A Generic Construction for Revocable Identity-Based Encryption with Subset Difference Methods

Kwangsu Lee*

Abstract

To deal with dynamically changing user's credentials in identity-based encryption (IBE), providing an efficient key revocation method is a very important issue. Recently, Ma and Lin proposed a generic method of designing a revocable IBE (RIBE) scheme that uses the complete subtree (CS) method by combining IBE and hierarchical IBE (HIBE) schemes. In this paper, we propose a new generic method for designing an RIBE scheme that uses the subset difference (SD) method instead of using the CS method. In order to use the SD method, we generically design an RIBE scheme by combining two-level HIBE and single revocation encryption (SRE) schemes. If the underlying HIBE and SRE schemes are adaptively (or selectively) secure, then our RIBE scheme is also adaptively (or selectively) secure. In addition, we show that the layered SD (LSD) method can be applied to our RIBE scheme and a chosen-ciphertext secure RIBE scheme also can be designed generically.

Keywords: Revocable identity-based encryption, Subset difference method, Generic construction.

*Sejong University, Seoul, Korea. Email: `kwangsu@sejong.ac.kr`.

1 Introduction

Identity-based encryption (IBE) is a new type of public-key encryption (PKE) that solve the public-key management problem in PKE by using a user's identity as a public key [41]. Since the first IBE scheme in bilinear maps was proposed by Boneh and Franklin [5], research on new types of cryptographic encryption such as IBE, hierarchical IBE (HIBE), attribute-based encryption (ABE), and predicate encryption (PE) has been actively studied as an important research topic [5,8,15,17]. Despite the long history of research on IBE, the IBE schemes have not been widely deployed in real environments. One reason of this problem is that unlike PKE schemes, which uses a public-key infrastructure to handle certificate issuance and revocation, it is not simple to revoke the private key of a user in IBE. Therefore, an important additional feature of IBE schemes is to support the private key revocation flexibly and efficiently.

The method of revoking the private key of a user in IBE has been studied since the initial IBE scheme was designed, but this method is not suitable for handling a large number of users [5]. The first revocable IBE (RIBE) scheme to efficiently handle large numbers of users was proposed by Boldyreva et al. [3]. The key design principle of their RIBE scheme is that a trusted center periodically creates and broadcasts an update key on time T for non-revoked users, along with the generation of a user's private key. In this case, if the private key of a user ID is not revoked in the update key on time T, the user can decrypt a ciphertext for his identity ID and the corresponding time T. In other words, the RIBE scheme proposed by Boldyreva et al. can be seen as a method to support the indirect private key revocation, in which the center decides the revocation of private keys instead of the sender. Specifically, Boldyreva et al. designed their RIBE scheme by combining a tree-based broadcast method with a fuzzy identity-based encryption scheme. After the work of Boldyreva et al., various RIBE schemes and extension schemes have been proposed to enhance the efficiency, security, and functionality of RIBE [9, 10, 18, 25, 28, 31, 36, 38–40, 43].

Currently, to design an RIBE scheme, we redesign an RIBE scheme from the beginning by directly modifying an efficient IBE scheme proposed before. This is problematic in that a new RIBE scheme must be designed again whenever a new IBE scheme having a different mathematical structure is proposed. Ma and Lin recently overcome this problem by suggesting a generic method of designing an RIBE scheme by using an IBE scheme as a black-box [32]. In their generic RIBE scheme with the complete subtree (CS) method, an update key consists of $O(r \log \frac{N}{r})$ IBE private keys and a ciphertext consists of $O(\log N)$ IBE ciphertexts where r is the number of revoked users and N is the number of users. In the RIBE scheme, reducing the size of update keys is an important issues since an update key should be broadcasted to all users for each time period. The motivation of this work is to reduce the update key size of the generic RIBE scheme. In tree-based broadcast encryption, there exists the subset difference (SD) method proposed by Naor et al. [34] which is more efficient than the CS method. Additionally, the layered SD (LSD) method which improved the SD method has also been proposed [16]. Therefore, we ask whether it is possible to design an RIBE scheme from an IBE scheme in a generic way using the SD/LSD method to reduce the size of update keys. If the SD/LSD method can be applied to a generic RIBE scheme, the size of an update key can be reduced from $O(r \log \frac{N}{r})$ key elements to $O(r)$ key elements.

1.1 Our Contributions

In this paper, we show that it is possible to design an RIBE scheme with the SD method in a generic way. As described above, the generic RIBE scheme with the CS method uses two-level HIBE and IBE schemes as basic building blocks [32]. On the contrary, our generic RIBE scheme with the SD method uses two-level HIBE and single revocation encryption (SRE) schemes as basic building blocks. The SRE scheme is a special type of broadcast encryption scheme in which a ciphertext is specified with a group label GL and a

Table 1: Comparison of revocable identity-based encryption schemes

Scheme	PP Size	SK Size	UK Size	CT Size	Model	DKER	Generic
BF [5]	$O(1)$	$O(1)$	$O(N-r)$	$O(1)$	SE	No	No
BGK [3]	$O(1)$	$O(\log N)$	$O(r\log\frac{N}{r})$	$O(1)$	SE	No	No
LV [31]	$O(\lambda)$	$O(\log N)$	$O(r\log\frac{N}{r})$	$O(1)$	AD	No	No
SE [39]	$O(\lambda)$	$O(\log N)$	$O(r\log\frac{N}{r})$	$O(1)$	AD	Yes	No
LLP [25]	$O(1)$	$O(\log^2 N)$	$O(r)$	$O(1)$	AD	Yes	No
ML1 [32]	$O(1)$	$O(1)$	$O(r\log\frac{N}{r})$	$O(\log N)$	AD	Yes	Yes
ML2 [32]	$O(\log N)$	$O(1)$	$O(r\log\frac{N}{r})$	$O(1)$	AD	Yes	Yes
Ours (SD)	$O(1)$	$O(1)$	$O(r)$	$O(\log^2 N)$	AD	Yes	Yes
Ours (LSD)	$O(1)$	$O(1)$	$O(r)$	$O(\log^{1.5} N)$	AD	Yes	Yes

Let λ be a security parameter, N be the number of maximum users, and r be the number of revoked users. We count the number of group elements to measure the size of parameters. We use symbols SE for selective IND-CPA and AD for adaptive IND-CPA.

revoked member label ML. The newly derived RIBE scheme with the SD method consists of $O(r)$ number of SRE private keys in an update key and $O(\log^2 N)$ number of SRE ciphertexts in a ciphertext. Compared with the previous generic RIBE scheme with the CS method, the size of an update key is reduced but the size of a ciphertext is increased. The detailed comparison of RIBE schemes is given in Table 1.

To analyze the security of our generic RIBE scheme with the SD method, we show that if the underlying HIBE and SRE schemes are adaptively (or selectively) secure under chosen plaintext attacks, then the proposed generic RIBE scheme is also adaptively (or selectively) secure under chosen plaintext attacks. The key idea of our proof is to first divide the types of an attacker according to the queries of the attacker, and to isolate the attacker of a specific type to break the security of the underlying HIBE or SRE scheme. However, this idea is not simple to apply since the SD method has a complicated subset cover structure unlike the CS method. To handle this complicated structure in a ciphertext, we introduce additional hybrid games in the security proof and handle each ciphertext element of the challenge ciphertext one by one.

In addition, we show that it is possible to reduce the size of a ciphertext by extending our generic RIBE scheme to use the more efficient LSD method instead of using the SD method, but this modified scheme increases the size of an update key slightly. We also show that our generic RIBE scheme which provides only chosen-plaintext attack (CPA) security can be extended to provide the security against the more powerful chosen-ciphertext attacker (CCA). To provide the CCA security of RIBE, the underlying HIBE and SRE schemes should provide the CCA security and a one-time signature scheme with strong unforgeability should be used.

1.2 Related Work

Certificate Revocation. The study of certificate revocation in public-key encryption has been the subject of much research. In reality, the most widely used certificate revocation method is to periodically issue a certificate revocation list (CRL) containing serial numbers of revoked user's certificates. In addition, a delta-CRL can be used to more efficiently issue the revocation information, and it is also possible to immediately check the state of a certificate by using the online certificate status protocol (OCSP) service. In the theoretical aspect, various certificate revocation methods which are more efficient than the traditional methods also have been proposed [2, 33, 35].

Broadcast Encryption. Public-key broadcast encryption (PKBE) provides the revocation of receivers because a sender can specify a receiver set S in a ciphertext directly [7]. Identity-based broadcast encryption (IBBE) can provide more powerful revocation than existing PKBE because the maximum number of users in the system can be exponential [11]. Identity-based revocation (IBR) can be viewed as a cryptographic scheme that implements direct user revocation because all system users except the revoked users can decrypt a ciphertext where a revoked set R is specified in the ciphertext [27, 29]. However, PKBE, IBBE, and IBR have the disadvantage that a user cannot be revoked after the creation of a ciphertext. Particularly, it is a critical problem in a cryptographic system in which ciphertexts are stored in cloud storage and a user accesses these ciphertexts later since the user cannot be revoked when his or her credential is expired.

Revocable IBE. Boneh and Franklin [5] proposed a revocation method for IBE such that a trusted center periodically issues a private key for a user by combining an identity and time as $ID\|T$, but this method is not scalable since a secure channel should be established for every time. The efficient and scalable RIBE scheme was proposed by Boldyreva et al. [3] by combining the complete subtree (CS) method and a fuzzy identity-based encryption scheme. In their RIBE scheme, a ciphertext is associated with a receiver's identity D and time T, and a trusted center periodically issues an update key one time T for non-revoked users to implement the indirect key revocation. A number of secure and efficient RIBE schemes using a broadcast method for key updates have been proposed [9, 18, 31, 36, 39, 43, 44]. Most of the RIBE schemes follow the CS method for update keys, but Lee et al. [25] showed that an RIBE scheme with the SD method can be designed to reduces the size of update keys. Recently, Ma and Lin [32] proposed a generic RIBE construction with the CS method by combining IBE and HIBE schemes.

Revocable HIBE. The first revocable HIBE (RHIBE) scheme, which provides the private key revocation in HIBE, was proposed by Seo and Emura [38]. They proposed an RHIBE scheme by applying the design principle of previous RIBE schemes to an HIBE scheme. To improve the initially proposed RHIBE scheme, Seo and Emura later introduced a history-free update method to reduce the size of private keys and update keys [40]. After that, Lee and Park have introduced a new RHIBE scheme with short private keys and short updated keys by introducing an intermediate private key in HIBE and using a modular design method [28]. In order to enhance the selective security of previous RHIBE schemes, Lee proposed an adaptively secure RHIBE scheme by applying the dual system encryption method [20].

Revocable ABE. ABE is an extension of IBE in which a ciphertext is associated with attributes and a private key is associated with an access structure, and the ciphertext of ABE can be decrypted by the private key of ABE if the attributes satisfies the access structure [15]. An revocable ABE (RABE) scheme was proposed by Boldyreva et al. [3] by following the design principle of their RIBE scheme. ABE is well-suited for environments such as cloud storage where multiple users access different ciphertexts since it can provide flexible access control. For such an environment, Sahai et al. [37] proposed a revocable-storage ABE (RS-ABE) scheme that supports ciphertext updates as well as user key revocation. Lee et al. [21, 23] proposed an improved RS-ABE scheme by using a self-updatable encryption scheme, and they also proposed an RS-ABE scheme that provides the CCA security [26]. A generic construction of ABE with direct revocation in which a revoked set is attached in a ciphertext was proposed by Yamada et al. [45].

1.3 Versions of This Paper and Corrections

Regrettably, the RIBE scheme of our previous paper [22] is insecure as Takayasu pointed out [42]. Our previous RIBE scheme was designed by combining HIBE, IBE, and IBR scheme. However, the previous RIBE scheme had a problem in that an attacker could decrypt the IBR ciphertext included in the challenge RIBE ciphertext by using another (non-revoked) IBR private key included in an update key. In this revised paper, we updated our RIBE scheme to use an SRE scheme instead of using IBE and IBR schemes to prevent the previous attack. By using an SRE scheme instead of using the IBE and IBR schemes, it was possible to design a more efficient RIBE scheme than before.

2 Preliminaries

In this section, we first review the definition and security model of HIBE and SRE. Next, we review the definition and security model of RIBE.

2.1 Hierarchical Identity-Based Encryption

Hierarchical identity-based encryption (HIBE) is an extension of IBE in which a hierarchical identity is used to represent a user's identity and the delegation of private keys is provided [13, 17]. In HIBE, a user receives a private key for his hierarchical identity from a trusted center, or receives a delegated private key from another user. If a sender creates a ciphertext for a receiver's hierarchical identity and transmits it to a receiver, then the receiver can decrypt the ciphertext by using his private key if the hierarchical identity of his private key is a prefix of the hierarchical identity of the ciphertext.

Let $HID = (ID_1, \ldots, ID_k)$ be an identity vector of size k. We let $HID|_j$ be a vector $(ID_1, \ldots, ID_j)$ of size j derived from HID. We define a function $Prefix(HID|_k)$ that returns a set of prefix vectors $\{HID|_j\}_{1 \le j \le k}$ where $HID|_k = (ID_1, \ldots, ID_k)$. The detailed syntax of HIBE is given as follows.

Definition 2.1 (Hierarchical Identity-Based Encryption, HIBE)**.** An HIBE scheme consists of five algorithms **Setup**, **GenKey**, **Delegate**, **Encrypt**, and **Decrypt**, which are defined as follows:

Setup($1^\lambda, L_{max}$). The setup algorithm takes as input a security parameter 1^λ and maximum hierarchical depth L_{max}. It outputs a master key MK and public parameters PP.

GenKey($HID|_k, MK, PP$). The key generation algorithm takes as input a hierarchical identity $HID|_k = (ID_1, \ldots, ID_k) \in \mathcal{I}^k$ where $k \le L_{max}$, the master key MK, and the public parameters PP. It outputs a private key $SK_{HID|_k}$.

Delegate($HID|_k, SK_{HID|_{k-1}}, PP$). The delegation algorithm takes as input a hierarchical identity $HID|_k$, a private key $SK_{HID|_{k-1}}$ for $HID|_{k-1}$, and the public parameters PP. It outputs a delegated private key $SK_{HID|_k}$.

Encrypt($HID|_\ell, M, PP$). The encryption algorithm takes as input a hierarchical identity $HID|_\ell = (ID_1, \ldots, ID_\ell) \in \mathcal{I}^\ell$ where $\ell \le L_{max}$, a message M, and public parameters PP. It outputs a ciphertext $CT_{HID|_\ell}$.

Decrypt($CT_{HID|_\ell}, SK_{HID|_k}, PP$). The decryption algorithm takes as input a ciphertext $CT_{HID|_\ell}$, a private key $SK_{HID|_k}$, and public parameters PP. It outputs a message M.

The correctness of HIBE is defined as follows: For all MK, PP generated by **Setup**($1^\lambda, L_{max}$), all $HID|_\ell, HID|_k$, any $SK_{HID|_k}$ generated by **GenKey**($HID|_k, MK, PP$) such that $HID|_k \in Prefix(HID|_\ell)$, it is required that

- **Decrypt**(**Encrypt**($HID|_\ell, M, PP$), $SK_{HID|_k}, PP$) $= M$.

The security model of HIBE is defined by extending the security model of IBE to include additional private key delegations [13, 17]. That is, an attacker can request delegated private key queries together with general private key queries. In this case, if the distribution of general private keys and the distribution of delegate private keys are the same, then we can only consider general private key queries to simplify the security model. The detailed security model of HIBE is given as follows.

Definition 2.2 (IND-CPA Security)**.** The IND-CPA security of HIBE is defined in terms of the following game between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$:

1. **Setup**: $\mathcal{C}$ generates a master key MK and public parameters PP by running **Setup**($1^\lambda, L_{max}$). It keeps MK to itself and gives PP to $\mathcal{A}$.

2. **Phase 1**: $\mathcal{A}$ may adaptively request a polynomial number of private key queries. In response, $\mathcal{C}$ gives a corresponding private key $SK_{HID|_k}$ to $\mathcal{A}$ by running **GenKey**($HID|_k, MK, PP$) for each query.

3. **Challenge**: $\mathcal{A}$ submits a challenge hierarchical identity $HID^*|_\ell$ and two messages M_0^*, M_1^* with the equal length subject to the restriction: for each $HID|_k$ of private key queries, $HID|_k \notin Prefix(HID^*|_\ell)$. $\mathcal{C}$ flips a random coin $\mu \in \{0,1\}$ and gives a challenge ciphertext CT^* to $\mathcal{A}$ by running **Encrypt**($HID^*|_\ell$, M_μ^*, PP).

4. **Phase 2**: $\mathcal{A}$ may continue to request private key queries.

5. **Guess**: $\mathcal{A}$ outputs a guess $\mu' \in \{0,1\}$ of μ, and wins the game if $\mu = \mu'$.

The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}_{\mathcal{A}}^{HIBE}(\lambda) = \left|\Pr[\mu = \mu'] - \frac{1}{2}\right|$ where the probability is taken over all the randomness of the game. An HIBE scheme is IND-CPA secure if for all PPT adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ is negligible in the security parameter λ.

2.2 Single Revocation Encryption

Single revocation encryption (SRE) is a special kind of broadcast encryption [24, 27], in which a user is specified with a group and member labels (GL, ML) and a ciphertext is generated for a group label GL and a revoked member label ML'. In SRE, a sender generates a ciphertext CT for group and revoked member labels (GL, ML) and a message M. A receiver who has a private key for his group and member labels (GL', ML') from a trusted central decrypts the ciphertext if the group labels are equal $GL = GL'$ but the member labels are not equal $ML \neq ML'$. The detailed syntax of SRE is given as follows.

Definition 2.3 (Single Revocation Encryption, SRE)**.** An SRE scheme consists of four algorithms **Setup**, **GenKey**, **Encrypt**, and **Decrypt**, which are defined as follows:

Setup(1^λ): The setup algorithm takes as input a security parameter 1^λ. It outputs a master key MK and public parameters PP.

GenKey($(GL, ML), MK, PP$): The private key generation algorithm takes as input labels (GL, ML), the master key MK, and public parameters PP. It outputs a private key $SK_{(GL,ML)}$.

Encrypt($(GL, ML), M, PP$): The encryption algorithm takes as input labels (GL, ML), a message $M \in \mathcal{M}$, and public parameters PP. It outputs a ciphertext $CT_{(GL,ML)}$.

Decrypt($CT_{(GL,ML)}, SK_{(GL',ML')}, PP$): The decryption algorithm takes as input a ciphertext $CT_{(GL,ML)}$, a private key $SK_{(GL',ML')}$, and public parameters PP. It outputs a message M.

The correctness of SRE is defined as follows: For all MK and PP generated by **Setup**(1^{λ}), SK_{ID} generated by **GenKey**($(GL',ML'), MK, PP$) for any (GL',ML'), and any (GL,ML) and any M, it is required that

- If $(GL = GL') \wedge (ML \neq ML')$, **Decrypt**(**Encrypt**($(GL,ML), M, PP$)$, SK_{(GL',ML')}, PP) = M$.

The security model of SRE is defined by extending the IND-CPA security model of PKBE [27]. In this model, an attacker requests private key queries on labels. In the challenge step, the attacker submits challenge labels (GL^*, ML^*) and the challenge messages M_0^*, M_1^* and receives a challenge ciphertext CT^*. The attacker additionally requests private key queries and finally guesses the hidden message in CT^*. The detailed description of the security model is given as follows.

Definition 2.4 (IND-CPA Security)**.** The security of SRE is defined in terms of the indistinguishability under chosen plaintext attacks (IND-CPA). The security game is defined as the following game between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$:

1. **Setup**: $\mathcal{C}$ runs **Setup**(1^{λ}) to generate a master key MK and public parameters PP. It keeps MK to itself and gives PP to $\mathcal{A}$.
2. **Query 1**: $\mathcal{A}$ adaptively requests private keys for labels $(GL_1, ML_1), \ldots, (GL_{q_1}, ML_{q_1})$. In response, $\mathcal{C}$ gives the corresponding private keys $SK_1, \ldots, SK_{q_1}$ to $\mathcal{A}$ by running **GenKey**($(GL_i, ML_i), MK, PP$).
3. **Challenge**: $\mathcal{A}$ submits challenge labels (GL^*, ML^*) and two messages M_0^*, M_1^* with the equal length subject to the restriction: for all (GL_i, ML_i) of private key queries, it is required that $(GL_i \neq GL^*)$ or $(GL_i = GL^*) \wedge (ML_i = ML^*)$. $\mathcal{C}$ flips a random coin $\mu \in \{0,1\}$ and gives the challenge ciphertext CT^* to $\mathcal{A}$ by running **Encrypt**($(GL^*, ML^*), M_{\mu}^*, PP$).
4. **Query 2**: $\mathcal{A}$ may continue to request private keys for labels $(GL_{q_1+1}, ML_{q_1+1}), \ldots, (GL_q, ML_q)$.
5. **Guess**: $\mathcal{A}$ outputs a guess $\mu' \in \{0,1\}$ of μ, and wins the game if $\mu = \mu'$.

The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}_{\mathcal{A}}^{SRE}(\lambda) = \left|\Pr[\mu = \mu'] - \frac{1}{2}\right|$ where the probability is taken over all the randomness of the game. A SRE scheme is secure under chosen plaintext attacks if for all PPT adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ in the above game is negligible in the security parameter λ.

2.3 Revocable Identity-Based Encryption

Revocable identity-based encryption (RIBE) is an extension of identity-based encryption (IBE) to support private key revocation [3]. In RIBE, each user receives a private key for his or her identity ID from a trusted center. The trusted center then periodically generates an update key which is associated with time T and a non-revoked user set, and then it broadcasts the update key through the public channel. In this case, if the private key of a user is not revoked in the update key, the user can derive a decryption key for ID and T by combining the private key and the update key, and this decryption key can be used to decrypt a ciphertext which is related with ID and T. The syntax of RIBE is given as follows.

Definition 2.5 (Revocable IBE, RIBE)**.** An RIBE scheme consists of seven algorithms **Setup**, **GenKey**, **UpdateKey**, **DeriveKey**, **Encrypt**, **Decrypt**, and **Revoke**, which are defined as follows:

Setup(1^λ): The setup algorithm takes as input a security parameter 1^λ. It outputs a master key MK, an (empty) revocation list RL, and public parameters PP.

GenKey(ID, MK, PP): The private key generation algorithm takes as input an identity $ID \in \mathcal{I}$, the master key MK, and public parameters PP. It outputs a private key SK_{ID}.

UpdateKey(T, RL, MK, PP): The update key generation algorithm takes as input update time $T \in \mathcal{T}$, the revocation list RL, the master key MK, and public parameters PP. It outputs an update key UK_T.

DeriveKey(SK_{ID}, UK_T, PP): The decryption key derivation algorithm takes as input a private key SK_{ID}, an update key UK_T, and public parameters PP. It outputs a decryption key $DK_{ID,T}$.

Encrypt(ID, T, M, PP): The encryption algorithm takes as input an identity $ID \in \mathcal{I}$, time T, a message $M \in \mathcal{M}$, and public parameters PP. It outputs a ciphertext $CT_{ID,T}$.

Decrypt($CT_{ID,T}, DK_{ID',T'}, PP$): The decryption algorithm takes as input a ciphertext $CT_{ID,T}$, a decryption key $DK_{ID',T'}$, and public parameters PP. It outputs a message M.

Revoke(ID, T, RL): The revocation algorithm takes as input an identity ID to be revoked and revocation time T, and a revocation list RL. It outputs an updated revocation list RL.

The correctness of RIBE is defined as follows: For all MK, RL, and PP generated by **Setup**(1^λ), SK_{ID} generated by **GenKey**(ID, MK, PP) for any ID, UK_T generated by **UpdateKey**(T, RL, MK, PP) for any T and RL such that $(ID, T_j) \notin RL$ for all $T_j \leq T$, $CT_{ID,T}$ generated by **Encrypt**(ID, T, M, PP) for any ID, T, and M, it is required that

- **Decrypt**($CT_{ID,T}$, **DeriveKey**(SK_{ID}, UK_T, PP)$, PP) = M$.

The security model of RIBE was first defined by Boldyreva et al. [3], and then this security model was extended by Seo and Emura [39] to support decryption key exposure resistance. In the security model of RIBE, an attacker can request a private key query for an identity ID, an update key query for time T, a decryption key query for ID and T, and a revocation query. In the challenge step, the attacker submits a challenge identity ID^*, challenge time T^*, and challenge messages M_0^*, M_1^*, and receives a challenge ciphertext CT^*. Note that the private key query for ID^* is not allowed in the IBE security model, but this private key query for ID^* is allowed in the RIBE security model. At this time, if the private key for ID^* is queried, then the private key for ID^* must be revoked in the update key on the challenge time T^*. The detailed definition of the RIBE security model is given as follows.

Definition 2.6 (IND-CPA Security)**.** The IND-CPA security of RIBE is defined in terms of the following experiment between a challenger $\mathcal{C}$ and a PPT adversary $\mathcal{A}$:

1. **Setup**: $\mathcal{C}$ generates a master key MK, a revocation list RL, a state ST, and public parameters PP by running **Setup**(1^λ). It keeps MK, RL to itself and gives PP to $\mathcal{A}$.

2. **Phase 1**: $\mathcal{A}$ adaptively request a polynomial number of queries. These queries are processed as follows:

    - If this is a private key query for an identity ID, then it gives the corresponding private key SK_{ID} to $\mathcal{A}$ by running **GenKey**(ID, MK, PP).

- If this is an update key query for time T, then it gives the corresponding update key $UK_{T,R}$ to $\mathcal{A}$ by running **UpdateKey**(T, RL, MK, PP).
- If this is a decryption key query for an identity ID and time T, then it gives the corresponding decryption key $DK_{ID,T}$ to $\mathcal{A}$ by running **DeriveKey**(SK_{ID}, UK_T, PP).
- If this is a revocation query for an identity ID and revocation time T, then it updates the revocation list RL by running **Revoke**(ID, T, RL, ST) with the restriction: The revocation query for time T cannot be queried if the update key query for the time T was already requested.

Note that we assume that the update key queries and the revocation queries are requested in non-decreasing order of time.

3. **Challenge**: $\mathcal{A}$ submits a challenge identity ID^*, challenge time T^*, and two challenge messages M_0^*, M_1^* with equal length with the following restrictions:
   - If a private key query for an identity ID such that $ID = ID^*$ was requested, then the identity ID^* should be revoked at some time T such that $T \leq T^*$.
   - The decryption key query for ID^* and T^* was not requested.

   $\mathcal{C}$ flips random $\mu \in \{0,1\}$ and obtains a ciphertext CT^* by running **Encrypt**(ID^*, T^*, M_μ^*, PP). It gives CT^* to $\mathcal{A}$.
4. **Phase 2**: $\mathcal{A}$ may continue to request a polynomial number of additional queries subject to the same restrictions as before.
5. **Guess**: Finally, $\mathcal{A}$ outputs a guess $\mu' \in \{0,1\}$, and wins the game if $\mu = \mu'$.

The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}_{\mathcal{A}}^{RIBE}(\lambda) = \left|\Pr[\mu = \mu'] - \frac{1}{2}\right|$ where the probability is taken over all the randomness of the experiment. An RIBE scheme is IND-CPA secure if for all PPT adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ is negligible in the security parameter λ.

3 Revocable IBE with SD

In this section, we first review the perfect binary tree and the subset difference method, and then we propose a generic construction for RIBE by combining subset difference, HIBE, and SRE schemes.

3.1 Binary Tree

A perfect binary tree $\mathcal{BT}$ is a tree data structure in which all internal nodes have two child nodes and all leaf nodes have the same depth. Let $N = 2^n$ be the number of leaf nodes in $\mathcal{BT}$. The number of all nodes in $\mathcal{BT}$ is $2N-1$ and we denote v_i as a node in $\mathcal{BT}$ for any $1 \leq i \leq 2N-1$. The depth d_i of a node v_i is the length of the path from a root node to the node. The root node of a tree has depth zero. The depth of $\mathcal{BT}$ is the length of the path from the root node to a leaf node. A level of $\mathcal{BT}$ is a set of all nodes at given depth.

Each node $v_i \in \mathcal{BT}$ has an identifier $L_i \in \{0,1\}^*$ which is a fixed and unique string. An identifier of each node is assigned as follows: Each edge in the tree is assigned with 0 or 1 depending on whether it is connected to the left or right child node. The identifier L_i of a node v_i is obtained by reading all labels of edges in a path from the root node to the node v_i. The root node has an empty identifier ε. For a node v_i, we define $Label(v_i)$ be the identifier of v_i and $Depth(v_i)$ be the depth d_i of v_i.

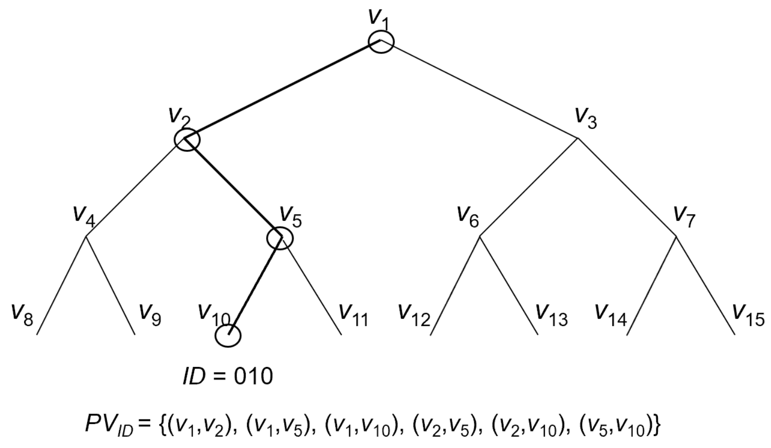

Figure 1: A path set for $ID = 010$ in the SD method

A subtree $\mathcal{T}_i$ in $\mathcal{BT}$ is defined as a tree that is rooted at a node $v_i \in \mathcal{BT}$. A subset S_i is defined as a set of all leaf nodes in $\mathcal{T}_i$. For any two nodes $v_i, v_j \in \mathcal{BT}$ where v_j is a descendant of v_i, $\mathcal{T}_{i,j}$ is defined as a subtree $\mathcal{T}_i - \mathcal{T}_j$, that is, all nodes that are descendants of v_i but not v_j. A subset $S_{i,j}$ is defined as the set of leaf nodes in $\mathcal{T}_{i,j}$, that is, $S_{i,j} = S_i \setminus S_j$.

For a perfect binary tree $\mathcal{BT}$ and a subset R of leaf nodes, $ST(\mathcal{BT}, R)$ is defined as the Steiner Tree induced by the set R and the root node, that is, the minimal subtree of $\mathcal{BT}$ that connects all the leaf nodes in R and the root node.

3.2 Subset Difference Method

The subset difference (SD) method is one instance of the subset cover (SC) framework proposed by Naor et al. [34] which was used for efficient symmetric key broadcast encryption. The SD method is more efficient than the complete subtree (CS) method because the size of the cover set representing the non-revoked users is smaller than that of the CS method. We follow the SD definition of Lee et al. [24]. The SD method uses a perfect binary tree and each user is located at a leaf node in the binary tree. The **Assign** algorithm computes a path set PV, which is consists of subsets associated with the path from the root node to a user's leaf node. The **Cover** algorithm derives a cover set CV that can effectively cover non-revoked leaf nodes. The **Match** algorithm can derive two related subsets if a user's leaf node is not revoked in the cover set. A simple example of the SD method is given in Figure 1 and 2. A detailed description of the SD method is given as follows.

SD.Setup(N): Let $N = 2^n$ be the number of leaf nodes. It sets a perfect binary tree $\mathcal{BT}$ of depth n and outputs $\mathcal{BT}$. Note that a user is assigned to a leaf node in $\mathcal{BT}$ and the collection $\mathcal{S}$ of SD is the set of all subsets $\{S_{i,j}\}$ where $v_i, v_j \in \mathcal{BT}$ and v_j is a descendant of v_i.

SD.Assign($\mathcal{BT}, v$): Let v be the leaf node of $\mathcal{BT}$ that is assigned to a user ID. Let $(v_{k_0}, v_{k_1}, \ldots, v_{k_n})$ be a path from the root node v_{k_0} to the leaf node $v_{k_n} = v$. It initializes a path set PV as an empty one. For all $i, j \in \{k_0, \ldots, k_n\}$ such that v_j is a descendant of v_i, it adds a subset $S_{i,j}$ defined by two nodes v_i and v_j in the path into PV. It outputs the path set $PV = \{S_{i,j}\}$.

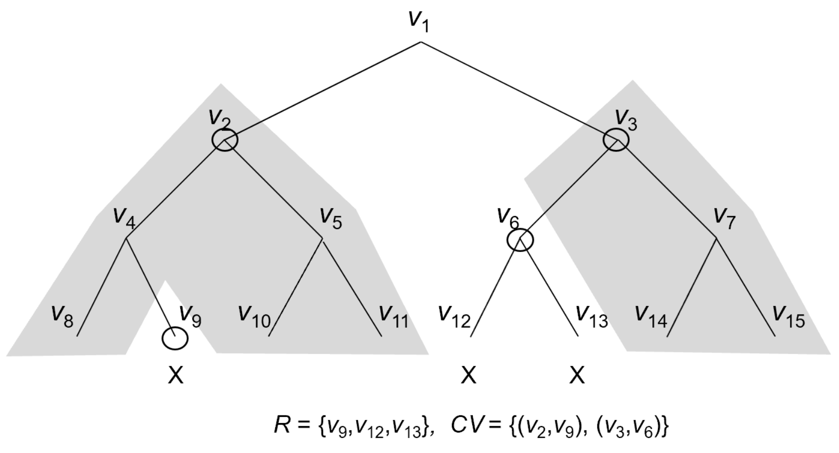

Figure 2: A cover set for $R = \{v_9, v_{12}, v_{13}\}$ in the SD method

SD.Cover($\mathcal{BT}, R$): Let R be a revoked set of leaf nodes (or users). It first sets a subtree $\mathcal{T}$ as $ST(\mathcal{BT}, R)$, and then it builds a cover set CV iteratively by removing nodes from $\mathcal{T}$ until $\mathcal{T}$ consists of just a single node as follows:

1. It finds two leaf nodes v_i and v_j in $\mathcal{T}$ such that the least-common-ancestor v of v_i and v_j does not contain any other leaf nodes of $\mathcal{T}$ in its subtree. Let v_l and v_k be the two child nodes of v such that v_i is a descendant of v_l and v_j is a descendant of v_k. If there is only one leaf node left, it makes $v_i = v_j$ to the leaf node, v to be the root of $\mathcal{T}$ and $v_l = v_k = v$.
2. If $v_l \neq v_i$, then it adds the subset $S_{l,i}$ to CV; likewise, if $v_k \neq v_j$, it adds the subset $S_{k,j}$ to CV_R.
3. It removes from $\mathcal{T}$ all the descendants of v and makes v a leaf node.

It outputs the cover set $CV = \{S_{i,j}\}$.

SD.Match(CV, PV): Let $CV = \{S_{i,j}\}$ and $PV = \{S_{i,j}\}$. It finds two subsets $S_{i,j} \in CV$ and $S_{i',j'} \in PV$ such that $(v_i = v_{i'}) \wedge (d_j = d_{j'}) \wedge (v_j \neq v_{j'})$ where d_j is the depth of v_j. If two subsets exist, then it outputs $(S_{i,j}, S_{i',j'})$. Otherwise, it outputs $\perp$.

The correctness of the SD scheme requires that if $v \notin R$, then **SD.Match**$(CV, PV) = (S_{i,j}, S_{i',j'})$ such that $(v_i = v_{i'}) \wedge (d_j = d_{j'}) \wedge (v_j \neq v_{j'})$ where $S_{i,j}$ is defined by two nodes v_i and v_j.

Lemma 3.1 ([34]). *Let $N = 2^n$ be the number of leaf nodes in a perfect binary tree and r be the size of a revoked set. In the SD method, the size of a path set is $O(\log^2 N)$ where the hidden constant is $1/2$ and the size of a cover set is at most $2r - 1$.*

3.3 Design Principle

In order to design a generic RIBE scheme with the SD method, we first analyze the generic RIBE scheme with the CS method proposed by Ma and Lin [32]. The key design principle of their RIBE scheme with the CS method is that the identity *ID* of a receiver can be fixed to the path of a binary tree and a ciphertext is associated with the path set of the receiver's identity *ID* where as the private key of a user is associated with the path set of a binary tree in directly constructed many RIBE schemes. Therefore, if the receiver's identity ID is not revoked in the CS method, there is a common node in the path set of the binary tree and a node in the cover set of an update key. Thus, the equality function of IBE can be used to handle this common node since the path can be related to IBE ciphertexts and the cover set can be related to IBE private keys.

However, this design method is difficult to apply to the SD method. The reason is that in the SD method, unlike the CS method, there are no common nodes in the path set and the cover set. To solve this problem, we use the new interpretation of the SD method which was used for an efficient public-key revocation (PKR) scheme and RIBE scheme by using the SD method [24, 25]. To design an efficient PKR scheme, Lee et al. [24] observed that the subset $S_{i,j}$ of the SD method can be interpreted as a set of single member revocation instead of the existing interpretation that the subset $S_{i,j}$ is a set of leaf nodes where each leaf node belongs to the subtree $\mathcal{T}_i$ but does not belong to the subtree $\mathcal{T}_j$. That is, if we consider a group set GL which consists of all nodes of the subtree $\mathcal{T}_i$ that has the same depth as the node v_j, the subset $S_{i,j}$ can be interpreted as the same as GL except that the node v_j is excluded from GL. Thus, $S_{i,j}$ can be interpreted as single member revocation because it revokes one node v_j in GL.

This interesting observation was also used to directly construct an RIBE scheme with the SD method by Lee et al. [25]. They used a degree-one polynomial in the exponent to implement single member revocation, but they only achieved an RIBE scheme in a non-generic way. In this work, we found that an SRE scheme can be used in a generic way to achieve single member revocation if an RIBE ciphertext is associated with a path set PV for a receiver's identity ID and an RIBE update key is associated with a cover set CV for a revoked set R. That is, given the subset $S_{i,j}$, if we set a group label $GL = L_i \| d_j$ and a member label $ML = L_j$ where L_i, L_j are identifiers of nodes v_i, v_j and d_j is the depth of v_j, then all members of the group GL can be represented by a label pair (GL, ML). In this case, a label pair (GL, ML) in a ciphertext and another label pair (GL', ML') in an update key can be matching pairs if the group labels are equal but the member labels are different such that $GL = GL' \wedge ML \neq ML'$. Thus, we can support the equality $GL = GL'$ and the inequality $ML \neq ML'$ by using an SRE scheme. In addition, to provide security against collusion attacks in the black-box construction, we divided the message M of a ciphertext into several secret shares by using a simple secret sharing scheme, and then encrypt these shares by using HIBE and SRE schemes. Additionally, we use the HIBE scheme to provide the decryption key exposure resistance.

3.4 Generic Construction

Let **HIBE** = (**Setup**, **GenKey**, **Delegate**, **Encrypt**, **Decrypt**) be a two-level HIBE scheme and **SRE** = (**Setup**, **GenKey**, **Encrypt**, **Decrypt**) be an SRE scheme that supports a single revoked identity. We define $GMLabels(S_{i,j}) = (GL = Label(v_i) \| Depth(v_j), ML = Label(v_j))$ where GL is a group label and ML is a member label. A simple example of a group of nodes derived from a subset $S_{i,j}$ is given in Figure 3. A generic RIBE scheme using the SD method is described as follows.

RIBE.Setup(1^λ)**:** Let $\mathcal{I} = \{0,1\}^n$ be the identity space.

1. It first obtains MK_{HIBE}, PP_{HIBE} by running **HIBE.Setup**$(1^\lambda, 2)$. It also obtains MK_{SRE}, PP_{SRE} by running **SRE.Setup**(1^λ).
2. It defines a binary tree $\mathcal{BT}$ by running **SD.Setup**(2^n) where $\mathcal{I} \in \{0,1\}^n$. Note that it will deterministically assign an identity ID to a leaf node $v \in \mathcal{BT}$ such that $Label(v) = ID$.
3. It outputs a master key $MK = (MK_{HIBE}, MK_{SRE})$, a revocation list $RL = \emptyset$, and public parameters $PP = (PP_{HIBE}, PP_{SRE}, \mathcal{BT})$.

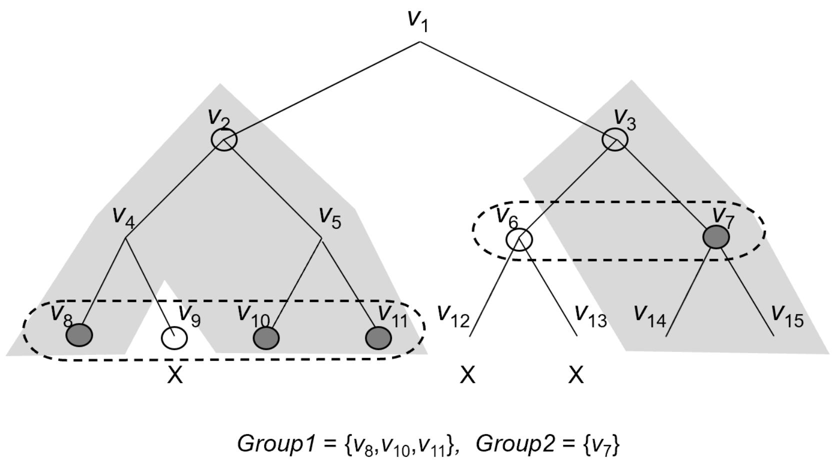

Figure 3: Single member revoked groups from the subsets $S_{2,9} = (v_2, v_9)$ and $S_{3,6} = (v_3, v_6)$

RIBE.GenKey(ID, MK, PP**):** It first obtains SK_{HIBE} by running **HIBE.GenKey**$((ID), MK_{HIBE}, PP_{HIBE})$. It outputs a private key $SK_{ID} = SK_{HIBE}$.

RIBE.UpdateKey(T, RL, MK, PP**):** To generate an update key for T, it proceeds as follows:

1. It initializes $RV = \emptyset$. For each $(ID_j, T_j) \in RL$, it adds a leaf node $v_j \in \mathcal{BT}$ which is associated with ID_j into RV if $T_j \leq T$. It obtains CV_T by running **SD.Cover**$(\mathcal{BT}, RV)$.
2. For each $S_{i,j} \in CV_T$, it sets labels $(GL, ML) = GMLabels(S_{i,j})$ and obtains $SK_{SRE,S_{i,j}}$ by running **SRE.GenKey**$((GL\|T, ML), MK_{SRE}, PP_{SRE})$.
3. Finally, it outputs an update key $UK_T = \big(CV_T, \{SK_{SRE,S_{i,j}}\}_{S_{i,j} \in CV_T}\big)$.

RIBE.DeriveKey(SK_{ID}, UK_T, PP**):** Let $SK_{ID} = SK_{HIBE}$. It first obtains DK_{HIBE} by running **HIBE.Delegate** $((ID, T), SK_{HIBE}, PP_{HIBE})$. It outputs a decryption key $DK_{ID,T} = (DK_{HIBE}, UK_T)$.

RIBE.Encrypt(ID, T, M, PP**):** To generate a ciphertext for ID and T, it proceeds as follows:

1. It selects random R_1 and sets $R_2 = M \oplus R_1$. It obtains CT_{HIBE} by running **HIBE.Encrypt**$((ID, T), R_1, PP_{HIBE})$.
2. Let v_{ID} be a leaf node associated with ID such that $ID = Label(v_{ID})$. Recall that the leaf node v_{ID} is fixed by the $Label$ function. It obtains PV_{ID} by running **SD.Assign**$(\mathcal{BT}, v_{ID})$.
3. For each $S_{i,j} \in PV_{ID}$, it sets labels $(GL, ML) = GMLabels(S_{i,j})$ and obtains $CT_{SRE,S_{i,j}}$ by running **SRE.Encrypt**$((GL\|T, ML), R_2, PP_{SRE})$. It creates $CT_{PV} = \big(PV_{ID}, \{CT_{SRE,S_{i,j}}\}_{S_{i,j} \in PV_{ID}}\big)$.
4. Finally, it outputs a ciphertext $CT_{ID,T} = (CT_{HIBE}, CT_{PV})$.

RIBE.Decrypt($CT_{ID,T}, DK_{ID',T'}, PP$**):** Let $CT_{ID,T} = (CT_{PV}, CT_{HIBE})$ and $DK_{ID',T'} = (DK_{HIBE}, UK_T)$. It proceeds as follows:

1. It first obtains R_1 by running **HIBE.Decrypt**$(CT_{HIBE}, DK_{HIBE}, PP_{HIBE})$.
2. It finds $(S_{i,j}, S_{i',j'}) =$ **SD.Match**(PV_{ID}, CV_T). It retrieves $CT_{SRE,S_{i,j}}$ from CT_{PV} and $SK_{SRE,S_{i',j'}}$ from UK_T. Next, it obtains R_2 by running **SRE.Decrypt**$(CT_{SRE,S_{i,j}}, SK_{SRE,S_{i',j'}}, PP_{SRE})$.

3. Finally, it outputs a message $M = R_1 \oplus R_2$.

RIBE.Revoke(ID, T, RL)**:** If $(ID, *)$ already exists in RL, it outputs RL. Otherwise, it adds (ID, T) to RL and outputs the updated RL.

3.5 Correctness

The correctness of the above RIBE scheme can be easily seen by using the correctness of the underlying HIBE, SRE and SD schemes. Let $CT_{ID,T} = (CT_{HIBE}, CT_{PV})$ be a ciphertext associated with ID and T, and $DK_{ID',T'} = (DK_{HIBE}, UK_T)$ be a decryption key is associated with ID' and T'. In this case, if the condition $ID = ID' \wedge T = T'$ is satisfied, then the random R_1 is correctly decrypted by running **HIBE.Decrypt**$(CT_{HIBE}, SK_{HIBE}, PP_{HIBE})$ because of the correctness of HIBE.

Now we show that random R_2 can be correctly decrypted from CT_{PV} and UK_T if the identity ID of the ciphertext is not revoked in the update key UK_T. Recall that the ciphertext CT_{PV} is associated with PV_{ID} and the update key UK_T is associated with CV_T. By the correctness of the SD scheme, the **SD.Match** algorithm outputs two subsets of $S_{i,j}, S_{i',j'}$ such that $(v_i = v_{i'}) \wedge (d_j = d_{j'}) \wedge (v_j \neq v_{j'})$ if the leaf node v_{ID} is not included in the revoked set RV. Let $CT_{SRE,S_{i,j}} \in CT_{PV}$ and $SK_{SRE,S_{i',j'}} \in UK_T$ be corresponding ciphertext and private key of $S_{i,j}$ and $S_{i',j'}$ respectively. From the definition of $GMLabels$, labels $(GL, ML) = GMLabels(S_{i,j})$ and $(GL', ML') = GMLabels(S_{i',j'})$ are obtained and they satisfy $GL = GL' \wedge ML \neq ML'$. Therefore, if the time T of the ciphertext is the same as the time T' of the update key, then random R_2 can be decrypted by running **SRE.Decrypt**$(CT_{SRE,S_{i,j}}, SK_{SRE,S_{i',j'}}, PP_{SRE})$ because of $GL \| T = GL' \| T'$ and $ML \neq ML'$ by the correctness of SRE.

3.6 Discussions

Layered Subset Difference. Since our generic RIBE scheme uses the SD method, the size of a ciphertext depends on the size of the PV set and the size of an update key depends on the size of the CV set in the SD method. Thus, the ciphertext and update key of generic RIBE consists of approximately $O(\log^2 N)$ IBE ciphertexts and $2r$ IBE private keys respectively where $N = 2^n$ is the number of users and r is the number of revoked users. In order to reduce the size of ciphertexts in this generic RIBE scheme, we can apply the layered subset difference (LSD) method of Halevy and Shamir [16]. If the LSD method is used instead of the SD method, the ciphertext and the update key of this general RIBE scheme consists of $O(\log^{1.5} N)$ IBE ciphertexts and $4r$ IBE private keys, respectively.

Chosen-Ciphertext Security. The CCA security model, which is stronger than the CPA security model, allows an adversary to request decryption queries on ciphertexts. The above generic RIBE construction only can derive a CPA secure RIBE scheme by using CPA secure HIBE and SRE schemes as building blocks. To derive a CCA secure RIBE scheme, we may try to use CCA secure encryption primitives as building blocks. However, this simple construction can not be CCA secure because it allows ciphertext elements reordering attacks. To solve this problem, we apply the CCA methodology for multiple encryption proposed by Dodis and Katz [12]. That is, a CCA secure RIBE scheme can be constructed by combining CCA secure HIBE and SRE schemes with a one-time signature (OTS) scheme with strong unforgeability. At this time, the underlying HIBE and SRE schemes should be modified to receive additional labels as inputs since the public key of OTS should be tied with ciphertexts. This approach also provides the decryption key exposure resistance (DKER) property since a decryption key is generated by using the delegation property of HIBE.

4 Security Analysis

In this section, we prove the IND-CPA security of the generic RIBE construction proposed in the previous section. The basic idea of this proof is to show that if there is an attacker that breaks the IND-CPA security of the RIBE scheme, then we can construct an algorithm that breaks the IND-CPA security of underlying HIBE or SRE schemes. In order to simplify the security proof, we try to prove the security by separating the attacker into two types. That is, the Type-I attacker does not request a private key query on the challenge identity ID^*, and the Type-II attacker requests a private key query on the identity ID^*.

First, since the Type-I attacker does not query the private key for the identity ID^*, we perform the proof that relates the security of the underlying HIBE scheme with the security of the RIBE scheme. Next, since the Type-II attacker queries the private key for ID^*, we perform the proof that relates the security of the underlying SRE scheme and the security of the RIBE scheme.

Theorem 4.1. *The generic RIBE scheme is IND-CPA secure if the underlying HIBE and SRE schemes are IND-CPA secure.*

Proof. Let ID^* be the challenge identity and T^* be the challenge time. We divide the behavior of an adversary as two types: Type-I and Type-II, which are defined as follows:

Type-I. An adversary is Type-I if it requests a private key for $ID \neq ID^*$ for all private key queries. In this case, the adversary can request a decryption key for ID and T such that $ID \neq ID^*$ or $ID = ID^* \wedge T \neq T^*$.

Type-II. An adversary is Type-II if it requests a private key for $ID = ID^*$ for some private key query. In this case, the private key for ID^* should be revoked at some time T such that $T \leq T^*$ by the restriction of the security model.

Let E_i be the event that $\mathcal{A}$ behaves like Type-i adversary. From Lemmas 4.2 and 4.3, we obtain the following result

$$\begin{aligned}\mathbf{Adv}_{\mathcal{A}}^{RIBE}(\lambda) &\leq \Pr[E_I] \cdot \mathbf{Adv}_{\mathcal{A}}^{RIBE}(\lambda) + \Pr[E_{II}] \cdot \mathbf{Adv}_{\mathcal{A}}^{RIBE}(\lambda) \\ &\leq \mathbf{Adv}_{\mathcal{B}}^{HIBE}(\lambda) + O(n^2)\mathbf{Adv}_{\mathcal{B}}^{SRE}(\lambda).\end{aligned}$$

This completes our proof. □

4.1 Type-I Adversary

The Type-I attacker does not request a private key query on the challenge ID^*, but can request decryption key queries such that $ID = ID^*$ and $T \neq T^*$. To deal with this attacker, we build a reduction algorithm that attacks an HIBE scheme and selects an SRE scheme by itself. In this case, this algorithm will be able to handle all queries of the Type-I attacker by using the queries for the HIBE scheme. The detailed proof is as follows.

Lemma 4.2. *For the Type-I adversary, the generic RIBE scheme is IND-CPA secure if the HIBE scheme is IND-CPA secure.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that attacks the RIBE scheme with a non-negligible advantage. An algorithm $\mathcal{B}$ that attacks the HIBE scheme is initially given public parameters PP_{HIBE} by a challenger $\mathcal{C}$. Then $\mathcal{B}$ that interacts with $\mathcal{A}$ is described as follows:

Setup: $\mathcal{B}$ generates MK_{SRE}, PP_{SRE} by running the **SRE.Setup** algorithm. It initializes $RL = \emptyset$ and gives $PP = (PP_{HIBE}, PP_{SRE}, \mathcal{BT})$ to $\mathcal{A}$.

Phase 1: $\mathcal{A}$ adaptively requests a polynomial number of private key, update key, decryption key, and revocation queries.

- For a private key query with an identity ID, $\mathcal{B}$ proceeds as follows: It receives SK_{HIBE} from $\mathcal{C}$ by querying a private key for ID since $ID \neq ID^*$ by the restriction of the Type-I adversary. It gives $SK_{ID} = SK_{HIBE}$ to $\mathcal{A}$.
- For an update key query with time T, $\mathcal{B}$ proceeds as follows: It simply generates UK_T by running the **RIBE.UpdateKey** algorithm since it knows MK_{SRE}. It gives UK_T to $\mathcal{A}$.
- For a decryption key query with an identity ID and time T, $\mathcal{B}$ proceeds as follows:
  1. It generates UK_T by running the **RIBE.UpdateKey** algorithm since it knows MK_{SRE}.
  2. It receives DK_{HIBE} from $\mathcal{C}$ by querying a private key for ID and T since $ID \neq ID^*$ or $ID = ID^* \wedge T \neq T^*$ by the restriction of the Type-I adversary.
  3. It gives $DK_{ID,T} = (DK_{HIBE}, UK_T)$ to $\mathcal{A}$.
- For a revocation query with an identity ID and time T, $\mathcal{B}$ proceeds as follows: It adds (ID, T) to RL if ID was not revoked before.

Challenge: $\mathcal{A}$ submits a challenge identity ID^*, challenge time T^*, and two challenge messages M_0^*, M_1^*. $\mathcal{B}$ proceeds as follows:

1. It first selects random R_2 and sets $R_{1,0} = M_0^* \oplus R_2, R_{1,1} = M_1^* \oplus R_2$. Next, it receives CT_{HIBE}^* from $\mathcal{C}$ by submitting ID^*, T^*, and two challenge messages $R_{1,0}, R_{1,1}$.
2. To creates CT_{PV}^* for ID^* and T^*, it simply runs the **RIBE.Encrypt** algorithm with the random R_2 as input.
3. It gives a challenge ciphertext $CT^* = (CT_{HIBE}^*, CT_{PV}^*)$ to $\mathcal{A}$.

Phase 2: Same as Phase 1.

Guess: Finally, $\mathcal{A}$ outputs a guess $\mu' \in \{0,1\}$. $\mathcal{B}$ also outputs μ'.

To complete the proof of this lemma, we need to analyze that the simulation described above is correct. For this, it is sufficient to check whether HIBE private key queries requested by the simulator satisfy the constraints of the HIBE security model. The simulator requests an HIBE private key query when processing an RIBE private key query of an adversary, and can only request an HIBE private key with the condition $ID \neq ID^*$ by the constraints of a Type-I adversary. And the simulator gets an HIBE challenge ciphertext for the challenge hierarchical identity (ID^*, T^*) when generating a challenge ciphertext. Therefore, the HIBE private key queries requested by the simulator satisfy the constraints of the HIBE security model since they does not correspond to the prefix of the challenge hierarchical identity. $\square$

4.2 Type-II Adversary

Since the Type-II attacker requests a private key query on the challenge ID^*, we can not handle the private key queries of the RIBE scheme by using the private key queries of the HIBE scheme in the proof. Therefore, we prove the security by relating the security of the SRE scheme with the security of the RIBE scheme against the Type-II attacker.

The main idea of the proof is to take advantage of the restriction of the RIBE security model such that if the attacker queries the private key for the challenge identity ID^*, then the corresponding private key for ID^* must be revoked from the update key on the challenge time T^*. Thus, the ciphertext CT^*_{PV} in the challenge ciphertext consists of the SRE ciphertexts associated with the subset $S_{i,j}$ belonging to the path set PV_{ID^*}, but the SRE private keys that can decrypt the corresponding ciphertext elements in CT^*_{PV} are not included in the update key for T^* because of the restriction. Using this fact, we can prove the security of the RIBE scheme against the Type-II attacker by using the security of the SRE scheme.

We prove the security by using hybrid games consisting of multiple sub-games because the ciphertext CT^*_{PV} is composed of many SRE ciphertexts. That is, in the hybrid games, a ciphertext which encrypts a random value related to M^*_0 is changed to another ciphertext which encrypts a random value related to M^*_1. In this hybrid steps, since the number of SRE ciphertexts in CT^*_{PV} is maximum $O(n^2)$, the proof can be completed by performing $O(n^2)$ hybrid games. The detailed proof is described as follows.

Lemma 4.3. *For the Type-II adversary, the generic RIBE scheme is IND-CPA secure if the SRE scheme is IND-CPA secure.*

Proof. Let ID^* be the challenge identity and PV_{ID^*} be the path set of ID^* where the number of subsets in PV_{ID^*} is $\ell = n(n-1)/2$. The challenge ciphertext is formed as $CT^* = (CT^*_{HIBE}, CT^*_{PV})$ where $CT^*_{PV} = (PV_{ID^*}, \{CT^*_{SRE,S_{i_k,j_k}}\}_{k=1}^{\ell})$. For the security proof, we define hybrid games $\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2, \mathbf{G}_3$ as follows:

Game $\mathbf{G}_0$. This game is the original security game defined in the security model except that the challenge bit μ is fixed to 0.

Game $\mathbf{G}_1$. This game is the same as the game $\mathbf{G}_0$ except that the settings of random R_1 and R_2 in the challenge ciphertext are changed. That is, R_2 are randomly chosen and R_1 is set as $M^*_0 \oplus R_2$.

Game $\mathbf{G}_2$ In this game, the generation of CT^*_{PV} in the challenge ciphertext CT^* is changed. That is, a random $R'_1 = M^*_1 \oplus R_2$ is encrypted instead of $R_1 = M^*_0 \oplus R_2$ to generate CT^*_{PV}.

For the analysis of security, we define additional sub-games $\mathbf{H}_0, \ldots, \mathbf{H}_\rho, \ldots, \mathbf{H}_\ell$ where $\mathbf{H}_0 = \mathbf{G}_1$ and $\mathbf{H}_\ell = \mathbf{G}_2$. The game $\mathbf{H}_\rho$ is similar to the game $\mathbf{H}_{\rho-1}$ except that $CT^*_{SRE,S_{i_\rho,j_\rho}}$ is an encryption on the random $R'_1 = M^*_1 \oplus R_2$. Specifically, $CT^*_{SRE,S_{i_k,j_k}}$ for $k \le \rho$ is an encryption on the random $R'_1 = M^*_1 \oplus R_2$ and $CT^*_{SRE,S_{i_k,j_k}}$ for $k > \rho$ is an encryption on the random $R_1 = M^*_0 \oplus R_2$.

Game $\mathbf{G}_3$ This game is the same as the game $\mathbf{G}_2$ except that the settings of random R'_1 and R_2 in the challenge ciphertext are changed. That is, R'_1 is randomly chosen and R_2 is set as $M^*_1 \oplus R'_1$. This game is the original security game in the security model except that the challenge bit μ is fixed to 1.

Let $S^{G_i}_{\mathcal{A}}$ be the event that $\mathcal{A}$ outputs 0 in a game $\mathbf{G}_i$. From Lemma 4.4, we obtain the following result

$$\begin{aligned}\mathbf{Adv}^{RIBE}_{\mathcal{A}}(\lambda) &\le \frac{1}{2}\left|\Pr[S^{G_0}_{\mathcal{A}}] - \Pr[S^{G_3}_{\mathcal{A}}]\right| \le \frac{1}{2}\left|\Pr[S^{G_1}_{\mathcal{A}}] - \Pr[S^{G_2}_{\mathcal{A}}]\right| \\ &\le \frac{1}{2}\Big(\sum_{\rho=1}^{\ell}\left|\Pr[S^{H_{\rho-1}}_{\mathcal{A}}] - \Pr[S^{H_\rho}_{\mathcal{A}}]\right|\Big) \le O(n^2)\mathbf{Adv}^{SRE}_{\mathcal{B}}(\lambda).\end{aligned}$$

This completes our proof. □

Lemma 4.4. *If the SRE scheme is IND-CPA secure, then no polynomial-time Type-II adversary can distinguish between $\mathbf{H}_{\rho-1}$ and $\mathbf{H}_{\rho}$ with a non-negligible advantage.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that attacks the RIBE scheme with a non-negligible advantage. An algorithm $\mathcal{B}$ that attacks the SRE scheme is initially given public parameters PP_{SRE} by a challenger $\mathcal{C}$. Then $\mathcal{B}$ that interacts with $\mathcal{A}$ is described as follows:

Setup: $\mathcal{B}$ generates MK_{HIBE}, PP_{HIBE} by running the **HIBE.Setup** algorithm. It initializes $RL = \emptyset$ and gives $PP = (PP_{HIBE}, PP_{SRE}, \mathcal{BT})$ to $\mathcal{A}$.

Phase 1: $\mathcal{A}$ adaptively requests a polynomial number of private key, update key, decryption key, and revocation queries.

- For a private key query with an identity ID, $\mathcal{B}$ proceeds as follows: It generates SK_{ID} by running the **RIBE.GenKey** algorithm since it knows MK_{HIBE}. It gives SK_{ID} to $\mathcal{A}$.
- For an update key query with time T, $\mathcal{B}$ proceeds as follows:
  1. It initializes $RV = \emptyset$. For each $(ID_j, T_j) \in RL$, it adds a leaf node $v_j \in \mathcal{BT}$ into RV if $T_j \leq T$. It obtains CV_T by running **SD.Cover**$(\mathcal{BT}, RV)$.
  2. For each $S_{i,j} \in CV_T$, it sets $(GL_k, ML_k) = GMLabels(S_{i,j})$ and receives $SK_{SRE,S_{i,j}}$ from $\mathcal{C}$ by submitting labels $(GL_k \| T, ML_k)$.
  3. It creates $UK_T = \big(CV_T, \{SK_{SRE,S_{i,j}}\}_{S_{i,j} \in CV_T}\big)$ and gives UK_T to $\mathcal{A}$.
- For a decryption key query with an identity ID and time T, $\mathcal{B}$ proceeds as follows:
  1. It retrieves $SK_{ID} = SK_{HIBE}$ by querying a private key to its own oracle. It also retrieves UK_T by querying an update key to its own oracle.
  2. Next, it generates a delegated key DK_{HIBE} of SK_{HIBE} by running the **HIBE.DelegateKey** algorithm for ID and T.
  3. It gives $DK_{ID,T} = (DK_{HIBE}, UK_T)$ to $\mathcal{A}$.
- For a revocation query with an identity ID and time T, $\mathcal{B}$ adds (ID, T) to RL if ID was not revoked before.

Challenge: $\mathcal{A}$ submits a challenge identity ID^*, challenge time T^*, and two challenge messages M_0^*, M_1^*. $\mathcal{B}$ proceeds as follows:

1. It first selects random R_1 and sets $R_{2,0} = M_0^* \oplus R_1, R_{2,1} = M_1^* \oplus R_1$. Next, it generates CT_{HIBE}^* by running **HIBE.Encrypt**$((ID^*, T^*), R_1, PP_{HIBE})$.
2. It obtains PV_{ID^*} by running **SD.Assign**$(\mathcal{BT}, v_{ID^*})$ where a leaf node v_{ID^*} is associated with ID^*. For each $S_{i,j} \in PV_{ID^*}$, it obtains $(GL_k, ML_k) = GMLabels(S_{i,j})$ and proceeds as follows:
   - If $k < \rho$, then it generates $CT_{IBE,S_{i,j}}^*$ by running **SRE.Encrypt**$((GL_k \| T^*, ML_k), R_{2,1}, PP_{SRE})$.
   - If $k = \rho$, then it receives $CT_{SRE,S_{i,j}}^*$ from $\mathcal{C}$ by submitting challenge labels $(GL_k \| T^*, ML_k)$ and challenge messages $R_{2,0}, R_{2,1}$.
   - If $k > \rho$, then it generates $CT_{SRE,S_{i,j}}^*$ by running **SRE.Encrypt**$((GL_k \| T^*, ML_k), R_{2,0}, PP_{SRE})$.

   It creates $CT_{PV}^* = \big(PV_{ID^*}, \{CT_{SRE,S_{i,j}}^*\}_{S_{i,j} \in PV_{ID^*}}\big)$.

3. It gives a challenge ciphertext $CT^* = (CT^*_{HIBE}, CT^*_{PV})$ to $\mathcal{A}$.

Phase 2: Same as Phase 1.
Guess: Finally, $\mathcal{A}$ outputs a guess $\mu' \in \{0,1\}$. $\mathcal{B}$ also outputs μ'.

To complete the proof of this lemma, we need to analyze that the simulation described above is correct. For this, it is sufficient to check whether the SRE private key query and the SRE challenge ciphertext requested by the simulator satisfy the constraints of the SRE security model. First, let ID^* and T^* be the challenge identity and challenge time of the RIBE scheme, and $(GL^*_k \| T^*, ML^*_k)$ be the challenge labels of the SRE scheme. In order to facilitate the analysis, we show that the simulator does not have any problem in generating the SRE private keys which are elements of the update key, in the following three cases.

- Case $T \neq T^*$: In order for the simulator to generate an update key, it must query an SRE private key for labels $(GL_k \| T, ML_k)$. If $T \neq T^*$ is established by the SRE security model, the simulator can query the SRE private key since $GL_k \| T \neq GL^*_k \| T^*$ is established.

- Case $T = T^* \wedge GL_k \neq GL^*_k$: The simulator must query an SRE private key for labels $(GL_k \| T, ML_k)$ to generate the update key. If $GL_k \neq GL^*_k$, the simulator can query the SRE private key since $GL_k \| \neq GL^*_k \| T^*$ is established.

- Case $T = T \wedge GL_k = GL^*_k$: To analyze this case, we use the constraints of the RIBE security model that the RIBE private key corresponding to ID^* should be revoked in the update key at time T^*. The subset $S^*_{i,j}$ corresponding to the labels (GL^*_k, ML^*_k) of the challenge ciphertext is defined by two nodes (v^*_i, v^*_j). Since $GL_k = GL^*_k$ is established, the SRE private key must also be requested for the SRE private key associated with the subset $S_{i,j}$ such that $v_i = v^*_i$ and $Depth(v_j) = Depth(v^*_j)$. However, the leaf node v^*_{ID} revoked by the Cover algorithm of the SD scheme must be located in the descendant nodes of the node v^*_j. Therefore, since $v_j = v^*_j$ is established, $ML_k = ML^*_k$ is obtained. Therefore, it is possible to query the private key of the label $(GL_k \| T, ML_k) = (GL^*_k \| T^*, ML^*_k)$ due to the constraints of the SRE security model.

This completes our proof. $\square$

5 Instantiations

In this section, we show that our generic RIBE construction can be instantiated as real RIBE schemes by using bilinear maps or lattices.

5.1 RIBE from Bilinear Maps

Previously, many RIBE schemes using the CS method were directly constructed on bilinear maps [3,31,39]. In addition, an RIBE scheme using the SD method was also directly constructed on bilinear maps [25]. Recently, a generic construction for RIBE using the CS method was proposed by Ma and Lin [32]. Nonetheless, different generic construction for RIBE using the SD/LSD method is still an interesting method because it allows different RIBE instantiations by changing the underlying cryptographic schemes and allows RIBE schemes with shorter update keys. Here, we will look at different instantiations of RIBE using the SD/LSD method that provide selective security or adaptive security.

First, we instantiate an efficient RIBE scheme that provides selective security by following the generic construction. To do this, we choose the two-level BB-HIBE scheme of Boneh and Boyen [4] that provides selective security in the DBDH assumption. For underlying SRE scheme, we choose the efficient SRE scheme of Lee and Park [27] which provides selective security in the DBDH assumption. For reference, we described the SRE scheme of Lee and Park in Appendix A. The resulting RIBE scheme that uses the SD/LSD method provides selective security under the DBDH assumption. We analyze the private key, update key, and ciphertext size of our generic RIBE scheme with the LSD method in an asymmetric bilinear group. In the MNT159 bilinear group, the size of the $\mathbb{G}$ group is 159 bits, and the size of the $\hat{\mathbb{G}}$ group and the $\mathbb{G}_T$ group is 954 bits. In the BB-HIBE scheme, the private key size is $2|\hat{\mathbb{G}}|$ and the ciphertext size is $3|\mathbb{G}| + |\mathbb{G}_T|$. In the LP-SRE scheme, the private key size is $4|\hat{\mathbb{G}}|$ and the ciphertext size is $3|\mathbb{G}| + |\mathbb{G}_T|$ where $|\mathbb{G}|$ denotes the size of a group element. In our RIBE scheme, the private key size is $2|\hat{\mathbb{G}}|$ since it consists of the private key of HIBE, and the update key size is $16 * r * |\hat{\mathbb{G}}|$ since it is composed of SRE private keys associated with a cover set, and the ciphertext size is approximately $0.5 * \log^{1.5} N * (3|\mathbb{G}| + |\mathbb{G}_T|)$ since it consists of SRE ciphertexts associated with a path set. Thus, if we set $N = 2^{32}$ and $r = 1000$, the private key size is 238 bytes, the update key size is 1908 kilobytes, and the ciphertext size is 16 kilobytes.

Next, we instantiate an RIBE scheme that provides adaptive security. To this security, we use the two-level HIBE scheme of Lewko and Waters [30] which provides adaptive security and the SRE scheme of Lee and Park [27]. The resulting RIBE scheme provides adaptive security under static assumptions in composite-order bilinear groups.

5.2 RIBE from Lattices

A number of RIBE schemes in lattices have been previously proposed [9,10,18,43]. Although the first lattice based RIBE scheme using the CS method did not provide decryption key exposure resistance (DKER), the new RIBE scheme using the CS method that allows DKER was recently proposed by using the delegation property of HIBE [9, 18]. In addition, a lattice based RIBE scheme using the SD method also has been proposed, but this scheme has a serious limitation such that the identity space is restricted to be small universe because the Lagrange interpolation technique is directly applied to lattices [10].

We use the previously proposed efficient lattice based two-level HIBE and SRE schemes to instantiate a lattice based RIBE scheme using the SD method. For the underlying HIBE scheme, we choose efficient HIBE scheme of Agrawal et al. [1] that provide selective security in the LWE assumption. For the underlying SRE scheme, we choose the ABE scheme for circuits [6, 14] since an SRE scheme can be instantiated from an ABE scheme that supports equality and inequality gates. Alternatively, we may modify the NIPE scheme [19] to handle equality by using the technique of HIBE [1].

We compare our RIBE scheme with the SD method and the RIBE scheme directly designed by Cheng and Zhang [10]. Cheng and Zhang derived their RIBE scheme in lattices by applying the design principle of the RIBE scheme of Lee et al. [25]. To use the technique of Lee et al., it is necessary to use the Lagrange interpolation to recover a polynomial value in decryption. In lattices, if Lagrange coefficients and noise values in ciphertexts are multiplied, then a large noise value is obtained in the decryption process, which should be removed to obtain a message. Since the resulting noise value is exponentially increased as the size of the identity space increases, their RIBE scheme has a serious problem that only a small universe of identity can be accepted. Therefore, our RIBE scheme with the SD method is the first lattice based RIBE scheme using the SD method that supports a large universe of identity and provides the DKER property.

6 Conclusion

In this paper, we proposed a new generic RIBE construction with the SD method. Our generic construction uses an HIBE scheme and SRE scheme as building blocks. The generic RIBE construction can be instantiated by bilinear maps or lattices, and the private key consists of an HIBE private key, the update key consists of $O(r)$ number of SRE private keys, and the ciphertext consists of $O(n^2)$ number of SRE ciphertexts. If our generic RIBE construction is extended to use the more efficient LSD method instead of the SD method, the ciphertext is reduced to $O(n^{1.5})$ number of SRE ciphertexts. In addition, if the underlying HIBE and SRE schemes provide the CCA security and a one-time signature is used, then a CCA secure RIBE scheme can be generically constructed.

There are some interesting open problems. The first problem is to reduce the size of a ciphertext in our generic RIBE scheme with the SD method. In the previous generic RIBE scheme with the CS method, the size of a ciphertext can be reduced by using an IBBE scheme. In our generic RIBE scheme with the SD method, it is difficult to reduce the size of a ciphertext since it uses an SRE scheme. The second problem is to design a generic RHIBE scheme with the SD method. To design a generic RHIBE scheme, the private key delegation is needed. It is unclear how to extend the SRE scheme to support key delegation.

Acknowledgements

We thank Atsushi Takayasu for pointing out the attack on our previous RIBE scheme.

References

[1] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 553–572. Springer, 2010.

[2] William Aiello, Sachin Lodha, and Rafail Ostrovsky. Fast digital identity revocation (extended abstract). In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 137–152. Springer, 1998.

[3] Alexandra Boldyreva, Vipul Goyal, and Virendra Kumar. Identity-based encryption with efficient revocation. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM Conference on Computer and Communications Security - CCS 2008*, pages 417–426. ACM, 2008.

[4] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 2004.

[5] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, editor, *Advances in Cryptology - CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.

[6] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic

circuit ABE and compact garbled circuits. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology - EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 533–556. Springer, 2014.

[7] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Victor Shoup, editor, *Advances in Cryptology - CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 258–275. Springer, 2005.

[8] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In Salil P. Vadhan, editor, *Theory of Cryptography - TCC 2007*, volume 4392 of *Lecture Notes in Computer Science*, pages 535–554. Springer, 2007.

[9] Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Khoa Nguyen. Revocable identity-based encryption from lattices. In Willy Susilo, Yi Mu, and Jennifer Seberry, editors, *Information Security and Privacy - ACISP 2012*, volume 7372 of *Lecture Notes in Computer Science*, pages 390–403. Springer, 2012.

[10] Shantian Cheng and Juanyang Zhang. Adaptive-id secure revocable identity-based encryption from lattices via subset difference method. In Javier López and Yongdong Wu, editors, *Information Security Practice and Experience - ISPEC 2015*, volume 9065 of *Lecture Notes in Computer Science*, pages 283–297. Springer, 2015.

[11] Cécile Delerablée. Identity-based broadcast encryption with constant size ciphertexts and private keys. In Kaoru Kurosawa, editor, *Advances in Cryptology - ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 200–215. Springer, 2007.

[12] Yevgeniy Dodis and Jonathan Katz. Chosen-ciphertext security of multiple encryption. In Joe Kilian, editor, *Theory of Cryptography - TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 188–209. Springer, 2005.

[13] Craig Gentry and Alice Silverberg. Hierarchical id-based cryptography. In Yuliang Zheng, editor, *Advances in Cryptology - ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer, 2002.

[14] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13*, pages 545–554. ACM, 2013.

[15] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security - CCS 2006*, pages 89–98. ACM, 2006.

[16] Dani Halevy and Adi Shamir. The LSD broadcast encryption scheme. In Moti Yung, editor, *Advances in Cryptology - CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 47–60. Springer, 2002.

[17] Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In Lars R. Knudsen, editor, *Advances in Cryptology - EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 466–481. Springer, 2002.

[18] Shuichi Katsumata, Takahiro Matsuda, and Atsushi Takayasu. Lattice-based revocable (hierarchical) IBE with decryption key exposure resistance. In Dongdai Lin and Kazue Sako, editors, *Public-Key Cryptography - PKC 2019*, volume 11443 of *Lecture Notes in Computer Science*, pages 441–471. Springer, 2019.

[19] Shuichi Katsumata and Shota Yamada. Non-zero inner product encryption schemes from various assumptions: LWE, DDH and DCR. In Dongdai Lin and Kazue Sako, editors, *Public-Key Cryptography - PKC 2019*, volume 11443 of *Lecture Notes in Computer Science*, pages 158–188. Springer, 2019.

[20] Kwangsu Lee. Revocable hierarchical identity-based encryption with adaptive security. Cryptology ePrint Archive, Report 2016/749, 2016. `http://eprint.iacr.org/2016/749`.

[21] Kwangsu Lee. Self-updatable encryption with short public parameters and its extensions. *Des. Codes Cryptogr.*, 79(1):121–161, 2016.

[22] Kwangsu Lee. A generic construction for revocable identity-based encryption with subset difference methods. *PLoS One*, 15(9):e0239053, 2020.

[23] Kwangsu Lee, Seung Geol Choi, Dong Hoon Lee, Jong Hwan Park, and Moti Yung. Self-updatable encryption: Time constrained access control with hidden attributes and better efficiency. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013*, volume 8269 of *Lecture Notes in Computer Science*, pages 235–254. Springer, 2013.

[24] Kwangsu Lee, Woo Kwon Koo, Dong Hoon Lee, and Jong Hwan Park. Public-key revocation and tracing schemes with subset difference methods revisited. In Miroslaw Kutylowski and Jaideep Vaidya, editors, *Computer Security - ESORICS 2014*, volume 8713 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2014.

[25] Kwangsu Lee, Dong Hoon Lee, and Jong Hwan Park. Efficient revocable identity-based encryption via subset difference methods. *Des. Codes Cryptogr.*, 85(1):39–76, 2017.

[26] Kwangsu Lee, Dong Hoon Lee, Jong Hwan Park, and Moti Yung. CCA security for self-updatable encryption: Protecting cloud data when clients read/write ciphertexts. *Comput. J.*, 62(4):545–562, 2019.

[27] Kwangsu Lee and Jong Hwan Park. Identity-based revocation from subset difference methods under simple assumptions. *IEEE Access*, 7:60333–60347, 2019.

[28] Kwangsu Lee and Seunghwan Park. Revocable hierarchical identity-based encryption with shorter private keys and update keys. *Des. Codes Cryptogr.*, 86(10):2407–2440, 2018.

[29] Allison B. Lewko, Amit Sahai, and Brent Waters. Revocation systems with very small private keys. In *IEEE Symposium on Security and Privacy*, pages 273–285. IEEE Computer Society, 2010.

[30] Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Daniele Micciancio, editor, *Theory of Cryptography - TCC 2010*, volume 5978 of *Lecture Notes in Computer Science*, pages 455–479. Springer, 2010.

[31] Benoît Libert and Damien Vergnaud. Adaptive-id secure revocable identity-based encryption. In Marc Fischlin, editor, *Topics in Cryptology - CT-RSA 2009*, volume 5473 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2009.

[32] Xuecheng Ma and Dongdai Lin. Generic constructions of revocable identity-based encryption. In Zhe Liu and Moti Yung, editors, *Information Security and Cryptology - Inscrypt 2019*, volume 12020 of *Lecture Notes in Computer Science*, pages 381–396. Springer, 2019.

[33] Silvio Micali. Efficient certificate revocation. Technical Report Technical Report TM-542b, 1996.

[34] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In Joe Kilian, editor, *Advances in Cryptology - CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 41–62. Springer, 2001.

[35] Moni Naor and Kobbi Nissim. Certificate revocation and certificate update. *IEEE J. Sel. Areas Commun.*, 18(4):561–570, 2000.

[36] Seunghwan Park, Kwangsu Lee, and Dong Hoon Lee. New constructions of revocable identity-based encryption from multilinear maps. *IEEE Trans. Inf. Forensic Secur.*, 10(8):1564–1577, 2015.

[37] Amit Sahai, Hakan Seyalioglu, and Brent Waters. Dynamic credentials and ciphertext delegation for attribute-based encryption. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology - CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 199–217. Springer, 2012.

[38] Jae Hong Seo and Keita Emura. Efficient delegation of key generation and revocation functionalities in identity-based encryption. In Ed Dawson, editor, *Topics in Cryptology - CT-RSA 2013*, volume 7779 of *Lecture Notes in Computer Science*, pages 343–358. Springer, 2013.

[39] Jae Hong Seo and Keita Emura. Revocable identity-based encryption revisited: Security model and construction. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *Public-Key Cryptography - PKC 2013*, volume 7778 of *Lecture Notes in Computer Science*, pages 216–234. Springer, 2013.

[40] Jae Hong Seo and Keita Emura. Revocable hierarchical identity-based encryption: History-free update, security against insiders, and short ciphertexts. In Kaisa Nyberg, editor, *Topics in Cryptology - CT-RSA 2015*, volume 9048 of *Lecture Notes in Computer Science*, pages 106–123. Springer, 2015.

[41] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology - CRYPTO '84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer, 1984.

[42] Atsushi Takayasu. Personal communication, 2021.

[43] Atsushi Takayasu and Yohei Watanabe. Lattice-based revocable identity-based encryption with bounded decryption key exposure resistance. In Josef Pieprzyk and Suriadi Suriadi, editors, *Information Security and Privacy - ACISP 2017*, volume 10342 of *Lecture Notes in Computer Science*, pages 184–204. Springer, 2017.

[44] Yohei Watanabe, Keita Emura, and Jae Hong Seo. New revocable IBE in prime-order groups: Adaptively secure, decryption key exposure resistant, and with short public parameters. In Helena Handschuh, editor, *Topics in Cryptology - CT-RSA 2017*, volume 10159 of *Lecture Notes in Computer Science*, pages 432–449. Springer, 2017.

[45] Kotoko Yamada, Nuttapong Attrapadung, Keita Emura, Goichiro Hanaoka, and Keisuke Tanaka. Generic constructions for fully secure revocable attribute-based encryption. In Simon N. Foley, Dieter Gollmann, and Einar Snekkenes, editors, *Computer Security - ESORICS 2017*, volume 10493 of *Lecture Notes in Computer Science*, pages 532–551. Springer, 2017.

A A Single Revocation Encryption Scheme

In this section, we describe the SRE scheme in the prime-order bilinear groups proposed by Lee and Park [27]. They constructed an efficient SRE scheme by combining IBE and IBR schemes.

SRE.Setup(1^λ): It first generates a bilinear group $\mathbb{G}$ of prime order p of bit size $\Theta(\lambda)$. Let g be a generator of $\mathbb{G}$. It chooses a random exponent $\alpha \in \mathbb{Z}_p$ and random elements $u, h, w, v \in \mathbb{G}$. It also chooses a random hash function H from $\mathcal{H}$. It outputs a master key $MK = \alpha$ and public parameters as

$$PP = \Big((p, \mathbb{G}, \mathbb{G}_T, e),\ g,\ u, h,\ w, v,\ H,\ \Omega = e(g,g)^\alpha \Big).$$

SRE.GenKey($(GL, ML), MK, PP$): It selects random exponents $r_1, r_2 \in \mathbb{Z}_p$ and outputs a private key by implicitly including (GL, ML) as

$$SK_{(GL,ML)} = \Big(K_0 = g^\alpha (u^{GL} h)^{r_1} w^{r_2},\ K_1 = (w^{ML} v)^{r_2},\ K_2 = g^{-r_1},\ K_3 = g^{-r_2} \Big).$$

SRE.Encrypt($(GL, ML), M, PP$): Let $M \in \{0,1\}^m$ be a message. It chooses a random exponent $t \in \mathbb{Z}_p$ and outputs a ciphertext by implicitly including (GL, ML) as

$$CT_{(GL,ML)} = \Big(C = H(\Omega^t) \oplus M,\ C_0 = g^t,\ C_1 = (u^{GL} h)^t,\ C_2 = (w^{ML} v)^t \Big).$$

SRE.Decrypt($CT_{(GL,ML)}, SK_{(GL',ML')}, PP$): If $(GL = GL') \wedge (ML \neq ML')$, then it outputs a message as

$$M = C \oplus H\big(e(C_0, K_0) \cdot e(C_1, K_2) \cdot \big(e(C_0, K_1) \cdot e(C_2, K_3) \big)^{-1/(ML' - ML)} \big).$$

Otherwise, it outputs $\perp$.

Theorem A.1 ([27]). *The SRE scheme is selectively secure under chosen plaintext attacks if the DBDH assumption holds.*