4-2011

Heterogeneous signcryption with key privacy

Qiong HUANG

Duncan S. WONG

Guomin YANG
Singapore Management University, gmyang@smu.edu.sg

The Computer Journal Advance Access published January 4, 2011

doi:10.1093/comjnl/bxq095

Heterogeneous Signcryption with Key Privacy

QIONG HUANG[1,*], DUNCAN S. WONG[1] AND GUOMIN YANG[2]

[1]*Department of Computer Science, City University of Hong Kong, Hong Kong (S.A.R) China*
[2]*Temasek Laboratories, National University of Singapore, Kent Ridge Singapore*
[*]*Corresponding author: csqhuang@cityu.edu.hk*

A signcryption scheme allows a sender to produce a ciphertext for a receiver so that both confidentiality and non-repudiation can be ensured. It is built to be more efficient and secure, for example, supporting insider security, when compared with the conventional sign-then-encrypt approach. In this paper, we propose a new notion called heterogeneous signcryption in which the sender has an identity-based secret key while the receiver is holding a certificate-based public key pair. Heterogeneous signcryption is suitable for practical scenarios where an identity-based user, who does not have a personal certificate or a public key, wants to communicate securely with a server which has a certificate with its public key. We propose two constructions and show their security under the model we define in the random oracle model. The model we define captures the insider security for both confidentiality and unforgeability. Both of the schemes also support public verifiability and key privacy, that is, an adversary cannot find out who the sender and receiver are from a ciphertext in the insider security model. The second scheme is the most efficient one computationally among all key-privacy-preserving signcryption schemes even when compared with schemes in an identity-based cryptographic setting or certificate-based public key setting.

Keywords: signcryption; key privacy; ciphertext anonymity; identity-based cryptography

Received 18 August 2010; revised 5 October 2010
Handling editor: Jong Hyuk Park

1. INTRODUCTION

Signcryption, introduced by Zheng [1], is a cryptographic primitive targeting to provide confidentiality and unforgeability simultaneously with shorter ciphertext and lower computational cost than the traditional method of signing and encrypting a message separately. It is suitable for applications that require secrecy and non-repudiation for message delivery on resource-constrained devices over low-bandwidth communication channels.

Existing signcryption schemes [2–8] are all *homogeneous*, that is, they are either solely public key based or identity based [9, 10]. A signcryption scheme in the certificate-based public key setting requires both the sender and the receiver to have public key pairs with the public keys certified by a certification authority; while a signcryption scheme in the identity-based setting requires both the sender and the receiver to use their identities as public keys and have their secret keys issued by a key generation center (KGC). There is no signcryption scheme allowing the sender and the receiver to use keys that are under different cryptographic settings. In this paper, we propose a new notion called heterogeneous signcryption, which allows a user (as the sender) to generate a ciphertext (also known as a signcrypted text) so that a server (as the receiver) can decrypt the ciphertext under a conventional private key and verify the recovered message and signature as with using an identity-based signature scheme.

The motivation of considering heterogeneous signcryption stems from practical needs with the fact that a heterogeneous setting is actually common in practice. For example, in a typical webmail or e-banking log-on process, the web server authenticates itself to a user through an secure sockets layer handshake using its certificate-based public key pair, and the user authenticates itself to the server using a password (i.e. symmetric-key setting). In this example, the entire mutual authentication process involves multiple rounds of message flows. Suppose the user has an identity-based user secret key while the server remains the same, that is, holding a certificate-based public key pair. The user can establish an

(implicit) mutually authenticated secure channel with the server by running a one-pass authenticated key exchange (AKE) protocol which can be built directly from a heterogeneous signcryption scheme. This will allow the user to establish a session key with the server by simply sending one signcrypted message to the server. The authenticity of the user is ensured by the unforgeability of the signcryption scheme, and the server authentication is carried out implicitly as only the server is able to obtain the session key [11, 12].

This heterogeneous model has also been suggested to the deployment of identity-based cryptosystems on the internet [13–15] where the public keys of the KGC and the identity-based parameter repository are certified under the conventional public key infrastructure (PKI) and the individual users obtain identity-based user secret keys from the KGC. With heterogeneous signcryption, these users can establish authenticated and secure channels with servers efficiently using the one-pass AKE mentioned above.

The above one-pass AKE protocol constructed from heterogeneous signcryption also fits perfectly the application of two-party secure roaming [16]. In two-party secure roaming, a foreign server is able to perform subscription validation, namely verifying that a mobile user is indeed a legitimate subscriber of another server (called the home server of the mobile user), without contacting the mobile user's home server. This can be achieved by using the above one-pass AKE protocol, that is, having the home server act as the KGC of its subscribers so that the mobile user obtains an identity-based secret key from its home server. Then during secure roaming, the mobile user generates a heterogeneous signcryption on some challenge (e.g. timestamp or some synchronized counter [16, 17]) so that the foreign server is able to verify the signature using the identity-based parameter (i.e. the KGC master public key) of the home server after decrypting the heterogeneous signcryption using its conventional private key.

In this paper, we propose a definition for heterogeneous signcryption and define three security models which capture confidentiality, unforgeability and key privacy (also known as ciphertext anonymity) for a secure heterogeneous signcryption scheme. Besides CCA2 security, the confidentiality model also captures chosen-KGC attacks and multi-KGC setting; the unforgeability model is comparable with the strong unforgeability notion of conventional signature and the key privacy captures the objective of achieving anonymity for both the sender and the receiver. All the models also capture the insider security.

We propose two efficient heterogeneous signcryption schemes and show their security under the models we define in the random oracle model. The first scheme requires the server to carry out bilinear pairing but not the user. Our second scheme further improves the efficiency by removing the bilinear pairing operation from the server side altogether.

In the next section, we review some previous work. In Section 3, we propose a definition and several security models for heterogeneous signcryption. In Section 4, we propose our first scheme, denoted as **Hetero-I**, and show its security. In Section 5, we propose another scheme, denoted as **Hetero-II**, which does not require any pairing operation. In Section 6, we compare these two heterogeneous schemes with the homogeneous ones which are either in the certificate-based public key setting or in the identity-based setting, in terms of computational complexity, ciphertext size as well as security. We conclude the paper in Section 7.

2. RELATED WORK

Since the introduction of signcryption by Zheng [1] in 1997, there have been many signcryption schemes proposed [2–8, 18–22]. They are either in a conventional public key setting or identity-based setting. In [23], Baek, Steinfeld and Zheng defined two security models for the confidentiality and unforgeability of signcryption. They are analogous to the corresponding indistinguishability-based semantic security against an adaptive chosen-ciphertext attack and existential unforgeability against an adaptive chosen-message attack for public key encryption and digital signature, respectively. In [21], An, Dodis and Rabin proposed the notion of *insider security* and showed that both of the generic sequential compositions, namely sign-then-encrypt and encrypt-then-sign, can derive insider secure signcryption schemes. However, these compositions may not have any advantage on reducing the size of ciphertext.

Malone-Lee and Mao proposed an efficient signcryption scheme under the conventional public key setting in [4]. The technique they proposed is similar to the encoding method of OAEP [24], and RSA is used as the underlying one-way trapdoor permutation. In the scheme, a message is 'double wrapped' by the RSA signature and encryption. The resulting signcrypted text (i.e. the ciphertext) has the same size as that of an RSA encryption or that of an RSA signature. Moreover, it supports *public verifiability* [2], which is an 'unwrapping' feature that allows the receiver to retrieve the sender's signature from the signcrypted text for public verification. In [25], Han proposed the notion of generalized signcryption, which includes the functions of encryption, signature and signcryption in a single primitive. This is useful for running on resource-constrained devices. In [22], Li and Wong proposed a generic construction of a signcryption scheme under the conventional public key setting and two instantiations. By making use of a special form of signature as the randomness of a randomness-recoverable encryption, they were able to achieve a smaller ciphertext size when compared with previous ones. The ciphertext size of one of the instantiations is currently the smallest among all the comparable signcryption schemes.

Signcryption also has many variants. In [5], Boyen introduced ciphertext anonymity to signcryption under the identity-based setting. Ciphertext anonymity requires that the ciphertext should

hide the identities of both the sender and the receiver. In [8], Li *et al*. proposed an efficient signcryption scheme with ciphertext anonymity under the conventional public key setting. The security of their scheme has been proved under a model by Libert and Quisquater [7].

The first identity-based signcryption scheme was proposed by Malone-Lee [26]. Libert and Quisquater proposed three more schemes in [6]. None of them supports ciphertext anonymity. The only one under the identity-based setting which has been proved secure while also supporting ciphertext anonymity is by Boyen [5].

Hybrid signcryption [27] is an extension of hybrid encryption which divides the encryption process into two parts, key encapsulation mechanism (KEM) and data encapsulation mechanism (DEM). A symmetric session key and a message-independent ciphertext component can be pre-generated at the KEM part so that a message is encrypted directly using the session key at the DEM part once the message becomes available. The hybrid signcryption is still either a purely conventional public key based or a purely identity-based one.

3. DEFINITION AND SECURITY MODELS

A *heterogeneous signcryption* scheme consists of the following probabilistic polynomial time (PPT) algorithms:

(i) MasterKeyGen: On input of 1^k where $k \in \mathbb{N}$ is a security parameter, it generates the KGC master public/private key pair (mpk, msk).
(ii) UserKeyGen: On input of msk and an identity $\mathsf{ID} \in \{0,1\}^k$, it generates a user secret key usk_{ID}.
(iii) ServerKeyGen: On input of 1^k, it generates a server public/private key pair (PK, SK).
(iv) H-Signcrypt: On input of mpk, a user identity ID, a user secret key usk_{ID}, a server public key PK and a message m, it returns a ciphertext c.
(v) H-Designcrypt: On input of a server private key SK and a ciphertext c, it returns a tuple which consists of a KGC master public key, an identity, a message and a signature, namely $(mpk, \mathsf{ID}, m, \sigma)$, or $\perp$ which indicates the failure of de-signcryption.
(vi) H-Ver: On input of mpk, a user identity ID, a message m and a signature σ, it returns 1/0 indicating a valid or an invalid signature, respectively.

In practice, the KGC performs MasterKeyGen and makes mpk public. The KGC also performs UserKeyGen and issues usk_{ID} to the user whose identity is ID. Every server independently generates its own public key pair (PK, SK) and makes PK public. The correctness requirement is defined as for any $k \in \mathbb{N}$, $(mpk, msk) \leftarrow \mathsf{MasterKeyGen}(1^k)$, $\mathsf{ID} \in \{0,1\}^k$, $usk_{\mathsf{ID}} \leftarrow \mathsf{UserKeyGen}(msk, \mathsf{ID})$, $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{ServerKeyGen}(1^k)$, and $m \in \mathsf{MSPC}(\mathrm{PK})$, we have $(mpk, \mathsf{ID}, m, \sigma) \leftarrow \mathsf{H\text{-}Designcrypt}(\mathrm{SK}, \mathsf{H\text{-}Signcrypt}(mpk, \mathsf{ID}, usk_{\mathsf{ID}}, \mathrm{PK}, m))$ and $1 \leftarrow \mathsf{H\text{-}Ver}(mpk, \mathsf{ID}, m, \sigma)$, where MSPC is the message space defined under a server public key.

In the definition above, we call the sender the user and the receiver the server for the purpose of readability. These terms can be generalized according to the target applications. The user is in the identity-based setting while the server is in the conventional public key setting. Note that H-Designcrypt does not have mpk or ID as input; instead, the server should be able to recover mpk and ID from c using SK. The purpose of this definition is for capturing ciphertext anonymity (i.e. key privacy), that is, the ciphertext c should not leak any information regarding who the sender is and which KGC from the sender has obtained his/her user secret key. In the following, we define three security models for capturing the requirements of confidentiality, unforgeability and ciphertext anonymity.

DEFINITION 3.1 (Confidentiality). *A heterogeneous signcryption scheme is semantically secure against an insider chosen-ciphertext attack (HS-IND-CCA) if no PPT adversary has a non-negligible advantage in the following game:*

(1) *On input of a security parameter $k \in \mathbb{N}$, the challenger runs* ServerKeyGen *to generate a server key pair* (PK, SK) *and gives* PK *to adversary $\mathcal{A}$.*
(2) *$\mathcal{A}$ makes a number of queries to the following oracle:*
   (a) ODesigncrypt*: On input of a ciphertext c, the oracle runs and returns* H-Designcrypt(SK, c) *which is either a tuple in the form $(mpk, \mathsf{ID}, m, \sigma)$ or $\perp$.*
(3) *$\mathcal{A}$ produces and sends to the challenger two plaintexts $m_0, m_1 \in$ MSPC(PK) of equal length, a KGC master public key mpk^*, an identity ID^* and a user secret key $usk^*_{\mathsf{ID}^*}$. The challenger flips a coin $b \overset{R}{\leftarrow} \{0,1\}$, computes $c^* =$ H-Signcrypt$(mpk^*, \mathsf{ID}^*, usk^*_{\mathsf{ID}^*}, \mathrm{PK}, m_b)$ and sends c^* to $\mathcal{A}$ as the challenge ciphertext.*
(4) *$\mathcal{A}$ makes new queries as above, but it cannot query* ODesigncrypt *with c^*.*
(5) *At the end of the game, $\mathcal{A}$ outputs a bit b' and wins if $b' = b$.*

$\mathcal{A}$'s advantage is defined as $Adv^{hs\text{–}ind\text{–}cca}(\mathcal{A}) = \Pr[b' = b] - (1/2)$ and the probability that $b' = b$ is the probability that $\mathcal{A}$ wins the game.

In the rest of the paper, we assume that, for any master (respectively, server) key pair, there is an efficient method of verifying whether it is in the range of the corresponding key generation algorithm, e.g. whether the secret key matches the public key.

This definition captures the advantage of an active adversary over an eavesdropper; that is, the adversary knows and has the full control of all signing keys (and even the KGC master key pairs). This also captures the insider security for confidentiality [5, 7, 21]. Unlike the unforgeability model for conventional signcryption schemes, the model above does not

need a signcryption oracle because $\mathcal{A}$ can always generate one using PK and its self-generated user secret key.

The model also captures *chosen-KGC attacks* and *multi-KGC setting* [28]. In the game, $\mathcal{A}$ is allowed to adaptively choose multiple KGCs for maximizing its advantage. This strong notion is generally not considered in pure identity-based signcryption, where a KGC is fixed by the challenger at the beginning of the confidentiality game and only chosen-ID attacks are considered. As different users may obtain user secret keys from different KGCs in practice, we therefore consider both chosen-KGC and chosen-ID attacks in the multi-KGC setting, where collusion between multiple KGCs can be considered. The scenario is also similar to that in the conventional oligarchy PKI model used on the internet nowadays.

DEFINITION 3.2 (Unforgeability). *A heterogeneous signcryption scheme is existentially unforgeable against chosen-ID and chosen-message insider attack (HS-EUF-ID-CMA) if no PPT forger has a non-negligible advantage in the following game:*

(1) *The challenger runs* MasterKeyGen *to generate the KGC master key pair* (mpk, msk) *and sends* mpk *to forger* $\mathcal{F}$.
(2) $\mathcal{F}$ *makes a number of queries to the following oracles:*
   (a) OCreateUser*: On input of an identity* ID*, if* ID *has not been created, the oracle runs* $usk_{\mathsf{ID}} \leftarrow$ UserKeyGen(msk, ID) *and stores* $(\mathsf{ID}, usk_{\mathsf{ID}})$ *into a list* List[1]*. The oracle returns nothing.*
   (b) ORevealUserKey*: On input of an identity* ID*, the oracle searches* List *for* ID*. If it is not found,* $\perp$ *is returned; otherwise, the corresponding* usk_{ID} *is returned.*
   (c) OSigncrypt*: On input of an identity* ID*, a server public key* PK *and a message* m*, the oracle searches* List *for* ID*. If it is not found,* $\perp$ *is returned; otherwise, it retrieves the corresponding* usk_{ID} *from* List *and returns* $c \leftarrow$ H-Signcrypt$(mpk, \mathsf{ID}, usk_{\mathsf{ID}}, \mathrm{PK}, m)$.
(3) $\mathcal{F}$ *outputs a ciphertext* c^**, a user identity* ID^* *and a server key pair* (PK, SK) *which is in the range of* ServerKeyGen *and wins the game if*
   (a) ID^* *is in* List *but has never been queried to* ORevealUserKey.
   (b) $(mpk, \mathsf{ID}^*, m^*, \sigma^*) \leftarrow$ H-Designcrypt(SK, c^*)
   (c) $1 \leftarrow$ H-Ver$(mpk, \mathsf{ID}^*, m^*, \sigma^*)$
   (d) c^* *is not the output of* OSigncrypt.

In the model above, we allow $\mathcal{F}$ to have the full control of the de-signcryption key pair (i.e. the server key pair) (PK, SK). This captures the notion of insider security for unforgeability. This model also captures a notion similar to strong unforgeability of conventional signature schemes. In the context of signcryption, the model allows $\mathcal{F}$ to query OSigncrypt with m^* with the same identity ID and a different server public key PK or even the *same* server public key PK.

DEFINITION 3.3 (Ciphertext Anonymity/Key Privacy). *A heterogeneous signcryption scheme is ciphertext anonymous against a chosen-ciphertext insider attack (HS-ANON-CCA) if no PPT distinguisher has a non-negligible advantage in the following game:*

(1) *The challenger runs* ServerKeyGen *twice to generate two distinct server key pairs* $(\mathrm{PK}_0, \mathrm{SK}_0)$ *and* $(\mathrm{PK}_1, \mathrm{SK}_1)$ *and gives* PK_0 *and* PK_1 *to distinguisher* $\mathcal{D}$.
(2) $\mathcal{D}$ *makes a number of queries to the following oracle:*
   (a) ODesigncrypt*: On input of a server public key* PK_i *(for* $i = 0$ *or 1) and a ciphertext* c*, the oracle returns* H-Designcrypt(SK_i, c) *which is either a tuple of* $(mpk, \mathsf{ID}, m, \sigma)$ *or* $\perp$.
(3) $\mathcal{D}$ *then produces a message* $m \in$ MSPC$(\mathrm{PK}_0) \cap$ MSPC(PK_1) *and two equal-length sets of KGC master public key, user identity and user secret key, that is,* $\{(mpk_i, \mathsf{ID}_i, usk_{\mathsf{ID}_i})\}_{i=0,1}$.
(4) *The challenger flips two coins* $b, b' \xleftarrow{R} \{0, 1\}$*, then computes a challenge ciphertext as* $c^* \leftarrow$ H-Signcrypt$(mpk_b, \mathsf{ID}_b, usk_{\mathsf{ID}_b}, \mathrm{PK}_{b'}, m)$ *and sends it to* $\mathcal{D}$.
(5) $\mathcal{D}$ *makes a number of new queries as above under the restriction that it cannot query* ODesigncrypt *with* (PK_i, c^*) *for* $i = 0, 1$.
(6) *At the end of the game,* $\mathcal{D}$ *outputs two bits* d, d' *and wins the game if* $(d, d') = (b, b')$.

$\mathcal{D}$*'s advantage is defined as* $Adv^{anon\text{–}cca}(\mathcal{D}) = \Pr[(d, d') = (b, b')] - (1/4)$.

The model above supports insider security in the context of ciphertext anonymity by allowing $\mathcal{D}$ to have full control on signing keys (i.e. user secret keys) as well as the KGC master key pair. As with the confidentiality model, this one also captures chosen-KGC attacks and a multi-KGC setting. The ciphertext anonymity defined above requires that a heterogeneous signcryption scheme should achieve ciphertext anonymity through the security of the server key pair only, which is comparable with the key privacy considered in the conventional public key setting [29].

4. OUR FIRST SCHEME: HETERO-I

In the first heterogeneous signcryption scheme we propose here, the server is required to carry out two pairing operations during de-signcryption while the client is not required to do so.

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two cyclic groups of order q, where q is a k-bit prime and $k \in \mathbb{N}$ is a security parameter. Let g be a generator of $\mathbb{G}$. A bilinear map is defined as $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ such that it is. (1) bilinear: for all $g_1, g_2 \in \mathbb{G}$ and $a, b \in \mathbb{Z}$, we

[1] Initially List is empty. Note that it is shared among all the oracles.

have $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$; (2) non-degenerate: $\hat{e}(g, g) \neq 1$, where 1 is the identity element of $\mathbb{G}_T$; and (3) computable: $\hat{e}(g_1, g_2)$ can be computed efficiently for any $g_1, g_2 \in \mathbb{G}$. Let $H_1 : \{0,1\}^* \to \mathbb{Z}_q$, $H_2 : \{0,1\}^* \to \mathbb{G}$ and $H_3 : \{0,1\}^* \to \{0,1\}^{n+4k}$ be hash functions where n denotes the message length and is in some polynomial of k. For security analysis, all hash functions are viewed as random oracles [30]. Define the message space MSPC to be $\{0,1\}^n$. Without loss of generality, we assume that an element in $\mathbb{G}$ can be represented in a k-bit binary string. **Hetero-I** is described as follows.

MasterKeyGen: On input of 1^k, it randomly chooses $x \in_R \mathbb{Z}_q$ and sets $msk := x$ and $mpk := g^x$.

UserKeyGen: On input of msk and an identity $\mathsf{ID} \in \{0,1\}^k$, it carries out the following steps:

(1) Compute $c' = H_1(mpk, \mathsf{ID}, R)$, where $R = g^r$ and $r \in_R \mathbb{Z}_q$ is randomly chosen.
(2) Compute $s = r + c'x \bmod q$.
The user secret key is $usk_{\mathsf{ID}} := (R, s)$.

ServerKeyGen: On input of 1^k, it randomly chooses $x' \in_R \mathbb{Z}_q$ and sets $\mathrm{SK} := x'$ and $\mathrm{PK} := g^{x'}$.

H-Signcrypt: On input of mpk, ID, $usk_{\mathsf{ID}} = (R, s)$, PK and message $m \in \{0,1\}^n$, the following steps are carried out:

(1) Compute
$V = H_2(mpk, \mathsf{ID}, m, U, \mathrm{PK}, R, \mathrm{PK}^t)^s$,
where $U = g^t$ and $t \in_R \mathbb{Z}_q$.
(2) Compute
$Z = (m \| mpk \| \mathsf{ID} \| R \| V) \oplus H_3(U, \mathrm{PK}, \mathrm{PK}^t)$.
The ciphertext is $c = (U, Z)$.

H-Designcrypt: The following steps are carried out on input of $\mathrm{SK} = x'$ and ciphertext $c = (U, Z)$:

(1) Compute
$(m \| mpk \| \mathsf{ID} \| R \| V) \leftarrow Z \oplus H_3(U, PK, U^{x'})$.
(2) Compute $c' = H_1(mpk, \mathsf{ID}, R)$.
(3) Check if $\hat{e}(V, g) \stackrel{?}{=}$
$\hat{e}(H_2(mpk, \mathsf{ID}, m, U, \mathrm{PK}, R, U^{x'}), R \cdot mpk^{c'})$.
(4) If so, output $(mpk, \mathsf{ID}, m, \sigma)$ where $\sigma = (R, U, \mathrm{PK}, D = U^{x'}, V)$; otherwise, output $\perp$.

H-Ver: On input of mpk, an identity $\mathsf{ID} \in \{0,1\}^k$, a message $m \in \{0,1\}^n$ and a signature $\sigma = (R, U, PK, D, V)$, the following steps are carried out:

(1) Compute $c' = H_1(mpk, \mathsf{ID}, R)$.
(2) Check if $\hat{e}(V, g) \stackrel{?}{=}$
$\hat{e}(H_2(mpk, \mathsf{ID}, m, U, \mathrm{PK}, R, D), R \cdot mpk^{c'})$.
(3) If so, output 1; otherwise, output 0.

The server (i.e. the receiver) can de-signcrypt without knowing who the user (i.e. the sender) is. Besides message m, mpk, ID and σ can all be recovered from the ciphertext c so that the server can verify their authenticity (in Step 4 of H-Designcrypt).

Security Analysis. Below we analyze the scheme under our proposed security models. First of all, we give the underlying number-theoretic assumption used in the proofs.

Computational Diffie–Hellman (CDH) Problem. Given $g^a, g^b \in \mathbb{G}$, where $a, b \in_R \mathbb{Z}_q$, compute g^{ab}.
The CDH assumption says that there is no PPT algorithm which can solve a random instance of the CDH problem with non-negligible probability. The security of our schemes relies on the CDH assumption defined in a *bilinear* group [31].

On the confidentiality of the scheme, we can see that V is generated on the value of D which cannot be computed from U if the server's private key is not known. This is due to the intractability of the CDH problem. Even with the knowledge of the user's secret key usk_{ID}, $\mathcal{A}$ cannot reconstruct V if D is unknown. Here $\mathcal{A}$ needs to know D in order to come up with a valid ciphertext, which renders ODesigncrypt 'useless' and explains why we can get chosen-ciphertext security from just a chosen-plaintext secure ElGamal encryption algorithm. We give more details in the proof of confidentiality.

THEOREM 4.1. *Let $k \in \mathbb{N}$ be a security parameter. Under the random oracle model, if there exists a PPT algorithm that breaks the HS-IND-CCA security (Definition 3.1) of* ***Hetero-I*** *above with advantage at least $\rho(k)$, then there exists a PPT algorithm that solves the CDH Problem with probability at least $2(1 - q_D/q)\rho(k)$, where q_D is the maximum number of* ODesigncrypt *queries made in the game of HS-IND-CCA.*

Proof. Seeking a contradiction, suppose that there exists a PPT $\mathcal{A}$ that wins the game in Definition 3.1 with advantage at least $\rho(k)$. We construct another PPT $\mathcal{B}$ to solve the CDH problem. Suppose $\mathcal{B}$ is given a random CDH problem instance $g^a, g^b \in \mathbb{G}$; then $\mathcal{B}$ sets up a simulated environment of HS-IND-CCA model for $\mathcal{A}$ as follows. $\mathcal{B}$ gives $\mathrm{PK} = g^b$ to $\mathcal{A}$ and simulates H_1 by returning an element chosen uniformly at random from $\mathbb{Z}_q$ for each new query. It maintains a list L1 for ensuring that the same value will be returned for the same query.

$\mathcal{B}$ also maintains two other lists L2 and L3 for H_2 and H_3, respectively. When a hash query $H_2(g_1, str, m, g_2, \mathrm{PK}', g_3, g_4)$ is received, where $str \in \{0,1\}^k$, $m \in \{0,1\}^n$ and $g_1, g_2, g_3, g_4 \in \mathbb{G}$, $\mathcal{B}$ checks if the query tuple $(g_1, str, m, g_2, \mathrm{PK}', g_3, g_4)$ is already in L2. If so, the existing result in L2 is returned. If not, $\mathcal{B}$ randomly chooses $w \leftarrow \mathbb{Z}_q$ and returns g^w to $\mathcal{A}$ while storing the query tuple together with $(g^w, w, \top)$ in L2. Furthermore, if $\hat{e}(g_2, \mathrm{PK}') = \hat{e}(g, g_4)$ and a tuple of the form $(\ldots, g_2, \ldots, \mathrm{PK}', \ldots, \top)$ is in L2, where '$\top$' is a special symbol, then $\mathcal{B}$ replaces '$\top$' in the entry of L2 with g_4. Hash queries to H_3 are handled similarly. An ODesigncrypt query on $c = (U, Z)$ is answered as follows.

(1) $\mathcal{B}$ looks for (U, PK, Λ) in L3 such that $\hat{e}(g, \Lambda) = \hat{e}(U, \text{PK})$ or $\Lambda = \top$.
   (a) If so, the existing return result in L3 will be used as the value of $H_3(U, \text{PK}, \Lambda)$.
   (b) Otherwise, $\mathcal{B}$ adds a new entry into L3 by storing $(U, \text{PK}, \top)$ as the query tuple and a value randomly drawn from the range of H_3 as the oracle return.

(2) $\mathcal{B}$ computes $m\|g_1\|str\|g_2\|g_3 = Z \oplus H_3(U, \text{PK}, \Lambda)$, where $m \in \{0,1\}^n$, $str \in \{0,1\}^k$ and $g_1, g_2, g_3 \in \mathbb{G}$. If any of these domains is not satisfied, $\mathcal{B}$ returns '$\perp$' indicating the invalidity of c. If all the domains are correct, $\mathcal{B}$ then simulates H_2 on $(g_1, str, m, U, \text{PK}, g_2, \Lambda)$ as described above.

(3) $\mathcal{B}$ checks if

$$\hat{e}(g, g_3) = \hat{e}(H_2(g_1, str, m, U, \text{PK}, g_2, \Lambda), g_2 \cdot g_1^{c'}),$$

where c' is the oracle simulation result of $H_1(g_1, str, g_2)$.

   (a) If it holds and $\hat{e}(g, \Lambda) = \hat{e}(U, \text{PK})$, then $(g_1, str, m, (g_2, U, \text{PK}, \Lambda, g_3))$ are returned.
   (b) If it holds but $\Lambda = \top$, then $\mathcal{B}$ halts with failure.
   (c) Otherwise, the symbol '$\perp$' is returned for rejection.

After $\mathcal{A}$ chooses two n-bit plaintexts m_0 and m_1 together with a KGC master public key $mpk^* \in \mathbb{G}$, an identity $\mathsf{ID}^* \in \{0,1\}^k$ and a user secret key $usk^*_{\mathsf{ID}^*} = (R^*, s^*)$, and requests $\mathcal{B}$ for a challenge ciphertext, $\mathcal{B}$ sets the challenge ciphertext to $c^* = (U^*, Z^*)$, where $U^* = g^a$ and Z^* is randomly drawn from $\{0,1\}^{n+4k}$. $\mathcal{B}$ also randomly picks $\check{b} \leftarrow 1/0$ and $t^* \leftarrow \mathbb{Z}_q$, and updates L2 by adding in $(mpk^*, \mathsf{ID}^*, m_{\check{b}}, g^a, \text{PK}, R^*, \top, g^{t^*}, t^*, \top)$, e.g. setting the result of $H_2(mpk^*, \mathsf{ID}^*, m_{\check{b}}, g^a, \text{PK}, \top)$ to g^{t^*}. Note that this entry will only be added in L2 if it is not in L2 yet. Similarly, L3 will also be updated with $(g^a, \text{PK}, \top)$ and the value of $H_3(g^a, \text{PK}, \top)$ will be set to $Z^* \oplus (m_{\check{b}}\|mpk^*\|\mathsf{ID}^*\|R^*\|g^{t^*s^*})$.

After that, $\mathcal{B}$ answers $\mathcal{A}$'s queries as before. If $\mathcal{A}$ queries H_2 or H_3 with $(g^a, \text{PK}, \Lambda^*)$, such that $\hat{e}(g^a, \text{PK}) = \hat{e}(\Lambda^*, g)$, then $\mathcal{B}$ outputs Λ^* and halts. If $\mathcal{A}$ halts without making this query, then $\mathcal{B}$ halts with failure.

The running time of $\mathcal{B}$ is in the polynomial of $\mathcal{A}$'s running time. To see that the simulated game is computationally indistinguishable from a real game, we note that H_1, H_2 and H_3 are simulated perfectly. For ODesigncrypt queries, except the following event, all are carried out perfectly too.

The exceptional event is at Step 4 when $(U, \text{PK}, \top)$ is in L3 and $(g_1, str, m, U, \text{PK}, g_2, \top)$ is in L2, while $\hat{e}(g, g_3) = \hat{e}(H_2(g_1, str, m, U, \text{PK}, g_2, \top), g_2 \cdot g_1^{c'})$. Let $\mathbf{E}_1$ be this event. This event implies that $\mathcal{A}$ has never queried H_2 on $(g_1, str, m, U, \text{PK}, g_2, \Lambda)$ or H_3 on (U, PK, Λ) for some $\Lambda \in \mathbb{G}$ such that $\hat{e}(g, \Lambda) = \hat{e}(U, \text{PK})$. Therefore, we have

$$\Pr[\mathbf{E}_1] \leq q_D/|\mathbb{G}| = q_D/q,$$

where q_D is the maximum number of ODesigncrypt queries made by $\mathcal{A}$. Hence, with probability at least $1 - q_D/q$, $\mathcal{B}$ does not fail and carries out the simulation perfectly.

Let $\mathbf{E}_2$ be the event that $(g^a, \text{PK}, \text{PK}^a)$ is queried on H_2 (together with some other parameters corresponding to the input of H_2) or H_3. $\bar{\mathbf{E}}_2$ denotes the event that $(g^a, \text{PK}, \text{PK}^a)$ is not queried on H_2 or H_3. Note that $\mathcal{B}$ solves the CDH problem instance in event $\mathbf{E}_2$.

Let $V_{\check{b}} = H_2(mpk^*, \mathsf{ID}^*, m_{\check{b}}, g^a, \text{PK}, R^*, \text{PK}^a)^{s^*}$. Then $c^* = (g^a, Z^*)$ is the signcryption of $m_{\check{b}}$ if we have

$$(m_{\check{b}}\|mpk^*\|\mathsf{ID}^*\|R^*\|V_{\check{b}}) = Z^* \oplus H_3(g^a, \text{PK}, \text{PK}^a).$$

In the event $\bar{\mathbf{E}}_2$, $(g^a, \text{PK}, \top)$ is in L3 and $(mpk^*, \mathsf{ID}^*, m_{\check{b}}, g^a, \text{PK}, R^*, \top)$ is in L2, while $\mathcal{A}$ has never queried H_2 or H_3 with the triple $(g^a, \text{PK}, \text{PK}^a)$. In this case, c^* looks random to $\mathcal{A}$. Hence, $\Pr[\mathcal{A} \text{ wins the game}|\bar{\mathbf{E}}_2] = 1/2$. From the proposition, we have

$$\begin{aligned}\Pr[\mathcal{A} \text{ wins the game}] &= \Pr[\mathcal{A} \text{ wins the game} \wedge \mathbf{E}_2] \\ &\quad + \Pr[\mathcal{A} \text{ wins the game} \wedge \bar{\mathbf{E}}_2] \\ &\geq \frac{1}{2} + \rho(k).\end{aligned}$$

Therefore, $\Pr[\mathbf{E}_2] + (1/2)\Pr[\bar{\mathbf{E}}_2] \geq (1/2) + \rho(k)$. Hence, $\Pr[\mathbf{E}_2] \geq 2\rho(k)$ and $\Pr[\mathbf{E}_2 \wedge \mathcal{B} \text{ does not fail}] \geq 2(1 - q_D/q)\rho(k)$. □

THEOREM 4.2. *Let $k \in \mathbb{N}$ be a security parameter. Under the random oracle model, if there exists a PPT algorithm that breaks the HS-EUF-ID-CMA security (Definition 3.2) of* ***Hetero-I*** *with advantage at least $\rho(k)$, then there exists a PPT algorithm that solves the CDH Problem with probability at least $((\rho(k)/q_c) - (q_1 q_c/q))((\rho(k)/q_c q_1) - (q_c/q) - (1/q))$, where q_1 and q_c are the maximum number of H_1 and* OCreateUser *queries made in the game of HS-EUF-ID-CMA.*

Before we give the proof, we first review the General Forking Lemma due to Bellare and Neven [32].

LEMMA 4.1 (General Forking Lemma [32]). *Fix an integer $Q \geq 1$ and a set H of size $h \geq 2$. Let A be a randomized algorithm that on input of $x, h_1, \ldots, h_Q$ returns a pair (J, σ), where $J \in \{0, 1, \ldots, Q\}$ and σ is referred to as a side output. Let IG be a randomized algorithm called the input generator. The accepting probability of A, denoted by acc, is defined as the probability that $J \geq 1$ in the experiment $\langle x \xleftarrow{R} IG; h_1, \ldots, h_Q \xleftarrow{R} H; (J, \sigma) \leftarrow A(x, h_1, \ldots, h_Q)\rangle$. The forking algorithm F_A associated to A is a randomized algorithm that takes input x proceeds as follows:*

Algorithm $F_A(x)$
  Pick coins ρ for A at random
  $h_1, \ldots, h_Q \xleftarrow{R} H$
  $(J, \sigma) \leftarrow A(x, h_1, \ldots, h_Q; \rho)$

If $J = 0$, *then return* $(0, \varepsilon, \varepsilon)$
$h'_1, \ldots, h'_Q \xleftarrow{R} H$
$(J', \sigma') \leftarrow A(x, h_1, \ldots, h_{J-1}, h'_J, \ldots, h'_Q; \rho)$
If $(J = J' \wedge h_J \neq h'_J)$, *then return* $(1, \sigma, \sigma')$; *Else return* $(0, \varepsilon, \varepsilon)$

Let $frk = \Pr[b = 1 : x \xleftarrow{\$} IG; (b, \sigma, \sigma') \leftarrow F_A(x)]$ *then* $frk \geq acc((acc/Q) - (1/h))$.

Proof. Suppose there exists a forger $\mathcal{F}$ that wins the game in Definition 3.2 with probability at least $\rho(k)$, We construct an algorithm $\mathcal{B}$ which solves the CDH problem in $\mathbb{G}$. Suppose $\mathcal{B}$ is given a random instance of the CDH problem $(X = g^a, Y = g^b) \in \mathbb{G}^2$; then $\mathcal{B}$ runs $\mathcal{F}$ as a subroutine to find g^{ab}. $\mathcal{B}$ sets up a simulated HS-EUF-ID-CMA game as follows.

$\mathcal{B}$ gives Y to $\mathcal{F}$ as the master public key mpk, randomly selects a number $1 \leq i \leq q_c$, and simulates OCreateUser as follows: for the jth $(j \neq i)$ OCreateUser query with identity ID_j, $\mathcal{B}$ generates a user secret key as follows: randomly selects $s_j, c_j \in_R \mathbb{Z}_q^2$, sets $R_j \leftarrow g^{s_j}/mpk^{c_j}$, and sets c_j as the value of $H_1(mpk, \mathrm{ID}_j, R_j)$. If there already exists a tuple $(mpk, ID_j, R_j, \cdot)$ in the list L1, then B aborts the game with failure. For the ith OCreateUser query, $\mathcal{B}$ randomly selects $r_i \in_R \mathbb{Z}_q$, computes $R_i \leftarrow g^{r_i}$ and issues an H_1 query with input $(mpk, \mathrm{ID}_i, R_i)$ to get c_i.

When a hash query $H_2(mpk, str, m, g_1, g_2, g_3, g_4)$ is received, where $str \in \{0,1\}^k$, $m \in \{0,1\}^n$ and $g_1, g_2, g_3, g_4 \in \mathbb{G}$, $\mathcal{B}$ checks if the query tuple $(mpk, str, m, g_1, g_2, g_3, g_4)$ is already in L2. If it exists, the existing result in L2 is returned. Otherwise, $\mathcal{B}$ randomly chooses $t \leftarrow \mathbb{Z}_q$ and returns X^t to $\mathcal{F}$; the tuple $(mpk, str, m, g_1, g_2, g_3, g_4, t, X^t)$ is then saved in L2. $\mathcal{B}$ simulates the H_3 oracle as in the previous proof.

For an ORevealUserKey query, if the identity is ID_i, then $\mathcal{B}$ aborts the game with failure, otherwise $\mathcal{B}$ returns the user secret key generated in the OCreateUser query to $\mathcal{F}$.

For an OSigncrypt query on an identity ID, a message m and a receiver's public key PK, all chosen by $\mathcal{F}$, $\mathcal{B}$ first checks if $\mathrm{ID} = \mathrm{ID}_i$. If not, $\mathcal{B}$ follows the H-Signcrypt algorithm to generate the ciphertext and returns it to $\mathcal{F}$. Otherwise (e.g. $\mathrm{ID} = \mathrm{ID}_i$), then $\mathcal{B}$ picks a random $r \leftarrow \mathbb{Z}_q$ and computes $U = g^r$; a random $t' \leftarrow \mathbb{Z}_q$ and sets $g^{t'}$ as the value of $H_2(mpk, \mathrm{ID}, m, U, \mathrm{PK}, R_i, \mathrm{PK}^r)$. After that, $\mathcal{B}$ computes $V = (R_i mpk^{c_i})^{t'}$. $\mathcal{B}$ then simulates H_3 as in the proof of Theorem 4.1 for obtaining $H_3(U, \mathrm{PK}, \mathrm{PK}^r)$, and computes the ciphertext $\sigma = (U, Z)$, where $Z = (m \| mpk \| \mathrm{ID}_i \| R_i \| V) \oplus H_3(U, \mathrm{PK}, \mathrm{PK}^r)$.

When $\mathcal{F}$ outputs a ciphertext σ^*, an identity ID and a receiver's key pair (PK, SK), if $\mathrm{ID} \neq \mathrm{ID}_i$, then $\mathcal{B}$ aborts the game with failure. Otherwise, $\mathcal{B}$ runs the H-Designcrypt algorithm on σ^* and SK, and gets back $m \| mpk \| \mathrm{ID}_i \| R'_i \| V$. If the forgery is valid, which means $\hat{e}(V, g) = \hat{e}(H_2(mpk, \mathrm{ID}_i, m, U, \mathrm{PK}, R'_i, U^{\mathrm{SK}}), R'_i mpk^{c'_i})$, where $c'_i = H_1(mpk, \mathrm{ID}_i, R'_i)$, then $\mathcal{B}$ gets $V = (g^{ta})^{r'_i + c'_i b}$ for some t that is known to $\mathcal{B}$. Note that if $H_2(mpk, \mathrm{ID}_i, m, U, \mathrm{PK}, R'_i, U^{\mathrm{SK}})$ has occurred in an OSigncrypt, then σ^* must have been generated in that OSigncrypt query, which contradicts the restriction to $\mathcal{F}$. So we can guarantee that $H_2(mpk, \mathrm{ID}_i, m, U, \mathrm{PK}, R'_i, U^{\mathrm{SK}})$ must have the form of g^{ta} for some t that is known to $\mathcal{B}$. Then two cases are considered:

(i) $R'_i = R_i$. In this case, $\mathcal{B}$ computes $\lambda = (V^{t^{-1}}/X^{r_i})^{c_i^{-1}} = g^{ab}$ and outputs λ as the solution to the CDH problem. Let $\mathbf{E}$ be the event that $\mathcal{B}$ aborts with failure during the game. The probability that $\mathcal{B}$ successfully solves the CDH problem is $\Pr[\bar{\mathbf{E}} \wedge \mathcal{F}$ succeeds$] \geq (\rho(k)/q_c) - (q_1 q_c/q)$, where the factor $(q_1 q_c/q)$ is the upper bound of a collision which would occur in patching H_1 in the simulation of oracle OCreateUser.

(ii) $R'_i \neq R_i$. In this case, the value of r'_i is unknown to $\mathcal{B}$. $\mathcal{B}$ then rewinds $\mathcal{F}$ to the point where the query $H_1(mpk \| ID_i \| R'_i)$ is performed, and returns a new random value c''_i as the answer of the hash query. By the General Forking Lemma, with probability at least $((\rho(k)/q_c) - (q_1 q_c/q))((\rho(k)/q_c q_1) - (q_c/q) - (1/q))$, the two forgeries $V = (g^{ta})^{r'_i + c'_i b}$ and $V' = (g^{t'a})^{r'_i + c''_i b}$ output by $\mathcal{F}$ are valid and $c'_i \neq c''_i$. Then $\mathcal{B}$ outputs $(V^{1/t}/V'^{1/t'})^{1/(c'_i - c''_i)} = g^{ab}$ and halts.

So $\mathcal{B}$ solves the CDH problem with probability at least $((\rho(k)/q_c) - (q_1 q_c/q))((\rho(k)/q_c q_1) - (q_c/q) - \frac{1}{q})$. □

THEOREM 4.3. *Let* $k \in \mathbb{N}$ *be a security parameter. Under the random oracle model, if there exists a PPT algorithm that breaks the HS-ANON-CCA security* (*Definition 3.3*) *of* ***Hetero-I*** *with advantage at least* $\rho(k)$, *then there exists a PPT algorithm that solves the CDH Problem with probability at least* $(4/3)(1 - q_D/q)\rho(k)$, *where* q_D *is the maximum number of* ODesigncrypt *queries made in the game of HS-ANON-CCA.*

Proof. Suppose $\mathcal{B}$ is given a random instance (g^a, g^c) of the CDH problem, then $\mathcal{B}$ runs $\mathcal{D}$ to find the solution g^{ac}. $\mathcal{B}$ picks two random elements $x, y \in_R \mathbb{Z}_q$ and sets the two challenge public keys as $\mathrm{PK}_0 = g^{xc}$ and $\mathrm{PK}_1 = g^{yc}$. $\mathcal{B}$ then simulates all the hash queries and ODesigncrypt queries as in the proof of Theorem 4.1. After completing the first stage, $\mathcal{D}$ outputs a plaintext $m \in \{0,1\}^n$, two tuples $\{mpk_0, \mathrm{ID}_0, usk_0\}$ and $\{mpk_1, \mathrm{ID}_1, usk_1\}$, and requests a challenge ciphertext. According to Definition 3.3, mpk_0 and mpk_1 are of equal length. Also according to the scheme, $\mathrm{ID}_0, \mathrm{ID}_1 \in \{0,1\}^k$ and $usk_0 = (R_0, s_0)$, $usk_1 = (R_1, s_1)$.

$\mathcal{B}$ sets the challenge ciphertext to $\sigma' = (U', Z')$, where $U' = g^a$ and Z' is randomly drawn from $\{0,1\}^{n+4k}$. $\mathcal{B}$ then tosses a random coin b, and updates L2 by adding in $(mpk_b, \mathrm{ID}_b, m, U', \mathrm{PK}_0, R_b, \top)$ and $(mpk_b, \mathrm{ID}_b, m, U', \mathrm{PK}_1, R_b, \top)$, randomly picking t_0, t_1 and setting g^{t_0}, g^{t_1} as the result of $H_2(mpk_b, \mathrm{ID}_b, m, U', \mathrm{PK}_0, R_b, \top)$ and $H_2(mpk_b, \mathrm{ID}_b, m, U', \mathrm{PK}_1, R_b, \top)$, respectively. Note that those entries will only be added in L2 if they are not in L2 yet. Similarly, L3 will also be updated with $(U', \mathrm{PK}_0, \top)$ and $(U', \mathrm{PK}_1, \top)$. The value of $H_3(U', \mathrm{PK}_0, \top)$

and $H_3(U', \mathrm{PK}_1, \top)$ are set to $Z' \oplus (m \| mpk_b \| \mathrm{ID}_b \| R_b \| g^{s_b t_0})$ and $Z' \oplus (m \| mpk_b \| \mathrm{ID}_b \| R_b \| g^{s_b t_1})$, respectively.

$\mathcal{B}$ answers $\mathcal{D}$'s queries as in the first stage. If $\mathcal{D}$ queries H_2 or H_3 with $(U', \mathrm{PK}_0, \Lambda)$, such that $\hat{e}(U', \mathrm{PK}_0) = \hat{e}(g, \Lambda)$, then $\mathcal{B}$ outputs $\Lambda^{x^{-1}}$ and halts. If $\mathcal{D}$ queries H_2 or H_3 with $(U', \mathrm{PK}_1, \Lambda)$, such that $\hat{e}(U', \mathrm{PK}_1) = \hat{e}(g, \Lambda)$, then $\mathcal{B}$ outputs $\Lambda^{y^{-1}}$ and halts. $\mathcal{B}$ halts with failure, if $\mathcal{D}$ halts without making those queries.

As in Theorem 4.1, the failure probability of $\mathcal{B}$ in answering ODesigncrypt queries is at most q_D/q. Let $\mathbf{E}$ be the event that $(U', \mathrm{PK}_0, \mathrm{PK}_0^a)$ or $(U', \mathrm{PK}_1, \mathrm{PK}_1^a)$ has been queried to H_2 or H_3. $\bar{\mathbf{E}}$ denotes that event $\mathbf{E}$ does not happen. Note that $\mathcal{B}$ solves the CDH problem in event $\mathbf{E}$.

Let $V_{b,b'} = H_2(mpk_b, \mathrm{ID}_b, m, U', \mathrm{PK}_{b'}, R_b, \mathrm{PK}_{b'}^{a})^{s_b}$. If the output of $Z' \oplus H_3(U', \mathrm{PK}_{b'}, \mathrm{PK}_{b'}^a)$ is $m \| mpk_b \| \mathrm{ID}_b \| R_b \| V_{b,b'}$, then $\sigma' = (U', Z')$ is the signcryption of m under mpk_b, ID_b, usk_b and $\mathrm{PK}_{b'}$. In event $\bar{\mathbf{E}}$, neither H_2 nor H_3 is queried with $(g^a, \mathrm{PK}_0, \mathrm{PK}_0^a)$ or $(g^a, \mathrm{PK}_1, \mathrm{PK}_1^a)$, and σ' looks random to $\mathcal{A}$; Thus, $\Pr[\mathcal{D} \text{ wins the game} | \bar{\mathbf{E}}] = 1/4$. From the proposition, we have

$$\begin{aligned} \Pr[\mathcal{D} \text{ wins the game}] &= \Pr[\mathbf{E}] + \frac{1}{4}(1 - \Pr[\mathbf{E}]) \\ &\geq \frac{1}{4} + \rho(k) \\ \Rightarrow \quad \Pr[\mathbf{E}] &\geq \frac{4}{3}\rho(k). \end{aligned}$$

Hence, $\mathcal{B}$ solves the CDH problem with probability at least $(4/3)(1 - q_D/q)\rho(k)$. □

5. OUR SECOND SCHEME: HETERO-II

In this section, we propose another construction, **Hetero-II**, which does not require any party to carry out bilinear pairing, and hence can further improve the efficiency of the scheme. However a bilinear group is still needed, and it is for the security proofs only. Compared with **Hetero-I** proposed in Section 4 above, one may consider **Hetero-II** as a modification of **Hetero-I** by replacing the bilinear pairing-based signature generation with a non-pairing-based identity-based signature generation algorithm which is inspired from a signature scheme proposed by Zhu *et al.* [33]. We reuse all the notations defined in Section 4 but change H_2 to map from $\{0, 1\}^*$ to $\mathbb{Z}_q$. Below is the description of **Hetero-II**.

MasterKeyGen: On input of 1^k, it randomly chooses $x \in_R \mathbb{Z}_q$ and sets $msk := x$ and $mpk := g^x$.

UserKeyGen: On input of msk and an identity $\mathsf{ID} \in \{0, 1\}^k$, it carries out the following steps:

(1) Compute $c' = H_1(mpk, \mathsf{ID}, R)$, where $R = g^r$ and $r \in_R \mathbb{Z}_q$.
(2) Compute $s = r - c'x \bmod q$.

The user secret key is $usk_{\mathsf{ID}} := (c', s)$.

ServerKeyGen: On input of 1^k, it randomly chooses $x' \in_R \mathbb{Z}_q$ and sets $\mathrm{SK} := x'$ and $\mathrm{PK} := g^{x'}$.

H-Signcrypt: On input of mpk, ID, $usk_{\mathsf{ID}} = (c', s)$, PK and message $m \in \{0, 1\}^n$, the following steps are carried out:

(1) Compute
$e' = H_2(mpk, \mathsf{ID}, m, U, \mathrm{PK}, c', \mathrm{PK}^t)$
where $U = g^t$ and $t \in_R \mathbb{Z}_q$.
(2) Compute $v = t - e's \bmod q$.
(3) Compute
$Z = (m \| mpk \| \mathsf{ID} \| c' \| v) \oplus H_3(U, \mathrm{PK}, \mathrm{PK}^t)$.

The ciphertext $c = (U, Z)$.

H-Designcrypt: The following steps are carried out on input of $\mathrm{SK} = x'$ and $c = (U, Z)$:

(1) Compute
$(m \| mpk \| \mathsf{ID} \| c' \| v) \leftarrow Z \oplus H_3(U, \mathrm{PK}, U^{x'})$.
(2) Compute $e' = H_2(mpk, \mathsf{ID}, m, U, \mathrm{PK}, c', U^{x'})$.
(3) Check if $c' \stackrel{?}{=} H_1(mpk, \mathsf{ID}, mpk^{c'}(U/g^v)^{1/e'})$.
(4) If so, output $(mpk, \mathsf{ID}, m, \sigma)$, where $\sigma = (U, c', \mathrm{PK}, D = U^{x'}, v)$; otherwise, output $\perp$.

H-Ver: On input of mpk, an identity $\mathsf{ID} \in \{0, 1\}^k$, a message $m \in \{0, 1\}^n$ and a signature $\sigma = (U, c', \mathrm{PK}, D, v)$, the following steps are carried out:

(1) Compute $e' = H_2(mpk, \mathsf{ID}, m, U, \mathrm{PK}, c', D)$.
(2) Check if $c' \stackrel{?}{=} H_1(mpk, \mathsf{ID}, mpk^{c'}(U/g^v)^{1/e'})$.
(3) If so, output 1; otherwise, output 0.

In H-Signcrypt, the randomness t is reused for both signature generation and ElGamal encryption. One may consider σ as a non-interactive proof system of a signature on the user's identity. In UserKeyGen, the KGC generates usk_{ID} as a signature on ID. In H-Signcrypt, the user conducts a non-interactive proof of usk_{ID} (i.e. the signature on the user's identity). Below we analyze its security in terms of confidentiality, unforgeability and ciphertext anonymity.

THEOREM 5.1. *Let $k \in \mathbb{N}$ be a security parameter. Under the random oracle model, if there exists a PPT algorithm that breaks the HS-IND-CCA security (Definition 3.1) of* ***Hetero-II*** *above with advantage at least $\rho(k)$, then there exists a PPT algorithm that solves the CDH Problem with probability at least $2(1 - q_D/q)\rho(k)$, where q_D is the maximum number of* ODesigncrypt *queries made in the game of HS-IND-CCA.*

Compared with **Hetero-I** (Section 4), we can see that ServerKeyGen is identical and the 'encryption' part in H-Signcrypt, namely using $H_3(U, \mathrm{PK}, \mathrm{PK}^t)$ to mask the message, the user's identity information and the signature, is also identical. Therefore, the proof technique for this theorem is the same as that for Theorem 4.1. In particular, we still need the bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ in the proof when evaluation of

a potential Diffie–Hellman tuple is required, for example, when simulating H_2 and H_3. We skip the details and refer readers to the proof of Theorem 4.1.

With regard to unforgeability, in the following, we show that the security of **Hetero-II** can be reduced to the existential unforgeability against a chosen-message attack (EUF-CMA) [34] of the Schnorr signature scheme [35], which was proven to be secure [36] in the random oracle model under the Discrete Logarithm assumption.

THEOREM 5.2. *Let $k \in \mathbb{N}$ be a security parameter. Under the random oracle model, if there exists a PPT algorithm that breaks the HS-EUF-ID-CMA security (Definition 3.2) of **Hetero-II** with advantage at least $\rho(k)$, then there exists a PPT algorithm that breaks the EUF-CMA security of the Schnorr signature scheme with advantage at least $(\rho(k)/q_c)((\rho(k)/q_c q_2) - (1/q))$, where q_2 and q_c are the maximum number of H_2 and* OCreateUser *queries made in the game of HS-EUF-ID-CMA.*

Proof. Assume that there exists a forger $\mathcal{F}$ which wins the game in Definition 3.2 with probability at least $\rho(k)$. We construct an algorithm $\mathcal{B}$ that breaks the EUF-CMA security of the Schnorr signature scheme. Suppose $\mathcal{B}$ is given a public key $pk = g^a \in \mathbb{G}$ of the Schnorr signature scheme. $\mathcal{B}$ has access to a signing oracle $\mathcal{O}_s$ and an H_1 oracle. $\mathcal{B}$ sets $mpk = pk$, and sets up a simulated HS-EUF-ID-CMA game for $\mathcal{F}$ as follows.

For all H_1 queries made by $\mathcal{F}$, $\mathcal{B}$ relays the queries to its own H_1 oracle and returns the answers it gets to $\mathcal{F}$. For H_2 and H_3, $\mathcal{B}$ simulates them by picking returns randomly from the corresponding output ranges while ensuring to return the same answers for repeated queries. Similar to the proof of Theorem 4.1, lists L2 and L3 are maintained. In particular, if $\mathcal{F}$ queries H_2 or H_3 with the input containing $(U, \mathrm{PK}, \Lambda)$ where $\hat{e}(U, \mathrm{PK}) = \hat{e}(g, \Lambda)$ while in the corresponding L2 or L3 there is a triple $(U, \mathrm{PK}, \top)$, then $\top$ in the list is replaced by Λ.

$\mathcal{B}$ simulates the OCreateUser query as follows: $\mathcal{B}$ randomly picks i from $[1, q_c]$. For $j \neq i$, $\mathcal{B}$ issues a signing query with message ID_j, and sets the signature returned by $\mathcal{O}_s$ as the user secret key of ID_j. For the ith OCreateUser query, $\mathcal{B}$ randomly selects $r_i \in_R \mathbb{Z}_q$, computes $R_i \leftarrow g^{r_i}$ and makes an H_1 query to get $c_i = H_1(pk, \mathsf{ID}_i, R_i)$.

For an ORevealUserKey query on ID_j, $\mathcal{B}$ returns the user secret key usk_{ID_j} generated in the OCreateUser query to $\mathcal{F}$ if $j \neq i$. If $j = i$, then $\mathcal{B}$ aborts with failure.

For the OSigncrypt query on an identity ID_j, if $j \neq i$, $\mathcal{B}$ simulates H-Signcrypt accordingly as usk_{ID_j} is known. If $j = i$, the ciphertext is generated as follows: randomly pick $v, e \in_R \mathbb{Z}_q$, set $U = g^v(R_i/mpk^{c_i})^e$ and e as the value of $H_2(mpk, \mathsf{ID}_i, m, U, \mathrm{PK}, c_i, \top)$, where $\top$ denotes the expected value of U^{SK}. As in the proof of Theorem 4.1, L2 is updated with $(mpk, \mathsf{ID}_i, m, U, \mathrm{PK}, c_i', \top)$.

Suppose $\mathcal{F}$ outputs a forgery (U^*, Z^*), an identity ID_j and a receiver's key pair (PK, SK), such that ID_j has never been queried to ORevealUserKey and $1 \leq j \leq q_c$, Then $\mathcal{B}$ runs H-Designcrypt to get $m^* \| mpk \| \mathsf{ID}_j \| c_j^* \| v^*$. If $j \neq i$, then $\mathcal{B}$ aborts with failure. If $j = i$, then $\mathcal{B}$ rewinds $\mathcal{F}$ to the point where the $H_2(mpk, \mathsf{ID}_i, m^*, U^*, \mathrm{PK}, c_i^*, U^{*\mathrm{SK}})$ query is performed and returns a new random $\tilde{e}^*$ as the value of $H_2(mpk, \mathsf{ID}_i, m^*, U^*, \mathrm{PK}, c_i^*, U^{*\mathrm{SK}})$. Let $(\tilde{U}^*, \tilde{Z}^*)$ denote the new forgery output by $\mathcal{F}$. By the General Forking Lemma, with probability at least $(\rho(k)/q_c)((\rho(k)/q_c q_2) - (1/q))$, $\mathcal{F}$ outputs two valid forgeries (U^*, Z^*) and $(\tilde{U}^*, \tilde{Z}^*)$, where $U^* = \tilde{U}^* = g^{t^*}$ for some $t^* \in \mathbb{Z}_q$, $Z^* = m^* \| mpk \| \mathsf{ID}_i \| c_i^* \| v^* \oplus H_3(U^*, \mathrm{PK}, U^{*\mathrm{SK}})$ and $\tilde{Z}^* = m^* \| mpk \| \mathsf{ID}_i \| c_i^* \| \tilde{v}^* \oplus H_3(U^*, \mathrm{PK}, U^{*\mathrm{SK}})$, and $e^* \neq \tilde{e}^*$. Since both forgeries are valid, we have $v^* = t^* - e^* s_i$ and $\tilde{v}^* = t^* - \tilde{e}^* s_i$, where $c_i^* = H_1(mpk, \mathsf{ID}_i, g^{s_i} mpk^{c_i^*})$. Then $\mathcal{B}$ computes $s_i \leftarrow (v^* - \tilde{v}^*)(\tilde{e}^* - e^*)^{-1} \bmod q$ and outputs (c_i^*, s_i) as the forged Schnorr signature on message ID_i.

Therefore, if $\mathcal{F}$ can win the HS-EUF-ID-CMA game with advantage $\rho(k)$, then $\mathcal{B}$ breaks the EUF-CMA security of the Schnorr signature scheme with advantage at least $(\rho(k)/q_c)((\rho(k)/q_c q_2) - 1/q))$. □

THEOREM 5.3. *Let $k \in \mathbb{N}$ be a security parameter. Under the random oracle model, if there exists a PPT algorithm that breaks the HS-ANON-CCA security (Definition 3.3) of **Hetero-II** with advantage at least $\rho(k)$, then there exists a PPT algorithm that solves the CDH Problem with probability at least $(4/3)(1 - q_D/q)\rho(k)$, where q_D is the maximum number of* ODesigncrypt *queries made in the game of HS-ANON-CCA.*

Similar to the relationship between Theorems 4.1 and 5.1, the proof of this theorem is similar to that of Theorem 4.3. We refer readers to the proof of Theorem 4.3 for details.

6. PERFORMANCE

Table 1 shows the performance of the two proposed constructions and that of existing signcryption schemes. In the table, the column **Setting** denotes the cryptographic setting that the schemes are designed to work on: PKI means that the scheme is working under the conventional certificate-based public key setting; ID means that the scheme is identity based; and Hetero means that it is a heterogeneous signcryption scheme. The column **Key Privacy** indicates whether the schemes support key privacy (i.e. ciphertext anonymity). Column **Size** shows the ciphertext size which is represented in the number of elements in $\mathbb{Z}_q$, in $\mathbb{G}$ which is an appropriate elliptic curve group and in $\{0, 1\}^k$ (for example, a user identity ID), where k is the security parameter or the length of an RSA modulus (e.g. $k = 1024$). Note that the size of an encrypted message is not counted as it is the same for all the schemes in the table. We may therefore consider the size specified in the table as the *size overhead* of the schemes. We do not make a distinction between a conventional elliptic curve group used for encryption (e.g. ElGamal encryption) and a pairing-friendly elliptic curve group in the table. In practice, the same group may be used by setting a minimum security level that the scheme can achieve.

TABLE 1. Performance comparison.

Scheme	Setting	Key Privacy	Size $\mathbb{G}, \mathbb{Z}_q, \{0,1\}^k$	Complexity BP, EXP, ECSM	Security
Zheng [1]/w ECC	PKI	No	0, 1, 1	0, 0, 3	OC[40], IU[40],-
BD [2]/w ECC	PKI	No	0, 1, 1	0, 0, 5	OC[a], IU[a], P
TBOS [4]	PKI	No	0, 0, 1	0, 4, 0	IC[a], IU, P
CYHC [41]	ID	No	1, 0, 1	6, 0, 3	IC, IU, P
LW1 [22]	PKI	No	1, 1, 1	1, 2, 3	IC, IU, P
LW2 [22]	PKI	No	2, 1, 0	1, 0, 6	IC, IU, P
Boyen ID-SC [5]	ID	Yes	2, 0, 1	5, 0, 6	IC, IU, P
LYWDC [8]	PKI	Yes	3, 0, 0	2, 0, 4	IC, IU, P
Hetero-I (Section 4)	Hetero	Yes	4, 0, 1	2,0,5	IC, IU, P
Hetero-II (Section 5)	Hetero	Yes	2, 2, 1	0, 0, 6	IC, IU, P

[a]Assume that the ElGamal encryption scheme under ECC is used in BD [2]. These papers did not have a proof on the properties but they are commonly believed to support them.

For example, the 512-bit curve A in PBC [37] may be used where the security multiplier is equal to 2. Its security level is comparable with 80-bit symmetric security.

The computational complexity is shown under the column named **Complexity** in the table. Here, we only consider those expensive operations, such as bilinear pairing – (BP), modular exponentiation (EXP, e.g. RSA) and elliptic curve scalar multiplication – (ECSM), and we consider the total computational complexity of both signcryption and de-signcryption. The column **Security** in the table illustrates the security level currently known that the schemes can achieve. *O* denotes outsider security; *I* denotes insider security, *C* denotes confidentiality, *U* denotes unforgeability and P denotes public verifiability. Most of the recently proposed signcryption schemes support public verifiability, but the earlier schemes such as [1] do not.

Among the schemes supporting key privacy, the two heterogeneous signcryption schemes generally have longer ciphertexts. We additionally include mpk when compared with Boyen's identity-based signcryption scheme [5] as we consider the multi-KGC setting in heterogeneous signcryption (discussions following Definition 3.1); and additionally include ID when compared with Li *et al.*'s conventional public key-based signcryption scheme [8]. In terms of computational complexity, the second heterogeneous signcryption (Section 5) gives the best performance among all the key privacy-preserving signcryption schemes. The main advantage is due to the elimination of bilinear pairing. Carrying out one ECSM operation is at least two to three times faster than carrying out one bilinear pairing operation [38, 39].

7. CONCLUDING REMARKS

We formalized heterogeneous signcryption and proposed two efficient schemes under this new setting. The notion allows a user who has an identity-based user secret key to generate a ciphertext for a server who has a conventional certificate-based public key pair. Besides the security models for confidentiality and unforgeability, we also proposed a security model for key privacy. Insider security is included in all the three security models. The Multi-KGC setting and chosen-KGC attacks are also captured in the confidentiality and key privacy model. The two proposed schemes have both proved secure in the random oracle model relying on standard number theoretic assumptions. They also support public verifiability. The non-pairing-based scheme is the most efficient one computationally among all key privacy-preserving signcryption schemes.

One future work is to extend the heterogeneous signcryption so that a user can also be the receiver of a ciphertext. Another possible future work is to extend the multi-KGC setting that we proposed in the confidentiality game of heterogeneous signcryption to that of the pure identity-based signcryption.

FUNDING

The work was supported by a grant from CityU (Project No. 7002585).

REFERENCES

[1] Zheng, Y. (1997) Digital Signcryption or How to Achieve Cost(signature & encryption) ≪ Cost(signature) + Cost(encryption). *Proc. CRYPTO 97*, Santa Barbara, CA, USA, August, Lecture Notes in Computer Science, 1294, pp. 165–179. Springer, Berlin.

[2] Bao, F. and Deng, R.H. (1998) A Signcryption Scheme with Signature Directly Verifiable by Public Key. *Proc. PKC 98*, Pacifico Yokohama, Japan, February, Lecture Notes in Computer Science, 1431, pp. 55–59. Springer, Berlin.

[3] Steinfeld, R. and Zheng, Y. (2000) A Signcryption Scheme Based on Integer Factorization. *Proc. ISW 00*, Wollongong, NSW,

Australia, December, Lecture Notes in Computer Science, 1975, pp. 308–322. Springer, Berlin.

[4] Malone-Lee, J. and Mao, W. (2003) Two birds one stone: Signcryption using RSA. *Proc. CT-RSA 03*, San Francisco, CA, USA, April, Lecture Notes in Computer Science, 2612, pp. 211–225. Springer, Berlin.

[5] Boyen, X. (2003) Multipurpose Identity-based Signcryption: A Swiss Army Knife for Identity-based Cryptography. *Proc. CRYPTO 03*, Santa Barbara, CA, USA, August, Lecture Notes in Computer Science, 2729, pp. 383–399. Springer, Berlin.

[6] Libert, B. and Quisquater, J.-J. (2003) New Identity Based Signcryption Schemes from Pairings. *IEEE Information Theory Workshop 2003*, Paris, France, April, pp. 155–158. IEEE.

[7] Libert, B. and Quisquater, J.-J. (2004) Efficient signcryption with key privacy from gap Diffie–Hellman Groups. *Proc. PKC 04*, Singapore, March, Lecture Notes in Computer Science, 2947, pp. 187–200. Springer, Berlin.

[8] Li, C.K., Yang, G., Wong, D.S., Deng, X. and Chow, S.S. (2007) An efficient signcryption scheme with key privacy. *Proc. EuroPKI 07*, Palma de Mallorca, Spain, June, Lecture Notes in Computer Science, 4582, pp. 78–93. Springer, Berlin.

[9] Shamir, A. (1984) Identity-based cryptosystems and signature schemes. *Proc. CRYPTO 84*, Santa Barbara, CA, USA, August, Lecture Notes in Computer Science, **196**, pp. 47–53. Springer, Berlin.

[10] Boneh, D. and Franklin, M. (2001) Identity based encryption from the Weil pairing. *Proc. CRYPTO 01*, Santa Barbara, CA, USA, August, Lecture Notes in Computer Science, 2139, pp. 213–229. Springer, Berlin.

[11] Krawczyk, H. (2005) HMQV: A high-performance secure Diffie–Hellman protocol. *Proc. CRYPTO 05*, Santa Barbara, CA, USA, August, Lecture Notes in Computer Science, 3621, pp. 546–566. Springer, Berlin.

[12] Okamoto, T., Tso, R. and Okamoto, E. (2005) One-way and Two-party authenticated ID-based key agreement Protocols Using Pairing. *Proc. MDAI 05*, Tsukuba, Japan, July, Lecture Notes in Computer Science, 3558, pp. 122–133. Springer, Berlin.

[13] Voltage Security, I. (2005). *Identity-based Encryption System*. United States Patent: 6,886,096.

[14] Voltage Security, I. (2006). *Identity-based-Encryption Messaging System with Public Parameter Host Servers*. United States Patent: 7,017,181.

[15] Appenzeller, G. Martin, L. and Schertler, M. (2009) *RFC 5408: Identity-based Encryption Architecture and Supporting Data Structures*. IETF RFC 5408, http://tools.ietf.org/html/rfc5408.

[16] Yang, G., Huang, Q., Wong, D.S. and Deng, X. (2010) Universal authentication protocols for anonymous wireless communications. *IEEE Trans. Wirel. Commun.*, **9**, 168–174.

[17] Wang, Y., Wong, D.S. and Huang, L. (2010) One-pass Key Establishment for Anonymous Wireless Roaming. *Proc. 2010 IEEE Int. Conf. Wireless Communications, Networking and Information Security (WCNIS 2010)*, Beijing, China, June, pp. 533–537. IEEE.

[18] Gamage, C., Leiwo, J. and Zheng, Y. (1999) Encrypted Message Authentication by Firewalls. *Proc. PKC 99*, Kamakura, Japan, March, Lecture Notes in Computer Science, **1560**, pp. 69–81. Springer, Berlin.

[19] Mu, Y. and Varadharajan, V. (2000) Distributed signcryption. *Proc. INDOCRYPT 2000*, Calcutta, India, December, Lecture Notes in Computer Science, 1977, pp. 155–164. Springer, Berlin.

[20] Yum, D.H. and Lee, P.J. (2002) New signcryption Schemes Based on KCDSA. *Proc. ICISC 2001*, Seoul, Korea, December, Lecture Notes in Computer Science, **2288**, pp. 305–317. Springer, Berlin.

[21] An, J. H., Dodis, Y. and Rabin, T. (2002) On the security of joint signature and encryption. *Proc. EUROCRYPT 2002*, Amsterdam, The Netherlands, April, Lecture Notes in Computer Science, 2332, pp. 83–107. Springer, Berlin.

[22] Li, C.K. and Wong, D.S. (2010) Signcryption from randomness recoverable public key encryption. *Inf. Sci.*, **180**, 549–559.

[23] Baek, J., Steinfeld, R. and Zheng, Y. (2002) Formal proofs for the security of signcryption. *Proc. PKC 2002*, Paris, France, February, Lecture Notes in Computer Science, **2274**, pp. 80–98. Springer, Berlin.

[24] Bellare, M. and Rogaway, P. (1994) Entity authentication and key distribution. *Proc. CRYPTO 93*, Santa Barbara, CA, USA, August, Lecture Notes in Computer Science, **773**, pp. 232–249. Springer, Berlin.

[25] Han, Y. (2007) Generalization of signcryption for resources-constrained environments. *Wirel. Commun. Mob. Comput.*, **7**, 919–931.

[26] Malone-Lee, J. (2002). Identity Based Signcryption. Cryptology ePrint Archive, Report 2002/098.

[27] Dent, A. (2005) Hybrid signcryption schemes with outsider security. *Proc. ISC 2005*, Singapore, September, Lecture Notes in Computer Science, **3650**, pp. 203 – 217. Springer, Berlin.

[28] Paterson, K.G. and Srinivasan, S. (2008) Security and Anonymity of Identity-based Encryption with Multiple Trusted Authorities. *Proc. Pairing 2008*, Egham, UK, September, Lecture Notes in Computer Science, **5209**, pp. 354–375. Springer, Berlin.

[29] Bellare, M., Boldyreva, A., Desai, A. and Pointcheval, D. (2001) Key-privacy in public-key encryption. *Proc. ASIACRYPT 2001*, Gold Coast, Australia, December, Lecture Notes in Computer Science, **2248**, pp. 566–582. Springer.

[30] Bellare, M. and Rogaway, P. (1993) Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. *Proc. ACM CCS 93*, Fairfax, November, pp. 62–73. ACM.

[31] Waters, B. (2005) Efficient identity-based encryption without random oracles. *Proc. EUROCRYPT 2005*, Aarhus, Denmark, May, Lecture Notes in Computer Science, **3494**, pp. 114–127. Springer.

[32] Bellare, M. and Neven, G. (2006) Multi-signatures in the Plain Public-key Model and a General Forking Lemma. *Proc. ACM CCS 2006*, Alexandria, VA, USA, November, pp. 390–399. ACM.

[33] Zhu, R.W., Yang, G. and Wong, D.S. (2007) An efficient identity-based key exchange protocol with KGS forward secrecy for low-power devices. *Theor. Comput. Sci.*, **378**, 198–207.

[34] Goldwasser, S., Micali, S. and Rivest, R. (1988) A digital signature scheme secure against adaptive chosen-message attack. *SIAM J. Comput.*, **17**, 281–308.

[35] Schnorr, C.P. (1991) Efficient signature generation by smart cards. *J. Cryptol.*, **4**, 161–174.

[36] Pointcheval, D. and Stern, J. (1996) Security proofs for signature Schemes. *Proc. EUROCRYPT 96*, Saragossa, Spain, May, Lecture Notes in Computer Science, **1070**, pp. 387–398. Springer, Berlin.

[37] PBC Library. *The pairing-based cryptography library*. http://crypto.stanford.edu/pbc/.

[38] MIRACL. *Multiprecision integer and rational arithmetic C/C++ library*. http://www.shamus.ie/.

[39] Xiong, X., Wong, D.S., and Deng, X. (2009) Tinypairing: Computing Tate Pairing on Sensor Nodes with Higher Speed and Less Memory. *Proc. NCA 2009*, Cambridge, MA, USA, July, pp. 187–194. IEEE Computer Society.

[40] Baek, J., Steinfeld, R. and Zheng, Y. (2007) Formal proofs for the security of signcryption. *J. Cryptol.*, **20**, 203–235.

[41] Chow, S., Yiu, S., Hui, L. and Chow, K. (2003) Efficient Forward and Provably Secure ID-based Signcryption Scheme with Public Verifiability and Public Ciphertext Authenticity. *Proc. ICISC 2003*, Seoul, Korea, November, Lecture Notes in Computer Science, **2971**, pp. 352–369. Springer, Berlin.