Server-Aided Revocable Identity-Based Encryption Revisited

Leixiao Cheng[1] and Fei Meng[2] *

[1] School of Mathematical Sciences, Fudan University, Shanghai 200433, China
14110180004@fudan.edu.cn
[2] School of Mathematics, Shandong University, Jinan Shandong 250100, China
201720214@mail.sdu.edu.cn

Abstract. Efficient user revocation has always been a challenging problem in identity-based encryption (IBE). Boldyreva et al. (CCS 2008) first proposed and formalized the notion of revocable IBE (RIBE) based on a tree-based revocation method. In their scheme, each user is required to store a number of long-term secret keys and all non-revoked users have to communicate with the key generation center periodically to update its decryption key. To reduce the workload on the user side, Qin et al. (ESORICS 2015) proposed a new system model, server-aided revocable IBE (SR-IBE). In SR-IBE model, each user is required to keep only one private key $\mathsf{Priv}_{\mathsf{ID}}$ and unnecessary to communicate with the key generation center or the server during key updating. However, in their security model, the challenge identity ID^* must be revoked once the private key $\mathsf{Priv}_{\mathsf{ID}^*}$ was revealed to the adversary. This is too restrictive since decrypting a ciphertext requires both the private key $\mathsf{Priv}_{\mathsf{ID}}$ and the long-term transformation key $\mathsf{sk}_{\mathsf{ID}}$.
In this paper, we first revisit Qin et al.'s security model and propose a stronger one called SSR-sID-CPA security. Specifically, ID^* is revoked only when both $\mathsf{sk}_{\mathsf{ID}^*}$ and $\mathsf{Priv}_{\mathsf{ID}^*}$ are revealed and the adversary is allowed to access short-term transformation keys oracle. We also prove that Qin et al.'s scheme is insecure under our new security model. Second, we construct a lattice-based SR-IBE scheme based on Katsumata's RIBE scheme (PKC 19), and show that our lattice-based SR-IBE scheme is SSR-sID-CPA secure. Finally, we propose a generic construction of SR-IBE scheme by combining a RIBE and a 2-level HIBE scheme. The security of the generic SR-IBE scheme inherits those of the underlying building blocks.

Keywords: IBE · Revocation · Server-aided.

1 Introduction

The concept of identity-based encryption was proposed by Shamir [28]. One practical issue of IBE in real applications is to find an effective revocation method

* Corresponding author

to revoke users in multi-user cryptosystems, because users may behave inappropriately or their secret keys may be compromised. In 2001, Boneh et al. [6] proposed a naive identity revocation mechanism, in which the up-to-date revocation list is controlled by a trusted authority called Key Generation Center (KGC), who issues secret key $sk_{\mathsf{id}||\mathsf{t}}$ for each non-revoked user id in every time period t. Only non-revoked users can decrypt ciphertext bound to their identity and the same time slot (i.e., $\mathsf{id}||\mathsf{t}$). However, this approach is inefficient since the KGC has to generate $O(N-r)$ new secret keys in each time period, where N is the total number of users and r is the number of revoked users in time period t. The workload of the KGC is proportional to the number of users N.

In 2008, Boldyreva et al. [4] proposed another revocation mechanism based on the tree-based revocation scheme of [19], and formalized the notion of revocable IBE (RIBE). In this mechanism, each user keeps $O(\log N)$ long-term secret keys and the KGC broadcasts $O(r\log(N/r))$ update keys for each time period t. Only non-revoked users can obtain their decryption keys from their long-term secret keys and the update keys. Compared with Boneh et al. [6], Boldyreva et al.'s revocation mechanism significantly reduces the total size of update keys from linear (i.e., $O(N-r)$) to logarithmic (i.e., $O(r\log(N/r))$) in the number of users. However, this solution still does not provide an efficient revocation for the following reasons: (1) the KGC has to stay online regularly and all non-revoked users need to communicate with the KGC and update their decryption keys periodically; (2) the sizes of update keys (i.e., $O(r\log(N/r))$) and users' secret keys $O(\log N)$ are logarithmical in the number of users.

Boldyreva et al. [4] also defined the selective-revocable-ID security for RIBE scheme, a security model that captures the standard notion of selective-identity security and formalizes the possible threats as much as possible. Seo-Emura [26] revisited the Boldyreva et al. security model by considering a realistic threat which they called *decryption key exposure* (DKE), and proved that the scheme of Boldyreva et al. [4] is vulnerable against DKE. Since then, there have been many follow-up works concerning RIBE scheme with DKE resistance (DKER) [12, 16, 23, 29, 30].

To reduce the workload on the user side in Boldyreva et al. [4] and to resist against DKE, Qin et al. [23] proposed a novel system model what they call server-aided revocable IBE (SR-IBE) (depicted in Fig. 1) along with a SR-IBE scheme with DKER under the decisional bilinear diffie-hellman (DBDH) assumption. In SR-IBE system, the server is assumed to be *untrusted* in the sense that it doesn't keep any secret data and only performs public storage and computation operations according to the system specification. The SR-IBE system in [23] requires no communication between users and the KGC during key updating. In addition, although the size of update keys from the KGC to the server is still logarithmic (i.e. $O(r\log(N/r))$), the size of the private key stored by each user is reduced from $O(\log N)$ to a constant (i.e. $O(1)$). Moreover, Qin et al. defined semantic security against adaptive-identity chosen plaintext security (SR-aID-CPA security) for SR-IBE, which captures both adaptive-ID attacks and DKE attacks. Also, they proved that their SR-IBE scheme is SR-aID-CPA secure

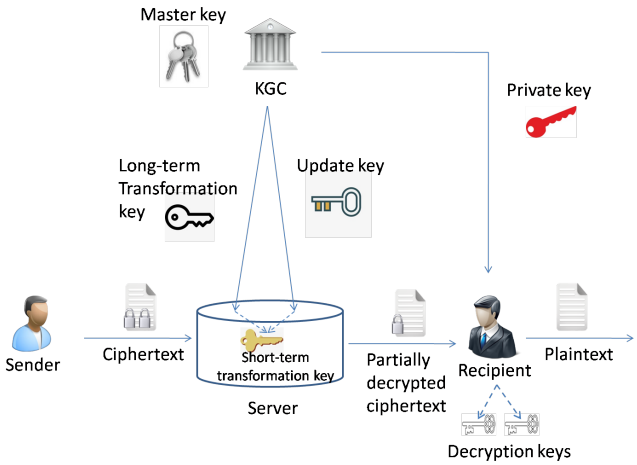

Fig. 1. System model of SR-IBE.

under the DBDH assumption. Following [23], Nguyen et al. [20] proposed the first lattice-based SR-IBE that satisfies selective-identity security (SR-sID-CPA security). One application of SR-IBE is encrypted email supporting lightweight devices in which an email server plays the role of the untrusted server so that only non-revoked users can read their email messages.

As mentioned above, the SR-IBE security model of [20, 23] has considered the DKE attack against users' local decryption keys. But the DKE attack proposed in [26] is actually defined for user's short-term transformation key. This is because the decryption key generated by a data user in a normal RIBE scheme is now created by the server and renamed as short-term transformation key in their SR-IBE schemes. From now on, in the SR-IBE model, we'll let DKE specifically refer to the local decryption key exposure, and use "STKE" to indicate the short-term transformation key exposure. Since the untrusted server could be operated by anyone, including the adversary, all short-term transformation keys could be exposed to the adversary.

1.1 Motivations and Contributions

Motivation. In the security model of SR-IBE [20, 23], the challenge identity ID^* must be revoked before the challenge time period t^* once the private key $\mathsf{Priv}_{\mathsf{ID}^*}$ was revealed to the adversary. But if we think it more carefully, we will find that this limitation is too strict for the following reason: as shown in Fig. 1, a ciphertext issued by the sender is first decrypted by the server into a partially decrypted ciphertext using a short-term transformation key obtained by combining the long-term transformation key $\mathsf{sk}_{\mathsf{ID}}$ and the update key, and then decrypted by the recipient into the plaintext using a decryption key delegated by the private key $\mathsf{Priv}_{\mathsf{ID}}$. Therefore, it may be only when both the long-term

transformation key for the first decryption and the private key for the second decryption are revealed that we need to consider revocation of the challenge identity ID^*. Noting that SR-IBE was introduced by envisioning the real-world use of IBE systems, their security definitions should be as close to the practical scenarios as possible. A more natural and weaker limitation may be: if *both* the long-term transformation key $\mathsf{sk}_{\mathsf{ID}^*}$ and the private key $\mathsf{Priv}_{\mathsf{ID}^*}$ for the challenge identity ID^* are revealed to the adversary, ID^* will be revoked before the challenge period time t^*.

Now, under the weaker limitation mentioned above, despite the exposure of the private key $\mathsf{Priv}_{\mathsf{ID}^*}$, as long as the long-term transformation key $\mathsf{sk}_{\mathsf{ID}^*}$ is not revealed to the adversary, the challenge identity ID^* does not need to be revoked before t^*. In this case (where ID^* is not revoked before t^*), the previous SR-IBE schemes (i.e. [20, 23]) are vulnerable to "STKE" attack for the same reason as mentioned in [26]: if the short-term transformation key $\mathsf{tk}_{\mathsf{ID}^*,\mathsf{t}}$ is exposed, the short-term transformation key $\mathsf{tk}_{\mathsf{ID}^*,\mathsf{t}^*}$ can be obtained by the adversary through a simple combination of $\mathsf{tk}_{\mathsf{ID}^*,\mathsf{t}}$, uk_{t} and $\mathsf{uk}_{\mathsf{t}^*}$. Then, the adversary obtains both $\mathsf{tk}_{\mathsf{ID}^*,\mathsf{t}}$ and $\mathsf{Priv}_{\mathsf{ID}^*}$, and can simply decrypt the challenge ciphertext encrypted under ID^* and t^*.

Contribution. Our contribution consists of three parts. First, we revisit Qin et al.'s security model [23], and enhance it by weakening the limitation on revoking the challenge identity and capturing both DKE attacks on the local decryption key and STKE attacks on the short-term transformation key. In our new security model, ID^* is revoked before the challenge period time t^* only when both the long-term transformation key $\mathsf{sk}_{\mathsf{ID}^*}$ and the private key $\mathsf{Priv}_{\mathsf{ID}^*}$ are revealed to the adversary. In addition, the adversary is allowed to issue short-term transformation key reveal queries. As mentioned above, these modifications are necessary because the previous schemes (i.e. [20,23]) are vulnerable under our new security model. In Section 3.2, we describe the formal definitions of our security models, SSR-sID-CPA security and SSR-aID-CPA security.

Recall that the lattice-based SR-IBE scheme in [20] is constructed by applying the double encryption technique to combine a RIBE without DKER scheme [8] and a two-level HIBE scheme [1]. The use of the double encryption technique makes it necessary to sample a series of gaussian matrices during the generations of both long-term transformation keys and update keys. In the end, they transformed the recipient's workload from decryption of a ciphertext obtained by encrypting the message M with RIBE [8] to decryption of a ciphertext obtained by encrypting the message M with HIBE [1]. However, It is difficult to measure whether or how much the workload on the recipient side has been reduced. In this paper, we construct a lattice-based SR-IBE scheme by combining a RIBE with DKER scheme [12] and a two-level HIBE scheme [1]. Compared with [20], we combine the two building blocks without using the double encryption technique, which reduces the amount of sampling required for the generations of the long-term transformation keys and update keys. In addition, we reduce the recipient's workload in a strict sense by transforming the recipien-

t's workload from decrypting the ciphertext of message M encrypted by RIBE with DKER [12] to decrypting the ciphertext of M encrypted by HIBE [1]. This is because the amount of workload required to decrypt a ciphertext of RIBE with DKER [12] is strictly greater than that required to decrypt a ciphertext of HIBE [1]. Moreover, we prove that our lattice-based SR-IBE scheme is secure under our new security model.

Third, we propose a generic construction of SR-IBE scheme by employing a RIBE with DKER scheme and a 2-level HIBE scheme, and prove that the security of SR-IBE scheme inherits those of the underlying building blocks. In other words, the SR-IBE is SSR-sID-CPA (*resp.* SSR-aID-CPA) secure if the underlying RIBE is selective-identity (*resp.* adaptive-identity) secure and the underlying 2-level HIBE is selective-identity (*resp.* adaptive-identity) secure.

1.2 Related Work

Boldyreva et al. [4] proposed the first RIBE scheme from bilinear map. Chen et al. [8] proposed the first lattice-base RIBE scheme. Seo and Emura [25] introduce the security notion of DKER along with a RIBE with DKER from bilinear map. Since then, several improvements and variants have been proposed [11, 13, 17, 22, 31]. Recently, Ma and Lin [16] proposed a construction of RIBE without DKER from IBE, and a generic construction of RIBE with DKER from IBE and 2-level HIBE. The first SR-IBE scheme was proposed by Qin et al. [23]. Nguyen et al. [20] proposed the first SR-IBE from lattice.

1.3 Roadmap

The rest of this paper is organized as follows. Section 2 describes some necessary preliminaries. Section 3 introduces our new security model of SR-IBE. Section 4 presents our constructions of lattice-based SR-IBE. Section 5 presents a generic construction of SR-IBE. In the end, We give a conclusion in Section 6.

2 Preliminaries

2.1 Notations

For a binary string α, let $|\alpha|$ denote its binary length. For a positive integer n, Let $[n]$ denote the set $\{1, \cdots, n\}$. If S is a finite set then $x \leftarrow S$ is the operation of choosing an element uniformly at random from S. For a probability distribution $\mathcal{D}$, $x \leftarrow \mathcal{D}$ denotes the operation of choosing an element according to $\mathcal{D}$. If γ is either an algorithm nor a set then $x \leftarrow \gamma$ is a simple assignment statement.

Let λ denote the security parameter. A function $f(\lambda)$ is *negligible*, denoted as $\mathsf{negl}(\lambda)$, if for every $c > 0$, there exists an λ_c such that $f(\lambda) < 1/\lambda^c$ for all $\lambda > \lambda_c$. An algorithm is probabilistic polynomial-time (PPT) computable if it is modeled as a probabilistic Turing machine whose running time is bounded

by some polynomial function $\mathsf{poly}(\lambda)$. Two distribution ensembles $\{X_\lambda\}_{\lambda\in\mathbb{N}}$ and $\{Y_\lambda\}_{\lambda\in\mathbb{N}}$ are *computationally indistinguishable*, if for any PPT algorithm D, and for sufficiently large λ, we have $|\Pr[D(X_\lambda)=1]-\Pr[D(Y_\lambda)=1]|$ is negligible in λ.

For notational convenience, we sometimes view a matrix as simply the set of its column vectors. For a matrix $\mathbf{T}=[\mathbf{t}_1\cdots\mathbf{t}_k]\in\mathbb{R}^{m\times k}$, let $\|\mathbf{T}\|$ denote the L_2 length of its longest column, i.e., $\|\mathbf{T}\|:=\max_i\|\mathbf{t}_i\|$ for $1\le i\le k$; let $s_1(\mathbf{T})$ denote the largest singular value of $\mathbf{T}$, i.e., $s_1(\mathbf{T}):=\sup_{\mathbf{u}\in\mathbb{R}^k,\|\mathbf{u}\|=1}\|\mathbf{Tu}\|$. Further, if $\mathbf{t}_1,\cdots,\mathbf{t}_k$ in $\mathbf{T}$ are linearly independent, let $\widetilde{\mathbf{T}}:=[\tilde{\mathbf{t}}_1,\cdots,\tilde{\mathbf{t}}_k]$, where $\tilde{\mathbf{t}}_1,\cdots,\tilde{\mathbf{t}}_k$ denote the Gram-Schmidt orthogonalization of $\mathbf{t}_1,\cdots,\mathbf{t}_k$. For two matrices $\mathbf{X}\in\mathbb{R}^{n\times m_1}$ and $\mathbf{Y}\in\mathbb{R}^{n\times m_2}$, let $[\mathbf{X}\mid\mathbf{Y}]\in\mathbb{R}^{n\times(m_1+m_2)}$ denote the concatenation of the columns of $\mathbf{X}$ and $\mathbf{Y}$. For two matrices $\mathbf{X}\in\mathbb{R}^{n\times m}$ and $\mathbf{Y}\in\mathbb{R}^{m\times k}$, we have $\|\mathbf{X}\|,\|\mathbf{X}^\top\|\le s_1(\mathbf{X})$, and $s_1(\mathbf{XY})\le s_1(\mathbf{X})\cdot s_1(\mathbf{Y})$.

2.2 The Binary Tree Data Structure and the CS method

The complete subtree (CS) method was introduced by Naor et al. [19]. A binary tree along with the CS method is an efficient revocation mechanism, which has been widely used in systems [5, 12, 21, 25]. To introduce this mechanism, we use the following notations: BT denotes a binary-tree. root denotes the root node of BT. θ denotes a node in the binary tree and η emphasizes that the node θ is a leaf node. The set $\mathsf{Path}(\mathsf{BT},\eta_{\mathsf{ID}})$ stands for the collection of nodes on the path from the leaf η_{ID} to the root (including η_{ID} and the root). If θ is a non-leaf node, then θ_ℓ, θ_r denote the left and right child of θ, respectively.

Before introducing the CS method, we recall the node selection algorithm $\mathsf{KUNodes}$ as in previous RIBE systems [5, 25]. The $\mathsf{KUNodes}$ algorithm takes as input a binary tree BT, a revocation list RL, and outputs a set of nodes Y, such that the subtrees with root $\theta\in\mathsf{Y}$ cover all leaves η_{ID} in BT for $\mathsf{ID}\notin\mathsf{RL}$ and do not cover any leaves η_{ID} for $\mathsf{ID}\in\mathsf{RL}$. The description of the $\mathsf{KUNodes}$ algorithm is as follows:

$\mathsf{KUNodes}(\mathsf{BT},\mathsf{RL})$:
$X,Y\leftarrow\emptyset$; $\forall\mathsf{ID}\in\mathsf{RL}$,add $\mathsf{Path}(\mathsf{BT},\eta_{\mathsf{ID}})$ to X;
$\forall\theta\in X$, if $\theta_\ell\notin X$ then add θ_ℓ to Y, if $\theta_r\notin X$ then add θ_r to Y;
If $Y=\emptyset$ then add root to Y; Return Y;

We adopt the definition of the CS method given by Katsumata et al. [12], which consists of the following four algorithms:

$\mathsf{CS.Setup}(N)\to\mathsf{BT}$**:** on input the number of users N, it outputs a binary tree BT with at least N and at most $2N$ leaves.

$\mathsf{CS.Assign}(\mathsf{BT},\mathsf{ID})\to(\eta_{\mathsf{ID}},\mathsf{BT})$**:** on input a binary tree BT and an identity ID, it randomly assigns the identity ID to a leaf node η_{ID}, to which no other identities have been assigned yet. Then, it outputs a leaf node η_{ID} and an "updated" binary tree BT.

$\mathsf{CS.Cover}(\mathsf{BT},\mathsf{RL})\to\mathsf{KUNodes}(\mathsf{BT},\mathsf{RL})$**:** on input a binary tree and a revocation list RL, it outputs a set of nodes $\mathsf{KUNodes}(\mathsf{BT},\mathsf{RL})$.

$\mathsf{CS.Match}(\mathsf{Path}(\mathsf{BT}, \eta_{\mathsf{ID}}), \mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL})) \rightarrow \theta/\emptyset$**:** on input $\mathsf{Path}(\mathsf{BT}, \eta_{\mathsf{ID}})$ and $\mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL})$, it outputs an arbitrary node $\theta \in \mathsf{Path}(\mathsf{BT}, \eta_{\mathsf{ID}}) \cap \mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL})$ if it exists. Otherwise, it outputs $\emptyset$.

2.3 Background on Lattice

Let $\mathbf{B} = \{\mathbf{b}_1 \cdots \mathbf{b}_m\} \subset \mathbb{Z}^m$ consist of m linearly independent vectors. The (full-rank-integer) m-dimensional lattice Λ generated by the *basis* $\mathbf{B}$ is $\Lambda = \mathcal{L}(\mathbf{B}) := \{\sum_{i\in[m]} x_i\mathbf{b}_i \mid x_i \in \mathbb{Z}\}$. For any positive integers n, m and $q \geq 2$, a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a vector $\mathbf{u} \in \mathbb{Z}_q^n$, we define $\Lambda_q^{\perp}(\mathbf{A}) := \{\mathbf{z} \in \mathbb{Z}^m : \mathbf{A}\cdot\mathbf{z} = \mathbf{0}_n \mod q\}$ and $\Lambda_q^{\mathbf{u}}(\mathbf{A}) := \{\mathbf{z} \in \mathbb{Z}^m : \mathbf{A}\cdot\mathbf{z} = \mathbf{u} \mod q\}$.

Discrete Gaussian over Lattice. Let Λ be a lattice in $\mathbb{Z}^m$. For any vector $\mathbf{c} \in \mathbb{R}^m$ and any parameter $\sigma \in \mathbb{R}_+$, define $\rho_{\sigma,\mathbf{c}}(\mathbf{x}) = \exp(-\pi\frac{\|\mathbf{x}-\mathbf{c}\|^2}{\sigma^2})$ and $\rho_{\sigma,\mathbf{c}}(\Lambda) = \sum_{\mathbf{x}\in\Lambda}\rho_{\sigma,\mathbf{c}}(\mathbf{x})$. The *discrete Gaussian distribution* over Λ with center $\mathbf{c}$ and Gaussian parameter σ is $\mathcal{D}_{\Lambda,\sigma,\mathbf{c}} = \frac{\rho_{\sigma,\mathbf{c}}(\mathbf{y})}{\rho_{\sigma,\mathbf{c}}(\Lambda)}$ for $\forall\mathbf{y} \in \Lambda$. If $\mathbf{c} = \mathbf{0}$, we conveniently use ρ_σ and $\mathcal{D}_{\Lambda,\sigma}$.

Lemma 1 ([10]). *Let Λ be an m-dimensional lattice. Let $\mathbf{T}$ be a basis for Λ, and suppose $\sigma \geq \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log m})$. Then $\Pr[\|\mathbf{x}\| > \sigma\sqrt{m} : \mathbf{x} \leftarrow \mathcal{D}_{\Lambda,\sigma}] \leq \mathsf{negl}(m)$.*

Lemma 2 ([10]). *Let $n, m, q > 0$ be positive integers with $m \geq 2n\lceil\log q\rceil$ and q a prime. Let σ be any positive real such that $\sigma \geq \omega(\sqrt{\log m})$. Then for $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$ and $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\sigma}$, the distribution of $\mathbf{u} = \mathbf{A}\mathbf{e} \mod q$ is statistically close to uniform over $\mathbb{Z}_q^n$. Furthermore, for a fixed $\mathbf{u} \in \mathbb{Z}_q^n$, the conditional distribution of $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\sigma}$, given $\mathbf{A}\mathbf{e} = \mathbf{u} \mod q$ for a uniformly random $\mathbf{A}$ in $\mathbb{Z}_q^{n\times m}$ is $\mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{A}),\sigma}$ with all but negligible probability.*

Sampling Algorithms. Let's first review two sampling algorithms in the following lemmas. The first algorithm generates a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ that is statistically close to uniform, together with a short trapdoor basis for the associated lattice $\Lambda_q^{\perp}(\mathbf{A})$. The second algorithm generates a basis for the lattice $\Lambda_q^{\perp}(\mathbf{G})$, where $\mathbf{G}$ is what they call the primitive matrix.

Lemma 3 ([2, 3, 18]). *Let $n, m, q > 0$ be positive integers with $m \geq 2n\lceil\log q\rceil$ and q a prime. Then, we have:*

- *[2, 3, 18] a PPT algorithm $\mathsf{TrapGen}(n, m, q)$ that outputs a pair $(\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}^{m\times m}$ such that $\mathbf{A}$ is statistically close to uniform and $\mathbf{T}_{\mathbf{A}}$ is a basis for $\Lambda_q^{\perp}(\mathbf{A})$ satisfying $\|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \leq O(\sqrt{n\log q})$.*
- *[18] a fixed full rank matrix $\mathbf{G} \in \mathbb{Z}_q^{n\times m}$ such that the lattice $\Lambda_q^{\perp}(\mathbf{G})$ has a publicly known basis $\mathbf{T}_{\mathbf{G}} \in \mathbb{Z}^{m\times m}$ with $\|\widetilde{\mathbf{T}_{\mathbf{G}}}\| \leq \sqrt{5}$.*

We review some of the algorithms that allow one to securely delegate a trapdoor of a lattice to an arbitrary higher-dimensional extension. They can be obtained by combining corresponding results in [1] and [7].

Lemma 4 ([1,7]). *Let $n, m, \bar{m}, q > 0$ be positive integers with $m > n$ and q a prime. Then, there exist PPT algorithms as follows:*

$\mathsf{ExtRndLeft}(\mathbf{A}, \mathbf{F}, \mathbf{T}_{\mathbf{A}}, \sigma) \to \mathbf{T}_{[\mathbf{A}|\mathbf{F}]}$**:** *On input matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{F} \in \mathbb{Z}_q^{n\times \bar{m}}$, a basis $\mathbf{T}_{\mathbf{A}}$ of $\Lambda_q^{\perp}(\mathbf{A})$, and a Gaussian parameter $\sigma \geq \|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \cdot \omega(\sqrt{\log n})$, outputs a matrix $\mathbf{T}_{[\mathbf{A}|\mathbf{F}]} \in \mathbb{Z}^{(m+\bar{m})\times(m+\bar{m})}$ distributed statistically close to $(\mathcal{D}_{\Lambda_q^{\perp}([\mathbf{A}|\mathbf{F}]),\sigma})^{m+\bar{m}}$.*

$\mathsf{ExtRndRight}(\mathbf{A}, \mathbf{G}, \mathbf{R}, \mathbf{T}_{\mathbf{G}}, \sigma) \to \mathbf{T}_{[\mathbf{A}|\mathbf{AR}+\mathbf{G}]}$**:** *On input full rank matrices $\mathbf{A}, \mathbf{G} \in \mathbb{Z}_q^{n\times m}$, a matrix $\mathbf{R} \in \mathbb{Z}_q^{m\times m}$, a basis $\mathbf{T}_{\mathbf{G}}$ of $\Lambda_q^{\perp}(\mathbf{G})$, and a Gaussian parameter $\sigma \geq s_1(\mathbf{R}) \cdot \|\widetilde{\mathbf{T}_{\mathbf{G}}}\| \cdot \omega(\sqrt{\log m})$, outputs a matrix $\mathbf{T}_{[\mathbf{A}|\mathbf{AR}+\mathbf{G}]} \in \mathbb{Z}^{2m\times 2m}$ distributed statistically close to $(\mathcal{D}_{\Lambda_q^{\perp}([\mathbf{A}|\mathbf{AR}+\mathbf{G}]),\sigma})^{2m}$.*

The following algorithms allow one to sample short vectors from a given lattice equipped with a short basis.

Lemma 5 ([1,10]). *Let $n, m, \bar{m}, q > 0$ be positive integers with $m \geq 2n\lceil \log q \rceil$ and q a prime. Then, we have the following algorithms:*

$\mathsf{SamplePre}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{u}, \sigma) \to \mathbf{e}$ **[10]:** *on input a full rank matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a basis $\mathbf{T}_{\mathbf{A}} \in \mathbb{Z}^{m\times m}$ of $\Lambda_q^{\perp}(\mathbf{A})$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, and a Gaussian parameter $\sigma \geq \|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \cdot \omega(\sqrt{\log m})$, it outputs a vector $\mathbf{e} \in \mathbb{Z}^m$ sampled from a distribution statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{A}),\sigma}$.*

$\mathsf{SampleLeft}(\mathbf{A}, \mathbf{F}, \mathbf{u}, \mathbf{T}_{\mathbf{A}}, \sigma) \to \mathbf{e}$ **[1]:** *On input a full rank matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a matrix $\mathbf{F} \in \mathbb{Z}_q^{n\times \bar{m}}$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, a basis $\mathbf{T}_{\mathbf{A}} \in \mathbb{Z}^{m\times m}$ of $\Lambda_q^{\perp}(\mathbf{A})$, and a Gaussian parameter $\sigma \geq \|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \cdot \omega(\sqrt{\log(m+\bar{m})})$, it outputs a vector $\mathbf{e} \in \mathbb{Z}^{m+\bar{m}}$ sampled from a distribution statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{u}}([\mathbf{A}|\mathbf{F}]),\sigma}$.*

Hardness Assumption. The learning with errors (LWE) assumption was introduced by Regev [24]. For integers n, m, a prime q, a real $\alpha \in (0,1)$ such that $\alpha q > 2\sqrt{n}$, and a PPT algorithm $\mathcal{A}$, the advantage for learning with errors problem $\mathsf{LWE}_{n,m,q,\mathcal{D}_{\mathbb{Z}^m,\alpha q}}$ of $\mathcal{A}$ is defined as $\big|\Pr[\mathcal{A}(\mathbf{A}, \mathbf{A}^{\top}\mathbf{s}+\mathbf{x}) = 1] - \Pr[\mathcal{A}(\mathbf{A}, \mathbf{v}) = 1]\big|$, where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{x} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\alpha q}$, $\mathbf{v} \leftarrow \mathbb{Z}_q^m$. We say that the LWE assumption holds if the above advantage is negligible for all PPT adversary $\mathcal{A}$.

2.4 Facts

Lemma 6 ([1,15]). *Let $\mathbf{R}$ be a $m\times k$ matrix chosen at random from $\{-1,1\}^{m\times k}$, then there exists a universal constant C such that $\Pr[s_1(\mathbf{R}) > C\sqrt{m+k}] < e^{-(m+k)}$.*

Lemma 7 ([1,9]). *Suppose that $m > (n+1)\log q + \omega(\log n)$ and that q is a prime. Let $\mathbf{A}, \mathbf{B}$ be matrices chosen uniformly in $\mathbb{Z}_q^{n\times m}$ and let $\mathbf{R}$ be an $m \times m$ matrix chosen uniformly in $\{-1,1\}^{m\times m} \mod q$. Then, for all vectors $\mathbf{w}$ in $\mathbb{Z}_q^m$, the distribution of $(\mathbf{A}, \mathbf{AR}, \mathbf{R}^{\top}\mathbf{w})$ is statistically close to the distribution of $(\mathbf{A}, \mathbf{B}, \mathbf{R}^{\top}\mathbf{w})$.*

Definition 1 (FRD [1]). *Let q be a prime and n a positive integer. We say that a function $\mathsf{H} : \mathbb{Z}_q^n \to \mathbb{Z}_q^{n\times n}$ is a full-rank difference (FRD) map if: for all distinct $\mathsf{ID}, \mathsf{ID}' \in \mathbb{Z}_q^n$, the matrix $\mathsf{H}(\mathsf{ID}) - \mathsf{H}(\mathsf{ID}') \in \mathbb{Z}_q^{n\times n}$ is full rank, and H is computable in polynomial time in $n \log q$.*

2.5 Revocable Identity-based Encryption

Recall that in PKC 2019, Katsumata et al. [12] re-specified the syntax and security definition for RIBE to address the ambiguity of previous security definitions [4,25–27] (e.g. in some cases it is not clear when the values such as secret keys and update keys are generated during the security game). Specifically, they mainly made the following four adjustments:

- An entity capable of deriving a secret key for a low-level user is modeled as a stateful entity, and is supposed to maintain a so-called "state" in addition to its own secret key. The roles of the "state" information (used by the revocation mechanism) and the secret key are merged in the syntax.
- They do not explicitly include the "revoke" algorithm, which adds a user to be revoked into the revocation list, as part of the syntax. Instead, a revocation list is treated as a subset of identity space and is considered together with the update key information.
- They explicitly separate the secret key generation and secret key reveal queries to capture cases when some secret key $\mathsf{sk}_{\mathsf{ID}}$ has been generated but not revealed to an adversary.
- They combine the "revoke" and "update key" queries into the single "revoke & update key" query and introduce a global counter t_{cu} representing the "current time period" which is coordinated with the adversary's revoke & update key query.

In this section, we introduce the more rigorous definition of RIBE and its security model given in [12]. Note that the default security definition captures the DKER security considered in [25]. For convenience, we sometimes refer to a RIBE scheme that satisfies the default security definition as RIBE with DKER.

Syntax. A RIBE scheme $\Pi = (\mathsf{Setup}, \mathsf{GenSK}, \mathsf{KeyUP}, \mathsf{GenDK}, \mathsf{Encrypt}, \mathsf{Decrypt})$ is described as follows:

$\mathsf{Setup}(1^\lambda) \to (\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}})$**:** On input the security parameter 1^λ, it outputs a public parameter pp and the KGC's secret key $\mathsf{sk}_{\mathsf{kgc}}$ (also called a master secret key). We assume that the message space $\mathcal{M}$, the time period space $\mathcal{T}$, and the identity space $\mathcal{ID}$ are determined only by the security parameter λ, and their descriptions are contained in pp.

$\mathsf{GenSK}(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}, \mathsf{ID}) \to (\mathsf{sk}_{\mathsf{ID}}, \mathsf{sk}'_{\mathsf{kgc}})$**:** On input a public parameter pp, the KGC's secret key $\mathsf{sk}_{\mathsf{kgc}}$, and an identity $\mathsf{ID} \in \mathcal{ID}$, it outputs a secret key $\mathsf{sk}_{\mathsf{ID}}$ for the identity ID and also the KGC's "updated" secret key $\mathsf{sk}'_{\mathsf{kgc}}$.

$\mathsf{KeyUP}(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}, \mathsf{t}, \mathsf{RL}_{\mathsf{t}}) \to (\mathsf{uk}_{\mathsf{t}}, \mathsf{sk}'_{\mathsf{kgc}})$**:** On input a public parameter pp, the KGC's secret key $\mathsf{sk}_{\mathsf{kgc}}$, a time period t and a revocation list $\mathsf{RL}_{\mathsf{t}} \subset \mathcal{ID}$, it outputs an update key uk_{t} and also the "updated" secret key $\mathsf{sk}'_{\mathsf{kgc}}$.

$\mathsf{GenDK}(\mathsf{pp}, \mathsf{sk}_{\mathsf{ID}}, \mathsf{uk}_{\mathsf{t}}) \to \mathsf{dk}_{\mathsf{ID},\mathsf{t}}$ **or** $\perp$**:** On input a public parameter pp, a secret key $\mathsf{sk}_{\mathsf{ID}}$ and an update key uk_{t}, it outputs a decryption key $\mathsf{dk}_{\mathsf{ID},\mathsf{t}}$ for the time period t or the special "invalid" symbol $\perp$ indicating that ID has been revoked.

$\mathsf{Encrypt}(\mathsf{pp}, \mathsf{ID}, \mathsf{t}, M) \to \mathsf{ct}_{\mathsf{ID},\mathsf{t}}$**:** On input a public parameter pp, an identity ID, a time period t and a message M, it outputs a ciphertext $\mathsf{ct}_{\mathsf{ID},\mathsf{t}}$.

$\mathsf{Decrypt}(\mathsf{pp}, \mathsf{dk}_{\mathsf{ID},\mathsf{t}}, \mathsf{ct}_{\mathsf{ID},\mathsf{t}}) \to M$**:** On input a public parameter pp, a decryption key $\mathsf{dk}_{\mathsf{ID},\mathsf{t}}$ and a ciphertext $\mathsf{ct}_{\mathsf{ID},\mathsf{t}}$, it outputs the decryption result M.

Correctness. For all $\lambda \in \mathbb{N}$, $(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}) \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathsf{ID} \in \mathcal{ID}$, $\mathsf{t} \in \mathcal{T}$, $M \in \mathcal{M}$ and $\mathsf{RL}_{\mathsf{t}} \subset \mathcal{ID} \backslash \{\mathsf{ID}\}$, the correctness requires that $M' = M$ to hold after executing the following procedures:

(1) $(\mathsf{sk}_{\mathsf{ID}}, \mathsf{sk}_{\mathsf{kgc}}) \leftarrow \mathsf{GenSK}(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}, \mathsf{ID})$.
(2) $(\mathsf{uk}_{\mathsf{t}}, \mathsf{sk}_{\mathsf{kgc}}) \leftarrow \mathsf{KeyUP}(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}, \mathsf{t}, \mathsf{RL}_{\mathsf{t}})$.
(3) $\mathsf{dk}_{\mathsf{ID},\mathsf{t}} \leftarrow \mathsf{GenDK}(\mathsf{pp}, \mathsf{sk}_{\mathsf{ID}}, \mathsf{uk}_{\mathsf{t}})$.
(4) $\mathsf{ct}_{\mathsf{ID},\mathsf{t}} \leftarrow \mathsf{Encrypt}(\mathsf{pp}, \mathsf{ID}, \mathsf{t}, M)$.
(5) $M' \leftarrow \mathsf{Decrypt}(\mathsf{pp}, \mathsf{dk}_{\mathsf{ID},\mathsf{t}}, \mathsf{ct}_{\mathsf{ID},\mathsf{t}})$.

Note that it is allowed to execute an arbitrary polynomial number of executions of GenSK in between steps (1) and (2). However, for simplicity and readability, the correctness is defined as above.

Security. Let the global counter t_{cu} be initialized to 1. Let SKList be a list that initially contains $(\mathsf{kgc}, \mathsf{sk}_{\mathsf{kgc}})$. Whenever a new secret key is generated for an identity $\mathsf{ID} \in \mathcal{ID}$ or the secret key $\mathsf{sk}_{\mathsf{kgc}}$ is updated due to the execution of GenSK or KeyUP in the security game, the challenger $\mathcal{C}$ will store $(\mathsf{ID}, \mathsf{sk}_{\mathsf{ID}})$ or update the corresponding entry $(\mathsf{kgc}, \mathsf{sk}_{\mathsf{kgc}})$ in SKList, and we will not mention this addition/update explicitly.

Let $\mathcal{O}^{\mathsf{RIBE}}$ be the set of queries as follows:

Secret Key Generation Query: Upon a query $\mathsf{ID} \in \mathcal{ID}$ from the adversary $\mathcal{A}$, where it is required that $(\mathsf{ID}, *) \notin \mathsf{SKList}$, the challenge $\mathcal{C}$ runs $(\mathsf{sk}_{\mathsf{ID}}, \mathsf{sk}'_{\mathsf{kgc}}) \leftarrow \mathsf{GenSK}(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}, \mathsf{ID})$ and returns nothing to $\mathcal{A}$.
We require that all identities ID appearing in the following queries be "activated" in the sense that $\mathsf{sk}_{\mathsf{ID}}$ is generated via this query and hence $(\mathsf{ID}, \mathsf{sk}_{\mathsf{ID}}) \in \mathsf{SKList}$.

Secret Key Reveal Query: Upon a query $\mathsf{ID} \in \mathcal{ID}$ from $\mathcal{A}$, $\mathcal{C}$ checks if the following condition is satisfied:

- If $\mathsf{t}_{\mathsf{cu}} \geq \mathsf{t}^*$ and $\mathsf{ID}^* \notin \mathsf{RL}_{\mathsf{t}^*}$, then $\mathsf{ID} \neq \mathsf{ID}^*$.

If this condition is *not* satisfied, then $\mathcal{C}$ returns $\perp$ to $\mathcal{A}$. Otherwise, $\mathcal{C}$ finds $\mathsf{sk}_{\mathsf{ID}}$ from SKList and returns it to $\mathcal{A}$.

Revoke & Update Key Query: Upon a query $\mathsf{RL} \subset \mathcal{ID}$ (which represents the set of identities that are going to be revoked in the next time period) from $\mathcal{A}$, $\mathcal{C}$ checks if the following conditions are satisfied simultaneously:

- $\mathsf{RL}_{\mathsf{t_{cu}}} \subset \mathsf{RL}$.
- If $\mathsf{t_{cu}} = \mathsf{t}^* - 1$ and $\mathsf{sk}_{\mathsf{ID}^*}$ for the challenge ID^* has been reveled to $\mathcal{A}$ via a secret key reveal query ID^*, then $\mathsf{ID}^* \in \mathsf{RL}$.

If these conditions are *not* satisfied, then $\mathcal{C}$ returns $\perp$ to $\mathcal{A}$. Otherwise, it increments the current time period by $\mathsf{t_{cu}} \leftarrow \mathsf{t_{cu}} + 1$, sets $\mathsf{RL}_{\mathsf{t_{cu}}} \leftarrow \mathsf{RL}$, runs $(\mathsf{uk}_{\mathsf{t_{cu}}}, \mathsf{sk}'_{\mathsf{kgc}}) \leftarrow \mathsf{KeyUP}(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}, \mathsf{t_{cu}}, \mathsf{RL}_{\mathsf{t_{cu}}})$ and returns $\mathsf{uk}_{\mathsf{t_{cu}}}$ to $\mathcal{A}$.

Decryption Key Reveal Query: Upon a query $(\mathsf{ID}, \mathsf{t}) \in \mathcal{ID} \times \mathcal{T}$ from $\mathcal{A}$, $\mathcal{C}$ checks if the following conditions are simultaneously satisfied:

- $\mathsf{t} \leq \mathsf{t_{cu}}$.
- $\mathsf{ID} \notin \mathsf{RL}_{\mathsf{t}}$.
- $(\mathsf{ID}, \mathsf{t}) \neq (\mathsf{ID}^*, \mathsf{t}^*)$.

If these conditions are *not* satisfied, then $\mathcal{C}$ returns $\perp$ to $\mathcal{A}$. Otherwise, it finds $\mathsf{sk}_{\mathsf{ID}}$ from SKList, runs $\mathsf{dk}_{\mathsf{ID},\mathsf{t}} \leftarrow \mathsf{GenDK}(\mathsf{pp}, \mathsf{sk}_{\mathsf{ID}}, \mathsf{uk}_{\mathsf{t}})$ and returns $\mathsf{dk}_{\mathsf{ID},\mathsf{t}}$ to $\mathcal{A}$.

Definition 2. *A RIBE (with DKER) scheme satisfies selective-identity (*resp. *adaptive-identity) security, if any PPT adversary $\mathcal{A}$ has negligible advantage* $\mathsf{Adv}^{\mathsf{RIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda)$ *(*resp. $\mathsf{Adv}^{\mathsf{RIBE\text{-}ad}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda)$*) in experiment* $\mathsf{Exp}^{\mathsf{RIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda)$ *(*resp. $\mathsf{Exp}^{\mathsf{RIBE\text{-}ad}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda)$*):*

$\mathsf{Exp}^{\mathsf{RIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda)$:
$\mathsf{ID}^*, \mathsf{t}^* \leftarrow \mathcal{A}$;
$(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}) \leftarrow \mathsf{Setup}(1^\lambda)$; $\mathsf{SKList} \leftarrow (\mathsf{kgc}, \mathsf{sk}_{\mathsf{kgc}})$; $(\mathsf{uk}_1, \mathsf{sk}'_{\mathsf{kgc}}) \leftarrow \mathsf{KeyUP}(\mathsf{sk}_{\mathsf{kgc}}, \mathsf{t_{cu}} = 1, \mathsf{RL}_1 = \emptyset)$;
$(M_0, M_1) \leftarrow \mathcal{A}^{\mathcal{O}^{RIBE}}(\mathsf{pp})$, *where* $|M_0| = |M_1|$;
$b \leftarrow \{0,1\}$; $\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*} \leftarrow \mathsf{Encrypt}(\mathsf{pp}, \mathsf{ID}^*, \mathsf{t}^*, M_b)$;
$b' \leftarrow \mathcal{A}^{\mathcal{O}^{RIBE}}(\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*})$;
Return 1 if $b' = b$ *and 0 otherwise.*

$\mathsf{Exp}^{\mathsf{RIBE\text{-}ad}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda)$:
$(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}) \leftarrow \mathsf{Setup}(1^\lambda)$; $\mathsf{SKList} \leftarrow (\mathsf{kgc}, \mathsf{sk}_{\mathsf{kgc}})$; $(\mathsf{uk}_1, \mathsf{sk}'_{\mathsf{kgc}}) \leftarrow \mathsf{KeyUP}(\mathsf{sk}_{\mathsf{kgc}}, \mathsf{t_{cu}} = 1, \mathsf{RL}_1 = \emptyset)$;
$(M_0, M_1, \mathsf{ID}^*, \mathsf{t}^*) \leftarrow \mathcal{A}^{\mathcal{O}^{RIBE}}(\mathsf{pp})$, *where it is required that the following conditions are satisfied simultaneously:*
- $|M_0| = |M_1|$,
- *if* $\mathsf{t}^* \leq \mathsf{t_{cu}}$, *then* $\mathcal{A}$ *has not submitted* $(\mathsf{ID}^*, \mathsf{t}^*)$ *as a decryption key reveal query,*
- *if* $\mathsf{sk}_{\mathsf{ID}^*}$ *has been revealed to* $\mathcal{A}$, *then it is required that* $\mathsf{ID}^* \in \mathsf{RL}_{\mathsf{t}^*}$;

$b \leftarrow \{0,1\}$; $\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*} \leftarrow \mathsf{Encrypt}(\mathsf{pp}, \mathsf{ID}^*, \mathsf{t}^*, M_b)$;
$b' \leftarrow \mathcal{A}^{\mathcal{O}^{RIBE}}(\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*})$;
Return 1 if $b' = b$ *and 0 otherwise.*

The advantage of $\mathcal{A}$ in the experiment $\mathsf{Exp}^{\mathsf{RIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda)$ *(resp.* $\mathsf{Exp}^{\mathsf{RIBE\text{-}ad}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda)$*) is defined as:*

$$\begin{aligned} &\mathsf{Adv}^{\mathsf{RIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda) = 2 \cdot \left| \Pr\left[\mathsf{Exp}^{\mathsf{RIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda) = 1\right] - \frac{1}{2} \right|, \\ \Big(\textit{resp.} \quad &\mathsf{Adv}^{\mathsf{RIBE\text{-}ad}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda) = 2 \cdot \left| \Pr\left[\mathsf{Exp}^{\mathsf{RIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda) = 1\right] - \frac{1}{2} \right| \Big). \end{aligned} \tag{1}$$

The weak security notions (i.e. security without DKER) are defined by changing the corresponding security experiments so that an adversary $\mathcal{A}$ is not allowed to make any decryption key reveal query. The advantage of the adversary $\mathcal{A}$ in weak selective-identity (*resp.* weak adaptive-identity) security experiment is defined as $\mathsf{Adv}^{\mathsf{RIBE\text{-}sel\text{-}weak}}_{\mathcal{A},\mathcal{O}^{\mathsf{RIBE}}}(\lambda)$ (*resp.* $\mathsf{Adv}^{\mathsf{RIBE\text{-}ad\text{-}weak}}_{\mathcal{A},\mathcal{O}^{\mathsf{RIBE}}}(\lambda)$).

Definition 3. *A RIBE scheme satisfies weak selective-identity (*resp. *weak adaptive-identity) security, if any PPT adversary $\mathcal{A}$ has negligible advantage* $\mathsf{Adv}^{\mathsf{RIBE\text{-}sel\text{-}weak}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda)$ *(*resp. $\mathsf{Adv}^{\mathsf{RIBE\text{-}ad\text{-}weak}}_{\mathcal{A},\mathcal{O}^{RIBE}}(\lambda)$*).*

For convenience, we sometimes refer to a RIBE scheme satisfying weak security as RIBE without DKER.

2.6 2-level Hierarchical Identity Based Encryption

In this section, we introduce the notion of 2-level hierarchical identity based encryption (HIBE) [1, 12].

Syntax. A 2-level HIBE scheme $\Pi = (\mathsf{Setup}, \mathsf{GenSK}, \mathsf{Delegate}, \mathsf{Encrypt}, \mathsf{Decrypt})$ is described as follows:

$\mathsf{Setup}(1^\lambda) \to (\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}})$**:** on input the security parameter 1^λ, it outputs a public parameter pp and the KGC's secret key $\mathsf{sk}_{\mathsf{kgc}}$ (also called a master secret key).
We assume that the message space $\mathcal{M}$ and the identity space $\mathcal{ID}$ are determined only by the security parameter λ, and their descriptions are contained in pp.

$\mathsf{GenSK}(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}, \mathsf{id}_1) \to \mathsf{sk}_{\mathsf{id}_1}$**:** on input a public parameter pp, the KGC's secret key $\mathsf{sk}_{\mathsf{kgc}}$ and $\mathsf{id}_1 \in \mathcal{ID}$, it outputs a secret key $\mathsf{sk}_{\mathsf{id}_1}$.

$\mathsf{Delegate}(\mathsf{pp}, \mathsf{sk}_{\mathsf{id}_1}, \mathsf{id}_2)$**:** on input a public parameter pp, a secret key $\mathsf{sk}_{\mathsf{id}_1}$ (of a first-level user with $\mathsf{id}_1 \in \mathcal{ID}$), a second-level identity $\mathsf{id}_2 \in \mathcal{ID}$, it outputs a secret key $\mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}$.

$\mathsf{Encrypt}(\mathsf{pp}, \mathsf{ID} = (\mathsf{id}_1, \mathsf{id}_2), M) \to \mathsf{ct}_{\mathsf{ID},\mathsf{t}}$**:** on input a public parameter pp, a level-2 user's identity $\mathsf{ID} = (\mathsf{id}_1, \mathsf{id}_2) \in (\mathcal{ID})^2$, and a message M, it outputs a ciphertext ct.

$\mathsf{Decrypt}(\mathsf{pp}, \mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}, \mathsf{ct}) \to M$**:** on input a public parameter pp, a decryption key $\mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}$ (for a lever-2 user with identity $\mathsf{ID} = (\mathsf{id}_1, \mathsf{id}_2)$), and a ciphertext ct, it outputs the decryption result M.

Correctness. For all $\lambda \in \mathbb{N}$, $(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}) \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathsf{ID} = (\mathsf{id}_1, \mathsf{id}_2) \in (\mathcal{ID})^2$, $\mathsf{sk}_{\mathsf{id}_1} \leftarrow \mathsf{GenSK}(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}, \mathsf{id}_1)$, $sk_{\mathsf{id}_1,\mathsf{id}_2} \leftarrow \mathsf{Delegate}(\mathsf{pp}, \mathsf{sk}_{\mathsf{id}_1}, \mathsf{id}_2)$, $M \in \mathcal{M}$ and $\mathsf{ct} \leftarrow \mathsf{Encrypt}(\mathsf{pp}, \mathsf{ID}, M)$, the correctness requires $\mathsf{Decrypt}(\mathsf{pp}, \mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}, \mathsf{ct}) = M$.

Security. Let SKList be a list that initially contains $(\mathsf{kgc}, \mathsf{sk}_{\mathsf{kgc}})$. Whenever a new secret key is generated for an identity $\mathsf{ID} \in (\mathcal{ID})^{\leq 2}$ during the security game, the identity/secret key pair $(\mathsf{ID}, \mathsf{sk}_{\mathsf{ID}})$ will be stored into SKList and we will not mention this addition explicitly. Let $\mathcal{O}^{\mathsf{HIBE}}$ be the set of queries as follows:

Level-1 Secret Key Generation Query: upon a query $\mathsf{id}_1 \in \mathcal{ID}$ from the adversary $\mathcal{A}$, the challenge $\mathcal{C}$ check if $(\mathsf{id}_1, *) \in \mathsf{SKList}$, and returns $\perp$ to $\mathcal{A}$ if this is the case. Otherwise, $\mathcal{C}$ runs $\mathsf{sk}_{\mathsf{id}_1} \leftarrow \mathsf{GenSK}(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}, \mathsf{id}_1)$ and returns nothing to $\mathcal{A}$.

Level-1 Secret Key Reveal Query: upon a query $\mathsf{id}_1 \in \mathcal{ID}$ from $\mathcal{A}$, $\mathcal{C}$ checks if $(\mathsf{id}_1, \mathsf{sk}_{\mathsf{id}_1}) \in \mathsf{SKList}$ for some $\mathsf{sk}_{\mathsf{id}_1}$ and $\mathsf{id}_1 \neq \mathsf{id}_1^*$. If this is *not* the case, then $\mathcal{C}$ returns $\perp$ to $\mathcal{A}$. Otherwise, it returns $\mathsf{sk}_{\mathsf{id}_1}$ to $\mathcal{A}$.

Level-2 Secret Key Reveal Query: upon a query $(\mathsf{id}_1, \mathsf{id}_2) \in (\mathcal{ID})^2$ from $\mathcal{A}$, $\mathcal{C}$ checks if $(\mathsf{id}_1, \mathsf{sk}_{\mathsf{id}_1}) \in \mathsf{SKList}$ for some $\mathsf{sk}_{\mathsf{id}_1}$, $((\mathsf{id}_1, \mathsf{id}_2), \mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}) \notin \mathsf{SKList}$ and $(\mathsf{id}_1, \mathsf{id}_2) \neq (\mathsf{id}_1^*, \mathsf{id}_2^*)$. If this is *not* the case, then $\mathcal{C}$ returns $\perp$ to $\mathcal{A}$. Otherwise, it runs $\mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2} \leftarrow \mathsf{Delegate}(\mathsf{pp}, \mathsf{sk}_{\mathsf{id}_1}, \mathsf{id}_2)$ and returns $\mathsf{sk}_{\mathsf{id}_1,\mathsf{id}_2}$ to $\mathcal{A}$.

Definition 4. *A 2-level HIBE scheme satisfies selective-identity (*resp. *adaptive-identity) security, if any PPT adversary $\mathcal{A}$ has negligible advantage $\mathsf{Adv}^{\mathsf{HIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{HIBE}}(\lambda)$ (*resp. $\mathsf{Adv}^{\mathsf{HIBE\text{-}ad}}_{\mathcal{A},\mathcal{O}^{HIBE}}(\lambda)$*) in experiment $\mathsf{Exp}^{\mathsf{HIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{HIBE}}(\lambda)$ (*resp. $\mathsf{Exp}^{\mathsf{HIBE\text{-}ad}}_{\mathcal{A},\mathcal{O}^{HIBE}}(\lambda)$*):*

$\mathsf{Exp}^{\mathsf{HIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{HIBE}}(\lambda)$:
$\mathsf{ID}^* = (\mathsf{id}_1^*, \mathsf{id}_2^*) \leftarrow \mathcal{A}$;
$(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}) \leftarrow \mathsf{Setup}(1^\lambda)$; $\mathsf{SKList} \leftarrow (\mathsf{kgc}, \mathsf{sk}_{\mathsf{kgc}})$;
$(M_0, M_1) \leftarrow \mathcal{A}^{\mathcal{O}^{RIBE}}(\mathsf{pp})$, *where* $|M_0| = |M_1|$;
$b \leftarrow \{0,1\}$; $\mathsf{ct}^* \leftarrow \mathsf{Encrypt}(\mathsf{pp}, \mathsf{ID}^* = (\mathsf{id}_1^*, \mathsf{id}_2^*), M_b)$;
$b' \leftarrow \mathcal{A}^{\mathcal{O}^{HIBE}}(\mathsf{ct}^*)$;
Return 1 if $b' = b$ and 0 otherwise.

$\mathsf{Exp}^{\mathsf{HIBE\text{-}ad}}_{\mathcal{A},\mathcal{O}^{HIBE}}(\lambda)$:
$(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}) \leftarrow \mathsf{Setup}(1^\lambda)$; $\mathsf{SKList} \leftarrow (\mathsf{kgc}, \mathsf{sk}_{\mathsf{kgc}})$;
$(M_0, M_1, \mathsf{ID}^* = (\mathsf{id}_1^*, \mathsf{id}_2^*)) \leftarrow \mathcal{A}^{\mathcal{O}^{HIBE}}(\mathsf{pp})$, *where it is required that the following conditions are satisfied simultaneously:*
- $|M_0| = |M_1|$,
- $((\mathsf{id}_1^*, \mathsf{id}_2^*), *) \notin \mathsf{SKList}$,
- $\mathsf{sk}_{\mathsf{id}_1^*}$ *has not been revealed to* $\mathcal{A}$;

$b \leftarrow \{0,1\}$; $\mathsf{ct}^* \leftarrow \mathsf{Encrypt}(\mathsf{pp}, \mathsf{ID}^* = (\mathsf{id}_1^*, \mathsf{id}_2^*), M_b)$;
$b' \leftarrow \mathcal{A}^{\mathcal{O}^{HIBE}}(\mathsf{ct}^*)$;
Return 1 if $b' = b$ and 0 otherwise.

The advantage of $\mathcal{A}$ in the experiment $\mathsf{Exp}^{\mathsf{HIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{HIBE}}(\lambda)$ (resp. $\mathsf{Exp}^{\mathsf{HIBE\text{-}ad}}_{\mathcal{A},\mathcal{O}^{HIBE}}(\lambda)$) is defined as:

$$\begin{aligned} &\mathsf{Adv}^{\mathsf{HIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{HIBE}}(\lambda) = 2 \cdot \left| \Pr\left[\mathsf{Exp}^{\mathsf{HIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{HIBE}}(\lambda) = 1\right] - \frac{1}{2} \right|, \\ \Big(\textit{resp.}\quad &\mathsf{Adv}^{\mathsf{HIBE\text{-}ad}}_{\mathcal{A},\mathcal{O}^{HIBE}}(\lambda) = 2 \cdot \left| \Pr\left[\mathsf{Exp}^{\mathsf{HIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{HIBE}}(\lambda) = 1\right] - \frac{1}{2} \right| \Big). \end{aligned} \tag{2}$$

3 Framework and Security model of SR-IBE

In Section 2.5 we described why and how Katsumata et al. [12] redefined the definition and security model of RIBE, and the four adjustments they made. In this section, by absorbing the ideas of Katsumata et al. [12], we first re-formalize the framework of server-aided revocable identity-based encryption scheme (SR-IBE) based on [12, 20, 23]. Then, we propose a new and more refined security model for SR-IBE.

Note that in the previous security model of SR-IBE [20, 23], the challenge identity ID^* must be revoked before the challenge time t^* if the private key $\mathsf{sk}_{\mathsf{ID}^*}$ was revealed to the adversary. However, decrypting a ciphertext $\mathsf{ct}_{\mathsf{ID},\mathsf{t}}$ issued by the sender requires not only the private key $\mathsf{Priv}_{\mathsf{ID}}$(for delegation of the local decryption key $\mathsf{dk}_{\mathsf{ID},\mathsf{t}}$), but also the long-term transformation key $\mathsf{sk}_{\mathsf{ID}}$ (for generating the short-term transformation key $\mathsf{tk}_{\mathsf{ID},\mathsf{t}}$). Therefore, it is more natural to revoke ID^* before t^* when *both* the private key $\mathsf{Priv}_{\mathsf{ID}^*}$ and the long-term transformation key $\mathsf{sk}_{\mathsf{ID}^*}$ for ID^* are revealed. We additionally consider STKE attack, which means that the adversary is allowed to issue short-term transformation key queries for $(\mathsf{ID},\mathsf{t}) \neq (\mathsf{ID}^*,\mathsf{t}^*)$. The new SR-IBE security model with the above two changes is stronger because as mentioned in Section 1.1, the previous SR-IBE schemes [20, 23] are insecure in our new security model.

3.1 Framework of SR-IBE

In this section, we re-formalize the framework of SR-IBE. Compared with previous SR-IBE [20, 23], we make the following adjustments:

- An entity capable of deriving a secret key is modeled as a stateful entity and is supposed to maintain a so-called "state" in addition to its own secret key.
- We remove the "revoke" algorithm.
- We integrate the system parameter generation algorithm $\mathsf{Sys}(1^\lambda)$ into the setup algorithm $\mathsf{Setup}(1^\lambda)$.

A SR-IBE involves four parties: KGC, sender, recipient and server. Algorithms among the parties are as following:

$\mathsf{Setup}(1^\lambda) \to (\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}})$ is run by the KGC. It takes as input a security parameter 1^λ and outputs a public parameter pp, and the KGC's secret key $\mathsf{sk}_{\mathsf{kgc}}$ (also called a master key). We assume that the system parameter params is contained in pp and pp is an implicit input of all other algorithms.

$\mathsf{L\text{-}TranKG}(\mathsf{sk_{kgc}}, \mathsf{ID}) \to (\mathsf{sk_{ID}}, \mathsf{sk'_{kgc}})$ is run by the KGC. It takes as input the KGC's secret key $\mathsf{sk_{kgc}}$, an identity ID, and may update the KGC's secret key $\mathsf{sk_{kgc}}$. Then, it outputs a long-term transformation key $\mathsf{sk_{ID}}$ and the KGC's "updated" state $\mathsf{sk'_{kgc}}$. The long-term transformation key $\mathsf{sk_{ID}}$ is sent to the server through a public channel.

$\mathsf{UpdKG}(\mathsf{sk_{kgc}}, \mathsf{t}, \mathsf{RL_t}) \to (\mathsf{uk_t}, \mathsf{sk'_{kgc}})$ is run by the KGC. It takes as input the KGC's secret key $\mathsf{sk_{kgc}}$, a time period t, a revocation list $\mathsf{RL_t}$, and may update the KGC's secret key $\mathsf{sk_{kgc}}$. Then, it outputs an update key $\mathsf{uk_t}$ and the KGC's "updated" state $\mathsf{sk'_{kgc}}$. The updated key $\mathsf{uk_t}$ is sent to the server through a public channel.

$\mathsf{S\text{-}TranKG}(\mathsf{sk_{ID}}, \mathsf{uk_t}) \to \mathsf{tk_{ID,t}}/\perp$ is run by the server. It takes as input a long-term transformation key $\mathsf{sk_{ID}}$ for identity ID, an update key $\mathsf{uk_t}$ for time period t, and outputs a short-term transformation key $\mathsf{tk_{ID,t}}$ or the special symbol $\perp$ indicating that ID has been revoked.

$\mathsf{PrivKG}(\mathsf{sk_{kgc}}, \mathsf{ID}) \to \mathsf{Priv_{ID}}$ is run by the KGC. It takes as input the KGC's secret key $\mathsf{sk_{kgc}}$ and the recipient's identity ID, and outputs a private key $\mathsf{Priv_{ID}}$ for the recipient. The private key must be sent to the recipient through a secure channel.

$\mathsf{DecKG}(\mathsf{Priv_{ID}}, \mathsf{t}) \to \mathsf{dk_{ID,t}}$ is run by the recipient himself. It takes as input his private key $\mathsf{Priv_{ID}}$ and a time period t, and outputs a decryption key $\mathsf{dk_{ID,t}}$.

$\mathsf{Enc}(\mathsf{ID}, \mathsf{t}, M) \to \mathsf{ct_{ID,t}}$ is run by the sender. It takes as input the recipient's identity ID, a time period t, a message M, and outputs a ciphertext $\mathsf{ct_{ID,t}}$. The ciphertext is sent to the server.

$\mathsf{Transform}(\mathsf{tk_{ID,t}}, \mathsf{ct_{ID,t}}) \to \mathsf{ct'_{ID,t}}$ is run by the server. It takes as input a short-term transformation key $\mathsf{tk_{ID,t}}$, a ciphertext $\mathsf{ct_{ID,t}}$ and outputs a partially decrypted ciphertext $\mathsf{ct'_{ID,t}}$. The partially decrypted ciphertext $\mathsf{ct'_{ID,t}}$ is sent to the recipient through a public channel.

$\mathsf{Dec}(\mathsf{dk_{ID,t}}, \mathsf{ct'_{ID,t}}) \to M/\perp$ is run by the recipient. It takes as input a decryption key $\mathsf{dk_{ID,t}}$, a partially decrypted ciphertext $\mathsf{ct'_{ID,t}}$ and outputs a message M or a symbol $\perp$.

Correctness The *correctness* for a SR-IBE requires that for all security parameter $\lambda \in \mathbb{N}$ and all message M, if ID is not revoked at time period t and if all parties follow the prescribed algorithms, then we have $\mathsf{Dec}(\mathsf{dk_{ID,t}}, \mathsf{ct'_{ID,t}}) = M$.

3.2 Security model of SR-IBE

Qin et al. [23] defined the semantic security against adaptive-identity chosen plaintext attacks for SR-IBE, i.e., SR-aID-CPA security. Nguyen et al. [20] considered the selective-identity security, i.e., SR-sID-CPA security, in which the adversary announces the challenge identity ID^* and time period t^* before the execution of algorithm Setup.

Based on [12,20,23], we consider a Stronger selective-identity (*resp.* adaptive-identity) security model for SR-IBE, and call it SSR-sID-CPA (*resp.* SSR-aID-CPA) security, mainly making the following two changes: ID^* will be revoked

before t^* only if both $\mathsf{sk}_{\mathsf{ID}^*}$ and $\mathsf{Priv}_{\mathsf{ID}^*}$ for ID^* are revealed to the adversary; the adversary is allowed to issue short-term transformation key reveal queries for $(\mathsf{ID}, \mathsf{t}) \neq (\mathsf{ID}^*, \mathsf{t}^*)$. In addition, we make the following adjustments:

- Separate the long-term transformation key generation and long-term transformation key reveal queries, and the private key generation and private key reveal queries.
- Merge the "revoke" and "update key" queries into the single "revoke & update key" query and introduce a global counter t_{cu} representing the "current time period".

Let SKList be a set that initially contains $(\mathsf{kgc}, \mathsf{sk}_{\mathsf{kgc}})$, and into which identity/long-term transformation key pairs $(\mathsf{ID}, \mathsf{sk}_{\mathsf{ID}})$ generated during the security game will be stored. From now on, whenever a new secret key $\mathsf{sk}_{\mathsf{ID}}$ is generated for ID or the secret key $\mathsf{sk}_{\mathsf{kgc}}$ is updated during the execution of $\mathsf{L\text{-}TranKG}$ or UpdKG, the challenger will store the pair $(\mathsf{ID}, \mathsf{sk}_{\mathsf{ID}})$ or update $(\mathsf{kgc}, \mathsf{sk}_{\mathsf{kgc}})$ in SKList, and we will not explicitly mention this addition/update. Also, let $\mathsf{PrivList}$ be a set that stores the identity/private key pairs $(\mathsf{ID}, \mathsf{Priv}_{\mathsf{ID}})$ and identity/decryption key pairs $((\mathsf{ID}, \mathsf{t}), \mathsf{dk}_{\mathsf{ID},\mathsf{t}})$ generated during the security game. The challenger will store the pairs $(\mathsf{ID}, \mathsf{Priv}_{\mathsf{ID}})$ (*resp.* $((\mathsf{ID}, \mathsf{t}), \mathsf{dk}_{\mathsf{ID},\mathsf{t}})$) in $\mathsf{PrivList}$ whenever a new private key $\mathsf{Priv}_{\mathsf{ID}}$ (*resp.* decryption key $((\mathsf{ID}, \mathsf{t}), \mathsf{dk}_{\mathsf{ID},\mathsf{t}})$) is generated for ID (*resp.* $(\mathsf{ID}, \mathsf{t})$) during the execution of PrivKG (*resp.* DecKG) and we will not explicitly mention these additions as well.

Definition 5 (SSR-sID-CPA security). *Let $\mathcal{O}^{\mathsf{SR\text{-}IBE}}$ be the set of queries as follows:*

Long-term Transformation Key Generation Query: *upon a query* ID *from the adversary $\mathcal{A}$, where it is required that* $(\mathsf{ID}, *) \notin \mathsf{SKList}$*, the challenge $\mathcal{C}$ runs* $(\mathsf{sk}_{\mathsf{ID}}, \mathsf{sk}'_{\mathsf{kgc}}) \leftarrow \mathsf{L\text{-}TranKG}(\mathsf{sk}_{\mathsf{kgc}}, \mathsf{ID})$ *and returns nothing to $\mathcal{A}$.*
We require that all identities ID *appearing in the following queries be "activated" in the sense that* $\mathsf{sk}_{\mathsf{ID}}$ *is generated via this query and hence* $(\mathsf{ID}, \mathsf{sk}_{\mathsf{ID}}) \in \mathsf{SKList}$*.*

Long-term Transformation Key Reveal Query: *upon a query* ID *from $\mathcal{A}$, $\mathcal{C}$ finds* $\mathsf{sk}_{\mathsf{ID}}$ *from* SKList *and returns it to $\mathcal{A}$.*

Revoke & Update Key Query: *upon a query* RL *(which represents the set of identities that are going to be revoked in the next time period) from $\mathcal{A}$, $\mathcal{C}$ checks whether* $\mathsf{RL}_{\mathsf{t}_{\mathsf{cu}}} \subset \mathsf{RL}$*. If not, it returns $\perp$; otherwise, it increments the current time period by* $\mathsf{t}_{\mathsf{cu}} \leftarrow \mathsf{t}_{\mathsf{cu}} + 1$*, sets* $\mathsf{RL}_{\mathsf{t}_{\mathsf{cu}}} \leftarrow \mathsf{RL}$*, runs* $(\mathsf{uk}_{\mathsf{t}_{\mathsf{cu}}}, \mathsf{sk}'_{\mathsf{kgc}}) \leftarrow \mathsf{UpdKG}(\mathsf{sk}_{\mathsf{kgc}}, \mathsf{t}_{\mathsf{cu}}, \mathsf{RL}_{\mathsf{t}_{\mathsf{cu}}})$ *and returns* $\mathsf{uk}_{\mathsf{t}_{\mathsf{cu}}}$ *to $\mathcal{A}$.*

Short-term Transformation Key Reveal Query: *upon a query* $(\mathsf{ID}, \mathsf{t})$ *from $\mathcal{A}$, $\mathcal{C}$ checks whether conditions* $\mathsf{t} \leq \mathsf{t}_{\mathsf{cu}}$*,* $\mathsf{ID} \notin \mathsf{RL}_{\mathsf{t}}$ *and* $(\mathsf{ID}, \mathsf{t}) \neq (\mathsf{ID}^*, \mathsf{t}^*)$ *meet at the same time. If not, it returns $\perp$; otherwise, it finds* $\mathsf{sk}_{\mathsf{ID}}$ *from* SKList*, runs* $\mathsf{tk}_{\mathsf{ID},\mathsf{t}} \leftarrow \mathsf{S\text{-}TranKG}(\mathsf{sk}_{\mathsf{ID}}, \mathsf{uk}_{\mathsf{t}})$ *and returns* $\mathsf{tk}_{\mathsf{ID},\mathsf{t}}$ *to $\mathcal{A}$.*

Private Key Generation Query: *upon a query* ID *from $\mathcal{A}$, where it is required that* $(\mathsf{ID}, *) \notin \mathsf{PrivList}$*, $\mathcal{C}$ runs* $\mathsf{Priv}_{\mathsf{ID}} \leftarrow \mathsf{PrivKG}(\mathsf{sk}_{\mathsf{kgc}}, \mathsf{ID})$ *and returns nothing to $\mathcal{A}$.*

Similarly, we require that all identities ID *appearing in the following queries be "activated" in the sense that* $\mathsf{Priv}_{\mathsf{ID}}$ *is generated via this query and hence* $(\mathsf{ID}, \mathsf{Priv}_{\mathsf{ID}}) \in \mathsf{PrivList}$.

Private Key Reveal Query: *upon a query* ID *from* $\mathcal{A}$*,* $\mathcal{C}$ *finds* $\mathsf{Priv}_{\mathsf{ID}}$ *from* $\mathsf{PrivList}$ *and returns it to* $\mathcal{A}$.

Decryption Key Reveal Query: *upon a query* $(\mathsf{ID}, \mathsf{t})$ *from* $\mathcal{A}$*, where it is required that* $((\mathsf{ID}, \mathsf{t}), *) \notin \mathsf{PrivList}$*,* $\mathcal{C}$ *finds* $\mathsf{Priv}_{\mathsf{ID}}$ *from* $\mathsf{PrivList}$*, runs* $\mathsf{dk}_{\mathsf{ID},\mathsf{t}} \leftarrow \mathsf{DecKG}(\mathsf{Priv}_{\mathsf{ID}}, \mathsf{t})$ *and returns* $\mathsf{dk}_{\mathsf{ID},\mathsf{t}}$ *to* $\mathcal{A}$.

*A SR-IBE scheme is SSR-sID-CPA (*resp. *SSR-aID-CPA) secure if any PPT adversary* $\mathcal{A}$ *has negligible advantage in experiment* $\mathsf{Exp}^{\mathsf{SSR-sID-CPA}}_{\mathcal{A},\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda)$ *(*resp. $\mathsf{Exp}^{\mathsf{SSR-aID-CPA}}_{\mathcal{A},\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda)$*) as follows.*

$\mathsf{Exp}^{\mathsf{SSR-sID-CPA}}_{\mathcal{A},\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda)$:
$\mathsf{ID}^*, \mathsf{t}^* \leftarrow \mathcal{A}$;
$(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}) \leftarrow \mathsf{Setup}(1^\lambda)$; $\mathsf{SKList} \leftarrow (\mathsf{kgc}, \mathsf{sk}_{\mathsf{kgc}})$; $\mathsf{PrivList} \leftarrow \emptyset$;
$(\mathsf{uk}_1, \mathsf{sk}'_{\mathsf{kgc}}) \leftarrow \mathsf{UpdKG}(\mathsf{sk}_{\mathsf{kgc}}, \mathsf{t}_{\mathsf{cu}} = 1, \mathsf{RL}_1 = \emptyset)$;
$M_0, M_1 \leftarrow \mathcal{A}^{\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\mathsf{pp})$, *where* $|M_0| = |M_1|$;
$b \leftarrow \{0,1\}$; $\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*} \leftarrow \mathsf{Enc}(\mathsf{ID}^*, \mathsf{t}^*, M_b)$;
$b' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*})$;
Return 1 if $b' = b$ *and 0 otherwise.*

$\mathsf{Exp}^{\mathsf{SSR-aID-CPA}}_{\mathcal{A},\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda)$:
$(\mathsf{pp}, \mathsf{sk}_{\mathsf{kgc}}) \leftarrow \mathsf{Setup}(1^\lambda)$; $\mathsf{SKList} \leftarrow (\mathsf{kgc}, \mathsf{sk}_{\mathsf{kgc}})$; $\mathsf{PrivList} \leftarrow \emptyset$;
$(\mathsf{uk}_1, \mathsf{sk}'_{\mathsf{kgc}}) \leftarrow \mathsf{UpdKG}(\mathsf{sk}_{\mathsf{kgc}}, \mathsf{t}_{\mathsf{cu}} = 1, \mathsf{RL}_1 = \emptyset)$;
$(M_0, M_1, \mathsf{ID}^*, \mathsf{t}^*) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\mathsf{pp})$, *where* $|M_0| = |M_1|$.
$b \leftarrow \{0,1\}$; $\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*} \leftarrow \mathsf{Enc}(\mathsf{ID}^*, \mathsf{t}^*, M_b)$;
$b' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*})$;
Return 1 if $b' = b$ *and 0 otherwise.*

The following restrictions are made in the experiments $\mathsf{Exp}^{\mathsf{SSR-sID-CPA}}_{\mathcal{A},\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda)$ *and* $\mathsf{Exp}^{\mathsf{SSR-aID-CPA}}_{\mathcal{A},\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda)$*:*

1. *If both* $\mathsf{sk}_{\mathsf{ID}^*}$ *and* $\mathsf{Priv}_{\mathsf{ID}^*}$ *for identity* ID^* *are revealed to the adversary, then* $\mathsf{ID}^* \in \mathsf{RL}_{\mathsf{t}}$ *for some* $\mathsf{t} \leq \mathsf{t}^*$.
2. *If* $\mathsf{ID}^* \notin \mathsf{RL}_{\mathsf{t}^*}$*, then* $\mathsf{DecKG}(\cdot)$ *can not be queried on* $(\mathsf{ID}^*, \mathsf{t}^*)$.

The advantage of $\mathcal{A}$ *in the experiment* $\mathsf{Exp}^{\mathsf{SSR-sID-CPA}}_{\mathcal{A},\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda)$ *(*resp. $\mathsf{Exp}^{\mathsf{SSR-aID-CPA}}_{\mathcal{A},\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda)$*) is defined as:*

$$\begin{aligned} &\mathsf{Adv}^{\mathsf{SSR-sID-CPA}}_{\mathcal{A},\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda) = 2 \cdot \left| \Pr\left[\mathsf{Exp}^{\mathsf{SSR-sID-CPA}}_{\mathcal{A},\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda) = 1\right] - \frac{1}{2} \right| \\ \Big(\textit{resp.} \quad &\mathsf{Adv}^{\mathsf{SSR-aID-CPA}}_{\mathcal{A},\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda) = 2 \cdot \left| \Pr\left[\mathsf{Exp}^{\mathsf{SSR-aID-CPA}}_{\mathcal{A},\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda) = 1\right] - \frac{1}{2} \right| \Big). \end{aligned} \tag{3}$$

The following lemma is a variant of the strategy-dividing lemma in [12]. It is useful when proving the security of SR-IBE scheme.

Lemma 8 (Strategy-Dividing Lemma [12]). *Let Π be a SR-IBE scheme, and let $\mathcal{A}$ be any PPT adversary against the SSR-sID-CPA (*resp. *SSR-aID-CPA) security of Π. Assume that there are n possible attack strategies for $\mathcal{A}$, Type-1, $\cdots$, Type-n, that satisfy the following conditions:*

1. *Type-1, $\cdots$, Type-n cover all possible strategies, and each Type-i is mutually exclusive.*
2. *For every $i \in [n]$, whether $\mathcal{A}$ has deviated from the Type-i strategy is "publicly detectable", in the sense that during the security game, as soon as $\mathcal{A}$ deviates from the Type-i strategy, it can be efficiently recognized given $\mathcal{A}$'s view at the moment it deviates from the strategy.*

*Then, there exist PPT adversaries $\mathcal{A}_1, \cdots \mathcal{A}_n$ against the SSR-sID-CPA (*resp. *SSR-aID-CPA) security of Π, such that $\mathcal{A}_i$ always follows the Type-i strategy for every $i \in [n]$, and*

$$\begin{aligned} \mathsf{Adv}^{\mathsf{SSR\text{-}sID\text{-}CPA}}_{\Pi,\mathcal{A}}(\lambda) &\leq \sum_{i\in[n]} \mathsf{Adv}^{\mathsf{SSR\text{-}sID\text{-}CPA}}_{\Pi,\mathcal{A}_i}(\lambda) \\ \big(\text{resp.} \quad \mathsf{Adv}^{\mathsf{SSR\text{-}aID\text{-}CPA}}_{\Pi,\mathcal{A}}(\lambda) &\leq \sum_{i\in[n]} \mathsf{Adv}^{\mathsf{SSR\text{-}aID\text{-}CPA}}_{\Pi,\mathcal{A}_i}(\lambda).\big) \end{aligned} \tag{4}$$

*In particular, if $\mathsf{Adv}^{\mathsf{SSR\text{-}sID\text{-}CPA}}_{\Pi,\mathcal{A}_i}(\lambda)$ (*resp. $\mathsf{Adv}^{\mathsf{SSR\text{-}aID\text{-}CPA}}_{\Pi,\mathcal{A}_i}(\lambda)$*) is negligible for all PPT adversaries $\mathcal{A}_i$ that always follow the Type-i strategy and for all $i \in [n]$, then Π satisfies SSR-sID-CPA (*resp. *SSR-aID-CPA) security for any PPT adversary $\mathcal{A}$ following an arbitrary strategy.*

4 Our Lattice-based SR-IBE scheme

Nguyen et al. [20] constructed the first lattice-based SR-IBE by combining Chen et al.'s RIBE without DKER [8] with the 2-level HIBE [1] via a double encryption technique [14]. More specifically, the sender first encrypts the message M under the HIBE to get an initial ciphertext of the form $(\mathbf{c}_2, c_0 \in \mathbb{Z}_q)$ and then encrypts the binary representation of c_0 (i.e., $\mathsf{bin}(c_0) \in \{0,1\}^{k=\lceil \log q \rceil}$) under the RIBE to obtain $(\mathbf{c}_1, \hat{\mathbf{c}}_0)$. The final ciphertext is defined as $(\mathbf{c}_1, \mathbf{c}_2, \hat{\mathbf{c}}_0)$ and sent to the server, who will invert the second step of the encryption mechanism and send the initial ciphertext $(\mathbf{c}_2, c_0)$ to the recipient. In the end, the recipient only works with the HIBE block. However, the use of double encryption technique makes it necessary to call $k \cdot O(\log N)$ and $k \cdot O(r \cdot \log(N/r))$ sampling algorithms when generating a long-term transformation key and an update key, respectively. In addition, although the SR-IBE scheme constructed by Nguyen et al. [20] converts the receiver's workload from decrypting the ciphertext encrypted with RIBE [8] to decrypting the ciphertext encrypted with HIBE (in [1]), it is difficult to measure the extent to which the recipient's workload is increased or decreased.

Moreover, as mention in Section 1.1, Nguyen et al.'s SR-IBE scheme is insecure under SSR-sID-CPA security.

In this section, we construct the first *SSR-sID-CPA* secure SR-IBE by employing Katsumata et al.'s RIBE with DKER [12] and a 2-level HIBE [1] as the building blocks, where Katsumata et al.'s lattice-based RIBE with DKER is constructed by combining a RIBE without DKER [8] and the 2-level HIBE [1]. Note that in Katsumata et al.'s RIBE with DKER, the recipient needs to decrypt a "RIBE without DKER" ciphertext and a 2-level HIBE ciphertext. However, in our server-aided model, the server will decrypt the "RIBE with DKER" part of the ciphertext for the recipient and the recipient only needs to decrypt the HIBE block in the end. Therefore, the recipient reduces the workload of decrypting the "RIBE without DKER" ciphertext.

Compared with Nguyen et al. [20], we combine the two building blocks without using the double encryption technique, which reduced the number of calls to the sampling algorithm from $k \cdot O(\log N)$ and $k \cdot O(r \cdot \log(N/r))$ to $O(\log N)$ and $O(r \cdot \log(N/r))$, respectively. In addition, We reduce the recipient's workload in a strict sense, but not to an incalculable extent. Moreover, our SR-IBE is secure under a stronger security model, i.e., SSR-sID-CPA security. In the following, we formally describe our SR-IBE scheme.

$\mathsf{Setup}(1^\lambda)$**:** On input a security parameter λ, the KGC proceeds as follows:

1. Choose positive integers $n, N, q, k, m, s_0, s_1, \alpha$ with q a prime, $k = \lceil \log q \rceil$, N as the maximal number of users that the system will support. Select an FRD map $\mathsf{H} : \mathbb{Z}_q^n \to \mathbb{Z}_q^{n \times n}$ (see Section 2.4). Let the identity space $\mathcal{ID} = \mathbb{Z}_q^n$, the time space $\mathcal{T} \subset \mathcal{ID} = \mathbb{Z}_q^n$ and the message space $\mathcal{M} = \{0,1\}$. Set the system parameter $\mathsf{params} = (n, N, q, k, m, s_0, s_1, \alpha, \mathsf{H}, \mathcal{ID}, \mathcal{T}, \mathcal{M})$.
2. Generate three independent pairs $(\mathbf{A}, \mathbf{T_A})$, $(\mathbf{B}, \mathbf{T_B})$ and $(\mathbf{C}, \mathbf{T_C})$ by running $\mathsf{TrapGen}(n, m, q)$.
3. Select $\mathbf{u} \leftarrow \mathbb{Z}_q^n$ and $\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_1, \mathbf{B}_2, \mathbf{C}_1, \mathbf{C}_2 \leftarrow \mathbb{Z}_q^{n \times m}$.
4. Create a binary tree by running $\mathsf{BT_{kgc}} \leftarrow \mathsf{CS.Setup}(N)$.
5. Set the public parameter $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_1, \mathbf{B}_2, \mathbf{C}_1, \mathbf{C}_2, \mathbf{u}, \mathsf{params})$ and the KGC's secret key $\mathsf{sk_{kgc}} = (\mathbf{T_A}, \mathbf{T_B}, \mathbf{T_C}, \mathsf{BT_{kgc}})$.
6. Output pp and $\mathsf{sk_{kgc}}$.

$\mathsf{L\text{-}TranKG}(\mathsf{sk_{kgc}}, \mathsf{ID})$**:** On input the KGC's secret key $\mathsf{sk_{kgc}}$ and an identity $\mathsf{ID} \in \mathcal{ID}$, the KGC goes as follows:

1. Run $(\mathsf{BT_{kgc}}, \eta_{\mathsf{ID}}) \leftarrow \mathsf{CS.Assign}(\mathsf{BT_{kgc}}, \mathsf{ID})$.
2. For each $\theta \in \mathsf{path}(\mathsf{BT_{kgc}}, \eta_{\mathsf{ID}})$, if $\mathbf{u}_{\mathsf{kgc},\theta}$ is undefined, the KGC picks $\mathbf{u}_{\mathsf{kgc},\theta} \leftarrow \mathbb{Z}_q^n$, update $\mathsf{sk_{kgc}}$ by storing $\mathbf{u}_{\mathsf{kgc},\theta}$ in the node $\theta \in \mathsf{BT_{kgc}}$. Then, it samples $\mathbf{e}_{\mathsf{ID},\theta} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathbf{A_{ID}}, \mathbf{T_A}, \mathbf{u}_{\mathsf{kgc},\theta}, s_1)$, where $\mathbf{A_{ID}} = [\mathbf{A} \mid \mathbf{A}_1 + \mathsf{H}(\mathsf{ID})\mathbf{G}] \in \mathbb{Z}_q^{n \times 2m}$. Note that $\mathbf{e}_{\mathsf{ID},\theta} \in \mathbb{Z}^{2m}$ and $\mathbf{A_{ID}} \cdot \mathbf{e}_{\mathsf{ID},\theta} = \mathbf{u}_{\mathsf{kgc},\theta}$.
3. Extend its basis by running

$$\mathbf{T}_{\mathbf{B}_{\mathsf{ID}}} \leftarrow \mathsf{ExtRndLeft}(\mathbf{B}, \mathbf{B}_1 + \mathsf{H}(\mathsf{ID})\mathbf{G}, \mathbf{T_B}, s_0), \tag{5}$$

   where $\mathbf{B_{ID}} = [\mathbf{B} \mid \mathbf{B}_1 + \mathsf{H}(\mathsf{ID})\mathbf{G}] \in \mathbb{Z}_q^{n \times 2m}$, $\mathbf{T}_{\mathbf{B}_{\mathsf{ID}}} \in \mathbb{Z}^{2m \times 2m}$.
4. Output $\mathsf{sk_{ID}} = \big(\mathsf{path}(\mathsf{BT_{kgc}}, \eta_{\mathsf{ID}}), (\mathbf{e}_{\mathsf{ID},\theta})_{\theta \in \mathsf{path}(\mathsf{BT_{kgc}}, \eta_{\mathsf{ID}})}, \mathbf{T}_{\mathbf{B}_{\mathsf{ID}}}\big)$ as the long-term transformation key and the updated secret key $\mathsf{sk'_{kgc}}$.

$\mathsf{UpdKG}(\mathsf{sk_{kgc}}, \mathsf{t}, \mathsf{RL_t})$**:** On input the KGC's secret key $\mathsf{sk_{kgc}}$, a time period $\mathsf{t} \in \mathcal{T}$, a revocation list $\mathsf{RL_t}$, the KGC works as follows:

1. Run $\mathsf{KUNodes}(\mathsf{BT_{kgc}}, \mathsf{RL_t}) \leftarrow \mathsf{CS.Cover}(\mathsf{BT_{kgc}}, \mathsf{RL_t})$ and check whether $\mathbf{u}_{\mathsf{kgc},\theta}$ is defined for each $\theta \in \mathsf{KUNodes}(\mathsf{BT_{kgc}}, \mathsf{RL_t})$. If not, the KGC picks $\mathbf{u}_{\mathsf{kgc},\theta} \leftarrow \mathbb{Z}_q^n$, update $\mathsf{sk_{kgc}}$ by storing $\mathbf{u}_{\mathsf{kgc},\theta}$ in node $\theta \in \mathsf{BT_{kgc}}$. Then, it samples $\mathbf{e}_{\mathsf{t},\theta} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathbf{A}_\mathsf{t}, \mathbf{T}_\mathbf{A}, \mathbf{u} - \mathbf{u}_{\mathsf{kgc},\theta}, s_1)$, where $\mathbf{A}_\mathsf{t} = [\mathbf{A} \mid \mathbf{A}_2 + \mathsf{H}(\mathsf{t})\mathbf{G}] \in \mathbb{Z}_q^{n\times 2m}$. Note that $\mathbf{e}_{\mathsf{t},\theta} \in \mathbb{Z}^{2m}$ and $\mathbf{A}_\mathsf{t} \cdot \mathbf{e}_{\mathsf{t},\theta} = \mathbf{u} - \mathbf{u}_{\mathsf{kgc},\theta}$.
2. Output $\mathsf{uk_t} = \big(\mathsf{KUNodes}(\mathsf{BT_{kgc}}, \mathsf{RL_t}), (\mathbf{e}_{\mathsf{t},\theta})_{\theta\in\mathsf{KUNodes}(\mathsf{BT_{kgc}},\mathsf{RL_t})}\big)$ as the update key and the (possibly) updated secret key $\mathsf{sk'_{kgc}}$.

$\mathsf{S\text{-}TranKG}(\mathsf{sk_{ID}}, \mathsf{uk_t})$**:** On input a long-term transformation key $\mathsf{sk_{ID}}$ for identity $\mathsf{ID} \in \mathcal{ID}$, an update key $\mathsf{uk_t}$ for time period $\mathsf{t} \in \mathcal{T}$, the server goes as follows:

1. Extract $\mathsf{path}(\mathsf{BT_{kgc}}, \eta_\mathsf{ID})$ in $\mathsf{sk_{ID}}$ and $\mathsf{KUNodes}(\mathsf{BT_{kgc}}, \mathsf{RL_t})$ in $\mathsf{uk_t}$ and run $\theta/\emptyset \leftarrow \mathsf{CS.Match}(\mathsf{path}(\mathsf{BT_{kgc}}, \eta_\mathsf{ID}), \mathsf{KUNodes}(\mathsf{BT_{kgc}}, \mathsf{RL_t}))$. If the output is $\emptyset$, the server outputs $\perp$. Otherwise, it extracts $\mathbf{e}_{\mathsf{ID},\theta}, \mathbf{e}_{\mathsf{t},\theta} \in \mathbb{Z}^{2m}$ in $\mathsf{sk_{ID}}, \mathsf{uk_t}$, respectively, and parses it as
$$\mathbf{e}_{\mathsf{ID},\theta} = [\mathbf{e}^\mathsf{L}_{\mathsf{ID},\theta} \mid \mathbf{e}^\mathsf{R}_{\mathsf{ID},\theta}], \quad \mathbf{e}_{\mathsf{t},\theta} = [\mathbf{e}^\mathsf{L}_{\mathsf{t},\theta} \mid \mathbf{e}^\mathsf{R}_{\mathsf{t},\theta}]. \tag{6}$$
where $\mathbf{e}^\mathsf{L}_{\mathsf{ID},\theta}, \mathbf{e}^\mathsf{R}_{\mathsf{ID},\theta}, \mathbf{e}^\mathsf{L}_{\mathsf{t},\theta}, \mathbf{e}^\mathsf{R}_{\mathsf{t},\theta} \in \mathbb{Z}^m$. Then the server computes
$$\mathbf{e}_{\mathsf{ID},\mathsf{t}} = [\mathbf{e}^\mathsf{L}_{\mathsf{ID},\theta} + \mathbf{e}^\mathsf{L}_{\mathsf{t},\theta} \mid \mathbf{e}^\mathsf{R}_{\mathsf{ID},\theta} \mid \mathbf{e}^\mathsf{R}_{\mathsf{t},\theta}]. \tag{7}$$
Note that $\mathbf{e}_{\mathsf{ID},\mathsf{t}} \in \mathbb{Z}^{3m}$ and $[\mathbf{A} \mid \mathbf{A}_1 + \mathsf{H}(\mathsf{ID})\mathbf{G} \mid \mathbf{A}_2 + \mathsf{H}(\mathsf{t})\mathbf{G}] \cdot \mathbf{e}_{\mathsf{ID},\mathsf{t}} = \mathbf{u}$.
2. Sample $\mathbf{g}_{\mathsf{ID},\mathsf{t}} \leftarrow \mathsf{SampleLeft}(\mathbf{B}_\mathsf{ID}, \mathbf{B}_2 + \mathsf{H}(\mathsf{t})\mathbf{G}, \mathbf{T}_{\mathbf{B}_\mathsf{ID}}, \mathbf{u}, s_1)$. Note that $\mathbf{g}_{\mathsf{ID},\mathsf{t}} \in \mathbb{Z}^{3m}$ and $[\mathbf{B} \mid \mathbf{B}_1 + \mathsf{H}(\mathsf{ID})\mathbf{G} \mid \mathbf{B}_2 + \mathsf{H}(\mathsf{t})\mathbf{G}] \cdot \mathbf{g}_{\mathsf{ID},\mathsf{t}} = \mathbf{u}$.
3. Output the short-term transformation key $\mathsf{tk}_{\mathsf{ID},\mathsf{t}} = (\mathbf{e}_{\mathsf{ID},\mathsf{t}}, \mathbf{g}_{\mathsf{ID},\mathsf{t}})$.

$\mathsf{PrivKG}(\mathsf{sk_{kgc}}, \mathsf{ID})$**:** On input the KGC's secret key $\mathsf{sk_{kgc}}$ and an identity $\mathsf{ID} \in \mathcal{ID}$, the KGC proceeds as follows:

1. Sample $\mathbf{T}_{\mathbf{C}_\mathsf{ID}} \leftarrow \mathsf{ExtRndLeft}(\mathbf{C}, \mathbf{C}_1 + \mathsf{H}(\mathsf{ID})\mathbf{G}, \mathbf{T}_\mathbf{C}, s_0)$, where $\mathbf{C}_\mathsf{ID} = [\mathbf{C} \mid \mathbf{C}_1 + \mathsf{H}(\mathsf{ID})\mathbf{G}] \in \mathbb{Z}_q^{n\times 2m}$.
2. Output the private key $\mathsf{Priv_{ID}} = \mathbf{T}_{\mathbf{C}_\mathsf{ID}} \in \mathbb{Z}^{2m\times 2m}$.

$\mathsf{DecKG}(\mathsf{Priv_{ID}}, \mathsf{t})$**:** On input the private key $\mathsf{Priv_{ID}} = \mathbf{T}_{\mathbf{C}_\mathsf{ID}}$ and a time period $\mathsf{t} \in \mathcal{T}$, the recipient works as follows:

1. Sample $\mathbf{d}_{\mathsf{ID},\mathsf{t}} \leftarrow \mathsf{SampleLeft}(\mathbf{C}_\mathsf{ID}, \mathbf{C}_2 + \mathsf{H}(\mathsf{t})\mathbf{G}, \mathbf{T}_{\mathbf{C}_\mathsf{ID}}, \mathbf{u}, s_1)$. Note that $\mathbf{d}_{\mathsf{ID},\mathsf{t}} \in \mathbb{Z}^{3m}$ and $[\mathbf{C} \mid \mathbf{C}_1 + \mathsf{H}(\mathsf{ID})\mathbf{G} \mid \mathbf{C}_2 + \mathsf{H}(\mathsf{t})\mathbf{G}] \cdot \mathbf{d}_{\mathsf{ID},\mathsf{t}} = \mathbf{u}$.
2. Output the decryption key $\mathsf{dk}_{\mathsf{ID},\mathsf{t}} = \mathbf{d}_{\mathsf{ID},\mathsf{t}}$.

$\mathsf{Enc}(\mathsf{ID}, \mathsf{t}, M)$**:** On input an identity $\mathsf{ID} \in \mathcal{ID}$, a time period $\mathsf{t} \in \mathcal{T}$, a message $M \in \mathcal{M}$, the sender works as follows:

1. Set
$$\begin{aligned}
\mathbf{A}_{\mathsf{ID},\mathsf{t}} &= [\mathbf{A} \mid \mathbf{A}_1 + \mathsf{H}(\mathsf{ID})\mathbf{G} \mid \mathbf{A}_2 + \mathsf{H}(\mathsf{t})\mathbf{G}] \in \mathbb{Z}_q^{n\times 3m} \\
\mathbf{B}_{\mathsf{ID},\mathsf{t}} &= [\mathbf{B} \mid \mathbf{B}_1 + \mathsf{H}(\mathsf{ID})\mathbf{G} \mid \mathbf{B}_2 + \mathsf{H}(\mathsf{t})\mathbf{G}] \in \mathbb{Z}_q^{n\times 3m} \\
\mathbf{C}_{\mathsf{ID},\mathsf{t}} &= [\mathbf{C} \mid \mathbf{C}_1 + \mathsf{H}(\mathsf{ID})\mathbf{G} \mid \mathbf{C}_2 + \mathsf{H}(\mathsf{t})\mathbf{G}] \in \mathbb{Z}_q^{n\times 3m}.
\end{aligned} \tag{8}$$
2. Sample $\mathbf{s}, \mathbf{s}', \mathbf{s}'' \leftarrow \mathbb{Z}_q^n$, $x \leftarrow D_{\mathbb{Z},\alpha q}$, $\mathbf{x}, \mathbf{x}', \mathbf{x}'' \leftarrow \mathcal{D}_{\mathbb{Z}^m,\alpha q}$.
3. Sample $\mathbf{R}_{11}, \mathbf{R}_{12}, \mathbf{R}_{21}, \mathbf{R}_{22}, \mathbf{R}_{31}, \mathbf{R}_{32} \leftarrow \{-1, 1\}^{m\times m}$.

4. Set

$$\begin{aligned} c_0 &= \mathbf{u}^\top(\mathbf{s}+\mathbf{s}'+\mathbf{s}'')+x+M\cdot\lfloor\frac{q}{2}\rfloor, \quad \mathbf{c}_1=\mathbf{A}_{\mathsf{ID},\mathsf{t}}^\top\mathbf{s}+\begin{bmatrix}\mathbf{x}\\ \mathbf{R}_{11}^\top\mathbf{x}\\ \mathbf{R}_{12}^\top\mathbf{x}\end{bmatrix},\\ \mathbf{c}_2 &= \mathbf{B}_{\mathsf{ID},\mathsf{t}}^\top\mathbf{s}'+\begin{bmatrix}\mathbf{x}'\\ \mathbf{R}_{21}^\top\mathbf{x}'\\ \mathbf{R}_{22}^\top\mathbf{x}'\end{bmatrix}, \quad \mathbf{c}_3=\mathbf{C}_{\mathsf{ID},\mathsf{t}}^\top\mathbf{s}''+\begin{bmatrix}\mathbf{x}''\\ \mathbf{R}_{31}^\top\mathbf{x}''\\ \mathbf{R}_{32}^\top\mathbf{x}''\end{bmatrix}. \end{aligned} \tag{9}$$

5. Output the ciphertext $\mathsf{ct}_{\mathsf{ID},\mathsf{t}}=(c_0,\mathbf{c}_1,\mathbf{c}_2,\mathbf{c}_3)\in\mathbb{Z}_q\times\mathbb{Z}_q^{3m}\times\mathbb{Z}_q^{3m}\times\mathbb{Z}_q^{3m}$.

$\mathsf{Transform}(\mathsf{tk}_{\mathsf{ID},\mathsf{t}},\mathsf{ct}_{\mathsf{ID},\mathsf{t}})$**:** On input a short-term transformation key $\mathsf{tk}_{\mathsf{ID},\mathsf{t}}=(\mathbf{e}_{\mathsf{ID},\mathsf{t}},\mathbf{g}_{\mathsf{ID},\mathsf{t}})$, a ciphertext $\mathsf{ct}_{\mathsf{ID},\mathsf{t}}=(c_0,\mathbf{c}_1,\mathbf{c}_2,\mathbf{c}_3)$, the server works as follows:

1. Compute $c_0'=c_0-\mathbf{e}_{\mathsf{ID},\mathsf{t}}^\top\mathbf{c}_1-\mathbf{g}_{\mathsf{ID},\mathsf{t}}^\top\mathbf{c}_2$.
2. Output the partially decrypted ciphertext $\mathsf{ct}'_{\mathsf{ID},\mathsf{t}}=(c_0',\mathbf{c}_3)\in\mathbb{Z}_q\times\mathbb{Z}_q^{3m}$.

$\mathsf{Dec}(\mathsf{dk}_{\mathsf{ID},\mathsf{t}},\mathsf{ct}'_{\mathsf{ID},\mathsf{t}})$**:** On input a decryption key $\mathsf{dk}_{\mathsf{ID},\mathsf{t}}=\mathbf{d}_{\mathsf{ID},\mathsf{t}}$, a partially decrypted ciphertext $\mathsf{ct}'_{\mathsf{ID},\mathsf{t}}$, the recipient works as follows:

1. Compute $c=c_0'-\mathbf{d}_{\mathsf{ID},\mathsf{t}}^\top\mathbf{c}_3\in\mathbb{Z}_q$.
2. Compare c and $\lfloor\frac{q}{2}\rfloor$ by treating them as integers in $\mathbb{Z}$, output 1 in case $|c-\lfloor\frac{q}{2}\rfloor|<\lfloor\frac{q}{4}\rfloor$ and 0 otherwise.

4.1 Correctness

Let a ciphertext $\mathsf{ct}_{\mathsf{ID},\mathsf{t}}$ be aimed for recipient ID and time period t. To check correctness, we only need to consider the situation where the ID has not been revoked at time period t. In this case, the server can construct a short-term transformation key $\mathsf{tk}_{\mathsf{ID},\mathsf{t}}$ for ID at time t.

Lemma 9. *Assume* $O\left(\left(\alpha+\alpha s_1 m\cdot\omega(\sqrt{\log m})\right)\cdot q\right)\leq q/5$ *hold with overwhelming probability, then the above SR-IBE scheme has negligible decryption error.*

Proof. Since ID is non-revoked at time period t, there exists one node $\theta\in\mathsf{path}(\mathsf{BT}_{\mathsf{kgc}},\eta_{\mathsf{ID}})\cap\mathsf{KUNodes}(\mathsf{BT}_{\mathsf{kgc}},\mathsf{RL}_{\mathsf{t}})$. The short-term transformation key generated by running $\mathsf{S\text{-}TranKG}(\mathsf{sk}_{\mathsf{ID}},\mathsf{uk}_{\mathsf{t}})$ is of the form $\mathsf{tk}_{\mathsf{ID},\mathsf{t}}=(\mathbf{e}_{\mathsf{ID},\mathsf{t}},\mathbf{g}_{\mathsf{ID},\mathsf{t}})$ such that

$$\begin{aligned} \mathbf{A}_{\mathsf{ID},\mathsf{t}}\cdot\mathbf{e}_{\mathsf{ID},\mathsf{t}}&=[\mathbf{A}\mid\mathbf{A}_1+\mathsf{H}(\mathsf{ID})\mathbf{G}\mid\mathbf{A}_2+\mathsf{H}(\mathsf{t})\mathbf{G}]\cdot\mathbf{e}_{\mathsf{ID},\mathsf{t}}=\mathbf{u},\\ \mathbf{B}_{\mathsf{ID},\mathsf{t}}\cdot\mathbf{g}_{\mathsf{ID},\mathsf{t}}&=[\mathbf{B}\mid\mathbf{B}_1+\mathsf{H}(\mathsf{ID})\mathbf{G}\mid\mathbf{B}_2+\mathsf{H}(\mathsf{t})\mathbf{G}]\cdot\mathbf{g}_{\mathsf{ID},\mathsf{t}}=\mathbf{u}. \end{aligned} \tag{10}$$

During the $\mathsf{Transform}$ procedure performed by the server, we have

$$
\begin{aligned}
c_0' =& c_0 - \mathbf{e}_{\mathsf{ID},\mathsf{t}}^\top \mathbf{c}_1 - \mathbf{g}_{\mathsf{ID},\mathsf{t}}^\top \mathbf{c}_2 \\
=& \mathbf{u}^\top(\mathbf{s}+\mathbf{s}'+\mathbf{s}'') + x + M \cdot \lfloor \frac{q}{2} \rfloor - \left(\mathbf{u}^\top \mathbf{s} + \mathbf{e}_{\mathsf{ID},\mathsf{t}}^\top \begin{bmatrix} \mathbf{x} \\ \mathbf{R}_{11}^\top \mathbf{x} \\ \mathbf{R}_{12}^\top \mathbf{x} \end{bmatrix} \right) - \\
& \left(\mathbf{u}^\top \mathbf{s}' + \mathbf{g}_{\mathsf{ID},\mathsf{t}}^\top \begin{bmatrix} \mathbf{x}' \\ \mathbf{R}_{21}^\top \mathbf{x}' \\ \mathbf{R}_{22}^\top \mathbf{x}' \end{bmatrix} \right) \\
=& \mathbf{u}^\top \mathbf{s}'' + x + M \cdot \lfloor \frac{q}{2} \rfloor - \mathbf{e}_{\mathsf{ID},\mathsf{t}}^\top \begin{bmatrix} \mathbf{x} \\ \mathbf{R}_{11}^\top \mathbf{x} \\ \mathbf{R}_{12}^\top \mathbf{x} \end{bmatrix} - \mathbf{g}_{\mathsf{ID},\mathsf{t}}^\top \begin{bmatrix} \mathbf{x}' \\ \mathbf{R}_{21}^\top \mathbf{x}' \\ \mathbf{R}_{22}^\top \mathbf{x}' \end{bmatrix}.
\end{aligned} \tag{11}
$$

The decryption key generated by running $\mathsf{DecKG}(\mathsf{Priv}_{\mathsf{ID}}, \mathsf{t})$ is of the form $\mathsf{dk}_{\mathsf{ID},\mathsf{t}} = \mathbf{d}_{\mathsf{ID},\mathsf{t}}$ such that

$$
\mathbf{C}_{\mathsf{ID},\mathsf{t}} \cdot \mathbf{d}_{\mathsf{ID},\mathsf{t}} = [\mathbf{C} \mid \mathbf{C}_1 + \mathsf{H}(\mathsf{ID})\mathbf{G} \mid \mathbf{C}_2 + \mathsf{H}(\mathsf{t})\mathbf{G}] \cdot \mathbf{d}_{\mathsf{ID},\mathsf{t}} = \mathbf{u}. \tag{12}
$$

During the Dec procedure performed by the recipient, we have

$$
\begin{aligned}
c =& c_0' - \mathbf{d}_{\mathsf{ID},\mathsf{t}}^\top \mathbf{c}_3 \\
=& M \cdot \lfloor \frac{q}{2} \rfloor + \underbrace{x - \mathbf{e}_{\mathsf{ID},\mathsf{t}}^\top \begin{bmatrix} \mathbf{x} \\ \mathbf{R}_{11}^\top \mathbf{x} \\ \mathbf{R}_{12}^\top \mathbf{x} \end{bmatrix} - \mathbf{g}_{\mathsf{ID},\mathsf{t}}^\top \begin{bmatrix} \mathbf{x}' \\ \mathbf{R}_{21}^\top \mathbf{x}' \\ \mathbf{R}_{22}^\top \mathbf{x}' \end{bmatrix} - \mathbf{d}_{\mathsf{ID},\mathsf{t}}^\top \begin{bmatrix} \mathbf{x}'' \\ \mathbf{R}_{31}^\top \mathbf{x}'' \\ \mathbf{R}_{32}^\top \mathbf{x}'' \end{bmatrix}}_{:= z(\text{“noise''})}.
\end{aligned} \tag{13}
$$

If we set the parameters appropriately, by Lemma 1 and 5, we know that

$$
\|\mathbf{e}_{\mathsf{ID},\mathsf{t}}\| \leq 2 \cdot \sqrt{3m} \cdot s_1, \quad \|\mathbf{g}_{\mathsf{ID},\mathsf{t}}\| \leq \sqrt{3m} \cdot s_1, \quad \|\mathbf{d}_{\mathsf{ID},\mathsf{t}}\| \leq \sqrt{3m} \cdot s_1. \tag{14}
$$

By Lemma 1, 5 and 6, we have

$$
\begin{aligned}
&\|[\mathbf{I} \mid \mathbf{R}_{11} \mid \mathbf{R}_{12}] \cdot \mathbf{e}_{\mathsf{ID},\mathsf{t}}\| \leq (1 + 2 \cdot \sqrt{2m}) \cdot 2 \cdot \sqrt{3m} \cdot s_1 \leq O(s_1 m), \\
&\|[\mathbf{I} \mid \mathbf{R}_{21} \mid \mathbf{R}_{22}] \cdot \mathbf{g}_{\mathsf{ID},\mathsf{t}}\| \leq O(s_1 m), \quad \|[\mathbf{I} \mid \mathbf{R}_{31} \mid \mathbf{R}_{32}] \cdot \mathbf{d}_{\mathsf{ID},\mathsf{t}}\| \leq O(s_1 m).
\end{aligned} \tag{15}
$$

By Lemma 1 and the standard Gaussian tail bound in [10], we have

$$
\begin{aligned}
\|([\mathbf{I} \mid \mathbf{R}_{11} \mid \mathbf{R}_{12}] \cdot \mathbf{e}_{\mathsf{ID},\mathsf{t}})^\top \cdot \mathbf{x}\| &\leq \|[\mathbf{I} \mid \mathbf{R}_{11} \mid \mathbf{R}_{12}] \cdot \mathbf{e}_{\mathsf{ID},\mathsf{t}}\| \cdot \alpha q \cdot \omega(\sqrt{\log m}), \\
&\leq O(s_1 m) \cdot \alpha q \cdot \omega(\sqrt{\log m}).
\end{aligned} \tag{16}
$$

Similarly, $\|([\mathbf{I} \mid \mathbf{R}_{21} \mid \mathbf{R}_{22}] \cdot \mathbf{g}_{\mathsf{ID},\mathsf{t}})^\top \cdot \mathbf{x}'\|, \|([\mathbf{I} \mid \mathbf{R}_{31} \mid \mathbf{R}_{32}] \cdot \mathbf{d}_{\mathsf{ID},\mathsf{t}})^\top \cdot \mathbf{x}''\| \leq O(s_1 m) \cdot \alpha q \cdot \omega(\sqrt{\log m})$. Thus, the noise z can be bounded with overwhelming probability as follows:

$$
\begin{aligned}
\|z\| &\leq |x| + \|\mathbf{e}_{\mathsf{ID},\mathsf{t}}^\top \begin{bmatrix} \mathbf{x} \\ \mathbf{R}_{11}^\top \mathbf{x} \\ \mathbf{R}_{12}^\top \mathbf{x} \end{bmatrix}\| + \|\mathbf{g}_{\mathsf{ID},\mathsf{t}}^\top \begin{bmatrix} \mathbf{x}' \\ \mathbf{R}_{21}^\top \mathbf{x}' \\ \mathbf{R}_{22}^\top \mathbf{x}' \end{bmatrix}\| + \|\mathbf{d}_{\mathsf{ID},\mathsf{t}}^\top \begin{bmatrix} \mathbf{x}'' \\ \mathbf{R}_{31}^\top \mathbf{x}'' \\ \mathbf{R}_{32}^\top \mathbf{x}'' \end{bmatrix}\|, \\
&\leq \alpha q + O(s_1 m) \cdot \alpha q \cdot \omega(\sqrt{\log m}), \\
&= O\left(\left(\alpha + \alpha s_1 m \cdot \omega(\sqrt{\log m}) \right) \cdot q \right).
\end{aligned} \tag{17}
$$

□

4.2 Parameter Selection

In this section, we provide an example of parameter selection of our SRIBE scheme. Note that we need to ensure that

- the "noise" term in the above section is less than $q/5$ with overwhelming probability (i.e., $O\left(\left(\alpha + \alpha s_1 m \cdot \omega(\sqrt{\log m})\right) \cdot q\right) \leq q/5$ by Lemma 9).
- algorithm Trap operates as specified (i.e., $m \geq 2n\lceil \log q \rceil$ by Lemma 3).
- algorithms $\mathsf{SampleLeft}$ and $\mathsf{ExtRndLeft}$ work as specified (i.e., $s_0 \geq O(\sqrt{n \log q}) \cdot \omega(\sqrt{\log m})$, $s_1 \geq s_0\sqrt{m} \cdot \omega(\sqrt{\log m})$ by Lemma 1, 3, 4 and 5).
- algorithm $\mathsf{ExtRndRight}$ works as specified in the security proof (i.e., $s_0 > \sqrt{m} \cdot \omega(\sqrt{\log m})$ by Lemma 3, 4 and 6).
- the hardness assumption of LWE applies (i.e., $\alpha q > 2\sqrt{n}$).

According to the above restrictions, we can set the parameters of our SR-IBE as follows:

$$\begin{aligned} &n = O(\lambda), \quad N = \mathsf{poly}(\lambda), \quad m = O(n \log q), \quad s_0 = \sqrt{m} \cdot \omega(\sqrt{\log n}), \\ &s_1 = m \cdot \omega(\log n), \quad \alpha = m^{-2} \cdot \omega((\log n)^{\frac{3}{2}})^{-1}, \quad q = n^{\frac{1}{2}} \cdot m^2 \cdot \omega((\log n)^{\frac{3}{2}}). \end{aligned} \tag{18}$$

Remark 1. For the sake of simplicity, we define the correctness of SR-IBE to have a probability of 1 in Section 3.1. Therefore, in order to be consistent with our definition, we can use standard techniques to modify our lattice-based construction so that there are no decryption errors by considering a bound on the secret/noise vector.

4.3 Security Analysis

Theorem 1. *The SR-IBE scheme described above is SSR-sID-CPA secure assuming the hardness of the* $\mathsf{LWE}_{n,m+1,q,\chi}$ *problem, where* $\chi = \mathcal{D}_{\mathbb{Z}^{m+1},\alpha q}$.

Proof. Let $\mathcal{A}$ be a PPT adversary who succeeds in breaking the SSR-sID-CPA security of our SR-IBE scheme with advantage $\mathsf{Adv}^{\mathsf{SSR\text{-}sID\text{-}CPA}}_{\mathcal{A},\mathcal{O}^{\mathsf{SR\text{-}IBE}}}(\lambda) = \epsilon$, let ID^* be the challenge identity and t^* be the challenge time period. Observe that the strategy taken by $\mathcal{A}$ can be divided into three types of strategies that are mutually exclusive as follows.

Type-I: $\mathcal{A}$ issues both private key reveal query and long-term transformation key reveal query on the challenge identity ID^*. In this case, the challenge identity ID^* must be revoked before the challenge time t^*.

Type-II: $\mathcal{A}$ issues private key reveal query on the challenge identity ID^*, but does not issue long-term transformation key reveal query on ID^*.

Type-III: $\mathcal{A}$ does not issue private key reveal query on the challenge identity ID^*.

By Lemma 8, $\mathcal{A}$ always follows one of the above strategies, and we only need to show that the advantage of $\mathcal{A}$ is negligible regardless of the strategy taken by $\mathcal{A}$.

We will provide three types of security proofs according to the three types of strategies taken by $\mathcal{A}$ as described above. In both proofs, we proceed with a sequence of games where the first game is identical to the SSR-sID-CPA game from Definition 5 and the adversary has no advantage in the last game. We will show that $\mathcal{A}$ cannot distinguish between the games, which will prove that the adversary has negligible advantage in winning the original SSR-sID-CPA game and will complete our proof.

Lemma 10. *The advantage of an adversary $\mathcal{A}_1$ using the Type-***I** strategy is negligible assuming the hardness of the $\mathsf{LWE}_{n,m+1,q,\chi}$, where $\chi = \mathcal{D}_{\mathbb{Z}^{m+1},\alpha q}$.

Proof. We define the sequence of games as follows:

$\mathsf{Game}_{\mathbf{I}\text{-}0}$: This is the original SSR-sID-CPA game from Definition 5.

$\mathsf{Game}_{\mathbf{I}\text{-}1}$: In this game, we change the way that the challenger generates $\mathbf{A}_1, \mathbf{A}_2$ in the public parameters. The $\mathsf{Game}_{\mathbf{I}\text{-}1}$ challenger samples $\mathbf{R}^*_{11}, \mathbf{R}^*_{12} \leftarrow \{-1, 1\}^{m\times m}$ at the setup phase and sets $\mathbf{A}_1, \mathbf{A}_2$ as

$$\mathbf{A}_1 = \mathbf{A}\mathbf{R}^*_{11} - \mathsf{H}(\mathsf{ID}^*)\mathbf{G}, \quad \mathbf{A}_2 = \mathbf{A}\mathbf{R}^*_{12} - \mathsf{H}(\mathsf{t}^*)\mathbf{G}. \tag{19}$$

Then the challenger keeps the matrices $\mathbf{R}^*_{11}, \mathbf{R}^*_{12}$ as part of $\mathsf{sk}_{\mathsf{kgc}}$. The remainder of the game is unchanged.

We now show that $\mathsf{Game}_{\mathbf{I}\text{-}0}$ is statistically indistinguishable from $\mathsf{Game}_{\mathbf{I}\text{-}1}$. Note that in $\mathsf{Game}_{\mathbf{I}\text{-}1}$ the matrices $\mathbf{R}^*_{11}, \mathbf{R}^*_{12}$ are used only in the construction of $\mathbf{A}_1, \mathbf{A}_2$ and in the construction of the challenge ciphertext where $\mathbf{z}_1 \leftarrow (\mathbf{R}^*_{11})^\top \mathbf{x}$, $\mathbf{z}_2 \leftarrow (\mathbf{R}^*_{12})^\top \mathbf{x}$. By Lemma 7, the distributions $(\mathbf{A}, \mathbf{A}\mathbf{R}^*_{11}, \mathbf{z}_1)$ and $(\mathbf{A}, \mathbf{A}\mathbf{R}^*_{12}, \mathbf{z}_2)$ are statistically close to the distributions $(\mathbf{A}, \mathbf{A}'_1, \mathbf{z}_1)$ and $(\mathbf{A}, \mathbf{A}'_2, \mathbf{z}_2)$ respectively, where $\mathbf{A}'_1, \mathbf{A}'_2 \leftarrow \mathbb{Z}_q^{n\times m}$. Hence, $\mathbf{A}_1, \mathbf{A}_2$ in $\mathsf{Game}_{\mathbf{I}\text{-}0}$ and $\mathsf{Game}_{\mathbf{I}\text{-}1}$ are indistinguishable.

$\mathsf{Game}_{\mathbf{I}\text{-}2}$: In this game, we change how we assign ID^* to the binary tree $\mathsf{BT}_{\mathsf{kgc}}$. Recall in the previous game, the challenger assigned ID^* to some random leaf η_{ID^*} of $\mathsf{BT}_{\mathsf{kgc}}$ when $\mathcal{A}_1$ issued a long-term transformation key query on ID^*. In this game, the $\mathsf{Game}_{\mathbf{I}\text{-}2}$ challenger chooses a random leaf in $\mathsf{BT}_{\mathsf{kgc}}$ to assign ID^* before providing $\mathcal{A}_1$ the public parameter pp. When $\mathcal{A}_1$ submitted a long-term transformation key query on some ID, if $\mathsf{ID} = \mathsf{ID}^*$, then the challenger proceeds with $\mathsf{L\text{-}TranKG}$ as if $(\mathsf{BT}_{\mathsf{kgc}}, \eta_{\mathsf{ID}^*}) \leftarrow \mathsf{CS.Assign}(\mathsf{BT}_{\mathsf{kgc}}, \mathsf{ID}^*)$ and otherwise it assigns ID to some random leaf of $\mathsf{BT}_{\mathsf{kgc}}$ that is not η_{ID^*}. Since the random assignment of ID^* made by the challenger is statistically hidden from $\mathcal{A}_1$, $\mathsf{Game}_{\mathbf{I}\text{-}1}$ and $\mathsf{Game}_{\mathbf{I}\text{-}2}$ are indistinguishable.

$\mathsf{Game}_{\mathbf{I}\text{-}3}$: In this game, we change the challenger so he does not have to use the trapdoor $\mathbf{T}_{\mathbf{A}}$ when generating the following short vectors: $(\mathbf{e}_{\mathsf{ID},\theta})_{\theta\in\mathsf{path}(\mathsf{BT}_{\mathsf{kgc}},\eta_{\mathsf{ID}})}$ in $\mathsf{sk}_{\mathsf{ID}}$, $(\mathbf{e}_{\mathsf{t},\theta})_{\theta\in\mathsf{KUNodes}(\mathsf{BT}_{\mathsf{kgc}},\mathsf{RL}_{\mathsf{t}})}$ in uk_{t}, and $\mathbf{e}_{\mathsf{ID},\mathsf{t}}$ in $\mathsf{tk}_{\mathsf{ID},\mathsf{t}}$. To achieve this goal, we

modify when and how the vectors $\mathbf{u}_{\mathsf{kgc},\theta}$ for each node $\theta \in \mathsf{BT}_{\mathsf{kgc}}$ are chosen. By the definition of the Type-**I** strategy, ID^* must be revoked before time period t^*. Hence, we have $\mathsf{path}(\mathsf{BT}_{\mathsf{kgc}}, \eta_{\mathsf{ID}^*}) \cap \mathsf{KUNodes}(\mathsf{BT}_{\mathsf{kgc}}, \mathsf{RL}_{\mathsf{t}^*}) = \emptyset$ by the property of the CS scheme.

Whenever $\mathcal{A}_1$ issues a long-term transformation key query, a revoke & update key query, or a short-term transformation key reveal query, the $\mathsf{Game}_{\mathbf{I}\text{-}3}$ challenger generates the vectors $\mathbf{u}_{\mathsf{kgc},\theta}$ for each node $\theta \in \mathsf{BT}_{\mathsf{kgc}}$ as follows:

- If $\theta \in \mathsf{path}(\mathsf{BT}_{\mathsf{kgc}}, \eta_{\mathsf{ID}^*})$, then it samples $\mathbf{e}_{\mathsf{ID}^*,\theta} \leftarrow \mathcal{D}_{\mathbb{Z}^{2m},s_1}$, sets $\mathbf{u}_{\mathsf{kgc},\theta} = \mathbf{A}_{\mathsf{ID}^*} \cdot \mathbf{e}_{\mathsf{ID}^*,\theta}$ for $\mathbf{A}_{\mathsf{ID}^*} = [\mathbf{A} \mid \mathbf{A}_1 + \mathsf{H}(\mathsf{ID}^*)\mathbf{G}]$, stores $\mathbf{u}_{\mathsf{kgc},\theta}$ in the node θ and keeps $\mathbf{e}_{\mathsf{ID}^*,\theta}$ secret.
- If $\theta \notin \mathsf{path}(\mathsf{BT}_{\mathsf{kgc}}, \eta_{\mathsf{ID}^*})$, then it samples $\mathbf{e}_{\mathsf{t}^*,\theta} \leftarrow \mathcal{D}_{\mathbb{Z}^{2m},s_1}$, sets $\mathbf{u}_{\mathsf{kgc},\theta} = \mathbf{A}_{\mathsf{t}^*} \cdot \mathbf{e}_{\mathsf{t}^*,\theta}$ for $\mathbf{A}_{\mathsf{t}^*} = [\mathbf{A} \mid \mathbf{A}_2 + \mathsf{H}(\mathsf{t}^*)\mathbf{G}]$ by implicitly setting $\mathsf{t}_{\mathsf{cu}} = \mathsf{t}^*$, stores $\mathbf{u}_{\mathsf{kgc},\theta}$ in the node θ and keeps $\mathbf{e}_{\mathsf{t}^*,\theta}$ secret.

Then, if $\mathcal{A}_1$ issues a long-term transformation key query on $\mathsf{ID} \neq \mathsf{ID}^*$, the $\mathsf{Game}_{\mathbf{I}\text{-}3}$ challenger first runs $\mathsf{ExtRndRight}(\mathbf{A}, \mathbf{G}, \mathbf{R}_{11}^*, \mathbf{T}_{\mathbf{G}}, s_0)$ to create the trapdoor $\mathbf{T}_{\mathbf{A}_{\mathsf{ID}}=[\mathbf{A}|\mathbf{A}_1+\mathsf{H}(\mathsf{ID})\mathbf{G}]=[\mathbf{A}|\mathbf{A}\mathbf{R}_{11}^*+(\mathsf{H}(\mathsf{ID})-\mathsf{H}(\mathsf{ID}^*))\mathbf{G}]}$ and then samples the short vectors $(\mathbf{e}_{\mathsf{ID},\theta})_{\theta \in \mathsf{path}(\mathsf{BT}_{\mathsf{kgc}},\eta_{\mathsf{ID}})}$ by running $\mathsf{SamplePre}(\cdot)$ with trapdoor $\mathbf{T}_{\mathbf{A}_{\mathsf{ID}}}$ and Gaussian parameter s_1. Otherwise, if $\mathsf{ID} = \mathsf{ID}^*$, the challenger simply returns $(\mathbf{e}_{\mathsf{ID}^*,\theta})_\theta$ which he has already generated without using $\mathbf{T}_{\mathbf{A}}$. Moreover, if the counter t_{cu} on which $\mathcal{A}_1$ queries a revoke & key update query is not t^*, the $\mathsf{Game}_{\mathbf{I}\text{-}3}$ challenger first runs $\mathsf{ExtRndRight}(\mathbf{A}, \mathbf{G}, \mathbf{R}_{12}^*, \mathbf{T}_{\mathbf{G}}, s_0)$ to create the trapdoor $\mathbf{T}_{\mathbf{A}_{\mathsf{t}}=[\mathbf{A}|\mathbf{A}_2+\mathsf{H}(\mathsf{t})\mathbf{G}]=[\mathbf{A}|\mathbf{A}\mathbf{R}_{12}^*+(\mathsf{H}(\mathsf{t})-\mathsf{H}(\mathsf{t}^*))\mathbf{G}]}$ and then samples the short vectors $(\mathbf{e}_{\mathsf{t},\theta})_{\theta \in \mathsf{KUNodes}(\mathsf{BT}_{\mathsf{kgc}},\mathsf{RL}_{\mathsf{t}})}$ by running $\mathsf{SamplePre}(\cdot)$ with trapdoor $\mathbf{T}_{\mathbf{A}_{\mathsf{t}}}$ and Gaussian parameter s_1. Otherwise, if $\mathsf{t}_{\mathsf{cu}} = \mathsf{t}^*$, the challenger simply returns $(\mathbf{e}_{\mathsf{t}^*,\theta})_\theta$ which he has already created without using $\mathbf{T}_{\mathbf{A}}$. Furthermore, when $\mathcal{A}_1$ issues a short-term transformation key reveal query on $(\mathsf{ID}, \mathsf{t})$, the challenger checks whether conditions $\mathsf{t} \leq \mathsf{t}_{\mathsf{cu}}$, $\mathsf{ID} \notin \mathsf{RL}_{\mathsf{t}}$ and $(\mathsf{ID}, \mathsf{t}) \neq (\mathsf{ID}^*, \mathsf{t}^*)$ meet at the same time. If not, it returns $\perp$; otherwise, it simply creates $\mathbf{e}_{\mathsf{ID},\mathsf{t}}$ by combing $(\mathbf{e}_{\mathsf{ID},\theta})_\theta$ and $(\mathbf{e}_{\mathsf{t},\theta})_\theta$. Since $\mathsf{path}(\mathsf{BT}_{\mathsf{kgc}}, \eta_{\mathsf{ID}^*}) \cap \mathsf{KUNodes}(\mathsf{BT}_{\mathsf{kgc}}, \mathsf{RL}_{\mathsf{t}^*}) = \emptyset$, the procedures described above are well-defined. Finally, according to Lemma 1, 2, 4, 5 and 6, the distributions of the short vectors given to $\mathcal{A}_1$ are distributed statistically close to those of the previous game. Therefore, $\mathsf{Game}_{\mathbf{I}\text{-}2}$ and $\mathsf{Game}_{\mathbf{I}\text{-}3}$ are indistinguishable.

$\mathsf{Game}_{\mathbf{I}\text{-}4}$: In this game we change how $\mathbf{A}$ is sampled. We generate $\mathbf{A}$ as a random matrix in $\mathbb{Z}_q^{n \times m}$ instead of generating it by running $\mathsf{Trap}(\cdot)$. By Lemma 3, $\mathsf{Game}_{\mathbf{I}\text{-}3}$ and $\mathsf{Game}_{\mathbf{I}\text{-}4}$ are indistinguishable.

$\mathsf{Game}_{\mathbf{I}\text{-}5}$: In this game we change the way the challenge ciphertext $\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*}$ is created. In this game, when the $\mathsf{Game}_{\mathbf{I}\text{-}5}$ challenger is issued a challenge query on (M_0, M_1) by $\mathcal{A}_1$, the challenger chooses a random bit $b \leftarrow \{0,1\}$, samples $v \leftarrow \mathbb{Z}_q$, $\mathbf{v} \leftarrow \mathbb{Z}_q^n$, $\mathbf{s}', \mathbf{s}'' \leftarrow \mathbb{Z}_q^n$, $\mathbf{x}', \mathbf{x}'' \leftarrow \mathcal{D}_{\mathbb{Z}^m,\alpha q}$, $\mathbf{R}_{21}, \mathbf{R}_{22}, \mathbf{R}_{31}, \mathbf{R}_{32} \leftarrow \{-1,1\}^{m \times m}$ and

sets

$$\begin{aligned}
&\mathbf{c}_0 = v + \mathbf{u}^\top(\mathbf{s}' + \mathbf{s}'') + M_b \cdot \lfloor \frac{q}{2} \rfloor, \quad \mathbf{c}_1 = \begin{bmatrix} \mathbf{v} \\ (\mathbf{R}_{11}^*)^\top \mathbf{v} \\ (\mathbf{R}_{12}^*)^\top \mathbf{v} \end{bmatrix}, \\
&\mathbf{c}_2 = \mathbf{B}_{\mathsf{ID}^*,\mathsf{t}^*}^\top \mathbf{s}' + \begin{bmatrix} \mathbf{x}' \\ \mathbf{R}_{21}^\top \mathbf{x}' \\ \mathbf{R}_{22}^\top \mathbf{x}' \end{bmatrix}, \quad \mathbf{c}_3 = \mathbf{C}_{\mathsf{ID}^*,\mathsf{t}^*}^\top \mathbf{s}'' + \begin{bmatrix} \mathbf{x}'' \\ \mathbf{R}_{31}^\top \mathbf{x}'' \\ \mathbf{R}_{32}^\top \mathbf{x}'' \end{bmatrix}.
\end{aligned} \tag{20}$$

Finally, the challenger outputs the challenge ciphertext as $\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*} = (c_0, \mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$. Since v is distributed uniformly at random over $\mathbb{Z}_q$ and independently of all other terms, the probability of $\mathcal{A}_1$ guessing whether $b = 0$ or $b = 1$ is exactly $1/2$. In the following, we only need to show that $\mathsf{Game}_{\mathbf{I}\text{-}4}$ and $\mathsf{Game}_{\mathbf{I}\text{-}5}$ are indistinguishable assuming the hardness of the $\mathsf{LWE}_{n,m+1,q,\chi}$ to complete the proof. To this end, we use $\mathcal{A}_1$ to construct a LWE adversary $\mathcal{B}_1$ as follows:

$\mathcal{B}_1$ is given the problem instance of LWE as $(\bar{\mathbf{A}}, \bar{\mathbf{v}}) \in \mathbb{Z}_q^{n\times(m+1)} \times \mathbb{Z}_q^{(m+1)}$ and aims to distinguish whether $\bar{\mathbf{v}} = \bar{\mathbf{A}}^\top \mathbf{s} + \bar{\mathbf{x}}$ for some $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\bar{\mathbf{x}} \leftarrow \mathcal{D}_{\mathbb{Z}^{m+1},\alpha q}$ or $\bar{\mathbf{v}} \leftarrow \mathbb{Z}_q^{(m+1)}$. Let the first column of $\bar{\mathbf{A}}$ be $\mathbf{u}^* \in \mathbb{Z}_q^n$ and the remaining columns be $\mathbf{A}^* \in \mathbb{Z}_q^{n\times m}$. Let the first coefficient of $\bar{\mathbf{v}}$ be v and the remaining columns be $\mathbf{v}$. Now, $\mathcal{B}_1$ sets $(\mathbf{A}, \mathbf{u}) = (\mathbf{A}^*, \mathbf{u}^*)$ and proceeds the setup as the $\mathsf{Game}_{\mathbf{I}\text{-}4}$ challenger. Furthermore, whenever $\mathcal{A}_1$ issues a query, $\mathcal{B}_1$ works as the $\mathsf{Game}_{\mathbf{I}\text{-}3}$ challenger and answers them without $\mathbf{T}_\mathbf{A}$. To generate the challenge ciphertext, $\mathcal{B}_1$ chooses $b \leftarrow \{0,1\}$ and generates the challenge ciphertext as in Eq.(20) using v, $\mathbf{v}$, and returns it to $\mathcal{A}_1$. Let b' denote the output of $\mathcal{A}_1$, then $\mathcal{B}_1$ outputs 1 if $b' = b$ and 0 otherwise. Note that if $(\bar{\mathbf{A}}, \bar{\mathbf{v}})$ is a valid LWE sample, i.e., $\bar{\mathbf{v}} = \bar{\mathbf{A}}^\top \mathbf{s} + \bar{\mathbf{x}}$ for some $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, then the view of $\mathcal{A}_1$ is the same as that of $\mathsf{Game}_{\mathbf{I}\text{-}4}$. Otherwise, i.e., $\bar{\mathbf{v}} \leftarrow \mathbb{Z}_q^{(m+1)}$, it is the same as that of $\mathsf{Game}_{\mathbf{I}\text{-}5}$. Therefore, $\mathsf{Game}_{\mathbf{I}\text{-}4}$ and $\mathsf{Game}_{\mathbf{I}\text{-}5}$ are indistinguishable assuming the hardness of the $\mathsf{LWE}_{n,m+1,q,\chi}$, where $\chi = \mathcal{D}_{\mathbb{Z}^{m+1},\alpha q}$. □

Lemma 11. *The advantage of an adversary $\mathcal{A}_2$ using the Type-***II** strategy is negligible assuming the hardness of the $\mathsf{LWE}_{n,m+1,q,\chi}$, where $\chi = \mathcal{D}_{\mathbb{Z}^{m+1},\alpha q}$.

Proof. The outline of this proof is essentially the same as that of Lemma 10. The difference is that in this proof, we modify the challenger so that he is able to simulate the game without $\mathbf{T}_\mathbf{B}$.

$\mathsf{Game}_{\mathbf{II}\text{-}0}$: This is the original SSR-sID-CPA game from Definition 5.

$\mathsf{Game}_{\mathbf{II}\text{-}1}$: In this game, we change the way that the challenger generates $\mathbf{B}_1, \mathbf{B}_2$ in the public parameters. The $\mathsf{Game}_{\mathbf{II}\text{-}1}$ challenger samples $\mathbf{R}_{21}^*, \mathbf{R}_{22}^* \leftarrow \{-1,1\}^{m\times m}$ at the setup phase and sets $\mathbf{B}_1, \mathbf{B}_2$ as

$$\mathbf{B}_1 = \mathbf{B}\mathbf{R}_{21}^* - \mathsf{H}(\mathsf{ID}^*)\mathbf{G}, \quad \mathbf{B}_2 = \mathbf{B}\mathbf{R}_{22}^* - \mathsf{H}(\mathsf{t}^*)\mathbf{G}. \tag{21}$$

Then the challenger keeps the matrices $\mathbf{R}_{21}^*, \mathbf{R}_{22}^*$ as part of $\mathsf{sk}_{\mathsf{kgc}}$. The remainder of the game is unchanged. Similar to the analysis of $\mathsf{Game}_{\mathbf{I}\text{-}1}$ in Lemma 10, by

Lemma 7, $\mathbf{B}_1, \mathbf{B}_2$ in $\mathsf{Game}_{\mathbf{II}\text{-}0}$ and $\mathsf{Game}_{\mathbf{II}\text{-}1}$ are indistinguishable, hence $\mathsf{Game}_{\mathbf{II}\text{-}0}$ is indistinguishable from $\mathsf{Game}_{\mathbf{II}\text{-}1}$.

$\mathsf{Game}_{\mathbf{II}\text{-}2}$: In this game, we change the challenger so he does not have to use the trapdoor $\mathbf{T}_{\mathbf{B}}$ when generating $\mathbf{T}_{\mathbf{B}_{\mathsf{ID}}=[\mathbf{B}|\mathbf{B}_1+\mathsf{H}(\mathsf{ID})\mathbf{G}]=[\mathbf{B}|\mathbf{B}\mathbf{R}^*_{21}+(\mathsf{H}(\mathsf{ID})-\mathsf{H}(\mathsf{ID}^*))\mathbf{G}]}$ in $\mathsf{sk}_{\mathsf{ID}}$ and $\mathbf{g}_{\mathsf{ID},\mathsf{t}}$ in $\mathsf{tk}_{\mathsf{ID},\mathsf{t}}$. To this end, we modify the ways $\mathbf{T}_{\mathbf{B}_{\mathsf{ID}}}$ and $\mathbf{g}_{\mathsf{ID},\mathsf{t}}$ are sampled, respectively. By the definition of the Type-**II** strategy, $\mathcal{A}_2$ does not issue long-term transformation key reveal query on ID^*. Then, when $\mathcal{A}_2$ issues a long-term transformation key query on $\mathsf{ID} \neq \mathsf{ID}^*$, the $\mathsf{Game}_{\mathbf{II}\text{-}2}$ challenger runs $\mathsf{ExtRndRight}(\mathbf{B}, \mathbf{G}, \mathbf{R}^*_{21}, \mathbf{T}_{\mathbf{G}}, s_0)$ to create $\mathbf{T}_{\mathbf{B}_{\mathsf{ID}}}$. By Lemma 4, the distribution of $\mathbf{T}_{\mathbf{B}_{\mathsf{ID}}}$ given to $\mathcal{A}_2$ is distributed statistically close to that of the previous game. Furthermore, when $\mathcal{A}_2$ issues a short-term transformation key reveal query on $(\mathsf{ID}, \mathsf{t})$, the challenger checks whether conditions $\mathsf{t} \leq \mathsf{t}_{\mathsf{cu}}$, $\mathsf{ID} \notin \mathsf{RL}_{\mathsf{t}}$ and $(\mathsf{ID}, \mathsf{t}) \neq (\mathsf{ID}^*, \mathsf{t}^*)$ meet at the same time. If not, it returns $\perp$. Otherwise, if $\mathsf{ID} \neq \mathsf{ID}^*$, the challenger samples $\mathbf{g}_{\mathsf{ID},\mathsf{t}}$ by running $\mathsf{SampleLeft}(\cdot)$ with $\mathbf{T}_{\mathbf{B}_{\mathsf{ID}}}$ and Gaussian parameter s_1. Otherwise $\mathsf{ID} = \mathsf{ID}^*$ and $\mathsf{t} \neq \mathsf{t}^*$, the challenger first runs $\mathsf{ExtRndRight}(\mathbf{B}, \mathbf{G}, \mathbf{R}^*_{22}, \mathbf{T}_{\mathbf{G}}, s_0)$ to create the trapdoor $\mathbf{T}_{\mathbf{B}_{\mathsf{t}}=[\mathbf{B}|\mathbf{B}_2+\mathsf{H}(\mathsf{t})\mathbf{G}]=[\mathbf{B}|\mathbf{B}\mathbf{R}^*_{22}+(\mathsf{H}(\mathsf{t})-\mathsf{H}(\mathsf{t}^*))\mathbf{G}]}$, samples $\mathbf{g}'_{\mathsf{ID}^*,\mathsf{t}}$ by running algorithm $\mathsf{SampleLeft}(\mathbf{B}_{\mathsf{t}}, \mathbf{B}_1 + \mathsf{H}(\mathsf{ID}^*)\mathbf{G}, \mathbf{u}, \mathbf{T}_{\mathbf{B}_{\mathsf{t}}}, s_1)$, and then obtains $\mathbf{g}_{\mathsf{ID}^*,\mathsf{t}}$ by rearranging elements in $\mathbf{g}'_{\mathsf{ID}^*,\mathsf{t}}$. According to Lemma 1, 2, 4, 5 and 6, the distribution of $\mathbf{g}_{\mathsf{ID},\mathsf{t}}$ given to $\mathcal{A}_2$ is distributed statistically close to that of the previous game. Therefore, $\mathsf{Game}_{\mathbf{II}\text{-}1}$ and $\mathsf{Game}_{\mathbf{II}\text{-}2}$ are indistinguishable.

$\mathsf{Game}_{\mathbf{II}\text{-}3}$: In this game we change how $\mathbf{B}$ is sampled. We generate $\mathbf{B}$ as a random matrix in $\mathbb{Z}_q^{n\times m}$ instead of generating it by running $\mathsf{Trap}(\cdot)$. By Lemma 3, $\mathsf{Game}_{\mathbf{II}\text{-}2}$ and $\mathsf{Game}_{\mathbf{II}\text{-}3}$ are indistinguishable.

$\mathsf{Game}_{\mathbf{II}\text{-}4}$: In this game we change the way the challenge ciphertext $\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*}$ is created. When the $\mathsf{Game}_{\mathbf{II}\text{-}4}$ challenger is issued a challenge query on (M_0, M_1) by $\mathcal{A}_2$, the challenger chooses a random bit $b \leftarrow \{0,1\}$, samples $v \leftarrow \mathbb{Z}_q$, $\mathbf{v} \leftarrow \mathbb{Z}_q^n$, $\mathbf{s}, \mathbf{s}'' \leftarrow \mathbb{Z}_q^n$, $\mathbf{x}, \mathbf{x}'' \leftarrow \mathcal{D}_{\mathbb{Z}^m,\alpha q}$, $\mathbf{R}_{11}, \mathbf{R}_{12}, \mathbf{R}_{31}, \mathbf{R}_{32} \leftarrow \{-1,1\}^{m\times m}$ and sets

$$\begin{aligned} &\mathbf{c}_0 = v + \mathbf{u}^\top(\mathbf{s}+\mathbf{s}'') + M_b \cdot \lfloor \frac{q}{2} \rfloor, \quad \mathbf{c}_1 = \mathbf{A}^\top_{\mathsf{ID}^*,\mathsf{t}^*}\mathbf{s} + \begin{bmatrix} \mathbf{x} \\ \mathbf{R}_{11}^\top\mathbf{x} \\ \mathbf{R}_{12}^\top\mathbf{x} \end{bmatrix}, \\ &\mathbf{c}_2 = \begin{bmatrix} \mathbf{v} \\ (\mathbf{R}^*_{21})^\top\mathbf{v} \\ (\mathbf{R}^*_{22})^\top\mathbf{v} \end{bmatrix}, \qquad \mathbf{c}_3 = \mathbf{C}^\top_{\mathsf{ID}^*,\mathsf{t}^*}\mathbf{s}'' + \begin{bmatrix} \mathbf{x}'' \\ \mathbf{R}_{31}^\top\mathbf{x}'' \\ \mathbf{R}_{32}^\top\mathbf{x}'' \end{bmatrix}. \end{aligned} \tag{22}$$

Finally, the challenger outputs the challenge ciphertext as $\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*} = (c_0, \mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$. Since v is distributed uniformly at random over $\mathbb{Z}_q$ and independently of all other terms, the probability of $\mathcal{A}_2$ guessing whether $b = 0$ or $b = 1$ is exactly $1/2$. To complete the proof, we only need to show that $\mathsf{Game}_{\mathbf{II}\text{-}3}$ and $\mathsf{Game}_{\mathbf{II}\text{-}4}$ are indistinguishable assuming the hardness of the $\mathsf{LWE}_{n,m+1,q,\chi}$. This process is similar to that of $\mathsf{Game}_{\mathbf{I}\text{-}5}$ in Lemma 10, so we omit it here. □

Lemma 12. *The advantage of an adversary $\mathcal{A}_3$ using the Type-***III** strategy is negligible assuming the hardness of the $\mathsf{LWE}_{n,m+1,q,\chi}$, where $\chi = \mathcal{D}_{\mathbb{Z}^{m+1},\alpha q}$.

Proof. The outline of this proof is essentially the same as that of Lemma 10. The difference is that in this proof, we modify the challenger so that he is able to simulate the game without $\mathbf{T_C}$.

$\mathsf{Game}_{\mathbf{III}\text{-}0}$: This is the original SSR-sID-CPA game from Definition 5.

$\mathsf{Game}_{\mathbf{III}\text{-}1}$: In this game, we change the way that the challenger generates $\mathbf{C}_1, \mathbf{C}_2$ in the public parameters. The $\mathsf{Game}_{\mathbf{III}\text{-}1}$ challenger samples $\mathbf{R}^*_{31}, \mathbf{R}^*_{32} \leftarrow \{-1,1\}^{m\times m}$ at the setup phase and sets $\mathbf{C}_1, \mathbf{C}_2$ as

$$\mathbf{C}_1 = \mathbf{C}\mathbf{R}^*_{31} - \mathsf{H}(\mathsf{ID}^*)\mathbf{G}, \quad \mathbf{C}_2 = \mathbf{C}\mathbf{R}^*_{32} - \mathsf{H}(\mathsf{t}^*)\mathbf{G}. \tag{23}$$

Then the challenger keeps the matrices $\mathbf{R}^*_{31}, \mathbf{R}^*_{32}$ as part of $\mathsf{sk}_{\mathsf{kgc}}$. The remainder of the game is unchanged. Similar to $\mathsf{Game}_{\mathbf{I}\text{-}1}$ in Lemma 10, by Lemma 7, $\mathbf{C}_1, \mathbf{C}_2$ in $\mathsf{Game}_{\mathbf{III}\text{-}0}$ and $\mathsf{Game}_{\mathbf{III}\text{-}1}$ are indistinguishable, hence $\mathsf{Game}_{\mathbf{III}\text{-}0}$ is indistinguishable from $\mathsf{Game}_{\mathbf{III}\text{-}1}$.

$\mathsf{Game}_{\mathbf{III}\text{-}2}$: In this game, we change the challenger so he does not have to use the trapdoor $\mathbf{T_C}$ when generating $\mathbf{T}_{\mathbf{C}_{\mathsf{ID}}=[\mathbf{C}|\mathbf{C}_1+\mathsf{H}(\mathsf{ID})\mathbf{G}]=[\mathbf{C}|\mathbf{C}\mathbf{R}^*_{31}+(\mathsf{H}(\mathsf{ID})-\mathsf{H}(\mathsf{ID}^*))\mathbf{G}]}$ in $\mathsf{Priv}_{\mathsf{ID}}$ and $\mathbf{d}_{\mathsf{ID},\mathsf{t}}$ in $\mathsf{dk}_{\mathsf{ID},\mathsf{t}}$. To this end, we change how $\mathbf{T}_{\mathbf{C}_{\mathsf{ID}}}$ and $\mathbf{d}_{\mathsf{ID},\mathsf{t}}$ are sampled. By the definitions of the Type-**III** strategy and the SSR-sID-CPA security, $\mathcal{A}_3$ does not issue private key reveal query on ID^* and only issues decryption key reveal queries on $(\mathsf{ID},\mathsf{t}) \neq (\mathsf{ID}^*,\mathsf{t}^*)$ (since $\mathsf{ID}^* \notin \mathsf{RL}_{\mathsf{t}^*}$). Then, when $\mathcal{A}_3$ issues a private key query on $\mathsf{ID} \neq \mathsf{ID}^*$, the $\mathsf{Game}_{\mathbf{III}\text{-}2}$ challenger runs $\mathsf{ExtRndRight}(\mathbf{C}, \mathbf{G}, \mathbf{R}^*_{31}, \mathbf{T_G}, s_0)$ to create $\mathbf{T}_{\mathbf{C}_{\mathsf{ID}}}$. By Lemma 4, the distribution of $\mathbf{T}_{\mathbf{C}_{\mathsf{ID}}}$ given to $\mathcal{A}_3$ is distributed statistically close to that of the previous game. Furthermore, when $\mathcal{A}_3$ issues a decryption key reveal query on $(\mathsf{ID},\mathsf{t}) \neq (\mathsf{ID}^*,\mathsf{t}^*)$, if $\mathsf{ID} \neq \mathsf{ID}^*$, the challenger samples $\mathbf{d}_{\mathsf{ID},\mathsf{t}}$ by running $\mathsf{SampleLeft}(\cdot)$ with $\mathbf{T}_{\mathbf{C}_{\mathsf{ID}}}$ and Gaussian parameter s_1. Otherwise $\mathsf{ID} = \mathsf{ID}^*$ and $\mathsf{t} \neq \mathsf{t}^*$, the challenger first runs $\mathsf{ExtRndRight}(\mathbf{C}, \mathbf{G}, \mathbf{R}^*_{32}, \mathbf{T_G}, s_0)$ to create the trapdoor $\mathbf{T}_{\mathbf{C}_{\mathsf{t}}=[\mathbf{C}|\mathbf{C}_2+\mathsf{H}(\mathsf{t})\mathbf{G}]=[\mathbf{C}|\mathbf{C}\mathbf{R}^*_{32}+(\mathsf{H}(\mathsf{t})-\mathsf{H}(\mathsf{t}^*))\mathbf{G}]}$, samples $\mathbf{d}'_{\mathsf{ID}^*,\mathsf{t}}$ by running algorithm $\mathsf{SampleLeft}(\mathbf{C}_{\mathsf{t}}, \mathbf{C}_1 + \mathsf{H}(\mathsf{ID}^*)\mathbf{G}, \mathbf{u}, \mathbf{T}_{\mathbf{C}_{\mathsf{t}}}, s_1)$, and then obtains $\mathbf{d}_{\mathsf{ID}^*,\mathsf{t}}$ by rearranging elements in $\mathbf{d}'_{\mathsf{ID}^*,\mathsf{t}}$. According to Lemma 1, 2, 4, 5 and 6, the distribution of $\mathbf{d}_{\mathsf{ID},\mathsf{t}}$ given to $\mathcal{A}_3$ is distributed statistically close to that of the previous game. Therefore, $\mathsf{Game}_{\mathbf{III}\text{-}1}$ and $\mathsf{Game}_{\mathbf{III}\text{-}2}$ are indistinguishable.

$\mathsf{Game}_{\mathbf{III}\text{-}3}$: In this game we change how $\mathbf{C}$ is sampled. We generate $\mathbf{C}$ as a random matrix in $\mathbb{Z}_q^{n\times m}$ instead of generating it by running $\mathsf{Trap}(\cdot)$. By Lemma 3, $\mathsf{Game}_{\mathbf{III}\text{-}2}$ and $\mathsf{Game}_{\mathbf{III}\text{-}3}$ are indistinguishable.

$\mathsf{Game}_{\mathbf{III}\text{-}4}$: In this game we change the way the challenge ciphertext $\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*}$ is created. When the $\mathsf{Game}_{\mathbf{III}\text{-}4}$ challenger is issued a challenge query on (M_0, M_1) by $\mathcal{A}_3$, the challenger chooses a random bit $b \leftarrow \{0,1\}$, samples $v \leftarrow \mathbb{Z}_q$, $\mathbf{v} \leftarrow \mathbb{Z}_q^n$,

$\mathbf{s}, \mathbf{s}' \leftarrow \mathbb{Z}_q^n$, $\mathbf{x}, \mathbf{x}' \leftarrow \mathcal{D}_{\mathbb{Z}^m, \alpha q}$, $\mathbf{R}_{11}, \mathbf{R}_{12}, \mathbf{R}_{21}, \mathbf{R}_{22} \leftarrow \{-1, 1\}^{m \times m}$ and sets

$$\begin{aligned} \mathbf{c}_0 &= v + \mathbf{u}^\top(\mathbf{s} + \mathbf{s}') + M_b \cdot \lfloor \frac{q}{2} \rfloor, \quad \mathbf{c}_1 = \mathbf{A}_{\mathsf{ID}^*, \mathsf{t}^*}^\top \mathbf{s} + \begin{bmatrix} \mathbf{x} \\ \mathbf{R}_{11}^\top \mathbf{x} \\ \mathbf{R}_{12}^\top \mathbf{x} \end{bmatrix}, \\ \mathbf{c}_2 &= \mathbf{B}_{\mathsf{ID}^*, \mathsf{t}^*}^\top \mathbf{s}' + \begin{bmatrix} \mathbf{x}' \\ \mathbf{R}_{21}^\top \mathbf{x}' \\ \mathbf{R}_{22}^\top \mathbf{x}' \end{bmatrix}, \quad \mathbf{c}_3 = \begin{bmatrix} \mathbf{v} \\ (\mathbf{R}_{31}^*)^\top \mathbf{v} \\ (\mathbf{R}_{32}^*)^\top \mathbf{v} \end{bmatrix}. \end{aligned} \tag{24}$$

Finally, the challenger outputs the challenge ciphertext as $\mathsf{ct}_{\mathsf{ID}^*, \mathsf{t}^*} = (c_0, \mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$. Since v is distributed uniformly at random over $\mathbb{Z}_q$ and independently of all other terms, the probability of $\mathcal{A}_3$ guessing whether $b = 0$ or $b = 1$ is exactly $1/2$. To complete the proof, we only need to show that $\mathsf{Game}_{\mathbf{III}\text{-}3}$ and $\mathsf{Game}_{\mathbf{III}\text{-}4}$ are indistinguishable assuming the hardness of the $\mathsf{LWE}_{n,m+1,q,\chi}$. This process is similar to that of $\mathsf{Game}_{\mathbf{I}\text{-}5}$ in Lemma 10, so we omit it here. □

According to Lemma 10, 11 and 12, we can conclude that the SR-IBE scheme is SSR-sID-CPA secure, which completes the proof of Theorem 1. □

5 Generic SR-IBE scheme

In this section, we propose a generic construction of SR-IBE scheme by employing a RIBE with DKER scheme and a 2-level HIBE scheme, and prove that it is SSR-sID-CPA (*resp.* SSR-aID-CPA) secure if the underlying RIBE is selective-identity (*resp.* adaptive-identity) secure and the underlying 2-level HIBE is selective-identity (*resp.* adaptive-identity) secure.

Let $\mathsf{r}.\Pi = (\mathsf{r.Setup}, \mathsf{r.GenSK}, \mathsf{r.KeyUP}, \mathsf{r.GenDK}, \mathsf{r.Encrypt}, \mathsf{r.Decrypt})$ be a RIBE (with DKER) scheme with identity space $\mathsf{r}.\mathcal{ID}$, message space $\mathsf{r}.\mathcal{M}$ and time period space $\mathsf{r}.\mathcal{T}$. Let $\mathsf{h}.\Pi = (\mathsf{h.Setup}, \mathsf{h.GenSK}, \mathsf{h.Delegate}, \mathsf{h.Encrypt}, \mathsf{h.Decrypt})$ be a 2-level HIBE scheme with identity space $\mathsf{h}.\mathcal{ID}$, message space $\mathsf{h}.\mathcal{M}$. Let $\mathsf{r}.\mathcal{ID} = \mathsf{h}.\mathcal{ID}$, $\mathsf{r}.\mathcal{M} = \mathsf{h}.\mathcal{M}$ and $\mathsf{r}.\mathcal{T} \subset \mathsf{h}.\mathcal{ID}$. We assume that the message space is finite and forms an abelian group with the addition "+" as the group operation.

In the following, using the RIBE with DKER scheme and the 2-level HIBE scheme described above as building blocks, we construct a SR-IBE scheme $\Pi = (\mathsf{Setup}, \mathsf{L\text{-}TranKG}, \mathsf{UpdKG}, \mathsf{S\text{-}TranKG}, \mathsf{PrivKG}, \mathsf{DecKG}, \mathsf{Enc}, \mathsf{Dec})$ with identity space $\mathcal{ID} = \mathsf{r}.\mathcal{ID} = \mathsf{h}.\mathcal{ID}$, message space $\mathcal{M} = \mathsf{r}.\mathcal{M} = \mathsf{h}.\mathcal{M}$ and time period space $\mathcal{T} = \mathsf{r}.\mathcal{T} \subset \mathsf{h}.\mathcal{ID}$.

$\mathsf{Setup}(1^\lambda)$**:** On input the security parameter 1^λ, it runs

$$(\mathsf{r.pp}, \mathsf{r.sk_{kgc}}) \leftarrow \mathsf{r.Setup}(1^\lambda), \quad (\mathsf{h.pp}, \mathsf{h.sk_{kgc}}) \leftarrow \mathsf{h.Setup}(1^\lambda). \tag{25}$$

Then it outputs $\mathsf{pp} := (\mathsf{r.pp}, \mathsf{h.pp})$ and $\mathsf{sk_{kgc}} := (\mathsf{r.sk_{kgc}}, \mathsf{h.sk_{kgc}})$. We assume that pp is an implicit input of all other algorithms.

$\mathsf{L\text{-}TranKG}(\mathsf{sk_{kgc}}, \mathsf{ID})$**:** On input the KGC's secret key $\mathsf{sk_{kgc}}$, an identity $\mathsf{ID} \in \mathcal{ID}$, it runs

$$(\mathsf{r.sk_{ID}}, \mathsf{r.sk'_{kgc}}) \leftarrow \mathsf{r.GenSK}(\mathsf{r.pp}, \mathsf{r.sk_{kgc}}, \mathsf{ID}). \tag{26}$$

Then it outputs a long-term transformation key $\mathsf{sk_{ID}} := \mathsf{r.sk_{ID}}$ for ID and the KGC's "updated" state $\mathsf{sk'_{kgc}} := (\mathsf{r.sk'_{kgc}}, \mathsf{h.sk_{kgc}})$.

$\mathsf{UpdKG}(\mathsf{sk_{kgc}}, \mathsf{t}, \mathsf{RL_t})$**:** On input the KGC's secret key $\mathsf{sk_{kgc}} = (\mathsf{r.sk_{kgc}}, \mathsf{h.sk_{kgc}})$, a time period $\mathsf{t} \in \mathcal{T}$, a revocation list $\mathsf{RL_t} \subset \mathcal{ID}$, it runs

$$(\mathsf{r.uk_t}, \mathsf{r.sk'_{kgc}}) \leftarrow \mathsf{r.KeyUP}(\mathsf{r.pp}, \mathsf{r.sk_{kgc}}, \mathsf{t}, \mathsf{RL_t}). \tag{27}$$

Then it outputs an update key $\mathsf{uk_t} := \mathsf{r.uk_t}$ and the "updated" state $\mathsf{sk'_{kgc}} := (\mathsf{r.sk'_{kgc}}, \mathsf{h.sk_{kgc}})$.

$\mathsf{S\text{-}TranKG}(\mathsf{sk_{ID}}, \mathsf{uk_t})$**:** On input a long-term transformation key $\mathsf{sk_{ID}} = (\mathsf{r.sk_{ID}}, \mathsf{h.sk_{ID}})$ for identity $\mathsf{ID} \in \mathcal{ID}$, an update key $\mathsf{uk_t} = \mathsf{r.uk_t}$ for time period $\mathsf{t} \in \mathcal{T}$, it runs

$$\mathsf{r.dk_{ID,t}} \leftarrow \mathsf{r.GenDK}(\mathsf{r.pp}, \mathsf{r.sk_{ID}}, \mathsf{r.uk_t}). \tag{28}$$

Then it outputs a short-term transformation key $\mathsf{tk_{ID,t}} := \mathsf{r.dk_{ID,t}}$ for time period t, except that if $\mathsf{r.dk_{ID,t}} = \perp$, then it returns the special symbol $\perp$ indicating that ID has been revoked.

$\mathsf{PrivKG}(\mathsf{sk_{kgc}}, \mathsf{ID})$**:** On input the KGC's secret key $\mathsf{sk_{ID}} = (\mathsf{r.sk_{ID}}, \mathsf{h.sk_{ID}})$ and an identity $\mathsf{ID} \in \mathcal{ID}$, it runs

$$\mathsf{h.sk_{ID}} \leftarrow \mathsf{h.GenSK}(\mathsf{h.pp}, \mathsf{h.sk_{kgc}}, \mathsf{ID}). \tag{29}$$

Then it outputs a private key $\mathsf{Priv_{ID}} := \mathsf{h.sk_{ID}}$.

$\mathsf{DecKG}(\mathsf{Priv_{ID}}, \mathsf{t})$**:** On input a private key $\mathsf{Priv_{ID}} = \mathsf{h.sk_{ID}}$ and a time period $\mathsf{t} \in \mathcal{T}$, it runs

$$\mathsf{h.sk_{ID,t}} \leftarrow \mathsf{h.Delegate}(\mathsf{h.pp}, \mathsf{h.sk_{ID}}, \mathsf{t}). \tag{30}$$

Then it outputs a decryption key $\mathsf{dk_{ID,t}} := \mathsf{h.sk_{ID,t}}$.

$\mathsf{Enc}(\mathsf{ID}, \mathsf{t}, M)$**:** On input an identity $\mathsf{ID} \in \mathcal{ID}$, a time period $\mathsf{t} \in \mathcal{T}$, a message $M \in \mathcal{M}$, it samples $(\mathsf{r}.M, \mathsf{h}.M) \in \mathcal{M}^2$ uniformly at random, such that

$$\mathsf{r}.M + \mathsf{h}.M = M. \tag{31}$$

Then, it runs

$$\mathsf{r.ct_{ID,t}} \leftarrow \mathsf{r.Encrypt}(\mathsf{r.pp}, \mathsf{ID}, \mathsf{t}, \mathsf{r}.M), \quad \mathsf{h.ct_{ID,t}} \leftarrow \mathsf{h.Encrypt}(\mathsf{h.pp}, (\mathsf{ID}, \mathsf{t}), \mathsf{h}.M). \tag{32}$$

Finally, it outputs a ciphertext $\mathsf{ct_{ID,t}} := (\mathsf{r.ct_{ID,t}}, \mathsf{h.ct_{ID,t}})$.

$\mathsf{Transform}(\mathsf{tk_{ID,t}}, \mathsf{ct_{ID,t}})$**:** On input a short-term transformation key $\mathsf{tk_{ID,t}} = \mathsf{r.dk_{ID,t}}$, a ciphertext $\mathsf{ct_{ID,t}} = (\mathsf{r.ct_{ID,t}}, \mathsf{h.ct_{ID,t}})$, it runs

$$\mathsf{r}.M \leftarrow \mathsf{r.Decrypt}(\mathsf{r.pp}, \mathsf{r.dk_{ID,t}}, \mathsf{r.ct_{ID,t}}) \tag{33}$$

and outputs a partially decrypted ciphertext $\mathsf{ct'_{ID,t}} := (\mathsf{r}.M, \mathsf{h.ct_{ID,t}})$.

$\mathsf{Dec}(\mathsf{dk}_{\mathsf{ID},\mathsf{t}}, \mathsf{ct}'_{\mathsf{ID},\mathsf{t}})$**:** On input a decryption key $\mathsf{dk}_{\mathsf{ID},\mathsf{t}} = \mathsf{h.sk}_{\mathsf{ID},\mathsf{t}}$, a partially decrypted ciphertext $\mathsf{ct}'_{\mathsf{ID},\mathsf{t}} = (\mathsf{r}.M, \mathsf{h.ct}_{\mathsf{ID},\mathsf{t}})$, it runs

$$\mathsf{h}.M \leftarrow \mathsf{h.Decrypt}(\mathsf{h.pp}, \mathsf{h.sk}_{\mathsf{ID},\mathsf{t}}, \mathsf{h.ct}_{\mathsf{ID},\mathsf{t}}). \tag{34}$$

Then it outputs a message $M = \mathsf{r}.M + \mathsf{h}.M$ except that if $\mathsf{r}.M = \perp$, it returns the special symbol $\perp$.

5.1 Correctness

The correctness of the SR-IBE scheme constructed above follows from the correctness of the underlying RIBE with DKER scheme and the underlying 2-level HIBE scheme.

5.2 Security

Theorem 2. *If the underlying RIBE scheme* $\mathsf{r}.\Pi$ *satisfies selective-identity (*resp. *adaptive-identity) security and the underlying 2-level HIBE scheme* $\mathsf{h}.\Pi$ *satisfies selective-identity (*resp. *adaptive-identity) security, then the resulting SR-IBE scheme* Π *satisfies SSR-sID-CPA (*resp. *SSR-aID-CPA) security.*

Proof. We only show the proof for SSR-sID-CPA security, the proof for SSR-aID-CPA security is similar. Let $\mathcal{A}$ be a PPT adversary who succeeds in breaking the SSR-sID-CPA security of our generic SR-IBE scheme Π, let ID^* be the challenge identity and t^* be the challenge time period. Observe that the strategy taken by $\mathcal{A}$ can be divided into two types of strategies that are mutually exclusive, where the first type can be further divided into two types of strategies that are mutually exclusive.

Type-I: $\mathcal{A}$ issues private key reveal query on the challenge identity ID^*.
- **Type-I-1:** $\mathcal{A}$ issues long-term transformation key reveal query on the challenge identity ID^* as well. In this case, the challenge identity ID^* must be revoked before the challenge time t^*.
- **Type-I-2:** $\mathcal{A}$ does not issue long-term transformation key reveal query on the challenge identity ID^*.

Type-II: $\mathcal{A}$ doesn't issue private key reveal query on the challenge identity ID^*.

By Lemma 8, $\mathcal{A}$ always follows one of the above strategies, and we only need to show that the advantage of $\mathcal{A}$ is negligible regardless of the strategy taken by $\mathcal{A}$. We will provide two types of security proofs according to the two types of strategies taken by $\mathcal{A}$ as described above to complete our proof in the following.

Lemma 13. *For any PPT Type-**I** adversary* $\mathcal{A}_1$*, there exists a PPT adversary* $\mathcal{B}_1$ *against the selective-identity security of the underlying RIBE scheme* $\mathsf{r}.\Pi$ *such that* $\mathsf{Adv}^{\mathsf{SSR\text{-}sID\text{-}CPA}}_{\mathcal{A}_1,\mathcal{O}^{SR\text{-}IBE}}(\lambda) = \mathsf{Adv}^{\mathsf{RIBE\text{-}sel}}_{\mathcal{B}_1,\mathcal{O}^{RIBE}}(\lambda)$.

Proof. Let $\mathcal{C}_1$ be the challenger in the experiment $\mathsf{Exp}^{\mathsf{RIBE\text{-}sel}}_{\mathcal{B}_1,\mathcal{O}^{\mathsf{RIBE}}}(\lambda)$ for the underling RIBE scheme, then the adversary $\mathcal{B}_1$ interacts with $\mathcal{A}_1$ and $\mathcal{C}_1$ as follows.

Initial: $\mathcal{A}_1$ announces to $\mathcal{B}_1$ the challenge ID^* and t^*, and the latter forwards them to $\mathcal{C}_1$.

Setup: $\mathcal{B}_1$ receives the public parameter $\mathsf{r.pp}$ and the update key $\mathsf{r.uk}_1$ from $\mathcal{C}_1$. Then $\mathcal{B}_1$ runs $(\mathsf{h.pp}, \mathsf{h.sk}_{\mathsf{kgc}}) \leftarrow \mathsf{h.Setup}(1^\lambda)$, sends $\mathsf{pp} := (\mathsf{r.pp}, \mathsf{h.pp})$ and $\mathsf{uk}_1 := \mathsf{r.uk}_1$ to $\mathcal{A}_1$, and keeps $\mathsf{h.sk}_{\mathsf{kgc}}$ secret. Also, $\mathcal{B}_1$ initializes the counter $\mathsf{t}_{\mathsf{cu}} := 1$ (which will always be synchronized by the one maintained by $\mathcal{C}_1$), generates two empty lists SKList and $\mathsf{PrivList}$.

Long-term Transformation Key Generation Query: For a the long-term transformation key generation query ID from the adversary $\mathcal{A}_1$ (where it is required that $(\mathsf{ID}, *) \notin \mathsf{SKList}$), $\mathcal{B}_1$ makes a secret key generation query ID to $\mathcal{C}_1$ (note that upon this query, $\mathcal{C}_1$ runs $(\mathsf{r.sk}_{\mathsf{ID}}, \mathsf{r.sk}'_{\mathsf{kgc}}) \leftarrow \mathsf{r.GenSK}(\mathsf{r.pp}, \mathsf{r.sk}_{\mathsf{kgc}}, \mathsf{ID})$ and returns nothing to $\mathcal{B}_1$). Then $\mathcal{B}_1$ goes as follows according to the strategy of $\mathcal{A}_1$:

- If $\mathcal{A}_1$ follows Type-**I**-1 strategy, then $\mathcal{B}_1$ further makes a secret key reveal query ID to $\mathcal{C}_1$, and receives $\mathsf{r.sk}_{\mathsf{ID}}$ from $\mathcal{C}_1$. Next, $\mathcal{B}_1$ sets $\mathsf{sk}_{\mathsf{ID}} := \mathsf{r.sk}_{\mathsf{ID}}$, and adds $(\mathsf{ID}, \mathsf{sk}_{\mathsf{ID}})$ into the list SKList.
- If $\mathcal{A}_1$ follows Type-**I**-2 strategy, then $\mathcal{B}_1$ proceeds as above except for $\mathsf{ID} = \mathsf{ID}^*$. For $\mathsf{ID} = \mathsf{ID}^*$, $\mathcal{B}_1$ does not make the secret reveal query ID^* to $\mathcal{C}_1$, and stores nothing in SKList.

Finally, $\mathcal{B}_1$ returns nothing to $\mathcal{A}_1$.

Long-term Transformation Key Reveal Query: For a long-term transformation key reveal query ID made by Type-**I** adversary $\mathcal{A}_1$, it is easy to check that the following condition is satisfied:

- If $\mathsf{t}_{\mathsf{cu}} \geq \mathsf{t}^*$ and $\mathsf{ID}^* \notin \mathsf{RL}_{\mathsf{t}^*}$, then $\mathsf{ID} \neq \mathsf{ID}^*$.

This is because for Type-**I**-1 adversary, ID^* must be revoked before t^*, and for Type-**I**-2 adversary, $\mathsf{ID} \neq \mathsf{ID}^*$ is satisfied. Therefore, the check does by the challenger $\mathcal{C}_1$ for the secret key reveal query ID is always satisfied. This guarantees that $(\mathsf{ID}, \mathsf{sk}_{\mathsf{ID}})$ is contained in the list SKList and thus $\mathcal{B}_1$ returns $\mathsf{sk}_{\mathsf{ID}}$ to $\mathcal{A}_1$.

Revoke & Update Key Query: For a revoke & update key query $\mathsf{RL} \subset \mathcal{ID}$ from $\mathcal{A}_1$, $\mathcal{B}_1$ checks whether $\mathsf{RL}_{\mathsf{t}_{\mathsf{cu}}} \subset \mathsf{RL}$. If not, $\mathcal{B}_1$ returns $\perp$ to $\mathcal{A}_1$. Otherwise, $\mathcal{B}_1$ forwards RL to $\mathcal{C}_1$. Note that upon this query, the checks performed by $\mathcal{C}_1$ listed below are satisfied simultaneously:

- $\mathsf{RL}_{\mathsf{t}_{\mathsf{cu}}} \subset \mathsf{RL}$.
- If $\mathsf{t}_{\mathsf{cu}} = \mathsf{t}^* - 1$ and $\mathsf{r.sk}_{\mathsf{ID}^*}$ for the challenge ID^* has been revealed to $\mathcal{B}_1$ via a secret key reveal query ID^*, then $\mathsf{ID}^* \in \mathsf{RL}$.

This is because for Type-**I**-1 adversary, ID^* must be revoked before t^*, and for Type-**I**-2 adversary, $\mathsf{r.sk}_{\mathsf{ID}^*}$ has never been revealed to $\mathcal{B}_1$ via a secret key reveal query ID^*. Therefore, $\mathcal{C}_1$ increments the current time period by $\mathsf{t}_{\mathsf{cu}} \leftarrow \mathsf{t}_{\mathsf{cu}} + 1$ and returns $\mathsf{r.uk}_{\mathsf{t}_{\mathsf{cu}}}$ to $\mathcal{B}_1$. Finally, $\mathcal{B}_1$ does the same increment for t_{cu} (which ensures that the counter t_{cu} maintained by $\mathcal{C}_1$ and that maintained by $\mathcal{B}_1$ are synchronized) and returns $\mathsf{uk}_{\mathsf{t}_{\mathsf{cu}}} := \mathsf{r.uk}_{\mathsf{t}_{\mathsf{cu}}}$ to $\mathcal{A}_1$.

Short-term Transformation Key Reveal Query: For a short-term transformation key reveal query $(\mathsf{ID}, \mathsf{t})$ from $\mathcal{A}_1$, $\mathcal{B}_1$ checks whether conditions $\mathsf{t} \leq \mathsf{t}_{\mathsf{cu}}$, $\mathsf{ID} \notin \mathsf{RL}_{\mathsf{t}}$ and $(\mathsf{ID}, \mathsf{t}) \neq (\mathsf{ID}^*, \mathsf{t}^*)$ meet at the same time. If not, $\mathcal{B}_1$ returns $\perp$ to $\mathcal{A}_1$. Otherwise, $\mathcal{B}_1$ forwards $(\mathsf{ID}, \mathsf{t})$ to $\mathcal{C}_1$ (note that upon this

query, the checks performed by $\mathcal{C}_1$ for the decryption key reveal query $(\mathsf{ID}, \mathsf{t})$ are satisfied because $\mathcal{C}_1$ does the same checks as $\mathcal{B}_1$) and receives $\mathsf{r.dk}_{\mathsf{ID},\mathsf{t}}$ from $\mathcal{C}_1$. Finally, $\mathcal{B}_1$ returns $\mathsf{tk}_{\mathsf{ID},\mathsf{t}} := \mathsf{r.dk}_{\mathsf{ID},\mathsf{t}}$ to $\mathcal{A}_1$.

Private Key Generation Query: For a private key generation query ID from $\mathcal{A}_1$ (where it its required that $(\mathsf{ID}, *) \notin \mathsf{PrivList}$), $\mathcal{B}_1$ runs $\mathsf{h.sk}_{\mathsf{ID}} \leftarrow \mathsf{h.GenSK}(\mathsf{h.pp}, \mathsf{h.sk}_{\mathsf{kgc}}, \mathsf{ID})$, sets $\mathsf{Priv}_{\mathsf{ID}} := \mathsf{h.sk}_{\mathsf{ID}}$, adds $(\mathsf{ID}, \mathsf{Priv}_{\mathsf{ID}})$ into the list $\mathsf{PrivList}$ and returns nothing to $\mathcal{A}_1$.

Private Key Reveal Query: For a private key reveal query ID from $\mathcal{A}_1$, $\mathcal{B}_1$ finds $\mathsf{Priv}_{\mathsf{ID}}$ from $\mathsf{PrivList}$ and returns it to $\mathcal{A}_1$.

Decryption Key Reveal Query: For a decryption key reveal query $(\mathsf{ID}, \mathsf{t})$ from $\mathcal{A}_1$ (where it is required that $((\mathsf{ID}, \mathsf{t}), *) \notin \mathsf{PrivList}$), $\mathcal{B}_1$ finds $\mathsf{Priv}_{\mathsf{ID}} = \mathsf{h.sk}_{\mathsf{ID}}$ from $\mathsf{PrivList}$ (it is required that ID is "activated" and hence $(\mathsf{ID}, \mathsf{Priv}_{\mathsf{ID}}) \in \mathsf{PrivList}$), runs $\mathsf{h.sk}_{\mathsf{ID},\mathsf{t}} \leftarrow \mathsf{h.Delegate}(\mathsf{h.pp}, \mathsf{h.sk}_{\mathsf{ID}}, \mathsf{t})$, sets $\mathsf{dk}_{\mathsf{ID},\mathsf{t}} := \mathsf{h.sk}_{\mathsf{ID},\mathsf{t}}$, adds $((\mathsf{ID}, \mathsf{t}), \mathsf{dk}_{\mathsf{ID},\mathsf{t}})$ into the list $\mathsf{PrivList}$ and returns $\mathsf{dk}_{\mathsf{ID},\mathsf{t}}$ to $\mathcal{A}_1$.

Challenge: Upon the challenge (M_0, M_1) from $\mathcal{A}_1$, $\mathcal{B}_1$ picks $\mathsf{h}.M \leftarrow \mathcal{M}$ uniformly at random, and then sets $\mathsf{r}.M_0 \leftarrow M_0 - \mathsf{h}.M$, $\mathsf{r}.M_1 \leftarrow M_1 - \mathsf{h}.M$. Then, $\mathcal{B}_1$ sends the challenge $(\mathsf{r}.M_0, \mathsf{r}.M_1)$ to $\mathcal{C}_1$ and receives $\mathcal{B}_1$'s challenge ciphertext $\mathsf{r.ct}_{\mathsf{ID}^*,\mathsf{t}^*} \leftarrow \mathsf{r.Encrypt}(\mathsf{r.pp}, \mathsf{ID}^*, \mathsf{t}^*, \mathsf{r}.M_b)$ from $\mathcal{C}_1$, where $b \leftarrow \{0, 1\}$ is $\mathcal{B}_1$'s challenge bit. Next, $\mathcal{B}_1$ runs $\mathsf{h.ct}_{\mathsf{ID}^*,\mathsf{t}^*} \leftarrow \mathsf{h.Encrypt}(\mathsf{h.pp}, (\mathsf{ID}^*, \mathsf{t}^*), \mathsf{h}.M)$ and returns the challenge ciphertext $\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*} := (\mathsf{r.ct}_{\mathsf{ID}^*,\mathsf{t}^*}, \mathsf{h.ct}_{\mathsf{ID}^*,\mathsf{t}^*})$ to $\mathcal{A}_1$.

Guess: A_1 outputs its guess bit b', $\mathcal{B}_1$ outputs b' as the guess of b.

Note that $\mathcal{B}_1$ perfectly simulates the SSR-sID-CPA security game for the Type-**I** adversary $\mathcal{A}_1$, so that the challenge bit b of $\mathcal{B}_1$ is that of $\mathcal{A}_1$'s. That is, the message encrypted in $\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*}$ is M_b. The probability that $\mathcal{B}_1$ succeeds in guessing $\mathcal{B}_1$'s challenge bit in the selective-identity security game is the same as the probability that $\mathcal{A}_1$ succeeds in guessing the challenge bit in the SSR-sID-CPA security game. Hence, we have $\mathsf{Adv}^{\mathsf{RIBE\text{-}sel}}_{\mathcal{B}_1,\mathcal{O}^{\mathsf{RIBE}}}(\lambda) = \mathsf{Adv}^{\mathsf{SSR\text{-}sID\text{-}CPA}}_{\mathcal{A}_1,\mathcal{O}^{\mathsf{SR\text{-}IBE}}}(\lambda)$, which completes the proof. □

Lemma 14. *For any PPT Type-**II** adversary $\mathcal{A}_2$, there exists a PPT adversary $\mathcal{B}_2$ against the selective-identity security of the underlying 2-level HIBE scheme* $\mathsf{h}.\Pi$ *such that* $\mathsf{Adv}^{\mathsf{SSR\text{-}sID\text{-}CPA}}_{\mathcal{A}_2,\mathcal{O}^{\mathit{SR\text{-}IBE}}}(\lambda) = \mathsf{Adv}^{\mathsf{HIBE\text{-}sel}}_{\mathcal{B}_2,\mathcal{O}^{\mathit{HIBE}}}(\lambda)$.

Proof. Let $\mathcal{C}_2$ be the challenger in the experiment $\mathsf{Exp}^{\mathsf{RIBE\text{-}sel}}_{\mathcal{A},\mathcal{O}^{\mathsf{HIBE}}}(\lambda)$ for the underling 2-level HIBE scheme, then the adversary $\mathcal{B}_2$ interacts with $\mathcal{A}_2$ and $\mathcal{C}_2$ as follows.

Initial: A_2 announces to $\mathcal{B}_2$ the challenge ID^* and t^*, and the latter sends the pair $(\mathsf{ID}^*, \mathsf{t}^*)$ as its own challenge 2-level hierarchical identity to $\mathcal{C}_2$.

Setup: $\mathcal{B}_2$ receives the public parameter $\mathsf{h.pp}$ from $\mathcal{C}_2$. Then, $\mathcal{B}_2$ initializes the counter $\mathsf{t_{cu}} := 1$, generates two empty lists SKList and $\mathsf{PrivList}$, runs $(\mathsf{r.pp}, \mathsf{r.sk}_{\mathsf{kgc}}) \leftarrow \mathsf{r.Setup}(1^\lambda)$ and $(\mathsf{r.uk}_1, \mathsf{r.sk}'_{\mathsf{kgc}}) \leftarrow \mathsf{r.KeyUP}(\mathsf{r.pp}, \mathsf{r.sk}_{\mathsf{kgc}}, 1, \mathsf{RL}_1 = \emptyset)$. Finally, $\mathcal{B}_2$ sends $\mathsf{pp} := (\mathsf{r.pp}, \mathsf{h.pp})$ and $\mathsf{uk}_1 := \mathsf{r.uk}_1$ to $\mathcal{A}_2$, and keeps $\mathsf{r.sk}_{\mathsf{kgc}}$ secret.

Long-term Transformation Key Generation Query: For a the long-term transformation key generation query ID from the adversary $\mathcal{A}_2$ (where it is required that $(\mathsf{ID}, *) \notin \mathsf{SKList}$), $\mathcal{B}_2$ runs $\mathsf{r.sk}_{\mathsf{ID}} \leftarrow \mathsf{r.GenSK}(\mathsf{r.pp}, \mathsf{r.sk}_{\mathsf{kgc}}, \mathsf{ID})$,

sets $\mathsf{sk}_{\mathsf{ID}} := \mathsf{r.sk}_{\mathsf{ID}}$, adds $(\mathsf{ID}, \mathsf{sk}_{\mathsf{ID}})$ into the list SKList, and returns nothing to $\mathcal{A}_2$.

Long-term Transformation Key Reveal Query: For a long-term transformation key reveal query ID from $\mathcal{A}_2$, $\mathcal{B}_2$ finds $\mathsf{sk}_{\mathsf{ID}}$ from SKList and returns it to $\mathcal{A}_2$.

Revoke & Update Key Query: For a revoke & update key query $\mathsf{RL} \subset \mathcal{ID}$ from $\mathcal{A}_2$, $\mathcal{B}_2$ checks whether $\mathsf{RL}_{\mathsf{t_{cu}}} \subset \mathsf{RL}$. If not, it returns $\perp$. Otherwise, $\mathcal{B}_2$ increments the current time period by $\mathsf{t_{cu}} \leftarrow \mathsf{t_{cu}} + 1$, sets $\mathsf{RL}_{\mathsf{t_{cu}}} \leftarrow \mathsf{RL}$, runs $(\mathsf{r.uk}_{\mathsf{t_{cu}}}, \mathsf{r.sk}'_{\mathsf{kgc}}) \leftarrow \mathsf{r.KeyUP}(\mathsf{r.pp}, \mathsf{r.sk}_{\mathsf{kgc}}, \mathsf{t_{cu}}, \mathsf{RL}_{\mathsf{t_{cu}}})$, and returns $\mathsf{uk}_{\mathsf{t_{cu}}} := \mathsf{r.uk}_{\mathsf{t_{cu}}}$ to $\mathcal{A}_2$.

Short-term Transformation Key Reveal Query: For a short-term transformation key reveal query $(\mathsf{ID}, \mathsf{t})$ from $\mathcal{A}_2$, $\mathcal{B}_2$ checks whether conditions $\mathsf{t} \leq \mathsf{t_{cu}}$, $\mathsf{ID} \notin \mathsf{RL}_{\mathsf{t}}$ and $(\mathsf{ID}, \mathsf{t}) \neq (\mathsf{ID}^*, \mathsf{t}^*)$ hold simultaneously. If not, $\mathcal{B}_2$ returns $\perp$ to $\mathcal{A}_2$. Otherwise, it is guaranteed that $\mathcal{B}_2$ has already generated $\mathsf{uk}_{\mathsf{t}} = \mathsf{r.uk}_{\mathsf{t}}$ and $\mathsf{r.sk}_{\mathsf{ID}}$. Then, $\mathcal{B}_2$ runs $\mathsf{r.dk}_{\mathsf{ID,t}} \leftarrow \mathsf{r.GenDK}(\mathsf{r.pp}, \mathsf{r.sk}_{\mathsf{ID}}, \mathsf{r.uk}_{\mathsf{t}})$ and returns $\mathsf{tk}_{\mathsf{ID,t}} := \mathsf{r.dk}_{\mathsf{ID,t}}$ to $\mathcal{A}_2$.

Private Key Generation Query: For a private key generation query ID from $\mathcal{A}_2$ (where it is required that $(\mathsf{ID}, *) \notin \mathsf{PrivList}$), $\mathcal{B}_2$ first makes a level-1 secret key generation query ID to $\mathcal{C}_2$ (note that upon this query, C_2 runs $\mathsf{h.sk}_{\mathsf{ID}} \leftarrow \mathsf{GenSK}(\mathsf{h.pp}, \mathsf{h.sk}_{\mathsf{kgc}}, \mathsf{ID})$, but returns nothing to $\mathcal{B}_2$). Then, $\mathcal{B}_2$ further makes a level-1 secret key reveal query ID to $\mathcal{C}_2$, and receives $\mathsf{h.sk}_{\mathsf{ID}}$ from $\mathcal{C}_2$. Finally, $\mathcal{B}_2$ sets $\mathsf{Priv}_{\mathsf{ID}} := \mathsf{h.sk}_{\mathsf{ID}}$, adds $(\mathsf{ID}, \mathsf{Priv}_{\mathsf{ID}})$ into the list $\mathsf{PrivList}$ and returns nothing to $\mathcal{A}_2$.

Private Key Reveal Query: For a private key reveal query ID from Type-**II** adversary $\mathcal{A}_2$, it is easy to check that the following conditions are satisfied:

- $(\mathsf{ID}, \mathsf{Priv}_{\mathsf{ID}}) \in \mathsf{PrivList}$ and $\mathsf{ID} \neq \mathsf{ID}^*$,

according to the strategy of $\mathcal{A}_2$. Therefore, the check does by the challenger $\mathcal{C}_2$ for the level-1 secret key reveal query ID is satisfied. This guarantees that $(\mathsf{ID}, \mathsf{Priv}_{\mathsf{ID}})$ is contained in the list $\mathsf{PrivList}$ and thus $\mathcal{B}_2$ returns $\mathsf{Priv}_{\mathsf{ID}}$ to $\mathcal{A}_2$.

Decryption Key Reveal Query: For a decryption key reveal query $(\mathsf{ID}, \mathsf{t})$ from $\mathcal{A}_2$ (where it is required that $((\mathsf{ID}, \mathsf{t}), *) \notin \mathsf{PrivList}$), $\mathcal{B}_2$ forwards it to $\mathcal{C}_2$. Note that upon this query, the checks performed by $\mathcal{C}_2$ for the level-2 secret key reveal query $(\mathsf{ID}, \mathsf{t})$ listed below are satisfied simultaneously:

- $(\mathsf{ID}, \mathsf{sk}_{\mathsf{ID}}) \in \mathsf{PrivList}$, $((\mathsf{ID}, \mathsf{t}), *) \notin \mathsf{PrivList}$ and $(\mathsf{ID}, \mathsf{t}) \neq (\mathsf{ID}^*, \mathsf{t}^*)$.

The first two conditions are obviously true. The last one also holds because $\mathsf{ID}^* \notin \mathsf{RL}_{\mathsf{t}^*}$ (according to $\mathcal{A}_2$'s strategy), $\mathsf{Dec}(\cdot)$ can not be queried on $(\mathsf{ID}^*, \mathsf{t}^*)$. Therefore, $\mathcal{C}_2$ returns $\mathsf{h.sk}_{\mathsf{ID,t}}$ to $\mathcal{B}_2$. Finally, $\mathcal{B}_2$ sets $\mathsf{dk}_{\mathsf{ID,t}} := \mathsf{h.sk}_{\mathsf{ID,t}}$, and returns $\mathsf{dk}_{\mathsf{ID,t}}$ to $\mathcal{A}_2$.

Challenge: Upon the challenge (M_0, M_1) from $\mathcal{A}_2$, $\mathcal{B}_2$ picks $\mathsf{r}.M \leftarrow \mathcal{M}$ uniformly at random, and then sets $\mathsf{h}.M_0 \leftarrow M_0 - \mathsf{r}.M$, $\mathsf{h}.M_1 \leftarrow M_1 - \mathsf{r}.M$. Then, $\mathcal{B}_2$ sends the challenge $(\mathsf{h}.M_0, \mathsf{h}.M_1)$ to $\mathcal{C}_2$ and receives $\mathcal{B}_2$'s challenge ciphertext $\mathsf{h.ct}_{\mathsf{ID}^*,\mathsf{t}^*} \leftarrow \mathsf{h.Encrypt}(\mathsf{h.pp}, (\mathsf{ID}^*, \mathsf{t}^*), \mathsf{h}.M_b)$ from $\mathcal{C}_2$, where $b \leftarrow \{0, 1\}$ is $\mathcal{B}_2$'s challenge bit. Next, $\mathcal{B}_2$ runs $\mathsf{r.ct}_{\mathsf{ID}^*,\mathsf{t}^*} \leftarrow \mathsf{r.Encrypt}(\mathsf{r.pp}, \mathsf{ID}^*, \mathsf{t}^*, \mathsf{r}.M)$ and returns the challenge ciphertext $\mathsf{ct}_{\mathsf{ID}^*,\mathsf{t}^*} := (\mathsf{r.ct}_{\mathsf{ID}^*,\mathsf{t}^*}, \mathsf{h.ct}_{\mathsf{ID}^*,\mathsf{t}^*})$ to $\mathcal{A}_2$.

Guess: A_2 outputs its guess bit b', $\mathcal{B}_2$ outputs b' as the guess of b.

Note that $\mathcal{B}_2$ perfectly simulates the SSR-sID-CPA security game for the Type-**II** adversary $\mathcal{A}_2$, so that the challenge bit b of $\mathcal{B}_2$ is that of $\mathcal{A}_2$'s. That is, the message encrypted in $\mathsf{ct}_{\mathsf{ID}^*,t^*}$ is M_b. The probability that $\mathcal{B}_2$ succeeds in guessing $\mathcal{B}_2$'s challenge bit in the selective-identity security game is the same as the probability that $\mathcal{A}_2$ succeeds in guessing the challenge bit in the SSR-sID-CPA security game. Hence, $\mathsf{Adv}^{\mathsf{HIBE\text{-}sel}}_{\mathcal{B}_2,\mathcal{O}^{\mathsf{HIBE}}}(\lambda) = \mathsf{Adv}^{\mathsf{SSR\text{-}sID\text{-}CPA}}_{\mathcal{A}_2,\mathcal{O}^{\mathsf{SR\text{-}IBE}}}(\lambda)$, which completes the proof. □

Due to Lemma 13 and 14, we can conclude that the generic SR-IBE scheme satisfies SSR-sID-CPA security, which completes the proof of Theorem 2. □

6 Conclusion

In this paper, we first revisit Qin et al.'s security model for SR-IBE [23], and propose a new security model called SSR-sID-CPA security by weakening the limitation on revoking the challenge identity and capturing both DKE attacks and STKE attacks. Then we propose a SSR-sID-CPA secure SR-IBE scheme from lattices. Compared with that of Nguyen et al. [20], our lattice-based SR-IBE removes the use of the double encryption technique, reduces the number of calls to sampling algorithm, and is proved to be more secure. Finally, we propose a generic construction of SR-IBE scheme, the security of which inherits those of the underlying building blocks.

References

1. Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, pages 553–572, 2010.
2. Miklós Ajtai. Generating hard instances of the short basis problem. In *Automata, Languages and Programming, 26th International Colloquium, ICALP'99, Prague, Czech Republic, July 11-15, 1999, Proceedings*, pages 1–9, 1999.
3. Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. *Theory Comput. Syst.*, 48(3):535–553, 2011.
4. Alexandra Boldyreva, Vipul Goyal, and Virendra Kumar. Identity-based encryption with efficient revocation. In *Proceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, Alexandria, Virginia, USA, October 27-31, 2008*, pages 417–426, 2008.
5. Alexandra Boldyreva, Vipul Goyal, and Virendra Kumar. Identity-based encryption with efficient revocation. *IACR Cryptology ePrint Archive*, 2012:52, 2012.
6. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, pages 213–229, 2001.
7. David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. *J. Cryptology*, 25(4):601–639, 2012.

8. Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Khoa Nguyen. Revocable identity-based encryption from lattices. In *Information Security and Privacy - 17th Australasian Conference, ACISP 2012, Wollongong, NSW, Australia, July 9-11, 2012. Proceedings*, pages 390–403, 2012.
9. Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.
10. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 197–206, 2008.
11. Yuu Ishida, Junji Shikata, and Yohei Watanabe. Cca-secure revocable identity-based encryption schemes with decryption key exposure resistance. *IJACT*, 3(3):288–311, 2017.
12. Shuichi Katsumata, Takahiro Matsuda, and Atsushi Takayasu. Lattice-based revocable (hierarchical) IBE with decryption key exposure resistance. In *Public-Key Cryptography - PKC 2019 - 22nd IACR International Conference on Practice and Theory of Public-Key Cryptography, Beijing, China, April 14-17, 2019, Proceedings, Part II*, pages 441–471, 2019.
13. 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.
14. Benoît Libert, Fabrice Mouhartem, and Khoa Nguyen. A lattice-based group signature scheme with message-dependent opening. In *Applied Cryptography and Network Security - 14th International Conference, ACNS 2016, Guildford, UK, June 19-22, 2016. Proceedings*, pages 137–155, 2016.
15. A. E. Litvak, Alalin Pajor, Mark Rudelson, and Nicole Tomczak-Jaegermann. Smallest singular value of random matrices and geometry of random polytopes. *Advances in Mathematics*, 195(2):491–523, 2005.
16. Xuecheng Ma and Dongdai Lin. A generic construction of revocable identity-based encryption. *IACR Cryptology ePrint Archive*, 2019:299, 2019.
17. Xianping Mao, Junzuo Lai, Kefei Chen, Jian Weng, and Qixiang Mei. Efficient revocable identity-based encryption from multilinear maps. *Security and Communication Networks*, 8(18):3511–3522, 2015.
18. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, pages 700–718, 2012.
19. Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, pages 41–62, 2001.
20. Khoa Nguyen, Huaxiong Wang, and Juanyang Zhang. Server-aided revocable identity-based encryption from lattices. In *Cryptology and Network Security - 15th International Conference, CANS 2016, Milan, Italy, November 14-16, 2016, Proceedings*, pages 107–123, 2016.
21. Juan Manuel González Nieto, Mark Manulis, and Dongdong Sun. Fully private revocable predicate encryption. In *Information Security and Privacy - 17th Australasian Conference, ACISP 2012, Wollongong, NSW, Australia, July 9-11, 2012. Proceedings*, pages 350–363, 2012.

22. Seunghwan Park, Kwangsu Lee, and Dong Hoon Lee. New constructions of revocable identity-based encryption from multilinear maps. *IEEE Trans. Information Forensics and Security*, 10(8):1564–1577, 2015.
23. Baodong Qin, Robert H. Deng, Yingjiu Li, and Shengli Liu. Server-aided revocable identity-based encryption. In *Computer Security - ESORICS 2015 - 20th European Symposium on Research in Computer Security, Vienna, Austria, September 21-25, 2015, Proceedings, Part I*, pages 286–304, 2015.
24. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 84–93, 2005.
25. Jae Hong Seo and Keita Emura. Revocable identity-based encryption revisited: Security model and construction. In *Public-Key Cryptography - PKC 2013 - 16th International Conference on Practice and Theory in Public-Key Cryptography, Nara, Japan, February 26 - March 1, 2013. Proceedings*, pages 216–234, 2013.
26. Jae Hong Seo and Keita Emura. Revocable hierarchical identity-based encryption. *Theor. Comput. Sci.*, 542:44–62, 2014.
27. Jae Hong Seo and Keita Emura. Revocable hierarchical identity-based encryption via history-free approach. *Theor. Comput. Sci.*, 615:45–60, 2016.
28. Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*, pages 47–53, 1984.
29. Atsushi Takayasu and Yohei Watanabe. Lattice-based revocable identity-based encryption with bounded decryption key exposure resistance. In *Information Security and Privacy - 22nd Australasian Conference, ACISP 2017, Auckland, New Zealand, July 3-5, 2017, Proceedings, Part I*, pages 184–204, 2017.
30. Shixiong Wang, Juanyang Zhang, Jingnan He, Huaxiong Wang, and Chao Li. Simplified revocable hierarchical identity-based encryption from lattices. In *Cryptology and Network Security - 18th International Conference, CANS 2019, Fuzhou, China, October 25-27, 2019, Proceedings*, pages 99–119, 2019.
31. 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 *Topics in Cryptology - CT-RSA 2017 - The Cryptographers' Track at the RSA Conference 2017, San Francisco, CA, USA, February 14-17, 2017, Proceedings*, pages 432–449, 2017.