Revocable Identity-Based Encryption from Lattices

Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Khoa Nguyen

Nanyang Technological University, Singapore
s080001@e.ntu.edu.sg
{hoonwei,lingsan,hxwang}@ntu.edu.sg
nguy0106@e.ntu.edu.sg

Abstract. In this paper, we present an identity-based encryption (IBE) scheme from lattices with efficient key revocation. We adopt multiple trapdoors from the Agrawal-Boneh-Boyen and Gentry-Peikerty-Vaikuntanathan lattice IBE schemes to realize key revocation, which in turn, makes use of binary-tree data structure. Using our scheme, key update requires logarithmic complexity in the maximal number of users and linear in the number of revoked users for the relevant key authority. We prove that our scheme is selective secure in the standard model and under the LWE assumption, which is as hard as the worst-case approximating short vectors on arbitrary lattices. Moreover, our key revocation techniques from lattices can be applied to obtain revocable functional encryption schemes in the similar setting.

Keywords: Lattice-based Cryptography, Identity-based Encryption, Key Revocation, Functional Encryption.

1 Introduction

The concept of *identity-based encryption* (IBE) was proposed by Shamir [36]. It allows a sender to encrypt a message using the recipient's identity as a public key. The private key corresponding to the public key or identity is generated by a key authority (or private key generator). IBE began to be studied extensively only after the seminal work of Boneh and Franklin [12] on practical pairing-based IBE systems, see for example [14, 38, 39]. Meanwhile, there also exist proposals on IBE systems based on quadratic residuosity [18, 15], although it is still not known how to build such systems that are secure in the standard model. In recent years, however, lattice-based IBE [22, 17, 1, 2] has received considerable attention from the cryptographic research community. Lattices have becoming an attractive and powerful tool to build a broad range of cryptographic primitives [7, 27, 9, 32, 33, 21]. This is so as many lattice-based constructions are quite efficient and typically simple to implement. Moreover they are all believed to be secure against attacks using quantum computers, a property not achievable by cryptographic primitives based on factoring or discrete logarithm.

A system user's public key may need to be removed for various reasons. For example, the private key corresponding to the public key has been stolen; the user has lost her private key; or the user is no longer a legitimate system user. In these cases, it is important that the public/private key pair be revoked and replaced by new keys. In the IBE setting, Boneh and Franklin [12] suggested that the sender appends the current validity period to the intended identity during encryption and the recipient periodically receives a new private key. Unfortunately, such solution requires the key authority to perform work that is linear in the number of non-revoked users. Further, the key authority needs to create and transmit a new key to each non-revoked user through some form of authenticated and secure channel. Boldyreva, Goyal and Kumar [11] recently proposed a

revocable IBE (RIBE) scheme that significantly reduces the key authority's workload (in terms of key revocation) to logarithmic (instead of linear) in the number of users, while keeping the scheme efficient for senders and receivers. Their RIBE scheme uses key revocation techniques based on binary-tree data structure, also used in [6, 29], and builds on a fuzzy IBE (FIBE) scheme introduced by Sahai and Waters [34] that is secure in the selective-ID model. Note that Boldyreva et al's RIBE is the first IBE scheme that supports non-interactive key revocation (in the sense that non-revoked users need not interact with the key authority in order to update their keys). Prior to their work, all revocation techniques require interactions between users and the key authority or some kind of trusted hardware. Moreover, the use of a binary-tree reduces the amount of work in key update from being proportional to logarithmic complexity in the maximal number of users. Libert and Vergnaud [26] subsequently proposed an RIBE scheme in the adaptive-ID model using similar key revocation techniques as with [11]. However, instead of making use of an FIBE scheme, they adopt a variant [25] of the Waters IBE scheme [38]. Nevertheless, all the above RIBE schemes are constructed from bilinear pairings.

In the spirit of expanding the study of lattice-based IBE, we show, in this paper, how to construct an RIBE scheme in the lattice setting.

1.1 Our Results

Our construction of RIBE from lattices makes use of the following building blocks: (i) lattice IBE proposed by Agrawal, Boneh, and Boyen [1]; (ii) trapdoors for lattice IBE proposed by Gentry, Peikerty, and Vaikuntanathan [22]; and (iii) the binary-tree data structure for key update used in [6, 29, 11, 26]. More specifically, we extend the lattice IBE scheme of [1] with trapdoors from [22] to enable non-interactive key revocation. As with prior work, the binary-tree data structure is used to improve the efficiency of secret key update, allowing us to achieve key update with logarithmic complexity in the maximal number of users and linear in the number of revoked users for the key authority.

We note that our RIBE scheme is not a straightforward combination of the aforementioned building blocks because we require that our user public key comprises two components: identity and time, in order to obtain the "non-interactive" property. Hence, our construction requires two instances of Agrawal et al.'s IBE scheme to deal with users' identities and times respectively. Further, we require a random n-vector $\mathbf{u}$ to be part of the public parameters that plays the role of linking identity to time for each node associated to the binary-tree. Briefly speaking, this can be achieved by randomly splitting the vector $\mathbf{u}$ into two vectors $\mathbf{u}_1, \mathbf{u}_2$ for each node to indicate identity and time, respectively.

We prove that our RIBE scheme is selective secure in the standard model and under the LWE assumption, which is as hard as the worst-case approximation of short vectors on arbitrary lattices [33, 31]. Simply applying the simulation techniques of [1] to our lattice setting does not work, since the trapdoors can respond to only all key (short vector) queries for all identities $\mathrm{id} \neq \mathrm{id}^*$ and times $\mathrm{t} \neq \mathrm{t}^*$. We address this by adopting the simulation trapdoors of [22]. That is, we sample a short vector from some distribution to generate $\mathbf{u}_1$ or $\mathbf{u}_2$ instead of generating both $\mathbf{u}_1$ and $\mathbf{u}_2$ randomly for each node in the simulation. Such $\mathbf{u}_1$ or $\mathbf{u}_2$ is indistinguishable from the uniform distribution. The sampled short vectors will be used to respond to a query for the challenge identity id^* and a query for the challenge time t^*.

Our key revocation techniques from lattices can be extended to functional encryption schemes [16, 34, 23, 24, 30] in the lattice setting, for example the inner product encryption (IPE) scheme of [5].

1.2 Related Work

Our work, which focuses on how to construct revocable IBE from lattices, is concurrent but independent from the very recent proposal of lattice FIBE in [4]. There is some similarity between [4] and our work, that is, attributes are embedded in the shares $\mathbf{u}_i$ of vector $\mathbf{u}$ in the construction and the shares $\mathbf{u}_i$ of the challenge attributes are generated by sampling random short vectors in the simulation. However, our approach is different in the sense that we directly and randomly split the vector $\mathbf{u}$ instead of using the Shamir secret-sharing scheme and the Lagrange interpolation formula. This makes our system more efficient. Another difference is that we make use of only one matrix associated with a trapdoor basis instead of ℓ matrices, where ℓ is the maximal number of attributes. Our method could also be applied to their large universe scheme and this significantly reduces the size of the master secret key.

We note that the idea of using more than one trapdoor in the keys has also been mentioned in Agrawal et al.'s hierarchical IBE (HIBE) scheme [3] and in the completely non-malleable public-key encryption scheme by Sapehi et al. [35].

2 Definitions

2.1 Notation

Throughout the paper we say that a function $\epsilon : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is negligible if $\epsilon(n)$ is smaller than all polynomial fractions for sufficiently large n. We say that an event happens with overwhelming probability if it happens with probability at least $1 - \epsilon(n)$ for some negligible function ϵ. We say that integer vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n \in \mathbb{Z}^m$ are $\mathbb{Z}_q$-linearly independent if they are linearly independent when reduced modulo q.

The statistical distance of two random variables X and Y over a discrete domain Ω is defined as $\Delta(X;Y) := \frac{1}{2}\sum_{s\in\Omega} |\Pr[X = s] - \Pr[Y = s]|$. We say that X is δ-uniform over Ω if $\Delta(X;U_\Omega) \leq \delta$ where U_Ω is a uniform random variable over Ω. Let $X(\lambda)$ and $Y(\lambda)$ be ensembles of random variables, we say that X and Y are statistically close if $d(\lambda) := \Delta(X(\lambda);Y(\lambda))$ is a negligible function of λ.

2.2 Syntax of RIBE

Here, we recall the definitions of security for RIBE as defined in [11].

Definition 1. *An identity-based encryption with efficient revocation or simply revocable IBE scheme has seven probabilistic polynomial-time (PPT) algorithms* **Setup**, **PriKeyGen**, **KeyUpd**, **DecKeyGen**, **Enc**, **Dec**, *and* **KeyRev** *with associated message space $\mathcal{M}$, identity space $\mathcal{I}$, and time space $\mathcal{T}$. We assume that the size of $\mathcal{T}$ is polynomial in the security parameter. Each algorithm is run by either one of three types of parties—key authority, sender or receiver. Key authority maintains a revocation list* RL *and state* ST. *In what follows, an algorithm is called stateful if it updates* RL *or* ST. *We treat time as discrete as opposed to continuous.*

Setup$(1^\lambda, N)$ *takes as input a security parameter* λ *and a maximal number of users* N*. It outputs a public parameters* PP*, a master key* MK*, a revocation list* RL *(initially empty), and a state* ST*. (This is run by the key authority.)*

PriKeyGen$(\mathsf{PP}, \mathsf{MK}, \mathrm{id}, \mathsf{ST})$ *takes as input the public parameters* PP*, the master key* MK*, an identity* $\mathrm{id} \in \mathcal{I}$*, and the state* ST*. It outputs a private key* $\mathsf{SK}_{\mathrm{id}}$ *and an updated state* ST*. (This is stateful and run by the key authority.)*

KeyUpd$(\mathsf{PP}, \mathsf{MK}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$ *takes as input the public parameters* PP*, the master key* MK*, a key update time* $\mathrm{t} \in \mathcal{T}$*, the revocation list* RL*, and the state* ST*. It outputs a key update* KU_{t}*. (This is run by the key authority.)*

DecKeyGen$(\mathsf{SK}_{\mathrm{id}}, \mathsf{KU}_{\mathrm{t}})$ *takes as input a private key* $\mathsf{SK}_{\mathrm{id}}$ *and key update* KU_{t}*. It outputs a decryption key* $\mathsf{DK}_{\mathrm{id,t}}$ *or a special symbol* $\perp$ *indicating that* id *was revoked. (This is deterministic and run by the receiver.)*

Enc$(\mathsf{PP}, \mathrm{id}, \mathrm{t}, \mathrm{m})$ *takes as input the public parameters* PP*, an identity* $\mathrm{id} \in \mathcal{I}$*, an encryption time* $\mathrm{t} \in \mathcal{T}$*, and a message* $\mathrm{m} \in \mathcal{M}$*. It outputs a ciphertext* $\mathsf{CT}_{\mathrm{id,t}}$*. (This is run by the sender. For simplicity and wlog we assume that* id, t *are efficiently computable from* $\mathsf{CT}_{\mathrm{id,t}}$*.)*

Dec$(\mathsf{PP}, \mathsf{DK}_{\mathrm{id,t}}, \mathsf{CT}_{\mathrm{id,t}})$ *takes as input the public parameters* PP*, a decryption key* $\mathsf{DK}_{\mathrm{id,t}}$*, and a ciphertext* $\mathsf{CT}_{\mathrm{id,t}}$*. It outputs a message* $\mathrm{m} \in \mathcal{M}$*. (This is deterministic and run by the receiver.)*

KeyRev$(\mathrm{id}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$ *takes as input an identity to be revoked* $\mathrm{id} \in \mathcal{I}$*, a revocation time* $\mathrm{t} \in \mathcal{T}$*, the revocation list* RL*, and the state* ST*. It outputs an updated revocation list* RL*. (This is stateful and run by the key authority.)*

The consistency condition requires that for all $\lambda \in \mathbb{N}$ and polynomials (in λ) N, all PP and MK output by setup algorithm Setup, all $\mathrm{m} \in \mathcal{M}, \mathrm{id} \in \mathcal{I}, \mathrm{t} \in \mathcal{T}$ and all possible valid states ST and revocation lists RL, if identity id was not revoked before or, at time t then the following experiment returns 1 except for a negligible probability:

$$(\mathsf{SK}_{\mathrm{id}}, \mathsf{ST}) \xleftarrow{\$} \mathbf{PriKeyGen}(\mathsf{PP}, \mathsf{MK}, \mathrm{id}, \mathsf{ST});$$

$$\mathsf{KU}_{\mathrm{t}} \xleftarrow{\$} \mathbf{KeyUpd}(\mathsf{PP}, \mathsf{MK}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$$

$$\mathsf{DK}_{\mathrm{id,t}} \leftarrow \mathbf{DecKeyGen}(\mathsf{SK}_{\mathrm{id}}, \mathsf{KU}_{\mathrm{t}}); \mathsf{CT}_{\mathrm{id,t}} \xleftarrow{\$} \mathbf{Enc}(\mathsf{PP}, \mathrm{id}, \mathrm{t}, \mathrm{m})$$

$$\text{If } \mathbf{Dec}(\mathsf{PP}, \mathsf{DK}_{\mathrm{id,t}}, \mathsf{CT}_{\mathrm{id,t}}) = \mathrm{m} \text{ then return 1 else return 0.}$$

Boldyreva et al. formalized and defined the selective-revocable-ID security in the following experiments. Their definition captures not only the standard notion of selective-ID security but also takes into account key revocation:

Initial: The adversary first outputs the challenge identity id^* and time t^*, and also some information state it wants to preserve.

Setup: It is run to generate public parameters PP, a master key MK, a revocation list RL (initially empty), and a state ST. Then PP is given to $\mathcal{A}$.

Query: $\mathcal{A}$ may adaptively make a polynomial number of queries of the following oracles (the oracles share state):

- The private key generation oracle $\mathbf{PriKeyGen}(\cdot)$ takes as input an identity id and runs $\mathbf{PriKeyGen}(\mathsf{PP}, \mathsf{MK}, \mathrm{id}, \mathsf{ST})$ to return a private key $\mathsf{SK}_{\mathrm{id}}$.
- The key update generation oracle $\mathbf{KeyUpd}(\cdot)$ takes as input time t and runs $\mathbf{KeyUpd}(\mathsf{PP}, \mathsf{MK}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$ to return a key update KU_{t}.

– The revocation oracle $\mathbf{KeyRev}(\cdot)$ takes as input an identity id and time t and runs $\mathbf{KeyRev}(\mathrm{id}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$ to update RL.

Challenge: $\mathcal{A}$ outputs the same length challenge $\mathrm{m}_{(0)}, \mathrm{m}_{(1)} \in \mathcal{M}$. A random bit β is chosen. $\mathcal{A}$ is given $\mathbf{Enc}(\mathsf{PP}, \mathrm{id}^*, \mathrm{t}^*, \mathrm{m}_{(\beta)})$.
Guess: The adversary may continue to make a polynomial number of queries of the following oracles as in query phase and outputs a bit β', and succeeds if $\beta' = \beta$.
The following restrictions must always hold:

1. $\mathbf{KeyUpd}(\cdot)$ and $\mathbf{KeyRev}(\cdot, \cdot)$ can be queried on time which is greater than or equal to the time of all previous queries, i.e., the adversary is allowed to query only in non-decreasing order of time. Also, the oracle $\mathbf{KeyRev}(\cdot, \cdot)$ cannot be queried at time t if $\mathbf{KeyUpd}(\cdot)$ was queried on t.
2. If $\mathbf{PriKeyGen}(\cdot)$ was queried on identity id^* then $\mathbf{KeyRev}(\cdot, \cdot)$ must be queried on $(\mathrm{id}^*, \mathrm{t})$ for some $\mathrm{t} \leq \mathrm{t}^*$, i.e., identity id^* must be in RL when $\mathbf{KeyUpd}(\cdot)$ is queried at time t^*.

We define the advantage of $\mathcal{A}$ as the quantity

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}sRID\text{-}CPA}}(\lambda) := \Pr[\beta' = \beta] - 1/2.$$

Definition 2. *The scheme RIBE is said to be* $\mathsf{IND\text{-}sRID\text{-}CPA}$ *secure if the function* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}sRID\text{-}CPA}}(\lambda)$ *is negligible in* λ *for any efficient* $\mathcal{A}$ *and polynomial* n.

3 Background on Lattices

In this section, we describe the required concepts from lattices.

3.1 Integer Lattices

Let $\mathbf{B} := [\mathbf{b}_1 | \ldots | \mathbf{b}_m] \in \mathbb{R}^{m \times m}$ be an $m \times m$ matrix whose columns are linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_m \in \mathbb{R}^m$. The m-dimensional full-rank lattice Λ generated by $\mathbf{B}$ is the set,

$$\Lambda := \mathcal{L}(\mathbf{B}) := \left\{ \mathbf{y} \in \mathbb{R}^m \text{ s.t. } \exists \mathbf{s} \in \mathbb{Z}^m, \mathbf{y} = \mathbf{B}\mathbf{s} = \sum_{i=1}^{m} \mathbf{s}_i \mathbf{b}_i \right\}$$

Here, we are interested in integer lattices, i.e, when $\mathcal{L}$ is a subset of $\mathbb{Z}^m$.

Definition 3. *For a prime* q, $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ *and* $\mathbf{u} \in \mathbb{Z}_q^n$, *define:*

$$\Lambda_q^{\perp}(\mathbf{A}) := \{\mathbf{e} \in \mathbb{Z}^m \text{ s.t. } \mathbf{A}\mathbf{e} = \mathbf{0} \,(\bmod q)\}$$
$$\Lambda_q^{\mathbf{u}}(\mathbf{A}) := \{\mathbf{e} \in \mathbb{Z}^m \text{ s.t. } \mathbf{A}\mathbf{e} = \mathbf{u} \,(\bmod q)\}$$

3.2 The Gram-Schmidt Norm and Trapdoors for Lattices

Let S be a set of vectors $S := \{\mathbf{s}_1, \ldots, \mathbf{s}_k\}$ in $\mathbb{R}^m$, we use $\|S\|$ to denote the Euclidean norm of the longest vector in S, i.e., $\|S\| := \max_i \sqrt{s_{i,1}^2 + \ldots + s_{i,m}^2}$ for $1 \leq i \leq k$, where $\mathbf{s}_i := (s_{i,1}, \ldots, s_{i,m})$. We use $\tilde{S} := \{\tilde{\mathbf{s}}_1, \ldots, \tilde{\mathbf{s}}_k\} \subset \mathbb{R}^m$ to denote the Gram-Schmidt orthogonalization of the vectors $\mathbf{s}_1, \ldots, \mathbf{s}_k$ in that order. We refer to $\|\tilde{S}\|$ as the Gram-Schmidt norm of S.

The problem of generating a random lattice $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ together with a full short basis $\mathbf{T_A}$ of $\Lambda_q^\perp(\mathbf{A})$ has been previously investigated by [8, 10]. Here we use a better result with tighter parameters which was recently discovered by Micciancio and Peikert [28].

Theorem 1. *Let $n \geq 1$, $q \geq 2$ be integers and $m = \lceil 2n \log q \rceil$. There is an efficient PPT algorithm $\mathsf{TrapGen}(q, n)$ that outputs a pair $(\mathbf{A} \in \mathbb{Z}_q^{n \times m}, \mathbf{T_A} \in \mathbb{Z}_q^{m \times m})$ such that $\mathbf{A}$ is statistically close to a uniform matrix in $\mathbb{Z}_q^{n \times m}$ and $\mathbf{T_A}$ is a basis for $\Lambda_q^\perp(\mathbf{A})$ satisfying $\|\widetilde{\mathbf{T_A}}\| \leq \mathcal{O}(\sqrt{n \log q})$ and $\|\mathbf{T_A}\| \leq \mathcal{O}(n \log q)$ with all but negligible probability in n.*

3.3 Discrete Gaussians

Let Λ be an m-dimensional lattice. For any vector $\mathbf{c} \in \mathbb{R}^m$ and any positive parameter $\sigma \in \mathbb{R}_{>0}$, define:

- $\rho_{\sigma,\mathbf{c}}(\mathbf{x}) := \exp\left(-\pi \frac{\|\mathbf{x}-\mathbf{c}\|^2}{\sigma^2}\right)$: a Gaussian-shaped function on $\mathbb{R}^m$ with center $\mathbf{c}$ and parameter σ,
- $\rho_{\sigma,\mathbf{c}}(\Lambda) := \sum_{\mathbf{x} \in \Lambda} \rho_{\sigma,\mathbf{c}}(\mathbf{x})$: the (always converging) sum of $\rho_{\sigma,\mathbf{c}}$ over Λ,
- $\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}$: the discrete Gaussian distribution over Λ with parameters σ and center $\mathbf{c}$,

$$\forall \mathbf{y} \in \Lambda, \quad \mathcal{D}_{\Lambda,\sigma,\mathbf{c}}(\mathbf{y}) := \frac{\rho_{\sigma,\mathbf{c}}(\mathbf{y})}{\rho_{\sigma,\mathbf{c}}(\Lambda)}.$$

For notational convenience, we abbreviate $\rho_{\sigma,\mathbf{0}}$ and $\mathcal{D}_{\Lambda,\sigma,\mathbf{0}}$ as ρ_σ and $\mathcal{D}_{\Lambda,\sigma}$.

The following lemmas from [22] is essential for our security proof.

Lemma 1. *There is an efficient PPT algorithm $\mathsf{SampleGaussian}$ that, given a basis $\mathbf{B}$ of an m-dimensional lattice $\Lambda = \mathcal{L}(\mathbf{B})$, a parameter $\sigma \geq \|\widetilde{\mathbf{B}}\| \cdot \omega(\sqrt{\log m})$, and a center $\mathbf{c} \in \mathbb{R}^m$, outputs a sample from a distribution that is statistically close to $\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}$.*

Let $\mathbf{B}_z$ be the standard basis for $\mathbb{Z}^m$, we use the $\mathsf{SampleGaussian}(\mathbf{B}_z, \sigma, 0)$ algorithm to sample from distribution $\mathcal{D}_{\mathbb{Z}^m,\sigma}$.

Lemma 2. *Let n and q be positive integers with q prime, and let $m \geq 2n \log q$. Then for all but a $2q^{-n}$ fraction of all $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and for any $\sigma \geq \omega(\sqrt{\log m})$, the distribution of the syndrome $\mathbf{u} = \mathbf{Ae}$ mod q is statistically close to uniform over $\mathbb{Z}_q^n$, where $\mathbf{e}$ is from $\mathcal{D}_{\mathbb{Z}^m,\sigma}$.*

3.4 Sampling Algorithms

The following $\mathsf{SampleLeft}$ [17, 1] and $\mathsf{SampleRight}$ [1] algorithms will be used to sample short vectors in our construction and in the simulation, respectively. Let $\mathbf{A}$ and $\mathbf{C}$ be matrices in $\mathbb{Z}_q^{n \times m}$ and let $\mathbf{R}$ be a matrix in $\{-1, 1\}^{m \times m}$. By using either a trapdoor for $\Lambda_q^\perp(\mathbf{A})$ or a trapdoor $\Lambda_q^\perp(\mathbf{C})$, we can sample a short vector $\mathbf{e}$ in $\Lambda_q^\perp(\mathbf{F})$ for some $\mathbf{u}$ in $\mathbb{Z}_q^n$, where $\mathbf{F} := (\mathbf{A}|\mathbf{AR} + \mathbf{C}) \in \mathbb{Z}_q^{n \times 2m}$. With appropriate parameters, the distribution of $\mathbf{e}$ produced by these two algorithms is statistically indistinguishable.

Theorem 2. *Let $q > 2$ and $m > n$. Then there is an efficient PPT algorithm* SampleLeft *that takes as input a rank n matrix $\mathbf{A}$ in $\mathbb{Z}_q^{n\times m}$, a matrix $\mathbf{M}$ in $\mathbb{Z}_q^{n\times m_1}$, a "short" basis $\mathbf{T_A}$ of $\Lambda_q^\perp(\mathbf{A})$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, and a gaussian parameter $\sigma > \|\widetilde{\mathbf{T_A}}\| \cdot \omega(\sqrt{\log(m+m_1)})$. It outputs a vector $\mathbf{e} \in \mathbb{Z}^{m+m_1}$ distributed statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{F}_1),\sigma}$ where $\mathbf{F}_1 := (\mathbf{A}|\mathbf{M})$. In particular, $\mathbf{e} \in \Lambda_q^{\mathbf{u}}(\mathbf{F}_1)$.*

Theorem 3. *Let $q > 2$ and $m > n$. There is an efficient PPT algorithm* SampleRight *that takes as input matrices $\mathbf{A}, \mathbf{C}$ in $\mathbb{Z}_q^{n\times m}$ where $\mathbf{C}$ is rank n, a uniform random matrix $\mathbf{R} \in \{-1,1\}^{m\times m}$, a basis $\mathbf{T_C}$ of $\Lambda_q^\perp(\mathbf{C})$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, and a gaussian parameter $\sigma > \|\widetilde{\mathbf{T_C}}\| \cdot \sqrt{m}\omega(\log(m))$. It outputs a vector $\mathbf{e} \in \mathbb{Z}^{2m}$ distributed statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{F}_2),\sigma}$ where $\mathbf{F}_2 := (\mathbf{A}|\mathbf{AR}+\mathbf{C})$. In particular, $\mathbf{e} \in \Lambda_q^{\mathbf{u}}(\mathbf{F}_2)$.*

We will also need the following lemma, generalization of the left over hash lemma due to Dodis et al. [20], in our proof.

Lemma 3. *Suppose that $m > (n+1)\log q + \omega(\log n)$ and that q is 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 $(\mathbf{A}, \mathbf{B}, \mathbf{R}^\top\mathbf{w})$.*

3.5 The LWE Hardness Assumption

The security of our construction can be reduced to the LWE (learning with errors) problem defined by Regev [33].

Definition 4. *Consider a prime q, a positive integer n, and a distribution χ over $\mathbb{Z}_q$, all public. An $(\mathbb{Z}_q, n, \chi)$-LWE problem instance consists of access to an unspecified challenge oracle $\mathcal{O}$, being, either, a noisy pseudo-random sampler $\mathcal{O}_{\mathbf{s}}$ carrying some constant random secret key $\mathbf{s} \in \mathbb{Z}_q^n$, or, a truly random sampler $\mathcal{O}_\$$, whose behaviors are respectively as follows:*

- *$\mathcal{O}_{\mathbf{s}}$: outputs samples of the form $(\mathbf{u}_i, v_i) = (\mathbf{u}_i, \mathbf{u}_i^\top\mathbf{s} + x_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, where, $\mathbf{s} \in \mathbb{Z}_q^n$ is a uniformly distributed persistent value invariant across invocations, $x_i \in \mathbb{Z}_q$ is a fresh sample from χ, and $\mathbf{u}_i$ is uniform in $\mathbb{Z}_q^n$.*
- *$\mathcal{O}_\$$: outputs truly uniform random samples from $\mathbb{Z}_q^n \times \mathbb{Z}_q$.*

The $(\mathbb{Z}_q, n, \chi)$-LWE problem allows repeated queries to the challenge oracle $\mathcal{O}$. We say that an algorithm $\mathcal{A}$ decides the $(\mathbb{Z}_q, n, \chi)$-LWE problem if $|\Pr[\mathcal{A}^{\mathcal{O}_{\mathbf{s}}} = 1] - \Pr[\mathcal{A}^{\mathcal{O}_\$} = 1]|$ is non-negligible for a random $\mathbf{s} \in \mathbb{Z}_q^n$.

Regev [33] and Peikert [31] showed that for some noise distribution χ, denoted $\overline{\Psi}_\alpha$, the LWE problem is at least as hard as the worst-case SIVP and GapSVP under a quantum reduction if the parameters are appropriately set.

Definition 5. *Consider a real parameter $\alpha = \alpha(n) \in (0,1)$ and a prime q. Let $\mathbb{T} := \mathbb{R}/\mathbb{Z}$ be the group of reals $[0,1)$ with addition modulo 1. Let Ψ_α be the distribution over $\mathbb{T}$ of a normal variable with mean 0 and standard deviation $\alpha/\sqrt{2\pi}$ then reduced modulo 1. Let $\lfloor x \rceil := \lfloor x + \frac{1}{2} \rfloor$ be the nearest integer to the real $x \in \mathbb{R}$. We then define $\overline{\Psi}_\alpha$ as the discrete distribution over $\mathbb{Z}_q$ of the random variable $\lfloor qX \rceil$ mod q where the random variable $X \in \mathbb{T}$ has distribution Ψ_α.*

3.6 Encoding Identities as Matrices

In our construction and proof of security, we require an injective encoding function $\mathsf{H} : \mathbb{Z}_q^n \to \mathbb{Z}_q^{n \times n}$ to map identities in $\mathbb{Z}_q^n$ to matrices in $\mathbb{Z}_q^{n \times n}$. Concrete construction of such a function can be found in [1, 19].

Definition 6. *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 an encoding with full-rank differences (FRD) if:*

1. *for all distinct $\mathbf{u}, \mathbf{v} \in \mathbb{Z}_q^n$, the matrix $\mathsf{H}(\mathbf{u}) - \mathsf{H}(\mathbf{v}) \in \mathbb{Z}_q^{n \times n}$ is full rank;*
2. *H is computable in polynomial time in $n \log q$.*

4 Lattice RIBE

4.1 The Binary-tree Data Structure

Our construction makes use of binary-tree data structure, as with [6, 29, 11, 26]. We denote the binary-tree by BT and its root node by root. If ν is a leaf node then $\mathsf{Path}(\nu)$ denotes the set of nodes on the path from ν to root (both ν and root inclusive). If θ is a non-leaf node then θ_ℓ, θ_r denote the left and right child of θ, respectively. We assume that all nodes in the tree are uniquely encoded as strings, and the tree is defined by all of its node descriptions.

Each user is assigned to a leaf node ν. Upon registration, the key authority provides the user with a set of distinct private keys for each node in $\mathsf{Path}(\nu)$.

At time t, the key authority determines the minimal set Y of nodes in BT such that none of the nodes in RL with corresponding time $\leq$ t (users revoked on or before t) have any ancestor (or, themselves) in the set Y, and all other leaf nodes (corresponding to non-revoked users) have exactly one ancestor (or, themselves) in the set. This algorithm, denoted by $\mathsf{KUNodes}$, takes as input a binary tree BT, a revocation list RL and a time t) and can be formally specified as follows:

$\mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL}, \mathrm{t})$
  $\mathsf{X}, \mathsf{Y} \leftarrow \emptyset$
  $\forall (\nu_i, \mathrm{t}_i) \in \mathsf{RL}$
    if $\mathrm{t}_i \leq \mathrm{t}$ then add $\mathsf{Path}(\nu_i)$ to X
  $\forall \theta \in \mathsf{X}$
    if $\theta_\ell \notin \mathsf{X}$ then add θ_ℓ to Y
    if $\theta_r \notin \mathsf{X}$ then add θ_r to Y
  If $\mathsf{Y} = \emptyset$ then add root to Y
  Return Y

The $\mathsf{KUNodes}$ algorithm marks all the ancestors of revoked nodes as revoked and outputs all the non-revoked children of revoked nodes. A graphical description is illustrated in Figure 1 of Appendix A.

The key authority then publishes a key update for all nodes of Y.

A user assigned to leaf ν is then able to form an effective decryption key for time t if the set Y contains a node in $\mathsf{Path}(\nu)$. By doing so, every update of the revocation list RL only requires the key authority to perform logarithmic work in the maximal number of users and linear in the number of revoked users.

4.2 The Agrawal et al. IBE Scheme

We use the Agrawal et al. lattice IBE scheme [1] as a building block for our construction. Briefly, their IBE scheme can be described as follows.

The public parameters in the scheme of [1] consist of three random $n \times m$ matrices over $\mathbb{Z}_q$ denoted by $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$ as well as a vector $\mathbf{u} \in \mathbb{Z}_q^n$. The master secret is a trapdoor $\mathbf{T_A}$ for the lattice $\Lambda_q^\perp(\mathbf{A})$. The secret key for an identity id is a short vector $\mathbf{e} \in \mathbb{Z}^{2m}$, which is generated using the $\mathsf{SampleLeft}$ algorithm of Theorem 2 and satisfies $\mathbf{F}_{\mathrm{id}}\mathbf{e} = \mathbf{u}$ in $\mathbb{Z}_q$ where $\mathbf{F}_{\mathrm{id}} := (\mathbf{A}|\mathbf{B}+\mathsf{H}(\mathrm{id})\mathbf{C}) \in \mathbb{Z}_q^{n\times 2m}$. In the security proof for a selective IBE security game, the adversary announces an identity id^* that it plans to attack. Instead of using a trapdoor for $\Lambda_q^\perp(\mathbf{A})$, it samples $\mathbf{C}$ at random and obtains a trapdoor $\mathbf{T_C}$ for $\Lambda_q^\perp(\mathbf{C})$. It also chooses the public parameter $\mathbf{A}$ at random and sets $\mathbf{B} := \mathbf{AR} - \mathsf{H}(\mathrm{id}^*)\mathbf{C}$, where $\mathbf{R}$ is a random matrix in $\{1,-1\}^{m\times m}$. Since $\mathbf{AR}$ is uniform and independent in $\mathbb{Z}_q^{n\times m}$, $\mathbf{B}$ is uniformly distributed as required. We then have

$$\mathbf{F}_{\mathrm{id}} := (\mathbf{A}|\mathbf{A}\cdot\mathbf{R} + \mathbf{C}') \in \mathbb{Z}_q^{n\times 2m},$$

where $\mathbf{C}' := (\mathsf{H}(\mathrm{id}) - \mathsf{H}(\mathrm{id}^*))\mathbf{C}$. To respond to a private key query for an identity $\mathrm{id} \neq \mathrm{id}^*$, the simulator could produce a short vector $\mathbf{e}$ satisfying $\mathbf{F}_{\mathrm{id}}\mathbf{e} = \mathbf{u}$ in $\mathbb{Z}_q$ by using the $\mathsf{SampleRight}$ algorithm of Theorem 3 and the basis $\mathbf{T_C}$. This is so since $\mathrm{id} \neq \mathrm{id}^*$ is full rank by the definition of FRD in Section 3.6 and therefore $\mathbf{T_C}$ is also a trapdoor for the lattice $\Lambda_q^\perp(\mathbf{C}')$. When $\mathrm{id} = \mathrm{id}^*$, the matrix $\mathbf{F}_{\mathrm{id}}$ no longer depends on $\mathbf{C}$ and the simulator's trapdoor is removed. The simulator can then produce a challenge ciphertext that helps to solve the given LWE challenge.

4.3 Intuition of Our Construction

We first consider how to create a link between an identity and a time for each node. In our construction, we use two instances of Agrawal et al.'s IBE scheme and its techniques to deal with users' identities and times respectively, but require only a single random vector $\mathbf{u} \in \mathbb{Z}_q^n$ in the public parameters. We split it into two random vectors $\mathbf{u}_1, \mathbf{u}_2$ for each node corresponding to identity and time, respectively. The randomly split $\mathbf{u}$ links identity to time for each node. Moreover, our technique does not require information about $\mathbf{u}_1, \mathbf{u}_2$ to be included in ciphertexts, and hence does not increase the size of ciphertexts.

Clearly, the simulator can answer all private key queries for all identities $\mathrm{id} \neq \mathrm{id}^*$, key update queries for all time $\mathrm{t} \neq \mathrm{t}^*$ by two trapdoors $\mathbf{T}_{\mathbf{C}_1}, \mathbf{T}_{\mathbf{C}_2}$. The main difficulty in the simulation is as follows. The simulator may be required to answer either a key update query at time t^* with node in $\mathsf{Path}(\nu^*)$ or a private key query for identity id^* and a key update query at time t^* without any node in $\mathsf{Path}(\nu^*)$, where id^* is assigned in ν^* (id^* must be revoked before or at time t^*). In other words, the simulator should answer either a query for identity id^* or a query for time t^* for each node. To overcome this difficulty, we use the $\mathsf{SampleGaussian}$ algorithm of Lemma 1 to sample a short vector and generate either $\mathbf{u}_1$ or $\mathbf{u}_2$ (instead of generating one of them randomly). Such $\mathbf{u}_1$ or $\mathbf{u}_2$ is indistinguishable from the uniform distribution, which is guaranteed by Lemma 2. More precisely, there are two possibilities for those nodes in $\mathsf{Path}(\nu^*)$ (we can pick a node ν^* beforehand and assign id^* to it if necessary) depending on whether or not identity id^* will be queried:

- If identity id^* is queried, then it must be revoked before or at time t^*. In this case, we set $\mathbf{u}_1$ to be the product of $\mathbf{F}_{\mathrm{id}^*}$ and a short vector $\mathbf{e}$ sampled by $\mathsf{SampleGaussian}(\mathbf{B}_z, \sigma, 0)$.

– If identity id^* is not queried. In this case, we set $\mathbf{u}_2$ to be the product of $\mathbf{F}_{\mathrm{t}^*}$ and a short vector $\mathbf{e}$ sampled by $\mathsf{SampleGaussian}(\mathbf{B}_z, \sigma, 0)$.

For those nodes that are not in $\mathsf{Path}(\nu^*)$, we set $\mathbf{u}_2$ to be the product of $\mathbf{F}_{\mathrm{t}^*}$ and a short vector $\mathbf{e}$ sampled by $\mathsf{SampleGaussian}(\mathbf{B}_z, \sigma, 0)$. We have probability $1/2$ to simulate the correct game and the adversary cannot distinguish which one is simulated.

4.4 Our RIBE Scheme

We now describe our RIBE scheme from lattices. At the end of each algorithm, we provide some intuition and/or remark (marked by the symbol "//") about the algorithm.

Setup(λ, N) On input a security parameter λ and a maximal number N of users, set the parameters q, n, m, σ, α as specified in Section 4.5 below. Next perform the following steps:

1. Use the $\mathsf{TrapGen}(q, n)$ algorithm to select a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ with a basis $\mathbf{T}_{\mathbf{A}}$ for $\Lambda_q^{\perp}(\mathbf{A})$ such that $||\widetilde{\mathbf{T}_{\mathbf{A}}}|| \leq \mathcal{O}(\sqrt{n \log q})$.
2. Select four uniformly random matrices $\mathbf{B}_1$, $\mathbf{B}_2$, $\mathbf{C}_1$, and $\mathbf{C}_2$ in $\mathbb{Z}_q^{n \times m}$.
3. Select a uniformly random vector $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^n$.
4. Let RL be an empty set and BT be a binary-tree with at least N leaf nodes, set $\mathsf{ST} := \mathsf{BT}$. Select an FRD map H as defined in Section 3.6.
5. Output RL, ST, the public parameters, and the master key MK,

$$\mathsf{PP} := \{\mathsf{H}, \mathbf{A}, \mathbf{B}_1, \mathbf{B}_2, \mathbf{C}_1, \mathbf{C}_2, \mathbf{u}\}, \quad \mathsf{MK} := \{\mathbf{T}_{\mathbf{A}}\}.$$

PriKeyGen$(\mathsf{PP}, \mathsf{MK}, \mathrm{id}, \mathsf{RL}, \mathsf{ST})$ On input the public parameters PP, the master key MK, an identity $\mathrm{id} \in \mathbb{Z}_q^n$, the revocation list RL, and the state ST, it picks an unassigned leaf node ν from BT and stores id in that node. It then performs the following steps:

1. For any $\theta \in \mathsf{Path}(\nu)$, if $\mathbf{u}_{\theta,1}, \mathbf{u}_{\theta,2}$ are undefined, then pick $\mathbf{u}_{\theta,1} \xleftarrow{\$} \mathbb{Z}_q^n$, set $\mathbf{u}_{\theta,2} := \mathbf{u} - \mathbf{u}_{\theta,1}$, and store them in node θ. Sample $\mathbf{e}_{\theta,1} \in \mathbb{Z}^{2m}$ as $\mathbf{e}_{\theta,1} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathbf{B}_1 + \mathsf{H}(\mathrm{id})\mathbf{C}_1, \mathbf{T}_{\mathbf{A}}, \mathbf{u}_{\theta,1}, \sigma)$.
2. Output $\mathsf{SK}_{\mathrm{id}} := \{(\theta, \mathbf{e}_{\theta,1})\}_{\theta \in \mathsf{Path}(\nu)}$, ST.

//The algorithm computes the id-component of the decryption key for all the nodes on the path from ν to root.

KeyUpd$(\mathsf{PP}, \mathsf{MK}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$ On input the public parameters PP, the master key MK, a time $\mathrm{t} \in \mathbb{Z}_q^n$, the revocation list RL, and the state ST, it performs the following steps:

1. $\forall \theta \in \mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL}, \mathrm{t})$, if $\mathbf{u}_{\theta,1}, \mathbf{u}_{\theta,2}$ are undefined, then pick $\mathbf{u}_{\theta,1} \xleftarrow{\$} \mathbb{Z}_q^n$, set $\mathbf{u}_{\theta,2} := \mathbf{u} - \mathbf{u}_{\theta,1}$, and store them in node θ. Sample $\mathbf{e}_{\theta,2} \in \mathbb{Z}^{2m}$ as $\mathbf{e}_{\theta,2} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathbf{B}_2 + \mathsf{H}(\mathrm{t})\mathbf{C}_2, \mathbf{T}_{\mathbf{A}}, \mathbf{u}_{\theta,2}, \sigma)$.
2. Output $\mathsf{KU}_{\mathrm{t}} := \{(\theta, \mathbf{e}_{\theta,2})\}_{\theta \in \mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL}, \mathrm{t})}$.

//The algorithm first finds a minimal set of nodes which contains an ancestor (or, the node itself) of all the non-revoked nodes. It then computes the t-component of the decryption key for all the nodes in that set.

DecKeyGen$(\mathsf{SK}_{\mathrm{id}}, \mathsf{KU}_{\mathrm{t}})$ On input a private secret key $\mathsf{SK}_{\mathrm{id}} := \{(i, \mathbf{e}_{i,1})\}_{i \in \mathsf{I}}$, $\mathsf{KU}_{\mathrm{t}} := \{(j, \mathbf{e}_{j,2})\}_{j \in \mathsf{J}}$ for some set of nodes I, J, it runs the following steps:

1. $\forall(i, \mathbf{e}_{i,1}) \in \mathsf{SK}_{\mathrm{id}}, (j, \mathbf{e}_{j,2}) \in \mathsf{KU}_{\mathrm{t}}$, if $\exists(i, j)$ s.t. $i = j$ then $\mathsf{DK}_{\mathrm{id,t}} \leftarrow (\mathbf{e}_{i,1}, \mathbf{e}_{j,2})$; else (if $\mathsf{SK}_{\mathrm{id}}$ and KU_{t} do not have any node in common) $\mathsf{DK}_{\mathrm{id,t}} \leftarrow \perp$.
2. Output $\mathsf{DK}_{\mathrm{id,t}}$.

// We can drop the subscripts i, j since they are equal, i.e., $\mathsf{DK}_{\mathrm{id,t}} := (\mathbf{e}_1, \mathbf{e}_2)$. The algorithm finds components of $\mathsf{SK}_{\mathrm{id}}$ and KU_{t} such that $\mathbf{F}_{\mathrm{id}}\mathbf{e}_1 + \mathbf{F}_{\mathrm{t}}\mathbf{e}_2 = \mathbf{u}$ since they are in the same node.

Enc$(\mathsf{PP}, \mathrm{id}, \mathrm{t}, \mathrm{m})$ On input the public parameters PP, an identity id, a time $\mathrm{t} \in \mathbb{Z}_q^n$, and a message m, it runs the following steps:

1. Set $\mathbf{F}_{\mathrm{id,t}} \leftarrow (\mathbf{A}|\mathbf{B}_1 + \mathsf{H}(\mathrm{id})\mathbf{C}_1|\mathbf{B}_2 + \mathsf{H}(\mathrm{t})\mathbf{C}_2) \in \mathbb{Z}_q^{n \times 3m}$.
2. Choose a uniformly random $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$.
3. For $i = 1, 2$, choose a uniformly random matrix $\mathbf{R}_i \xleftarrow{\$} \{-1, 1\}^{m \times m}$.
4. Choose noise $x \xleftarrow{\overline{\Psi}_\alpha} \mathbb{Z}_q$ and noise vectors $\mathbf{y} \xleftarrow{\overline{\Psi}_\alpha^m} \mathbb{Z}_q^m$ and for $i = 1, 2$ set $\mathbf{z}_i \leftarrow \mathbf{R}_i^\top \mathbf{y} \in \mathbb{Z}_q^m$. (The distribution $\overline{\Psi}_\alpha$ is as defined by Definition 5)
5. Set $c_0 \leftarrow \mathbf{u}^\top \mathbf{s} + x + \mathrm{m}\lfloor \frac{q}{2} \rfloor \in \mathbb{Z}_q$, $\mathbf{c}_1 \leftarrow \mathbf{F}_{\mathrm{id,t}}^\top \mathbf{s} + \begin{bmatrix} \mathbf{y} \\ \mathbf{z}_1 \\ \mathbf{z}_2 \end{bmatrix} \in \mathbb{Z}_q^{3m}$.
6. Output the ciphertext $\mathsf{CT}_{\mathrm{id,t}} := (c_0, \mathbf{c}_1) \in \mathbb{Z}_q \times \mathbb{Z}_q^{3m}$.

Dec$(\mathsf{PP}, \mathsf{DK}_{\mathrm{id,t}}, \mathsf{CT}_{\mathrm{id,t}})$ On input the public parameters PP, a decryption key $\mathsf{DK}_{\mathrm{id,t}} := (\mathbf{e}_1, \mathbf{e}_2)$, and a ciphertext $\mathsf{CT}_{\mathrm{id,t}} := (c_0, \mathbf{c}_1)$, it runs the following steps:

1. Parse $\mathbf{c}_1$ as $\begin{bmatrix} \mathbf{c}_{1,0} \\ \mathbf{c}_{1,1} \\ \mathbf{c}_{1,2} \end{bmatrix}$, where $\mathbf{c}_{1,i} \in \mathbb{Z}_q^m$.
2. Compute $w \leftarrow c_0 - \mathbf{e}_1^\top \begin{bmatrix} \mathbf{c}_{1,0} \\ \mathbf{c}_{1,1} \end{bmatrix} - \mathbf{e}_2^\top \begin{bmatrix} \mathbf{c}_{1,0} \\ \mathbf{c}_{1,2} \end{bmatrix} \in \mathbb{Z}_q$.
3. Compare w and $\lfloor \frac{q}{2} \rfloor$ treating them as integers in $\mathbb{Z}$. If they are close, i.e., if $\left| w - \lfloor \frac{q}{2} \rfloor \right| < \lfloor \frac{q}{4} \rfloor$, output 1, otherwise output 0.

KeyRev$(\mathrm{id}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$ On input an identity id, a time t, the revocation list RL, and the state ST, the algorithm adds $(\mathrm{id}, \mathrm{t})$ to RL for all nodes ν associated with identity id and returns RL.

4.5 Parameters, Correctness and Security

As in [3], the following error term is bounded by $[q\sigma m\alpha\,\omega(\sqrt{\log m}) + \mathcal{O}(\sigma m^{3/2})]$, that is

$$w = c_0 - \mathbf{e}_1^\top \begin{bmatrix} \mathbf{c}_{1,0} \\ \mathbf{c}_{1,1} \end{bmatrix} - \mathbf{e}_2^\top \begin{bmatrix} \mathbf{c}_{1,0} \\ \mathbf{c}_{1,2} \end{bmatrix} = \mathrm{m}\lfloor \frac{q}{2} \rfloor + \underbrace{x - \mathbf{e}_1^\top \begin{bmatrix} \mathbf{y} \\ \mathbf{z}_1 \end{bmatrix} - \mathbf{e}_2^\top \begin{bmatrix} \mathbf{y} \\ \mathbf{z}_2 \end{bmatrix}}_{\text{error term}}.$$

We can similarly set the parameters (q, m, σ, α) to ensure that the error term is less than $q/5$ and the system works:

$$m = 2n^{1+\delta}, \qquad q = m^2\sqrt{n} \cdot \omega(\log n),$$
$$\sigma = m \cdot \omega(\log n), \qquad \alpha = [m^2 \cdot \omega(\log n)]^{-1},$$

and round up m to the nearest larger integer and q to the nearest larger prime. We choose δ such that $n^\delta > \lceil \log q \rceil = \mathcal{O}(\log n)$.

We show that our RIBE construction is secure in the following theorem:

Theorem 4. *The RIBE system is* IND-sRID-CPA *secure provided that the* $(\mathbb{Z}_q, n, \bar{\Psi}_\alpha)$*-LWE assumption holds.*

Our proof proceeds in a sequence of games where the first game is identical to the IND-sRID-CPA game from Definition 2 and the adversary has no advantage in the last game. We show that a PPT adversary cannot distinguish between the games. This proves that the adversary has negligible advantage in winning the original IND-sRID-CPA game.

To simplify reduction, we divide adversaries into two types:

- Type I adversaries choose to be challenged on the target identity id^* but which is revoked before or on time t^*.
- Type II adversaries do not challenge the target identity id^* at any time.

We pick a random bit $rev \xleftarrow{\$} \{0,1\}$ as a guess for the type of adversarial behavior that it will be faced with. All the sequence of games for these two types of adversaries are similar except for Game 2. We will show that the adversary cannot distinguish which type of challenger simulated and thus we have probability 1/2 to simulate the correct game.

Game 0 This is the original IND-sRID-CPA game from Definition 2.

Game 1 In Game 1 we change the way that the challenger generates $\mathbf{B}_1, \mathbf{B}_2$ in the public parameters. Let id^* be the identity and t^* be the time that $\mathcal{A}$ intends to attack. The Game 1 challenger chooses $\mathbf{R}_1^*, \mathbf{R}_2^*$ at the setup phase and constructs $\mathbf{B}_1, \mathbf{B}_2$ as

$$\mathbf{B}_1 \leftarrow \mathbf{A}\mathbf{R}_1^* - \mathsf{H}(\mathrm{id}^*)\mathbf{C}_1 \tag{1}$$

$$\mathbf{B}_2 \leftarrow \mathbf{A}\mathbf{R}_2^* - \mathsf{H}(\mathrm{t}^*)\mathbf{C}_2 \tag{2}$$

The remainder of the game is unchanged.

We now show that Game 0 is statistically indistinguishable from Game 1. Observe that in Game 1 the matrices $\mathbf{R}_1^*, \mathbf{R}_2^*$ are used only in the construction of $\mathbf{B}_1, \mathbf{B}_2$ and in the construction of the challenge ciphertext where $\mathbf{z}_1 \leftarrow (\mathbf{R}_1^*)^\top \mathbf{y}$, $\mathbf{z}_2 \leftarrow (\mathbf{R}_2^*)^\top \mathbf{y}$. By Lemma 3 the distribution $(\mathbf{A}, \mathbf{A}\mathbf{R}_1^*, \mathbf{z}_1)$ and $(\mathbf{A}, \mathbf{A}\mathbf{R}_2^*, \mathbf{z}_2)$ are statistically close to the distribution $(\mathbf{A}, \mathbf{B}_1', \mathbf{z}_1)$ and $(\mathbf{A}, \mathbf{B}_2', \mathbf{z}_2)$ respectively, where $\mathbf{B}_1', \mathbf{B}_2'$ are uniform $\mathbb{Z}_q^{n \times m}$ matrices. It follows that in the adversary's view, the matrices $\mathbf{A}\mathbf{R}_1^*$ and $\mathbf{A}\mathbf{R}_2^*$ are statistically close to uniform and therefore $\mathbf{B}_1$ and $\mathbf{B}_2$ as defined in the above equation (1)(2) are close to uniform. Hence, $\mathbf{B}_1, \mathbf{B}_2$ in Games 0 and 1 are indistinguishable.

Game 2 In Game 2 we change the way that $\mathbf{u}_{\theta,1}, \mathbf{u}_{\theta,2}$ for each node are chosen and the challenger generates short vectors for private key query of id^* and update key query of t^* as follows (wlog, we can pick a node ν^* from BT beforehand, to which id^* may be assigned):

- If $rev = 0$, we simulate a game to face Type I adversaries. We generate $\mathbf{u}_{\theta,1}$ and $\mathbf{u}_{\theta,2}$ for every node θ in BT as follows:

- if $\theta \in \mathsf{Path}(\nu^*)$, then $\mathbf{e}_{\theta,1} \leftarrow \mathsf{SampleGaussian}(\mathbf{B}_z, \sigma, 0)$. Set $\mathbf{u}_{\theta,1} \leftarrow \mathbf{F}_{\mathrm{id}} \cdot \mathbf{e}_{\theta,1}$ and $\mathbf{u}_{\theta,2} \leftarrow \mathbf{u} - \mathbf{u}_{\theta,1}$. Store them in the node θ.
- if $\theta \notin \mathsf{Path}(\nu^*)$, then $\mathbf{e}_{\theta,2} \leftarrow \mathsf{SampleGaussian}(\mathbf{B}_z, \sigma, 0)$. Set $\mathbf{u}_{\theta,2} \leftarrow \mathbf{F}_{\mathrm{id}} \cdot \mathbf{e}_{\theta,2}$ and $\mathbf{u}_{\theta,1} \leftarrow \mathbf{u} - \mathbf{u}_{\theta,2}$. Store them in the node θ.

Since id^* has been revoked before the update key query for t^* is queried, we have $\mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL}, \mathrm{t}^*) \cap \mathsf{Path}(\nu^*) = \emptyset$. Then the challenger responds a private key query for id^* by using $\{(\theta, \mathbf{e}_{\theta,1})\}_{\theta \in \mathsf{Path}(\nu^*)}$ and a update key query for t^* by using $\{(\theta, \mathbf{e}_{\theta,2})\}_{\theta \in \mathsf{KUNodes}(\mathsf{BT},\mathsf{RL},\mathrm{t}^*)}$.

– If $rev = 1$, we simulate a game to face Type II adversaries. We generate $\mathbf{u}_{\theta,1}$ and $\mathbf{u}_{\theta,2}$ for every node θ in BT as follows:

- $\mathbf{e}_{\theta,2} \leftarrow \mathsf{SampleGaussian}(\mathbf{B}_z, \sigma, 0)$. Set $\mathbf{u}_{\theta,2} \leftarrow \mathbf{F}_{\mathrm{id}}\mathbf{e}_{\theta,2}$ and $\mathbf{u}_{\theta,1} \leftarrow \mathbf{u} - \mathbf{u}_{\theta,2}$. Store them in the node θ.

Since id^* is never queried, the challenger responds a update key query for t^* by using $\{(\theta, \mathbf{e}_{\theta,2})\}_{\theta \in \mathsf{KUNodes}(\mathsf{BT},\mathsf{RL},\mathrm{t}^*)}$.

From Lemma 1, all above $\mathbf{e}_{\theta,1}$ $(i = 1, 2)$ are sampled from $\mathcal{D}_{\mathbb{Z}^m,\sigma}$, which is statistically close to the distribution of $\mathcal{D}_{\Lambda_q^{\mathbf{u}_{\theta,i}}}$ $(i = 1, 2)$ in the real scheme. In particular, we have $\mathbf{e}_{\theta,1} \in \mathcal{D}_{\Lambda_q^{\mathbf{u}_{\theta,1}}(\mathbf{F}_{\mathrm{id}^*}),\sigma}$ and $\mathbf{e}_{\theta,2} \in \mathcal{D}_{\Lambda_q^{\mathbf{u}_{\theta,2}}(\mathbf{F}_{\mathrm{t}^*}),\sigma}$. Since $\mathbf{F}_{\mathrm{id}^*}$ and $\mathbf{F}_{\mathrm{t}^*}$ could be viewed as random matrices from $\mathbb{Z}_q^{n\times 2m}$, we argue that $\mathbf{u}_{\theta,1}$ or $\mathbf{u}_{\theta,2}$ is statistically close to uniform over $\mathbb{Z}_q^n$ from Lemma 2. Hence, the adversary cannot distinguish which type of challenger simulated and we have probability $1/2$ to simulate the correct game. Therefore, Games 1 and 2 are indistinguishable if the correct game is simulated.

Game 3 We now change how $\mathbf{A}$, $\mathbf{C}_1$, $\mathbf{C}_2$ in PP are chosen. In Game 3 we generate $\mathbf{A}$ as random matrix in $\mathbb{Z}_q^{n\times m}$, but generate $\mathbf{C}_1, \mathbf{C}_2$ using algorithm $\mathsf{TrapGen}$ so that $\mathbf{C}_1, \mathbf{C}_2$ is a random matrix in $\mathbb{Z}_q^{n\times m}$, but the challenger has a trapdoor $\mathbf{T}_{\mathbf{C}_1}, \mathbf{T}_{\mathbf{C}_2}$ for $\Lambda_q^{\perp}(\mathbf{C}_1), \Lambda_q^{\perp}(\mathbf{C}_2)$. The choice of $\mathbf{B}_1, \mathbf{B}_2$ remains as in Game 1 and 2, i.e. $\mathbf{B}_1 = \mathbf{A} \cdot \mathbf{R}_1^* - \mathsf{H}(\mathrm{id}^*)\mathbf{C}_1$, $\mathbf{B}_2 = \mathbf{A} \cdot \mathbf{R}_2^* - \mathsf{H}(\mathrm{t}^*)\mathbf{C}_2$.

To respond to a private key query for $\mathrm{id} \neq \mathrm{id}^*$ and key update query $\mathrm{t} \neq \mathrm{t}^*$, we will use the trapdoor $\mathbf{T}_{\mathbf{C}_1}$ and $\mathbf{T}_{\mathbf{C}_2}$ instead of $\mathbf{T}_{\mathbf{A}}$, respectively. Let

$$\mathbf{F}_{\mathrm{id}} := (\mathbf{A}|\mathbf{B}_1 + \mathsf{H}(\mathrm{id})\mathbf{C}_1) = (\mathbf{A}|\mathbf{A}\mathbf{R}_1^* + (\mathsf{H}(\mathrm{id}) - \mathsf{H}(\mathrm{id}^*))\mathbf{C}_1),$$
$$\mathbf{F}_{\mathrm{t}} := (\mathbf{A}|\mathbf{B}_2 + \mathsf{H}(\mathrm{t})\mathbf{C}_2) = (\mathbf{A}|\mathbf{A}\mathbf{R}_2^* + (\mathsf{H}(\mathrm{t}) - \mathsf{H}(\mathrm{t}^*))\mathbf{C}_2),$$

by the definition of FRD in Section 3.6, $[\mathbf{H}(\mathrm{id}) - \mathbf{H}(\mathrm{id}^*)]$ and $[\mathbf{H}(\mathrm{t}) - \mathsf{H}(\mathrm{t}^*)]$ are non-singular. Therefore, $\mathbf{T}_{\mathbf{C}_1}$ and $\mathbf{T}_{\mathbf{C}_2}$ are also trapdoors for $\Lambda_q^{\perp}(\mathbf{C}_1')$ and $\Lambda_q^{\perp}(\mathbf{C}_2')$ respectively, where $\mathbf{C}_1' := (\mathsf{H}(\mathrm{id}) - \mathsf{H}(\mathrm{id}^*))\mathbf{C}_1$ and $\mathbf{C}_2' := (\mathsf{H}(\mathrm{t}) - \mathsf{H}(\mathrm{t}^*))\mathbf{C}_2$. Moreover, since $\mathbf{C}_1, \mathbf{C}_2$ are rank n, so are $\mathbf{C}_1'$ and $\mathbf{C}_2'$. The challenger can now respond to all private key queries for $\mathrm{id} \neq \mathrm{id}^*$ by running

$$\mathbf{e}_{\theta,1} \leftarrow \mathsf{SampleRight}(\mathbf{A}, \mathbf{C}_1', \mathbf{R}_1^*, \mathbf{T}_{\mathbf{C}_1}, \mathbf{u}_{\theta,1}, \sigma) \in \mathbb{Z}_q^{2m},$$

and key update queries for $\mathrm{t} \neq \mathrm{t}^*$ by running

$$\mathbf{e}_{\theta,2} \leftarrow \mathsf{SampleRight}(\mathbf{A}, \mathbf{C}_2', \mathbf{R}_2^*, \mathbf{T}_{\mathbf{C}_2}, \mathbf{u}_{\theta,2}, \sigma) \in \mathbb{Z}_q^{2m}.$$

Since the σ used in the system is sufficiently large, these $\mathbf{e}_{\theta,1}$ and $\mathbf{e}_{\theta,2}$ are distributed close to $\mathcal{D}_{\Lambda_q^{\mathbf{u}_{\theta,1}}(\mathbf{F}_{\mathrm{id}}),\sigma}$ and $\mathcal{D}_{\Lambda_q^{\mathbf{u}_{\theta,2}}(\mathbf{F}_{\mathrm{t}}),\sigma}$ as in Game 2 by Theorem 3. The challenger responds to private key

query for id^* and key update query for t^* the same as in Game 2 rather than being sampled with $\mathsf{SampleRight}$, where these $\mathbf{e}_{\theta,1}$ and $\mathbf{e}_{\theta,2}$ have been already defined in Game 2.

Game 3 is otherwise the same as Game 2. Since $\mathbf{A}$, $\mathbf{C}_1$, $\mathbf{C}_2$ and responses to private key and key update queries are statistically close to those in Game 2, the adversary's advantage in Game 3 is at most negligibly different from its advantage in Game 2.

Game 4 Game 4 is identical to Game 3 except that the challenge ciphertext $(c_0^*, \mathbf{c}_1^*)$ is always chosen as a random independent element in $\mathbb{Z}_q \times \mathbb{Z}_q^{3m}$. Since the challenge ciphertext is always a fresh random element in the ciphertext space, $\mathcal{A}$'s advantage in this game is zero.

It remains to show that Game 3 and Game 4 are computationally indistinguishable for a PPT adversary, which we do by giving a reduction from the LWE problem.

Reduction from LWE. Suppose $\mathcal{A}$ has non-negligible advantage in distinguishing Games 3 and 4. We use $\mathcal{A}$ to construct an LWE algorithm $\mathcal{B}$. Recall from Definition 3.5 that an LWE problem instance is provided as a sampling oracle $\mathcal{O}$ which can be either truly random $\mathcal{O}_\$$ or a noisy pseudorandom $\mathcal{O}_{\mathbf{s}}$ for some secret $\mathbf{s} \in \mathbb{Z}_q^n$. The simulator $\mathcal{B}$ uses the adversary $\mathcal{A}$ to distinguish between the two, and proceeds as follows:

Instance: $\mathcal{B}$ requests from $\mathcal{O}$ and receives, for each $i = 0, \ldots, m$, a fresh pair $(\mathbf{u}_i, v_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.

Initial: $\mathcal{A}$ announces to $\mathcal{B}$ the identity id^* and time t^* that it intends to attack.

Setup: $\mathcal{B}$ constructs the system's public parameters PP as follows:

1. Assemble the random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ from m of the previously given LWE samples by letting the j-th column of $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ be the n-vector $\mathbf{u}_j$ for all $j = 1, \ldots, m$.
2. Assign the zeroth LWE sample (so far unused) to become the public random n-vector $\mathbf{u}_0 \in \mathbb{Z}_q^n$.
3. The remainder of the public parameters, namely $\mathbf{B}_1$, $\mathbf{C}_1$, $\mathbf{B}_2$, and $\mathbf{C}_2$, are constructed as in Game 3 using id^*, $\mathbf{R}_1^*$, t^*, and $\mathbf{R}_2^*$.

Query: $\mathcal{B}$ answers each private key and key update queries as in Game 3.

Challenge: $\mathcal{B}$ prepares, when prompted by $\mathcal{A}$ with a message bit $\mathrm{m}^* \in \{0, 1\}$, a challenge ciphertext for the target identity id^*, as follows:

1. Let $v_0, \ldots, v_m$ be entries from the LWE instance.
   Set $\mathbf{v}^* = \begin{bmatrix} v_1 \\ \vdots \\ v_m \end{bmatrix} \in \mathbb{Z}_q^m$.
2. Blind the message bit by letting $c_0^* = v_0 + \mathrm{m}^* \lfloor \frac{q}{2} \rfloor \in \mathbb{Z}_q$
3. Set $\mathbf{c}_1^* = \begin{bmatrix} \mathbf{v}^* \\ (\mathbf{R}_1^*)^\top \mathbf{v}^* \\ (\mathbf{R}_2^*)^\top \mathbf{v}^* \end{bmatrix} \in \mathbb{Z}_q^{3m}$.
4. Choose a random bit $r \xleftarrow{\$} \{0, 1\}$. If $r = 0$ send $(c_0^*, \mathbf{c}_1^*)$ to the adversary. If $r = 1$ choose a random $(c_0, \mathbf{c}_1) \in \mathbb{Z}_q \times \mathbb{Z}_q^{3m}$ and send $(c_0, \mathbf{c}_1)$ to the adversary.

We argue that when the LWE oracle is pseudorandom (i.e. $\mathcal{O} = \mathcal{O}_{\mathbf{s}}$) then $(c_0^*, \mathbf{c}_1^*)$ is distributed exactly as in Game 3. First, observe that $\mathbf{F}_{\mathrm{id}^*} = (\mathbf{A} | \mathbf{A}\mathbf{R}_1^*)$ and $\mathbf{F}_{\mathrm{t}^*} = (\mathbf{A} | \mathbf{A}\mathbf{R}_2^*)$. Second, by definition of $\mathcal{O}_s$ we know that $\mathbf{v}^* = \mathbf{A}^\top \mathbf{s} + \mathbf{y}$ for some random noise vector $\mathbf{y} \in \mathbb{Z}_q^m$ distributed as

$\bar{\Psi}_\alpha^m$. Therefore, $\mathbf{c}_1^*$ defined in step (3) above satisfies

$$\mathbf{c}_1^* = \begin{bmatrix} \mathbf{A}^\top \mathbf{s} + \mathbf{y} \\ (\mathbf{R}_1^*)^\top \mathbf{A}^\top \mathbf{s} + (\mathbf{R}_1^*)^\top \mathbf{y} \\ (\mathbf{R}_2^*)^\top \mathbf{A}^\top \mathbf{s} + (\mathbf{R}_2^*)^\top \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{A}^\top \mathbf{s} + \mathbf{y} \\ (\mathbf{A}\mathbf{R}_1^*)^\top \mathbf{s} + (\mathbf{R}_1^*)^\top \mathbf{y} \\ (\mathbf{A}\mathbf{R}_2^*)^\top \mathbf{s} + (\mathbf{R}_2^*)^\top \mathbf{y} \end{bmatrix} = (\mathbf{F}_{\mathrm{id}^*,\mathrm{t}^*})^\top \mathbf{s} + \begin{bmatrix} \mathbf{y} \\ (\mathbf{R}_1^*)^\top \mathbf{y} \\ (\mathbf{R}_2^*)^\top \mathbf{y} \end{bmatrix}$$

and the quantity on the right is precisely the id_1 part of a valid challenge ciphertext in Game 3. Also note that $v_0 = \mathbf{u}_0^\top \mathbf{s} + x$, just as the c_0 part of the challenge ciphertext in Game 3.

When $\mathcal{O} = \mathcal{O}_\$$ we have that v_0 is uniform in $\mathbb{Z}_q$ and $\mathbf{v}^*$ are uniform in $\mathbb{Z}_q^m$. Therefore $\mathbf{c}_1^*$ as defined in step (3) above is uniform and independent in $\mathbb{Z}_q^{3m}$ by the standard left over hash lemma (e.g. Theorem 8.38 of [37]). Hence, the challenge ciphertext is always uniform in $\mathbb{Z}_q \times \mathbb{Z}_q^{3m}$, as in Game 4.

Guess: After being allowed to make additional queries, $\mathcal{A}$ guesses if it is interacting with a Game 3 or Game 4 challenger. Our simulator outputs $\mathcal{A}$'s guess as the answer to the LWE challenge it is trying to solve.

We already argued that when $\mathcal{O} = \mathcal{O}_{\mathbf{s}}$ the adversary's view is as in Game 3. When $\mathcal{O} = \mathcal{O}_\$$ the adversary's view is as in Game 4. Hence, $\mathcal{B}$'s advantage in solving LWE is the same as $\mathcal{A}$'s advantage in distinguishing Games 3 and 4, as required. This completes the description of algorithm $\mathcal{B}$ and completes the proof.

5 Extensions and Open Problem

5.1 FIBE and HVE

It is not difficult to extend our techniques to obtain a d-out-of-d fuzzy identity-based encryption (FIBE) scheme and a hidden vector encryption (HVE) scheme [13]. Briefly, we require d instances of Agrawal et al.'s IBE scheme and randomly split the single vector $\mathbf{u}$ into d parts. To construct a d-out-of-d FIBE scheme, we sort d identities alphabetically when generate secret keys and ciphertexts. This way, d identities associated with a secret key can always match an associated ciphertext. Moreover, to construct an HVE scheme, we generate a secret key for each wildcard position by sampling a short vector from $\Lambda_q^{\mathbf{u}_i}(A)$, where $\mathbf{u}_i$ is the associated share. The resulting FIBE and HVE schemes may be of independent interest, although they can also be obtained from Agrawal et al.'s HIBE scheme [1].

5.2 Revocable Functional Encryption

Our techniques can also be used to enhance the IPE scheme of [5] with key revocation. Briefly, we assign each vector (associated with private key) to a leaf node of a complete binary tree. We then use an instance of the IPE scheme for vector and an instance of Agrawal et al.'s IBE scheme to deal with time periods. Moreover, we require that only a random n-vector $\mathbf{u}$ in used for the public parameters. The resulting revocable IPE scheme can be proven using similar proof techniques as ours.

5.3 Open Problem

We have proven our RIBE scheme to be selective-ID secure under the LWE assumption. However, we leave open the problem of how to construct an adaptive-ID secure RIBE scheme [26].

Acknowledgments. We thank the referees for helpful feedback. Research of the authors is supported in part by the National Research Foundation of Singapore under Research Grant NRF-CRP2-2007-03.

References

[1] S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (h)ibe in the standard model. In *EUROCRYPT*, pages 553–572, 2010.

[2] S. Agrawal, D. Boneh, and X. Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical ibe. In *CRYPTO*, pages 98–115, 2010.

[3] S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (h)ibe in the standard model. http://crypto.stanford.edu/∼dabo/pubs/papers/latticebb.pdf..

[4] S. Agrawal, X. Boyen, V. Vaikuntanathan, P. Voulgaris, and H. Wee. Fuzzy identity based encryption from lattices. *IACR Cryptology ePrint Archive*, 2011:414, 2011.

[5] S. Agrawal, D. M. Freeman, and V. Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In *ASIACRYPT*, pages 21–40, 2011.

[6] W. Aiello, S. Lodha, and R. Ostrovsky. Fast digital identity revocation (extended abstract). In *CRYPTO*, pages 137–152, 1998.

[7] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC*, pages 99–108, 1996.

[8] M. Ajtai. Generating hard instances of the short basis problem. In *ICALP*, pages 1–9, 1999.

[9] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *STOC*, pages 284–293, 1997.

[10] J. Alwen and C. Peikert. Generating shorter bases for hard random lattices. In *STACS*, pages 75–86, 2009.

[11] A. Boldyreva, V. Goyal, and V. Kumar. Identity-based encryption with efficient revocation. In *ACM Conference on Computer and Communications Security*, pages 417–426, 2008.

[12] D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003.

[13] D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC*, pages 535–554, 2007.

[14] D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In *EUROCRYPT*, pages 440–456, 2005.

[15] D. Boneh, C. Gentry, and M. Hamburg. Space-efficient identity based encryption without pairings. In *FOCS*, pages 647–657, 2007.

[16] D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In *TCC*, pages 253–273, 2011.

[17] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, pages 523–552, 2010.

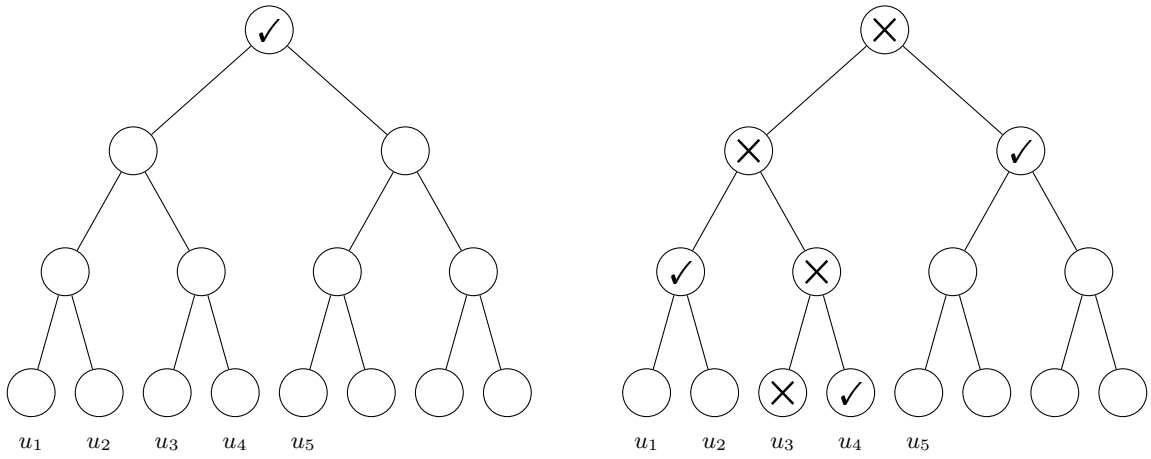
[18] C. Cocks. An identity based encryption scheme based on quadratic residues. In *IMA Int. Conf.*, pages 360–363, 2001.

[19] R. Cramer and I. Damgård. On the amortized complexity of zero-knowledge protocols. In *CRYPTO*, pages 177–191, 2009.

[20] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.

[21] C. Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.
[22] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.
[23] J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *EUROCRYPT*, pages 146–162, 2008.
[24] A. B. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *EUROCRYPT*, pages 62–91, 2010.
[25] B. Libert and D. Vergnaud. Towards black-box accountable authority ibe with short ciphertexts and private keys. In *Public Key Cryptography*, pages 235–255, 2009.
[26] B. Libert and D. Vergnaud. Adaptive-id secure revocable identity-based encryption. In *CT-RSA*, pages 1–15, 2009.
[27] D. Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions from worst-case complexity assumptions. In *FOCS*, pages 356–365, 2002.
[28] D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. *IACR Cryptology ePrint Archive*, 2011:501, 2011.
[29] M. Naor and K. Nissim. Certificate revocation and certificate update. *IEEE Journal on Selected Areas in Communications*, 18(4):561–570, 2000.
[30] T. Okamoto and K. Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *CRYPTO*, pages 191–208, 2010.
[31] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *STOC*, pages 333–342, 2009.
[32] O. Regev. New lattice-based cryptographic constructions. *J. ACM*, 51(6):899–942, 2004.
[33] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93, 2005.
[34] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473, 2005.
[35] R. Sepahi, R. Steinfeld, and J. Pieprzyk. Lattice-based completely non-malleable pke in the standard model (poster). In *ACISP*, pages 407–411, 2011.
[36] A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53, 1984.
[37] V. Shoup. *A computational introduction to number theory and algebra*. Cambridge University Press, 2008. ISBN 9780521516440.
[38] B. Waters. Efficient identity-based encryption without random oracles. In *EUROCRYPT*, pages 114–127, 2005.
[39] B. Waters. Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. In *CRYPTO*, pages 619–636, 2009.

A Figure 1



(a) No user is revoked

(b) User u_3 is revoked

Fig. 1. A graphical description of the KUNodes algorithm.