \times \mathbb{Z}_{p_1}^* \to \{0, 1\}^n$; and $H_4 : \mathbb{Z}_{p_1}^* \times \mathbb{Z}_{p_1}^* \times \{0, 1\}^n \times \mathbb{Z}_{p_1}^* \to \mathbb{Z}_{p_2}^*$. The public parameters $params$ and master key msk are as follows: $params = (p_1, p_2, g, g_1, H_1, H_2, H_3, H_4)$, $msk = \alpha$.

SetKeyPair: Given identity $ID = \{0, 1\}^*$ and $params$, this algorithm is run at the user side. A random element $x_{ID} \in \mathbb{Z}_{p_2}^*$ is chosen, and this value is set as the user's private key $sk_{ID} = x_{ID}$. The user's public key value is $U_{ID} = g^{x_{ID}}$. The key pair will be $(sk_{ID} = x_{ID}, U_{ID} = g^{x_{ID}})$.

Certification: To generate a certificate for an identity from inputs $ID = \{0, 1\}^*$, U_{ID} (received from the user) and $params$, the CA chooses a random value $\beta_{ID} \in \mathbb{Z}_{p_2}^*$. It computes $P_{ID} = g^{\beta_{ID}}$. The CA updates the public key of the user corresponding to the identity ID: $pk_{ID} = (U_{ID}, P_{ID})$. The CA then computes the certificate, $Cert_{ID} = \beta_{ID} + \alpha H_2(ID, U_{ID}, P_{ID})$.

Signcryption: Let $sk_S = x_S, (U_S = g^{x_S}, P_S)$, and $Cert_S$ be the private key, public key, and certificate of the sender, respectively. Here, $sk_R = x_R, (U_R = g^{x_R}, P_R)$, and $Cert_R$ are the private key, public key, and certificate of the receiver, respectively. To generate a signcrypted message of message $M = \{0, 1\}^n$ with (ID_S, ID_R), that is, the identities of the sender and receiver, respectively, the sender selects a random value $r \in \mathbb{Z}_{p_2}^*$ and computes the following:

a) $k = (U_R P_R g_1^{H_2(ID_R, U_R, P_R)})^r \pmod{p_1}$

b) $C_0 = g^r \pmod{p_1}$

c) $C_1 = H_3(k) \oplus M$

d) $C_2 = Cert_S + x_S H_4(U_S, P_S, C_1, C_0) + rH_1(ID_S, P_S, C_1, C_0)$

The sender outputs $C = (C_0, C_1, C_2)$.

Designcryption: To designcrypt the signcrypted message $C = (C_0, C_1, C_2)$, the receiver can execute the following steps separately:

a) Check whether $g^{C_2} = P_S g_1^{H_2(ID_S, U_S, P_S)} U_S^{H_4(U_S, P_S, C_1, C_0)} C_0^{H_1(ID_S, P_S, C_1, C_0)}$. If this equation holds, move to the next step. Otherwise, return $\perp$ and terminate the algorithm.

b) $M = H_3(C_0^{x_R + Cert_R} \bmod p_1) \oplus C_1$, and return the result, M.

2. Correctness

The correctness of our scheme is confirmed as follows:

a) We have $g^{C_2} = g^{Cert_S + x_S H_4(U_S, P_S, C_1, C_0) + rH_1(ID_S, P_S, C_1, C_0)}$, where $Cert_S = \beta_S + \alpha H_2(ID_S, U_S, P_S)$. Therefore,

$$g^{C_2} = g^{(\beta_S + \alpha H_2(ID_S, U_S, P_S) + x_S H_4(U_S, P_S, C_1, C_0) + rH_1(ID_S, P_S, C_1, C_0))}$$
$$= P_S g_1^{H_2(ID_S, U_S, P_S)} U_S^{H_4(U_S, P_S, C_1, C_0)} C_0^{H_1(ID_S, P_S, C_1, C_0)}.$$

b) We then have $C_0^{x_R + Cert_R} = (g^r)^{x_r + (\beta_R + \alpha H_2(ID_R, U_R, P_R))}$ $= U_R^r (g^{\beta_R})^r g_1^{H_2(ID_R, U_R, P_R)} = (U_R P_R g_1^{H_2(ID_R, U_R, P_R)})^r = k$.

Thus, $M = H_3(C_0^{x_R + Cert_R} \bmod p_1) \oplus C_1$
$= H_3(k) \oplus (H_3(k) \oplus M) = M$.

3. Security Proofs

The main idea of the security proofs for Theorem 1 is to have the CDH attacker $\mathcal{B}$ simulate the "environment" of the Type I and II attackers $\mathcal{A}_I$ and $\mathcal{A}_{II}$, respectively, until it can compute a Diffie-Hellman key, g^{ab} of g^a and g^b, using the abilities of $\mathcal{A}_I$ and $\mathcal{A}_{II}$. As described in Section II, $\mathcal{A}_I$ and $\mathcal{A}_{II}$ will issue various queries to the random oracles, the $\mathbf{O}^{\mathbf{CreateUser}}$ oracle, the $\mathbf{O}^{\mathbf{RequestPrivateKey}}$ oracle, the $\mathbf{O}^{\mathbf{Certification}}$ oracle, the $\mathbf{O}^{\mathbf{Signcryption}}$ oracle, and the $\mathbf{O}^{\mathbf{Designcryption}}$ oracle. $\mathcal{B}$ will respond to these queries with answers distributed identically as those in a real attack.

To answer to adversary $\mathcal{A}_I$, $\mathcal{B}$ sets g^a as a part of the challenge ciphertext and g^b ($g^b = g_1$) as the public key of the CA. On the other hand, to answer adversary $\mathcal{A}_{II}$, $\mathcal{B}$ sets g^a as a part of the challenge ciphertext, but uses g^b to generate a public key associated with the challenger identity (in this case, it is the public key of ID_θ, which will be described in the security proof), and the public key of the CA is set up as g^a, where $\mathcal{B}$ knows random $\alpha \in \mathbb{Z}_p^*$, and gives the master key $msk = \alpha$ of the CA to $\mathcal{A}_{II}$.

To prove the confidentiality of the proposed scheme, we prove the following theorem.

Theorem 1: Suppose that the CDH is intractable. The CBSC scheme above is IND-CBSC-CCA secure in the random oracle model.

This theorem can be proved by the following lemmas: Lemma 1 for the Type I adversary, and Lemma 2 for the Type II adversary.

Lemma 1: Suppose that H_1, H_2, H_3, H_4 are random oracles and $\mathcal{A}_I$ is an IND-CBSC-CCA2 Type I adversary that has advantage ϵ and running time τ against the CBSC scheme above. Here, $\mathcal{A}_I$ is allowed to make at most q_{cu} queries to the oracle $\mathbf{O}^{\mathbf{CreateUSer}}$, q_{pri} queries to the oracle $\mathbf{O}^{\mathbf{RequestPrivateKey}}$, q_{cer} queries to the oracle $\mathbf{O}^{\mathbf{Certification}}$, q_{sc} queries to the oracle $\mathbf{O}^{\mathbf{Signcryption}}$, q_{dsc} queries to the oracle $\mathbf{O}^{\mathbf{Designcryption}}$, and q_i queries to the random oracle H_i (i = 1, 2, 3, 4). An algorithm A_{CDH} exists to solve the CDH problem with the following advantage:

$$\epsilon' \geq \frac{\epsilon}{q_{cu} q_3}\left(1 - q_{sc}\frac{q_1 + q_2 + q_4 + 3q_{sc}}{2^k}\right)\left(1 - \frac{q_{dsc}}{2^k}\right),$$

and the running time $\tau' \leq \tau + (2q_{cu} + 5q_{sc} + 3q_{dsc})t_{exp}$, where t_{exp} denotes the time for an exponentiation.

Proof: We construct an algorithm $\mathcal{B}$ that solves the CDH problem by using $\mathcal{A}_I$. Here, $\mathcal{B}$ is given an instance of the CDH problem: p, q, g, g^a, g^b. To answer to adversary $\mathcal{A}_I$, $\mathcal{B}$ will set g^a as a part of the challenge ciphertext and g^b ($g^b = g_1$) as the public key of the CA. Here, $\mathcal{B}$ simulates a challenger and answers queries from $\mathcal{A}_I$ as below:

Setup: $\mathcal{B}$ randomly chooses $\theta \in [1, q_{cu}]$ (where q_{cu} is the number of queries to the $\mathbf{O}^{\mathbf{CreateUSer}}$ oracle), and sets $g_1 = g^b$. The *params* are set as ($p_1, p_2, g, g_1, H_1, H_2, H_3, H_4$). Then, the *params* are given to $\mathcal{A}_I$, which can query all oracles below at any time during its attack. Then, $\mathcal{B}$ answers the queries as follows:

$\mathbf{H_1}$-queries: $\mathcal{B}$ maintains a list, **$\mathbf{H_1}$List**, of the tuples $< ID_i, P_{ID_i}, C_{1,i}, C_{0,i}, h_{1,i} >$. Upon receiving the query (ID_i, $P_{ID_i}, C_{1,i}, C_{0,i}$), if **$\mathbf{H_1}$List** contains $< ID_i, P_{ID_i}, C_{1,i}, C_{0,i}, h_{1,i} >$ then $\mathcal{B}$ returns $h_{1,i}$ to $\mathcal{A}_I$. Otherwise, it randomly picks $h_{1,i} \in \mathbb{Z}_{p_2}^*$, returns $h_{1,i}$ to $\mathcal{A}_I$, and adds $< ID_i, P_{ID_i}, C_{1,i}$,

http://dx.doi.org/10.4218/etrij.16.0115.0983

$C_{0,i}, h_{1,i} >$ to **H₁List**.

H_2-queries: $\mathcal{B}$ maintains a list, **H_2List**, of tuples $< ID_i, U_{ID_i}, P_{ID_i}, h_{2,i} >$. Upon receiving the query (ID_i, U_{ID_i}, P_{ID_i}), if **H_2List** contains $< ID_i, U_{ID_i}, P_{ID_i}, h_{2,i} >$, then $\mathcal{B}$ returns $h_{2,i}$ to $\mathcal{A}_\mathrm{I}$. Otherwise, it randomly picks $h_{2,i} \in \mathbb{Z}^*_{p_2}$, returns $h_{2,i}$ to $\mathcal{A}_\mathrm{I}$, and adds $< ID_i, U_{ID_i}, P_{ID_i}, h_{2,i} >$ to **H_2List**.

H_3-queries: $\mathcal{B}$ maintains a list, **H_3List**, of tuples $<k_i, h_{3,i}>$. Upon receiving query (k_i), if **H_3List** contains $<k_i, h_{3,i}>$, then $\mathcal{B}$ returns $h_{3,i}$ to $\mathcal{A}_\mathrm{I}$. Otherwise, it randomly picks $h_{3,i} \in \{0,1\}^n$, returns $h_{3,i}$ to $\mathcal{A}_\mathrm{I}$ and adds $<k_i, h_{3,i}>$ to **H_3List**.

H_4-queries: $\mathcal{B}$ maintains a list, **H_4List**, of tuples $< U_{ID_i}, P_{ID_i}, C_{1,i}, C_{0,i}, h_{4,i} >$. Upon receiving query ($U_{ID_i}$, $P_{ID_i}, C_{1,i}, C_{0,i}$), if **$H_4$List** contains $< U_{ID_i}, P_{ID_i}, C_{1,i}, C_{0,i}, h_{4,i} >$, then $\mathcal{B}$ returns $h_{4,i}$ to $\mathcal{A}_\mathrm{I}$. Otherwise, it randomly picks $h_{4,i} \in \mathbb{Z}^*_{p_2}$, returns $h_{4,i}$ to $\mathcal{A}_\mathrm{I}$, and adds $< U_{ID_i}, P_{ID_i}, C_{1,i}, C_{0,i}, h_{4,i} >$ to **H_4List**.

Phase I:

$\mathbf{O}^{\text{CreateUser}}$: $\mathcal{B}$ maintains a user list, *UserList*: $< ID_i, sk_i, pk_i, Cert_i >$. Upon receiving the ID_i the following occurs:

- If $i = \theta$, then $\mathcal{B}$ randomly chooses $\beta_\theta \in \mathbb{Z}^*_{p_2}$, $x_\theta \in \mathbb{Z}^*_{p_2}$ and sets $U_\theta = g^{x_\theta}$. It computes $P_\theta = g^{\beta_\theta}$, inserts $< ID_{\theta_A}, x_\theta, (U_\theta, P_\theta), \perp>$ into *UserList* and responds with (U_θ, P_θ) to $\mathcal{A}_\mathrm{I}$.
- Otherwise, $\mathcal{B}$ generates sk_i, pk_i, $Cert_i$ as normal.

$\mathbf{O}^{\text{RequestPrivateKey}}$: Upon the input of identity ID_i, if $i = \theta$, $\mathcal{B}$ aborts the game. Otherwise, $\mathcal{B}$ searches for sk_i in the *UserList* and responds with the sk_i to $\mathcal{A}_\mathrm{I}$ if sk_i exists.

$\mathbf{O}^{\text{Certification}}$: Upon the input of identity ID_i, if $i = \theta$, $\mathcal{B}$ aborts the game. Otherwise, $\mathcal{B}$ searches for $Cert_i$ on the *UserList* and responds with the entry to $\mathcal{A}_\mathrm{I}$ if $Cert_i$ exists.

$\mathbf{O}^{\text{Signcryption}}$: $\mathcal{A}_\mathrm{I}$ gives $\mathcal{B}$ a tuple $<m, ID_\mathrm{S}, ID_\mathrm{R}>$. There are two cases:

- If $ID_\mathrm{S} = ID_\theta$, $\mathcal{B}$ randomly chooses $C_2 \in \mathbb{Z}^*_{p_2}$ and $C_1 \in \{0,1\}^n$, and randomly chooses h_1, h_2, and h_4 from $\mathbb{Z}^*_{p_2}$. $\mathcal{B}$ runs the simulation for $\mathbf{O}^{\text{CreateUser}}$ to obtain U_θ, and computes $C_0 = g^{C_2} / \left(g^{\beta_\theta} g_1^{h_2} U_\theta^{h_4} \right)$. After that, $\mathcal{B}$ updates **H_1List** with a new tuple $<ID_\theta, P_\theta, C_1, C_0, h_1>$, **$H_2$List** with a new tuple $<ID_\theta, U_\theta, P_\theta, h_2>$, and **$H_4$List** with a tuple $<U_\theta, P_\theta, C_1, C_0, h_4>$. Finally, $\mathcal{B}$ sends to $\mathcal{A}_\mathrm{I}$ the signcrypted message of m:$C = (C_0, C_1, C_2,)$.
- Otherwise, $\mathcal{B}$ makes a signcrypted message as normal.

$\mathbf{O}^{\text{Designcryption}}$: $\mathcal{A}_\mathrm{I}$ gives $\mathcal{B}$ a tuple $<(C_0, C_1, C_2), ID_\mathrm{S}, ID_\mathrm{R}>$.

- If $ID_\mathrm{R} = ID_\theta$, $\mathcal{B}$ runs **H_1-queries** to obtain a tuple $<ID_\mathrm{R}, U_\mathrm{R}, P_\mathrm{R}>$. $\mathcal{B}$ randomly chooses $M \in \{0,1\}^n$. If (k, h_3) is on **H_3List**, (U_S, P_S, C_1, C_0) is on **H_4List**, and (ID_S, P_S, C_1, C_0) is on **H_1List** such that $C_1 = h_3 \oplus M$ and $g^{C_2} = P_\mathrm{S} g_1^{h_2} U_\mathrm{S}^{h_4} C_0^{h_1}$, then $\mathcal{B}$ outputs M as an answer for query $\mathcal{A}_\mathrm{I}$.
- Otherwise, $\mathcal{B}$ operates as normal.

Challenge: $\mathcal{A}_\mathrm{I}$ outputs two messages (M_0, M_1) together with $(ID^*_\mathrm{S}, ID^*_\mathrm{R})$. If $ID^*_\mathrm{R} \neq ID_\theta$, $\mathcal{B}$ aborts the game. Otherwise, $\mathcal{B}$ runs $\mathbf{O}^{\text{CreateUser}}$ for ID^*_S and ID^*_R to obtain two tuples $(ID^*_\mathrm{S}, sk^*_\mathrm{S}, pk^*_\mathrm{S}, Cert^*_\mathrm{S})$ and $(ID_\theta, x_\theta, (U_\theta, P_\theta), \perp)$. $\mathcal{B}$ randomly chooses the values $C^*_2 \in \mathbb{Z}^*_{p_2}$ and $\gamma \in (0,1)$, sets $C_0 = g^a$, and runs **H_2-queries** with input $(ID^*_\theta, U^*_\theta, P_\theta)$ to obtain h^*_2. It then computes $k^* = (P_\theta U_\theta g_1^{h^*_2})^a = (g^a)^{x_\theta} (g^a)^{\beta_\theta} (g^{ab})^{h^*_2}$ and $C^*_1 = H_3(k^*) \oplus M_\gamma$. Finally, $\mathcal{B}$ outputs $C^* = (C^*_0, C^*_1, C^*_2)$.

Phase II: $\mathcal{A}_\mathrm{I}$ continues to query as in **Phase I** but with certain restrictions: (1) A query with $< ID^*_\mathrm{R} >$ cannot be submitted to the $\mathbf{O}^{\text{Certification}}$ oracle and (2) a decryption query with $< C^*, ID^*_\mathrm{S}, ID^*_\mathrm{R} >$ cannot be submitted to the $\mathbf{O}^{\text{Designcryption}}$ oracle.

Guess: $\mathcal{A}_I$ outputs guess $\gamma' \in \{0,1\}$ for γ and sends it to $\mathcal{B}$. The challenger searches in **H_3List** and outputs a guess

$$T = \left(\frac{k}{(g^a)^{x_\theta} (g^b)^{\beta_\theta}} \right)^{\frac{1}{h^*_2}}.$$

In the security proof, challenger $\mathcal{B}$ does not directly use guess γ', which is returned by adversary $\mathcal{A}_\mathrm{I}$, but can compute the value g^{ab}. This event only happens if $\mathcal{B}$ chooses the correct tuple from **H_3List**, where $k = k^*$. Indeed, by replacing k with k^*, we have

$$T = \left(\frac{k}{(g^a)^{x_\theta} (g^b)^{\beta_\theta}} \right)^{\frac{1}{h^*_2}} = \left(\frac{(P_\theta U_\theta g_1^{h^*_2})^a}{(g^a)^{x_\theta} (g^b)^{\beta_\theta}} \right)^{\frac{1}{h^*_2}}$$

$$= \left(\frac{(g^a)^{x_\theta} (g^a)^{\beta_\theta} (g^{ab})^{h^*_2}}{(g^a)^{x_\theta} (g^b)^{\beta_\theta}} \right)^{\frac{1}{h^*_2}} = g^{ab}.$$

Security analysis

The simulation will be successful if any of the following events occur:

E_1: $\mathcal{A}_\mathrm{I}$ chooses $ID^*_\mathrm{R} = ID_\theta$. This event will occur with the following probability: $1/q_{\mathrm{cu}}$.

E_2: $\mathcal{A}_\mathrm{I}$ does not query $\mathbf{O}^{\text{RequestPrivateKey}}$ on identity ID_θ. This event will occur with the following probability: $1 - (1/q_{\mathrm{cu}})$.

E_3: $\mathcal{A}_\mathrm{I}$ does not query $\mathbf{O}^{\text{Certification}}$ for identity ID_θ. This event will happen with the following probability: $1 - (1/q_{\mathrm{cu}})$.

E_4: $\mathcal{B}$ does not abort answer $\mathcal{A}_\mathrm{I}$ in a $\mathbf{O}^{\text{Signcryption}}$ query because of collisions in H_1, H_2, H_4. This event will happen with the following probability: $(1 - q_{\mathrm{sc}}[q_1 + q_2 + q_4 + 3q_{\mathrm{sc}}]/2^k)$.

E_5: $\mathcal{B}$ does not reject any valid ciphertext at certain points of the game. This event will occur with the following probability: $(1 - q_{\mathrm{dsc}}/2^k)$.

We define E as the probability that the simulation will be successful. We know that E_1 implies E_2 and E_3. Therefore, we have the following:

$$Pr[E] = Pr[E_1 \cap E_2 \cap E_3 \cap E_4 \cap E_5] \geq$$

$$\frac{1}{q_{\text{cu}}}\left(1 - q_{\text{sc}}\frac{q_1 + q_2 + q_5 + 3q_{\text{sc}}}{2^k}\right)\left(1 - \frac{q_{\text{dsc}}}{2^k}\right).$$

Because $\mathcal{B}$ selects the correct tuple from **H_3List** with probability $(1/q_3)$, the advantage of $\mathcal{B}$ in solving the CDH problem is

$$\epsilon' \geq \frac{\epsilon}{q_{\text{cu}}q_3}(1 - q_{\text{sc}}\frac{q_1 + q_2 + q_5 + 3q_{\text{sc}}}{2^k})(1 - \frac{q_{\text{dsc}}}{2^k}).$$

In addition, the running time is $\tau' \leq \tau + (2q_{\text{cu}} + 5q_{\text{sc}} + 3q_{\text{dsc}})t_{\exp}$, where $t_{\exp}$ denotes the time for an exponentiation. ■

Lemma 2: Suppose that H_1, H_2, H_3, H_4 are random oracles and $\mathcal{A}_{\text{II}}$ is an IND-CBSC-CCA2 Type II adversary that has advantage ϵ and running time τ against the CBSC scheme above. Here, $\mathcal{A}_{\text{II}}$ is allowed to make at most q_{cu} queries to oracle $\mathbf{O}^{\textbf{CreateUser}}$, q_{pri} queries to oracle $\mathbf{O}^{\textbf{RequestPrivateKey}}$, q_{sc} queries to oracle $\mathbf{O}^{\textbf{Signcryption}}$, q_{dsc} queries to oracle $\mathbf{O}^{\textbf{Designcryption}}$, and q_i queries to the random oracle H_i ($i = 1, 2, 3, 4$). Algorithm A_{CDH} exists to solve the CDH problem with the following advantage:

$$\epsilon' \geq \frac{\epsilon}{q_{\text{cu}}q_3}\left(1 - q_{\text{sc}}\frac{q_1 + q_2 + q_4 + 3q_{\text{sc}}}{2^k}\right)\left(1 - \frac{q_{\text{dsc}}}{2^k}\right),$$

and the running time $\tau' \leq \tau + (2q_{\text{cu}} + 5q_{\text{sc}} + 3q_{\text{dsc}})t_{\exp}$, where $t_{\exp}$ denotes the time for an exponentiation.

Proof: We construct an algorithm $\mathcal{B}$ that solves the CDH problem by using $\mathcal{A}_{\text{II}}$. Here, $\mathcal{B}$ is given an instance of the CDH problem, p, q, g, g^a, g^b and will set g^a as a part of the challenge ciphertext and use g^b to generate a public key associated with the challenger identity. $\mathcal{B}$ simulates a challenger and answers queries from $\mathcal{A}_{\text{II}}$ as below:

Setup: $\mathcal{B}$ randomly chooses $\alpha \in \mathbb{Z}_p^*$ and $\theta \in [1, q_{\text{cu}}]$, and sets $g_1 = g^a$. The *params* are set as $(p_1, p_2, g, g_1, H_1, H_2, H_3, H_4)$. Then, *params* and $msk = \alpha$ are given to $\mathcal{A}_{\text{II}}$. Here, $\mathcal{A}_{\text{II}}$ can query the oracles **H_1-queries**, **H_2-queries**, **H_3-queries**, and **H_4-queries**, which are described in the proof of Lemma 1 at any time during the attack.

Phase I

$\mathbf{O}^{\textbf{CreateUser}}$**:** $\mathcal{B}$ maintains a user list *UserList* : $<ID_i, sk_i, pk_i, Cert_i>$. Upon receiving ID_i, the following occurs:

- If $i = \theta$, then $\mathcal{B}$ randomly chooses $\beta_\theta \in \mathbb{Z}_{p_2}^*$ and $x_\theta \in \mathbb{Z}_{p_2}^*$, sets $U_\theta = g^{x_\theta}$, computes $P_\theta = (g^b)^{\beta_\theta}$, inserts $< ID_{\theta_A}, x_\theta, (U_\theta, P_\theta), \perp>$ into the *UserList*, and responds to $\mathcal{A}_{\text{II}}$ with (U_θ, P_θ).
- Otherwise, $\mathcal{B}$ generates sk_i, pk_i as normal.

$\mathbf{O}^{\textbf{RequestPrivateKey}}$**:** Upon inputting identity ID_i, if $i = \theta$, $\mathcal{B}$ aborts the game. Otherwise, $\mathcal{B}$ searches for sk_i in the *UserList* and responds to $\mathcal{A}_{\text{II}}$ with sk_i.

$\mathbf{O}^{\textbf{Signcryption}}$**:** $\mathcal{A}_{\text{II}}$ gives $\mathcal{B}$ a tuple $<m, ID_S, ID_R>$. There are two cases:

- If $ID_S = ID_\theta$, $\mathcal{B}$ randomly chooses $C_1 \in \{0,1\}^n$, $C_2 \in \mathbb{Z}_{p_2}^*$, and $(h_1, h_2, h_4) \in \mathbb{Z}_{p_2}^*$. A simulation is run for $\mathbf{O}^{\textbf{CreateUser}}$ to obtain U_θ, and $C_0 = g^{C_2} / \left((g^b)^{\beta_\theta} g_1^{h_2} U_\theta^{h_4}\right)$ is computed. Next, $\mathcal{B}$ updates **H_1List** with a new tuple $<ID_\theta, P_\theta, C_1, C_0, h_1>$, **$H_2$List** with a new tuple $<ID_\theta, U_\theta, P_\theta, h_2>$, and **$H_4$List** with a tuple $<U_\theta, P_\theta, C_1, C_0, h_4>$. Finally, $\mathcal{B}$ sends the signcryption result $C = (C_0, C_1, C_2)$ to $\mathcal{A}_{\text{II}}$.
- Otherwise, $\mathcal{B}$ makes a signcrypted message as normal.

$\mathbf{O}^{\textbf{Designcryption}}$**:** $\mathcal{A}_{\text{II}}$ gives $\mathcal{B}$ a tuple $<(C_0, C_1, C_2), ID_S, ID_R>$.

- If $ID_R = ID_\theta$, $\mathcal{B}$ runs **H_1-queries** to obtain a tuple $<ID_R, U_R, P_R>$. $\mathcal{B}$ randomly chooses $M \in \{0,1\}^n$. If (k, h_3) is in **H_3List**, (U_S, P_S, C_1, C_0) is in **H_4List**, and (ID_S, P_S, C_1, C_0) is in **H_1List** such that $C_1 = h_3 \oplus M$ and $g^{C_2} = P_S g_1^{h_2} U_S^{h_4} C_0^{h_1}$, then $\mathcal{B}$ outputs M as an answer to the $\mathcal{A}_{\text{II}}$ query.
- Otherwise, $\mathcal{B}$ operates normally.

Challenge: $\mathcal{A}_{\text{II}}$ outputs two messages (M_0, M_1) together with (ID_S^*, ID_R^*). If $ID_R^* \neq ID_\theta$, $\mathcal{B}$ aborts the game. Otherwise, $\mathcal{B}$ runs $\mathbf{O}^{\textbf{CreateUser}}$ for ID_S^* and ID_R^* to obtain two tuples $(ID_S^*, sk_S^*, pk_S^*, Cert_S^*)$ and $(ID_\theta, x_\theta, (U_\theta, P_\theta), \perp)$. Here, $\mathcal{B}$ picks the values $C_2^* \in \mathbb{Z}_{p_2}^*$ and $\gamma \in (0,1)$ randomly, sets $C_0^* = g^a$, and runs **H_2-queries** with input $(ID_\theta^*, U_\theta^*, P_\theta)$ to obtain h_2^*. It computes $k^* = (P_\theta U_\theta g_1^{h_2^*})^a = (g^a)^{x_\theta} (g^{ab})^{\beta_\theta} (g^a)^{\alpha h_2^*}$ and $C_1^* = H_3(k^*) \oplus M_\gamma$. Finally, $\mathcal{B}$ outputs $C^* = (C_0^*, C_1^*, C_2^*)$.

Phase II

$\mathcal{A}_{\text{II}}$ continues to query as in **Phase I** but with some restrictions: (1) A query with $< ID_R^* >$ cannot be submitted to $\mathbf{O}^{\textbf{Certification}}$ oracle, and (2) a decryption query with $< C^*, ID_S^*, ID_R^* >$ cannot be submitted to $\mathbf{O}^{\textbf{Designcryption}}$ oracle.

Guess: $\mathcal{A}_{\text{II}}$ outputs guess $\gamma' \in \{0,1\}$ for γ and sends it to $\mathcal{B}$. The challenger searches **H_3List** and outputs a guess $T = \left[k/\left((g^a)^{x_\theta} (g^a)^{\alpha h_2^*}\right)\right]^{1/\beta_\theta}$. If challenger $\mathcal{B}$ chooses the correct tuple from **H_3List**, then $k = k^*$. In this case, we have the following: $T = g^{ab}$.

Security analysis

The simulation will be successful if any of the following events occur:

E_1: $\mathcal{A}_{\text{II}}$ chooses $ID_R^* = ID_\theta$. This event will occur with the following probability: $1/q_{\text{cu}}$.

E_2: $\mathcal{A}_{\text{II}}$ does not query $\mathbf{O}^{\textbf{RequestPrivateKey}}$ on the identity ID_θ. This event will occur with the following probability: $1 - (1/q_{\text{cu}})$.

http://dx.doi.org/10.4218/etrij.16.0115.0983

E_3: $\mathcal{A}_{\text{II}}$ does not query $\mathbf{O}^{\text{Certification}}$ for the identity ID_θ. This event will happen with the following probability: $1 - (1/q_{\text{cu}})$.
E_4: $\mathcal{B}$ does not abort answer $\mathcal{A}_{\text{II}}$ in query $\mathbf{O}^{\text{Signcryption}}$ because of collisions in H_1, H_2, H_4. This event will occur with the following probability: $(1 - q_{\text{sc}}[q_1 + q_2 + q_4 + 3q_{\text{sc}}]/2^k)$.
E_5: $\mathcal{B}$ does not reject any valid ciphertext at certain points of the game. This event will happen with the following probability: $(1 - q_{\text{dsc}}/2^k)$.

We define E as the probability that the simulation will be successful. We know that E_1 implies E_2 and E_3. Therefore, we have

$$\begin{aligned} Pr[E] &= Pr[E_1 \cap E_2 \cap E_3 \cap E_4 \cap E_5] \\ &\geq \frac{1}{q_{\text{cu}}}\left(1 - q_{\text{sc}}\frac{q_1 + q_2 + q_5 + 3q_{\text{sc}}}{2^k}\right)\left(1 - \frac{q_{\text{dsc}}}{2^k}\right). \end{aligned}$$

Because $\mathcal{B}$ selects the correct tuple from **H$_3$List** with the probability $(1/q_3)$, the advantage of $\mathcal{B}$ in solving the CDH problem is

$$\epsilon' \geq \frac{\epsilon}{q_{\text{cu}} q_3}\left(1 - q_{\text{sc}}\frac{q_1 + q_2 + q_4 + 3q_{\text{sc}}}{2^k}\right)\left(1 - \frac{q_{\text{dsc}}}{2^k}\right).$$

In addition, the running time is $\tau' \leq \tau + (2q_{\text{cu}} + 5q_{\text{sc}} + 3q_{\text{dsc}})t_{\exp}$, where $t_{\exp}$ denotes the time for an exponentiation. ■

To prove the unforgeability of the proposed scheme, we prove the following theorem.

Theorem 2: Suppose the DL problem is intractable. The CBSC scheme above is EUF-CBSC-CMA secure in the random oracle model.

The theorem can be proved through the following two lemmas: Lemma 3 for a Type I adversary, and Lemma 4 for a Type II adversary.

Lemma 3: Suppose that H_1, H_2, H_3, H_4 are random oracles and $\mathcal{A}_{\text{I}}$ is a EUF-CBSC-CMA Type I adversary that has advantage ϵ and running time τ against the proposed scheme. Here, $\mathcal{A}_{\text{I}}$ is allowed to make at most q_{cu} queries to oracle $\mathbf{O}^{\text{CreateUser}}$, q_{pri} queries to oracle $\mathbf{O}^{\text{RequestPrivateKey}}$, q_{cer} queries to oracle $\mathbf{O}^{\text{Certification}}$, q_{sc} queries to oracle $\mathbf{O}^{\text{Signcryption}}$, q_{dsc} queries to oracle $\mathbf{O}^{\text{Designcryption}}$, and q_i queries to random oracle H_i (i = 1, 2, 3, 4). Algorithm A_{DL} can then be used to solve the DL problem with the following advantage:

$$\epsilon' \geq \frac{\epsilon}{q_{\text{cu}}}\left(1 - q_{\text{sc}}\frac{q_1 + q_2 + q_4 + 3q_{\text{sc}}}{2^k}\right)\left(1 - \frac{q_{\text{dsc}}}{2^k}\right),$$

and the running time is $\tau' \leq \tau + (2q_{\text{cu}} + 5q_{\text{sc}} + 3q_{\text{dsc}})t_{\exp}$, where $t_{\exp}$ denotes the time for an exponentiation.

Proof: We construct an algorithm $\mathcal{B}$ that solves the DL problem by using $\mathcal{A}_{\text{I}}$. $\mathcal{B}$ is given an instance of the DL problem, p, q, g, g^a, simulates a challenger, and answers queries from $\mathcal{A}_{\text{I}}$ as follows:

$\mathcal{B}$ randomly chooses $\theta \in [1, q_{\text{cu}}]$ and sets $g_1 = g^a$. The *params* are set as (p_1, p_2, g, g_1, H_1, H_2, H_3, H_4), and are given to $\mathcal{A}_{\text{I}}$. Here, $\mathcal{A}_{\text{I}}$ is allowed to make queries to four random oracles as in IND-CBSC-CCA2 Game I: **H$_1$-queries**, **H$_2$-queries**, **H$_3$-queries**, and **H$_4$-queries**. In addition, $\mathcal{A}_I$ also makes queries to oracles $\mathbf{O}^{\text{CreateUser}}$, $\mathbf{O}^{\text{RequestPrivateKey}}$, $\mathbf{O}^{\text{Certification}}$, $\mathbf{O}^{\text{Signcryption}}$, and $\mathbf{O}^{\text{Designcryption}}$ as in IND-CBSC-CCA2 game I.

Forge: $\mathcal{A}_{\text{I}}$ outputs a forgery signcryption $C^* = (C_0^*, C_1^*, C_2^*)$ of message M^* together with $(ID_{\text{S}}^*, ID_{\text{R}}^*)$, which is not produced by signcrypt query $\mathbf{O}^{\text{Signcryption}}$, and ID_{S}^* is not submitted to certification query $\mathbf{O}^{\text{Certification}}$. If $ID_{\text{S}}^* \neq ID_\theta$, $\mathcal{B}$ aborts the game. Otherwise, by applying the forking lemma [41], $\mathcal{B}$ replays $\mathcal{A}_{\text{I}}$ with the same tape but a different choice of hash function H_2. In addition, $\mathcal{B}$ can obtain another valid signature, $C' = (C_0^*, C_1^*, C_2')$, where $g^{C_2'} = P_{\text{S}}^* g_1^{h_{2'}} U_{\text{S}}^{*h_4^*} C_0^{*h_1^*}$. Because C^* is a forged signcryption, we have $g^{C_2^*} = P_{\text{S}}^* g_1^{h_2^*} U_{\text{S}}^{*h_4^*} C_0^{*h_1^*}$. From these two equations and by replacing $g_1 = g^a$, $\mathcal{B}$ computes $a = (C_2^* - C_2')/(h_2^* - h_{2'})$.

Security analysis

The simulation will succeed if the following events hold:
E_1: $\mathcal{B}$ does not abort when answering all oracle queries.
E_2: $\mathcal{A}_{\text{I}}$ outputs a forgery with $ID_{\text{R}} = ID_\theta$.

Through the same security analysis of IND-CBSC-CCA2 game I, we have

$$\begin{aligned} &Pr[E] \\ &= Pr[E_1 \cap E_2] \geq \frac{1}{q_{\text{cu}}}\left(1 - q_{\text{sc}}\frac{q_1 + q_2 + q_4 + 3q_{\text{sc}}}{2^k}\right)\left(1 - \frac{q_{\text{dsc}}}{2^k}\right). \end{aligned}$$

Therefore, the advantage of $\mathcal{B}$ in solving the DL problem is

$$\epsilon' \geq \frac{\epsilon}{q_{\text{cu}}}\left(1 - q_{\text{sc}}\frac{q_1 + q_2 + q_4 + 3q_{\text{sc}}}{2^k}\right)\left(1 - \frac{q_{\text{dsc}}}{2^k}\right),$$

and the running time is $\tau' \leq \tau + (2q_{\text{cu}} + 5q_{\text{sc}} + 3q_{\text{dsc}})t_{\exp}$, where $t_{\exp}$ denotes the time for an exponentiation. ■

Lemma 4: Suppose that H_1, H_2, H_3, H_4 are random oracles, and $\mathcal{A}_{\text{II}}$ is an EUF-CBSC-CMA Type II adversary that has advantage ϵ and running time τ against the proposed scheme. Here, $\mathcal{A}_{\text{II}}$ is allowed to make at most q_{cu} queries to oracle $\mathbf{O}^{\text{CreateUser}}$, q_{pri} queries to oracle $\mathbf{O}^{\text{RequestPrivateKey}}$, q_{sc} queries to oracle $\mathbf{O}^{\text{Signcryption}}$, q_{dsc} queries to oracle $\mathbf{O}^{\text{Designcryption}}$, and q_i queries to random oracle H_i (i = 1, 2, 3, 4). Algorithm A_{DL} is then used to solve the DLP problem with the following advantage:

$$\epsilon' \geq \frac{\epsilon}{q_{\text{cu}}}\left(1 - q_{\text{sc}}\frac{q_1 + q_2 + q_4 + 3q_{\text{sc}}}{2^k}\right)\left(1 - \frac{q_{\text{dsc}}}{2^k}\right),$$

and the running time is $\tau' \leq \tau + (2q_{\text{cu}} + 5q_{\text{sc}} + 3q_{\text{dsc}})t_{\exp}$, where $t_{\exp}$ denotes the time for an exponentiation.

Proof: We construct an algorithm $\mathcal{B}$ that solves the DPL problem by using $\mathcal{A}_{\text{II}}$, and $\mathcal{B}$ is given an instance of the DL problem, p, q, g, g^a, and $\mathcal{B}$ simulates a challenger and answers queries from $\mathcal{A}_{\text{II}}$ as follows:

In addition $\mathcal{B}$ randomly chooses $\theta \in [1, q_{\text{cu}}]$ and $\alpha \in \mathbb{Z}^*_{p_2}$, and sets $g_1 = g^a$. The *params* are set as (p_1, p_2, g, g_1, H_1, H_2, H_3, H_4), and *msk* is set as α. Then, *params* and *msk* are given to $\mathcal{A}_{\text{II}}$, which is allowed to make queries to fiour random oracles as in IND-CBSC-CCA2 game II: **H_1-queries**, **H_2-queries**, **H_3-queries**, and **H_4-queries**. When $\mathcal{A}_{\text{II}}$ queries oracle $\mathbf{O}^{\text{CreateUser}}$, $\mathcal{B}$ interacts as follows:

$\mathbf{O}^{\text{CreateUser}}$: Here $\mathcal{B}$ maintains a user list *UserList*: <ID_i, sk_i, pk_i, $Cert_i$>. Upon receiving ID_i the following occurs:

- If $i = \theta$, then $\mathcal{B}$ randomly chooses $\beta_\theta \in \mathbb{Z}^*_{p_2}$, and sets $U_\theta = g^a$. It computes $P_\theta = g^{\beta_\theta}$, inserts $< ID_{\theta_A}, \perp, (U_\theta, P_\theta) >$ to the *UserList*, and responds to $\mathcal{A}_{\text{II}}$ with (U_θ, P_θ).
- Otherwise, $\mathcal{B}$ generates sk_i, pk_i, and $Cert_i$ as normal.

In addition $\mathcal{A}_{\text{II}}$ can make queries to oracles $\mathbf{O}^{\text{RequestPrivateKey}}$, $\mathbf{O}^{\text{Signcryption}}$, and $\mathbf{O}^{\text{Designcryption}}$, as in IND-CBSC-CCA2 game II.

Forge: Here $\mathcal{A}_{\text{II}}$ outputs a forgery signcryption $C^* = (C_0^*, C_1^*, C_2^*)$ of message M^* together with $(ID_{\text{S}}^*, ID_{\text{R}}^*)$, which is not produced by querying to oracle $\mathbf{O}^{\text{Signcryption}}$, and ID_{S}^* is not submitted to oracle $\mathbf{O}^{\text{CreateUser}}$. If $ID_{\text{S}}^* \neq ID_\theta$, $\mathcal{B}$ aborts the game. Otherwise, by applying the forking lemma [42], $\mathcal{B}$ replays $\mathcal{A}_{\text{II}}$ with the same tape but a different choice of hash function H_2. Here, $\mathcal{B}$ can obtain another valid signature $C' = (C_0^*, C_1^*, C_2')$, where $g^{C_2'} = P_S^* g_1^{h_2^*} U_S^{*h_{4'}} C_0^{*h_1^*}$. Because C^* is a forged signcryption, we have $g^{C_2^*} = P_S^* g_1^{h_2^*} U_S^{*h_4^*} C_0^{*h_1^*}$. From these two equations and by replacing $U_{\text{S}} = g^a$, $\mathcal{B}$ computes $a = (C_2^* - C_2')/(h_4^* - h_{4'})$.

Security analysis

The simulation will succeed if the following events hold:

E_1: $\mathcal{B}$ does not abort when answering all of the oracle queries.

E_2: $\mathcal{A}_{\text{II}}$ outputs the forgery with $ID_{\text{R}} = ID_\theta$.

Through the same security analysis of IND-CBSC-CCA2 game II, we have

$$Pr[E]$$
$$= Pr[E_1 \cap E_2] \geq \frac{1}{q_{\text{cu}}}\left(1 - q_{\text{sc}}\frac{q_1 + q_2 + q_4 + 3q_{\text{sc}}}{2^k}\right)\left(1 - \frac{q_{\text{dsc}}}{2^k}\right).$$

Therefore, the advantage of $\mathcal{B}$ in solving the DL problem is

$$\epsilon' \geq \frac{\epsilon}{q_{\text{cu}}}\left(1 - q_{\text{sc}}\frac{q_1 + q_2 + q_4 + 3q_{\text{sc}}}{2^k}\right)\left(1 - \frac{q_{\text{dsc}}}{2^k}\right),$$

and the running time is $\tau' \leq \tau + (2q_{\text{cu}} + 5q_{\text{sc}} + 3q_{\text{dsc}})t_{\exp}$, where $t_{\exp}$ denotes the time for an exponentiation. ■

IV. Performance Comparison

Table 1 shows a comparison between our scheme and other CBSC schemes. In this table, m, e, and p denote multiplication, exponentiation, and pairing, respectively. The overheads of the hash and mathematical operations are very small compared to those of the exponentiation and pairing operations, and thus we ignored them in our comparison. Overall, the performance of our scheme is slightly better than the scheme in [40], which is another CBSC scheme without pairing. The difference that makes our scheme more efficient is that we can separate the decryption and verification functions. The verification function can be shared with a server, which has a strong computational capability. From Table 1, we can see that, in most cases, the verification function requires more computational cost compared to the decryption function. When we allow another party to take care of the verification function, our scheme requires only one exponentiation for the decryption function. Note that it is impossible for other schemes to apply the same separation because they require decryption prior to verification.

Table 1. Performance comparison.

Scheme	Signcrypt	Decrypt	Verify
[37]	$1p + 5m$	$1p + 1m$	$3p + 1e + 2m$
[38]	$3p + 4e$	$1p + 1e$	$2p + 3e$
[40]	$2e + 3m$	$2p + 1e$	$1p + 1e + 2m$
[39]	$4e + 4m$	$4e + 3m$	0
Our CBSC	$3e + 4m$	$1e$	$4e + 2m$

Table 2. Cost of the basic operations in relation to that of elliptic curve scalar multiplication [42].

Operation	Notation	Cost
Bilinear pairing	p	150
Scalar multiplication	m	1
Exponentiation	e	36

Table 3. Performance comparison for scalar multiplication on an elliptic curve.

Scheme	Signcrypt	Decrypt	Verify
[37]	155	151	488
[38]	447	186	372
[40]	75	336	188
[39]	148	147	0
Our CBSC	112	36	147

ETRI Journal, Volume 38, Number 4, August 2016
http://dx.doi.org/10.4218/etrij.16.0115.0983

In Table 1, the scheme in [39] requires one more exponentiation for signcryption compared to ours, and the decryption of our scheme requires only one exponentiation, as compared to $(4e + 3m)$ of the scheme in [39]. Our scheme can be more efficient if the values $g_1^{H_2(ID_R, U_R, P_R)}$ and $g_1^{H_2(ID_S, U_S, P_S)}$ are pre-computed because these values are independent of the signcrypted message. In this case, signcryption requires only $(2e + 4m)$, and verification requires only $(3e + 3m)$.

We use Table 2 from [42], where one unit = one scalar multiplication on MNT curves with 80-bit security. By combining Table 2 with Table 1, we obtain Table 3, which gives us a clearer view of the performance levels of the existing CBSC schemes.

From Table 3, we can see that the scheme in [40] is slightly more efficient than ours in terms of signcryption, but requires significantly more computations than ours for decryption and verification. Overall, our scheme can be considered one of the most efficient CBSC schemes at the present time.

V. Conclusion

In this paper, we proposed an efficient CBSC scheme without pairing, and proved it to be both IND-CBSC- CCA2 and EUF-CBSC-CMA secure in the random oracle model. We compared our scheme with other CBSC schemes, and showed that it is currently one of the most efficient CBSC schemes. In addition, our scheme has a new interesting feature, that is, the direct verification of a signcrypted message using public keys.

References

[1] A. Shamir, "Identity-Based Cryptosystems and Signature Schemes," *Adv. cryptology*, vol. 196, 1984, pp. 47–53.

[2] S.S. Al-Riyami and K.G. Paterson, "Certificateless Public Key Cryptography," *Int. Conf. Theory Appl. Cryptology Inform. Security*, Taipei, Taiwan, Nov.30–Dec. 3, 2003, pp. 452–473.

[3] C. Gentry, "Certificate-Based Encryption and the Certificate Revocation Problem," *Int. Conf. Theory Appl. Cryptographic Techn.*, Warsaw, Poland, May 4–8, 2003, pp. 272–293.

[4] Y. Zheng, "Digital Signcryption or How to Achieve Cost (Signature & Encryption) ≪ Cost (Signature) + Cost (Encryption)," *Annu. Int. Cryptolofy Conf.*, Santa Barbara, CA, USA, Aug. 17–21, 1997, pp. 165–179.

[5] W. Diffie and M.E. Hellman, "New Directions in Cryptography," *IEEE Trans. Inform. Theory*, vol. 22, no. 6, 1976, pp. 644–654.

[6] D. Boneh and M. Franklin, "Identity-Based Encryption from the Weil Pairing," *Annu. Int. Cryptology Conf.*, Santa Barbara, CA, USA, Aug. 19–23, 2001, pp. 213–229.

[7] C.I. Fan et al., "Anonymous Multi-receiver Certificate-Based Encryption," *Int. Conf. Cyber-Enabled Distrib. Comput. Knowl. Discovery*, Beijing, China, Oct. 10–12, 2013, pp. 19–26.

[8] J. Hur, C. Park, and S.O. Hwang, "Privacy-Preserving Identity-Based Broadcast Encryption," *Inform. Fusion*, vol. 13, no. 4, Oct. 2012, pp. 296–303.

[9] I. Kim, S.O. Hwang, and S. Kim, "An Efficient Anonymous Identity-Based Broadcast Encryption for Large-Scale Wireless Sensor Networks.," *Ad Hoc Sensor Wireless Netw.*, vol. 14, no. 1–2, 2012, pp. 27–39.

[10] I. Kim and S.O. Hwang, "An Optimal Identity-Based Broadcast Encryption Scheme for Wireless Sensor Networks," *IEICE Trans. Commun.*, vol. E96.B, no. 3, Mar. 2013, pp. 891–895.

[11] F. Bao and R.H. Deng, "A Signcryption Scheme with Signature Directly Verifiable by Public Key," *Int. Workshop Practice Theory Public Key Cryptography*, Yokohama, Japan, Feb. 5–6, 1998, pp. 55–59.

[12] M. Seo and K. Kim, "Electronic Funds Transfer Protocol Using Domain-Verifiable Signcryption Scheme," *Inform. Security Cryptology-ICISC'99*, vol. 1787, 1999, pp. 269–277.

[13] Y. Mu and V. Varadharajan, "Distributed Signcryption," *Int. Conf. Cryptology*, Calcutta, India, Dec. 2000, pp. 155–164.

[14] D. Kwak and S.J. Moon, "Efficient Distributed Signcryption Scheme as Group Signcryption," *Int. Conf. ACNS*, Kunming, China, Oct. 16–19, 2003, pp. 403–417.

[15] X. Boyen, "Multipurpose Identity-Based Signcryption," *Annu. Int. Cryptology Conf.*, Santa Barbara, CA, USA, Aug. 17–21, 2003, pp. 383–399.

[16] J. Baek, R. Steinfeld, and Y. Zheng, "Formal Proofs for the Security of Signcryption," *Int. Workshop Practice Theory Public Key Cryptosystems*, Paris, France, Feb. 12–14, 2002, pp. 80–98.

[17] F. Li, Y. Hu, and C. Zhang, "An Identity-Based Signcryption Scheme for Multi-domain Ad Hoc Networks," *Int. Conf. ACNS*, Nhuhai, China, June 5–8, 2007, pp. 373–384.

[18] B. Zhang and Q. Xu, "An Id-Based Anonymous Signcryption Scheme for Multiple Receivers Secure in the Standard Model," *AST/UCMA/ISA/CAN Conf.*, Miyazaki, Japan, June 23–25, 2010, pp. 15–27.

[19] S. Duan and Z. Cao, "Efficient and Provably Secure Multi-receiver Identity-Based Signcryption," *Australasian Conf. ACISP*, Melbourne, Australia, July 3–5, 2006, pp. 195–206.

[20] I. Kim and S.O. Hwang, "Efficient Identity-Based Broadcast Signcryption Schemes," *Security Commun. Netw.*, vol. 7, no. 5, May 2014, pp. 914–925.

[21] G. Yu et al., "Provable Secure Identity Based Generalized Signcryption Scheme," *Theoretical Comput. Sci.*, vol. 411, no. 40, Sept. 2010, pp. 3614–3624.

[22] S.S.D. Selvi et al., "Efficient and Provably Secure Certificateless Multi-receiver Signcryption," *Int. Conf. ProvSec*, Shanghai, China, Oct. 30–Nov. 1, 2008, pp. 52–67.

[23] S.S. Al-Riyami and K.G. Paterson, "CBE from CL-PKE: A Generic Construction and Efficient Schemes," *Int. Workshop*

Theory Practice Public Key Cryptography, Les Diablerets, Switzerland, Jan. 23–26, 2005, pp. 398–415.
[24] W. Wu et al., "A Provably Secure Construction of Certificate-Based Encryption from Certificateless Encryption," *Comput. J.*, vol. 55, 2012.
[25] F. Li, X. Xin, and Y. Hu, "Efficient Certificate-Based Signcryption Scheme from Bilinear Pairings," *Int. J. Comput. Appl.*, vol. 30, no. 2, 2008, pp. 129–133.
[26] Y. Lu and J. Li, "Constructing Efficient Certificate-Based Encryption Scheme with Pairing in the Standard Model," *IEEE Int. Conf. Inform. Theory Inform. Security*, Beijing, China, Dec. 17–19, 2010, pp. 234–237.
[27] Z. Shao, "Enhanced Certificate-Based Encryption from Pairings," *Comput. Electr. Eng.*, vol. 37, no. 2, Mar. 2011, pp. 136–146.
[28] C. Sur et al., "Certificate-Based Proxy Re-encryption for Public Cloud Storage," *Int. Conf. Innovative Mobile Internet Services Ubiquitous Comput.*, Taichung, Taiwan, July 2013, pp. 159–166.
[29] T. Hyla and J. Pejaś, "Certificate-Based Encryption Scheme with General Access Structure," *Int. Conf. CISIM*, Venice, Italy, Sept. 26–28, 2012, pp. 41–55.
[30] P. Morillo and C. Ràfols, "Certificate-Based Encryption without Random Oracles," *Cryptology ePrint Archive*, Report 2006/12, 2006. https://eprint.iacr.org/2006/012.pdf
[31] J.K. Liu and J. Zhou, "Efficient Certificate-Based Encryption in the Standard Model," *Int. Conf. SCN*, Amalfi, Italy, Sept. 10–12, 2008, pp. 144–155.
[32] D. Galindo, P. Morillo, and C. Ràfols, "Improved Certificate-Based Encryption in the Standard Model," *J. Syst. Softw.*, vol. 81, no. 7, 2008, pp. 1218– 1226.
[33] J. Baek, R. Safavi-Naini, and W. Susilo, "Certificateless Public Key Encryption without Pairing," *Int. Conf. ISC*, Singapore, Sept. 20–23, 2005, pp. 134–148.
[34] J. Lai, W. Kou, and K. Chen, "Self-Generated-Certificate Public Key Encryption without Pairing and Its Application," *Inform. Sci.*, vol. 181, no. 11, June 2011, pp. 2422– 2435.
[35] G. Stephanides, "Short-Key Certificateless Encryption," *Int. Conf. Lightw. Security Privacy: Devices, Protocols Appl.*, Istanbul, Turkey, Mar. 14–15, 2011, pp. 69–75.
[36] D.H. Yum and P.J. Lee, "Generic Construction of Certificateless Encryption," *Int. Conf. ICCSA*, Assisi, Italy, May 14–17, 2004, pp. 802–811.
[37] M. Luo, Y. Wen, and H. Zhao, "A Certificate-Based Signcryption Scheme," *Int. Conf. Comput. Sci. Inform. Technol.*, Singapore, Aug. 29 – Sept. 2, 2008, pp. 17–23.
[38] J. Li et al., "Certificate-Based Signcryption with Enhanced Security Features," *Comput. Math. Appl.*, vol. 64, no. 6, Sept. 2012, pp. 1587–1601.
[39] Y. Lu and J. Li, "Provably Secure Certificate-Based Signcryption Scheme without Pairings," *KSII Trans. Internet Inform. Syst.*, vol. 8, no. 7, 2014, pp. 2554–2571.
[40] Y. Lu and J. Li, "Efficient Certificate-Based Signcryption Secure against Public Key Replacement Attacks and Insider Attacks," *Scientific World J.*, vol. 2014, 2014, p. 295419.
[41] D. Pointcheval and S. Jacques. "Security Proofs for Signature Schemes," *Int. Conf. Theory Appl. Cryptographic Techn.*, Saragossa, Spain, May12–16, 1996, pp. 387–398.
[42] X. Boyen, "The BB1 Identity-Based Cryptosystem: A standard for Encryption and Key Encapsulation," *Submissions IEEE P*, vol. 1363, 2006.

Minh-Ha Le received his BS degree in electronics and telecommunication in 2012 from the Post and Telecommunication Institute of Technology, Ho Chi Minh City, Vietnam. He is currently an MS degree candidate with the Department of Electronics and Computer Engineering of Hongik University, Sejong, Rep. of Korea. His research interests include cryptography and network security.

Seong Oun Hwang received his BS degree in mathematics in 1993 from Seoul National University, Rep. of Korea, his MS degree in computer and communications engineering in 1998 from Pohang University of Science and Technology, Rep. of Korea, and his PhD in computer science from Korea Advanced Institute of Science and Technology, Daejeon, Rep. of Korea. He worked as a software engineer at LG-CNS Systems, Inc. from 1994 to 1996. He worked as a senior researcher at ETRI, Daejeon, Rep. of Korea, from 1998 to 2007. Since 2008, he has been working as an associate professor with the Department of Computer and Information Communications Engineering of Hongik University, Sejong, Rep. of Korea. His research interests include cryptography and cyber security. He is a member of the IEEE.