Almost all existing hierarchical identity-based encryption (HIBE) schemes fully secure in the standard model at present have a drawback that at least one ciphertext size or private key size must rely on the hierarchy depth of identity. This drawback increases the computation and communication cost. In order to solve the problem, a new HIBE scheme with high efficiency is proposed, which has constant size ciphertext and private key, i.e. both ciphertext size and private key size are independent of the level of the hierarchy. What’s more, the proposed scheme is fully secure in the standard model with a tight reduction. To the best of our knowledge, it is the first scheme that both ciphertext and private key achieve $O(1)$ size with full security in the standard model.

Keywords: HIBE, standard model, identity-based encryption, selective-id model, constant size, provable security

1. INTRODUCTION

Identity (ID) based cryptosystems first were introduced by Shamir [1] which simplify key management procedures in certificate-based public key setting. The main idea of ID-based cryptosystems is that the identity information of each user works as his/her public key. In other words, the user’s public key can be calculated directly from his/her identity instead of being extracted from a certificate issued by a certificate authority (CA). Private Key Generator (PKG) is responsible for computing the private key of each user based on his public key. However, using a single PKG is not practical in large scale, so Gentry-Silverberg [2] and Horwitz-Lynn [3] extended ID-based cryptography to hierarchical ID-based (HIB) cryptography. In a hierarchical ID-based cryptosystem, multiple PKGs are used and are arranged in a hierarchical (tree) structure. The root PKG generates private keys for its children PKGs which are responsible for generating private keys for the next level of PKGs. The PKGs in the leaves is responsible for generating private keys for users in the corresponding domain. Hence users can be divided into different domains, which reduce the workload for the root PKG, especially in a large community. Due to the hierarchical property of hierarchical identity-based encryption (HIBE), it is applied in many areas where there are hierarchical administrative issues, such as large companies or e-government systems. Recently HIBE also is applied to Health Record
The first efficient construction of HIBE was given by Gentry and Silverberg [2] in the random oracles. Boneh and Boyen [6] presented the first construction of HIBE without random oracles based on decision BDH (Bilinear Diffie-Hellman). So far, many HIBE schemes were proposed [7-15] without random oracles. However, Zhang et al. [16] pointed out that all previous HIBE schemes have a drawback that the private key or the ciphertext depends on the hierarchy or the maximum hierarchy in either. This drawback directly increases the computation cost, communication cost of the senders, and storage cost of the users. In order to overcome this drawback, Zhang et al. [16] changed the master key to two parts (master private key and shared private key) and proposed a HIBE that the ciphertext size as well as the private-key size is independent of the hierarchy depth and the maximum hierarchy. However, the HIBE scheme of Zhang et al. is only IND-D-CPA (indistinguishability against adaptive identity and adaptive chosen plaintext attack) secure in the selective identity model and the security reduction is not tight. So, in this paper, based on Zhang et al.’s HIBE scheme and Gentry’s IBE scheme [17], we will present a new HIBE that is IND-ID-CCA (chosen ciphertext attack) fully secure in the standard model with a tight reduction. And the ciphertext size (always only four group elements) as well as the private-key size (always only three group elements) is independent of the hierarchy depth and the maximum hierarchy. It is a desirable feature since it is the first HIBE scheme that achieves $O(1)$ size of ciphertext and private key with full security in the standard model.

The rest of the paper is organized as follows. In section 2, we give some preliminaries. In section 3, we present the construction and the security proof of our HIBE scheme, and make the efficiency comparison with previous HIBE schemes. In section 4, we conclude this paper.

2. PRELIMINARIES

2.1 Hierarchical Identity-Based Encryption (HIBE)

Let $l$ denote the maximum hierarchy of a HIBE scheme. An $l$-level HIBE scheme consists of four algorithms: Set-Up, Key Generation, Encryption and Decryption.

**Set-Up:** On input a security parameter $k$, it outputs $(params, msk)$, where $msk$ is the master key and $params$ is the public parameter.

**Key Generation:** On input an identity vector $ID$ (where $|ID| \leq l$) and $(params, msk)$, it outputs a private key $d_{ID}$ of $ID$.

**Encryption:** On input an identity $ID$ (where $|ID| \leq l$), a message $M$ and $params$, namely $(ID, M, params)$, it outputs a ciphertext $C$ on $M$.

**Decryption:** On input the private key $d_{ID}$ of $ID$ (where $|ID| \leq l$), a ciphertext $C$ and the public parameter $params$, namely $(d_{ID}, C, params)$, it outputs the plaintext message $M$ of $C$ or bad if the ciphertext $C$ is invalid.
2.2 Secure Model

IND-ID-CCA: the security model of a HIBE is defined as the indistinguishability against adaptive identity and adaptive chosen ciphertext attack for HIBE (IND-ID-CCA), and it is an interactive game between an adversary and a simulator as follows:

**Setup:** The simulator generates system public parameter params and sends params to the adversary.

**Query Phase 1:** The adversary adaptively issues queries \( q_1, q_2, \ldots, q_m \), where each \( q_i \) (\( 1 \leq i \leq m \)) is one of the following:

- **Key Generation Query:** The adversary issues a private key query for identity \( ID \) (where \( |ID| \leq l \)). The simulator generates a private key \( d_{ID} \) of \( ID \) and sends \( d_{ID} \) to the adversary.
- **Decryption Query:** The adversary issues a ciphertext \( C_i \) on an identity \( ID \) (where \( |ID| \leq l \)). The simulator generates the plaintext \( M_i \) corresponding to \( C_i \) and sends \( M_i \) to the adversary.

**Challenge Phase:** The adversary submits two plaintexts \( M_0, M_1 \in G \) and an identity \( ID^* \) (where \( |ID^*| \leq l \)). The simulator picks randomly \( b \in \{0, 1\} \), sets the challenge ciphertext \( C^* = \text{Encryption}(params, ID^*, Mb) \) and sends \( C^* \) to the adversary.

**Query Phase 2:** The adversary issues queries adaptively as Phase 1 with the restriction that it cannot query \( (C^*, ID^*) \) to the Decryption Query.

**Guess:** Finally, the adversary outputs a guess \( b' \). The adversary wins the above game if \( b' = b \) and \( ID^* \) or any of its proper prefixes had never been queried to the Key Generation Query in Phases 1 and 2.

We call the adversary in the above game an IND-ID-CCA adversary. The advantage of the adversary in this game is defined as \( \text{Adv}_A = |\Pr[b' = b] - \frac{1}{2}| \).

We say that a HIBE scheme is \( (t, \varepsilon, q_E, q_d) \)-IND-ID-CCA secure, if no \( t \)-time adversary has at least \( \varepsilon \) advantage in winning the above game with making at most \( q_E \) key generation queries and at most \( q_d \) decryption queries.

If we restrict that the adversary can only make key generation query in the above game, a HIBE scheme is said to be \( (t, \varepsilon, q_d) \)-IND-ID-CPA secure, namely no \( t \)-time adversary has at least \( \varepsilon \) advantage in winning the above game with making at most \( q_d \) key generation queries.

2.3 Bilinear Map

Let \( G \) and \( G_T \) be two groups whose orders are a prime \( p \), and \( g \) be a generator of \( G \). A mapping \( e: G \times G \to G_T \) is called a bilinear map if it satisfies the following properties:

(1) **Bilinearity:** For all \( u, v \in G \) and \( a, b \in \mathbb{Z}_p \), \( e(u^a, v^b) = e(u, v)^{ab} \).
(2) Non-degeneracy: $e(g, g) \neq 1$;
(3) Computability: For any $u, v \in G$, there exists an efficient algorithm to compute $e(u, v)$.

2.4 Hardness Assumption

The security of our HIBE scheme is based on a hardness assumption called truncated decision $q$-ABDHE (augmented bilinear Diffie-Hellman exponent) assumption proposed by Gentry in [13, 17].

Given a tuple of $q + 4$ elements $(g, g^\alpha, g^{\alpha^2}, \ldots, g^{\alpha^q}, g', g'^{\alpha^{q+2}}, Z) \in G^{q+3} \times G_T$ as input, output 1 if $Z = e(g_{q+1}, g')$ or output 0. An algorithm $B$ has advantage $\epsilon$ to solve the truncated decision $q$-ABDHE problem if

$$\left| \Pr[B(g, g_1, \ldots, g_q, g', g'^{\alpha^{q+2}}, Z) = 0] - \Pr[B(g, g_1, \ldots, g_q, g', g'^{\alpha^{q+2}}, W) = 0] \right| \geq \epsilon,$$

where $g_i, g'_i$ denote $g^{\alpha_i}, g'^{\alpha_i}$ and the probability is over the random choice of generators $g, g' \in G, \alpha \in Z_p^*$, $Z \in G_T$ and the random bits consumed by $B$.

We say the truncated decision $(t, \epsilon, q)$-ABDHE assumption holds in $G$ and $G_T$ if no $t$-time algorithm has at least $\epsilon$ advantage in solving the truncated decision $q$-ABDHE problem in $G$ and $G_T$.

3. THE PROPOSED SCHEME

In this section, we firstly describe our HIBE scheme in detail. Next we make the security and efficiency analysis of our scheme.

3.1 Our Scheme

3.1.1 Our HIBE scheme

Let $G$ be a group generated by $g$ whose order is a prime $p$, $l$ be the maximum depth of a HIBE and $ID_k = (v_1, v_2, \ldots, v_l)(1 \leq k \leq l)$ denote a $k$th level identity, where $v_i = (v_{i1}, v_{i2}, \ldots, v_{in})(1 \leq i \leq k)$ and $n$ is the bit number of every identity $v_i$. Let $v_j \in \{0, 1\}(1 \leq j \leq n)$. Our HIBE scheme consists of four phases.

Set-Up Phase: Pick randomly $\alpha, r_0 \in Z_p, h_0 \in G$ and set $g_1 = g^\alpha$. Choose randomly $\alpha_1, \alpha_2, \ldots, \alpha_n, \beta_1, \ldots, \beta_n \in Z_p$ and set $a_{i1} = g^{\alpha_{i1}}, a_{i2} = g^{\alpha_{i2}}, \ldots, a_{in} = g^{\alpha_{in}}, b_{i1} = g^{\beta_{i1}}, \ldots, b_{in} = g^{\beta_{in}}$, where $1 \leq i \leq l$.

At the $(i - 1)$th level, PKG$_{i-1}(1 \leq i \leq l)$ is given the shared master key $(\alpha_1, \alpha_2, \ldots, \alpha_n, \beta_1, \beta_2, \ldots, \beta_n)$, where $i \leq j \leq l$.

$\alpha$ is the master key that is only known by the root PKG (we refer to the root PKG as PKG$_0$, namely depth 0), and the public parameter is $\text{params} = \{g, g_1, h_0, a_{i1}, \ldots, a_{in}, b_{i1}, \ldots, b_{in}\}$, where $1 \leq i \leq l$.

Key Generation Phase: Assume that $ID_k = (v_1, v_2, \ldots, v_l)(1 \leq k \leq l)$ is an identity for
which a private key is required, where \( v_i = (v_{i1}, v_{i2}, \ldots, v_{in}) (1 \leq i \leq k) \) and \( v_j \in \{0, 1\} (1 \leq j \leq n) \). A private key for \( ID_k \) is generated by the following steps:

1. Define a function
   \[
   h_w = \prod_{j=1}^{n} g^{a_{ij}v_j} = \prod_{j=1}^{n} a_{ij}^{b_{ij}^{1-v_j}}, \text{ where } 1 \leq i \leq k.
   \]

2. Choose \( r_1, r_2 \in \mathbb{Z}_p \) and compute the private key \( d_{ID_k} = (d_0, d_1, d_2, d_3) \) for identity \( ID_k \):
   \[
   d_0 = (h_0g^{-r_1})^{\pi_{ij}} (\prod_{i=1}^{n} h_w^{v_i})^{\frac{1}{2}} = (h_0g^{-r_1})^{-\pi_{ij}} (\prod_{i=1}^{n} h_w^{v_i})^{\frac{1}{2}}.
   \]
   \[
   d_1 = g^{r_2}, \quad d_2 = r_1, \quad d_3 = (g^{r_3})^{\beta}.
   \]

   The private key \( d_{ID_k} = (d_0, d_1, d_2, d_3) \) of \( ID_k \) also can be derived by its parent \( PKG_{i-1} \) with \( ID_{k-1} = (v_1, v_2, \ldots, v_{k-1}) \) using its private key \( d_{ID_{k-1}} = (d'_0, d'_1, d'_2, d'_3) \) and its shared master key \( (\alpha_1, \alpha_2, \alpha_3, \alpha_4, \beta_1, \ldots, \beta_n) \):

1. Define a function
   \[
   T(v_k) = \sum_{j=1}^{n} (\alpha_j v_j + \beta_j (1-v_j)).
   \]

2. Compute
   \[
   d_0 = d'_0 \cdot (d'_i)^{T(v_k)}
   \]
   \[
   = (h_0g^{-r_1})^{\pi_{ij}} (\prod_{i=1}^{n} h_w^{v_i})^{\frac{1}{2}} \cdot g^{r_2} \sum_{j=1}^{n} (\alpha_j v_j + \beta_j (1-v_j))
   \]
   \[
   = (h_0g^{-r_1})^{\pi_{ij}} (\prod_{i=1}^{n} h_w^{v_i})^{\frac{1}{2}} g^{(\alpha_1 v_1 + \ldots \alpha_n v_n + \beta_1 (1-v_1) + \ldots + \beta_n (1-v_n)) r_2}
   \]
   \[
   = (h_0g^{-r_1})^{\pi_{ij}} (\prod_{i=1}^{n} h_w^{v_i})^{\frac{1}{2}} (\prod_{i=1}^{n} a_{ij}^{b_{ij}^{1-v_j}})^{\frac{1}{2}}
   \]
   \[
   = (h_0g^{-r_1})^{\pi_{ij}} (\prod_{i=1}^{n} h_w^{v_i})^{\frac{1}{2}} (h_{k2})^{r_2}
   \]
   \[
   d_1 = d'_1, \quad d_2 = d'_2, \quad d_3 = d'_3.
   \]

   Then, \( d_{ID_k} = (d_0, d_1, d_2, d_3) \) is the private key of \( ID_k \).

**Encryption Phase:** Assume that \( M \) is a message and \( ID_k = (v_1, v_2, \ldots, v_l) (1 \leq k \leq l) \) is an identity on which the encryption is required. The ciphertext of \( M \) on the identity \( ID_k \) is generated as follows: pick randomly \( s \in \mathbb{Z}_p \) and compute

\[
C = (C_1, C_2, C_3, C_4)
\]
\[
= (g_1^{s}, e(g, g)^v, M \cdot e(g, g)^{-v}, (\prod_{i=1}^{n} h_m)^{r_1}).
\]

where
\[
(\prod_{i=1}^{n} h_m)^{r_1} = \prod_{i=1}^{n} (\prod_{j=1}^{n} a_{ij}^{v_j} h_j^{1-v_j})^{r_1}.
\]
Decryption Phase: After getting a ciphertext \( C = (C_1, C_2, C_3, C_4) \) on an identity \( ID_k \) and a message \( M \), the plaintext \( M \) of \( C \) can be recovered by using the private key \( d_{ID_k} = (d_0, d_1, d_2, d_3) \) of \( ID_k \):

\[
M = C_3 \cdot e(C_2, d_4) \cdot C_2 \cdot e(d_3, C_2)
\]

\[
= M_1 \cdot e(g, h_0)^{-r} \cdot e(g, g^{-a_0}) \cdot e((g, g^{-a_0}) \cdot (\prod_{i=1}^l h_i)^{\gamma}) \cdot e(g, g)^{a_0}
\]

\[
= M_1 \cdot e(g, h_0)^{-r} \cdot e(g, h_0^{-a_0}) \cdot e((g, g^{-a_0}) \cdot (\prod_{i=1}^l h_i)^{\gamma}) \cdot e(g, g)^{a_0}
\]

\[
= M_1 \cdot e(g, h_0)^{-r} \cdot e(g, h_0^{-a_0}) \cdot e((g, g^{-a_0}) \cdot (\prod_{i=1}^l h_i)^{\gamma}) \cdot e(g, g)^{a_0}
\]

\[
= M_1 \cdot e(g, h_0)^{-r} \cdot e(g, h_0^{-a_0}) \cdot e((g, g^{-a_0}) \cdot (\prod_{i=1}^l h_i)^{\gamma}) \cdot e(g, g)^{a_0}
\]

\[
= M
\]

So, \( C = (C_1, C_2, C_3, C_4) \) is a correctness ciphertext.

3.1.2 Forward-secure and backward-secure HIBE scheme

Note that in most of existing HIBE schemes (such as [2, 3, 6-15]), A PKG is able to derive the private keys for its children nodes and all its descendant nodes, although there is no literature to explicitly make “A PKG is able to derive the private keys for its children nodes and all its descendant nodes” as an essential property of a HIBE. In order to hold the property, in the Set-Up Phase of our scheme, the system firstly sets up shared master keys for all 1st to \( l \)th level, and next the \((i-1)\)th level’s PKG \( P_{i-1} \) \((1 \leq i \leq l)\) is issued with all the shared master keys \((\alpha_{j1}, \alpha_{j2}, \ldots, \alpha_{jn}, \beta_{j1}, \beta_{j2}, \ldots, \beta_{jn}) \) \((i \leq j \leq l)\) of its children nodes and all descendant nodes. After getting these shared master keys, PKG \( P_{i-1} \) can repeatedly use them to derive the private keys for all the \( i \)th to \( l \)th level according to Key Generation Phase.

However, we think that “A PKG is able to derive the private keys for its children nodes and all its descendant nodes” is not a very good property. Let us consider the following case: The private key of the \((i-1)\)th level’s PKG (namely PKG\(_{i-1}\), where \(1 \leq i \leq l\)) is exposed accidentally. It is obvious that a malicious user who gets the private key of PKG\(_{i-1}\) can derive the private keys for PKG\(_{i-1}\)’s children nodes and all PKG\(_{i-1}\)’s descendant nodes. So, in order to security, the private keys of the \( i \)th to \( l \)th level’s PKG (namely the private keys of the children nodes and all descendant nodes of PKG\(_{i-1}\)) must all be discarded. And even all the children nodes and all descendant nodes of PKG\(_{i-1}\) can’t be used as private key generator for users in the corresponding domain. In other words, HIBE schemes with the property “A PKG is able to derive the private keys for its children nodes and all its descendant nodes” are only forward-secure (disclosure of a domain PKG’s secret does not compromise the secrets of higher-level PKGs) but are not backward-secure (disclosure of a domain PKG’s secret does not compromise the secrets of lower-level PKGs).
It is easy to make our scheme satisfy both forward-security and backward-security. In order to do so, we only need to modify the first phase “Set-Up Phase” of our scheme, and let “… At the \((i - 1)\)th level, PKG\(_i\) (1 ≤ \(i\) ≤ \(l\)) is given the shared master key \((\alpha_{i1}, \alpha_{i2}, ..., \alpha_{in}, \beta_{i1}, \beta_{i2}, ..., \beta_{in})\)…” replace the original “… At the \((i - 1)\)th level, PKG\(_i\) (1 ≤ \(i\) ≤ \(l\)) is given the shared master key \((\alpha_{i1}, \alpha_{i2}, ..., \alpha_{in}, \beta_{i1}, \beta_{i2}, ..., \beta_{in})\), where \(i ≤ j ≤ l\), …”. In other words, PKG\(_i\) (1 ≤ \(i\) ≤ \(l\)) is only issued with the shared master key of the \(i\)th level. As PKG\(_i\) only holds the shared master key \((\alpha_{i1}, \alpha_{i2}, ..., \alpha_{in}, \beta_{i1}, \beta_{i2}, ..., \beta_{in})\) of its direct children nodes, PKG\(_i\) can only derive the private keys for its direct children nodes not all its descendant nodes. Therefore, even if the private key of the \((i - 1)\)th level’s PKG (PKG\(_i\)) is exposed accidentally, it doesn’t affect the private keys of any other levels including all ancestor nodes and all descendant nodes of the \((i - 1)\)th level’s PKG.

So, from the security of the private key of each PKG in a HIBE scheme, “A PKG can only derive the private keys for its direct children nodes not all its descendant nodes” is better than “A PKG is able to derive the private keys for its children nodes and all its descendant nodes”.

### 3.1.3 Potential application

The above forward-secure and backward-secure HIBE scheme is still essentially a HIBE. So like other HIBEs, the new HIBE scheme can be used in most environments where the structure/organization of the system is in hierarchy, e.g. large companies or e-government systems, etc. where there are hierarchical administrative issues. Or maybe it can be used in other potential situations in future. Next, we will present a simple example to illustrate how to use our HIBE scheme. The example first was shown in [3], and here we modify this example to be suitable for our scheme.

Assume that there is a university (named AZ) with three hierarchically administrative structures, and its hierarchy is school (first level), department (second level) and teacher (third level) from high to low.

**PKG Distribution**  
In the first level (namely school level), there are domain PKGs-1, who can request their domain keys from the root PKG. In the second level (namely department level), there are domain PKGs-2, who can request their domain keys from their parent domain PKGs-1. Lastly, there are teachers (third level), who can request private keys from their parent domain PKGs-2. Each teacher and each domain has a primitive ID (PID), which is an arbitrary string. Such as a teacher Alice works for network department of computer school i.e. network.computer.AZ and her email address is alice@network.computer.AZ, her PID is alice and her department’s PID is network, her school’s PID is computer.AZ. The public key of a teacher consists of a tuple of PIDs: the PID of the teacher, the PID of the teacher’s upper domain PKGs-2 and the PID of the domain PKGs-2’s upper domain PKGs-1 (the public key is also called the user’s address). As with IBE systems, it is clear that a sender can derive the receiver’s public key offline. It is easy to expand three levels to more levels by allowing subdomains, subsubdomains, and so on.

Next, we use our HIBE to the above instance.

**Parameter Generation and Key Issue**  
There is a root PKG (the 0th level), who gen-
erates the system parameter $\text{params}$, a master key and all shared master keys $\alpha_1, \alpha_2, \ldots, \alpha_{pn}$, $\beta_1, \beta_2, \ldots, \beta_{pn} \in \mathbb{Z}_p$, where $1 \leq i \leq 3$. Publish the system parameter $\text{params}$ publicly.

For the $(i-1)$th level (where $2 \leq i \leq 3$), the root PKG sends the shared master key $(\alpha_1, \alpha_2, \ldots, \alpha_{pn}, \beta_1, \beta_2, \ldots, \beta_{pn})$ to PKGs-1 by a secure channel, namely domain PKGs-1 of school (first level) gets the shared master key $(\alpha_{21}, \alpha_{22}, \ldots, \alpha_{2n}, \beta_{21}, \beta_{22}, \ldots, \beta_{2n})$ and domain PKGs-2 of department (second level) gets the shared master key $(\alpha_{31}, \alpha_{32}, \ldots, \alpha_{3n}, \beta_{31}, \beta_{32}, \ldots, \beta_{3n})$.

Running The following steps (including key generation, encryption and decryption) will be in accordance with the proposed HIBE scheme. For example, Alice (from network department of computer school, her public key is her email address alice@network.computer.AZ) requests a private key from its parent domain PKGs-2. With PKGs-2’s shared master key and PKGs-2’s private key that has previously requested from the PKGs-1, PKGs-2 uses Key Generation Phase to generate a private key and sends the private key to Alice. Alice can use the private key to decrypt any ciphertext encrypted by her public key alice@network.computer.AZ.

As PKGs-1 and PKGs-2 only hold the shared master key of its direct children nodes respectively, PKGs-1 or PKGs-2 can only derive the private keys for its direct children nodes not all its descendant nodes. Therefore, even if the private key of the $(i - 1)$th level’s PKG is exposed accidentally, it doesn’t affect the private keys of any other levels including all ancestor nodes and all descendant nodes of the $(i - 1)$th level’s PKG. Such as the private key of the PKGs-1 is exposed accidentally, it doesn’t affect the private keys of PKGs-0 and the teachers.

Therefore, our HIBE is not only forward-secure but also backward-secure.

It is easy to expand the above three levels to more levels by allowing subdomains, subsubdomains, and so on. The use method of our HIBE scheme in more levels (more than three levels) is similar to the above process.

3.2 Security Analysis

Theorem 1 Assume that the truncated decision $(\varepsilon, t, q)$-ABDHE assumption holds in $G$, the proposed HIBE scheme is $(\varepsilon', t', q_{E}, q_{d})$ IND-ID-CCA secure with

$$q > q_{E}, \quad q > q_{d}, \quad t < t' + O((l + q)q \tau), \quad \varepsilon' = \varepsilon,$$

where $q_{E}$ is the number of the private key generation queries and $\tau$ is the time for an exponentiation in $G$.

Proof: Assume that there is a $(\varepsilon', t', q_{E}, q_{d})$ – adversary $A$ to break the proposed scheme. Then, we will use $A$ to construct an algorithm $B$ that solves the truncated decision $q$-ABDHE problem with probability at least $\varepsilon$ and time at most $t$.

$B$ takes a random truncated decision $q$-ABDHE challenge $(g', g'^{\alpha}, g, g_1, \ldots, g_p, Z)$, where $Z$ is either $e(g_{q_{E}+1}, g')$ or a random element of $G_T$, $g_i = g'^{\alpha}$, and $g'_i = g_i^{\alpha}$. $B$ doesn’t know the value $\alpha$. To be able to use $A$ to solve the problem, $B$ must be able to simulate a simulator for $A$. Such a simulation can be created as follows.
(1) Setup Phase: B generates a random polynomial \( f(x) \in \mathbb{Z}_p[x] \) of degree \( q \). It sets \( h_0 = g^{f(0)} \) that \( h_0 \) can be computed from \((g, g_1, \ldots, g_q)\). B chooses randomly \( r_0, a_1, a_2, \ldots, a_n, \beta_1, \beta_2, \ldots, \beta_n \in \mathbb{Z}_p \), where \( 1 \leq i \leq l \).

If \( r_0 = \alpha \), B uses \( \alpha \) to solve the truncated decision \( q \)-ABDHE problem immediately. Else, B sets \( a_1 = (g_1 g_2 \cdots g_q)^{b_1} = g^{(a_1 b_1) a_2}, a_2 = (g_2 g_3 \cdots g_q)^{b_2} = g^{(a_2 b_2) a_3}, \ldots, a_n = (g_n g_{n+1} \cdots g_q)^{b_n} = g^{(a_n b_n) a_1} \), where \( 1 \leq i \leq l \).

The master key is \( \alpha \) that is not known to B, and the shared master key in the \((i - 1)\)th level is \((\alpha_0, \alpha_2, \ldots, \alpha_n, \beta_1, \beta_2, \ldots, \beta_n)(i \leq j \leq l)\) that is known to B. The public parameter \( \text{params} = \{g, g_1, r_0, h_0, a_1, \ldots, a_n, b_1, \ldots, b_n\} \) is sent to A, where \( 1 \leq i \leq l \).

(2) Query Phase 1: In this phase, A is allowed to make adaptively \( q \) \( q \) times private key queries and \( q \) \( q \) times decryption queries:

- **Key Generation Query**: Assume that \( ID_k = (v_1, v_2, \ldots, v_k) \) is the identity that A submits to ask for the private key. B answers the query in the following way:
  (a) B picks randomly \( r_2 \in \mathbb{Z}_p \). Define \( H(\alpha) = (f(\alpha) - f(r_0))/(\alpha - r_0) \) that is a \( q - 1 \) degree polynomial.
  (b) Set a private key \( d_{ID} = (d_0, d_1, d_2, d_3) \) of \( ID_A \) to be:

\[
\begin{align*}
   d_0 &= g^{r \cdot \alpha}((\prod_{i=1}^{l} \Pi_{j=1}^{n} g^{a_j b_j^{x_i}})^2) \\
   &\quad = g^{\frac{r \cdot \alpha}{(\prod_{i=1}^{l} \Pi_{j=1}^{n} h_j)^2}} \\
   &\quad = (g^{r \cdot \alpha})^{\frac{1}{(\prod_{i=1}^{l} \Pi_{j=1}^{n} h_j)^2}} \\
   &\quad = (h_i g_i^{-f(\alpha)})^{x_i} (\prod_{i=1}^{l} h_i)^2 \\
   d_1 &= g^2, d_2 = f(r_0), d_3 = (g_2 g_3 \cdots g_q)^{y_2}.
\end{align*}
\]

It is obvious that \( d_{ID} = (d_0, d_1, d_2, d_3) \) is a valid private key for the identity \( ID_k \).

- **Decryption Query**: Assume that A submits a ciphertext \( C \) on \( ID_k \) for decryption.

B first executes the **Key Generation Query** on identity \( ID_k \) and generates the private key \( d_{ID} \) of \( ID_k \). Next, B decrypts \( C \) using \( d_{ID} \) and returns the plaintext \( M \) of \( C \) to A.

(3) Challenge Phase: After the Phase 1, A outputs two equal length messages \( M_0, M_1 \in G \) and an identity \( ID_A = (v_1, v_2, \ldots, v_n) \) which it wants to challenge. B picks randomly \( b \in \{0, 1\} \). Let \( G(x) = x^{p^2}, G_i(x) = (G(x) - G(r_0))(x - r_0) \) which is a polynomial of degree \( q + 1 \). Let \( H_3(\alpha) = (f(\alpha) - f(r_0))/(\alpha - r_0) \). Next, B generates a ciphertext \( C' = (C_1, C_2, C_3, C_4) \) of \( M_b \) on the identity \( ID_A \) as follows:

\[
\begin{align*}
   C'_1 &= g^{r \cdot \alpha} - G(r_0) \in G, \\
   C'_2 &= Z \cdot e(g, \Pi_{i=0}^{l} g^{a_i b_i}), \\
   C'_3 &= M_f(e(C_1, g^{r \cdot \alpha}) \cdot C_2^{f(r_0)}), \\
   C'_4 &= \prod_{i=1}^{l} (C_{i,0}^{x_i} g^{a_i b_i} g^{r \cdot \alpha} b_i^{a_1 b_1} - C_{i,1}^{x_i} b_i^{a_1 b_1} C_{i,0}^{x_i} b_i^{a_1 b_1} - C_{i,1}^{x_i} b_i^{a_1 b_1} C_{i,0}^{x_i} b_i^{a_1 b_1}).
\end{align*}
\]
where $G_{ij}$ is the coefficient of $\alpha'$ in $G_i(x)$. Send $C' = (C'_1, C'_2, C'_3, C'_4)$ to $A$ as the challenge ciphertext. There is a natural restriction on the adversary that it had never queried the private key on $ID^*_jk$ or any of its proper prefixes in Phase 1.

If $Z = e(g_{q^t}, g')$, $C'$ is a valid challenge ciphertext of $M_b$ on $ID^*_k$. Because: Let $s = G_1(x) \log g_g'$, then

$$
C'_1 = g^{e(g_{q^t}g_{g')})} = g^{e(g_{q^t}, g')},
$$

$$
C'_2 = Z \cdot e(g', \prod_{n=0}^t g^{a_{n+1}^t}) = e(g_{q^t}, g') \cdot e(g', \prod_{n=0}^t g^{a_{n+1}^t}) = e(g_{q^t}, g^{G_1(x)}) = e(g, g')^s,
$$

$$
C'_3 = M_s \cdot (e(1, g) \cdot e(g, g'))^s = M_s \cdot e(1, g) \cdot e(g, g')^s,
$$

$$
C'_4 = \prod_{i=1}^t (C'_{i-1})^{\sum_{j=i}^{q^t} a_j^i} = \prod_{i=1}^t (g^{a_{i-1}^{q^t}} \cdot g^{a_{i-1}^{q^t}})^{\sum_{j=i}^{q^t} a_j^i} = \prod_{i=1}^t (g^{a_{i-1}^{q^t}})^{\sum_{j=i}^{q^t} a_j^i} = \prod_{i=1}^t \prod_{n=0}^t (a_n^i h_n^i)^{\sum_{j=i}^{q^t} a_j^i} = \prod_{n=0}^t (a_n^i h_n^i)^{\sum_{j=i}^{q^t} a_j^i}.
$$

Else, if $Z$ is a random element of $G_T$, the ciphertext will give no information about $M_b$ to $A$.

(4) Query Phase 2: A continues to issue queries as in Phase 1 and $B$ responds as before. $A$ cannot query the private key on $ID^*_k$ or any of its proper prefixes and cannot query the $(C', C'_k)$ to the Decryption Query.

(5) Guess: Finally, $A$ outputs a guess $b' \in \{0, 1\}$. If $b = b'$, $B$ outputs 1 as the solution to the truncated decision $q$-ABDHE problem, namely $Z = e(g_{q^t}, g')$, else $B$ outputs 0, namely $Z$ is a random element of $G_T$.

Probability Analysis: Since $f(x)$ is a uniformly random polynomial of degree $q$, from $A$’s view, the value of $f(\alpha)$ is uniformly random and independent of any $\alpha \in Z_p$, and the keys issued by $B$ are appropriately distributed. Therefore, if $Z = e(g_{q^t}, g')$, $B$’s simulation is perfect. Otherwise, if $Z$ is a random element in $G_T$, $A$ can’t get any information about $M_b$. So, $B$’s success probability to solve the given truncated decision $q$-ABDHE problem is the same as $A$’s, namely $\epsilon' = \epsilon$.

Time Complexity: The time complexity of algorithm $B$ is dominated by exponentiation computation in the key generation queries and decryption queries in $G$. Each such query requires $O(l + q)$ exponentiation in $G$. Since $A$ makes at most $(q_E + q_d)(< 2q)$ queries, so $t < t' + O((l + q)t)\tau$, where $\tau$ is the maximum time of an exponentiation in $G$.

3.3 Efficiency Analysis

In Table 1, we compare our scheme with other available HIBE schemes. From the Table 1, we can see that the private key size or the ciphertext size of other schemes [6-15]
depends on the hierarchy or the maximum hierarchy, and only our scheme and Zhang et al.’s scheme [16] have both the constant size, i.e., $O(1)$ size.

Compared further the public key size of our scheme with that of Zhang et al.’s scheme, their scheme needs $O(l)$ that is less than our scheme $O(nl)$ at first glance. But, in Zhang et al.’s scheme, for every submitted identity $ID_i (1 \leq i \leq l)$ for a private key, PKG must compute and issue corresponding parameter $h_{ia} (1 \leq k \leq l)$ for $ID_k$. The number of published $h_{ia}$ will increase with submitted identity $ID_i (1 \leq i \leq l)$. For example, the number of submitted identity $ID_i (1 \leq i \leq l)$ for a private key, PKG must compute and issue corresponding parameter $h_{ia} (1 \leq k \leq i)$ for $ID_k$. The number of published $h_{ia}$ will increase with the maximum depth $nl$ of our scheme. So, their scheme needs more public keys than our scheme in essence. What’s more, Zhang et al.’ scheme is only IND-ID-CPA secure in the selective-id model and the security reduction is not tight, but our scheme is IND-ID-CCA fully secure in the standard model with tight reduction.

Symbol description: $pk$ denotes the private key and PK denotes the public key. $k$ denotes the hierarchy depth, and $l$ denotes the maximum depth, and $n$ denotes the bit number of every identity, and $s$ denotes the number of block to store the $n$-bits identity where each block size is $n/s$.

### Table 1. Comparison of efficiency.

<table>
<thead>
<tr>
<th>scheme</th>
<th>security model</th>
<th>ciphertext size</th>
<th>private key size</th>
<th>public key size</th>
</tr>
</thead>
<tbody>
<tr>
<td>[6]</td>
<td>selective-id</td>
<td>$O(k)$</td>
<td>$O(k)$</td>
<td>$O(l)$</td>
</tr>
<tr>
<td>[7]</td>
<td>full</td>
<td>$O(k)$</td>
<td>$O(k)$</td>
<td>$O(nl + l)$</td>
</tr>
<tr>
<td>[8]</td>
<td>selective-id</td>
<td>$O(1)$</td>
<td>$O(l - k)$</td>
<td>$O(l)$</td>
</tr>
<tr>
<td>[9]</td>
<td>full</td>
<td>$O(k)$</td>
<td>$O(k)$</td>
<td>$O(l + s)$</td>
</tr>
<tr>
<td>[10]</td>
<td>full</td>
<td>$O(1)$</td>
<td>$O(l - k)$</td>
<td>$O(l + s)$</td>
</tr>
<tr>
<td>[12]</td>
<td>full</td>
<td>$O(k)$</td>
<td>$O(k)$</td>
<td>$O(l)$</td>
</tr>
<tr>
<td>[13]</td>
<td>full</td>
<td>$O(1)$</td>
<td>$O(l - k)$</td>
<td>$O(l)$</td>
</tr>
<tr>
<td>[14]</td>
<td>full</td>
<td>$O(1)$</td>
<td>$O(l - k)$</td>
<td>$O(l)$</td>
</tr>
<tr>
<td>[15]</td>
<td>full</td>
<td>$O(1)$</td>
<td>$O(l - k)$</td>
<td>$O(l)$</td>
</tr>
<tr>
<td>[16]</td>
<td>selective-id</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
<td>$O(nl)$</td>
</tr>
</tbody>
</table>

### 4. CONCLUSIONS

In this paper, we proposed a new HIBE scheme based on the truncated decision $q$-ABDH problem. Compared with the previous HIBE schemes, our scheme has an obvious advantage: it is the first fully secure HIBE scheme in the standard model that not only its private key achieves $O(1)$ size but also its ciphertext also achieves $O(1)$ size. Due to the constant private key and ciphertext, it can reduce the computation cost, communication cost of the sender, and storage cost of the user although adding a few public parameters. A nature question left open by this paper is to construct a HIBE scheme with shorter public key.
REFERENCES

Xiaoming Hu (胡小明) received the Ph.D. degree in Department of Computer Application Technology, Shanghai Jiao Tong University in 2009. Now, she is working in School of Computer and Information, Shanghai Second Polytechnic University, Shanghai, China. Her current research interests include cryptography, information security and network security.

Jian Wang (王见) received the Ph.D. degree in Department of Computer Application Technology, Shanghai Jiao Tong University in 2009. Now, he is working in School of Computer and Information, Shanghai Second Polytechnic University, Shanghai, China. His current research interests include network security, P2P network and sensor network.

Huajie Xu (许华杰) received the Ph.D. degree in the School of Computer Science and Technology, Huazhong University of Science and Technology in 2008. He is working in the School of Computer and Electronic Information, Guangxi University, Nanning, China. His current research interests include network security and wireless sensor network.

Yinchun Yang (杨寅春) received the M.S. degree in College of Information Security, Shanghai Jiao Tong University in 2004. Now, she is working in School of Computer and Information, Shanghai Second Polytechnic University, Shanghai, China. Her current research interests include information security and wireless network security.