Revocable Identity-Based Encryption from Lattices

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

Division of Mathematical Sciences
School of Physical & Mathematical Sciences
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.

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

1 Introduction

The concept of *identity-based encryption* (IBE) was proposed by Shamir [31]. 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 [11] on practical pairing-based IBE systems, see for example [12, 32, 33]. Meanwhile, there also exist proposals on IBE systems based on quadratic residuosity [16, 13], 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 [20, 14, 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 [6, 23, 8, 27, 28, 19]. 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

W. Susilo, Y. Mu, and J. Seberry (Eds.): ACISP 2012, LNCS 7372, pp. 390–403, 2012.

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 [11] 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 [10] 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 [5, 25], and builds on a fuzzy IBE (FIBE) scheme introduced by Sahai and Waters [29] 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 [22] subsequently proposed an RIBE scheme in the adaptive-ID model using similar key revocation techniques as with [10]. However, instead of making use of an FIBE scheme, they adopt a variant [21] of the Waters IBE scheme [32]. 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 [20]; and (iii) the binary-tree data structure for key update used in [5, 25, 10, 22]. More specifically, we extend the lattice IBE scheme of [1] with trapdoors from [20] 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 [28, 26]. 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 $\mathsf{id} \neq \mathsf{id}^*$ and times $\mathsf{t} \neq \mathsf{t}^*$. We address this by adopting the simulation trapdoors of [20]. 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^*.

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. [30].

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 [10].

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 $\mathcal{T}$.*

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 $(\mathsf{PP}, \mathsf{MK})$ output by **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 by time t then, for $(\mathsf{SK}_{\mathrm{id}}, \mathsf{ST}) \overset{\$}{\leftarrow}$ **PriKeyGen**$(\mathsf{PP}, \mathsf{MK}, \mathrm{id}, \mathsf{ST})$, $\mathsf{KU}_{\mathrm{t}} \overset{\$}{\leftarrow}$ **KeyUpd**$(\mathsf{PP}, \mathsf{MK}, \mathrm{t}, \mathsf{RL}, \mathsf{ST})$, $\mathsf{DK}_{\mathrm{id,t}} \leftarrow$ **DecKeyGen**$(\mathsf{SK}_{\mathrm{id}}, \mathsf{KU}_{\mathrm{t}})$ we have **Dec**$(\mathsf{PP}, \mathsf{DK}_{\mathrm{id,t}},$ **Enc**$(\mathsf{PP}, \mathrm{id}, \mathrm{t}, \mathrm{m})) = \mathrm{m}$.

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* IND-sRID-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 $\mathbf{B} := [\mathbf{b}_1|\dots|\mathbf{b}_m] \in \mathbb{R}^{m\times m}$ be an $m \times m$ matrix whose columns are linearly independent vectors $\mathbf{b}_1, \dots, \mathbf{b}_m \in \mathbb{R}^m$. 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:*

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

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., $\|S\| := \mathsf{max}_i \sqrt{s_{i,1}^2 + \ldots + s_{i,m}^2}$ 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 [7, 9]. Here we use a better result with tighter parameters which was recently discovered by Micciancio and Peikert [24].

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 [20] 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 $\mathbf{u} = \mathbf{A}\mathbf{e}$ 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 SampleLeft [14, 1] and 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 $\mathbf{F} := (\mathbf{A}|\mathbf{A}\mathbf{R} + \mathbf{C}) \in \mathbb{Z}_q^{n \times 2m}$. 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 $\mathbf{T}_{\mathbf{A}}$ of $\Lambda_q^{\perp}(\mathbf{A})$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, and a gaussian parameter $\sigma > \|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \cdot \omega(\sqrt{\log(m + m_1)})$. 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 $\mathbf{T}_{\mathbf{C}}$ of $\Lambda_q^{\perp}(\mathbf{C})$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, and a gaussian parameter $\sigma > \|\widetilde{\mathbf{T}_{\mathbf{C}}}\| \cdot \sqrt{m}\omega(\log(m))$. 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 $\mathbf{F}_2 := (\mathbf{A}|\mathbf{A}\mathbf{R} + \mathbf{C})$. 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. [18], 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 $(\mathbf{A}, \mathbf{A}\mathbf{R}, \mathbf{R}^{\top}\mathbf{w})$ 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 [28].

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 [28] and Peikert [26] 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* $\lfloor x \rceil := \lfloor x + \frac{1}{2} \rceil$ *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* $\lfloor xX \rceil$ *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, 17].

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 [5, 25, 10, 22]. 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 $\leq \mathrm{t}$ (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 $\mathsf{X}$
  $\forall \theta \in \mathsf{X}$
    if $\theta_\ell \notin \mathsf{X}$ then add $\theta_\ell$ to $\mathsf{Y}$
    if $\theta_r \notin \mathsf{X}$ then add $\theta_r$ to $\mathsf{Y}$
  If $\mathsf{Y} = \emptyset$ then add $\mathsf{root}$ to $\mathsf{Y}$
  Return $\mathsf{Y}$
```

The $\mathsf{KUNodes}$ algorithm marks all the ancestors of revoked nodes as revoked and outputs all the non-revoked children of revoked nodes.

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 $\mathbf{T_A}$ for $\Lambda_q^{\perp}(\mathbf{A})$ such that $||\widetilde{\mathbf{T_A}}|| \leq \mathcal{O}(\sqrt{n \log q})$.
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 $\mathbf{u} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$.
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,

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

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 $\mathbf{u}_{\theta,1} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$, 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 $\mathbf{e}_{\theta,1} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathbf{B}_1 + \mathsf{H}(\mathrm{id})\mathbf{C}_1, \mathbf{T_A}, \mathbf{u}_{\theta,1}, \sigma)$.
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 $\mathbf{u}_{\theta,1} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$, 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 $\mathbf{e}_{\theta,2} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathbf{B}_2 + \mathsf{H}(\mathrm{t})\mathbf{C}_2, \mathbf{T_A}, \mathbf{u}_{\theta,2}, \sigma)$.
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 $\mathbf{s} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$.
3. For $i = 1, 2$, choose a uniformly random matrix $\mathbf{R}_i \overset{\$}{\leftarrow} \{-1, 1\}^{m\times m}$.
4. Choose noise $x \overset{\overline{\Psi}_\alpha}{\longleftarrow} \mathbb{Z}_q$ and noise vectors $\mathbf{y} \overset{\overline{\Psi}_\alpha^m}{\longleftarrow} \mathbb{Z}_q^m$ 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.*

We have given some intuition of our security proof earlier in Section 4.3. Due to space constraints, the detail is given in the full version of this paper [15].

5 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 [22].

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] Agrawal, S., Boneh, D., Boyen, X.: Efficient Lattice (H)IBE in the Standard Model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010)

[2] Agrawal, S., Boneh, D., Boyen, X.: Lattice Basis Delegation in Fixed Dimension and Shorter-Ciphertext Hierarchical IBE. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 98–115. Springer, Heidelberg (2010)

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

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

[5] Aiello, W., Lodha, S.P., Ostrovsky, R.: Fast Digital Identity Revocation. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 137–152. Springer, Heidelberg (1998)

[6] Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: STOC, pp. 99–108 (1996)

[7] Ajtai, M.: Generating Hard Instances of the Short Basis Problem. In: Wiedermann, J., Van Emde Boas, P., Nielsen, M. (eds.) ICALP 1999. LNCS, vol. 1644, pp. 1–9. Springer, Heidelberg (1999)

[8] Ajtai, M., Dwork, C.: A public-key cryptosystem with worst-case/average-case equivalence. In: STOC, pp. 284–293 (1997)

[9] Alwen, J., Peikert, C.: Generating shorter bases for hard random lattices. In: STACS, pp. 75–86 (2009)

[10] Boldyreva, A., Goyal, V., Kumar, V.: Identity-based encryption with efficient revocation. In: ACM Conference on Computer and Communications Security, pp. 417–426 (2008)

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

[12] Boneh, D., Boyen, X., Goh, E.-J.: Hierarchical Identity Based Encryption with Constant Size Ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005)

[13] Boneh, D., Gentry, C., Hamburg, M.: Space-efficient identity based encryption without pairings. In: FOCS, pp. 647–657 (2007)

[14] Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai Trees, or How to Delegate a Lattice Basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010)

[15] Chen, J., Lim, H.W., Ling, S., Wang, H., Nguyen, T.T.K.: Revocable identity-based encryption from lattices. IACR Cryptology ePrint Archive, 2011/583 (2011)

[16] Cocks, C.: An identity based encryption scheme based on quadratic residues. In: IMA Int. Conf., pp. 360–363 (2001)

[17] Cramer, R., Damgård, I.: On the Amortized Complexity of Zero-Knowledge Protocols. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 177–191. Springer, Heidelberg (2009)

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

[19] Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC, pp. 169–178 (2009)

[20] Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC, pp. 197–206 (2008)

[21] Libert, B., Vergnaud, D.: Towards Black-Box Accountable Authority IBE with Short Ciphertexts and Private Keys. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 235–255. Springer, Heidelberg (2009)

[22] Libert, B., Vergnaud, D.: Adaptive-ID Secure Revocable Identity-Based Encryption. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 1–15. Springer, Heidelberg (2009)

[23] Micciancio, D.: Generalized compact knapsacks, cyclic lattices, and efficient one-way functions from worst-case complexity assumptions. In: FOCS, pp. 356–365 (2002)

[24] Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. IACR Cryptology ePrint Archive, 2011/501 (2011)

[25] Naor, M., Nissim, K.: Certificate revocation and certificate update. IEEE Journal on Selected Areas in Communications 18(4), 561–570 (2000)

[26] Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: STOC, pp. 333–342 (2009)

[27] Regev, O.: New lattice-based cryptographic constructions. J. ACM 51(6), 899–942 (2004)

[28] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC, pp. 84–93 (2005)

[29] Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)

[30] Sepahi, R., Steinfeld, R., Pieprzyk, J.: Lattice-Based Completely Non-malleable PKE in the Standard Model (Poster). In: Parampalli, U., Hawkes, P. (eds.) ACISP 2011. LNCS, vol. 6812, pp. 407–411. Springer, Heidelberg (2011)

[31] Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)

[32] Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)

[33] Waters, B.: Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)