Generic Constructions of RIBE via Subset Difference Method

Xuecheng Ma\(^1,2\) and Dongdai Lin\(^1,2\)

\(^1\) State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
\(^2\) School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China
\{"maxuecheng,ddlin\}@iie.ac.cn

Abstract. Revocable identity-based encryption (RIBE) is an extension of IBE which can support a key revocation mechanism, and it is important when deploying an IBE system in practice. Boneh and Franklin (Crypto’01) presented the first generic construction of RIBE, however, their scheme is not scalable where the size of key updates is linear in the number of users in the system. The first generic construction of RIBE is presented by Ma and Lin with complete subtree (CS) method by combining IBE and hierarchical IBE (HIBE) schemes. Recently, Lee proposed a new generic construction using the subset difference (SD) method by combining IBE, identity-based revocation (IBR), and two-level HIBE schemes. In this paper, we present a new primitive called Identity-Based Encryption with Ciphertext Delegation (CIBE) and propose a generic construction of RIBE scheme via subset difference method using CIBE and two-level HIBE as building blocks. CIBE is a special type of Wildcarded IBE (WIBE) and Identity-Based Broadcast Encryption (IBBE). Furthermore, we show that CIBE can be constructed from IBE in a black-box way. The ciphertext in our generic RIBE scheme can be reduced to constant by using IBBE. Instantiating the underlying IBBE scheme with a proper concrete scheme, we can obtain a RIBE scheme with constant-size public parameter, ciphertext, private key and \(O(r)\) key updates in the selective-ID model. Additionally, our generic RIBE scheme can be easily converted to a sever-aided RIBE scheme which is more suitable for lightweight devices.

Key words: Generic Construction, Revocable IBE, Subset Difference, DKER

1 Introduction

Identity-Based Encryption (IBE) was introduced by Shamir [47], to eliminate the need for maintaining a certificate based Public Key Infrastructure (PKI) in the traditional Public Key Encryption (PKE) setting. The first IBE scheme was proposed by Boneh and Franklin [9] in the random oracle model [4]. Since then, there are many follow-up works [6, 7, 50, 21, 51, 15, 10, 2, 3, 11–13, 22, 52, 53, 19]. A hierarchical IBE (HIBE) scheme [23, 25] generalizes the concept of IBE by forming levels of a hierarchy. For an \(\ell\)-level HIBE, a hierarchical identity is a vector of maximal \(\ell\) identities, and a user at level \(i\) can generate a secret key for its descendants at level \(j\) (where \(i < j \leq \ell\)).

To address the challenge of key revocation in IBE setting, Boneh and Franklin [9] presented the notion of revocable IBE and proposed a naive method to add a simple revocation mechanism to any IBE system as follows. A sender encrypts a message using a receiver’s identity concatenated with the current time period, i.e., \(id||T\) and the Key Generation Center (KGC) issues the private key \(sk_{id||T}\) for each non-revoked user in every time period. However, BF-RIBE scheme is inefficient. The number of private keys issued in every time period is linear in the number of all users in the system hence the scheme does not scale well when there are a large number of users.

Boldyreva, Goyal and Kumar [5] proposed the first scalable revocable IBE (RIBE) scheme by combining the fuzzy IBE scheme of Sahai and Waters [43] with the complete subtree (CS) method [36]. In the definition of security in BGK-RIBE, the adversary is only given access to the secret key oracle, the revocation oracle and the key update oracle. Seo and Emura [44, 46] introduced a stronger security notion called decryption key exposure resistance (DKER) which captures the realistic attack where an additional decryption key oracle is given. In the definition of DKER security experiment, an exposure of a user’s decryption key at some time period will not compromise the confidentiality of ciphertexts which are encrypted for different time periods.
It attracted many follow-up works concerning R(H)IBE schemes with DKER [20, 27, 29, 31, 32, 35, 39, 40, 42, 46, 49].

Server-aided RIBE [41, 16, 37] is a variant of RIBE where almost all of the workload on the user side can be delegated to an untrusted third-party server. The server is untrusted in the sense that it does not possess any secret information. Each user only needs to store a short long-term private key without having to communicate with KGC.

Ma and Lin [34] proposed a generic construction of RIBE using complete subtree method by combining IBE and HIBE in a black-box way which solved the open problem presented in [44]. In their first scheme, an update key consists of $O(r \cdot \ell)$ IBE private keys and a ciphertext consists of $O(\ell)$ IBE ciphertexts where $r$ is the number of revoked users and $\ell$ is the bit length of an identity. And they also made some optimization using HIBE or IBBE [17] which makes the ciphertext size constant. Currently, Lee [30] proposed a generic RIBE scheme with the subset difference method by using IBE, identity-based revocation (IBR), and two-level HIBE schemes as basic building blocks. Their scheme reduced the size of an update key from $O(r \cdot \ell)$ key elements to $O(r)$ key elements but the ciphertext size increased to $O(\ell^2)$ number of IBE and IBR ciphertexts. In addition, they showed how to reduce the ciphertext size by extending their generic RIBE scheme to use the more efficient LSD method instead of using the SD method.

1.1 Our Contributions.

In order to construct a generic RIBE scheme using SD method, we first present a new primitive called identity based encryption with ciphertext delegation (CIBE). Contrary to HIBE where an identity secret key can decrypt ciphertexts encrypted under its descendants, an identity secret key can decrypt ciphertexts encrypted under its ancestors in a CIBE scheme. In addition, the plaintext encrypted under an identity id is confidential if the adversary does not know the secret key of id or descendants of id. It is obvious that the new primitive CIBE is a special type of wildcarded IBE (WIBE) [1] where the wildcard "*" just appears at the end portion of the pattern. It can also be viewed as a special type of IBBE where we encrypt a plaintext under all descendants of id. Moreover, we will show that CIBE can be constructed from IBE in a black-box way.

In this paper, we propose a generic construction of RIBE with SD method by combining CIBE and a two-level HIBE. Our technique and building blocks are different from Lee’s generic RIBE scheme although both of our two schemes use SD method. In our generic construction, the key update size is $O(r)$ CIBE keys and the ciphertext is $O(\ell^2)$ CIBE ciphertexts and one HIBE ciphertext. Furthermore, CIBE can be constructed from IBE in a black-box way so we can give a generic construction of RIBE with SD method by using IBE and HIBE as building blocks. However the secret key size of the generic CIBE scheme from IBE is $O(\ell)$ which results in a $O(r \cdot \ell)$ size of key update in the generic RIBE scheme. Although the key update is not short as that of Lee’s construction and the ciphertext size is bigger than that of generic construction using CS method [34], it shows the possibility that generically construct RIBE with SD method using IBE and HIBE and we wish to give a CIBE scheme with shorter secret key from IBE in the future. Furthermore, we can construct CIBE from WIBE (IBBE), our generic RIBE scheme consists of $O(r)$ WIBE (IBBE) secret keys in key update and $O(\ell^2)$ WIBE (IBBE) ciphertexts and one HIBE ciphertext in a ciphertext. In addition, the layered SD (LSD) method can be applied to our generic RIBE scheme which reduces the ciphertext and the update key to $O(\ell^{1.5})$ CIBE ciphertexts and $4r$ CIBE private keys, respectively. Last but not least, we can reduce the ciphertext size by using IBBE solves the open problem presented by Lee [30]. Note that it is difficult to reduce the ciphertext size in Lee’s scheme since it uses an IBR scheme. Instantiating the underlying IBBE and HIBE schemes with proper concrete schemes, we can obtain a RIBE scheme with constant-size public parameter, ciphertext, private key and $O(r)$ key updates in the selective-ID model.

1.2 Our Technique

Let us first describe the generic RIBE scheme using CS method proposed by Ma and Lin. Ma and Lin observed that the design principle of CS method in the symmetric setting [36]. In symmetric

\[3\] In fact, WIBE is a special type of IBBE.
Broadcast encryption [36], every user corresponds to a leaf node in a complete binary tree and holds all secret keys corresponding to the nodes in the path from root to the associated leaf. The non-revoked users are covered by complete subtrees and the plaintext is encrypted by secret keys corresponding the root nodes of the subtrees which cover all the non-revoked users. A user is not revoked if and only if there is a ciphertext encrypted under a secret key corresponding a node in the path from the root to its associated leaf node. In RIBE setting, Ma and Lin view an identity id as a leaf in a complete binary tree with depth |id|. A plaintext is encrypted under the identifiers of nodes in the path from the root to |id| using IBE and KGC broadcast a set of IBE secret keys associated with the root node of the complete subtrees which cover all non-revoked users. The key update consists of secret key of one ancestor of |id| iff |id| is not revoked. We only describe the behind idea for realizing revocation mechanism in the scheme of Ma and Lin. For the security reason, they divided the plaintexts into two secret shares, one is encrypted using HIBE and the other is encrypted using IBE under all ancestors of |id|.

Unlike CS method that covers non-revoked users by complete subtrees, SD method covers non-revoked users using subsets $CV_r = \{S_{i,j}\}$ where $S_{i,j}$ is presented by two nodes $(v_i, v_j)$ and $v_i$ is an ancestor of $v_j$ and $S_{i,j}$ contains all leaves which are descendants of $v_i$ but not of $v_j$. The Assign algorithm assigns secret keys corresponding nodes which are adjacent to the nodes in the path from root to $v_{id}$ but not ancestors of $v_{id}$. Let $PV_{id} = \{S'_{i,j}\}$ denote the node subset assigned to $id$ where $S'_{i,j}$ is presented by $(v_i', v_j')$. The assign algorithm and cover algorithm guarantees that $id$ is not revoked iff there exists $S_{i,j} \in CV_r$ and $S'_{i,j} \in PV_{id}$ such that $v_i = v_i'$ and $v_j = v_j'$ is an ancestor of $v_j$. In order to apply SD method to RIBE, we divide the plaintext into to shares. One is encrypted under $id$ using HIBE and the other is encrypted under $T$ concatenating corresponding nodes in $PV_{id}$ using CIBE. KGC broadcasts CIBE secret keys $\{sk_{S_{i,j}}\}_{S_{i,j} \in CV_n}$ in time $T$. The ciphertext delegation property of CIBE guarantees that the CIBE ciphertext part can be decrypted using key update iff there is an identity in the ciphertext which is a prefix of $S_{i,j}$.

### 1.3 Related Works

**Revocable IBE.** The first revocable IBE scheme from any IBE was presented by Boneh and Franklin [9], however their proposal was not scalable and need a secret channel. Boldyreva et al. [5] proposed the first scalable RIBE combining fuzzy IBE and CS method. A number of secure and efficient RIBE schemes using a broadcast method for key updates have been proposed [14, 28, 33, 40, 44, 48]. Most of the RIBE schemes follow the CS method for update keys, but Lee et al. [31] showed that an RIBE scheme with the SD method can be designed to reduces the size of update keys. Recently, Ma and Lin [34] proposed a generic RIBE construction with the CS method by combining IBE and HIBE schemes. Subsequently, Lee proposed a generic RIBE scheme with SD method using IBE, IBR and HIBE as building blocks.

**Revocable HIBE.** Seo and Emura [34] presented the first revocable HIBE (RHIBE) scheme with history-preserving updates, wherein a low-level user must know the history of key updates performed by ancestors in the current time period which makes the scheme very complex. Subsequently, Seo and Emura [45] presented a new method to construct RHIBE that implements history-free updates. After that, there are some follow-up works concerning about efficiency [32], stronger security [29] or assumptions without pairing [28].

### 2 Preliminaries

#### 2.1 Notations

Throughout the paper we use the following notation: We use $\lambda$ as the security parameter and write $\operatorname{negl}(\lambda)$ to denote that some function $f(\cdot)$ is negligible in $\lambda$. An algorithm is PPT if it is modeled as a probabilistic Turing machine whose running time is bounded by some function $\operatorname{poly}(\lambda)$. If $S$ is a finite set, then $s \leftarrow S$ denotes the operation of picking an element $s$ from $S$ uniformly at random. If $A$ is a probabilistic algorithm, then $y \leftarrow A(x)$ denotes the action of running $A(x)$ on input $x$ with uniform coins and outputting $y$. Let $[n]$ denotes $\{1, \ldots, n\}$. Let $\{0,1\}^{[i,j]}$ denotes all binary strings with length in $[i, j]$. For a bit string $a = (a_1, \ldots, a_n) \in \{0,1\}^n$, and $i, j \in [n]$ with $i \leq j$, we write $a_{[i,j]}$ to denote the substring $(a_i, \ldots, a_j)$ of $a$. For any two strings $u$ and $v$, $|u|$ denotes
the length of $u$ and $u||v$ denotes their concatenation. Let $BT$ be a complete binary tree. For two strings $s$ and $t$ of length $\ell$, we use $s =_\ell t$ to denote $s$ matches $t$ and $s \neq_\ell t$ to denote $s$ does not match $t$. We define $s =_\ell t$ iff $s_i = t_i \lor t_i = *$ for all $i \in \{1, ..., \ell\}$ and $s \neq_\ell t$ iff $s_i \neq t_i \land t_i \neq *$ for some $i \in \{1, ..., \ell\}$.

2.2 Identity-Based Encryption with Ciphertext Delegation

An IBE with ciphertext delegation scheme CIBE consists of four algorithms Setup, KeyGen, Enc, and Dec, which are defined as follows:

1. **Setup**$(1^\lambda)$: The setup algorithm takes as input a security parameter $1^\lambda$ and outputs a master key $MK$ and public parameter $PP$.
   - **KeyGen**(MK, id): This algorithm takes as input the master secret key $MK$ and an identity $id \in \{0, 1\}^{\ell'}$, it outputs the identity secret key $sk_{id}$.
   - **Enc**(PP, id, $\mu$): This algorithm takes as input the public parameter $PP$, an identity $id \in \{0, 1\}^{\ell'}$ where $\ell' \leq \ell$, and a plaintext $\mu$, it outputs a ciphertext $c$.
   - **Dec**(sk$_{id}$, $c$): This algorithm takes as input a secret key $sk_{id}$ for identity $id$ and a ciphertext $c$, it outputs a plaintext $\mu$.

**Correctness**: The correctness of CIBE is defined as follows: For all security parameters $1^\lambda$, two identities $id \in \{0, 1\}^{\ell'}$, $id' \in \{0, 1\}^{\ell'}$ and plaintext $\mu$, the following holds:

$$\Pr[\text{Dec}(sk_{id'}, \text{Enc}(PP, id, \mu)) = \mu] = 1$$

where $id$ is a prefix of $id'$, $(PP, MK) \leftarrow \text{Setup}(1^\lambda)$ and $sk_{id'} \leftarrow \text{KeyGen}(MK, id')$.

**Multi-Identity Adaptive Security**: For any PPT adversary $A$, there is a negligible function $\negl(\cdot)$ such that the following holds:

$$\Pr_{\lambda}[\text{IND-mCID-CPA}(A) = 1] - \frac{1}{2} \leq \negl(\lambda)$$

where IND-mCID-CPA$(A)$ is shown in Figure 1.

If $q = 1$, we call the above experiment as single-identity adaptive security (IND-CID-CPA). Ad-

<table>
<thead>
<tr>
<th>Experiment IND-mCID-CPA$(A)$ :</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. $(PP, MK) \leftarrow \text{Setup}(1^\lambda)$</td>
</tr>
<tr>
<td>2. $(\mu_0, \mu_1, id'_1, ..., id'_q) \leftarrow A^{\text{KeyGen}(MK, \cdot)}(PP)$ where $q$ is a polynomial of $\lambda$, $</td>
</tr>
<tr>
<td>3. $\beta \leftarrow {0, 1}$</td>
</tr>
<tr>
<td>4. ${c'<em>i \leftarrow \text{Enc}(PP, id'<em>i, \mu</em>\beta)}</em>{i \in [q]}$</td>
</tr>
<tr>
<td>5. $\beta' \leftarrow A^{\text{KeyGen}(MK, \cdot)}(PP, c'_1, ..., c'_q)$ and for each query $id$ to KeyGen($MK, \cdot$) we have that any $id' \in {id'_1, ..., id'_q}$ is not a prefix of $id$ (or equal to $id$).</td>
</tr>
<tr>
<td>6. Output 1 if $\beta = \beta'$ and 0 otherwise.</td>
</tr>
</tbody>
</table>

**Fig. 1.** The multi-identity adaptive security experiment of CIBE

Additionally, we can define the selective security analogously where the adversary first commit the challenge identities before obtaining the public parameter. Obliviously, single-identity security is a special case of multi-identity security. For the other direction, we will show that single-identity security implies multi-identity security.

**Lemma 1.** An CIBE scheme is multi-identity adaptively (selectively) secure if it is single-identity adaptively (selectively) secure.
Proof. Since the proof for the adaptive-ID security and that for selective-ID security are essentially the same, we only show the proof for the former.

We prove the lemma by hybrid arguments. First, we define $q + 1$ hybrid games $\mathcal{H}_0, \ldots, \mathcal{H}_q$ where $\mathcal{H}_0$ is the real IND-mID-CPA game and for all $i \in [q]$, $\mathcal{H}_i$ is the same as $\mathcal{H}_{i-1}$ except the way that the challenger generates the challenge ciphertext. In $\mathcal{H}_i$, the challenger computes the challenge ciphertext as $\{c^*_j \leftarrow \text{Enc}(\text{PP}, \text{id}^*_i, 0)\}_{j \in \{1, \ldots, i\}}$ and $\{c^*_j \leftarrow \text{Enc}(\text{PP}, \text{id}^*_i, \mu_j)\}_{j \in \{i+1, \ldots, q\}}$ where $0$ is an all-zeros string with the same length of $\mu_0$ and $\beta$ is randomly chosen from $\{0, 1\}$. Let $S_i$ denote the event that the output of IND-mCID-CPA game is $1$ in $\mathcal{H}_i$. In $\mathcal{H}_q$, the challenge ciphertext is encryption of zeros so $\Pr[S_q] = \frac{1}{2}$. We will show that $|\Pr[S_{i-1}] - \Pr[S_i]| \leq \text{negl}(\lambda)$ for all $i \in [q]$ and finish the proof. We construct a PPT algorithm $\mathcal{B}$ such that $|\Pr[S_{i-1}] - \Pr[S_i]|$ is equal to the probability that $\mathcal{B}$ breaks single-identity adaptive-ID security of CIBE. The detail of the algorithm $\mathcal{B}$ is as follows:

1. $\mathcal{B}$’s challenger sends the public parameter $\text{PP}$ to $\mathcal{B}$ and $\mathcal{B}$ forwards it to $\mathcal{A}$.
2. When $\mathcal{A}$ queries secret key for identity $\text{id}$, $\mathcal{B}$ makes secret key query for $\text{id}$ and sends $\text{sk}_\text{id}$ to $\mathcal{A}$. Then $\mathcal{A}$ sends $q$ challenge identities $\text{id}_1^*, \ldots, \text{id}_q^*$ and two plaintexts $(\mu_0, \mu_1)$ with the same length.
3. $\mathcal{B}$ randomly chooses a bit $\beta$ and sends $(0, \mu_\beta, \text{id}^*_1)$ to its challenger, where $[0] = [\mu_0] = [\mu_1]$.
   The challenger randomly chooses a bit $b$ and outputs $c^*_0 = \text{Enc}(\text{PP}, \text{id}^*_1, 0)$ if $b = 0$ and $c^*_b = \text{Enc}(\text{PP}, \text{id}^*_1, \mu_\beta)$ if $b = 1$. Then, $\mathcal{B}$ computes $\{c^*_j \leftarrow \text{Enc}(\text{PP}, \text{id}^*_j, 0)\}_{j \in \{1, \ldots, i-1\}}$ and $\{c^*_j \leftarrow \text{Enc}(\text{PP}, \text{id}^*_j, \mu_j)\}_{j \in \{i+1, \ldots, q\}}$. Finally, it outputs $c^* = (c^*_1, \ldots, c^*_q)$.
4. $\mathcal{B}$ answers the secret key queries as Step 2. $\mathcal{A}$ outputs a guess $\beta'$ of $\beta$. $\mathcal{B}$ outputs $b' = 0$ if $\beta' = \beta$ and outputs $b' = 1$ otherwise.

Note that the identity $\text{id}_{\mathcal{A}}$ submits to secret key oracle with the restriction that no one identity in $\{\text{id}^*_1, \ldots, \text{id}^*_q\}$ is a prefix of $\text{id}$. For all $i \in \{0, 1, \ldots, q\}$, $\mathcal{B}$ does not query secret key for $\text{id}$ where there exists a challenge identity that is a prefix of $\text{id}$ in $\mathcal{H}_i$. If $b = 0$, $\mathcal{B}$ perfectly simulates the challenger in $\mathcal{H}_i$, and otherwise, it perfectly simulates that in $\mathcal{H}_{i-1}$. Moreover, the probability that $b' = b$ satisfies:

$$
\Pr[b' = b] = \Pr[b' = b | b = 0] \Pr[b = 0] + \Pr[b' = b | b = 1] \Pr[b = 1] = \frac{1}{2} \Pr[b' = b | b = 0] + \frac{1}{2} \Pr[b' = b | b = 1] = \frac{1}{2} \Pr[b = 0] + \frac{1}{2} (1 - \Pr[b' \neq b | b = 1]) \\
= \frac{1}{2} + \frac{1}{2} (\Pr[b' = \beta | b = 0] - \Pr[b' = \beta | b = 1]) \\
= \frac{1}{2} + \frac{1}{2} (\Pr[S_i] - \Pr[S_{i-1}])
$$

The single-identity adaptive security of CIBE guarantees that $|\Pr[b' = b] - \frac{1}{2}| \leq \text{negl}(\lambda)$ so $|\Pr[S_i] - \Pr[S_{i-1}]| \leq \text{negl}(\lambda)$ for all $i \in [q]$. Hence, $|\Pr[S_0] - \Pr[S_q]| = |\Pr[S_0] - \frac{1}{2}| \leq \text{negl}(\lambda)$. We complete the proof.

We can construct CIBE from IBE in a black-box way. The $\text{Setup}$, $\text{Enc}$ and $\text{Dec}$ algorithms are the same of those of underlying IBE scheme. To generate a secret key for an identity $\text{id}$ in CIBE scheme, we generate secret keys for all prefixes of $\text{id}$ using $\text{KeyGen}$ algorithm of IBE.

2.3 Wildcarded Identity-Based Encryption

A wildcarded identity-based encryption scheme consists of four probabilistic polynomial-time (PPT) algorithms ($\text{Setup}$, $\text{KeyGen}$, $\text{Enc}$, $\text{Dec}$) defined as follows:

- $\text{Setup}(1^\lambda)$: This algorithm takes as input the security parameter $1^\lambda$, and outputs a public parameter $\text{PP}$ and a master secret key $\text{MK}$.
- $\text{KeyGen}(\text{MK}, \text{id})$: This algorithm takes as input the master secret key $\text{MK}$ and an identity $\text{id} \in \{0, 1\}^\ell$, it outputs the identity secret key $\text{sk}_{\text{id}}$. 

5


- **Enc**(PP, P, µ): This algorithm takes as input the public parameter PP, a pattern P ∈ {0, 1,*}^ℓ, and a plaintext µ, it outputs a ciphertext c.

- **Dec**(sk_id, c): This algorithm takes as input a secret key sk_id for identity id and a ciphertext c, it outputs a plaintext µ.

The following correctness and security properties must be satisfied:

- **Correctness**: For all security parameters 1^λ, any identity id ∈ {0, 1}^f, any pattern P ∈ {0, 1,*}^f and plaintext µ ∈ M, the following holds:

\[
\Pr[\text{Dec}(\text{sk}_\text{id}, \text{Enc}(PP, P, \mu)) = \mu] = 1
\]

where id = *P, (PP, MK) ← Setup(1^λ) and sk_id ← KeyGen(MK, id).

- **Adaptive Security**: For any PPT adversary A, there is a negligible function negl(·) such that the following holds:

\[
\text{Adv}^{\text{IND-WID-CPA}}_A = |\Pr[\text{IND-WID-CPA}(A) = 1] - \frac{1}{2}| \leq \text{negl}(\lambda)
\]

where IND-WID-CPA(A) is shown in Figure 2.

---

**Experiment IND-WID-CPA(A):**

1. (PP, MK) ← Setup(1^λ)
2. (P_0, P_1, *P) ← A_{KeyGen(MK,·)}(PP) where |P_0| = |P_1| and for each query id to KeyGen(MK,·) we have that id ≠ *P.
3. β ← {0, 1}
4. c* ← Enc(PP, *P, µ_β)
5. β' ← A_{KeyGen(MK,·)}(PP, c*) and for each query id to KeyGen(MK,·) we have that id ≠ *P.
6. Output 1 if β = β' and 0 otherwise.

---

**Fig. 2.** The adaptive security experiment of WIBE

It is obvious that CIBE is a special type of WIBE when encrypt under an identity id ∈ {0, 1}^f we encrypt under a pattern P = id || * || *...|| * where |P| = ℓ and ℓ ≤ f. In addition, CIBE is also a special type of IBBE, when encrypt under an identity id we encrypt under all descendants of id (including id) using IBBE.

### 2.4 Hierarchical Identity-Based Encryption

An HIBE scheme consists of four algorithms **Setup**, **KeyDer**, **Enc**, and **Dec**, which are defined as follows:

- **Setup**(1^λ, ℓ): The setup algorithm takes as input a security parameter 1^λ and maximum hierarchical depth ℓ. It outputs a master key MK and public parameter PP.
- **KeyDer**(PP, sk_id|k−1, id|k): This algorithm takes as input a secret key sk_id|k−1 of hierarchical identity id|k−1 = (I_1,...,I_(k−1)) ∈ ℱ^k−1, a hierarchical identity id|k = (I_1,...,I_k) ∈ ℱ^k and the public parameter PP. Note that sk_id|0 = MK. It outputs a secret key sk_id|k for id|k.
- **Enc**(id|k, µ, PP). The encryption algorithm takes as input a hierarchical identity id|k = (I_1,...,I_k) ∈ ℱ^k, a message µ, and public parameters PP. It outputs a ciphertext c_{id|k}.
- **Dec**(c_{id|k}, sk_id|k, PP): The decryption algorithm takes as input a ciphertext c_{id|k}, a private key sk_id|k, and public parameters PP. It outputs a message µ or ⊥.

- **Correctness**: For all (PP, MK) ← Setup(1^λ), all id|k₀, id'|k₁, Dec(Enc(id'|k₁, µ, PP), sk_id'|k₂, PP) = µ, where id|k₀ is a prefix of id'|k₁, sk_id|k₀ ← KeyDer(PP, MK, id|k₀) and sk_id'|k₁ ← KeyDer(PP, sk_id|k₀, id'|k₁).

- **Adaptive Security**: For any PPT adversary A, there is a negligible function negl(·) such that the following holds:

---

6
\[ Adv^{\text{IND-HID-CPA}}_{\mathcal{A}} = \Pr_{\mathcal{A}} [\text{IND-HID-CPA}(\mathcal{A}) = 1] - \frac{1}{2} \leq \text{negl}(\lambda) \]

where IND-HID-CPA(\mathcal{A}) is shown in Figure 3.

Experiment IND-HID-CPA(\mathcal{A}):

1. \((\mathcal{P}, \mathcal{K}) \leftarrow \text{Setup}(1^\lambda)\)
2. \((\mu_0, \mu_1, \text{id}^*) \leftarrow \mathcal{A}_{\text{KeyDer}(\mathcal{M}^*):}(\mathcal{P})\) where \(|\mu_0| = |\mu_1|\) and for each query id to KeyDer(\mathcal{M}, \cdot) we have that id is not a prefix of id*.
3. \( \beta \leftarrow \{0,1\} \)
4. \(c^* \leftarrow \text{Enc}(\mathcal{P}, \text{id}^*, \mu_\beta)\)
5. \(\beta' \leftarrow \mathcal{A}_{\text{KeyDer}(\mathcal{M}^*):}(\mathcal{P}, c^*)\) and for each query id to KeyDer(\mathcal{M}, \cdot) we have that id is not a prefix of id*.
6. Output 1 if \(\beta = \beta'\) and 0 otherwise.

Fig. 3. The adaptive security experiment of HIBE

2.5 Subset Difference Method

The subset difference (SD) method is a special instance of the subset cover framework introduced by Naor, Naor, and Lotspeich [36] which becomes a general methodology for scalable revocation. There is a complete binary tree \(BT\) with \(2^\ell\) leaves. In our generic RIBE scheme, we view user identity as a leaf in \(BT\). We define \(T_i\) as a complete binary subtree where its root is node \(v_i\). For two nodes in the tree \((v_i; v_j)\) such that \(v_i\) is an ancestor of \(v_j\), a valid subtree \(T_{i,j}\) is defined as \(T_i - T_j\). A valid subset \(S_{i,j}\) is represented by \((v_i; v_j)\) which is defined as the set of leaf nodes that belong to \(T_{i,j}\), i.e. a leaf \(u \in S_{i,j}\) if \(v_i\) is an ancestor of \(u\) but \(v_j\) is not. For a full binary tree \(BT\) and a subset \(R\) of leaf nodes, \(ST(BT, R)\) is defined as the Steiner Tree induced by the set \(R\) and the root node, that is, the minimal subtree of \(BT\) that connects all the leaf nodes in \(R\) and the root node. Specifically, the subset difference method is defined as follows:

- **SD.Setup(\(N_{\text{max}}\)):** This algorithm takes as input the maximum number \(N_{\text{max}}\) of users. Let \(N_{\text{max}} = 2^\ell\) for simplicity. Let \(BT\) denote a complete binary tree of depth \(\ell\). The corresponding leaf node in the tree of an identity \(\text{id} \in \{0,1\}^{\ell}\) is the terminal node walking from the root directed by \(\text{id}\). For an identity \(\text{id} = \text{id}_0||\text{id}_1||...||\text{id}_{\ell-1}\), if \(\text{id}_i\) is 0, go left, otherwise go right at depth \(i\). Note that the root is at depth 0. We set the identifier of the root as 2, so the identifier of corresponding node of id of 2||\text{id}. The collection \(S\) of SD is the set of all subsets \(\{S_{i,j}\}\) where \(v_i; v_j \in BT\) and \(v_i\) is an ancestor of \(v_j\).

- **SD.Assign(\(BT, \text{id}\)):** This algorithm takes as input the tree \(BT\) and an identity id \(\in \{0,1\}^{\ell}\). Let \(v_id\) be the corresponding leaf node in \(BT\) of id. Let \((v_{k_0}, v_{k_1}, ..., v_{k_\ell})\) be the path from the root node \(v_{k_0}\) to the leaf node \(v_{k_\ell} = v_id\) and \((v_{k_0}', ..., v_{k_\ell}')\) be the nodes just “hanging off” the path, i.e. they are adjacent to the path but not ancestors of \(v_id\). It first sets a private set \(PV_{id}\) as an empty set. For all \(i \in \{k_0, k_1, ..., k_\ell\}\) and \(j \in \{k_0', ..., k_\ell'\}\) where \(v_i\) is an ancestor of \(v_j\), it adds the subset \(S_{i,j}\) presented by two nodes \((v_i; v_j)\) into \(PV_{id}\). It outputs the private set \(PV_{id}\).

- **SD.Cover(\(BT, R\)):** This algorithm takes as input the tree \(BT\) and a revoked set \(R\) of users. It first sets a subtree \(T\) as \(ST(BT, R)\), and then it builds a covering set \(CV_R\) iteratively by removing nodes from \(T\) until \(T\) consists of just a single node as follows:

  (a) It finds two leaf nodes \(v_i\) and \(v_j\) in \(T\) where the least-common-ancestor \(v\) of \(v_i\) and \(v_j\) does not contain any other leaf nodes of \(T\) in its subtree. Let \(v_i\) and \(v_k\) be the two child nodes of \(v\) where \(v_i\) is an ancestor of \(v_k\) and \(v_k\) is an ancestor of \(v_j\). If there is only one leaf node left, it makes \(v_i = v_j = v\) as the root of \(T\) and \(v_i = v_k = v\).

  (b) If \(v_i \neq v_j\), then it adds the subset \(S_{i,j}\) to \(CV_R\); Similarly, if \(v_j \neq v_k\), it adds the subset \(S_{k,j}\) to \(CV_R\).
We give an example of SD.Assign algorithm and SD.Cover algorithm in Figure 4 and Figure 5 respectively.

\[ R = \{v_9, v_{10}\}, \quad CV = \{(v_9, v_4)\} \]

In order to present our generic RIBE scheme, we first define two functions which encode \( CV_R \) and \( PV_{id} \) to identities respectively. For nodes \( v_i \) and \( v_j \) in \( T_{v_i} \), we define \( H_K : (v_i, v_j) \rightarrow \{0, 1\}^{v_j} \) which maps \( S_{i,j} \) to the identifier of \( v_j \) in \( T_{v_i} \) where \( T_{v_i} \) denotes the complete subtree rooted at \( v_i \). Let \( H_E : \{0, 1\}^l \rightarrow \{0, 1, 2\}^{l+1} \) be a function mapping an identity \( id \in \{0, 1\}^l \) to a set of encodings of \( S_{i,j} \in PV_{id} \). Specifically, \( H_E(x) \) is defined as follows. Obtain \( PV_{id} \) by computing SD.Assign\((BT, id)\). Output \( \{H_K(S_{i,j})\}_{S_{i,j} \in PV_{id}} \). From the definition of SD.Assign, we know that there exists an identifier \( id' \) in \( H_E(id) \) which is a prefix of \( H_K(S_{i,j}) \) iff \( v_i \in S_{i,j} \). In Figure 5, \( CV = (v_9, v_4) \), \( H_K(v_9, v_4) = 201 \). In figure 4, \( PV_{v_{12}} \) is
\{(v_0,v_1),(v_0,v_6),(v_0,v_{11}),(v_2,v_6),(v_2,v_{11}),(v_5,v_{11})\} and \(H_E(PV_{v_2}) = \{20, 211, 2100, 221, 2200, 2120\}\). There exists “20” which is a prefix of “201” since \(v_{12}\) is not revoked. Moreover, \(PV_{v_3}\) is \(\{\{v_0,v_2\},(v_0,v_3),(v_0,v_{10}),(v_1,v_3),(v_1,v_{10}),(v_4,v_{10}\}\}\) and \(H_E(PV_{v_3}) = \{21, 200, 211, 220, 2211, 2021\}\). There exists no element in \(H_E(v_3)\) which is a prefix of “201” since \(v_3\) is revoked.

3 A Generic Construction of Revocable Identity-Based Encryption

3.1 Definition and Security Model

Similar to the definition in [44], a revocable IBE scheme has seven probabilistic polynomial-time (PPT) algorithms (Setup, KeyGen, KeyUp, DkGen, Enc, Dec, Revoke) with associated message space \(M\), identity space \(ID\), and time space \(\hat{T}\).

- **Setup(\(1^\lambda, N_{max}\))**: This algorithm takes as input a security parameter \(\lambda\) and a maximal number of users \(N_{max}\). It outputs a public parameter PP, a master secret key MK, a revocation list RL (initially empty), and a state ST.
- **KeyGen(MK, id, ST)**: This algorithm takes as input the master secret key MK, an identity id, and the state ST. It outputs a secret key sk_\(id\) and an update state ST_\(id\).
- **KeyUp(MK, T, RL, ST)**: This algorithm takes as input the master secret key MK, a time period \(T \in \hat{T}\), the revocation list RL, and the state ST. It outputs a key update KU_\(T\).
- **DkGen(sk_\(id\), KU_\(T\))**: This algorithm takes as input a secret key sk_\(id\) and the key update KU_\(T\). It outputs a decryption key dk_\(id,T\) or a special symbol \(\bot\) indicating that id was revoked.
- **Enc(PP, id, T, \mu)**: This algorithm takes as input the public parameter PP, an identity id, a time period \(T\) and a message \(\mu \in M\). It outputs a ciphertext c. It outputs a message \(\mu \in M\).
- **Dec(dk_\(id,T\), c)**: This algorithm takes as input a decryption secret key dk_\(id,T\) and a ciphertext. It outputs a message \(\mu \in M\).
- **Revoke(id, T, RL)**: This algorithm takes as input an identity id, a revocation time \(T \in \hat{T}\) and the revocation list RL. It outputs a revocation list RL.

It satisfies the following conditions:

- **Correctness**: For all \(\lambda\) and polynomials (in \(\lambda\)) N, all PP and MK output by setup algorithm Setup, all \(\mu \in M\), id \(\in ID\), \(T \in \hat{T}\) and all possible valid states ST and revocation list RL, if identity id was not revoked before or, at time \(T\) then there exists a negligible function negl(\(\cdot\)) such that the following holds:

\[
\Pr[\text{Dec}(dk_{id,T}, \text{Enc}(PP, id, T, \mu)) = \mu] \geq 1 - \text{negl}(\lambda)
\]

where \(sk_{id} \leftarrow \text{KeyGen}(MK, id, ST)\), KU_\(T\) \(\leftarrow \text{KeyUp}(MK, T, RL, ST)\) and \(dk_{id,T} \leftarrow \text{DkGen}(sk_{id}, KU_T)\).

- **Adaptive Security**: For any PPT adversary \(A\), there is a negligible function negl(\(\cdot\)) such that the advantage of \(A\) satisfies:

\[
\text{Adv}_A^{\text{IND-RID-CPA}} = |\Pr[\text{IND-RID-CPA}(A) = 1]| - \frac{1}{2^n} \leq \text{negl}(\lambda)
\]

where IND-RID-CPA(\(A\)) is shown in Figure 6. Note that the experiment defined in Figure 6 captures decryption key exposure attack.

3.2 Construction

Let (CIBE.Setup, CIBE.Enc, CIBE.KeyGen, CIBE.Dec) be a CIBE scheme with \(ID = \{0, 1, 2\}^{\ell+1,2^{\ell+1}}\) and (HIBE.Setup, HIBE.Enc, HIBE.KeyDer, HIBE.Dec) be a two-level HIBE scheme where the element identity is in \(\{0,1\}^{\ell}\). We assume the HIBE scheme and the CIBE scheme have the same plaintext space \(M\) which is finite and forms a group with the group operation “ + ”.

Utilizing the above primitives, we will show how to construct a generic RIBE scheme II = (Setup, KeyGen, KeyUp, DkGen, Encrypt, Decrypt, Revoke) as follows. In our RIBE scheme, the plaintext space is \(M\) and identity space is \(\{0,1\}^{\ell}\). Moreover, we assume the time period space \(\hat{T}\) is a subset of the identity space, i.e. \(\hat{T} \subseteq \{0,1\}^{\ell}\). More specifically, our RIBE scheme is shown as follows:
mutually exclusive and cover all possibilities:

we can construct a PPT algorithm

We will prove the adaptive-ID security and the proof for selective-ID security is exactly

ID (selective-ID) secure.

3.3 Security Analysis

A 2. Type-2 adversary: A

\begin{align*}
\text{Experiment } \text{IND-RID-CPA}(A) : \\
1. (PP, MK) \leftarrow \text{Setup}(1^\lambda, N) \\
2. (\mu_0, \mu_1, id^*, T^*,:) \leftarrow A^{\text{KeyGen}()} \text{(MK, : KeyUp}() \text{., RL, ST}) \text{.DkGen(.,: Revoke(.,}(PP) \text{ where } |\mu_0| = |\mu_1| \\
3. \beta \leftarrow \{0, 1\} \\
4. c^* \leftarrow \text{Enc}(PP, id^*, T^*, \mu_1) \\
5. \beta^* \leftarrow A^{\text{KeyGen}()} \text{(MK, : KeyUp}() \text{., RL, ST}) \text{.DkGen(.,: Revoke(.,}(PP, c^*). \\
6. Output 1 if \beta = \beta^* \text{ and 0 otherwise.}
\end{align*}

The following restriction must hold:

- KeyUp(MK,., RL, ST) and Revoke(.,) can be queried on time which is greater than or equal to the
time of all previous queries, i.e. the adversary is allowed to query only in non-decreasing order of
time. Also, the oracle Revoke(.,) cannot be queried at time T if KeyUp(MK,., RL, ST) was queried
on time T.

- If KeyGen(MK,.) was queried on identity id^*, then Revoke{id^*,T} must be queried for some T \leq T^*,
i.e. (id^*, T) must be on revocation list RL when KeyUp(MK,., RL, ST) is queried on T^*.

- DkGen(id^*, T^*) cannot be queried.

- DKG(.,) cannot be queried on time T before KeyUp(,) was queried on T.

\[ \text{Fig. 6. The adaptive security experiment of revocable IBE} \]

- Setup(1^\lambda, N_{max}) : Run HIBE.Setup(1^\lambda, 2) \rightarrow (HPP, HMK) \text{ and CIBE.Setup}(1^\lambda) \rightarrow (CPP, CMK).

- SD.Setup(1^\lambda, N_{max}). Output MK = HMK, an empty revocation list RL, a secret state ST =
CMK and public parameter PP = (HPP, CPP).

- KeyGen(P, MK, id) : Parse PP as (HPP, CPP), and output hsk_{id} \leftarrow HIBE.KeyDer(HPP, HMK, id).

- KeyUp(P, ST, RL, T) : If there exists (id', T') \in RL for some T' \leq T, add the identifier of
id' to BL to R. Then, obtain CV_{R,T} = \{S_{i,j}\} by running SD.Cover(BL, R). For each S_{i,j} \in
CV_{R,T}, compute csk_{S_{i,j}} \leftarrow CIBE.KeyGen(CPP, CMK, T, H_{K(S_{i,j}))}. Output updated key KU_{T} =
\{S_{i,j}, csk_{S_{i,j}}\}_{S_{i,j} \in CV_{R,T}}.

- Enc(P, id, T, \mu) : Parse PP as HPP and CPP. Randomly choose \mu_0, \mu_1 \text{ with the condition that } u = \mu_0 + \mu_1. \text{ Compute } c_0 \leftarrow HIBE.Enc(HPP, id || T, \mu_0) . \text{ For each id' \in H_{E}(id), compute }
\text{csk}_{id'} \leftarrow \text{CIBE.Enc}(CPP, T || id', \mu_1). \text{ Output } c = \{c_0, T, \{id', c_{id'}\}_{id' \in H_{E}(id)}\}.

- DkGen(s_{id}, KU_{T}) : Parse KU_{T} as \{S_{i,j}, csk_{S_{i,j}}\}_{S_{i,j} \in CV_{R,T}}. Output PV_{id} by computing SD.Assign
(BL, id). If SD.Match(CV_{R,T}, PV_{id}) outputs (S_{i,j}, S_{i',j}'), fetch csk_{S_{i,j}}; otherwise, output \perp and
abrupt. Compute hsk_{id,T} \leftarrow HIBE.KeyDer(HPP, hsk_{id,T}). Output dk_{id,T} = (hsk_{id,T}, T, S_{i,j}, csk_{S_{i,j}}).

- Dec(CPP, c, sk_{id,T}) : Parse sk_{id,T} as (hsk_{id,T}, T, S_{i,j}, csk_{S_{i,j}}). Parse c as c_0, T, \{id', c_{id'}\}_{id' \in H_{E}(id)}.
if T \neq T', abort; Otherwise, find the identifier id' which is a prefix of H_{K}(S_{i,j}), compute
\mu_1 \leftarrow \text{CIBE.Dec}(CPP, csk_{S_{i,j}}, c_{id'}) and \mu_0 \leftarrow HIBE.Dec(HPP, hsk_{id,T}, c_0). Output \mu = \mu_0 + \mu_1.

- Revoke(ST, RL, T, id) : It adds (id, T) to RL and outputs the updated revocation list RL.

3.3 Security Analysis

**Theorem 1.** The revocable IBE is adaptive-ID (selective-ID) secure with decryption key exposure resilience if the underlying CIBE scheme and the underlying two-level HIBE scheme are adaptive-ID (selective-ID) secure.

**Proof.** We will prove the adaptive-ID security and the proof for selective-ID security is exactly
the same. For any PPT adversary against the adaptive-ID security with DKER of revocable IBE,
we can construct a PPT algorithm B against the adaptive-ID security of the underlying CIBE or
HIBE scheme. B randomly guesses an adversarial type among the following two types which are
mutually exclusive and cover all possibilities:

1. Type-1 adversary: A issues a secret key query for id^* hence id^* has to be revoked before T^*.
2. Type-2 adversary: A does not issue a secret key query for id^*.
Note that B’s guess is independent of the attack that A chooses, so the probability that B guesses right is $\frac{1}{2}$. We separately describe B’s strategy by its guess.

**Type-1 adversary:** We will show that if adversary $A_1$ makes a type-1 attack successfully, there exists an adversary $B_1$ breaking the multi-identity adaptive security of CIBE defined in Figure 1.

$B_1$ proceeds as follows:

- **Setup:** $B_1$ obtains a public parameter CPP from its challenger. It generates $(HPP, HMK) \leftarrow HIBE.Setup(1^\lambda, 2)$ and sends $(HPP, CPP) \rightarrow A_1$. $B_1$ keeps HMK as the master secret key and initial revocation list $RL$ and an identifier set $R$ as empty set.

- **KeyGen:** When receiving a secret key query for id, if there exists a record of $(id, hsk_d)$ return $hsk_d$. Otherwise, $B_1$ generates the secret key normally by running $hsk_d \leftarrow HIBE.KeyDer(HMK, id)$ and record $(id, hsk_d)$.

- **Revoke:** $B_1$ receives $(id, T)$ from $A_1$, and adds $(id, T)$ to $RL$.

- **KeyUp:** Upon receiving $T$, for all $(id', T') \in RL$ where $T' \leq T$, add the identifier of $id'$ in $BT$ to $R$. Then, obtain $CV_{R,T} = \{S_{i,j}\}$ by running SD.Cover($BT, R$). For each $S_{i,j} \in CV_{R,T}$, compute $H_K(S_{i,j})$. Query secret keys for $\{T||H_K(S_{i,j})\}_{i,j \in CV_{R,T}}$. Output $K_{U_T} = \{(S_{i,j}, csk_{S_{i,j}})\}_{i,j \in CV_{R,T}}$.

- **DkGen:** When receiving $(id, T)$, if there exists a record $(id, hsk_d)$ fetch $hsk_d$. Otherwise, $B_1$ normally runs the HIBE.KeyDer algorithm with HMK and record $(id, hsk_d)$. Note that KeyUp($T$) has been queried before. $B_1$ outputs $DkGen(hsk_d, K_{U_T})$.

- **Challenge:** $A_1$ outputs an identity $id^*$, a time period $T^*$ and two plaintexts $\mu_0, \mu_1$, with the same length. $B_1$ randomly samples $\mu \in \mathcal{M}$ and sends $\{T||id^*\}_{\omega \in H_E(\omega^*)}$ as challenger identities and $\mu_0' = \mu_0 - \mu$ and $\mu_1' = \mu_1 - \mu$ as the challenge plaintexts. The challenger randomly chooses a challenge bit $\beta$ and sends the challenge ciphertexts $\{c_{id'} \leftarrow CIBE.Enc(CPP, T||id^*, \mu_\beta)\}_{\omega \in H_E(\omega^*)}$ to $B_1$. $B_1$ computes $c_0' = HIBE.Enc(HPP, id^*||T^*, \mu)$ and sends $c' = (c_0', T^*, \{id', c_{id'} \}_{\omega \in H_E(\omega^*)})$ to $A_1$.

- **Guess:** $A_1$ outputs a guess bit $\beta'$ and $B_1$ set $\beta'$ as its guess.

Due to $id^*$ has been revoked at or before $T^*$, $\nu_{id^*}$ is not covered by $CV_{R,T}$, i.e. SD.Match($CV_{R,T}, PV_{id^*}$) outputs ⊥. The property of the encoding function $H_K$ and $H_E$ guarantees that no one identifier $id'$ in $H_E(\omega^*)$ is a prefix of $H_K(S_{i,j})$ where $S_{i,j} \in CV_{R,T}$. So $B_1$ does not ask any secret key queries for id where some challenge identity is a prefix of $id$. $B_1$ perfectly simulates $A_1$’s view so that $B_1$’s challenge bit is also $A_1$’s challenge bit. $B_1$ just forwards $A_1$’s guess so the probability that $B_1$ wins in multi-identity adaptive security game of CIBE scheme is equal to the probability that $A_1$ wins in adaptive-ID security with decryption key exposure game of RIBE scheme.

**Type-2 adversary:** If there exists an adversary $A_2$ who makes a type-2 attack successfully, we can construct an adversary $B_2$ breaking adaptive-ID security of the underlying HIBE scheme.

$B_2$ proceeds as follows:

- **Setup:** $B_2$ obtains a public parameter HPP from its challenger. It generates $(CPP, CMK) \leftarrow CIBE.Setup(1^\lambda)$ and sends $(HPP, CPP) \rightarrow A_2$. $B_2$ keeps CMK as the state.

- **KeyGen:** When receiving a secret key query for id, $B_2$ just forwards the secret key query to its challenger and sends the challenger’s response to $A_2$.

- **Revoke:** $B_2$ receives $(id, T)$ from $A_2$, and adds $(id, T)$ to $RL$.

- **KeyUp:** When $A_2$ makes a key update query for time $T$, $B_2$ generates the updated key normally by using CMK.

- **DkGen:** Upon receiving $(id, T)$, B queries secret key oracle for id||T and obtains $hsk_{id||T}$. Note that KeyUp($T$) has been queried. Then runs the DkGen algorithm normally.

- **Challenge:** $A_2$ outputs a challenge identity $id^*$, a time period $T^*$ and two plaintexts $\mu_0$ and $\mu_1$ with the same length. $B_1$ randomly samples $\mu \in \mathcal{M}$ and sends $id^*||T^*, \mu_0' = \mu_0 - \mu$ and $\mu_1' = \mu_1 - \mu$ to its challenger. $B_1$ receives the challenge ciphertext $c_0' = HIBE.Enc(HPP, id^*||T^*, \mu_0')$ where $\beta$ is $B_2$’s challenge bit chosen randomly by its challenger. For each $id' \in H_E(id^*)$, compute $c_{id'} \leftarrow CIBE.Enc(CPP, T||id', \mu_\beta)$ and sends $c = (c_0', T^*, \{id', c_{id'} \}_{\omega \in H_E(\omega^*)})$ to $A_2$.

- **Guess:** $A_2$ outputs a guess bit $\beta'$ and $B_2$ sets $\beta'$ as its guess.

For the KeyUp oracle, $B_2$ can respond by itself because it has the state. For the KeyGen oracle, $A_2$ never requests secret key for the challenge identity $id^*$; And DkGen($id^*, T^*$) is never queried, so $B_2$ never requests secret keys for $id^*||T^*$ or its ancestors. $B_2$ perfectly simulates $A_2$’s view so
that $B_2$'s challenge bit is also $A_2$'s challenge bit. $B_2$ just forwards $A_2$'s guess so the probability that $B_2$ wins in adaptive-ID security game of HIBE scheme is equal to the probability that $A_2$ wins in adaptive-ID security with decryption key exposure game of RIBE scheme.

When we put the results for two types of adversary together, we can conclude that the revocable IBE is adaptive-ID secure if both the underlying IBE and HIBE schemes are adaptive-ID secure.

3.4 Extensions

Layered Subset Difference. In our generic RIBE construction, the ciphertext and update key are $O(\ell^2)$ CIBE ciphertexts plus a HIBE ciphertext and $2r$ CIBE private keys respectively where $\ell$ is the bit length of identity and $r$ is the number of revoked users. We can use layered subset difference (LSD) method [24] to reduce the ciphertext size. If we replace the SD algorithms by LSD algorithms in our generic RIBE construction, the ciphertext and the update key are $O(\ell^{1.5})$ CIBE ciphertexts plus a HIBE ciphertext and $4r$ CIBE private keys respectively.

Constant Ciphertext. Due to CIBE is a special type of IBBE, we can directly replace CIBE by IBBE. The ciphertext in this generic construction consists of $O(\ell^2)/O(\ell^{1.5})$ IBBE ciphertexts plus a HIBE ciphertext if we use SD/LSD method. In addition, we can replace all $O(\ell^2)/O(\ell^{1.5})$ IBBE ciphertexts $e_i$ encrypted under set $S_i$ by one IBBE ciphertext $c$ encrypted under a set $S = \cup S_i$ since all $e_i$ encrypt the same plaintext. So we can reduce the ciphertext to be one IBBE ciphertexts and one HIBE ciphertext.

Server-Aided RIBE. In server-aided model, there is a semi-honest server without any secret key information that takes almost all the workload on users. The server is curious but honestly performs the procedure. More specifically, the server partially decrypts the ciphertexts using the key update and leaves less decryption task to users. It is easy to convert our scheme to be server-aided, given the key update $KU_T = \{ S_{i,j}, csk_{S_{i,j}} | S_{i,j} \in CV_{a,T} \}$ and a ciphertext $c = \{ c_0, T, \{ id', c_{id'} | id' \in H_E(id) \} \}$, the server first computes $PV_{id} \leftarrow SD.Assign(BT, id)$. If $SD.Match(CV_{R,T}, PV_{id})$ outputs $(S_{i,j}, S'_{i,j}, c_{id'})$, fetch $csk_{S_{i,j}}$ from $KU_T$ and compute $\mu_1 \leftarrow CIBE.Dec(CPP, csk_{S_{i,j}}, c_{id'})$ where $id'$ is a prefix of $H_E(S_{i,j})$. Finally, the server sends $(c_0, T, \mu_1)$ as the transformed ciphertext to the receiver. The receiver only needs to operate the key extraction and decryption algorithm of underlying HIBE scheme. The receiver does not need to communicate with KGC in every key update.

3.5 Instantiation

If we instantiate the underlying two-level HIBE scheme with BBG-HIBE [8] and the underlying IBBE scheme with a IBBE scheme with constant size public parameter, ciphertext and private key presented in [26], we can obtain a selectively secure RIBE with constant public parameter, ciphertext, private key and $O(\tau)$ key update. To the best of our knowledge, it is the first RIBE scheme that simultaneously realizes constant size of public parameter, ciphertext, private key and $O(\tau)$ number of key update. Additionally, if we instantiate the underlying two-level HIBE scheme with BBG-HIBE [8] and the underlying WIBE scheme with BBG-WIBE scheme [1], we can obtain an adaptively secure RIBE in the random oracle where the sizes of ciphertexts and key update are $O(\ell^{2.5})$ and $O(\tau)$ respectively (using LSD method). The ciphertext-policy attribute-based encryption presented in [38] implies WIBE and combining the result in [18] that HIBE can be constructed from IBE, we can obtain a RIBE scheme based on RSA.

4 Conclusion

In this paper, we presented a new primitive called IBE with ciphertext delegation (CIBE) where an identity secret key can decrypt ciphertexts encrypted under its ancestors. CIBE is a special type of WIBE and IBBE and can be constructed from IBE in a black-box way. We then proposed a generic RIBE scheme via subset difference method using CIBE and two-level HIBE as building blocks. In our generic RIBE scheme, the key update consists of $O(\tau)$ CIBE private keys and ciphertext consists of $O(\ell^2)$ CIBE ciphertexts and one HIBE ciphertext. The ciphertext size can be reduced to $O(\ell^{1.5})$ by using layered subset difference method. Moreover, the generic RIBE scheme can
be converted to a server-aided RIBE scheme and be instantiated efficiently. We can reduce the ciphertext size using IBBE and the instantiated RIBE scheme has constant-size public parameter, ciphertext, private key and $O(r)$ key update. We can obtain RIBE based on RSA assumption if we instantiate the underlying buildings based on RSA.

References


