Generalized Hierarchical Identity-Based Signcryption

Hao Wang
School of computer science and technology, Shandong University, Jinan, China
Email: whatsdu@gmail.com

Qiuliang Xu[1] and Xiufeng Zhao[1,2]
1 School of computer science and technology, Shandong University, Jinan, China
2 Institute of Electronic Technology, Information Engineering University, Zhengzhou, China
Email: xuqiuliang@sdu.edu.cn, zhao_xiu_feng@163.com

***Abstract*—In this paper, we propose a generic method to construct Hierarchical Identity-Based Signcryption scheme. Using this method, a Hierarchical Identity-Based Signcryption scheme can be converted from any Hierarchical Identity-Based Encryption scheme. Then, we give a concrete instantiation, which is the first constant-size fully secure hierarchical identity-based signcryption scheme in the standard model. Furthermore, our scheme can achieve CCA2 security level without using any additional cryptography primitive.**

***Index Terms*—hierarchical identity-based signcryption, fully secure, constant-size ciphertext, composite order bilinear group**

I. Introduction

Identity-Based Encryption (IBE) is a public-key encryption scheme where one's public key can be freely set to any value (such as one's identity): An authority that holds a master secret key can take any arbitrary identifier and extract a secret key corresponding to this identifier. Anyone can then encrypt messages using the identifier as a public encryption key, and only the holder of the corresponding secret key can decrypt these messages. This concept was introduced by Shamir [13], a partial solution was proposed by Maurer and Yacobi [14], and the first fully functional IBE systems were described by Boneh and Franklin [1] and Cocks [4].

IBE system can greatly simplify the public-key infrastructure for encryption solutions, but they are still not as general as one would like. Many organizations have a hierarchical structure, perhaps with one central authority, several sub-authorities and sub-sub-authorities and many individual users, each belonging to a small part of the organization tree. We would like to have a solution where each authority can delegate keys to its sub-authorities, who in turn can keep delegating keys further down the hierarchy to the users. An IBE system that allows delegation as above is called Hierarchical Identity-Based Encryption (HIBE). In HIBE, messages are encrypted for identity-vectors, representing nodes in the identity hierarchy. This concept was introduced by Horwitz and Lynn [9], who also described a partial solution to it, and the first fully functional HIBE system was described by Gentry and Silverberg [10].

In many situations we want to enjoy confidentiality, authenticity and non-repudiation of message simultaneously. The general IBE (HIBE) can not guarantee the authenticity and non-repudiation. A traditional method to solve this problem is to digitally sign a message then followed by an encryption (signature-then-encryption) that can have two problems: low efficiency and high cost of such summation, and the case that any arbitrary scheme cannot guarantee the security. Signcryption is a relatively cryptographic technique that is supposed to fulfill the functionalities of digital signature and encryption in a single logical step and can effectively decrease the computational costs and communication overheads in comparison with the traditional signature-then-encryption schemes. The first signcryption scheme was introduced by Yuliang Zheng in 1997 [18]. Zheng also proposed an elliptic curve-based signcryption scheme that saves 58% of computational and 40% of communication costs when it is compared with the traditional elliptic curve-based signature-then-encryption schemes [19]. There are also many other signcryption schemes that are proposed throughout the years, each of them having its own problems and limitations, while they are offering different level of security services and computational costs.

By combining identity-based cryptology and signcryption, Malone-Lee [20] proposed the first identity-based signcryption (IBSC) scheme along with a security model. But Libert and Quisquater [21] pointed out that Malone-Lee's scheme is not semantically secure. Then, Chow et al. [22] proposed an identity-based signcryption scheme that can provide both public verifiability and forward security. In 2003, Boyen [23] proposed an anonymity identity-based signcryption scheme in the random oracle model. Then, Chen and Malone-Lee improved Boyen's scheme in efficient [24]. In 2009, Yu

Manuscript received Dec. 30, 2009; accepted Mar. 1, 2010.
Corresponding author: Qiuliang Xu, xuqiuliang@sdu.edu.cn

doi:10.4304/jcp.5.7.1078-1085

et al. [25] proposed the first identity-based signcryption scheme without random oracles. Similar to IBSC, Chow et al. [7] proposed the concept of hierarchical identity-based signcryption (HIBSC) by combining HIBS and HIBE. Then, Yuen and Wei proposed the first constant-size HIBSC without random oracles [17], but they used an interactive intractability assumption and selective-id model in their reductionist security proof. It is an open problem to avoid these assumption and model.

Our contribution In this paper, we give a generic method to construct HIBSC scheme. Using this method, a HIBSC scheme can be converted from any HIBE scheme. Then, we give a concrete instantiation from the constant-size fully secure HIBE scheme introduced by Lewko et al. [11]. Our HIBSC scheme is the first constant-size fully secure hierarchical identity-based signcryption scheme in the standard model. Furthermore, our scheme can achieve CCA2 security level without using any additional cryptography primitive.

Organization In Section 2, we formally define the HIBE system and the HIBSC system, give the complete security definition, and give an introduction of composite order bilinear groups. In Section 3, we present our method for converting a HIBE scheme into HIBSC scheme, and prove the security of the HIBSC scheme in Section 4. In Section 5, we give a concrete instantiation which is fully secure with constant-size ciphertexts. In Section 6, we show how to enhance the security of our HIBSC, and give a modified HIBSC scheme which can achieve CCA2 security level without using additional cryptography primitive. In Section 7, we conclude and discuss open directions for further research.

II. BACKGROUND

A. Hierarchical Identity Based Encryption

A Hierarchical Identity Based Encryption scheme has five algorithms: **Setup**, **KeyGen**, **Delegate, Encrypt**, and **Decrypt**.

$\mathbf{Setup}(\lambda) \to PK, MSK$ The setup algorithm takes a security parameter λ as input and output the public parameters PK and a master secret key MSK.

$\mathbf{KeyGen}(MSK, \vec{I}) \to SK_{\vec{I}}$ The key generation algorithm takes the master secret key and an identity vector $\vec{I}$ as input and outputs a private key $SK_{\vec{I}}$.

$\mathbf{Delegate}(PK, SK_{\vec{I}}, I) \to SK_{\vec{I}:I}$ The delegation algorithm takes a secret key for the identity vector $\vec{I}$ of depth d and an identity I as input and outputs a secret key for the depth $d+1$ identity vector $\vec{I}:I$ formed by concatenating I onto the end of $\vec{I}$.

$\mathbf{Encrypt}(PK, M, \vec{I}) \to CT$ The encryption algorithm takes the public parameters PK, a message M, and an identity vector $\vec{I}$ as input and outputs a ciphertext CT.

$\mathbf{Decrypt}(PK, CT, SK) \to M$ The decryption algorithm takes the public parameters PK, a ciphertext CT, and a secret key SK as input and output the message M, if the ciphertext was an encryption to an identity vector $\vec{I}$ and the secret key is for the same identity vector.

Notice that the decryption algorithm is only required to work when the identity vector for the ciphertext matches the secret key exactly. However, someone who has a secret key for a prefix of this identity vector can delegate to themselves the required secret key and also decrypt.

B. Hierarchical Identity Based Signcryption

A Hierarchical Identity Based Encryption scheme has five algorithms: **Setup**, **KeyGen**, **Delegate**, **Signcrypt**, and **Unsigncrypt**.

The **Setup**, **KeyGen** and **Delegate** algorithms are same as those in the HIBE system. We describe the **Signcrypt** and **Unsigncrypt** algorithms as follow:

$\mathbf{Signcrypt}(PK, \vec{I}_S, \vec{I}_R, SK_S, M) \to SCT$ The signcryption algorithm takes the public parameters PK, a message M, the identity and secret key of sender, the identity of receiver as input and outputs a ciphertext $SCT = (C, \sigma)$.

$\mathbf{Unsigncrypt}(PK, SCT, \vec{I}_S, SK_R) \to M$ The unsigncrypt algorithm takes the public parameters PK, a ciphertext SCT, the identity of sender, the secret key of receiver as input and outputs M if σ is valid corresponding to M. Otherwise, it outputs the symbol $\perp$.

C. Security Definition for HIBSC

The security definition of HIBSC includes two properties: indistinguishability and existential unforgeability. Then, we introduce a stronger property, strong existential unforgeability

1) Indistinguishability

We define the indistinguishability against adaptive chosen identity and adaptive chosen ciphertext/plaintext attack for HIBSC (IND-ID-CCA2/CPA), as in the following game:

Setup. The challenger will run the **Setup** algorithm and gives the public parameters PK to the adversary.

The challenger will also initialize a set $S = \phi$, which will be the set of private keys it has created, but not given out.

Phase 1. The adversary makes repeated queries of one of five types:

Create The attacker gives the challenger an identity-vector $\vec{I}$. The challenger creates a key for the vector, but does net give it to the adversary. It instead adds the key to the set S and gives the attacker a reference to it.

Delegate The attacker specifies a key $SK_{\vec{I}}$ in the set S for an identity $\vec{I}$. Then it gives the challenger an identity I'. The challenger runs the $\mathbf{Deleagte}(PK, SK_{\vec{I}}, I')$ algorithm to get a new secret key $SK_{\vec{I}:I'}$ and adds this to the set S.

Reveal The attacker specifies an element of the set S for a secret key SK. The challenger removes the item from the set S and gives the attacker the secret key. We note at this point there is no need for the challenger to allow more delegate queries on the key since the attacker can run them itself.

Signcrypt The attacker specifies sender identity $\vec{I}_S$, receiver identity $\vec{I}_R$ and message M. The challenger runs **Signcrypt**$(PK, \vec{I}_S, \vec{I}_R, SK_S, M)$, gives the attacker the valid ciphertext SCT corresponding to $(\vec{I}_S, \vec{I}_R, M)$.

Unsigncrypt The attacker specifies a sender identity $\vec{I}_S$, a receiver identity $\vec{I}_R$, and a ciphertext $SCT = (C, \sigma)$. The challenger will output a message M for a valid σ or will output $\perp$ otherwise.

Challenge. The adversary submits two equal length messages M_0^* and M_1^* and challenge identity vectors $(\vec{I}_S^*, \vec{I}_R^*)$ with the restriction that each identity vector $\vec{I}$ given out in the key phase must not be a prefix of $\vec{I}_R^*$. The challenger then flips a random coin β, and signcrypts M_β^* under $(\vec{I}_S^*, \vec{I}_R^*)$. The resulting ciphertext SCT^* is given to the adversary.

Phase 2. Phase 1 is repeated with the restriction that any revealed identity vector $\vec{I}$ is not a prefix of $\vec{I}_R^*$, and SCT^* is not sent to the *Unsigncrypt* oracle.

Guess. The adversary output a guess β' of β.

The advantage of an adversary $\mathcal{A}$ in this game is defined as $\Pr[\beta' = \beta] - 1/2$. We note that the model can be easily converted to handle chosen-plaintext attacks by disallowing the *Unsigncrypt* queries in Phase 1 and Phase 2.

Definition 1 A hierarchical identity-based Signcryption scheme is IND-ID-CCA2/CPA secure if all polynomial time adversary have at most a negligible advantage in the above game.

2) Existential Unforgeability

We define the existential unforgeability against adaptive chosen identity and adaptive chosen plaintext attack for HIBSC (UF-ID-CPA), as in the following game:

Setup. The challenger runs **Setup** algorithm. It gives the adversary the resulting public key PK and keeps the master secret key MSK to itself.

Queries. The adversary makes repeated queries of one of four types: *Create* query, *Delegate* query, *Reveal* query, and *Signcrypt* query. All the queries are same as those in the indistinguishability game.

Forgery. The adversary outputs a tuple $(SCT^* = (C^*, \sigma^*), \vec{I}_S^*, \vec{I}_R^*)$. The adversary $\mathcal{A}$ wins if the following holds: $M^* \leftarrow$ **Unsigncrypt**$(PK, SCT^*, \vec{I}_S^*, SK_R^*)$, each identity vector $\vec{I}$ given out in the key phase must not be a prefix of $\vec{I}_S^*$ and M^* is not queried during the *Signcrypt* query phase. $\mathcal{A}$'s advantage is the probability that he wins.

Definition 2 A hierarchical identity-based Signcryption scheme is UF-ID-CPA secure if all polynomial time adversaries have at most a negligible advantage in the above game.

3) Strong Existential Unforgeability

Strong existential unforgeability against adaptive chosen identity and adaptive chosen plaintext attack (SUF-ID-CPA) is defined using the following game:

Setup, Queries and Forgery: Same as in the existential unforgeability game.

The adversary $\mathcal{A}$ wins if the following holds: $M^* \leftarrow$ **Unsigncrypt**$(PK, SCT^*, \vec{I}_S^*, SK_R^*)$, each identity vector $\vec{I}$ given out in the key phase must not be a prefix of $\vec{I}_S^*$ and (M^*, σ^*) is not generated during the *Signcrypt* query phase. $\mathcal{A}$'s advantage is the probability that he wins.

Definition 3 A hierarchical identity-based Signcryption scheme is SUF-ID-CPA secure if all polynomial time adversaries have at most a negligible advantage in the above game.

D. Composite Order Bilinear Groups

Composite order bilinear groups were first introduced in [8]. We define them by using a group generator $\mathcal{G}$, an algorithm which takes a security parameter λ as input and outputs a description of a bilinear group G. In our case, $\mathcal{G}$ outputs $(p_1, p_2, p_3, G, G_T, e)$ where p_1, p_2, p_3 are distinct primes, G and G_T are cyclic groups of order $n = p_1 p_2 p_3$, and $e: G^2 \to G_T$ is map such that:

1.(Bilinear) $\forall g, h \in G$, $a, b \in Z_n$, $e(g^a, h^b) = e(g, h)^{ab}$

2.(Non-degenerate) $\exists g \in G$ such that $e(g, g)$ has order n in G_T.

We further require that the group operations in G and G_T as well as the bilinear map e are computable in polynomial time with respect to λ. Also, we assume the group descriptions of G and G_T include generators of the respective cyclic groups. We let G_{p_1}, G_{p_2}, and G_{p_3} denote the subgroups of order p_1, p_2, and p_3 in G respectively. We note that when $h_i \in G_{p_i}$ and $h_j \in G_{p_j}$ for $i \neq j$, $e(h_i, h_j)$ is the identity element in G_T. To see this, suppose $h_1 \in G_{p_1}$ and $h_2 \in G_{p_2}$. We let g denote a generator of G. Then, $g^{p_1 p_2}$ generates G_{p_3}, $g^{p_1 p_3}$ generates G_{p_2}, and $g^{p_2 p_3}$ generates G_{p_1}. Hence, for some α_1, α_2, $h_1 = (g^{p_2 p_3})^{\alpha_1}$ and $h_2 = (g^{p_1 p_3})^{\alpha_2}$. We note:

$$e(h_1, h_2) = e(g^{p_2 p_3 \alpha_1}, g^{p_1 p_3 \alpha_2}) = e(g^{\alpha_1}, g^{p_3 \alpha_2})^{p_1 p_2 p_3} = 1$$

This orthogonality property of G_{p_1}, G_{p_2}, G_{p_3} will be a principal tool in our constructions.

III. CONVERT HIBE INTO HIBSC

As noted by Boneh [1, Section 6] and formalized in [5], the key derivation of an identity-based encryption scheme immediately gives rise to a standard signature scheme. Similarly, Gentry and Silverberg [10] observed that any two-level hierarchical identity-based encryption scheme can be transformed into an IBS scheme. In this section, we gave a similar method for converting any Hierarchical

Identity-Based Encryption scheme into a Hierarchical Identity-Based Signcryption.

First of all, we should choose a hash function, which can map any element in the message space $SPACE_M$ into the identity space $SPACE_{ID}$, $H: SPACE_M \to SPACE_{ID}$. Using this hash function, we can transform a HIBE scheme into a HIBSC scheme as follow:

A Hierarchical Identity Based Signcryption scheme has five algorithms: **Setup**, **KeyGen**, **Delegate**, **Signcrypt**, and **Unsigncrypt**.

The **Setup**, **KeyGen**, **Delegate** algorithms are same as those in the HIBE system (Section 2), and the public parameters PK include the hash function H additionally.

Signcrypt$(PK, M, \vec{I}_S, \vec{I}_R, K_S) \to SCT$ The signcryption algorithm takes the public parameters PK, a message M, the identity and secret key of sender, the identity of receiver as input and calculate $\sigma = \textbf{Delegate}(PK, SK_{\vec{I}_S}, H(M)) = SK_{\vec{I}_S:H(M)}$, $C = \textbf{Encrypt}(PK, M, \vec{I}_R)$. Then, it outputs a ciphertext $SCT = (C, \sigma)$, where the **Delegate** and **Encrypt** are the delegation and encryption algorithms of the HIBE scheme.

Notice that this method for making a signature may cause an attack, anybody who knows this signature can personate the identity $\vec{I}_S : H(M)$. In fact, this attack can be ignored easily. The elements of identity mark with Tag_{ID}, and $H(M)$ marks with Tag_M. Both of these tags are public verifiability. If and only if all the elements of an identity marked with Tag_{ID}, it is a valid identity.

Unsigncrypt$(PK, SCT, \vec{I}_S, SK_R) \to M$ The unsigncrypt algorithm takes the public parameters PK, a ciphertext SCT, the identity of sender $\vec{I}_S$, and the secret key of receiver SK_R as input, and computes $M = \textbf{Decrypt}(PK, SCT, SK_R)$. Then, it verifies whether this equation $\textbf{Decrypt}(PK, \textbf{Encrypt}(PK, r, \vec{I}_S : H(M)), \sigma) = r$ holds, where $r \in_R SPACE_M$, the **Decrypt** and **Encrypt** are the decryption and encryption algorithms of the HIBE scheme. It outputs the symbol $\perp$ if the verification fails. Otherwise, it outputs M.

IV. Security of Our HIBSC

We prove the security of our HIBSC scheme from following two aspects:

A. Indistinguishability

Theorem 1 Our HIBSC scheme is IND-sID/ID-CPA /CCA security, iff it is converted from a HIBE scheme, which is secure in the same security model, using our method.

Proof. Without loss of generality, we just proof this theorem in the IND-ID-CCA2 model. Given any PPT adversary $\mathcal{A}$ attacking our HIBSC in an adaptive chosen-ciphertext attack, we construct a PPT adversary $\mathcal{A}'$ attacking the HIBE in an adaptive chosen-ciphertext attack. Relating the success probabilities of these adversaries gives the desired result. We now define adversary $\mathcal{A}'$ as follows:

Setup(λ) outputs (PK, MSK) and $\mathcal{A}'$ is given PK. Adversary $\mathcal{A}'$, in turn, run $\mathcal{A}$ on input λ and PK.

Phase 1 The adversary $\mathcal{A}$ makes repeated queries of one of five types:

Create Once $\mathcal{A}$ makes the *Create* query for an identity-vector $\vec{I}$, $\mathcal{A}'$ makes the same query. Then the key for the vector $\vec{I}$ is created, but isn't given to $\mathcal{A}'$. Instead the key is added to the set S and a reference to it is given to $\mathcal{A}'$. Then $\mathcal{A}'$ transfers this reference to $\mathcal{A}$.

Delegate $\mathcal{A}$ specifies a key $SK_{\vec{I}}$ in the set S for an identity $\vec{I}$. Then it makes a *Delegate* query for an identity I'. Then $\mathcal{A}'$ makes the oracle query $Deleagte(PK, SK_{\vec{I}}, I')$ to add a new secret key $SK_{\vec{I}:I'}$ to the set S.

Reveal $\mathcal{A}$ specifies an element $\vec{I}$ of the set S for a secret key $SK_{\vec{I}}$. Then $\mathcal{A}'$ makes the oracle query $Reveal(\vec{I})$ to get the secret key $SK_{\vec{I}}$, and gives it to $\mathcal{A}$.

Signcrypt $\mathcal{A}$ specifies sender identity $\vec{I}_S$, receiver identity $\vec{I}_R$ and message M. Then $\mathcal{A}'$ makes the oracle query $Reveal(\vec{I}_S)$ to get $SK_{\vec{I}_S}$ (We suppose that $\vec{I}_S$ is already in the set S), calculates $\sigma = \textbf{Delegate}(PK, SK_{\vec{I}_S}, H(M)) = SK_{\vec{I}_S:H(M)}$, $C = \textbf{Encrypt}(PK, M, \vec{I}_R)$, and gives the ciphertext $SCT = (C, \sigma)$ to $\mathcal{A}$.

Unsigncrypt $\mathcal{A}$ specifies a ciphertext $SCT = (C, \sigma)$, and receiver identity $\vec{I}_R$. Then $\mathcal{A}'$ makes the oracle query $Decrypt(PK, C)$ to get a message M. If the signature σ is valid, $\mathcal{A}'$ gives $\mathcal{A}$ the message M, otherwise a symbol $\perp$.

Challenge. $\mathcal{A}$ submits two equal length messages M_0^*, M_1^* and challenge identity vectors $(\vec{I}_S^*, \vec{I}_R^*)$ with the restriction that each identity vector $\vec{I}$ given out in the key phase must not be a prefix of $\vec{I}_R^*$. $\mathcal{A}'$ makes the oracle query $Challenge(M_0^*, M_1^*, \vec{I}_R^*)$ to get a ciphertext C^*. Then, $\mathcal{A}'$ makes the oracle queries $Deleagte(PK, SK_{\vec{I}_S^*}, H(M)) \to \sigma^*$ and $Reveal(\vec{I}_S^* : H(M))$ in turn to get σ^*. $\mathcal{A}'$ gives $SCT^* = (C^*, \sigma^*)$ to $\mathcal{A}$.

Phase 2. Phase 1 is repeated with the restriction that any revealed identity vector $\vec{I}$ is not a prefix of $\vec{I}_R^*$, and C^* is not sent to the *Unsigncrypt* oracle.

Guess. The adversary $\mathcal{A}$ outputs a guess β' of β, then gives it to $\mathcal{A}'$ for the guess of $\mathcal{A}'$.

If $\mathcal{A}$ has a non-negligible advantage to win the above game, $\mathcal{A}'$ has a same advantage to win the corresponding security game.

B. Existential Unforgeability

Theorem 2 Our HIBSC system is Existential Unforgeability security, iff it is converted from an IND-sID-CPA secure HIBE, using our method.

We omit the proof of this theorem, because it can be proved using the method introduced in [1, 16, 12].

V. Concrete Instantiation

In this section, we gave a concrete instance of HIBSC. This scheme is transformed from an IND-ID-CPA secure HIBE [11], using our generic method introduced in Section 3.

A. Our Construntion

A Hierarchical Identity Based Encryption scheme has five algorithms: **Setup**, **KeyGen**, **Delegate**, **Encrypt**, and **Decrypt**.

Setup: The setup algorithm chooses a bilinear group G of order $N = p_1 p_2 p_3$. We let l denote the maximum depth of the HIBSC. The setup algorithm chooses $g, h, u_1, ...u_l, v \in G_{p_1}$, $X_3 \in G_{p_3}$, $\alpha \in Z_N$, and a hash function $H: G_T \to Z_N$. The public parameters are $PK = \{g, h, u_1, ...u_l, v, X_3, e(g,g)^{\alpha}, H\}$, and the master secret key is $MSK = \alpha$.

KeyGen$(PK, MSK, \vec{I} = (I_1, ..., I_j))$: The key generation algorithm chooses $r_{\vec{I}} \in Z_N$ randomly and also chooses random elements R_3, R_3', $R_{j+1}, \ldots, R_l$, $R_m \in G_{p_3}$. It outputs $SK_{\vec{I}} = (K_1, K_2, E_{j+1}, ..., E_l, E_m)$, where

$$K_1 = g^{r_{\vec{I}}} R_3,$$
$$K_2 = g^{\alpha} (u_1^{I_1} \cdots u_j^{I_j} h)^{r_{\vec{I}}} R_3',$$
$$E_{j+1} = u_{j+1}^{r_{\vec{I}}} R_{j+1},$$
$$\cdots\cdots$$
$$E_l = u_l^{r_{\vec{I}}} R_l,$$
$$E_m = v^{r_{\vec{I}}} R_m.$$

Delegate: Given a key $K_1', K_2', E_{j+1}', ..., E_l'$ for $\vec{I} = (I_1, ..., I_j)$, the delegation algorithm creates a key for $\vec{I}' = (I_1, ..., I_{j+1})$ as follow. It chooses a random element $r_{\vec{I}'} \in Z_N$ and random elements $\tilde{R}_3$, $\tilde{R}_3'$, $\tilde{R}_{j+2}, \ldots, \tilde{R}_l$, $\tilde{R}_m \in G_{p_3}$. The new key is set as:

$$K_1 = K_1' g^{r_{\vec{I}'}} \tilde{R}_3,$$
$$K_2 = K_2' (u_1^{I_1} \cdots u_j^{I_j} h)^{r_{\vec{I}'}} (E_{j+1}')^{I_{j+1}} u_{j+1}^{r_{\vec{I}'} \cdot I_{j+1}} \tilde{R}_3',$$
$$E_{j+2} = E_{j+2}' u_{j+2}^{r_{\vec{I}'}} \tilde{R}_{j+2},$$
$$\cdots\cdots$$
$$E_l = E_l' u_l^{r_{\vec{I}'}} \tilde{R}_l,$$
$$E_m = E_m' v^{r_{\vec{I}'}} \tilde{R}_m.$$

We note that this new key is fully randomized: its only tie to the previous key is in the values $I_1, ..., I_j$.

Signcrypt$(PK, M, \vec{I}_S = (I_{1'}, ..., I_{i'}), SK_S = (K_{S,1}, K_{S,2}, E_{S,i+1}, ..., E_{S,l}, E_{S,m}), \vec{I}_R = (I_1, ..., I_j))$: The sender randomly chooses $s, s' \in Z_N$ and $\hat{R}_3, \hat{R}_3' \in G_{p_3}$. It sets:

$$\sigma_1 = K_{S,1} \cdot g^{s'} \hat{R}_3$$
$$\sigma_2 = K_{S,2} \cdot (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} h)^{s'} (E_m)^{H(M)} v^{s' \cdot H(M)} \hat{R}_3',$$ and
$$C_0 = M \cdot e(g,g)^{\alpha \cdot s},\ C_1 = (u_1^{I_1} \cdots u_j^{I_j} h)^s,\ C_2 = g^s.$$

The sender sends the tuple $SCT = (C_0, C_1, C_2, \sigma_1, \sigma_2)$ to the receiver.

Unsigncrypt$(PK, SCT, \vec{I}_S = (I_{1'}, ..., I_{i'}), SK_R = (K_{R,1}, K_{R,2}, E_{R,j+1}, ..., E_{R,l}, E_{R,m}))$: Received a tuple $SCT = (C_0, C_1, C_2, \sigma_1, \sigma_2)$, the receiver decrypts the ciphertext as follows:

$$\omega = \frac{e(K_{R,2}, C_2)}{e(K_{R,1}, C_1)}$$
$$= \frac{e(g^{\alpha} (u_1^{I_1} \cdots u_j^{I_j} h)^{r_{\vec{I}}} R_3', g^s)}{e(g^{r_{\vec{I}}} R_3, (u_1^{I_1} \cdots u_j^{I_j} h)^s)} = e(g,g)^{\alpha \cdot s},$$
$$M = C_0 / \omega$$

Then, it verifies:

$$\frac{e(\sigma_2, g)}{e(\sigma_1, (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot h \cdot v^{H(M)}))} \stackrel{?}{=} e(g,g)^{\alpha}$$

The receiver accepts the message if and only if the above equation holds.

Correctness:

$$\frac{e(\sigma_2, g)}{e(\sigma_1, (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot h \cdot v^{H(M)}))}$$
$$= \frac{e(g^{\alpha} (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot h \cdot v^{H(M)})^{r_S + s'} R_m^{H(M)} \hat{R}_3', g)}{e(g^{r_S + s'} \hat{R}_3, (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot h \cdot v^{H(M)}))}$$
$$= e(g,g)^{\alpha}$$

B. Security of HIBSC

Theorem 3 The HIBSC scheme is IND-ID-CPA and UF-ID-CPA security.

Proof. This HIBSC scheme is converted from an IND-ID-CPA secure HIBE scheme, using our method introduced in the Section 3. According to Theorem 2, the above HIBSC scheme is IND-ID-CPA and UF-ID-CPA security.

VI. From CPA to CCA2

The above HIBSC scheme is IND-ID-CPA and UF-ID-CPA security. We can use the method introduced by Canetti et al. in [6] to modify this scheme to get IND-ID-CCA2 security, but the efficiency must be reduced. In fact, we only need make a small modification, exchanging the order of "sign" and "encrypt", and make a transformation from weak unforgeability into strong unforgeability using the method introduced by Boneh et al. in [3]. After the modification and transformation, we get a new HIBSC scheme, which is IND-ID-CCA2 and SUF-ID-CPA security.

A. Modified HIBSC scheme

Before introducing how to enhance the security, we give a modified hierarchical identity-based signcryption scheme mHIBSC, in which we only exchange the order of "sign" and "encrypt". The mHIBSC scheme also has five algorithms: **Setup**, **KeyGen**, **Delegate**, **Signcrypt,** and **Unsigncrypt**. The **Setup**, **KeyGen**, **Delegate** algorithms are same as those in our HIBSC scheme introduced in the section 5. We describe the modified **Signcrypt** and **Unsigncrypt** algorithms as follow:

Signcrypt$(PK, M, \vec{I}_S = (I_{1'},...,I_{i'}), SK_S = (K_{S,1}, K_{S,2}, E_{S,i+1}, ..., E_{S,l}, E_{S,m}), \vec{I}_R = (I_1,...,I_j))$: The sender randomly chooses $s, s' \in Z_N$ and $\hat{R}_3, \hat{R}_3^{'} \in G_{p_3}$. It sets:

$C_0 = M \cdot e(g,g)^{\alpha \cdot s}$,

$C_1 = (u_1^{I_1} \cdots u_j^{I_j} h)^s$,

$C_2 = g^s$,

$\sigma_1 = K_{S,1} \cdot g^{s'} \hat{R}_3$,

$\sigma_2 = K_{S,2} \cdot (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} h)^{s'} (E_m)^{H(C_0)} v^{s' \cdot H(C_0)} \hat{R}_3^{'}$.

The sender sends the tuple $SCT = (C_0, C_1, C_2, \sigma_1, \sigma_2)$ to the receiver.

Unsigncrypt$(PK, SCT, \vec{I}_S = (I_{1'},...,I_{i'}), SK_R = (K_{R,1}, K_{R,2}, E_{R,j+1},...., E_{R,l}, E_{R,m}))$: Received a tuple $SCT = (C_0, C_1, C_2, \sigma_1, \sigma_2)$, the receiver verifies:

$$\frac{e(\sigma_2, g)}{e(\sigma_1, (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot h \cdot v^{H(C_0)}))} \stackrel{?}{=} e(g,g)^{\alpha}$$

It outputs the symbol $\perp$ if the verification fails. Otherwise, it decrypts the ciphertext as follows:

$$\omega = \frac{e(K_{R,2}, C_2)}{e(K_{R,1}, C_1)} = \frac{e(g^{\alpha}(u_1^{I_1} \cdots u_j^{I_j} h)^{r_{\tilde{i}}} R_3^{'}, g^s)}{e(g^{r_{\tilde{i}}} R_3, (u_1^{I_1} \cdots u_j^{I_j} h)^s)} = e(g,g)^{\alpha \cdot s},$$

$M = C_0 / \omega$, and outputs M.

Correctness:

$$\frac{e(\sigma_2, g)}{e(\sigma_1, (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot h \cdot v^{H(C_0)}))}$$

$$= \frac{e(g^{\alpha}(u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot h \cdot v^{H(C_0)})^{r_S + s'} R_m^{H(C_0)} \overline{R}_3^{'}, g)}{e(g^{r_S + s'} \overline{R}_3, (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot h \cdot v^{H(C_0)}))}$$

$$= e(g,g)^{\alpha}$$

B. From Weak Unforgeability to Strong Unforgeability

The famous results of Canetti et al. [6], further improved upon by Boneh and Katz [2], show how to build a CCA2-secure Identity-Based encryption scheme from a 2-level HIBE scheme. We can use this method to build IND-ID-CCA2 secure HIBSC based our modified scheme. First of all, we will convert this scheme from UF-ID-CPA secure into SUF-ID-CPA, using the general transformation introduced by Boneh et al. [3].

We build a new strongly unforgeable system $\text{HIBSC}_{\text{new}}$ also included five algorithms: **Setup**$_{new}$, **KeyGen**$_{new}$, **Delegate**$_{new}$, **Signcrypt**$_{new}$, and **Unsigncrypt**$_{new}$. The **KeyGen**$_{new}$, **Delegate**$_{new}$ algorithms are same as those in the HIBSC scheme introduced in the section 5. We describe the modified **Setup**$_{new}$, **Signcrypt**$_{new}$ and **Unsigncrypt**$_{new}$ algorithms as follow:

Setup$_{new}$: To generate the public key, select random generators $\tilde{g}, \tilde{h} \in G_T$ and a hash function $\tilde{H}: \{0,1\}^* \to Z_N$. Next run **Setup** to obtain a master secret key MSK and public key PK. The public and master secret keys for the new system are: $PK' = (PK, \tilde{g}, \tilde{h}, \tilde{H})$ and $MSK' = (MSK)$.

Signcrypt$_{new}(PK, M, \vec{I}_S = (I_{1'},...,I_{i'}), SK_S = (K_{S,1}, K_{S,2}, E_{S,i+1},...., E_{S,l}, E_{S,m}), \vec{I}_R = (I_1,...,I_j))$: The sender chooses random elements $s, x, y \in Z_N$ and $\hat{R}_3, \hat{R}_3^{'} \in G_{p_3}$. It sets:

$C_0 = M \cdot e(g,g)^{\alpha \cdot s}$,

$C_1 = (u_1^{I_1} \cdots u_j^{I_j} h)^s$,

$C_2 = g^s$,

$\sigma_1 = K_{S,1} \cdot g^x \hat{R}_3 = g^{(r_S + x)} R_3 \hat{R}_3$, then computes

$t \leftarrow \tilde{H}(C_0 \| \sigma_1)$,

$c_0 \leftarrow \tilde{g}^t \tilde{h}^y$,

$\sigma_2 = K_{S,2} \cdot (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} h)^x (E_m)^{H(c_0)} v^{x \cdot H(c_0)} \hat{R}_3^{'}$

$= g^{\alpha} (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot v^{H(c_0)} \cdot h)^{(r_S + x)} R_3 R_m^{H(c_0)} \hat{R}_3^{'}$.

The sender sends the tuple $SCT = (C_0, C_1, C_2, \sigma_1, \sigma_2, y)$ to the receiver.

Unsigncrypt$_{new}(PK, SCT, \vec{I}_S = (I_{1'},...,I_{i'}), K_R = (K_{R,1}, K_{R,2}, E_{R,j+1},...., E_{R,l}, E_{R,m}))$: Received a tuple $SCT = (C_0, C_1, C_2, \sigma_1, \sigma_2, y)$, the receiver computes $\hat{t} \leftarrow \tilde{H}(C_0 \| \sigma_1)$, $c_0 \leftarrow \tilde{g}^{\hat{t}} \tilde{h}^y$, then checks

$$\frac{e(\sigma_2, g)}{e(\sigma_1, (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot h \cdot v^{H(c_0)}))} \stackrel{?}{=} e(g,g)^{\alpha}.$$

It outputs the symbol $\perp$ if the verification fails. Otherwise, it decrypts the ciphertext as follows:

$$\omega = \frac{e(K_{R,2}, C_2)}{e(K_{R,1}, C_1)}$$

$$= \frac{e(g^{\alpha}(u_1^{I_1} \cdots u_j^{I_j} h)^{r_{\tilde{i}}} R_3^{'}, g^s)}{e(g^{r_{\tilde{i}}} R_3, (u_1^{I_1} \cdots u_j^{I_j} h)^s)} = e(g,g)^{\alpha \cdot s},$$

$M = C_0 / \omega$, and outputs M.

Correctness:

$$\frac{e(\sigma_2, g)}{e(\sigma_1, (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot h \cdot v^{H(c_0)}))}$$

$$= \frac{e(g^{\alpha}(u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot h \cdot v^{H(c_0)})^{r_S + x} R_3 R_m^{H(c_0)} \hat{R}_3^{'}, g)}{e(g^{r_S + x} R_3 \hat{R}_3, (u_{1'}^{I_{1'}} \cdots u_{i'}^{I_{i'}} \cdot h \cdot v^{H(c_0)}))} = e(g,g)^{\alpha}$$

C. Security of HIBSC_{new}

Theorem 4 The mHIBSC scheme is IND-ID-CPA and UF-ID-CPA security.

We omit the proof of this theorem, because it is easy to see that the security level of the modified scheme mHIBSC and the original scheme HIBSC are same.

Theorem 5 The HIBSC$_{new}$ scheme is IND-ID-CCA2 and SUF-ID-CPA security.

Lemma 1 HIBSC$_{new}$ system is SUF-ID-CPA security.

Our proof is based on the technology used by Boneh et al. in [3].

Proof. Suppose $\mathcal{A}$ is a forger that (t,q,ε) - breaks strong unforgeability of HIBSC$_{new}$. Forger $\mathcal{A}$ is first given a public key $(PK,\tilde{g},\tilde{h},\tilde{H})$.

Forger $\mathcal{A}$ asks for signcryption on $(\vec{I}_{S1},\vec{I}_{R1},M_1)$,…, $(\vec{I}_{Sq},\vec{I}_{Rq},M_q)$ and is given signcryption $SCT_i=(C_{i,0}, C_{i,1},C_{i,2},\sigma_{i,1},\sigma_{i,2},y_i)$ for $i=1,...,q$ on these tuples. Let $t_i=\tilde{H}(C_{i,0}\,\|\,\sigma_{i,1})$ and $c_{i,0}\leftarrow\tilde{g}^{t_i}\tilde{h}^{y_i}$ for $i=1,...,q$. Let $(\hat{C}_0,\hat{\sigma}_1,\hat{\sigma}_2,\hat{y})$ be the forgery produced by $\mathcal{A}$, let $\hat{t}=\tilde{H}(\hat{C}_0\,\|\,\hat{\sigma}_1)$, and let $\hat{c}_0\leftarrow\tilde{g}^{\hat{t}}\tilde{h}^{\hat{y}}$. We distinguish among three types of forgeries:

Type I. A forgery where $\hat{c}_0=c_{i,0}$ and $\hat{t}=t_i$ for some $i\in\{1,...,q\}$.

Type II. A forgery where $\hat{c}_0=c_{i,0}$ and $\hat{t}\neq t_i$ for some $i\in\{1,...,q\}$.

Type III. Any other forger ($\hat{c}_0\neq c_{i,0}$ for $i\in\{1,...,q\}$).

A successful forgery must output a forgery of Type I, Type II, or Type III. We show that a Type I forgery can be used to break the collision-resistance of $\tilde{H}$, a Type II forgery can be used to solve discrete log in G_T, and a Type III forgery can be used to break existential unforgeability of the underlying signcryption scheme mHIBSC. Our simulator can flip a coin at the beginning of the simulation to guess which type of forgery the adversary will produce and set up the simulation appropriately. In all three cases the simulation is perfect. We start by describing how to use a Type III forgery which is the more interesting case.

Type III forger: Suppose algorithm $\mathcal{A}$ is a Type III forger that (t,q,ε) - breaks strong unforgeability of HIBSC$_{new}$. We construct a simulator $\mathcal{B}$ that (t,q,ε) - breaks existential unforgeability of mHIBSC. $\mathcal{B}$ is given a public key PK. $\mathcal{B}$'s goal is to produce a pair (c_0,σ) where $\sigma=(\sigma_1,\sigma_2)$ is a valid signature on c_0 and c_0 is not among $\mathcal{B}$'s chosen queries. $\mathcal{B}$ runs $\mathcal{A}$ as follow.

Setup. Algorithm $\mathcal{B}$ generates the public key PK' as follow.

1. Select a random generator $\tilde{g}\in G_T$.
2. Select random exponents $a\in Z_N^*$ and set $\tilde{h}\leftarrow g^a$.
3. Select a hash function: $\tilde{H}:\{0,1\}^*\rightarrow Z_N$.
4. Provide the public key $PK'=(PK,\tilde{g},\tilde{h},\tilde{H})$ to $\mathcal{A}$.

Signcrypt Queries. Algorithm $\mathcal{A}$ issues up q *signcrypt* queries. Algorithm $\mathcal{B}$ responds to a query on a triple $(\vec{I}_S,\vec{I}_R,M)$ as follow.

1. Select a random exponent $\omega\in Z_N$ and set $c_0\leftarrow\tilde{g}^{\omega}$.
2. Ask $\mathcal{B}$'s challenger for a signature on c_0, and obtain a signature (σ_1,σ_2) on c_0.
3. Compute $(C_0,C_1,C_2)\leftarrow$ **Encrypt**$(PK,M,\vec{I}_R)$.
4. Compute $t\leftarrow\tilde{H}(C_0\,\|\,\sigma_1)$.
5. Set $y\leftarrow(\omega-t)/a$.
6. Return $SCT\leftarrow(C_0,C_1,C_2,\sigma_1,\sigma_2,y)$ to $\mathcal{A}$.

Indeed, $c_0\leftarrow\tilde{g}^{\omega}=\tilde{g}^{a\cdot y+t}=\tilde{g}^t\tilde{h}^y$ and y is uniform in Z_N as required. Hence, SCT is a valid signcryption on triple $(\vec{I}_S,\vec{I}_R,M)$.

Output. Finally, algorithm $\mathcal{A}$ outputs a forgery $(\hat{C}_0,\hat{C}_1,\hat{C}_2,\hat{\sigma}_1,\hat{\sigma}_2,\hat{y})$. Algorithm $\mathcal{B}$ produces a weak forgery on the underlying scheme as follow.

1. Compute $\hat{t}=\tilde{H}(\hat{C}_0\,\|\,\hat{\sigma}_1)$.
2. Compute $\hat{c}_0\leftarrow\tilde{g}^{\hat{t}}\tilde{h}^{\hat{y}}$.
3. Output $(\hat{c}_0,(\hat{\sigma}_1,\hat{\sigma}_2))$.

Note that $\hat{c}_0\notin\{c_1,...c_q\}$ because if $\hat{c}_0=c_{i,0}$ for some $i\in\{1,...,q\}$ then, either $\hat{t}=t_i$ (a Type I forgery) or $\hat{t}\neq t_i$ (a Type II forgery). Therefore $\mathcal{B}$ produces a forgery on some new $\hat{c}_0$ for the underlying scheme whenever $\mathcal{A}$ produces a Type III forgery, as required.

As space is limited, we omit showing how to use a Type I or Type II forgery. The method of making these forgery types can be found in [3].

In summary, we showed how to use all three forgery types to break existential unforgeability of the underlying signcrypt scheme, collision-resistance of $\tilde{H}$, or discrete log. This completes the proof of Lemma 1.

Lemma 2 HIBSC$_{new}$ system is IND-ID-CCA2 security.

The results of Canetti et al. [6], further improved upon by Boneh and Katz [2], show how to build a CCA-secure identity-based encryption scheme from a 2-level HIBE scheme. This result is easily extended to n-level HIBE. An n-level IND-ID-CCA secure HIBE can be built form an n+1-level IND-sID-CPA HIBE and a strongly unforgeable one-time signature scheme. Our mHIBSC system is IND-ID-CPA secure (**Theorem 4**), and our HIBSC$_{new}$ system is SUF-ID-CPA security (**Lemma 1**), so our HIBSC$_{new}$ system is IND-ID-CCA2 security.

Based Lemma 1 and Lemma 2, we can statement that our HIBSC$_{new}$ scheme is IND-ID-CCA2 and SUF-ID-CPA security.

VII. CONCLUSIONS

We have introduced a generic method to construct HIBSC scheme. Using our method, a HIBSC scheme can be easily converted form any HIBE scheme. But we note that, the efficiency of the HIBSC scheme relies on the delegation algorithm of the HIBE scheme seriously. So we should choose these HIBE schemes, which have efficient delegation algorithm as far as possible. Then, we

proposed a concrete instantiation, which is the first constant-size fully secure hierarchical identity-based signcryption scheme in the standard model. Furthermore, our scheme can achieve CCA2 security level without using additional cryptography primitive, but it needs exchange the order of "sign" and "encrypt". Since this form is a little different from the usual signcryption schemes, it remains an open problem to construct a constant-size fully secure HIBSC scheme as usual form in the IND-ID-CCA2 security model without using additional cryptography primitive.

ACKNOWLEDGMENT

The authors wish to thank anonymous reviewers for giving helpful suggestions. This work is supported by the National Nature Science Foundation of China under Grant No. 60873232, the National Nature Science Foundation of Shandong Province under Grant No. Y2007G37, and Graduate Independent Innovation Foundation of Shandong University under Grant No. yzc09043.

REFERENCES

[1] Dan Boneh, Matthew K. Franklin: Identity-Based Encryption from the Weil Pairing. CRYPTO 2001: 213-229.
[2] Dan Boneh, Jonathan Katz: Improved Efficiency for CCA-Secure Cryptosystems Built Using Identity-Based Encryption. CT-RSA 2005: 87-103.
[3] Dan Boneh, Emily Shen, Brent Waters: Strongly Unforgeable Signatures Based on Computational Diffie-Hellman. Public Key Cryptography 2006: 229-240.
[4] C. Cocks: An Identity Based Encryption Scheme Based on Quadratic Residues. IMA Int. Conf. 2001: 360-363.
[5] Yang Cui, Eiichiro Fujisaki, Goichiro Hanaoka, Hideki Imai, Rui Zhang: Formal Security Treatments for Signatures from Identity-Based Encryption. ProvSec 2007: 218-227.
[6] Ran Canetti, Shai Halevi, Jonathan Katz: Chosen-Ciphertext Security from Identity-Based Encryption. EUROCRYPT 2004: 207-222.
[7] Sherman S. M. Chow, Tsz Hon Yuen, Lucas Chi Kwong Hui, Siu-Ming Yiu: Signcryption in Hierarchical Identity Based Cryptosystem. SEC 2005: 443-457 .
[8] Dan Boneh, Eu-Jin Goh, Kobbi Nissim: Evaluating 2-DNF Formulas on Ciphertexts. TCC 2005: 325-341.
[9] Jeremy Horwitz, Ben Lynn: Toward Hierarchical Identity-Based Encryption. EUROCRYPT 2002: 466-481.
[10] Craig Gentry, Alice Silverberg: Hierarchical ID-Based Cryptography. ASIACRYPT 2002: 548-566.
[11] Allison Lewko, Brent Waters: Fully Secure HIBE with Short Ciphertexts. Cryptology ePrint Archive, Report 2009/482, 2009.
[12] Kenneth G. Paterson, Jacob C. N. Schuldt: Efficient Identity-Based Signatures Secure in the Standard Model. ACISP 2006: 207-222.
[13] Adi Shamir: Identity-Based Cryptosystems and Signature Schemes. CRYPTO 1984: 47-53.
[14] Ueli M. Maurer, Yacov Yacobi: Non-interactive Public-Key Cryptography. EUROCRYPT 1991: 498-507.
[15] Elaine Shi, Brent Waters: Delegating Capabilities in Predicate Encryption Systems. ICALP (2) 2008: 560-578.
[16] Brent Waters: Efficient Identity-Based Encryption Without Random Oracles. EUROCRYPT 2005: 114-127.
[17] Tsz Hon Yuen, Victor K. Wei: Constant-Size Hierarchical Identity-Based Signature/Signcryption without Random Oracles. Cryptology ePrint Archive, Report 2005/412, 2005.
[18] Yuliang Zheng: Digital Signcryption or How to Achieve Cost(Signature & Encryption) << Cost(Signature) + Cost(Encryption). CRYPTO 1997: 165-179.
[19] Yuliang Zheng, Hideki Imai: How to Construct Efficient Signcryption Schemes on Elliptic Curves. Inf. Process. Lett. 68(5): 227-233 (1998).
[20] John Malone-Lee. Identity Based Signcryption. Cryptology ePrint Archive, Report 2002/098, 2002.
[21] Benoît Libert and Jean-Jacques Quisquater: A new identity based signcryption scheme from pairings. Proceedings of the IEEE Information Theory Workshop 2003: 155-158.
[22] Sherman S. M. Chow, Siu-Ming Yiu, Lucas Chi Kwong Hui, K. P. Chow: Efficient Forward and Provably Secure ID-Based Signcryption Scheme with Public Verifiability and Public Ciphertext Authenticity. ICISC 2003: 352-369.
[23] Xavier Boyen: Multipurpose Identity-Based Signcryption (A Swiss Army Knife for Identity-Based Cryptography). CRYPTO 2003: 383-399.
[24] Liqun Chen, John Malone-Lee: Improved Identity-Based Signcryption. Public Key Cryptography 2005: 362-379.
[25] Yong Yu, Bo Yang, Ying Sun, Shenglin Zhu: Identity based signcryption scheme without random oracles. Computer Standards & Interfaces 31(1): 56-62 (2009).

Hao Wang was born in Shandong, China in 1984. He has completed his Bachelor of Science in Computational Mathematics in Qufu Normal University, China in 2007. He is a Ph.D. candidate of Shandong University, interested in the research on public key cryptography including identity based cryptography, encryption, key agreement etc.

Qiuliang Xu was born in Shandong, China in 1960. He has a Ph.D. in Applied Mathematics (1999) and an M.S. in Computational Mathematics (1985) from Shandong University, Jinan, China.

He is a Professor of Computer Science in Shandong University, where he has been since 1985. His main interest is public key cryptography including encryption, digital signature, crypto-graphic protocol etc.

Xiufeng Zhao was born in Shandong, China in 1977. She has completed her Bachelor of Science in Applied Mathematics in Qufu Normal University, China in 2000, and has her M.S in Applied Mathematics from Northwestern Polytechnical University. She is a Ph.D. candidate of Shandong University, interested in the research on cryptographic protocol, key agreement, etc.