HIBGE: Monitorable and Traceable Anonymous Encryption in Cloud Computing

Dawei Li[1], Jianwei Liu[2], Qianhong Wu[2], Zhenyu Guan[2]

[1] School of Electronic and Information Engineering, Beihang University, China
[2] School of Cyber Science and Technology, Beihang University, China
{lidawei, liujianwei, qianhong.wu, guanzhenyu}@buaa.edu.cn

Abstract

Privacy problem is an emerging concern when we protect information security in cloud computing. In some application scenarios, the users may require to receive the computing results anonymously from the cloud server. However, complete anonymity brings regulatory issues to practical use. We usually need the encrypted results monitorable by superiors in case of illegal information, and once there are some disputes occurring, a trusted third-party arbitration institution is also required to trace the recipients without any need of decryption. Aiming at this problem, we propose a new cryptographic primitive named Hierarchical Identity-Based Group Encryption (HIBGE). In an HIBGE system, the recipients are organized in a tree-like structure. Recipients with the same function or belonging to the same department make up a group and they are managed by a group manager. When encrypting, the cloud can use the recipient's identity as the public key, and others only know which group the message is sent to, but cannot know the exact recipient's identity. Besides, the higher level can monitor the lower's message, and group manager can trace the recipients' identity. We then construct a concrete HIBGE scheme, and prove this scheme is semantic secure, anonymous and traceable, with a perfectly zero-knowledge proof. HIBGE scheme can be widely applied in cloud computing.

Keywords: Hierarchical Identity-Based Group Encryption, Cloud computing, Anonymity, Traceability, Provable security

1 Introduction

With the network development, cloud computing has become a hot. Protecting data security and privacy of cloud computing is getting more and more important. In cloud computing, the computing data is transmitted from cloud server to recipients, and the data should be encrypted securely to avoid being wiretapped. To protect the recipients' privacy data from being collected by others, the data should be encrypted anonymously. However, once the data is illegal, complete anonymity gives the recipients too much power and makes it difficult for data monitoring and identity tracing. Usually the recipients can be divided into different levels and groups according to their function and business. Thus higher level should have the ability to audit the data, and once there are some disputes occurring, there should be a manager who can trace the recipient identity and give out an arbitration of the dispute.

In order to realize the anonymity and traceability of group membership, the first Group Encryption (GE) scheme was proposed by Kiayias et al. [1] in 2007, in which there are four parties involved: senders, verifiers, a group manager (GM) and an opening authority (OA). GM can manage the recipients who must prove himself to GM before joining the group and generating private keys. Messages in this group are anonymous, but OA can discover the identity of the recipient if there are disputes occurring.

However, there are still some drawbacks in GE. For example, the public key in GE is a series of irregular string, which is generated from a Public Key Infrastructure (PKI). Thus it is hard to check the correctness of the public key, and if PKI is offline, the GE will stop working. Identity-based cryptosystems and signature schemes were first proposed in 1984 by Shamir [2], in which the public key is the ID of the recipient, which can be represented by his e-mail address or his telephone number. Therefore, the correctness of the public key can be checked easily, to avoid error code of the encryption procedure due to a bit error in public key. This idea was not realized practically until Boneh and Franklin found the fully functional scheme in 2010, Identity-Based Encryption (IBE), using Weil Pairing [3]. The scheme they proposed is secure under chosen-ciphertext attack in random oracle model, and a public key generator (PKG) distributes the private key to the member in this system. To combine both GE and IBE, Luo et al. proposed Identity-Based Group Encryption (IBGE) scheme that realizes group encryption in the identity-based

*Corresponding Author: Zhenyu Guan; E-mail: guanzhenyu@buaa.edu.cn

DOI: 10.3966/160792642019032002031

cryptography setting [4].

Moreover, in practical applications, each group member has to generate its private key from a single PKG, which requires a large bandwidth and heavy burden of a single PKG. Besides, group members are often organized in a tree-like structure, which is more flexible in practice. Therefore, the IBE scheme was improved by Gentry and Silverberg and they proposed Hierarchical Identity-Based Encryption (HIBE) [5].

In view of the above application requirements, and inspired by the relevant research results, we proposed Hierarchical Identity-Based Group Encryption (HIBGE) cryptosystem for cloud computing with all above properties. In the HIBGE cryptosystem, members working in different groups can generate private keys by hierarchy, and the public key to encryption is recipient's identity, with proven security and anonymity, and the message is traceable in case of disputes occurring.

1.1 Our Contribution

We define a new cryptographic primitive named Hierarchical Identity-Based Group Encryption (HIBGE) cryptosystem, to meet the function requirements in the above scenarios. Our main contribution is summarized as follows.

- We formalize the cryptosystem notion and security notion. The system is composed of seven procedures and a verify protocol executed by five roles. In this system, the group manager generates its public key and private key by itself, and group members generate keys from their higher level. When encrypting a message, the sender uses the recipient's identity and the recipient's group manager's public key. The recipient uses its private key for decrypting and the group manager uses its private key for tracing. To ensure system functional requirements, the security model includes semantic security, anonymity and traceability by a game between a challenger and an adversary with a very strong ability.
- We construct a concrete HIBGE scheme on composite-order bilinear groups. In order to achieve traceability, we divide the ciphertext into two parts, where the first part is the encryption of message by using the recipient's identity as the public key, and the other part is the encryption of recipient's identity by using the group manager's public key. Besides, the first part of the ciphertext should be a completely anonymous HIBE scheme and we reform the anonymous HIBE scheme proposed by Caro et al. [6]. The second part of the ciphertext should be a CCA2 secure encryption scheme. However, if the identities used in both parts are inconsistent, it will cause the traceability invalid. Thus we come up with a zero-knowledge proof protocol to ensure the consistence. To avoid prover in the protocol to learn the recipient's hierarchy, we extend the identity vector used in both parts of ciphertext with a string of fixed parameters. And the protocol can be non-interactive to reduce the communication times.
- We prove the HIBGE scheme is semantic secure and anonymous game by game with the help of semi-functional ciphertext and semi-functional key. And we prove the zero-knowledge proof protocol is perfectly zero-knowledge. Finally, the traceability of this scheme is proved, which means that there is no polynomial time adversary, who can produce a valid ciphertext to lead the group manager to a wrong target tracing.

1.2 Related Work

Kiayias et al. [1] proposed Group Encryption and a modular design using zero-knowledge proofs, digital signatures, anonymous CCA-secure public-key encryption and commitment schemes. They showed an efficient instantiation by using Paillier's cryptosystem [7] and Camenisch-Lysyanskaya signatures [8]. By using Fiat-Shamir paradigm, the interaction can be removed [9]. Qin et al. [10] considered a sort of similar primitive with non-interactive proofs and short ciphertext, called Group Decryption. Libert et al. [11] proposed a traceable GE but message is anonymous to members.

Identity-based encryption and signature schemes were first proposed by Shamir [2], and it was first realized by Boneh and Boyen [3]. IBE was then constructed without random oracles by Boneh and Boyen [12], and they defined two security models that are selective identity secure and standard security model. Then Gentry [13] presented an IBE system that is fully secure in the standard model without random oracles. Waters [14] presented a new methodology for proving security of encryption systems using Dual System Encryption. Boneh et al. [15] proposed an approach for designing function private IBE schemes. And Chen et al. [16] proposed identity-based hash and signature schemes without key exposure.

Hierarchical ID-Based Cryptography was proposed by Gentry and Siverberg [5], which included hierarchical identity-based encryption and signature schemes that have total collusion resistance on all levels and chosen ciphertext security in random oracle model. Boyen and Waters [17] added anonymity to HIBE and gave a proof of security in standard model. Waters [14] presented fully secure HIBE systems under the simple and established decisional Bilinear Diffie-Hellman and decisional Linear assumptions. Ducas [18] showed how to use asymmetric pairings to convert a large family of HIBE constructions into anonymous HIBE systems. Lewko and Waters [19] proposed fully secure HIBE with short ciphertext, and Caro et al. [6] added anonymity to it, and proved the security which including semantic security and anonymity, by constructing semi-functional key and semi-functional ciphertext.

Data security and privacy protection in cloud computing has been developed a lot in recent years [20]. Zhang et al. [21] proposed the anonymous access control scheme in cloud computing. Shen et al. [22] proposed traceable and anonymous group data sharing schemes in cloud computing, and they designed block design-based key agreement protocol that supports multiple participants. Chen et al. proposed public verifiable scheme and public auditing protocol for cloud data [23-25]. Li et al. [26] proposed fuzzy identity-based data integrity auditing scheme in cloud. And Wu et al. [27] constructed efficient IBE scheme for cloud computing that supports equality test.

1.3 Notation

HIBGE is a new cryptographic primitive for cloud computing. To describe HIBGE much better, we introduce several notations in this paper. Table 1 summarizes these notations and their corresponding meanings.

Table 1. Notations

Notation	Description	Notation	Description
λ	Security Parameter	ϵ	Negligible Function of λ
$\in_R$	Randomly Picked Element	$\overrightarrow{ID_j}$	Identity Vector
ID_j	Identity	j	ID_j 's Level

We define λ as security parameter, and ϵ as a negligible function of λ. We use $a \in_R \mathbb{G}$ to represent picking a random element a in group $\mathbb{G}$. For hierarchical members whose level is j, we use ID_j to denote his identity, and $\overrightarrow{ID_j}$ to denote his identity vector. If $\overrightarrow{ID_j} = (ID_1, \ldots, ID_j)$, it means that ID_j generates his private key from his higher-level member ID_{j-1}, and so on.

1.4 Organization

We start with the necessary basic definitions and assumptions in Section 2. Then we present an HIBGE cryptosystem, and define the security model for HIBGE in Section 3. A concrete HIBGE scheme is constructed based on composite order bilinear groups in Section 4. We prove the correctness and security of this HIBGE scheme, which is semantic secure, anonymous, traceable and perfectly zero-knowledge in Section 5. Finally, we analyze the efficiency of the proposed scheme in Section 6.

2 Preliminaries

2.1 Composite Order Bilinear Groups

Let p be a large prime. $\mathbb{G}$ and $\mathbb{G}_T$ are two cyclic groups of order p. g is a generator of $\mathbb{G}$, and $e: \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a bilinear map. We say that $\mathbb{G}$ and $\mathbb{G}_T$ are bilinear groups if e satisfies following properties:

- Bilinearity: $\forall u, v \in \mathbb{G}$ and $\forall a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u,v)^{ab} = e(u^b, v^a)$.
- Non-degeneracy: $e(g,g) \neq 1$.
- Computability: group operation $e(u,v)$ for u, $v \in_R G$ can be efficiently computed.

In this scheme, we use bilinear groups of different orders [28]. If $p_1, p_2, p_3, p_4 \in_R \mathbb{Z}_{poly(|\lambda|)}$, which means $p_1, \ldots, p_4$ are chosen randomly and the length is polynomial of parameter λ. Then it computes integer $N = p_1 p_2 p_3 p_4$. Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of order N, and $\mathbb{G}_{p_1}, \ldots, \mathbb{G}_{p_4}$ are subgroups of $\mathbb{G}$ with order $p_1, \ldots, p_4$. We use an important property in this scheme, *orthogonality property*, which means that if g and h are elements of different order subgroups, $e(g,h) = 1$.

2.2 Bilinear Diffie-Hellman Assumption (BDH)

BDH Problem. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two bilinear groups of order q, and let $e: \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. The BDH problem is as follows: Select a generator g of $\mathbb{G}$, and the (g, g^a, g^b, g^c) for some $a, b, c \in \mathbb{Z}_p$ is given to compute $e(g,g)^{abc} \in \mathbb{G}_T$ [12]. An algorithm $\mathcal{A}$ has advantage ϵ in solving BDH if

$$Pr[\mathcal{A}(g, g^a, g^b, g^c) = e(g,g)^{abc}] \geq \epsilon$$

where the probability is over the random choice of g, a, b, c and random bits used in $\mathcal{A}$.

Definition 1. The (t, ϵ)-BDH assumption holds in $\mathbb{G}$ if there is no algorithm that has advantage at least ϵ in solving the BDH problem in polynomial time t.

An algorithm $\mathcal{B}$ that outputs $b \in \{0,1\}$ has advantage ϵ in solving *decision* BDH problem if

$$\begin{aligned} &|Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^{abc}) = 0] \\ &-Pr[\mathcal{B}(g, g^a, g^b, g^c, R) = 0]| \geq \epsilon \end{aligned}$$

where the probability is over the random choice of g,a,b,c, random bits used in $\mathcal{B}$ and random choice of $R \in_R \mathbb{G}_T$.

Definition 2. The (t,ϵ)- *decision* BDH assumption holds in $\mathbb{G}$ if there is no algorithm that has advantage at least ϵ in solving *decision* BDH problem in polynomial time t.

2.3 Necessary Assumptions

Assumption 1. For bilinear map $e:\mathbb{G}\times\mathbb{G}\to\mathbb{G}_T$, in which $\mathbb{G}$ and $\mathbb{G}_T$ are of order $N = p_1p_2p_3p_4$, pick random elements

$$A_1,B_1 \in_R \mathbb{G}_{p_1}, A_2,B_2 \in_R \mathbb{G}_{p_2}, A_3,B_3 \in_R \mathbb{G}_{p_3},$$
$$A_4 \in_R \mathbb{G}_{p_4}, R_{123} \in_R \mathbb{G}_{p_1p_2p_3}, R_{13} \in_R \mathbb{G}_{p_1p_3}$$

thus we get the set $D = (N,\mathbb{G},\mathbb{G}_T,e,A_1,A_3,$ $A_4,B_1A_2,B_2B_3)$. We define that algorithm $\mathcal{B}_1$ has advantage ϵ in breaking Assumption 1 if

$$|\,Pr[\mathcal{B}_1(D,R_{123})=1]-Pr[\mathcal{B}_1(D,R_{13})=1]\,|\le\epsilon$$

Definition 3. Assumption 1 holds in $\mathbb{G}$ if there is no probabilistic polynomial time algorithm $\mathcal{B}_1$ that has advantage at least ϵ in breaking Assumption 1.

Assumption 2. For bilinear map $e:\mathbb{G}\times\mathbb{G}\to\mathbb{G}_T$, in which $\mathbb{G}$ and $\mathbb{G}_T$ are of order $N = p_1p_2p_3p_4$, pick random elements

$$\alpha,s,r \in_R \mathbb{Z}_N, A_1 \in_R \mathbb{G}_{p_1}, A_2,B_2,D_2 \in_R \mathbb{G}_{p_2},$$
$$A_3 \in_R \mathbb{G}_{p_3}, A_4 \in_R \mathbb{G}_{p_4}, R_T \in_R \mathbb{G}_T, R_1 = e(A_1,A_1)^{\alpha s}$$

thus we get the set $D = (N,\mathbb{G},\mathbb{G}_T,e,A_1,A_2,A_3,A_4,$ $A_1^{\alpha}D_2, A_1^sB_2, A_2^r, D_2^r)$. We define that algorithm $\mathcal{B}_2$ has advantage ϵ in breaking Assumption 2 if

$$|\,Pr[\mathcal{B}_2(D,R_1)=1]-Pr[\mathcal{B}_2(D,R_T)=1]\,|\le\epsilon$$

Definition 4. Assumption 2 holds in $\mathbb{G}$ if there is no probabilistic polynomial time algorithm $\mathcal{B}_2$ that has advantage at least ϵ in breaking Assumption 2.

Assumption 3. For bilinear map $e:\mathbb{G}\times\mathbb{G}\to\mathbb{G}_T$, where $\mathbb{G}$ and $\mathbb{G}_T$ are of order $N = p_1p_2p_3p_4$, pick random elements $s,r' \in_R \mathbb{Z}_N, A_1,B_1,D_1 \in_R \mathbb{G}_{p_1}, A_2,B_2,D_2,F_2,$ $J_2 \in_R \mathbb{G}_{p_2}, A_3 \in_R \mathbb{G}_{p_3}, A_4,B_4,D_4,F_4 \in_R \mathbb{G}_{p_4}, A_{24},B_{24},$ $D_{24} \in_R \mathbb{G}_{p_2p_4}, R_{124} \in_R \mathbb{G}_{p_1p_2p_4}, R_1 = B_1^sD_{24}$ thus we get the set $D = (N,\mathbb{G},\mathbb{G}_T,e,A_1,A_2,A_3,A_4,D_1,D_1^sA_{24},D_1^{r'},$ $B_1F_4, B_1^{r'}J_2, A_1^{r'}B_2, A_1^sB_{24})$. We define that algorithm $\mathcal{B}_3$ has advantage ϵ in breaking Assumption 3 if

$$|\,Pr[\mathcal{B}_3(D,R_1)=1]-Pr[\mathcal{B}_3(D,R_{124})=1]\,|\le\epsilon$$

Definition 5. Assumption 3 holds in $\mathbb{G}$ if there is no probabilistic polynomial time algorithm $\mathcal{B}_3$ that has advantage at least ϵ in breaking Assumption 3.

2.4 Zero-Knowledge Interactive Proof

Zero-knowledge interactive proof (ZKIP) system was first proposed by Goldwasser, Micali and Rackoff, which means that the proof yields nothing beyond the validity of the assertion [29]. A ZKIP system has a Prover and a Verifier with common input x. Prover has a secrete input ω as a witness. For language L, after interaction between Prover and Verifier, if $x \in L$, Verifier is convinced with overwhelming probability. Thus this condition is defined as completeness. Otherwise, if $x \notin L$, no matter what Prover does, Verifier would not be fooled with overwhelming probability. And this condition is defined as soundness. While during this proof interaction, no information except validity is leaked. Concretely, completeness and soundness conditions are defined as follows:

- **Completeness**:
  $\forall x \in L, Pr[V_{P(x,\omega)}(x) = accept] \ge 1-\epsilon$
- **Soundness**: $\forall P, \forall x \notin L, Pr[V_{P(x,\omega)}(x) = accept] \le \epsilon$

To take an instance, we describe a three-move protocol based on discrete logarithm problem. For common input (p,g,x), in which p is prime and g is a generator, Prover picks auxiliary input ω as his witness, and computes $g^{\omega} = x \pmod p$.

- Prover picks random $r \in_R Z_p^*$, and computes $y = g^r \pmod p$. Prover sends y to Verifier.
- On receiving y, Verifier replies to Prover with a random bit $b \in_R \{0,1\}$
- If $b=0$, Prover sends $z=r$. If $b=1$, Prover sends $z = r+\omega \pmod{p-1}$
- Verifier accepts if $g^z = x^b y \pmod p$

3 HIBGE: Modelling and Definitions

3.1 Hierarchical Identity-Based Group Encryption

In this section, we propose a new cryptographic primitive named Hierarchical Identity-Based Group Encryption (HIBGE) cryptosystem as described in Figure 1. The HIBGE cryptosystem is composed of following five parties:

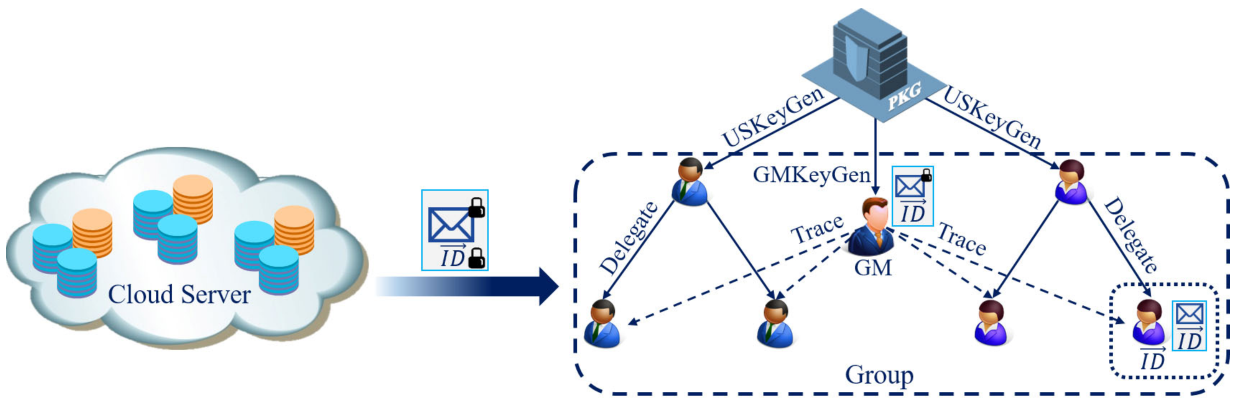

Figure 1. System model

- *Private Key Generator (PKG)* can generate a private key for a group member corresponding to his identity vector $\overrightarrow{ID}$.
- *Group Manager (GM)* can administrate the group members and trace the message recipient if there is something delinquent.
- *Member* can be divided into two roles relatively, an higher-level member and a lower-level member, but the role is not absolute. A higher-level member can delegate private key to the lower-level member, and thus the higher-level member has the ability to monitor the lower-level member's message. All the message recipients are in the group, and the GM will trace the message to find its recipients if necessary.
- *Senders* can be in the group or not. A sender can use the message recipient's identity vector $\overrightarrow{ID}$ as the public key to encrypt his message. And the sender can be cloud server in cloud computing.
- *Verifiers* can prove the identity vector used to encrypt message is identical to the identity vector hiding in the ciphertext, and the GM can play the verifier's role by verify the ciphertext occasionally to ensure its validity.

With these five parties, the HIBGE scheme works by following seven procedures and a verify protocol:

- $(Params, MSK) \leftarrow$ **Setup**(λ, l) : This is a polynomial algorithm executed by PKG, which takes as input a security parameter λ and the maximum depth of the identity vector l, and outputs the system parameters $Params$ and master key MSK.
- $(GPK, GSK) \leftarrow$ **GMKeyGen**$(Params)$: This is a polynomial algorithm executed by GM, which takes as input the security parameter λ that is the same in the procedure $Setup(\lambda, l)$. And it outputs the GM public key and private key pair (GPK, GSK).
- $(USK_j) \leftarrow$ **USKeyGen**$(Params, MSK, \overrightarrow{ID_j})$: This is a polynomial algorithm executed by PKG, which takes as input the system parameters $Params$, the PKG's master key MSK and the member's identity vector $\overrightarrow{ID_j}$, which is equal to $(ID_1, \ldots, ID_j)$ with $j \le l$, and outputs a private key USK_j.
- $(USK_{j+1}) \leftarrow$ **Delegate**$(Params, \overrightarrow{ID_j}, ID_{j+1}, USK_j)$: This is a polynomial algorithm executed by the higher-level group member whose identity is $\overrightarrow{ID_j}$, which takes as input the system parameters $Params$, the identity vector $\overrightarrow{ID_j}$, the lower-level member identity ID_{j+1} with $j+1 \le l$, and the higher-level member's private key USK_j. And the algorithm finally outputs the lower-level member ID_{j+1}'s private key USK_{j+1}.
- $(C) \leftarrow$ **Encrypt**$(Params, GPK, \overrightarrow{ID_j}, M)$: This is a polynomial algorithm executed by $Sender$, which takes as input the systems parameters $Params$, the Recipient's identity vector $\overrightarrow{ID_j}$ and the GM's public key GPK. Finally the algorithm outputs the ciphertext C.
- $(M) \leftarrow$ **Decrypt**$(Params, C, USK_j)$: This is a polynomial algorithm executed by Recipient, which takes as input the systems parameters $Params$, the Recipient's private key USK_j and the ciphertext C, and outputs the message M.
- $(\overrightarrow{ID}) \leftarrow$ **Trace**(C, GSK) : This is a polynomial algorithm executed by GM, which takes as input the ciphertext C and the GM's private key GSK, and outputs the $\overrightarrow{ID}$ which forms the ciphertext.

The **Verify**(C) protocol is constructed to prove that the identity vector $\overrightarrow{ID}$ used to encrypt message is identical to the $\overrightarrow{ID}$ in ciphertext. This protocol should be a zero-knowledge interaction or non-interaction protocol, which means that Prover verifies the ciphertext is valid but yields nothing else through the whole protocol. The Verifier outputs ***accept*** if the proof is validity, and then the ciphertext can be sent in the group. Otherwise the ciphertext delivery would be ***rejected***.

Definition 6 (Correctness). We say the HIBGE scheme is correct if the following correctness-game returns ***true*** with overwhelming probability.

- $(Params, MSK) \leftarrow Setup(\lambda, l)$
- $(GPK, GSK) \leftarrow GMKeyGen(\lambda)$
- $USK_j \leftarrow USKeyGen(Params, MSK, \overrightarrow{ID_j})$, where $\overrightarrow{ID_j} = (ID_1, \ldots, ID_j)$ and $j \in_R [1, l]$
- $USK_{j+1} \leftarrow Delegate(Params, \overrightarrow{ID_j}, ID_{j+1}, USK_j)$ for $j \in [1, l-1]$
- $C_0 \leftarrow Encrypt(Params, GPK, \overrightarrow{ID_j}, M)$ and C_1 is equal to C_0 except that the recipient's identity in C_1 is tampered.
- Verify that all the following equality are satisfied, and return ***true***.

$$Verify(C_0) = accept, Verify(C_1) = reject$$
$$Decrypt(C_0, Params, USK_j) = M$$
$$Trace(C_0) = \overrightarrow{ID_j}$$

3.2 Security Model for Hierarchical Identity-Based Group Encryption

The security of the HIBGE scheme includes semantic security, anonymity and traceability. We describe the security definition for HIBGE scheme by following two games between an adversary $\mathcal{A}$ and a challenger $\mathcal{B}$.

Semantic security and anonymity. To protect the ciphertext not to be analyzed easily by adversary and to hide the identity information from ciphertext, semantic security and anonymity are necessary. Thus they are defined by the following game.

- **Setup:** The challenger $\mathcal{B}$ runs $Setup$ on input security parameter λ, keeps MSK privately, and forwards system parameters $Params$ to the adversary $\mathcal{A}$.
- **Phase 1:** In this phase, $\mathcal{A}$ can adaptively make $USKeyGen$, $Delegate$ and $Trace$ queries, and then he gets the group member's private key USK_j for identity vector $\overrightarrow{ID_j}$.
- **Challenge:** After Phase 1, adversary $\mathcal{A}$ selects two messages M_0 and M_1 of equal length, and two identities vector $\overrightarrow{ID_0}$ and $\overrightarrow{ID_1}$, which have not appeared in Phase 1 private key queries. Challenger B chooses random $c \in \{0,1\}$, and runs algorithm $Encrypt(Params, M_c, ID_c, GPK)$. And the output of the algorithm is given to $\mathcal{A}$.
- **Phase 2:** It is similar to Phase 1 except that $\overrightarrow{ID}$ should not be a prefix of either $\overrightarrow{ID_0}$ or $\overrightarrow{ID_1}$, and $Trace$ query is not permitted.
- **Guess:** The adversary $\mathcal{A}$ outputs a guess $c' \in \{0,1\}$. If $c' = c$, $\mathcal{A}$ wins the game.

We define the advantage that adversary $\mathcal{A}$ wins the game is $Adv_{\mathcal{A}} = | Pr[c' = c] - \frac{1}{2} |$.

Definition 7. The Hierarchical Identity-Based Encryption scheme is semantic secure and anonymous, if for all polynomial time adversary $\mathcal{A}$ and security parameter λ, the advantage that $\mathcal{A}$ wins the game satisfies :

$$Adv_{\mathcal{A}} = | Pr[c' = c] - \frac{1}{2} | \leq \epsilon$$

Traceability. In case that the message is illegal or there are disputes occurring, the GM can trace the recipient of the message. However, if the adversary uses the recipient's identity vector as public key, but puts other member's identity vector in ciphertext for tracing, the GM will trace a wrong target. Therefore, it is necessary to insure the traceability to prevent the adversary from using other's identity vector in ciphertext. Thus the traceability is defined by following game.

- **Setup:** The challenger $\mathcal{B}$ runs $Setup$ and $GMKenGen$ on input security parameter λ. And $\mathcal{B}$ gives the output $Params$, GSK to challenger $\mathcal{A}'$.
- **Inspect phase:** $\mathcal{A}'$ can adaptively make $GMKeyGen$ query, $USKeyGen$ query, $Delegate$ query, $Encrypt$ query, $Decrypt$ query and $Trace$ query to $\mathcal{B}$, And $\mathcal{B}$ responds all the outputs of these queries. Besides, $\mathcal{A}'$ can play the role as a Prover to interact with Verifier while he makes a Verify query in the zero-knowledge proof system.
- **Output:** $\mathcal{A}'$ encrypts message M and outputs the ciphertext C', which can be decrypted back to M when queried to $Decrypt$. Finally, adversary $\mathcal{A}'$ wins the game if the output of $Trace(C')$ is a wrong identity vector of the recipient.

We define the advantage that adversary $\mathcal{A}'$ wins the game is

$$Adv_{\mathcal{A}'} = Pr[Trace(Encrypt_{\mathcal{A}'}(M, \overrightarrow{ID})) \neq \overrightarrow{ID}]$$

Definition 8. The Hierarchical Identity-Based Encryption scheme is traceable, if for all polynomial time adversary $\mathcal{A}'$ and security parameter λ, the advantage that $\mathcal{A}'$ wins the game satisfies:

$$Adv_{\mathcal{A}'} = Pr[Trace(Encrypt_{\mathcal{A}'}(M, \overrightarrow{ID})) \neq \overrightarrow{ID}] \leq \epsilon$$

4 A Concrete HIBGE Scheme

In this section, we construct the Hierarchical Identity-Based Group Encryption scheme based on the

Public-Key Anonymous HIBE scheme constructed by Caro, Iovino and Persiano [6], and modify it to be anonymous in group encryption and verify protocol. Private key consists of two parts, one for decrypting ciphertext and another for delegate private keys to lower-level group members. Ciphertext also consists of two parts, one for encrypted message and another for encrypted identity vector which ensures traceability. Then we construct a zero-knowledge interactive (and non-interactive) protocol Verify, to ensure the validity of ciphertext.

- ***Setup*(λ,*l*)** : This algorithm picks random prime p_1, p_2, p_3, p_4, the length of which are polynomial of λ. Then it computes integer $N = p_1 p_2 p_3 p_4$. Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of order N, and let $e: \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map. The algorithm picks random $g_1, u_1, v_1, \ldots, v_l \in_R \mathbb{G}_{p_1}$, $g_3 \in_R \mathbb{G}_{p_3}$, $g_4, u_4 \in_R \mathbb{G}_{p_4}$, $\alpha, m_1, \ldots, m_l \in_R \mathbb{Z}_N$, and let $U = u_1 \times u_4$. PKG publishes the system parameters $Params = (N, v_1, \ldots, v_l, g_1, g_3, g_4, e(g_1, g_1)^{\alpha}, U, m_1, \ldots, m_l)$, and keeps master key $MSK = (u_1, \alpha)$.
- **GMKeyGen**(*Params*) : To generate a private key for GM, this algorithm picks random $\beta_{11}, \beta_{12}, \beta_{21}, \beta_{22}, \beta_3 \in_R \mathbb{Z}_N$, and a one-way hash function H. Let $\overline{\mathbb{G}}$ be an Abelian group of prime order p, in which the decision Diffie-Hellman problem is hard, and the algorithm picks random $\overline{g}_1, \overline{g}_2$, $\overline{v}_1, \ldots, \overline{v}_l \in_R \overline{\mathbb{G}}$. Then it computes $X_1 = \overline{g}_1^{\beta_{11}} \overline{g}_2^{\beta_{12}}$, $X_2 = \overline{g}_1^{\beta_{21}} \overline{g}_2^{\beta_{22}}$, $Y = \overline{g}_1^{\beta_3}$. GM keeps $GSK = (\beta_{11}, \beta_{12}, \beta_{21}$, $\beta_{22}, \beta_3)$, and publishes
$$GPK = (\overline{g}_1, \overline{g}_2, X_1, X_2, Y, H, \overline{v}_1, \ldots, \overline{v}_l).$$
- **USKeyGen**($Params, MSK, \overrightarrow{ID_j} = (ID_1, \ldots, ID_j)$): Given MSK and the identity vector of the group member, this algorithm picks random $r_1, r_2 \in_R \mathbb{Z}_N$. Let $k \in \{1,2\}$, and the algorithm picks random $E_{k,1}, E_{k,2}, E_{k,j+1}, \ldots, E_{k,l} \in_R \mathbb{G}_{p_3}$. Then it computes $USK = (S_{k,0}, S_{k,1}, S_{k,j+1}, \ldots, S_{k,l})$ for $k = 1$ and 2.
$$S_{k,0} = g_1^{r_k} E_{k,1},$$
$$S_{1,1} = g_1^{\alpha} (v_1^{ID_1} \ldots v_j^{ID_j} v_{j+1}^{m_{j+1}} \ldots v_l^{m_l} u_1)^{r_1} E_{1,2},$$
$$S_{2,1} = (v_1^{ID_1} \ldots v_j^{ID_j} v_{j+1}^{m_{j+1}} \ldots v_l^{m_l} u_1)^{r_2} E_{2,2},$$
$$S_{k,j+1} = v_{j+1}^{r_k} E_{k,j+1}, \ldots, S_{k,l} = v_l^{r_k} E_{k,l}.$$
When $k = 1$, this part of USK is used to decrypt messages, and when $k = 2$, the other part of USK is used to generate USK for his corresponding lower-level members.
- **Delegate**($Params, \overrightarrow{ID_j}, ID_{j+1}, USK_j$) : In the case that a lower-level member whose identity is ID_{j+1} with $j+1 \le l$, calls for his private key, the corresponding higher-level member, whose the private key is $USK_j = (S'_{k,0}, S'_{k,1}, S'_{k,j+1}, \ldots, S'_{k,l})$ for $k = 1$ and 2, runs the algorithm. Add the ID_{j+1} into higher-level member's identity vector $\overrightarrow{ID_j}$ to be a new identity vector of the lower-level member, which is $\overrightarrow{ID}_{j+1} = (ID_1, \ldots, ID_{j+1})$. The algorithm picks random $r'_1, r'_2 \in_R \mathbb{Z}_N$, and $E_{k,1}, E_{k,2}, E_{k,j+1}, \ldots, E_{k,l} \in_R \mathbb{G}_{p_3}$. Then it computes $USK_{j+1} = (S_{k,0}, S_{k,1}, S_{k,j+1}, \ldots, S_{k,l})$ for $k = 1$ and 2. And the distribution of USK_{j+1} is identical to it of USK_j in which the $\overrightarrow{ID}_{j+1} = (ID_1, \ldots, ID_{j+1})$.
$$S_{1,0} = S'_{1,0} (S'_{2,0})^{r'_1} E_{1,1}, S_{2,0} = (S'_{2,0})^{r'_2} E_{2,1},$$
$$S_{1,1} = S'_{1,1} (S'_{2,1})^{r'_1} ((S'_{1,j+1})(S'_{2,j+1})^{r'_1})^{(ID_{j+1} - m_{j+1})} E_{1,2},$$
$$S_{2,1} = (S'_{2,1})^{r'_2} (S'_{2,j+1})^{r'_2 (ID_{j+1} - m_{j+1})} E_{2,2},$$
$$S_{1,j+2} = S'_{1,j+2} (S'_{2,j+2})^{r'_1} E_{1,j+2}, \ldots, S_{1,l} = S'_{1,l} (S'_{2,l})^{r'_1} E_{1,l},$$
$$S_{2,j+2} = (S'_{2,j+2})^{r'_2} E_{2,j+2}, \ldots, S_{2,l} = (S'_{2,l})^{r'_2} E_{2,l}.$$
- **Encrypt**($Params, GPK, \overrightarrow{ID_j}, M$) : This algorithm picks random $s, n, z_1, z_2 \in_R \mathbb{Z}_N$. If the message is $M \in \mathbb{G}_T$ and the public key is $\overrightarrow{ID_j} = (ID_1, \ldots, ID_j)$, the ciphertext is $C = (C_1, C_2) = (C_{1,0}, C_{1,1}, C_{1,2}, C_{2,0}, C_{2,1}, C_{2,2}, C_{2,3})$. C_1 is used to encrypt message, and C_2 is the encrypted recipient's identity.
$$C_{1,0} = g_1^s g_4^{z_1}, C_{1,1} = (v_1^{ID_1} \ldots v_j^{ID_j} v_{j+1}^{m_{j+1}} \ldots v_l^{m_l} U)^s g_4^{z_2},$$
$$C_{1,2} = Me(g_1, g_1)^{\alpha s}, C_{2,0} = \overline{g}_1^n, C_{2,1} = \overline{g}_2^n,$$
$$C_{2,2} = Y^n (\overline{v}_1^{ID_1} \ldots \overline{v}_j^{ID_j} \overline{v}_{j+1}^{m_{j+1}} \ldots \overline{v}_l^{m_l}),$$
$$C_{2,3} = X_1^n X_2^{nH(C_{2,0} C_{2,1} C_{2,2})}$$
- **Verify**(C) : To verify the encrypted $\overrightarrow{ID}$ is identical to the $\overrightarrow{ID}$ used to encrypt message, this algorithm roles as an interactive or non-interactive zero-knowledge proof protocol. Prover(Sender) proves to Verifier(GM) that he uses the same $\overrightarrow{ID}$ in *Encrypt* algorithm, but leaks nothing else through the whole protocol. The protocol can be denoted as
$$ZK\left\{ n, s, \overrightarrow{ID_j} \,\middle|\, C_{1,0}, C_{1,1}, C_{2,0}, C_{2,1}, C_{2,2}, C_{2,3} \right\}$$
And we utilize intermediate values L_1, L_2 to make

the protocol easier, and thus the zero-knowledge proof is converted into

$$ZK\left\{n,s,\overrightarrow{ID_j}\ \middle|\ \begin{matrix} C_{1,0},C_{1,1},C_{2,0},C_{2,1},C_{2,2},C_{2,3}, \\ L_1=C_{2,2}^{s},L_2=C_{2,0}^{s}=\overline{g}_1^{ns} \end{matrix}\right\}$$

Then the interactive protocol is as follows.

1. Prover picks random $\overline{n},\overline{s}$, $\overline{z}_1$, $\overline{z}_2$ and $\overline{ID_i}\in_R \mathbb{Z}_N$, for $i=1,\dots,l$, and runs the algorithm *Encrypt* again to get the ciphertext $\overline{C}$ except that $\overline{C}_{2,3}$ $=X_1^{\overline{n}}X_2^{\overline{n}H(C_{2,0}C_{2,1}C_{2,2})}$. Then he let $\overline{L}_1=Y^{\overline{ns}}(\overline{v}_1^{\overline{ID}_1}\dots\overline{v}_l^{\overline{ID}_l})^{\overline{s}}$, and $\overline{L}_2=\overline{C}_{2,0}^{\overline{s}}=\overline{g}_1^{\overline{ns}}$. Prover sends $(\overline{C},\overline{L}_1,\overline{L}_2)$ to Verifier, but keeps the random integers privately.

2. Verifier picks random $\omega\in_R\mathbb{Z}_N$, and sends it to Prover.

3. Prover computes $\varphi_1=\overline{s}+\omega s \pmod N$, $\varphi_2=\overline{n}+\omega n \pmod N$. For $i=1,\dots,l$, $\varphi_{3,i}=ID_i\omega+\overline{ID_i} \pmod N$, $\varphi_{4,i}=\overline{s}\,\overline{ID_i}+\omega sID_i \pmod N$, and note that $ID_i=m_i$ for $i>j$. $\varphi_5=\overline{ns}\ +\omega ns \pmod N$, $\varphi_6=\overline{z}_1+\omega z_1$, $\varphi_7=\overline{z}_2+\omega z_2$. Prover sends it back to Verifier.

4. Finally, Verifier checks whether all the following equations are satisfied.

$$g_1^{\varphi_1}g_4^{\varphi_6}=C_{1,0}^{\omega}\overline{C}_{1,0},U^{\varphi_1}g_4^{\varphi_7}v_1^{\varphi_{4,1}}\dots v_l^{\varphi_{4,l}}=C_{1,1}^{\omega}\overline{C}_{1,1},$$
$$\overline{g}_1^{\varphi_2}=C_{2,0}^{\omega}\overline{C}_{2,0},\overline{g}_2^{\varphi_2}=C_{2,1}^{\omega}\overline{C}_{2,1},$$
$$Y^{\varphi_2}\overline{v}_1^{\varphi_{3,1}}\dots\overline{v}_l^{\varphi_{3,l}}=C_{2,2}^{\omega}\overline{C}_{2,2},$$
$$(X_1X_2^{H(C_{2,0}C_{2,1}C_{2,2})})^{\varphi_2}=C_{2,3}^{\omega}\overline{C}_{2,3},$$
$$Y^{\varphi_5}\overline{v}_1^{\varphi_{4,1}}\dots\overline{v}_l^{\varphi_{4,l}}=L_1^{\omega}\overline{L}_1,\overline{g}_1^{\varphi_5}=L_2^{\omega}\overline{L}_2$$

If all the equations are satisfied after checked, Verifier outputs ***accept***, otherwise ***reject***. This algorithm outputs ***accept*** if and only if the $\overrightarrow{ID}$ which is used to encrypt message is the same to the $\overrightarrow{ID}$ which is encrypted in the ciphertext and can be used in algorithm *Trace*.

Besides, this protocol can be converted into a non-interactive zero-knowledge proof by using a collision-resistant hash function $\overline{H}$. The non-interactive protocol is the same as the interactive protocol except that ω in stage 2 is generated by Prover and $\omega=\overline{H}(C,\overline{C},\overline{L}_1,\overline{L}_2)$.

- **Decrypt**$(Params,C,USK_j)$: The C_1 part of ciphertext C is used in this algorithm. Recipient uses the private key USK_j to compute and get the message.

$$M=\frac{C_{1,2}e(S_{1,0},C_{1,1})}{e(S_{1,1},C_{1,0})}$$

- **Trace**(C,GSK): The GM can trace the Recipient if the message is illegal by running this algorithm. If the algorithm *Verify* outputs ***accept***, GM computes

$$\overline{v}_1^{ID_1}\dots\overline{v}_l^{ID_l}=\frac{C_{2,2}}{C_{2,0}^{\beta_3}}$$

Then GM computes $\overline{v}_i^{ID_i}$ for all ID_i and finds the $\overrightarrow{ID}^*$ that satisfies $\overline{v}_1^{ID_1^*}\dots\overline{v}_l^{ID_l^*}=\overline{v}_1^{ID_1}\dots\overline{v}_l^{ID_l}$. Finally, it outputs $\overrightarrow{ID}^*$ who is the ciphertext Recipient.

5 Correctness and Security of HIBGE Scheme

5.1 Correctness of HIBGE scheme

We use the *Definition 6* to prove this scheme satisfies correctness. Firstly, the message M can be decrypted correctly from ciphertext for

$$\frac{C_{1,2}e(S_{1,0},C_{1,1})}{e(S_{1,1},C_{1,0})}$$
$$=\frac{Me(g_1,g_1)^{\alpha s}e(g_1^{r_1}E_{1,1},(v_1^{ID_1}\dots v_j^{ID_j}v_{j+1}^{m_{j+1}}\dots v_l^{m_l}U)^s g_4^{z_2})}{e(g_1^{\alpha}(v_1^{ID_1}\dots v_j^{ID_j}v_{j+1}^{m_{j+1}}\dots v_l^{m_l}u_1)^{r_1}E_{1,2},g_1^{s}g_4^{z_1})}$$
$$=\frac{Me(g_1,g_1)^{\alpha s}e(g_1,(v_1^{ID_1}\dots v_j^{ID_j}v_{j+1}^{m_{j+1}}\dots v_l^{m_l}))^{r_1 s}}{e(g_1,g_1)^{\alpha s}e(v_1^{ID_1}\dots v_j^{ID_j}v_{j+1}^{m_{j+1}}\dots v_l^{m_l},g_1)^{r_1 s}}=M$$

Secondly, the Recipient can be traced correctly for

$$C_{2,0}^{\beta_{11}}C_{2,1}^{\beta_{12}}=X_1^{n},C_{2,0}^{\beta_{21}}C_{2,1}^{\beta_{22}}=X_2^{n},C_{2,0}^{\beta_3}=Y^{n}$$

Thus this algorithm outputs $\overline{g}_3^{H(ID)}=\dfrac{C_{2,2}}{Y^n}$, if

$$C_{2,0}^{\beta_{11}+\beta_{21}H(C_{2,0}C_{2,1}C_{2,2})}C_{2,1}^{\beta_{12}+\beta_{22}H(C_{2,0}C_{2,1}C_{2,2})}=C_{2,3}$$

5.2 Zero-Knowledge Proof

The *Verify* algorithm is a zero-knowledge proof protocol, and we sketch the proof of it by proving three properties: completeness, soundness and zero-knowledge. And we proved that the protocol is perfectly zero-knowledge.

- **Completeness:** The completeness of this protocol is trivial.
- **Soundness:** We construct a knowledge extractor *M*, which has complete control over Prover and stops in expected polynomial time. Thus the probability that *M* outputs a string of valid φ_i is the same as the probability that a possible cheating Prover' convinces a truthful Verifier. However, *M* computes φ_i with no knowledge of random $\overline{s},\overline{n},\overline{ID_i},\overline{z}_1,\overline{z}_2$.

This implies that M would generate valid φ_i accepted by Verifier with negligible probability so as the Prover'. The soundness of M demonstrates the soundness of the underlying protocol.

- **Zero-knowledge:** We construct a simulator M, which simulate the whole protocol without knowledge of $s, n, \overrightarrow{ID}, z_1, z_2$. M has complete control over the possibly cheating Verifier. M guesses a challenge ω', and chooses random φ_i' for $i = 1, \dots, 7 \in_R \mathbb{Z}_N$. M sends φ_i' to Verifier'. If M succeeds, he repeats the protocol, each time with different random challenges ω'. Note that regardless of the value of ω', the distribution of φ_i' are the same as the distribution of message sent by truthful Prover. Thus Verifier's replay in the protocol is independent of ω'. Therefore, the probability that the simulation with ω' is equal to the probability in a real execution of protocol. Thus the output of the simulation procedure is perfectly indistinguishable from a real execution of the protocol, even with a cheating Verifier.

5.3 Semantic Security and Anonymity of HIBGE

Security of Hierarchical Identity-Based Group Encryption scheme includes semantic security, anonymity and traceability. As introduced in section 3, we prove semantic security and anonymity by running game based on *Definition* 7, and we prove traceability by running game based on *Definition* 8. Referring to the proof made by Angolo, Vincenzo and Giuseppe [6], we prove semantic security and anonymity of HIBGE by following versions of game $q + 7$ times, in which adversary $\mathcal{A}$ makes q key queries to challenger $\mathcal{B}$.

Game$_1$: is an original HIBGE scheme.

Game$_2$: is the same to $game_1$ except that the C_2 part of ciphertext is generated at random.

Claim 1: For any algorithm $\mathcal{A}$, $Adv_{\mathcal{A}}(game_1) = Adv_{\mathcal{A}}(game_2)$.

Proof. Suppose that H is a one-way hash function, and $n, \beta_1, \beta_2, \beta_3, \beta_4 \in_R \mathbb{Z}_N, \bar{g}_1, \bar{g}_2, \bar{g}_3 \in_R \bar{G}$ are chosen randomly. Then the distribution of C_2 is uniform, which means $game_1$ and $game_2$ are the same in adversary's view.

Game$_3$: is the same to $game_2$ except that each key query is sent to a new key generation algorithm, in which way the *Delegate* algorithm will not run anymore.

Claim 2: For any algorithm $\mathcal{A}$, $Adv_{\mathcal{A}}(game_2) = Adv_{\mathcal{A}}(game_3)$.

Proof. Suppose that all the random parameters in both *USKenGen* and *Delegate* are chosen randomly, then the private keys generated by both algorithms distribute identically, which means $game_2$ and $game_3$ are the same in adversary's view.

Game$_4$: is the same to $game_3$ except that adversary $\mathcal{A}$ is not permitted to make a key query $\overrightarrow{ID}^*$, which is a prefix of previous key query $\overrightarrow{ID} \pmod{p_2}$.

Claim 3: If there is a probabilistic polynomial algorithm $\mathcal{A}$ such that $Adv_{\mathcal{A}}(game_3) - Adv_{\mathcal{A}}(game_4) = \epsilon$, there must be a probabilistic polynomial algorithm $\mathcal{B}$ breaking Assumption 1 with advantage $Adv_{\mathcal{B}}(Assumption1) \geq \frac{\epsilon}{3}$.

Proof. Suppose that there is a probabilistic polynomial algorithm $\mathcal{A}$, such that $Adv_{\mathcal{A}}(game_3) - Adv_{\mathcal{A}}(game_4) = \epsilon$, which means $\mathcal{A}$ would make a key query $\overrightarrow{ID}^*$ that is a prefix of previous key query $\overrightarrow{ID} \pmod{p_2}$ with probability ϵ. Thus there must exist a query $ID_i^* \neq ID_i \pmod{N}$, but $p_2 \mid (ID_i^* - ID_i)$. Let $a = \gcd((ID_i^* - ID_i), N)$, thus $a \in \{p_1p_2, p_2p_3, p_2p_4, p_1p_2p_3, p_1p_2p_4, p_2p_3p_4\}$. Let $b = N / a$, thus the situation can be divided into three cases, and each occurs with probability at least $\frac{\epsilon}{3}$.

- $p_1 \mid b$
- $p_1 \nmid b, p_4 \mid b$
- $p_1 \nmid b, p_4 \nmid b, p_3 \mid b$

Taking case 1 as an instance, we can thus run algorithm $\mathcal{B}$ to break Assumption 1. $\mathcal{B}$ gets the set $D = (N, \mathbb{G}, \mathbb{G}_T, e, A_1, A_3, A_4, B_1A_2, B_2B_3)$ and R, and runs the *Setup* algorithm with $g_1 = A_1, g_3 = A_3$, $g_4 = A_4$, to get *MSK* and *Params*. Then $\mathcal{B}$ runs $\mathcal{A}$, and replies to $\mathcal{A}$'s key queries according *MSK*. Finally, $\mathcal{B}$ computes $a = \gcd((ID_i^* - ID_i), N)$. If $e(B_1A_2^a, B_2B_3) = 1$, then $\mathcal{B}$ tests whether $e(B_1A_2, R^b) = 1$. If the equation is satisfied, $\mathcal{B}$ would declare that $R \in \mathbb{G}_{p_1p_2}$, else $R \in \mathbb{G}_{p_1p_2p_3}$. This method is also applicable to case 2 and case 3 with little modification that g_1 and g_4 are exchanged in *Params* for case 2, or g_1 and g_3 are exchanged in *Params* for case 3.

Game$_{5,i}$: is the same to $game_4$ except that for i from 0 to q, the ciphertext sent to $\mathcal{A}$ is semi-functional ciphertext and the first i private keys are semi-functional keys [19].

- **Semi-functional Ciphertext:** We define $C_{1,0} = g_2^{x_1} C'_{1,0}, C_{1,1} = g_2^{x_1 x_c} C'_{1,1}, C_{1,2} = C'_{1,2}$ are semi-functional ciphertext in which we randomly choose $x_1, x_c \in_R \mathbb{Z}_N$ and g_2 is a generator of group $\mathbb{G}_{p_2}$, and C' is ciphertext encrypted by original encryption algorithm.

- **Semi-functional Key:** For randomly chosen $z_1, z_2, z_{3,j+1}, z_{4,j+1}, z_k \in_R \mathbb{Z}_N$ for j from 1 to ℓ, and private keys generated by original key generation algorithm, we define semi-functional key as follows:

$$S_{1,0} = g_2^{z_1} S'_{1,0}, S_{1,1} = g_2^{z_1 z_k} S'_{1,1},$$
$$S_{1,j+1} = g_2^{z_1 z_{3,j+1}} S'_{1,j+1}, \ldots, S_{1,l} = g_2^{z_1 z_{3,l}} S'_{1,l},$$
$$S_{2,0} = g_2^{z_1 z_2} S'_{2,0}, S_{2,1} = g_2^{z_1 z_2 z_k} S'_{2,1},$$
$$S_{2,j+1} = g_2^{z_1 z_2 z_{4,j+1}} S'_{2,j+1}, \ldots, S_{2,l} = g_2^{z_1 z_2 z_{4,l}} S'_{2,l}$$

Using these semi-functional definition, the decryption will also valid if $x_c = z_k$.

Claim 4: If there is a probabilistic polynomial algorithm $\mathcal{A}$ such that $Adv_{\mathcal{A}}(game_4) - Adv_{\mathcal{A}}(game_{5,0}) = \epsilon$, there must be a probabilistic polynomial algorithm $\mathcal{B}$ breaking Assumption 1 with advantage $Adv_{\mathcal{B}}(Assumption1) \geq \epsilon$.

Proof. $\mathcal{B}$ gets the set $D = (N, \mathbb{G}, \mathbb{G}_T, e, A_1, A_3, A_4, B_1A_2, B_2B_3)$ and R. $\mathcal{B}$ picks random $\alpha, a, b, c_1, \ldots, c_l \in_R \mathbb{Z}_N$, and runs the *Setup* algorithm with $g_1 = A_1, g_3 = A_4, g_4 = A_3, u_4 = g_4^a, u_1 = g_1^b, v_j = g_1^{c_j}$ for j from 1 to ℓ to get *MSK* and *Params*. $\mathcal{B}$ runs $\mathcal{A}$ and reply to all key queries from $\mathcal{A}$. Finally, $\mathcal{A}$ sends two pairs to $\mathcal{B}$ (M_0, ID_0) and (M_1, ID_1), and $\mathcal{B}$ randomly chooses $\beta \in_R \{0,1\}$, and encrypts the message by computing $C_{1,0} = R, C_{1,1} = R^{ID_{\beta,1}c_1 + \ldots + ID_{\beta,j}c_j + b}$, $C_{1,2} = M_\beta e(R, g_1)^\alpha$. Thus we can see that if $R \in \mathbb{G}_{p_1p_2p_3}$, R can be denoted by $g_1^{r_1} g_2^{r_2} g_3^{r_3}$ and C_1 is a semi-functional ciphertext. Otherwise, if $R \in \mathbb{G}_{p_1p_3}$, R can be denoted by $g_1^{r_1} g_3^{r_3}$ and C_1 is a normal ciphertext.

Claim 5: If there is a probabilistic polynomial algorithm $\mathcal{A}$, such that the advantage $\mathcal{A}$ wins the game $Adv_{\mathcal{A}}(game_{5,i-1}) - Adv_{\mathcal{A}}(game_{5,i}) = \epsilon$, there must be a probabilistic polynomial algorithm $\mathcal{B}$ breaking Assumption 1 with advantage $Adv_{\mathcal{B}}(Assumption1) \geq \epsilon$.

Proof. $\mathcal{B}$ gets the set $D = (N, \mathbb{G}, \mathbb{G}_T, e, A_1, A_3, A_4, B_1A_2, B_2B_3)$ and R. $\mathcal{B}$ picks random $\alpha, a, b, c_1, \ldots, c_l \in_R \mathbb{Z}_N$, and runs the *Setup* algorithm with $g_1 = A_1, g_3 = A_3$, $g_4 = A_4, u_4 = g_4^a$, $u_1 = g_1^b, v_j = g_1^{c_j}$ for j from 1 to ℓ, to get *MSK* and *Params*. $\mathcal{B}$ runs $\mathcal{A}$ and reply to all key queries from $\mathcal{A}$. For k-th key query, if $k > i$, private key is generated by original algorithm. Otherwise, if $k < i$, $\mathcal{B}$ generates semi-functional key and gives it to $\mathcal{A}$. This semi-functional key is generated by following method: For $n \in \{1,2\}$, $\mathcal{B}$ picks random $r_n, f, z_2, \omega, \omega_{n,2}$, $\omega_{n,j+1}, \ldots, \omega_{n,l} \in_R \mathbb{Z}_N$, and sets $z_1 = 1, x_1 = 1, x_2 = 0$.

$$S_{n,0} = g_1^{r_k} (B_2B_3)_n^z f,$$
$$S_{n,1} = g_1^\alpha (B_2B_3)^\omega (v_1^{ID_{k,1}} \ldots v_j^{ID_{k,j}})^{r_n} g_3^{\omega_{n,2}},$$
$$S_{n,2} = v_{j+1}^{r_n} (B_2B_3)^{\omega_{n,j+1}}, \ldots, S_{n,l} = v_l^{r_n} (B_2B_3)^{\omega_{n,l}}$$

If $k = i$, for $n \in \{1,2\}$, $\mathcal{B}$ sets $z_k = c_1 ID_{k,1} + \ldots + c_j ID_{k,j} + b$. $\mathcal{B}$ randomly chooses $r'_2, \omega_{n,2}$, $\omega_{n,j+1}, \ldots, \omega_{n,l} \in_R \mathbb{Z}_N$, and computing private key in following method:

$$S_{1,0} = R, S_{1,1} = g_1^\alpha R^{z_k} g_3^{\omega_{1,2}},$$
$$S_{1,j+1} = R^{c_{j+1}} g_3^{\omega_{1,j+1}}, \ldots, S_{1,l} = R^{c_l} g_3^{\omega_{1,l}},$$
$$S_{2,0} = R^{r'_2}, S_{2,1} = R^{r'_2 z_k} g_3^{\omega_{2,2}},$$
$$S_{2,j+1} = R^{r'_2 c_{j+1}} g_3^{\omega_{2,j+1}}, \ldots, S_{2,l} = R^{r'_2 c_l} g_3^{\omega_{2,l}}$$

Thus we can see that if $R \in \mathbb{G}_{p_1p_2p_3}$, R can be denoted by $g_1^{r_1} g_2^{r_2} g_3^{r_3}$ and the key is a semi-functional key. Otherwise, if $R \in \mathbb{G}_{p_1p_3}$, R can be denoted by $g_1^{r_1} g_3^{r_3}$ and $S_{n,0}, \ldots, S_{n,l}$ is a normal private key.

Finally, $\mathcal{A}$ sends (M_0, ID_0) and (M_1, ID_1) to $\mathcal{B}$. And $\mathcal{B}$ randomly chooses $\beta \in_R \{0,1\}$ and $z, z' \in_R \mathbb{Z}_N$, and encrypts the message by computing

$$C_{1,0} = B_1A_2 g_4^{z'}, C_{1,1} = (B_1A_2)^{ID_{\beta,1}c_1 + \ldots + ID_{\beta,j}c_j + b},$$
$$C_{1,2} = M_\beta e(B_1A_2, g_1)^\alpha$$

Since there is no $\overrightarrow{ID}^*$ that is a prefix of the $\overrightarrow{ID}$ above, thus $z_c = ID_{\beta,1}c_1 + \ldots + ID_{\beta,j}c_j + b \pmod{p_2}$ and z_k is independent and distributed uniformly. Then if $\mathcal{B}$ tests whether t-th key is semi-functional key with the help of semi-functional ciphertext, we will get $z_c = z_k$, and decrypt the ciphertext. Therefore, if $\mathcal{A}$ wins the $game_{5,i-1}$, $\mathcal{B}$ will break Assumption 1 that $R \in \mathbb{G}_{p_1p_3}$. Otherwise, if $\mathcal{A}$ wins the $game_{5,i}$, $\mathcal{B}$ will break Assumption 1 that $R \in \mathbb{G}_{p_1p_2p_3}$.

Game$_6$: is the same to $Game_{5,q}$ except that the ciphertext $C_{1,2}$ is generated at random in $\mathbb{G}_T$.

Claim 6: If there is a probabilistic polynomial algorithm $\mathcal{A}$ such that $Adv_{\mathcal{A}}(game_{5,q}) - Adv_{\mathcal{A}}(game_6) = \epsilon$, there must be a probabilistic polynomial algorithm $\mathcal{B}$ breaking Assumption 2 with advantage $Adv_{\mathcal{B}}(Assumption2) \geq \epsilon$.

Proof. $\mathcal{B}$ gets the set $D = (N, \mathbb{G}, \mathbb{G}_T, e, A_1, A_2, A_3, A_4, A_1^\alpha D_2, A_1^s B_2, A_2^r, D_2^r)$ and R. $\mathcal{B}$ picks random $\alpha, a, b, c_1, \ldots, c_l \in_R \mathbb{Z}_N$, and runs the *Setup* algorithm with $g_1 = A_1, g_3 = A_3, g_4 = A_4, u_4 = g_4^a, u_1 = g_1^b, v_j = g_1^{c_j}$ for j from 1 to l, to get MSK and $Params$, in which $e(g_1, g_1)^\alpha = e(A_1^\alpha A_2, g_1)$.

To reply to $\mathcal{A}$'s key query, $\mathcal{B}$ generates semi-functional key and gives it to $\mathcal{A}$. This semi-functional key is generated by following method:

For $n \in \{1,2\}$, $\mathcal{B}$ picks random $r_n, z, z', \omega_{n,1}, \omega_{n,2}, \omega_{n,j+1}, \ldots, \omega_{n,l}, z_{n,j+1}, \ldots, z_{n,l} \in_R \mathcal{Z}_N$, and sets

$$V_1 = A_2^z, V_1' = A_1^\alpha D_2 A_2^{z'}, V_2 = V_1^r, V_2' = D_2^r A_2^{rz'}.$$
$$S_{n,0} = g_1^{r_n} V_n g_3^{\omega_{n,1}}, S_{n,1} = V_n'(v_1^{ID_1} \ldots v_j^{(ID_j)} u_1)^{r_n} g_3^{\omega_{n,2}}$$
$$S_{n,j+1} = v_{j+1}^{r_n} A_2^{z_{n,j+1}} g_3^{\omega_{n,j+1}}, \ldots, S_{n,l} = v_l^{r_n} A_2^{z_{n,l}} g_3^{\omega_{n,l}}$$

Finally, $\mathcal{A}$ sends (M_0, ID_0) and (M_1, ID_1) to $\mathcal{B}$. And $\mathcal{B}$ randomly chooses $\beta \in_R \{0,1\}$ and $z, z' \in_R \mathbb{Z}_N$, and encrypts the message by computing

$$C_{1,0} = A_1^s B_2 g_4^{z'}, C_{1,1} = (A_1^s B_2)^{ID_{\beta,1}c_1 + \ldots + ID_{\beta,j}c_j + b}, C_{1,2} = M_\beta R$$

Thus we can see that if $\mathcal{B}$ declares the truth that $R = e(A_1, A_1)^{\alpha s}$, $\mathcal{A}$ would win the game to declare that the $C_{1,2}$ is a semi-functional ciphertext for message M_β. Otherwise, if $\mathcal{B}$ declares the truth that R is a random element of $\mathbb{G}_T$, $\mathcal{A}$ would win the game to declare that the $C_{1,2}$ is a semi-functional ciphertext for a random message.

Game$_7$: is the same to $Game_6$ except that the ciphertext $C_{1,1}$ is generated at random in $\mathbb{G}_{p_1 p_2 p_4}$.

Claim 7: If there is a probabilistic polynomial algorithm $\mathcal{A}$ such that $Adv_{\mathcal{A}}(game_6) - Adv_{\mathcal{A}}(game_7) = \epsilon$, there must be a probabilistic polynomial algorithm $\mathcal{B}$ breaking Assumption 3 with advantage $Adv_{\mathcal{B}}(Assumption3) \geq \epsilon$.

Proof. $\mathcal{B}$ gets the set $D = (N, \mathbb{G}, \mathbb{G}_T, e, A_1, A_2, A_3, A_4, D_1, D_1^s A_{24}, D_1^{r'}, B_1 F_4, B_1^{r'} J_2, A_1^{r'} B_2, A_1^s B_{24})$ and R. $\mathcal{B}$ picks random $\alpha, c_1, \ldots, c_l \in_R \mathbb{Z}_N$, and runs the *Setup* algorithm with $g_1 = A_1, g_3 = A_3, g_4 = A_4, U = B_1 F_4$, $v_j = D_1^{c_j}$ for j from 1 to ℓ, to get MSK and $Params$. To reply to $\mathcal{A}$'s key query, $\mathcal{B}$ generates semi-functional key and gives it to $\mathcal{A}$. This semi-functional key is generated by following method: For $n \in \{1,2\}$, $\mathcal{B}$ picks random $r_n', z_{n,j+1}, \ldots, z_{n,l}, \omega_{n,1}, \omega_{n,2}, \omega_{n,j+1}, \ldots, \omega_{n,l} \in_R \mathcal{Z}_N$, and sets $V_1 = g_1^\alpha, V_2 = 1$.

$$S_{n,0} = (A_1^{r'} B_2)^{r_n'} g_3^{\omega_{n,1}},$$
$$S_{n,1} = V_n((D_1^{r'})^{c_1 ID_1 + \ldots + c_j ID_j}(B_1^{r'} J_2))^{r_n'} g_3^{\omega_{n,2}}$$
$$S_{n,j+1} = (D_1^{r'})^{r_n' c_{j+1}} g_2^{z_{n,j+1}} g_3^{\omega_{n,j+1}}$$
$$\ldots\ldots$$
$$S_{n,l} = (D_1^{r'})^{r_n' c_l} g_2^{z_{n,l}} g_3^{\omega_{n,l}}$$

Finally, $\mathcal{A}$ sends (M_0, ID_0) and (M_1, ID_1) to $\mathcal{B}$. And $\mathcal{B}$ randomly chooses $\beta \in_R \{0,1\}$ and $z, z' \in_R \mathbb{Z}_N$, and encrypts the message by computing $C_{1,0} = A_1^s B_{24}, C_{1,1} = R(D_1^s A_{24})^{c_1 ID_1 + \ldots + c_j ID_j}, C_{1,2} \in_R \mathbb{G}_T$.

Thus we can see that if $\mathcal{B}$ declares the truth that $R = B_1^s D_{24}$, $\mathcal{A}$ would win the game to declare that the $C_{1,1}$ is a semi-functional ciphertext for identity vector ID_β. Otherwise, if $\mathcal{B}$ declares the truth that R is a random element of $\mathbb{G}_{p_1 p_2 p_4}$, $\mathcal{A}$ would win the game to declare that the $C_{1,0}$ and $C_{1,1}$ are semi-functional ciphertext at random in $\mathbb{G}_{p_1 p_2 p_4}$.

Claim 8: If Assumption 1, Assumption 2 and Assumption 3 hold, the HIBGE scheme is semantic security and anonymous.

Proof. From claim 1 to claim 7 we can see that, it's indistinguishable between $game_j$ and $game_{j+1}$, for j from 1 to 6. Therefore, the real game is indistinguishable with $Game_7$. Since in $Game_7$, all ciphertext are generated at random, adversary $\mathcal{A}$ has no advantage in winning the game. Thus in the real game, for all polynomial time adversary $\mathcal{A}$, the advantage that $\mathcal{A}$ wins the game satisfies:

$$Adv_{\mathcal{A}} = |Pr[c' = c] - \frac{1}{2}| \leq \epsilon$$

5.4 Traceability of HIBGE Scheme

Adversary $\mathcal{A}$ and challenger $\mathcal{B}$ run the traceability game defined in section 3. To win the game, $\mathcal{A}$ encrypts C_1 part with public key $\overrightarrow{ID}$, and encrypts C_2 part with public key $\overrightarrow{ID}'$. The proof is accepted if and only if

$$\overline{s}(\overline{v}_1^{\overline{ID}_1} \ldots \overline{v}_l^{\overline{ID}_l}) + \omega s(\overline{v}_1^{ID_1} \ldots \overline{v}_l^{ID_l})$$
$$= \overline{s}(\overline{v}_1^{\overline{ID}'_1} \ldots \overline{v}_l^{\overline{ID}'_l}) + \omega s(\overline{v}_1^{ID'_1} \ldots \overline{v}_l^{ID'_l}) \pmod N$$

Thus the random ω can satisfy this equation with negligible probability. Therefore, the HIBGE scheme is traceability.

6 Performance Analysis

6.1 Theoretical Analysis

We analyze the efficiency of the proposed scheme in this section. As shown in Table 2, for composite-order groups $\mathbb{G}_N$ where $N = p_1 p_2 p_3 p_4$, we denote ℓ_N as the binary length of elements in group $\mathbb{G}_N$, j as the length of the recipient's identity vector, τ_e as one exponent operation time, τ_m as one multiplication operation, τ_p as one pairing operation time, and τ_h as one hash operation time. It can be seen that the length and the time expended is linearly with the max level of this scheme. Though the verification time is much longer than the others, it is just occasionally executed by the group manager.

Table 2. Theoretical performance

(N,ℓ)-HIBGE	
Order	$N = p_1p_2p_3p_4$
$Params$ Size	$(2\ell+5)\ell_N$
GPK Size	$(\ell+6)\ell_N$
USK Size	$2(\ell-j+2)\ell_N$
USKeyGen Time	$(3\ell+5-j)\tau_m+(3\ell+4-j)\tau_e$
Encryption Time	$(2\ell+5)\tau_m+(2\ell+10)\tau_e+\tau_h$
Decryption Time	$2\tau_p+2\tau_m$
Verification Time	$(11\ell+25)\tau_m+(7\ell+36)\tau_e+2\tau_h$

6.2 Experimental Analysis

We implement the proposed HIBGE scheme with Java Pairing-Based Cryptography Library (JPBC) [30] to evaluate its performance. The experiment is run on 3.7GHz Inter Xeon E5-1620 v2 platform with 16GB RAM running 64-bit Windows 10 Professional operation system. The elliptic curve is Type A1 with $y^2 = x^3 + x$ for Tate symmetric pairings. The order of the group is $N = p_1p_2p_3p_4 = 3 \bmod 4$ and the primes are 512-bit large randomly picked by the system. We run the scheme for 20 rounds and calculate the average execution time it takes.

As shown in Figure 2, we test the time consumptions required for setup and group setup, and both the time consumptions are under 10s, which is executed only once at the beginning of the system. We test the time consumptions required for secret key generation with the recipient's level ranging from 1 to 10, and for secret key delegation with the recipient's level ranging from 2 to 10, and both the time consumptions are decreasing with the recipient's level, because the length of the secret key is decreasing with the recipient's level. Then we test the time consumptions required for encryption and decryption with the recipient's level ranging from 1 to 10, and the time consumption of decryption is only about 600ms, which means that it does not require any high performance devices of recipients. Finally, we test the time consumptions required for tracing with the recipient's maximal level ranging from 1 to 10, which is executed only when disputes occurring, and for verification with the recipient's level ranging from 1 to 10, which is executed only occasionally.

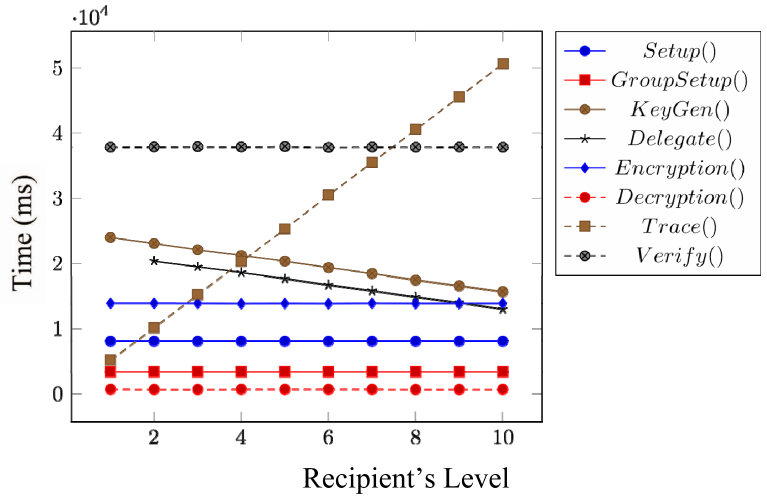

Figure 2. Experimental performance

Above all, the time consumptions of algorithms that executed by third parties are high, while the time consumptions of decryption that executed by recipients are relatively low. Thus our proposed scheme is suitable for actual deployment with high performance requirement of third parties.

7 Conclusion

In this paper, we proposed a new primitive, referred to as Hierarchical Identity-Based Group Encryption. This cryptosystem allows anonymous and traceable group communication and reduces the demand of bandwidth and computing burden of a single PKG. We defined the security model for HIBGE and we constructed a concrete HIBGE scheme according to this and we proved this scheme is semantic secure and anonymous, and it leaks no information in verifying procedure other than validity. Besides, this scheme is traceable if needed. HIBGE scheme is applicable to many existing application scenarios, and an application of this scheme has been published in [31] to achieve secure and anonymous data transmission in the space information network.

Acknowledgments

This paper is supported by the National Key R&D Program of China under grant No. 2017YFB1400700, by the Natural Science Foundation of China through projects 61672083, 61370190, 61532021, 61702028, 61472429, and 61402029, by the National Cryptography Development Fund through project MMJJ20170106, by the planning fund project of ministry of education through project 12YJAZH136 and by the Beijing Natural Science Foundation through project 4132056.

References

[1] A. Kiayias, Y. Tsiounis, M. Yung, Group Encryption, Advances in Cryptology - ASIACRYPT 2007, *13th International Conference on the Theory and Application of Cryptology and Information Security*, Kuching, Malaysia, 2007, pp. 181-199.

[2] A. Shamir, Identity-based Cryptosystems and Signature Schemes, *Advances in Cryptology, Proceedings of CRYPTO '84*, Santa Barbara, CA, 1984, pp. 47-53.

[3] D. Boneh, X. Boyen, Efficient Selective-id Secure Identity-based Encryption without Random Oracles, *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques*, Interlaken, Switzerland, 2004, pp. 223-238.

[4] X. Luo, Y. Ren, J. Liu, J. Hu, W. Liu, Z. Wang, W. Xu, Q. Wu, Identity-based Group Encryption, *Information Security and Privacy - 21st Australasian Conference, ACISP 2016*, Melbourne, Australia, 2016, pp. 87-102.

[5] C. Gentry, A. Silverberg, Hierarchical Id-based Cryptography, *Advances in Cryptology - ASIACRYPT 2002, 8th International Conference on the Theory and Application of Cryptology and Information Security*, Queenstown, New Zealand, 2002, pp. 548-566.

[6] A. D. Caro, V. Iovino, G. Persiano, Fully Secure Anonymous HIBE and Secret-key Anonymous IBE with Short Ciphertexts, *Pairing-Based Cryptography - Pairing 2010 4th International Conference*, Yamanaka Hot Spring, Japan, 2010, pp. 347-366.

[7] P. Paillier, Public-key Cryptosystems Based on Composite Degree Residuosity Classes, *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques*, Prague, Czech Republic, 1999, pp. 223-238.

[8] J. Camenisch, A. Lysyanskaya, A Signature Scheme with Efficient Protocols, *Security in Communication Networks, Third International Conference, SCN 2002*, Amalfi, Italy, 2002, pp. 268- 289.

[9] U. Feige, *Alternative Models for Zero Knowledge Interactive Proofs*, Department of Computer, 1990.

[10] B. Qin, Q. Wu, W. Susilo, Y. Mu, Publicly Verifiable Privacy-preserving Group Decryption, *Information Security and Cryptology, 4th International Conference, Inscrypt 2008*, Beijing, China, 2008, pp. 72-83.

[11] B. Libert, M. Yung, M. Joye, T. Peters, Traceable Group Encryption, *Public-Key Cryptography - PKC 2014 - 17th International Conference on Practice and Theory in Public-Key Cryptography*, Buenos Aires, Argentina, 2014, pp. 592-610.

[12] D. Boneh, X. Boyen, Secure Identity Based Encryption without Random Oracles, *Advances in Cryptology - CRYPTO 2004, 24th Annual International Cryptology Conference*, Santa Barbara, CA, 2004, pp. 443-459.

[13] C. Gentry, Practical Identity-based Encryption without Random Oracles, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, St. Petersburg, Russia, 2006, pp. 445-464.

[14] B. Waters, Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference*, Santa Barbara, CA, 2009, pp. 619-636.

[15] D. Boneh, A. Raghunathan, G. Segev, Functionprivate Identity-based Encryption: Hiding the Function in Functional Encryption, *Advances in Cryptology CRYPTO 2013 - 33rd Annual Cryptology Conference*, Santa Barbara, CA, 2013, pp. 461-478.

[16] X. Chen, F. Zhang, W. Susilo, H. Tian, J. Li, K. Kim, Identity-based Chameleon Hashing and Signatures without Key Exposure, *Inf. Sci.*, Vol. 265, pp. 198-210, May, 2014.

[17] X. Boyen, B. Waters, Anonymous Hierarchical Identitybased Encryption (without random oracles), *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference*, Santa Barbara, CA, 2006, pp. 290-307.

[18] L. Ducas, Anonymity from Asymmetry: New Constructions

for Anonymous HIBE, *Topics in Cryptology - CT-RSA 2010, The Cryptographers' Track at the RSA Conference 2010*, San Francisco, CA, 2010, pp. 148-164.

[19] A. B. Lewko, B. Waters, New Techniques for Dual System Encryption and Fully Secure HIBE with Short Ciphertexts, *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010*, Zurich, Switzerland, 2010, pp. 455-479.

[20] L. Xiong, Z. Xu, High-capacity and Lossless Reversible Data Hiding for Encrypted Multimedia Data in Cloud Computing, *Journal of Internet Technology*, Vol. 19, No. 4, pp. 1187-1193, July, 2018.

[21] Y. Zhang, J. Li, X. Chen, H. Li, Anonymous Attribute-based Proxy Re-encryption for Access Control in Cloud Computing, *Security and Communication Networks*, Vol. 9, No. 14, pp. 2397-2411, July, 2016.

[22] J. Shen, T. Zhou, D. He, Y. Zhang, X. Sun, Y. Xiang, Block Design-based Key Agreement for Group Data Sharing in Cloud Computing, *IEEE Transactions on Dependable and Secure Computing*, July, 2017.

[23] X. Chen, J. Li, J. Weng, J. Ma, W. Lou, Verifiable Computation over Large Database with Incremental Updates, *IEEE Trans. Computers*, Vol. 65, No. 10, pp. 3184-3195, December, 2015.

[24] X. Chen, J. Li, X. Huang, J. Ma, W. Lou, New Publicly Verifiable Databases with Efficient Updates, *IEEE Trans. Dependable Sec. Comput.*, Vol. 12, No. 5, pp. 546-556, October, 2014.

[25] J. Shen, J. Shen, X. Chen, X. Huang, W. Susilo, An Efficient Public Auditing Protocol with Novel Dynamic Structure for Cloud Data, *IEEE Trans. Information Forensics and Security*, Vol. 12, No. 10, pp. 2402-2415, May, 2017.

[26] Y. Li, Y. Yu, G. Min, W. Susilo, J. Ni, K. K. R. Choo, Fuzzy Identity-based Data Integrity Auditing for Reliable Cloud Storage Systems, *IEEE Transactions on Dependable and Secure Computing*, Vol. PP, No. 99, pp. 1-1, February, 2017.

[27] L. Wu, Y. Zhang, K. R. Choo, D. He, Efficient and Secure Identity-based Encryption Scheme with Equality Test in Cloud Computing, *Future Generation Comp. Syst.*, vol. 73, pp. 22-31, August, 2017.

[28] D. Boneh, E. Goh, K. Nissim, Evaluating 2-dnf Formulas on Ciphertexts, *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005*, Cambridge, MA, 2005, pp. 325-341.

[29] S. Goldwasser, S. Micali, C. Rackoff, The Knowledge Complexity of Interactive Proof-systems (extended abstract), *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, Providence, Rhode Island, 1985, pp. 291-304.

[30] A. De Caro, V. Iovino, jpbc: Java Pairing Based Cryptography, *Proceedings of the 16th IEEE Symposium on Computers and Communications, ISCC 2011*, Kerkyra, Corfu, Greece, 2011, pp. 850-855.

[31] D. Li, J. Liu, W. Liu, Secure and Anonymous Data Transmission System for Cluster Organised Space Information Network, *2016 IEEE International Conference on Smart Cloud, SmartCloud 2016*, New York, NY, 2016, pp. 228-233.

Biographies

Dawei Li was born in Shandong. He received the B.S. degree from Beihang University, Beijing, China in 2015. He is currently working toward the Ph.D. degree in Electronic and Information Engineering, Beihang University, Beijing, China. His research interests include applied cryptography and blockchain.

Jianwei Liu was born in Shandong. He received his Ph.D. degree in Communication and Electronic System from Xidian University Shaanxi, China in 1998. He is now a Professor of Cyber Science and Technology at Beihang University, Beijing, China. His current research interests include cryptography, and information and network security.

Qianhong Wu was born in SiChuan. He received the Ph.D. degree in Cryptography from Xidian University, Shanxi, China in 2004. He is now a Professor of Cyber Science and Technology at Beihang University, Beijing, China. His current research interests include cryptography, data security and privacy, and blockchain.

Zhenyu Guan received the Ph.D. degree in Electronic Engineering from Imperial College London, UK in 2013. Since then, he has joined Beihang University (China) as a Lecturer. He is a member of IEEE and IEICE. His current research interests include cryptography engineering, security of IOT, blockchain.