Revocable Identity-based Encryption from Codes with Rank Metric[⋆]

Donghoon Chang[1], Amit Kumar Chauhan[2], Sandeep Kumar[1,3], and Somitra Kumar Sanadhya[2]

[1] Indraprastha Institute of Information Technology (IIIT-D), Delhi, India
{donghoon,sandeepk}@iiitd.ac.in

[2] Indian Institute of Technology, Ropar, India
{2017csz0008,somitra}@iitrpr.ac.in

[3] Department of Mathematics, Shaheed Bhagat Singh College, University of Delhi, India
sandeep_kumar@sbs.du.ac.in

Abstract. In this paper, we present an identity-based encryption scheme from codes with efficient key revocation. Recently, in Crypto 2017, Gaborit et al. proposed a first identity-based encryption scheme from codes with rank metric, called RankIBE. To extract the decryption key from any public identity, they constructed a trapdoor function which relies on RankSign, a signature scheme proposed by Gaborit et al. in PQCrypto 2014. We adopt the same trapdoor function to add efficient key revocation functionality in the RankIBE scheme. Our revocable IBE scheme from codes with rank metric makes use of a binary tree data structure to reduce the amount of work in terms of key updates for the key authority. The total size of key updates requires logarithmic complexity in the maximum number of users and linear in the number of revoked users. We prove that our revocable IBE scheme is selective-ID secure in the random oracle model, under the hardness of three problems: the Rank Syndrome Decoding (RSD) problem, the Augmented Low Rank Parity Check Code (LRPC$^+$) problem, and the Rank Support Learning (RSL) problem.

Keywords : Code-based Cryptography, Identity-based Encryption, Key Revocation, Rank Metric, LRPC Codes, RSD Problem.

1 Introduction

The security of traditional public-key cryptosystems relies mainly on the hardness of factoring large integers, solving discrete logarithmic problems, etc. In the presence of quantum computers, these hard problems would be solvable in polynomial time using Shor's algorithm [33]. Therefore, it is the need of time to design and analyze post-quantum secure cryptosystems, the importance of which has also been reflected in the efforts made by NIST for standardization of post-quantum secure cryptographic protocols [11]. The currently known post-quantum secure cryptosystem emerge from one of these fields: lattice-based, code-based, hash-based and multivariate polynomial based cryptosystems. In particular, our proposed revocable identity-based encryption scheme relies on hard problems from codes with rank metric.

Code-based Cryptography. The history of code based cryptography is as old as of public key cryptography. The first code based encryption scheme relying on Hamming metric, McEliece cryptosystem [27] was introduced in 1978, that uses binary Goppa codes. Its security is based on indistinguishability of Goppa codes from random codes and the inherent complexity of decoding a random linear code, which is considered to be NP-complete [7].

[⋆] A preliminary version of this paper appears in the proceedings of the RSA Conference Cryptographers' Track 2018 (CT-RSA 2018). This is the full version.

Although it provides fast encryption and decryption procedures, it requires an extremely large public key. Till date, the original proposal made by McEliece has been extensively analyzed and unbroken, but the large public key size makes it impractical to use. Various attempts have been made to overcome this drawback, in terms of using quasi-cyclic codes with different underlying algebraic code, mainly subfamilies of alternant codes [6, 17, 28]. However, most of these were broken by using structural attacks [15]. To prevent the structural attacks and to reduce the public key size, low-density parity-check codes (LDPC) [23] with quasi-cyclic parity check matrix were introduced and analyzed in [3–5]. The drawback of LDPC codes is that the low weight rows can be seen as low weight codewords in the dual code [30]. In 2013, a promising variant of McEliece cryptosystem (with small key) based on quasi-cyclic moderate density parity-check codes (QC-MDPC) [29] was introduced with a security reduction to syndrome decoding problem for a random quasi-cyclic linear code.

In 1985, Gabidulin [16] proposed rank metric as an alternative to Hamming metric. Indeed, Gabidulin [16] showed that it is possible to construct a rank analogue of Reed-Solomon codes, called Gabidulin codes. The generic syndrome decoding problem for the rank metric is considered to be harder than for the Hamming metric. Many variants of McEliece cryptosystem were proposed based on different masking scheme of Gabidulin codes, but most of these were broken by using structural attacks because of the strong algebraic structure of these codes. To avoid structural attacks, Gaborit et al. [19] introduced Low Rank Parity Check (LRPC) codes, similar to LDPC/MDPC codes. One of the major advantages of LRPC codes is that the decoding error probability can be made arbitrarily small by choosing suitable parameters. Moreover, the complexity of best-known attacks against rank-metric based cryptosystems grows very quickly with the size of parameters. It is possible to obtain a general instance of the rank syndrome decoding problem for (say) 2^{80} security with small public key [19].

Identity-based Cryptography. The idea of identity-based cryptography was first introduced by Shamir [32] in 1984, where the public key of a user is his identity (e.g., email address). The private key corresponds to public identity is issued by a trusted authority called private key generator (PKG), who has the knowledge of some extra secret information to generate private keys. This simplifies the public key infrastructure (PKI) and eliminates the requirement of certificate authorities. In his seminal work, Shamir also proposed a concrete implementation of identity-based signature (IBS) scheme. However, he conjectured that the identity-based encryption (IBE) scheme exists as well. In 2001, Boneh and Franklin [9] proposed a fully functional IBE, built on elliptic curves with bilinear pairings. In 2010, Agrawal et al. [1] proposed an efficient IBE based on lattices. Recently, in 2017, Gaborit et al. [18] also proposed a solution to a long standing open problem of building an IBE from codes.

The problem of efficient revocation, has been widely studied in both PKI and IBE settings. In the IBE setting, Boneh and Franklin [9] suggested that users renew their private key periodically. However, their proposal requires PKG has to be online for the process of key updates and keeping the PKG online can be a bottleneck for a vast number of users. In 2008, Boldyreva et al. [8] significantly improved the technique suggested by Boneh and Franklin [9] and reduced the authority's periodic workload to be logarithmic (instead of linear) in the number of users while keeping the scheme efficient for senders and receivers. Their revocable IBE scheme [8] uses a binary tree data structure. A similar idea of building a revocable IBE scheme from lattices was adapted by Chen et al. [10]. They extended IBE scheme of Agrawal et al. [1] to revocable IBE by adopting binary tree data structure. Later, Wang and Bi [34] also introduced an identity-based broadcast encryption from lattice-based delegation technique. Motivated by all these developments, we also build an efficiently revocable IBE scheme from codes.

1.1 Our Results

We construct a revocable IBE (RIBE) from codes with rank metric in the random oracle model. Our construction of RIBE makes use of the following building blocks: (i) IBE from codes with rank metric [18]; (ii) trapdoors using RankSign from codes with rank metric [21]; and (iii) the binary tree data structure for key update used in [2,8,10,25,31].

We note that our RIBE scheme is not a straightforward combination of the aforementioned building blocks since we require that a user's public key consists of two components: identity (id) and time (t), in order to obtain the non-interactive key revocation procedure. Thus, our construction requires two instances of Gaborit et al.'s IBE scheme to deal with identities and times respectively. Furthermore, Gaborit et al.'s IBE requires the public key as $(\boldsymbol{A}, \boldsymbol{G})$, whereas our RIBE requires the public key $(\boldsymbol{A}, \boldsymbol{G}, \boldsymbol{u})$. We require an extra random vector $\boldsymbol{u}$ to link the identity with time for each node associated to the binary tree. Briefly speaking, this can be achieved by randomly splitting the vector $\boldsymbol{u}$ into two vectors $\boldsymbol{u}_1$, $\boldsymbol{u}_2$ for each node corresponding to identity and time, respectively. A similar idea is also used in lattice-based RIBE construction of Chen et al. [10]. However, embedding identity and time attributes with additive shares $\boldsymbol{u}_1$, $\boldsymbol{u}_2$ of vector $\boldsymbol{u}$ in our construction is a different approach.

Though the key generation process is different in our RIBE scheme but the encryption and decryption process is quite similar to IBE [18], in terms of construction and computational requirement. We are able to add revocable functionality without any increase in the size of ciphertext. The inclusion of binary tree data structure improves the efficiency of secret key updates. The key authority needs to perform key updates which has logarithmic complexity in the maximal number of users and linear complexity in the number of revoked users.

We prove that our RIBE scheme is selective-ID secure in the random oracle model. The security of RIBE relies on three hard problems: Rank Syndrome Decoding (RSD) problem, Rank Support Learning (RSL) problem and the Augmented Low Rank Parity Check Code (LRPC$^+$) problem.

1.2 Organization of the Paper

This paper is organized as follows. Section 2 presents the basic definitions, Section 3 covers the background on codes with rank metric, RankSign signature scheme, and how to sample secrets using trapdoors. We then describe the construction of revocable IBE (RIBE) from codes with rank metric in Section 4. Section 4.3 proves that RIBE is IND-sRID-CPA secure in the random oracle model and Section 4.4 suggests general parameters. Section 5 concludes the work. In Appendix A, we discuss the hardness of rank syndrome decoding problem. In Appendix B, we state and prove the lemma for the indistinguishability of syndromes generated using augmented parity-check matrix. In Appendix C, we provide the complete security proof of our RIBE.

2 Definitions

2.1 Notation

Let $\mathbb{N}$ denote the set of natural numbers and $\{0,1\}^*$ denotes the set of all binary strings of finite length. We let $\lambda \in \mathbb{N}$ to be a security parameter. We say that a function $\epsilon : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is negligible if $\epsilon(\lambda)$ is smaller than all polynomial fractions for sufficiently large λ. We say that an event happens with overwhelming probability if it happens with probability at least $1 - \epsilon(\lambda)$ for some negligible function ϵ. If S is a finite set then $x \overset{\$}{\leftarrow} S$ denotes that x is selected uniformly at random from S. If $\mathcal{D}$ is a distribution, $x \leftarrow \mathcal{D}$ denotes that x is chosen at random according to $\mathcal{D}$. Let q denote a power of prime p. The finite field with q elements is denoted by $\mathbb{F}_q$ and more generally for any positive integer m the finite field with q^m elements is denoted by $\mathbb{F}_{q^m}$. We use bold lowercase and capital letters to denote vectors and matrices respectively. For two matrices $\boldsymbol{A}$, $\boldsymbol{B}$ of compatible dimensions, we let $(\boldsymbol{A}|\boldsymbol{B})$ and $\left(\frac{\boldsymbol{A}}{\boldsymbol{B}}\right)$ respectively denote the horizontal and vertical concatenations of $\boldsymbol{A}$ and $\boldsymbol{B}$.

2.2 Syntax of Revocable IBE

Here, we recall the definition of revocable IBE scheme from [8].

Definition 1 (Revocable IBE). An identity-based encryption with efficient revocation or simply *Revocable IBE* scheme $\mathsf{RIBE} = (\mathcal{S}, \mathcal{SK}, \mathcal{KU}, \mathcal{DK}, \mathcal{E}, \mathcal{D}, \mathcal{R})$ is defined by seven algorithms and has 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 an entity which is of one of the following types – key authority, sender or receiver. Key authority maintains a revocation list RL and state ST. We say an algorithm is stateful if it updates RL or ST.

- The stateful *setup* algorithm $\mathcal{S}$ (run by key authority): takes as input the security parameter 1^λ and the number of users N, and outputs public parameters PP, master secret key MSK, revocation list RL (initially empty) and state ST.
- The stateful *private key generation* algorithm $\mathcal{SK}$ (run by key authority): takes as input public parameters PP, master secret key MSK, identity $\mathsf{id} \in \mathcal{I}$ and state ST, and outputs the private key $\mathsf{SK}_{\mathsf{id}}$ and an updated state ST.
- The *key update generation* algorithm $\mathcal{KU}$ (run by key authority): takes as input the public parameters PP, master secret key MSK, key update time $\mathsf{t} \in \mathcal{T}$, revocation list RL and state ST, and outputs key update KU_{t}.
- The deterministic *decryption key generation* algorithm $\mathcal{DK}$ (run by receiver): takes as input the private key $\mathsf{SK}_{\mathsf{id}}$ and key update KU_{t}, and outputs decryption key $\mathsf{DK}_{\mathsf{id},\mathsf{t}}$, or a special symbol $\perp$ indicating that id was revoked. (We say that an identity id was revoked at time t if revocation algorithm $\mathcal{R}$ was run by key authority on input $(\mathsf{id}, \mathsf{t}, \mathsf{RL}, \mathsf{ST})$ for any RL, ST.)
- The *encryption* algorithm $\mathcal{E}$ (run by sender): takes as input the public parameters PP, identity $\mathsf{id} \in \mathcal{I}$, encryption time $\mathsf{t} \in \mathcal{T}$ and message $m \in \mathcal{M}$, and outputs ciphertext c. For simplicity and without loss of generality, we assume that id, t are efficiently computable from c.
- The *decryption* algorithm $\mathcal{D}$ (run by receiver): takes as input the decryption key $\mathsf{DK}_{\mathsf{id},\mathsf{t}}$ and ciphertext c, and outputs a message $m \in \mathcal{M}$ or a special symbol $\perp$ indicating that the ciphertext is invalid.
- The stateful *revocation* algorithm $\mathcal{R}$ (run by key authority): takes as input the identity $\mathsf{id} \in \mathcal{I}$ to be revoked, revocation time $\mathsf{t} \in \mathcal{T}$, revocation list RL and state ST, and outputs an updated revocation list RL.

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

$$\begin{aligned}
&(\mathsf{SK}_{\mathsf{id}}, \mathsf{ST}) \xleftarrow{\$} \mathcal{SK}(\mathsf{PP}, \mathsf{MSK}, \mathsf{id}, \mathsf{ST});\ \mathsf{KU}_{\mathsf{t}} \xleftarrow{\$} \mathcal{KU}(\mathsf{PP}, \mathsf{MSK}, \mathsf{t}, \mathsf{RL}, \mathsf{ST}) \\
&\mathsf{DK}_{\mathsf{id},\mathsf{t}} \xleftarrow{\$} \mathcal{DK}(\mathsf{SK}_{\mathsf{id}}, \mathsf{KU}_{\mathsf{t}});\ c \xleftarrow{\$} \mathcal{E}(\mathsf{PP}, \mathsf{id}, \mathsf{t}, m) \\
&\text{If } \mathcal{D}(\mathsf{DK}_{\mathsf{id},\mathsf{t}}, c) = m, \text{ then return 1 else return 0.}
\end{aligned}$$

2.3 Security of Revocable IBE

Boldyreva et al. [8] formalized the selective-revocable-ID security that captures the usual notion of selective-ID security and also takes revocation into account. In addition to a private key generation oracle $\mathcal{SK}(\cdot)$ that outputs private keys for identities of its choice, the adversary is allowed to revoke users at will using a dedicated oracle $\mathcal{R}(\cdot,\cdot)$ (taking as input identities id and time t) and can obtain key update information (which is assumed to be public) for any time t via queries to $\mathcal{KU}(\cdot)$. We follow the same definition from Boldyreva et al. [8]. For an adversary $\mathcal{A}$ and number of users N, we define the following experiment:

Experiment $\mathsf{Exp}^{\mathsf{ind-srid-cpa}}_{\mathcal{A},\mathsf{RIBE}}(1^\lambda)$:

$(\mathsf{id}^*, \mathsf{t}^*, state) \xleftarrow{\$} \mathcal{A}(1^\lambda)$
$(\mathsf{PP}, \mathsf{MSK}, \mathsf{RL}, \mathsf{ST}) \xleftarrow{\$} \mathcal{S}(1^\lambda, N)$
$(m_0, m_1, state) \leftarrow \mathcal{A}^{\mathcal{SK}(\cdot),\mathcal{KU}(\cdot),\mathcal{R}(\cdot,\cdot)}(\mathsf{PP}, state)$
$\beta \xleftarrow{\$} \{0,1\}$
$c^* \leftarrow \mathsf{E}(\mathsf{PP}, \mathsf{id}^*, \mathsf{t}^*, m_b)$
$\beta' \leftarrow \mathcal{A}^{\mathcal{SK}(\cdot),\mathcal{KU}(\cdot),\mathcal{R}(\cdot,\cdot)}(\mathsf{PP}, c^*, state)$
If $\beta' = \beta$, then return 1, else return 0.

The following conditions must always hold:

- $m_0, m_1 \in \mathcal{M}$ and $|m_0| = |m_1|$.
- $\mathcal{KU}(\cdot)$ and $\mathcal{R}(\cdot,\cdot)$ 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 $\mathcal{R}(\cdot,\cdot)$ cannot be queried at time t if $\mathcal{KU}(\cdot)$ was queried at time t.
- If $\mathcal{SK}(\cdot)$ was queried on identity id^* then $\mathcal{R}(\cdot,\cdot)$ must be queried on $(\mathsf{id}^*, \mathsf{t}^*)$ for any time $\mathsf{t} \le \mathsf{t}^*$, i.e., identity id^* must be in RL when key update oracle $\mathcal{KU}(\cdot)$ is queried at time t^*.

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

$$\mathsf{Adv}^{\mathsf{ind\text{-}srid\text{-}cpa}}_{\mathcal{A},\mathsf{RIBE}}(\lambda) := \left| \Pr[\beta' = \beta] - \frac{1}{2} \right|.$$

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

3 Background on Codes with Rank Metric

Definition 3. ***(Rank metric over*** $\mathbb{F}^n_{q^m}$***)*** *Let* $\boldsymbol{x} = (x_1, x_2, \cdots, x_n) \in \mathbb{F}^n_{q^m}$ *and consider an arbitrary basis* $(\beta_1, \beta_2, \cdots, \beta_m)$ *of* $\mathbb{F}^m_{q^m}$ *of* $\mathbb{F}_{q^m}$ *viewed as an* m*-dimensional vector space over* F_q*. Then each entry* x_j *in this basis can be written as* $x_j = \sum_{i=1}^m x_{ij}\beta_i$*. The* $m \times n$ *matrix associated with* $\boldsymbol{x}$ *is given by* $\boldsymbol{M}(\boldsymbol{x}) = (x_{ij})_{\substack{1\le i\le m\\ 1\le j\le n}}$*. The rank weight of* $\boldsymbol{x}$*, denoted by* $\|\boldsymbol{x}\|$ *is defined as:*

$$\|\boldsymbol{x}\| = \text{Rank } \boldsymbol{M}(\boldsymbol{x}).$$

The rank distance between elements $\boldsymbol{x}$ *and* $\boldsymbol{y}$*, denoted as* $\mathsf{d}(\boldsymbol{x},\boldsymbol{y})$ *is defined by* $\mathsf{d}(\boldsymbol{x},\boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|$.

Note 1. It can be easily seen that the rank weight is invariant under the choice of basis. One can refer to [26] for more properties of codes with rank metric.

Rank Code. A rank code $\mathcal{C}$ of length n and dimension k is a subspace of dimension k of $\mathbb{F}_{q^m}^n$, embedded with the rank metric. The generator matrix $\boldsymbol{G}$ of $\mathcal{C}$, is of size $k \times n$, consisting of k linearly independent rows.

Minimum Rank distance: Let $\mathcal{C}$ be a rank code over $\mathbb{F}_{q^m}$ then $r \overset{\text{def}}{=\!=} \min_{c_1 \neq c_2 \in \mathcal{C}} \mathsf{d}(c_1, c_2)$ is the minimum rank distance of $\mathcal{C}$.

Dual Code. One can define usual inner product on $\mathbb{F}_{q^m}^n$, to define dual of $\mathcal{C}$. The dual code $\mathcal{C}'$ has dimension $n-k$ and the corresponding generator matrix say $\boldsymbol{H}$ of size $(n-k) \times n$, forms a parity check matrix for $\mathcal{C}$.

Support of $\boldsymbol{x}$. Let $\boldsymbol{x} = (x_1, x_2, \cdots, x_n) \in \mathbb{F}_{q^m}^n$ be a vector of rank weight r. Define the set $E = \langle x_1, x_2, \ldots, x_n \rangle_{\mathbb{F}_q}$, the $\mathbb{F}_q$-linear subspace of $\mathbb{F}_{q^m}$ generated by the linear combinations of $x_1, x_2, \cdots, x_n$ over $\mathbb{F}_q$. The subspace E is called the support of $\boldsymbol{x}$ and is denoted by $\mathsf{Supp}(\boldsymbol{x})$.

3.1 Bounds for Rank Metric Codes.

To present the analogues of Singleton and Gilbert-Varshamov bound for codes with rank metric, we recall the following definitions (given a vector $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$):

- Sphere of radius ℓ centered at $\boldsymbol{x}$: $\mathcal{S}(\boldsymbol{x}, n, m, q, \ell) = \{\boldsymbol{y} \in \mathbb{F}_{q^m}^n \mid \mathsf{d}(\boldsymbol{x}, \boldsymbol{y}) = \ell\}$.
- Ball of radius ℓ centered at $\boldsymbol{x}$: $\mathcal{B}(\boldsymbol{x}, n, m, q, \ell) = \cup_{i=0}^{\ell} \mathcal{S}(\boldsymbol{x}, n, m, q, i)$.

Since the rank metric is invariant under the translation of vectors, the volume of a sphere and ball does not depend on the center. Therefore, we can define $\mathcal{S}(n, m, q, \ell)$ which is equal to number of $m \times n$, q-ary matrices of rank ℓ, where $0 \leq \ell \leq \min(m, n)$. Clearly $\mathcal{S}(n, m, q, 0) = 1$. Moreover, one can show that [26]:

$$S(n, m, q, \ell) = \prod_{j=0}^{\ell-1} \frac{(q^n - q^j)(q^m - q^j)}{q^\ell - q^j}.$$

$$B(n, m, q, \ell) = \sum_{i=0}^{\ell} S(n, m, q, i).$$

Definition 4 (Rank Gilbert-Varshamov bound (RGV)). *For a linear code $[n, k]$ over $\mathbb{F}_{q^m}$ with rank metric, the Rank Gilbert-Varshamov (RGV) bound is defined as the smallest integer ℓ, such that $\mathcal{B}(n, m, q, \ell) \geq q^{m(n-k)}$.*

From decoding point of view, the Gilbert-Varshamov bound for a code $\mathcal{C}$, with parity check matrix $\boldsymbol{H}$, is the smallest weight r such that for any syndrome $\boldsymbol{s}$, there exists on average a codeword $\boldsymbol{x}$ of weight r such that $\boldsymbol{H}\boldsymbol{x}^{\mathsf{T}} = \boldsymbol{s}$. In the case of codes with rank metric, for $m = n$, asymptotically we have [26]:

$$\frac{RGV(n, k, m, q)}{n} \sim 1 - \sqrt{\frac{k}{n}}.$$

Definition 5 (Singleton Bound). *The singleton bound for codes with rank metric of minimum rank r, is given by $r \leq n - k + 1$; when $n > m$ this bound can be rewritten as [26]: $r \leq 1 + \left\lfloor \frac{(n-k)m}{n} \right\rfloor$.*

3.2 Low Rank Parity Check Codes

Definition 6 (Low Rank Parity Check Codes [19]). *A Low Rank Parity Check (LRPC) code of rank d, length n and dimension k over $\mathbb{F}_{q^m}$ is a code defined by an $(n-k) \times n$ parity check matrix $\boldsymbol{H} = (h_{ij})$, such that all its coordinates h_{ij} belong to the same $\mathbb{F}_q$-subspace $\boldsymbol{F}$ of dimension d of $\mathbb{F}_{q^m}$. We denote by $\{\boldsymbol{F}_1, \boldsymbol{F}_2, \dots, \boldsymbol{F}_d\}$ a basis of $\boldsymbol{F}$.*

The decoding error probability for LRPC codes can be made arbitrarily small up to $\dfrac{n-k}{d}$ errors [19].

Definition 7 (Augmented Low Rank Parity Check (LRPC^+) Codes [21]). *Let $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ be a homogeneous matrix of full-rank and of weight d, i.e., all its entries belong to the same $\mathbb{F}_q$-vector subspace of dimension d. Let $\boldsymbol{R} \in \mathbb{F}_{q^m}^{(n-k)\times \ell}$ be a random matrix. Let $\boldsymbol{P} \in GL_{n-k}(\mathbb{F}_{q^m})$ and $\boldsymbol{Q} \in GL_{n+\ell}(\mathbb{F}_q)$ be two invertible matrices. Let $\boldsymbol{H'} = \boldsymbol{P}(\boldsymbol{R}|\boldsymbol{H})\boldsymbol{Q}$ be a parity-check matrix of a code $\mathcal{C}$ of type $[n+\ell, \ell+k]$. By definition, such a code is an LRPC^+ code. If $\ell = 0$, $\mathcal{C}$ is an LRPC code.*

Definition 8 (Simple Codes [18]). *A code $\mathcal{C}$ is said to be (n,k,ℓ)-simple when it has a parity-check matrix $\boldsymbol{H}$ of the form*

$$\boldsymbol{H} = \left(\mathbf{I}_{n-k} \middle| \begin{matrix} \mathbf{0}_\ell \\ \boldsymbol{R} \end{matrix} \right)$$

where $\boldsymbol{I}_{n-k}$ is the $(n-k)\times(n-k)$ identity matrix, $\mathbf{0}_\ell$ is the zero-matrix of size $\ell \times k$ and $\boldsymbol{R}$ is a matrix over $\mathbb{F}_{q^m}$ of size $(n-k-\ell)\times k$. It is called a random simple code if $\boldsymbol{R}$ is chosen uniformly at random among matrices of this size.

Decoding of Simple Code. Let $\mathcal{C}$ be a random (n,k,ℓ)-simple code with $\ell < \dfrac{m+n-\sqrt{(m-n)^2+4km}}{2}$ and w an integer. If $w \leq \ell$, then $\mathcal{C}$ can decode an error of weight w with probability of failure $p_f \sim \dfrac{1}{q^{\ell-w+1}}$ when $q \to \infty$ [18].

3.3 Hard Problems for Rank-based Cryptography

The security of code based cryptosystems generally relies on the hardness of syndrome decoding problem. We define the rank metric version of this problem.

Definition 9 (Rank (Metric) Syndrome Decoding Problem (RSD)). *Let $\boldsymbol{H}$ be a full rank $(n-k)\times n$ matrix over $\mathbb{F}_{q^m}^n$ with $k \leq n$, $\boldsymbol{s} \in \mathbb{F}_{q^m}^{n-k}$ and w be an integer. The problem is to find $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ such that $\mathsf{Rank}(\boldsymbol{x}) = w$ and $\boldsymbol{Hx} = \boldsymbol{s}$. We denote this problem as the $\mathsf{RSD}_{q,m,n,k,w}$ problem.*

The RSD problem has recently been proven hard in [22] on probabilistic reduction. We also discuss the hardness of RSD problem in Appendix A. This problem has an equivalent dual version. Let $\boldsymbol{H}$ be a parity-check matrix of a code $\mathcal{C}$ and $\boldsymbol{G}$ be a generator matrix. Then the RSD problem is equivalent to find $\boldsymbol{m} \in \mathbb{F}_{q^m}^k$ and $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ such that $\boldsymbol{mG} + \boldsymbol{x} = \boldsymbol{y}$ with $\mathsf{Rank}(\boldsymbol{x}) = w$ and $\boldsymbol{y}$ some preimage of $\boldsymbol{s}$ by $\boldsymbol{H}$.

Definition 10 (Decisional Rank Syndrome Decoding Problem (DRSD)). *Let $\boldsymbol{G}$ be a full rank $k \times n$ matrix over $\mathbb{F}_{q^m}$, $\boldsymbol{m} \in \mathbb{F}_{q^m}^k$ and $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ of weight w. Can we distinguish the pair $(\boldsymbol{G}, \boldsymbol{mG}+\boldsymbol{x})$ from $(\boldsymbol{G}, \boldsymbol{y})$ with $\boldsymbol{y} \xleftarrow{\$} \mathbb{F}_{q^m}^n$?*

The hardness of the DRSD problem is proven in [18].

Definition 11 (Rank Support Learning (RSL) Problem [18]). *Let $\boldsymbol{A}$ be a random full-rank matrix of size $(n-k)\times n$ over $\mathbb{F}_{q^m}$ and U be a subspace of $\mathbb{F}_{q^m}$ of dimension w. Let $\mathcal{O}$ be an oracle which gives samples of the form $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{v})$, where $\boldsymbol{v} \overset{\$}{\leftarrow} U^n$. The $\mathsf{RSL}_{q,m,n,k,w}$ problem is to recover U given only access to the oracle. We say that the problem is $(N, \mathsf{t}, \epsilon)$-hard if for every probabilistic algorithm $\mathcal{A}$ running in time t, we have*

$$Pr[\mathcal{A}(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{V}) = U] \leq \epsilon, \quad \boldsymbol{V} \overset{\$}{\leftarrow} U^{n\times N}$$

When we are allowed to make exactly N calls to the oracle, we denote this problem by $\mathsf{RSL}_{q,m,n,k,w,N}$ problem. The pair $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{V})$ is referred to as an instance of the $\mathsf{RSL}_{q,m,n,k,w,N}$ problem. The corresponding decisional problem, namely DRSL, *is to distinguish $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{V})$ from $(\boldsymbol{A}, \boldsymbol{Y})$ where $\boldsymbol{Y} \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^{(n-k)\times N}$.*

The $\mathsf{RSL}_{q,m,n,k,w,N}$ problem is proven as hard as $\mathsf{RSD}_{q,m,n,k,w}$ problem in [18].

Definition 12 (LRPC$^+$ Problem [21]). *Given an augmented LRPC code, distinguish it from a random code with the same parameters.*

The hardness of this problem is studied in [21, 24].

3.4 RankSign Algorithm

We will use RankSign algorithm [21] to construct trapdoors which will be used to generate the secret keys corresponding to identity and time in our RIBE. The security of RankSign algorithm relies on the hardness of the RSD problem.

In short, the RankSign algorithm uses an efficient decoding algorithm which takes input a random word of the syndrome space (obtained from the hash of the file we want to sign) and outputs a word of small weight with the given syndrome. This is an instance of the RSD problem. However, the parity-check matrix $\boldsymbol{H}$ has a trapdoor which makes the RSD problem easy. The public key is a description of the code which hides its structure, while the secret key reveals the structure of the code, which allows the signer to solve the RSD problem. The RankSign algorithm does not compute a codeword of weight below the Gilbert-Varshamov bound, but instead a codeword of rank weight between the Gilbert-Varshamov and the Singleton bound. The idea is to use a family of augmented Low Rank Parity Check Codes, and an adapted decoding algorithm (called the General Errors/Erasures Decoding algorithm) to produce such a codeword from any syndrome. The decoding algorithm is probabilistic, and the parameters of the code have to be chosen precisely in order to have a probability of success very close to 1. One can refer to [21] for more details.

3.4.1 Sampling Secrets using Trapdoors from RankSign Algorithm.

Similar to the approach of Gaborit et al. [18], we also adapt the RankSign algorithm to construct a trapdoor, by which one can sample the secrets corresponding to a public identity. Associated to a matrix $\boldsymbol{A} \in \mathbb{F}_{q^m}^{(n-k)\times n}$, we define the function $f_{\boldsymbol{A}}$ as follows:

$$\begin{aligned} f_{\boldsymbol{A}} : \mathbb{F}_{q^m}^{n-k} \times \mathbb{F}_{q^m}^{n} &\to \mathbb{F}_{q^m}^{n} \\ (\boldsymbol{s}, \boldsymbol{e}) &\to \boldsymbol{s}\boldsymbol{A} + \boldsymbol{e} \end{aligned}$$

The matrix $\boldsymbol{A}$ is generated with a trapdoor $\boldsymbol{T}$ such that $f_{\boldsymbol{A}}$ is a trapdoor function: from a random $\boldsymbol{p} \in \mathbb{F}_{q^m}^n$, with the trapdoor $\boldsymbol{T}$, one can sample $(\boldsymbol{s}, \boldsymbol{e}) = f_{\boldsymbol{A}}^{-1}(\boldsymbol{p})$ such that $\boldsymbol{e}$ is indistinguishable from a random element in W_r, the set of all words of rank r and of length n.

We extend the same approach to generate secrets corresponding to two attributes identity and time, but these two attributes are bound together in the sense that two secrets will make a complete decryption key in RIBE setting. In our case, from a random $\boldsymbol{p} \in \mathbb{F}_{q^m}^n$ and $\boldsymbol{u} \in \mathbb{F}_{q^m}^n$, with the trapdoor $\boldsymbol{T}$, one can sample $(\boldsymbol{s}, \boldsymbol{e}) = f_{\boldsymbol{A}}^{-1}(\boldsymbol{p}+\boldsymbol{u})$ such that $\boldsymbol{e}$ is indistinguishable from a random element in W_r.

4 Revocable IBE from Codes with Rank Metric

4.1 The Binary Tree Data Structure

Our construction makes use of binary tree data structure as described in [8]. We denote the binary tree by BT and its root node by root. If v is a leaf node then $\mathsf{Path}(v)$ stands for the set of nodes on the path from v to the root (inclusive of both v and root). Each user is assigned to a leaf node v. Upon registration, the key authority provides the user with a set of distinct private keys for each node in $\mathsf{Path}(v)$. Whenever θ is a non-leaf node, θ_ℓ and θ_r denote the left and right children 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.

The KUNodes algorithm run by the key authority, at each time t, determines the minimal set $Y \subset \mathsf{BT}$ of nodes that contains an ancestor of all leaves corresponding to non-revoked users. This minimal set precisely contains nodes for which key updates have to be published in such a way that only non-revoked users will be able to generate the appropriate decryption key for the matching time. It first marks all ancestors of users that were revoked by time t as revoked nodes. Then, it inserts in Y the non-revoked children of revoked nodes. It can be formally specified as follows:

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

The key authority then publishes a key update for all the nodes of Y. A user assigned to leaf v is then able to form an effective decryption key for time t if the set Y contains a node in $\mathsf{Path}(v)$. A graphical description is given in Appendix D.

4.2 Our RIBE Construction

Our Revocable IBE (RIBE) scheme consists of following seven PPT algorithms:

1. **Setup** $\mathcal{S}(1^\lambda, N)$: on input the security parameter λ and a maximal number N of users, set the parameters (n, m, k, d, ℓ) as specified in subsection 4.4.

- Let $H_1 : \{0,1\}^* \rightarrow \mathbb{F}_{q^m}^{n+\ell}$ and $H_2 : \{0,1\}^* \rightarrow \mathbb{F}_{q^m}^{n+\ell}$ be two cryptographic hash functions.
- Let $\boldsymbol{H}$ is a parity-check matrix of an LRPC code of weight d over $\mathbb{F}_{q^m}^{(n-k)\times n}$. Let $\boldsymbol{R} \in \mathbb{F}_{q^m}^{(n-k)\times \ell}$ be a random matrix. Let $\boldsymbol{P} \in GL_{n-k}(\mathbb{F}_{q^m})$ and $\boldsymbol{Q} \in GL_{n+\ell}(\mathbb{F}_q)$ be two invertible matrices. Let $\boldsymbol{A}$ be a full rank $(k+\ell)\times(n+\ell)$ matrix over $\mathbb{F}_{q^m}$ such that $\boldsymbol{H'}\boldsymbol{A}^\mathsf{T} = \boldsymbol{0}$ with $\boldsymbol{H'} = \boldsymbol{P}(\boldsymbol{R}|\boldsymbol{H})\boldsymbol{Q}$ and the trapdoor $\boldsymbol{T}$ is $(\boldsymbol{P}, \boldsymbol{Q})$.
- Define $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k'\times n'}$ a generator matrix of a public *simple code* $\mathcal{C}'$ which can decode errors of weight up to $2wr$, where w is the weight of a homogeneous matrix used in encryption algorithm, and r is the rank weight of error vector $\boldsymbol{e}$ of length $n+\ell$.
- Let RL be an empty set and BT be a binary tree with at least N leaf nodes, set $\mathsf{ST} := \mathsf{BT}$.
- Select a uniformly random vector $\boldsymbol{u} \xleftarrow{\$} \mathbb{F}_{q^m}^{n+\ell}$.
- Output RL, ST, the public parameters, and the master key MSK,

$$\mathsf{PP} := (\boldsymbol{A}, \boldsymbol{G}, \boldsymbol{u}), \qquad \mathsf{MSK} := \boldsymbol{T} = (\boldsymbol{P}, \boldsymbol{Q}).$$

2. **Private key generation** $\mathcal{SK}(\mathsf{PP}, \mathsf{MSK}, \mathsf{id}, \mathsf{ST})$: on input the public parameters PP, the master secret key MSK, an identity id and the state ST, it picks an unassigned leaf node v from BT and stores id in that node. It then performs the following steps:
   - $\forall \theta \in \mathsf{Path}(v)$, if $\boldsymbol{u}_{\theta,1}$, $\boldsymbol{u}_{\theta,2}$ are undefined, then pick $\boldsymbol{u}_{\theta,1} \xleftarrow{\$} \mathbb{F}_{q^m}^{n+\ell}$, set $\boldsymbol{u}_{\theta,2} := \boldsymbol{u} - \boldsymbol{u}_{\theta,1}$, and store them in node θ.
   - Compute $\boldsymbol{p}_1 = H_1(\mathsf{id})$ and syndrome $\boldsymbol{x}_{\theta,1} = \boldsymbol{H'}\boldsymbol{p}_1^\mathsf{T} + \boldsymbol{H'}\boldsymbol{u}_{\theta,1}^\mathsf{T}$.
   - Sample $\boldsymbol{e}_{\theta,1} \in \mathbb{F}_{q^m}^{n+\ell}$ of rank weight r, as $\boldsymbol{H'}\boldsymbol{e}_{\theta,1}^\mathsf{T} = \boldsymbol{x}_{\theta,1}$ using RankSign algorithm with trapdoor $\boldsymbol{T}$.
   - Compute $\boldsymbol{s}_{\theta,1} \in \mathbb{F}_{q^m}^{k+\ell}$ as $\boldsymbol{p}_1 + \boldsymbol{u}_{\theta,1} = \boldsymbol{s}_{\theta,1}\boldsymbol{A} + \boldsymbol{e}_{\theta,1}$.
   - Output $\mathsf{SK}_\mathsf{id} := \{(\theta, \boldsymbol{s}_{\theta,1})\}_{\theta\in\mathsf{Path}(v)}$, ST.

3. **Key update generation** $\mathcal{KU}(\mathsf{PP}, \mathsf{MSK}, \mathsf{t}, \mathsf{RL}, \mathsf{ST})$: on input the public parameters PP, the master secret key MSK, a time $\mathsf{t} \in \mathbb{F}_{q^m}^n$, the revocation list RL, and the state ST, it performs the following steps:
   - For all $\theta \in \mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL}, \mathsf{t})$, if $\boldsymbol{u}_{\theta,1}$, $\boldsymbol{u}_{\theta,2}$ are undefined, then pick $\boldsymbol{u}_{\theta,2} \xleftarrow{\$} \mathbb{F}_{q^m}^{n+\ell}$, set $\boldsymbol{u}_{\theta,1} := \boldsymbol{u} - \boldsymbol{u}_{\theta,2}$, and store them in node θ.
   - Compute $\boldsymbol{p}_2 = H_2(\mathsf{t})$ and syndrome $\boldsymbol{x}_{\theta,2} = \boldsymbol{H'}\boldsymbol{p}_2^\mathsf{T} + \boldsymbol{H'}\boldsymbol{u}_{\theta,2}^\mathsf{T}$.
   - Sample $\boldsymbol{e}_{\theta,2} \in \mathbb{F}_{q^m}^{n+\ell}$ of rank weight r, as $\boldsymbol{H'}\boldsymbol{e}_{\theta,2}^T = \boldsymbol{x}_{\theta,2}$ using RankSign algorithm with trapdoor $\boldsymbol{T}$.
   - Compute $\boldsymbol{s}_{\theta,2} \in \mathbb{F}_{q^m}^{k+\ell}$ as $\boldsymbol{p}_2 + \boldsymbol{u}_{\theta,2} = \boldsymbol{s}_{\theta,2}\boldsymbol{A} + \boldsymbol{e}_{\theta,2}$.
   - Output $\mathsf{KU}_\mathsf{t} := \{(\theta, \boldsymbol{s}_{\theta,2})\}_{\theta\in\mathsf{KUNodes}(\mathsf{BT},\mathsf{RL},\mathsf{t})}$, ST.

4. **Decryption key generation** $\mathcal{DK}(\mathsf{SK}_\mathsf{id}, \mathsf{KU}_\mathsf{t})$: on input a private secret key $\mathsf{SK}_\mathsf{id} := \{(i, \boldsymbol{s}_{i,1})\}_{i\in I}$, and key update $\mathsf{KU}_\mathsf{t} := \{(j, \boldsymbol{s}_{j,2})\}_{j\in J}$ for some set of nodes I, J; it performs the following steps:
   - $\forall (i, \boldsymbol{s}_{i,1}) \in \mathsf{SK}_\mathsf{id}$, $(j, \boldsymbol{s}_{j,2}) \in \mathsf{KU}_\mathsf{t}$, if $\exists (i,j)$ such that $i = j$ then $\mathsf{DK}_{\mathsf{id},\mathsf{t}} \leftarrow (\boldsymbol{s}_{i,1}, \boldsymbol{s}_{j,2})$; else (if SK_id and KU_t do not have any node in common) $\mathsf{DK}_{\mathsf{id},\mathsf{t}} \leftarrow \perp$.
   - Output $\mathsf{DK}_{\mathsf{id},\mathsf{t}}$.

We can drop the subscripts i, j since they are equal, i.e., $\mathsf{DK}_{\mathsf{id},\mathsf{t}} := (\boldsymbol{s}_1, \boldsymbol{s}_2)$. The algorithm finds components of SK_id and KU_t (since they are in the same node) such that $(\boldsymbol{p}_1 + \boldsymbol{p}_2 + \boldsymbol{u}) = (\boldsymbol{s}_1 + \boldsymbol{s}_2)\boldsymbol{A} + (\boldsymbol{e}_1 + \boldsymbol{e}_2)$.

5. **Encryption** $\mathcal{E}(\mathsf{PP}, \mathsf{id}, \mathsf{t}, \boldsymbol{m})$: on input the public parameters PP, an identity id, a time t, and a message $\boldsymbol{m} \in \mathbb{F}_{q^m}^{k'}$, it performs the following steps:
   - Compute $\boldsymbol{p}_1 = H_1(\mathsf{id})$ and $\boldsymbol{p}_2 = H_2(\mathsf{t})$.
   - Generate a random homogeneous matrix $\boldsymbol{V} \in \mathbb{F}_{q^m}^{(n+\ell)\times n'}$ of weight w.
   - Compute the pair $(\boldsymbol{C}, \boldsymbol{x})$ of $\boldsymbol{m}$ as

$$\left(\frac{\boldsymbol{A}}{\boldsymbol{p}_1 + \boldsymbol{p}_2 + \boldsymbol{u}}\right) \boldsymbol{V} + \left(\frac{\boldsymbol{0}}{\boldsymbol{mG}}\right) = \left(\frac{\boldsymbol{C}}{\boldsymbol{x}}\right)$$

   - Output the ciphertext $\mathsf{CT} = (\mathsf{id}, \mathsf{t}, \boldsymbol{C}, \boldsymbol{x})$.

6. **Decryption** $\mathcal{D}(\mathsf{PP}, \mathsf{DK}_{\mathsf{id},\mathsf{t}}, \mathsf{CT})$: on input the public parameters PP, a decryption key $\mathsf{DK}_{\mathsf{id},\mathsf{t}} := (\boldsymbol{s}_1, \boldsymbol{s}_2)$, and a ciphertext $\mathsf{CT} = (\mathsf{id}, \mathsf{t}, \boldsymbol{C}, \boldsymbol{x})$, it performs the following steps:
   - Compute $\boldsymbol{p}_1 = H_1(\mathsf{id})$ and $\boldsymbol{p}_2 = H_2(\mathsf{t})$.
   - Use the decryption key $(\boldsymbol{s}_1, \boldsymbol{s}_2)$ with $\boldsymbol{s} = \boldsymbol{s}_1 + \boldsymbol{s}_2$ to compute

$$\begin{aligned}\left(\boldsymbol{s} \mid -1\right) \left(\frac{\boldsymbol{C}}{\boldsymbol{x}}\right) &= \boldsymbol{sC} - \boldsymbol{x} \\ &= -(\boldsymbol{e}_1 + \boldsymbol{e}_2)\boldsymbol{V} - \boldsymbol{mG}.\end{aligned}$$

   - Since $\boldsymbol{V}$ is a homogeneous matrix of weight w, and $\boldsymbol{e}_1$, $\boldsymbol{e}_2$ are the error vectors of rank r, we have $\|(\boldsymbol{e}_1 + \boldsymbol{e}_2)\boldsymbol{V}\| \leq 2wr$. Therefore, by using the decoding algorithm of $\mathcal{C}'$, we can recover $\boldsymbol{m}$.

7. **Revocation** $\mathcal{R}(\mathsf{id}, \mathsf{t}, \mathsf{RL}, \mathsf{ST})$: on input an identity id, a time t, the revocation list RL, and the state ST; let v be the leaf node associated with id. To revoke the identity id at time t, add (v, t) to RL, and return RL.

4.3 Security Result

Theorem 1. *Suppose the hash functions H_1 and H_2 are random oracles, and the DRSD, DRSL and LRPC^+ assumptions hold. Then RIBE scheme is IND-sRID-CPA secure in the random oracle model. More precisely, if there exists an adversary $\mathcal{A}$ against the IND-sRID-CPA security, who makes at most q_{H_1} and q_{H_2} distinct queries to the H_1 and H_2 random oracles, then the advantage of adversary $\mathcal{A}$ is given by the following expression*

$$\epsilon_{\mathsf{ribe}} \leq \left(q_{H_1} + q_{H_2}\right) \cdot \left(\frac{2}{q} + \epsilon_{\mathsf{drsd}}\right) + \epsilon_{\mathsf{lrpc}^+} + \epsilon_{\mathsf{drsl}},$$

where ϵ_{ribe}, ϵ_{drsd}, ϵ_{drsl} and $\epsilon_{\mathsf{lrpc}^+}$ are respectively the bound on the advantage of the attacks against the RIBE system, the DRSD, DRSL and LRPC$^+$ problems.

Proof. The complete proof is given in Appendix C, but we provide its intuition here. We show that a probabilistic polynomial time adversary $\mathcal{A}$ cannot distinguish between the games which proves that the adversary has a negligible advantage in winning the original IND-sRID-CPA game. In moving from game G_0 to G_1, we randomly generate the decryption keys without the knowledge of the trapdoor, and the following relationship still holds between the decryption key and the public key: $\boldsymbol{p}_1+\boldsymbol{p}_2+\boldsymbol{u} = (\boldsymbol{s}_1+\boldsymbol{s}_2)\boldsymbol{A}+(\boldsymbol{e}_1+\boldsymbol{e}_2)$. To ensure that no information is leaked about the decryption keys during the game, we consider two kinds of adversaries:

- Type 1 Adversary: It chooses to be challenged on the targeted identity id^* but is revoked before or on time t^*.
- Type 2 Adversary : It does not challenge the target identity id^* at any time.

The main difficulty we face in simulating the private key generation and key update oracles with identity $\mathsf{id} = \mathsf{id}^*$ and time $\mathsf{t} = \mathsf{t}^*$ respectively, for Type-1 adversary. We need to simulate the queries in such a way that revoked user id^* does not get key update information at time t^*, since it is revoked at a time t^*. In brief, on private key query id^*, for nodes $\theta \in \mathsf{Path}(v^*)$, we choose $\boldsymbol{s}_{\theta,1}$, $\boldsymbol{e}_{\theta,1}$, $\boldsymbol{p}_1$ randomly and define the shares $\boldsymbol{u}_{\theta,1}$ and $\boldsymbol{u}_{\theta,2}$ such that $\boldsymbol{u} = \boldsymbol{u}_{\theta,1} + \boldsymbol{u}_{\theta,2}$. On key update query t^*, for nodes $\theta \notin \mathsf{Path}(v^*)$, we choose $\boldsymbol{s}_{\theta,2}$, $\boldsymbol{e}_{\theta,2}$, $\boldsymbol{p}_2$ randomly and define the shares $\boldsymbol{u}_{\theta,2}$ and $\boldsymbol{u}_{\theta,1}$ such that $\boldsymbol{u} = \boldsymbol{u}_{\theta,1} + \boldsymbol{u}_{\theta,2}$. As a consequence, Type-1 adversary does not get the key update information at time t^* for the identity id^*. On the contrary, it is easy to simulate the queries for Type-2 adversary since it does not query id^* at any time. Finally, we have a new tuple $(\boldsymbol{p}_1, \boldsymbol{u}_1, \boldsymbol{s}_1, \boldsymbol{e}_1)$ which is random, therefore the advantage of adversary to distinguish a RSD pair $(\boldsymbol{A}, \boldsymbol{p} + \boldsymbol{u} = \boldsymbol{s}\boldsymbol{A} + \boldsymbol{e})$ with a random one, is bounded by ϵ_{drsd} plus some decoding error probability.

In moving from game G_1 to G_2, we define matrix $\boldsymbol{A}$ to be a random matrix. Note that $\boldsymbol{A}$ is used to generate the codewords. Thus, the advantage of adversary in distinguishing an augmented LRPC code from a random code, is bounded by $\epsilon_{\mathsf{lrpc}^+}$.

In moving from game G_2 to G_3, we randomly choose challenged ciphertext, then the problem is reducible to DRSL problem. Thus, the distinguishing advantage of adversary is bounded by ϵ_{drsl}.

At the end, in game G_4 we bound the advantage of adversary to guess the bit β hidden in perfectly random ciphertext, which is 1/2. This justifies the bound on the advantage of adversary to break RIBE.

4.4 General Parameters

Here, we discuss the size of parameters for our RIBE scheme against the best known attacks. The parameters used in our scheme are as follows. Let q is the size of the base field $\mathbb{F}_q$ and m is the degree of the extension field $\mathbb{F}_{q^m}$; n is the length of the hidden LRPC code; ℓ is the number of random columns added to the LRPC to hide it; k is the dimension of the LRPC code and r is the rank weight of the signature $\boldsymbol{e}$ computed by the $\mathsf{RankSign}$ algorithm; (n', k', ℓ') are the parameters of a simple code that can correct up to $2wr$ errors. To make the density of decodable syndrome close to 1, these parameters must satisfy the following three conditions [21]:

$$n = d(n-k); \quad (r-\ell)(m-r) + (n-k)(rd-m) = 0; \quad r = \ell + \frac{n-k}{d}.$$

Observe that the three conditions mentioned above are homogeneous if d is constant. Thus, we can make another set of parameters from one set by multiplying all the parameters (except for d) by a constant. d is the weight of the LRPC$^+$ code used for public parameters, which should not be too small to ensure the security of public parameters.

A Practical Set of Parameters. From the security result in Theorem 1, we have $\epsilon_{\mathsf{ribe}} \leq \left(q_{H_1} + q_{H_2}\right) \cdot \left(\frac{2}{q} + \epsilon_{\mathsf{drsd}}\right) + \epsilon_{\mathsf{lrpc}^+} + \epsilon_{\mathsf{drsl}}$. We need $\epsilon_{\mathsf{ribe}} < 2^{-\lambda}$, where λ is the security parameter. Since the first term only depends upon q and the number of queries q_{H_1} and q_{H_2}, thus we need $q > (q_{H_1} + q_{H_2})2^{\lambda+1}$ to hold. We stress that the size of data and computation time are linear in the logarithmic of q. Moreover, since all the combinatorial attacks are polynomial in q, thus they are inefficient to break RIBE. Furthermore, the success of algebraic attacks depends upon the hardness of LRPC^+ and DRSD problems.

The size of parameters for our RIBE are similar to IBE of Gaborit et al. [18] except the case that we have to choose the parameters of the simple code in such a way that it can decode up to $2wr$ errors and the decoding error with failure probability $\approx \frac{1}{q^{\ell'-2wr+1}}$ is small. As an example, we take the standard values $\lambda = 128$ for the security parameter and $q_{H_1} = q_{H_2} = 2^{60}$, and $q = 2^{192}$ will suffice the standard security requirement.

Scheme	n	$n-k$	m	q	d	ℓ	r	d_{GV}	d_{sign}	Public Key Size (Bytes)	n'	k'	ℓ'	w
RIBE	100	20	96	2^{192}	5	12	16	11	20	4,497,408 of $(\boldsymbol{A}, \boldsymbol{u})$	96	9	66	2

With these parameters one can achieve decoding failure probability $p_f \approx 2^{-576}$, which is negligible.

5 Conclusion and Open Problems

This paper introduced a revocable identity-based encryption scheme, called RIBE from codes with rank metric, and proved its selective-ID security in the random oracle model, under the hardness of DRSD, DRSL, and LRPC$^+$ problems. As a future work, it might be possible to construct an adaptive-ID secure RIBE scheme. Another open problem is to construct an adaptive secure IBE and RIBE schemes from rank metric codes in the standard model.

References

1. S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, pages 553–572, 2010.
2. W. Aiello, S. Lodha, and R. Ostrovsky. Fast digital identity revocation (extended abstract). In *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, pages 137–152, 1998.
3. M. Baldi, M. Bodrato, and F. Chiaraluce. *A New Analysis of the McEliece Cryptosystem Based on QC-LDPC Codes*, pages 246–262. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.
4. M. Baldi, F. Chiaraluce, and R. Garello. On the Usage of Quasi-Cyclic Low-Density Parity-Check Codes in the McEliece Cryptosystem. In *2006 First International Conference on Communications and Electronics*, pages 305–310, Oct 2006.
5. M. Baldi, F. Chiaraluce, R. Garello, and F. Mininni. Quasi-Cyclic Low-Density Parity-Check Codes in the McEliece Cryptosystem. In *2007 IEEE International Conference on Communications*, pages 951–956, June 2007.
6. T. P. Berger, P.-L. Cayrel, P. Gaborit, and A. Otmani. *Reducing Key Length of the McEliece Cryptosystem*, pages 77–97. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
7. E. Berlekamp, R. McEliece, and H. van Tilborg. On the inherent intractability of certain coding problems (corresp.). *IEEE Transactions on Information Theory*, 24(3):384–386, May 1978.
8. A. Boldyreva, V. Goyal, and V. Kumar. Identity-based encryption with efficient revocation. In *Proceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, Alexandria, Virginia, USA, October 27-31, 2008*, pages 417–426, 2008.
9. D. Boneh and M. K. Franklin. Identity-Based Encryption from the Weil Pairing. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, pages 213–229, 2001.
10. J. Chen, H. W. Lim, S. Ling, H. Wang, and K. Nguyen. Revocable identity-based encryption from lattices. In *Information Security and Privacy - 17th Australasian Conference, ACISP 2012, Wollongong, NSW, Australia, July 9-11, 2012. Proceedings*, pages 390–403, 2012.

11. L. Chen, S. Jordan, Y. K. Liu, D. Moody, R. Peralta, R. Perlner, and D. Smith-Tone. Report on post-quantum cryptography. National Institute of Standards and Technology Internal Report 8105, 2016.
12. F. L. dit Vehel and L. Perret. Algebraic decoding of codes in rank metric. In *Proceedings of YACC06*, June 2006.
13. J.-C. Faugère, M. S. El Din, and P.-J. Spaenlehauer. Computing Loci of Rank Defects of Linear Matrices Using GrÖBner Bases and Applications to Cryptology. In *Proceedings of the 2010 International Symposium on Symbolic and Algebraic Computation*, ISSAC '10, pages 257–264, New York, NY, USA, 2010. ACM.
14. J.-C. Faugère, F. Levy-dit Vehel, and L. Perret. *Cryptanalysis of MinRank*, pages 280–296. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.
15. J.-C. Faugère, A. Otmani, L. Perret, and J.-P. Tillich. *Algebraic Cryptanalysis of McEliece Variants with Compact Keys*, pages 279–298. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
16. E. M. Gabidulin. Theory of codes with maximum rank distance. *Probl. Peredachi Inf., (21)*, pages 3–16, 1985.
17. P. Gaborit. Shorter keys for code based cryptography. *In Internatinal Workshop on Coding and Cryptography-WCC'2205*, pages 81–91, 2004.
18. P. Gaborit, A. Hauteville, D. H. Phan, and J. Tillich. Identity-based encryption from codes with rank metric. *IACR Cryptology ePrint Archive*, 2017:514, 2017.
19. P. Gaborit, G. Murat, O. Ruatta, and G. Zémor. Low rank parity check codes and their application to cryptography. *In Proceedings of the Workshop on Coding and Cryptography WCC'2013*, 2013.
20. P. Gaborit, O. Ruatta, and J. Schrek. On the complexity of the rank syndrome decoding problem. *IEEE Transactions on Information Theory*, 62(2):1006–1019, Feb 2016.
21. P. Gaborit, O. Ruatta, J. Schrek, and G. Zémor. Ranksign: An efficient signature algorithm based on the rank metric. In *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014, Waterloo, ON, Canada, October 1-3, 2014. Proceedings*, pages 88–107, 2014.
22. P. Gaborit and G. Zmor. On the hardness of the decoding and the minimum distance problems for rank codes. *IEEE Transactions on Information Theory*, 62(12):7245–7252, Dec 2016.
23. R. G. Gallager. *Low-Density Parity -Check Codes.* PhD thesis, M.I.T. Press, 1963.
24. A. Hauteville and J. P. Tillich. New algorithms for decoding in the rank metric and an attack on the lrpc cryptosystem. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 2747–2751, June 2015.
25. B. Libert and D. Vergnaud. Adaptive-id secure revocable identity-based encryption. In *Topics in Cryptology - CT-RSA 2009, The Cryptographers' Track at the RSA Conference 2009, San Francisco, CA, USA, April 20-24, 2009. Proceedings*, pages 1–15, 2009.
26. P. Loidreau. Asymptotic behaviour of codes in rank metric over finite fields. *Des. Codes Cryptography*, 71(1):105–118, 2014.
27. R. J. McEliece. A Public-Key Cryptosystem Based On Algebraic Coding Theory. *Deep Space Network Progress Report*, 44:114–116, Jan. 1978.
28. R. Misoczki and P. S. L. M. Barreto. *Compact McEliece Keys from Goppa Codes*, pages 376–392. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
29. R. Misoczki, J. P. Tillich, N. Sendrier, and P. S. L. M. Barreto. MDPC-McEliece: New McEliece variants from Moderate Density Parity-Check codes. In *2013 IEEE International Symposium on Information Theory*, pages 2069–2073, July 2013.
30. C. Monico, J. Rosenthal, and A. Shokrollahi. Using low density parity check codes in the McEliece cryptosystem. In *2000 IEEE International Symposium on Information Theory*, pages 215–, 2000.
31. M. Naor and K. Nissim. Certificate revocation and certificate update. *IEEE Journal on Selected Areas in Communications*, 18(4):561–570, 2000.
32. A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology, Proceedings of CRYPTO'84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*, pages 47–53, 1984.
33. P. W. Shor. Polynominal time algorithms for discrete logarithms and factoring on a quantum computer. In *Algorithmic Number Theory, First International Symposium, ANTS-I, Ithaca, NY, USA, May 6-9, 1994, Proceedings*, page 289, 1994.

34. J. Wang and J. Bi. Lattice-based identity-based broadcast encryption scheme. *IACR Cryptology ePrint Archive*, 2010:288, 2010.

A Hardness of Rank Syndrome Decoding Problem

The RSD problem has recently been proven hard in [22] on probabilistic reduction. From a practical point of view, the complexity of attacks grows very fast because of the existing structure in codes. In case of Hamming metric, a key step in the attacks is to find out the number of codewords of length n and weight w, which equals to Newton binomial coefficient $\binom{n}{w}$ (exponential in nature); whereas in case of rank metric, the number of such codewords corresponds to number of subspaces of dimension w in $\mathbb{F}_{q^m}$, which is equal to Gaussian binomial coefficient of size approximately equal to $q^{w(m-w)}$, whose value is also exponential but with a quadratic term in exponent. Next, we discuss the complexity of two types of possible generic attacks.

Combinatorial Attacks: These attacks adapt the basic information set decoding approach in rank metric context, whose complexity increases very fast with large values of q (as seen above). The recent improvements based on birthday paradox does not fit in case of rank metric because of the different notion of support. In practice, when $m \leq n$, the best combinatorial attacks have complexity $\mathcal{O}((n-k)^3 m^3 q^{(r-1)\lfloor \frac{(k+1)m}{n} \rfloor})$ [20].

Algebraic Attacks: The nature of the rank metric suites the algebraic attacks and solving equations by Gröbner basis. These attacks are largely independent of q and in some cases also independent of m. There exist different settings of algebraic equation, that can be solved with Gröbner basis [12–14, 20]. Algebraic attacks solve the system of equations over the base field $\mathbb{F}_q$, where the number of unknowns is quadratic in the length of the code. As the complexity of Gröbner basis technique is exponential in the number of unknowns, it turns out that these attacks have exponential complexity in the length of the code.

B Distinguishing Lemma

We introduce a slightly modified lemma similar to the result of [18], to prove the security of RIBE.

Lemma 1. *Let $\boldsymbol{H'}$ be an augmented parity-check matrix and $\boldsymbol{A}$ be a generator matrix of the associated code. The two following distributions are computationally indistinguishable:*

- *Suppose $\mathcal{D}_0$ be the distribution $(\boldsymbol{p}, \boldsymbol{u}, \boldsymbol{s}, \boldsymbol{e})$ where $\boldsymbol{p} \xleftarrow{\$} \mathbb{F}_{q^m}^{k+\ell}$, $\boldsymbol{u} \xleftarrow{\$} \mathbb{F}_{q^m}^{n+\ell}$, $\boldsymbol{e} \in W_r$ is sampled from RankSign algorithm such that $\boldsymbol{H'}\boldsymbol{e}^\intercal = \boldsymbol{H'}\boldsymbol{p}^\intercal + \boldsymbol{H'}\boldsymbol{u}^T$ and $\boldsymbol{s}$ is the solution of the linear system $\boldsymbol{x}\boldsymbol{A} = \boldsymbol{p} + \boldsymbol{u} - \boldsymbol{e}$ of unknown $\boldsymbol{x}$.*
- *Suppose $\mathcal{D}_1$ be the distribution $(\boldsymbol{p}', \boldsymbol{u}', \boldsymbol{s}', \boldsymbol{e}')$ with $\boldsymbol{u}' \xleftarrow{\$} \mathbb{F}_{q^m}^{n+\ell}$, $\boldsymbol{s}' \xleftarrow{\$} \mathbb{F}_{q^m}^{k+\ell}$, $\boldsymbol{e'} \xleftarrow{\$} W_r$ (where W_r be a set of codewords of rank weight r), and $\boldsymbol{p}' + \boldsymbol{u}' = \boldsymbol{s}'\boldsymbol{A} + \boldsymbol{e}'$.*

The maximum advantage ϵ of adversary to distinguish $\mathcal{D}_0$ from $\mathcal{D}_1$ is bounded by : $\epsilon \leq \frac{2}{q} + \epsilon_{\mathsf{drsd}}$, where q is the size of base field $\mathbb{F}_q$ and ϵ_{drsd} is the bound on the successful probability of the attacks against the DRSD problem.

Proof. Let $\mathcal{D}_2$ be the distribution $(\boldsymbol{s}, \boldsymbol{e})$ where $\boldsymbol{s} \xleftarrow{\$} \mathbb{F}_{q^m}^{k+\ell}$ and $\boldsymbol{e}$ is a signature of $\boldsymbol{s}$ by RankSign with the public key $\boldsymbol{H'}$ (i.e., $\|\boldsymbol{e}\| \leq r$ and $\boldsymbol{H'}\boldsymbol{e}^\intercal = \boldsymbol{s}$).

Let $\mathcal{D}_3$ be the distribution $(\boldsymbol{s}' = \boldsymbol{H}'\boldsymbol{e}'^{\top}, \boldsymbol{e}'^{\top})$ with $\boldsymbol{e}' \overset{\$}{\leftarrow} W_r$. Now, as proved in Theorem 2 of [21], if one sample $(\boldsymbol{H}'\boldsymbol{e}'^{\top}, \boldsymbol{e}'^{\top})$ according to distribution $\mathcal{D}_3$, then the probability that $(\boldsymbol{H}'\boldsymbol{e}'^{\top}, \boldsymbol{e}'^{\top})$ is not decodable, is less than $\frac{2}{q}$. Therefore, the advantage of an adversary to distinguish $\mathcal{D}_2$ from $\mathcal{D}_3$ is bounded by $\frac{2}{q}$.

To prove the lemma, now let us examine the distribution $\mathcal{D}_0$. Since $\boldsymbol{H}'$ is a linear map and $\boldsymbol{p} \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^{n+\ell}$, $\boldsymbol{u} \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^{n+\ell}$, and $\boldsymbol{s} = \boldsymbol{H}'\boldsymbol{p}^{\top} + \boldsymbol{H}'\boldsymbol{u}^{\top}$ is uniformly distributed among $\mathbb{F}_{q^m}^{k+\ell}$. This implies $(\boldsymbol{s}, \boldsymbol{e}) \leftarrow \mathcal{D}_2$. Moreover, $(\boldsymbol{p} + \boldsymbol{u} - \boldsymbol{e})$ is uniformly distributed among the codewords generated by $\boldsymbol{A}$, hence $\boldsymbol{s} \overset{\$}{\leftarrow} \mathbb{F}_{q^m}^{k+\ell}$.

According to the indistinguishability of $\mathcal{D}_2$ and $\mathcal{D}_3$, the distribution of $\boldsymbol{e}'$ and $\boldsymbol{e}$ are computationally indistinguishable, and $\boldsymbol{s}$ and $\boldsymbol{s}'$ are both uniformly distributed. Finally, based on the assumption that the DRSD problem is hard, $(\boldsymbol{p} + \boldsymbol{u})$ and $(\boldsymbol{p}' + \boldsymbol{u}')$ are indistinguishable except the negligible advantage ϵ_{drsd}.

Putting all together, the advantage of an adversary to distinguish $\mathcal{D}_0$ and $\mathcal{D}_1$ is bounded by $\frac{2}{q} + \epsilon_{\mathsf{drsd}}$. □

C Security Proof of RIBE

Proof. Our proof proceeds in a sequence of games where the first game is identical to the IND-sRID-CPA game from Definition 2. In the last game of the sequence, the adversary has zero advantage. We show a PPT adversary cannot distinguish between the games which will prove that the adversary has a negligible advantage in winning the original IND-sRID-CPA game. The DRSD problem is used in proving that games G_0 and G_1 are indistinguishable; LRPC$^+$ problem is used in proving that games G_1 and G_2 are indistinguishable; and DRSL problem is used in proving that games G_2 and G_3 are indistinguishable.

To ensure that no information is leaked about the decryption keys during the game, we consider two kinds of adversaries:

- Type 1 Adversary: It chooses to be challenged on the targeted identity id^* but is revoked before or on time t^*.
- Type 2 Adversary : It does not challenge the target identity id^* at any time.

At the start of the game, we flip a coin $rvk \overset{\$}{\leftarrow} \{0, 1\}$ for the type of adversary with whom we will be faced. All the sequence of the games for these two types of adversaries are similar except for game G_1. 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.

Regardless of the value of rvk, for each node $\theta \in \mathsf{BT}$, we split the public key element $\boldsymbol{u} \in \mathbb{F}_{q^m}^{n+\ell}$ into two specific additive shares $(\boldsymbol{u}_{\theta,1}, \boldsymbol{u}_{\theta,2})$ such that $\boldsymbol{u} = \boldsymbol{u}_{\theta,1} + \boldsymbol{u}_{\theta,2}$. We model hash functions H_1 and H_2 as random oracles.

Game $\mathbf{G}_0$. This is the original IND-sRID-CPA game from Definition 2. An adversary $\mathcal{A}$ outputs the challenge identity id^* to be targeted at time t^*. Then RIBE.Setup is run, and the adversary $\mathcal{A}$ is fed with the public parameters $\mathsf{PP} = (\boldsymbol{A}, \boldsymbol{G}, \boldsymbol{u})$. $\mathcal{A}$ can ask queries to H_1, H_2 and private key, update key oracles. $\mathcal{A}$ also ouputs a pair of messages (m_0, m_1). Next a challenge ciphertext is produced by flipping a coin β and producing a ciphertext $\mathsf{CT}^* = (\boldsymbol{C}^*, \boldsymbol{x}^*) = \mathcal{E}(\mathsf{PP}, \mathsf{id}^*, \mathsf{t}^*, m_\beta)$.

On input $\mathsf{CT}^* = (\boldsymbol{C}^*, \boldsymbol{x}^*)$, $\mathcal{A}$ in guessing phase can coninue to ask queries to H_1, H_2 and private key, update key oracles, which are different from id^* and t^* depending on the type of adevrsary as described above. Finally, $\mathcal{A}$ outputs β'.

We denote by S_0 the event $\beta' = \beta$ and use the same notation S_n in any game $\mathbf{G}_n$ for $n = 1, 2, \dots$ as given below.

$$\mathsf{Adv}^{\mathsf{IND\text{-}sRID\text{-}CPA}}_{\mathcal{A},\mathsf{RIBE}}(\lambda) := \epsilon_{\mathsf{ribe}} = \left|\Pr[\mathsf{S}_0] - \frac{1}{2}\right|,$$

where ϵ_{ribe} is the bound on the advantage of attacks against the RIBE scheme.

As this is a real attack game, for a private key query on identity id, the private key generation algorithm $\mathcal{SK}(\mathsf{PP}, \mathsf{MSK}, \mathsf{id}, \mathsf{RL}, \mathsf{ST})$ is run and the private key is given to $\mathcal{A}$. We recall this algorithm : let $v \in \mathsf{BT}$ be the node that we assign to id.

- $\forall \theta \in \mathsf{Path}(v)$, if $\boldsymbol{u}_{\theta,1}$, $\boldsymbol{u}_{\theta,2}$ are undefined, then pick $\boldsymbol{u}_{\theta,1} \xleftarrow{\$} \mathbb{F}_{q^m}^{n+\ell}$, set $\boldsymbol{u}_{\theta,2} := \boldsymbol{u} - \boldsymbol{u}_{\theta,1}$, and store them in node θ.
- Compute $\boldsymbol{p}_1 = H_1(\mathsf{id})$ and syndrome $\boldsymbol{x}_{\theta,1} = \boldsymbol{H}'\boldsymbol{p}_1^{\mathsf{T}} + \boldsymbol{H}'\boldsymbol{u}_{\theta,1}^{\mathsf{T}}$.
- Sample $\boldsymbol{e}_{\theta,1} \in \mathbb{F}_{q^m}^{n+\ell}$ of weight r, as $\boldsymbol{H}'\boldsymbol{e}_{\theta,1}^{\mathsf{T}} = \boldsymbol{x}_{\theta,1}$ using RankSign algorithm with trapdoor $\boldsymbol{T}$.
- Compute $\boldsymbol{s}_{\theta,1} \in \mathbb{F}_{q^m}^{k+\ell}$ as $\boldsymbol{p}_1 + \boldsymbol{u}_{\theta,1} = \boldsymbol{s}_{\theta,1}\boldsymbol{A} + \boldsymbol{e}_{\theta,1}$.
- Output $\mathsf{SK}_{\mathsf{id}} := \{(\theta, \boldsymbol{s}_{\theta,1})\}_{\theta \in \mathsf{Path}(v)}$, ST.

For a update key query at time t, the key update generation algorithm $\mathcal{KU}(\mathsf{PP}, \mathsf{MSK}, \mathsf{t}, \mathsf{RL}, \mathsf{ST})$ is run and the updated key is given to $\mathcal{A}$. We also recall this algorithm : let $v \in \mathsf{BT}$ be the node that we assign to id.

- For all $\theta \in \mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL}, \mathsf{t})$, if $\boldsymbol{u}_{\theta,1}$, $\boldsymbol{u}_{\theta,2}$ are undefined, then pick $\boldsymbol{u}_{\theta,2} \xleftarrow{\$} \mathbb{F}_{q^m}^{n+\ell}$, set $\boldsymbol{u}_{\theta,1} := \boldsymbol{u} - \boldsymbol{u}_{\theta,2}$, and store them in node θ.
- Compute $\boldsymbol{p}_2 = H_2(\mathsf{t})$ and syndrome $\boldsymbol{x}_{\theta,2} = \boldsymbol{H}'\boldsymbol{p}_2^{\mathsf{T}} + \boldsymbol{H}'\boldsymbol{u}_{\theta,2}^{\mathsf{T}}$.
- Sample $\boldsymbol{e}_{\theta,2} \in \mathbb{F}_{q^m}^{n+\ell}$ of weight r, as $\boldsymbol{H}'\boldsymbol{e}_{\theta,2}^{T} = \boldsymbol{x}_{\theta,2}$ using RankSign algorithm with trapdoor $\boldsymbol{T}$.
- Compute $\boldsymbol{s}_{\theta,2} \in \mathbb{F}_{q^m}^{k+\ell}$ as $\boldsymbol{p}_2 + \boldsymbol{u}_{\theta,2} = \boldsymbol{s}_{\theta,2}\boldsymbol{A} + \boldsymbol{e}_{\theta,2}$.
- Output $\mathsf{KU}_{\mathsf{t}} := \{(\theta, \boldsymbol{s}_{\theta,2})\}_{\theta \in \mathsf{KUNodes}(\mathsf{BT},\mathsf{RL},\mathsf{t})}$, ST.

Finally, the decryption key generation $\mathcal{DK}(\mathsf{SK}_{\mathsf{id}}, \mathsf{KU}_{\mathsf{t}})$ algorithm is run : on input a private key $\mathsf{SK}_{\mathsf{id}} := \{(i, \boldsymbol{s}_{i,1})\}_{i \in I}$, and key update $\mathsf{KU}_{\mathsf{t}} := \{(j, \boldsymbol{s}_{j,2})\}_{j \in J}$ for some set of nodes I, J; it performs the following steps.

- $\forall (i, \boldsymbol{s}_{i,1}) \in \mathsf{SK}_{\mathsf{id}}$, $(j, \boldsymbol{s}_{j,2}) \in \mathsf{KU}_{\mathsf{t}}$, if $\exists (i,j)$ s.t. $i = j$ then $\mathsf{DK}_{\mathsf{id},\mathsf{t}} \leftarrow (\boldsymbol{s}_{i,1}, \boldsymbol{s}_{j,2})$; else (if $\mathsf{SK}_{\mathsf{id}}$ and KU_{t} do not have any node in common) $\mathsf{DK}_{\mathsf{id},\mathsf{t}} \leftarrow \perp$.
- Output $\mathsf{DK}_{\mathsf{id},\mathsf{t}} = (s_1, s_2)$.

Game $\mathbf{G}_1$. In this game, we modify the answers to the private key queries and key update queries so that it does not require the trapdoor $\boldsymbol{T}$ anymore. In order to make the answers consistent, we also need to simulate the queries of random oracles H_1 and H_2. To do so, we maintain two lists $\mathsf{List}_{\mathsf{H}1}$ and $\mathsf{List}_{\mathsf{H}2}$, initially set to be empty. $\mathsf{List}_{\mathsf{H}1}$ stores the tuple $(\mathsf{id}, \boldsymbol{p}_1, \boldsymbol{u}_1, \boldsymbol{s}_1, \boldsymbol{e}_1)$, where $\boldsymbol{p}_1$ is the value that we respond to the H_1 query on id, and $\boldsymbol{s}_1$ is the secret key which corresponds to the public key $\boldsymbol{p}_1$ we generate. $\mathsf{List}_{\mathsf{H}2}$ stores the tuple $(\mathsf{t}, \boldsymbol{p}_2, \boldsymbol{u}_2, \boldsymbol{s}_2, \boldsymbol{e}_2)$, where $\boldsymbol{p}_2$ is the value that we respond to the H_2 query on t, $\boldsymbol{s}_2$ is the secret key which corresponds to the public key $\boldsymbol{p}_2$ we generate.

The challenger generates vectors for the private key query of id^* and key update query of t^* as follows (without loss of generality, we pick a node v^* from BT beforehand, to which id^* may be assigned):

- If $rvk = 0$, we simulate the game to face Type-1 adversary.
  - On identity id^* query : if there exists $\boldsymbol{p}_1$ such that $(\mathsf{id}^*, \boldsymbol{p}_1, \cdot, \cdot, \cdot) \in H_1$, return $\boldsymbol{p}_1$; otherwise we pick a random vector $\boldsymbol{p}_1 \xleftarrow{\$} \mathbb{F}_{q^m}^{n+\ell}$. Next, we generate $\boldsymbol{u}_{\theta,1}$ and $\boldsymbol{u}_{\theta,2}$ for every node θ in BT as follows:
    - $*$ For all $\theta \in \mathsf{Path}(v^*)$, we pick $\boldsymbol{s}_{\theta,1} \xleftarrow{\$} \mathbb{F}_{q^m}^{n-k}$ and $\boldsymbol{e}_{\theta,1} \xleftarrow{\$} \mathbb{F}_{q^m}^{n}$. We then generate $\boldsymbol{u}_{\theta,1} = \boldsymbol{s}_{\theta,1}\boldsymbol{A} + \boldsymbol{e}_{\theta,1} - \boldsymbol{p}_1$, and set $\boldsymbol{u}_{\theta,2} = \boldsymbol{u} - \boldsymbol{u}_{\theta,1}$. We store $\boldsymbol{u}_{\theta,1}$ and $\boldsymbol{u}_{\theta,2}$ in node θ.
    - $*$ For each node θ, we add the tuple $(\mathsf{id}^*, \boldsymbol{p}_1, \boldsymbol{u}_{\theta,1}, \boldsymbol{s}_{\theta,1}, \boldsymbol{e}_{\theta,1})$ to the List_{H_1}.
  - On time t^* query : if there exists $\boldsymbol{p}_2$ such that $(\mathsf{t}^*, \boldsymbol{p}_2, \cdot, \cdot, \cdot) \in H_2$, return $\boldsymbol{p}_2$; otherwise we pick a random string $\boldsymbol{p}_2 \xleftarrow{\$} \mathbb{F}_{q^m}^{n+\ell}$. Next, we generate $\boldsymbol{u}_{\theta,1}$ and $\boldsymbol{u}_{\theta,2}$ for every node θ in BT as follows:
    - $*$ For all $\theta \notin \mathsf{Path}(v^*)$, we pick $\boldsymbol{s}_{\theta,2} \xleftarrow{\$} \mathbb{F}_{q^m}^{n-k}$ and $\boldsymbol{e}_{\theta,2} \xleftarrow{\$} \mathbb{F}_{q^m}^{n}$. We then generate $\boldsymbol{u}_{\theta,2} = \boldsymbol{s}_{\theta,2}\boldsymbol{A} + \boldsymbol{e}_{\theta,2} - \boldsymbol{p}_2$, and set $\boldsymbol{u}_{\theta,1} = \boldsymbol{u} - \boldsymbol{u}_{\theta,2}$. We store $\boldsymbol{u}_{\theta,1}$ and $\boldsymbol{u}_{\theta,2}$ in node θ.
    - $*$ For each node θ, we add the tuple $(\mathsf{t}^*, \boldsymbol{p}_2, \boldsymbol{u}_{\theta,2}, \boldsymbol{s}_{\theta,2}, \boldsymbol{e}_{\theta,2})$ to the List_{H_2}.

  Since id^* has been revoked before adversary $\mathcal{A}$ has queried for update key at time t^*, thus no ancestor of v^* lies in the set Y determined by $\mathsf{KUNodes}$ at time t^*, i.e., we have $\mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL}, \mathsf{t}) \cap \mathsf{Path}(v^*) = \phi$. Then the challenger responds a private key query for id^* but using $\{(\theta, \boldsymbol{s}_{\theta,1})\}_{\theta \in \mathsf{Path}(v^*)}$ and a update key query for t^* by using $\{(\theta, \boldsymbol{s}_{\theta,2})\}_{\theta \in \mathsf{KUNodes}(\mathsf{BT},\mathsf{RL},\mathsf{t}^*)}$.

- If $rvk = 1$, we simulate the game to face Type-2 adversary. Recall that Type-2 adversary $\mathcal{A}$ does not query the private key of id^* at any time. We generate $\boldsymbol{u}_{\theta,1}$ and $\boldsymbol{u}_{\theta,2}$ for every node θ in BT as follows:
  - On time t query : if there exists $\boldsymbol{p}_2$ such that $(\mathsf{t}, \boldsymbol{p}_2, \cdot, \cdot, \cdot) \in H_2$, return $\boldsymbol{p}_2$; otherwise we pick a random string $\boldsymbol{p}_2 \xleftarrow{\$} \mathbb{F}_{q^m}^{n+\ell}$. Next, we generate $\boldsymbol{u}_{\theta,1}$ and $\boldsymbol{u}_{\theta,2}$ for every node θ in BT as follows:
    - $*$ We pick $\boldsymbol{s}_{\theta,2} \xleftarrow{\$} \mathbb{F}_{q^m}^{n-k}$ and $\boldsymbol{e}_{\theta,2} \xleftarrow{\$} \mathbb{F}_{q^m}^{n}$. We generate $\boldsymbol{u}_{\theta,2} = \boldsymbol{s}_{\theta,2}\boldsymbol{A} + \boldsymbol{e}_{\theta,2} - \boldsymbol{p}_2$, and set $\boldsymbol{u}_{\theta,1} = \boldsymbol{u} - \boldsymbol{u}_{\theta,2}$. We store $\boldsymbol{u}_{\theta,1}$ and $\boldsymbol{u}_{\theta,2}$ in node θ.
    - $*$ For each node θ, we add the tuple $(\mathsf{t}, \boldsymbol{p}_2, \boldsymbol{u}_{\theta,2}, \boldsymbol{s}_{\theta,2}, \boldsymbol{e}_{\theta,2})$ to the List_{H_2}.

Since id^* has never been queried, the challenger responds a update key query for t^* by using $\{(\theta, \boldsymbol{s}_{\theta,2})\}_{\theta \in \mathsf{KUNodes}(\mathsf{BT},\mathsf{RL},\mathsf{t}^*)}$. Note that whenever adversary $\mathcal{A}$ asks identity id query to H_1, we return $\boldsymbol{p}_1$ to $\mathcal{A}$, whereas for private key queries, we return $\boldsymbol{s}_1$ to $\mathcal{A}$. Similarly, whenever adversary $\mathcal{A}$ asks time t query to H_2, we return $\boldsymbol{p}_2$ to $\mathcal{A}$, whereas for key update queries we return $\boldsymbol{s}_2$ to $\mathcal{A}$. We also note that for each node $\theta \in \mathsf{Path}(v)$, the set of q_{H_1} tuples $(\boldsymbol{p}_1, \boldsymbol{u}_{\theta,1}, \boldsymbol{s}_{\theta,1}, \boldsymbol{e}_{\theta,1})$ in the previous game are sampled from the distribution $\mathcal{D}_0$ (as defined in Lemma 1) and the set of q_{H_1} tuples $(\boldsymbol{p}_1, \boldsymbol{u}_{\theta,1}, \boldsymbol{s}_{\theta,1}, \boldsymbol{e}_{\theta,1})$ in this game are sampled from the distribution $\mathcal{D}_1$ (as defined in Lemma 1). Similarly, the set of q_{H_2} tuples $(\boldsymbol{p}_2, \boldsymbol{u}_{\theta,2}, \boldsymbol{s}_{\theta,2}, \boldsymbol{e}_{\theta,2})$ in the previous game are sampled from the distribution $\mathcal{D}_0$ and the set of q_{H_2} tuples $(\boldsymbol{p}_2, \boldsymbol{u}_{\theta,2}, \boldsymbol{s}_{\theta,2}, \boldsymbol{e}_{\theta,2})$ in this game are sampled from the distribution $\mathcal{D}_1$. Thus, from Lemma 1, we have

$$|\mathsf{Pr}[\mathsf{S}_1] - \mathsf{Pr}[\mathsf{S}_0]| \leq (q_{H_1} + q_{H_2}).\left(\frac{2}{q} + \epsilon_{\mathsf{drsd}}\right),$$

where ϵ_{drsl} is the bound on the successful probability of the attacks against the DRSD problem.

Game $\mathbf{G}_2$. In this game, we define matrix $\boldsymbol{A}$ to be a random matrix. We keep all the simulations for secret key and update key queries exactly same as in game G_1. By the assumption that the LRPC^+ problem is hard, i.e., it is hard to distinguish augmented LRPC code from a random code, this game is indistinguishable from the previous game G_1. Therefore, we have

$$|\mathsf{Pr}[\mathsf{S}_2] - \mathsf{Pr}[\mathsf{S}_1]| \leq \epsilon_{\mathsf{lrpc}^+},$$

where $\epsilon_{\mathsf{lrpc}^+}$ is the bound on the successful probability of the attacks against the LRPC^+ problem.

Game $\mathbf{G}_3$. In this game, we replace $(\boldsymbol{C}^*, \boldsymbol{x}^*)$ by $(\boldsymbol{C}^* \xleftarrow{\$} \mathbb{F}_{q^m}^{(k+\ell)\times n'}, \boldsymbol{x}^* \xleftarrow{\$} \mathbb{F}_{q^m}^{n'})$. As $\boldsymbol{x}^*$ is perfectly random, $\boldsymbol{x}^* - \boldsymbol{m}^*\boldsymbol{G}$ is also perfectly random. In other words, this game replaces

$$\left(\frac{\boldsymbol{A}}{\boldsymbol{p}_1 + \boldsymbol{p}_2 + \boldsymbol{u}}\right)\boldsymbol{V} = \left(\frac{\boldsymbol{C}^*}{\boldsymbol{x}^* - \boldsymbol{m}^*\boldsymbol{G}}\right)$$

by a perfectly random matrix. The indistinguishability of two games G_2 and G_3 follows from the hardness of the DRSL problem, applying it to the matrix $\boldsymbol{A}' = \left(\frac{\boldsymbol{A}}{\boldsymbol{p}_1 + \boldsymbol{p}_2 + \boldsymbol{u}}\right)$, which is perfectly random because $\boldsymbol{A}$, $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ and $\boldsymbol{u}$ are all perfectly random. Thus, we have

$$|\mathsf{Pr}[\mathsf{S}_3] - \mathsf{Pr}[\mathsf{S}_2]| \leq \epsilon_{\mathsf{drsl}},$$

where ϵ_{drsl} is the bound on the successful probability of the attacks against the DRSL problem.

Game $\mathbf{G}_4$. In this last game, as the challenge ciphertext $\mathsf{CT}^* = (\boldsymbol{C}^*, \boldsymbol{x}^*)$ is perfectly random and independent, the bit β is perfectly hidden to any adversary $\mathcal{A}$. Therefore, we have

$$\mathsf{Pr}[\mathsf{S}_3] = \mathsf{Pr}[\mathsf{S}_4] \;\text{ and }\; \mathsf{Pr}[\mathsf{S}_4] = \frac{1}{2}.$$

Putting all together, we have

$$\epsilon_{\mathsf{ribe}} \leq \left(q_{H_1} + q_{H_2}\right) \cdot \left(\frac{2}{q} + \epsilon_{\mathsf{drsd}}\right) + \epsilon_{\mathsf{lrpc}^+} + \epsilon_{\mathsf{drsl}}.$$

This completes the proof. □

D Example of Revocation using KUNodes Algorithm

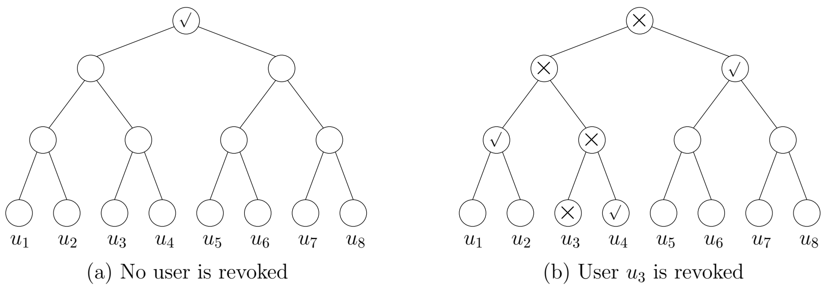
✓
u_1 u_2 u_3 u_4 u_5 u_6 u_7 u_8
(a) No user is revoked
×
× ✓
✓ ×
× ✓
u_1 u_2 u_3 u_4 u_5 u_6 u_7 u_8
(b) User u_3 is revoked

Fig. 1: A graphical description of the $\mathsf{KUNodes}(\mathsf{BT}, \mathsf{RL}, \mathsf{t})$ algorithm.