Post-Quantum One-Time Linkable Ring Signature and Application to Ring Confidential Transactions in Blockchain (Lattice RingCT v1.0) (*Cryptography ePrint Archive Version*)

Wilson Abel Alberto Torres (✉)[1] , Ron Steinfeld (✉)[1], Amin Sakzad (✉)[1], Joseph K. Liu (✉)[1], Veronika Kuchta[1], Nandita Bhattacharjee[1], Man Ho Au[2], and Jacob Cheng[3]

[1] Faculty of IT, Monash University, Melbourne, Australia
{Wilson.Torres,Ron.Steinfeld,Amin.Sakzad,Joseph.Liu,Veronika.Kuchta, Nandita.Bhattacharjee}@monash.edu
[2] Hong Kong Polytechnic University, Hung Hom, Hong Kong
csallen@comp.polyu.edu.hk
[3] Collinstar Capital, Melbourne, Australia
jacob@collinstar.com

Abstract. In this paper, we construct a Lattice-based one-time Linkable Ring Signature (L2RS) scheme, which enables the public to verify if two or more signatures were generated by same signatory, whilst still preserving the anonymity of the signatory. The L2RS provides unconditional anonymity and security guarantees under the Ring Short Integer Solution (Ring-SIS) lattice hardness assumption. The proposed L2RS scheme is extended to be applied in a protocol that we called *Lattice Ring Confidential transaction (Lattice RingCT) RingCT v1.0*, which forms the foundation of the privacy-preserving protocol in any post-quantum secure cryptocurrency such as Hcash.

Keywords: Linkable Ring Signature, Lattice-Based Cryptography, Post-Quantum Cryptography, Cryptocurrencies

1 Introduction

The notion of a *Ring Signature* scheme was initially formalised in [1]. This scheme allows signing a message on behalf of a spontaneous group of signers, while preserving the anonymity of the signer. The creation of a ring signature does not require members of a group to cooperate, meaning that this scheme will not longer have a manager who eventually can reveal the identity of the signer, and thus the anonymity will be unconditionally preserved. This approach was a remarkable security improvement when compared with the group signature scheme [2] where a group manager was part of its construction. Later, an extended property called *Linkability* was introduced in a ring signature scheme, under the name of *Linkable Spontaneous Anonymous Group* but is now known as *Linkable Ring Signature* [3]. The linkability property of ring signatures allows one to detect if two signatures were generated by the same signer (using the same private-key) whilst still preserving their anonymity. This scheme was proved to be secure under the discrete logarithm assumption and in Random Oracle Model (ROM). In comparison with previous unlinkable ring signature schemes, this scheme adds an efficient algorithm to verify the linkability property. Each signature (σ) is accompanied by a label (or tag), which is computed based on the signer's private key and a hash function modelled as a random oracle in a deterministic manner. The label can be used by the linking algorithm the check whether two signatures are created by the same signer. Specifically, if the labels accompanying two signatures are the same, it means that the two signatures are created by the same signer. This particular feature opens the possibility of many practical scenarios [3–5], such as, cryptocurrency, in particular the RingCT confidential transaction protocol adapted in Monero cryptocurrency [6], and e-voting applications.

Nevertheless, the above ring signature schemes are based on classical number-theory mathematical assumptions, for instance, the hardness of discrete logarithm [7, 8] and factoring large numbers [9]. As a consequence, they are believed to be vulnerable with the onset of powerful quantum computers [10]. This situation has sparked the primarily motivation of researchers in the area of post-quantum cryptography to construct secure approaches against these type of computers. Among the alternatives, lattice-based cryptography has attracted the attention of this field due to its distinguishing features and new applications. Algorithms based on lattices tend to be efficient, simple, highly parallelisable and provide strong provable security guarantees [11, 12].

1.1 Contribution

- We construct a Lattice-based one-time Linkable Ring Signature (**L2RS**) scheme. Our L2RS is a generalisation of the BLISS [13] scheme which is currently one of the practical lattice digital signatures. L2RS provides unconditional anonymity as well as unforgeability security guarantees under the hardness of standard lattice assumptions.
- We devise a new cryptocurrency privacy-preserving protocol that we call **Lattice RingCT v1.0**. This protocol employs our proposed post-quantum L2RS as a fundamental building block along with a homomorphic commitment primitive to provide post-quantum secure confidential transactions which forms the foundation of the privacy-preserving protocol for blockchain cryptocurrencies, such as Hcash.

This paper is organised in eight parts, including the introduction. Section 2 gives a brief background of the current linkable ring signature approaches. After describing the technical description used in Section 3 and the security model in Section 4, this research shows the construction of the L2RS scheme in Section 5 along with the security analysis in Section 6. In Section 7, we present an application of this L2RS in a cryptocurrency protocol that we called Lattice

RingCT v1.0. Finally, a performance analysis of these proposals is presented in Section 8.

2 Related Work

Linkable Ring Signature (LRS) primitive is receiving attention thanks to its distinguishing capabilities of anonymously detecting if two linkable ring signatures are being signed by same signatory. Most of the current linkable ring signature schemes along with different variants [3, 5, 14–25] rely on the hardness assumptions of classical cryptography. Technically, this primitive uses a linkability tag that has a secure relationship with the signer's publick-key, then the LRS uses this tag to verify whether or not a singer signs two signatures. Monero, a cryptocurrency application, exploits this property to prevent double spending while keeping the user's anonymity [6].

However, this primitive and its variants will be vulnerable to quantum attacks [10, 26, 12]. This situation has led to a new area in the field of cryptography called *Post-Quantum Cryptography*, aimed at constructing new cryptographic algorithms that are intractable even in the presence of powerful quantum computers. Among the current post-quantum cryptographic proposals [12, 27], lattice-based cryptography has attracted the attention of cryptographers. It is a candidate to be standardised as a post-quantum cryptography solution due to its efficiency, parallelism, uniqueness and strong security assurances under the *worst-case hardness* of lattice problems, which is significantly better than the *average-case hardness* of other cryptographic constructions [28, 11].

Digital signatures which are constructed based on lattice-based cryptography can be categorised into GGH/NTRUSign [29, 30], Hash-and-sign [31] and Fiat-Shamir signatures [32]. Fiat-Shamir transformation [33, 34] is used by the Bimodal Lattice Signature Scheme (BLISS) [13], which is currently one of the most practical lattice-based digital signature schemes. BLISS has been constructed using the following well known lattice-based cryptography problems, the Short Integer Solution (SIS) [35], Ring-SIS [36] and the Ring-LWE (Learning With Errors) [37] problems [4]. The Ring-SIS version of BLISS offers practical runtime and key sizes. Moreover, this scheme uses a probabilistic test based on rejection sampling technique to make the distribution of the private-key independent, an important property that completely hides the private-key from any adversary.

Several lattice-based ring signatures schemes have been proposed in [38–43] and there were recently three LRS proposals based on lattice-based cryptography. The first of these constructions [44], is based on the development of a lattice-based weak Pseudo Random Function (wPRF), an accumulator scheme (Acc) and a framework named as Zero-Knowledge Arguments of Knowledge (ZKAoK). These techniques are used to construct LRS schemes where the security guarantees for the LRS properties' *unforgeability, anonymity, linkability and non-slanderability* rely on the lattice problems. The second lattice LRS scheme

[4] The *Ring*-SIS and *Ring*-LWE refer to the *Ring* mathematical structure and differ from the *Ring* in the *Ring Signature* scheme.

[45], uses ideal lattices along with a lattice-based homomorphic commitment in its construction. The security properties are based on the hardness of lattices; however, there is no discussion as to how to secure the scheme in terms of *non-slanderability.* This scheme is shown to be used in a cryptocurrency application. The last lattice LRS proposal [46], is devised using lattice-based variants named Module-SIS and Module-LWE problems and its security properties rely on the lattice assumptions.

Our (L2RS) scheme was designed independently and concurrently with [46]. The schemes share similar features, but our scheme offers unconditional anonymity. The construction of this work, which we call Lattice-based one-time Linkable Ring Signature (L2RS), is an extension of BLISS, a demonstrated practical lattice-based digital signature [13]. It is secure in terms of *unforgeability, linkability and non-slanderability* under the lattice hardness of the Ring-SIS problem and unlike the above Lattice-based LRS schemes [44, 45] and [46], the L2RS scheme achieves *unconditional anonymity*, meaning that this scheme will be secure even if an adversary has unlimited computational resources and time. As an application of this construction, we designed the Lattice RingCT v1.0, a cryptocurrency protocol that provides confidential transactions and which its security guarantees rely on our post-quantum cryptographic L2RS scheme.

3 Preliminaries

The ring $\mathcal{R} = \mathbb{Z}[x]/f(x)$ is a degree-n polynomial ring, where $f(x)$ is a polynomial of degree of n. The ring $\mathcal{R}_q$ is then defined to be the quotient ring $\mathcal{R}_q = \mathcal{R}/(q\mathcal{R}) = \mathbb{Z}_q[x]/f(x)$, where $\mathbb{Z}_q$ denotes the set of all positive integers modulo q (a prime number $q = 1 \bmod 2n$) in the interval $[-q/2, q/2]$ and $f(x) = x^n + 1$ where n is a power of 2. The challenge $\mathcal{S}_{n,\kappa}$, is the set of all binary vectors of length n and weight κ. Two hash functions modeled as Random Oracle Model (ROM), H_1 with range $\mathcal{S}_{n,\kappa} \subseteq \mathcal{R}_{2q}$, and H_2 with range $\mathcal{R}_q^{1\times(m-1)}$. When we use $x \leftarrow D$, it means that x is chosen from the distribution D, and $y \leftarrow \mathcal{R}_q$ means that y is chosen uniformly at random according to $\mathcal{R}_q$. Matrices are written in bold upper case letters whereas vectors are represented in bold lower case letters, where vectors are column vectors and $\mathbf{v}^T$ is the transpose of the vector $\mathbf{v}$. The hardness assumption of this work is the Ring-SIS (Short Integer Solution) problem and this is defined as follows.

Definition 1 ($\mathcal{R}$-SIS$^{\mathcal{K}}_{q,n,m,\beta}$ problem). (Based on [13], Def. 2.3). *Let denote $\mathcal{K}$ some uniform distribution over the ring $\mathcal{R}_q^{n\times m}$. Given a random matrix $\mathbf{A} \in \mathcal{R}_q^{n\times m}$ sampled from $\mathcal{K}$ distribution, find a non-zero vector $\mathbf{v} \in \mathcal{R}_q^m$ such that $\mathbf{Av} = \mathbf{0}$ and $\|\mathbf{v}\|_2 \leq \beta$, , where $\|\cdot\|_2$ denotes the Euclidean norm.*

Lemma 1 (Leftover Hash Lemma (LHL)). (Based on [13], Lemma B.1). *Let $\mathcal{H}$ be a universal hash family of hash functions from X to Y. If $h \leftarrow \mathcal{H}$ and $x \leftarrow X$ are chosen uniformly and independently, then the statistical distance between (h,h(x)) and the uniform distribution on $\mathcal{H} \times Y$ is at most $\frac{1}{2}\sqrt{|Y|/|X|}$.*

Remark 1. We use this lemma for a SIS family of hash function $H(\mathbf{S}_0) = \mathbf{A}'_0 \cdot \mathbf{S}_0 \in \mathcal{R}_q$, with $\mathbf{S}_0 \in \mathsf{Dom}_{\mathbf{S}_0}$, where each function is indexed by $\mathbf{A}'_0 \in \mathcal{R}_q^{1\times(m-1)}$. The $\mathsf{Dom}_{\mathbf{S}_0} \subseteq \mathcal{R}_q^{1\times(m-1)}$ consists of a vector of $\mathcal{R}_q$ elements with coefficients in set $\Gamma \overset{\text{def}}{=} (-2^\gamma, 2^\gamma)$. This is a universal hash family if $s - s'$ is invertible in $\mathcal{R}_q$ for all distinct pairs s, s' in $\Gamma^n \subseteq \mathcal{R}_q$. This can be guaranteed by appropriate choice q of $\mathcal{R}_q$, e.g. as shown in ([47], Corollary 1.2), it is sufficient to use q such that $f(x) = x^n + 1$ factors into k irreducible factors $\mod q$ and $2^\gamma < \frac{1}{\sqrt{k}} \cdot q^{1/k}$. We assume that $\mathcal{R}_q$ is chosen to satisfy this condition.

Lemma 2 (Rejection Sampling). (Based on [13], Lemma 2.1). *Let V be an arbitrary set, and $h : V \to \mathbb{R}$ and $f : \mathbb{Z}^m \to R$ be probability distributions. If $g_v : \mathbb{Z}^m \to R$ is a family of probability distributions indexed by $v \in V$ with the property that there exists a $M \in \mathbb{R}$ such that $\forall v \in V, \forall \mathbf{v} \in \mathbb{Z}^m, M \cdot g_v(\mathbf{z}) \geq f(\mathbf{z})$. Then the output distributions of the following two algorithms are identical:*

1. *$v \leftarrow h, z \leftarrow g_v$, output$(\mathbf{z}, v)$ with probability $f(\mathbf{z})/(M \cdot g_v(\mathbf{z}))$.*
2. *$v \leftarrow h, z \leftarrow f$, output$(\mathbf{z}, v)$ with probability $1/M$.*

Definition 2 (Gaussian Distribution). *The discrete Gaussian distribution over $\mathbb{Z}^m$ with standard deviation $\sigma \in \mathbb{R}$ and center at zero, is defined by $D^m_\sigma(\boldsymbol{x}) = \rho_\sigma(\boldsymbol{x})/\rho_\sigma(\mathbb{Z}^m)$, where ρ_σ is m dimensional Gaussian function $\rho_\sigma(\boldsymbol{x}) = \exp\left(\frac{-\|\boldsymbol{x}\|^2}{2\sigma^2}\right)$.*

4 Security model

4.1 Structure of Lattice-based one-time Linkable Ring Signature (L2RS)

An L2RS scheme has four PPT algorithms (Setup, KeyGen, SigGen, SigVer, SigLink). In addition, the correctness of this scheme is satisfied by the Signature correctness SigGen Correctness and the Linkability correctness SigLink Correctness. These algorithms are defined as follows:

- Setup: a PPT algorithm that takes the security parameter λ and produces the Public Parameters (Pub-Params).
- KeyGen: a PPT algorithm that by taking the Pub-Params, it produces a pair of keys: the public-key and the private-key.
- SigGen: a PPT algorithm that receives a singer π's private-key, a message μ and the list of users' public-keys in the ring signature L, and outputs a signature $\sigma_L(\mu)$.
- SigVer: a PPT algorithm that takes a signature $\sigma_L(\mu)$, a list of public-keys L and the message μ, and it verifies if this signature was legitimately created, this algorithm outputs either: **Accept** or **Reject**.
- SigLink: a PPT algorithm that inputs two valid signatures $\sigma_L(\mu_1)$ and $\sigma_L(\mu_2)$ and it anonymously determines if these signatures were produced by same signer π. Thus, this algorithm has a deterministic output: **Linked** or **Unlinked**.

Correctness requirements:

- SigGen Correctness: this guarantees that valid signatures signed by honest signers will be accepted with overwhelming probability by a verifier.
- SigLink Correctness: this ensures that if two signatures $\sigma_L(\mu_1)$ and $\sigma_L(\mu_2)$ are signed by an honest signer π, SigLink will output **Linked** with overwhelming probability.

4.2 Oracles for adversaries

The following oracles will be available to any adversary who tries to break the security of an L2RS scheme:

1. $\mathbf{A}_i \leftarrow \mathcal{JO}(\perp)$. The *Joining Oracle*, on request, adds a new user to the system. It returns the public-key $\mathbf{A} \in \mathcal{R}_{2q}^{1\times m}$ of the new user.
2. $\mathbf{S}_i \leftarrow \mathcal{CO}(\mathbf{A}_i)$. The *Corruption Oracle*, on input a public-key $\mathbf{A}_i \in \mathcal{R}_{2q}^{1\times m}$ that is a query output of $\mathcal{JO}$, returns the corresponding private-key $\mathbf{S}_i \in \mathcal{R}_q^{m\times 1}$.
3. $\sigma'_L(\mu) \leftarrow \mathcal{SO}(w, L, \mathbf{A}_\pi, \mu)$. The *Signing Oracle*, a group size w, a set L of w public-keys, the public-key of the signer $\mathbf{A}_\pi$, and a message μ, returns a valid signature $\sigma'_L(\mu)$.

4.3 Threat Model

- One-time Unforgeability. One time unforgeability for the L2RS scheme is defined in the following game between a simulator $\mathcal{S}$ and an adversary $\mathcal{A}$ who has access to the oracles $\mathcal{JO}$, $\mathcal{CO}$, $\mathcal{SO}$ and the random oracle:
  1. $\mathcal{S}$ generates and gives the list of public-keys L to $\mathcal{A}$.
  2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
  3. $\mathcal{A}$ gives $\mathcal{S}$ a ring signature size w, a set L of w public-keys, a message μ and a signature $\sigma_L(\mu)$.

  $\mathcal{A}$ wins the game if:
  - Verify$(w, L, \mu, \sigma_L(\mu))$=accept.
  - All of the public-keys in L are query outputs of $\mathcal{JO}$.
  - No public-key in L have been input to $\mathcal{CO}$.
  - $\sigma_L(\mu)$ is not a query output of $\mathcal{SO}$.
  - No signing key $\mathbf{A}_\pi$ was queried more than once to $\mathcal{SO}$.

  The advantage of the one-time unforgeability in the L2RS scheme is denoted by
  $$\mathbf{Advantage}_{\mathcal{A}}^{ot-unf}(\lambda) = \Pr[\mathcal{A} \text{ wins the game }]$$

  Definition 3 (One-Time Unforgeability). *The L2RS scheme is one-time unforgeable if for all PPT adversary $\mathcal{A}$, $\mathbf{Advantage}_{\mathcal{A}}^{ot-unf}(\lambda)$ is negligible.*

- Unconditional Anonymity. It should not be possible for an adversary $\mathcal{A}$ to tell the public-key of the signer with a probability larger than $1/w$, where w is the cardinality of the ring signature, even assuming that the adversary has unlimited computing resources.

  Unconditional anonymity for $\mathsf{L2RS}$ schemes is defined in the following game between a simulator $\mathcal{S}$ and an unbounded adversary $\mathcal{A}$ who has access to the oracle $\mathcal{JO}$.

  1. $\mathcal{S}$ generates and gives the list of public-keys L to $\mathcal{A}$.
  2. $\mathcal{A}$ may query $\mathcal{JO}$ according to any adaptive strategy.
  3. $\mathcal{A}$ gives $\mathcal{S}$, a group size w, a set L of w public-keys which are the outputs of $\mathcal{JO}$, a message μ. Parse the set L as $\{\mathbf{A}_1, \ldots, \mathbf{A}_w\}$. $\mathcal{S}$ randomly picks $\pi \in \{1, \ldots, w\}$ and computes $\sigma_\pi = \mathsf{Sign}(w, L, \mathbf{S}_\pi, \mu)$, where $\mathbf{S}_\pi$ is a corresponding private-key of $\mathbf{A}_\pi$. Then, σ_π is given to $\mathcal{A}$.
  4. $\mathcal{A}$ outputs a guess $\pi' \in \{1, \ldots, w\}$.

  The anonymity advantage of the L2RS scheme is denoted by

$$\mathbf{Advantage}_{\mathcal{A}}^{Anon}(\lambda) = \left| \Pr[\pi' = \pi] - \frac{1}{w} \right|$$

  Definition 4 (Unconditional Anonymity). *The $\mathsf{L2RS}$ scheme is unconditional anonymous if for any unbounded adversary $\mathcal{A}$, $\mathbf{Advantage}_{\mathcal{A}}^{Anon}(\lambda)$ is zero.*

- Linkability. It should be infeasible for a signer to generate two signatures such that they are determined **unlinked** using the $\mathsf{SigLink}$ algorithm. In this scenario, the adversary attempts to generate two signatures, using only one private-key $\mathbf{S}_\pi$. To describe this, we use the interaction between a simulator $\mathcal{S}$ and an adversary $\mathcal{A}$:

  1. The $\mathcal{A}$ queries the $\mathcal{JO}$ multiple times and $\mathcal{CO}$ only once to get the private-key $\mathbf{S}_\pi$, corresponding to the public-key $\mathbf{A}_\pi$.
  2. The $\mathcal{A}$ outputs two signatures $\sigma_L(\mu)$ and $\sigma'_{L'}(\mu')$ and two lists of public-keys L and L'.

  the $\mathcal{A}$ wins the game if:

  - The public-keys in L and L' are outputs of $\mathcal{JO}$.
  - By calling SigVer on input $\sigma_L(\mu)$ and $\sigma'_{L'}(\mu')$, it outputs **Accept** on both inputs.
  - Finally, it gets **Unlinked**, when calling $\mathsf{SigLink}$ on input $\sigma_L(\mu)$ and $\sigma'_{L'}(\mu')$.

  Thus the advantage of the linkability in the L2RS scheme is denoted by

$$\mathbf{Advantage}_{\mathcal{A}}^{Link}(\lambda) = \Pr[\mathcal{A} \text{ wins the game}].$$

  Definition 5 (Linkability). *The $\mathsf{L2RS}$ scheme is linkable if for all PPT adversary $\mathcal{A}$, $\mathbf{Advantage}_{\mathcal{A}}^{Link}$ is negligible.*

– Non-slanderability. It should be infeasible for an adversary to generate a valid signature that is **linked** with respect to a signature created by an honest user. This means that an adversary can frame an honest user for signing a valid signature so the adversary can produce another valid signature such that the SigLink algorithm outputs **Linked**. To describe this, we use the interaction between a simulator $\mathcal{S}$ and an adversary $\mathcal{A}$:

  1. The $\mathcal{S}$ generates and gives the list of public-keys L to $\mathcal{A}$.
  2. The $\mathcal{A}$ queries the $\mathcal{JO}$ and $\mathcal{CO}$ to obtain $\mathbf{A}_\pi$ and $\mathbf{S}_\pi$, respectively.
  3. $\mathcal{A}$ gives the generated parameters to $\mathcal{S}$.
  4. $\mathcal{S}$ uses the private-key $\mathbf{S}_\pi$ and calls the $\mathcal{SO}$ to output a valid signature $\sigma_L(\mu)$, which is given to $\mathcal{A}$.
  5. The $\mathcal{A}$ uses the remaining keys of the ring signature $(w-1)$ to create a second signature $\sigma'_L(\mu)$ by calling the $\mathcal{SO}$ algorithm.

  the $\mathcal{A}$ wins the game if:

  - The verification algorithm SigVer, on input $\sigma_L(\mu)$ and $\sigma'_L(\mu)$, outputs **Accept**.
  - The keys $\mathbf{A}_\pi$ and $\mathbf{S}_\pi$ were not used to generated the second signature $\sigma'_L(\mu)$.
  - When calling the SigLink on input $\sigma_L(\mu)$ and $\sigma'_L(\mu)$, it outputs **Linked**.

  Thus the advantage of the non-slanderability in the L2RS scheme is denoted by

$$\textbf{Advantage}_{\mathcal{A}}^{NS}(\lambda) = \Pr[\mathcal{A} \text{ wins the game}].$$

Definition 6 (Non-Slanderability). *The L2RS scheme is* non-slanderable *if for all PPT adversary $\mathcal{A}$,* $\textbf{Advantage}_{\mathcal{A}}^{NS}$ *is negligible.*

5 L2RS Scheme description

5.1 Setup

By receiving the security parameter λ, this L2RS.Setup algorithm randomly chooses $\mathbf{A}'_0 = (\mathbf{a}_{0,1}, \dots, \mathbf{a}_{0,m-1}) \leftarrow \mathcal{R}_q^{1\times(m-1)}$ and $\mathbf{H}'_0 = (\mathbf{h}_{0,1}, \dots, \mathbf{h}_{0,m-1}) \leftarrow \mathcal{R}_q^{1\times(m-1)}$. This outputs the public parameters (Pub-Params): $\mathbf{A}'_0$ and $\mathbf{H}'_0$.

Remark 2. To prevent malicious attack, L2RS.Setup incorporates a trapdoor in $\mathbf{A}'_0$ or $\mathbf{H}'_0$, in practice L2RS.Setup would generate $\mathbf{A}'_0$ and $\mathbf{H}'_0$ based on the cryptographic Hash function H_2 evaluated of two distinct and fixed constants.

Definition 7 (Function L2RS.Lift). *This function maps $\mathcal{R}_q^{1\times m}$ to $\mathcal{R}_{2q}^{1\times m}$ with respect to a public parameter $\boldsymbol{A}'_0 \in \mathcal{R}_q^{1\times(m-1)}$. Given $\boldsymbol{a}'_1 \in \mathcal{R}_q$, we let L2RS.Lift$(\boldsymbol{A}'_0, \boldsymbol{a}'_1) \triangleq (2\cdot \boldsymbol{A}'_0, -2\cdot \boldsymbol{a}'_1 + q \bmod 2q) \in \mathcal{R}_{2q}^{1\times m}$.*

5.2 Key Generation - KeyGen

This algorithm receives the public parameters Pub-Params: $\mathbf{A}'_0$ and $\mathbf{H}'_0$.

1. To generate a key pair in $\mathcal{R}_q$, we:
   - Pick $(\mathbf{s}_{0,1}, \ldots, \mathbf{s}_{0,m-1})$, where every component is chosen uniformly and independently with coefficients in $(-2^\gamma, 2^\gamma)$.
   - Define $\mathbf{S}_0^T = (\mathbf{s}_{0,1}, \ldots, \mathbf{s}_{0,m-1}) \in \mathcal{R}_q^{1\times(m-1)}$, and let $\mathbf{S}^T = (\mathbf{S}_0^T, 1) \in \mathcal{R}_q^{1\times m}$.
   - Compute $\mathbf{a}'_1 = \mathbf{A}'_0 \cdot \mathbf{S}_0 \bmod q \in \mathcal{R}_q$.
   - Return $(\mathbf{A}'_0, \mathbf{a}'_1) \in \mathcal{R}_q^{1\times m}$, $(\mathbf{S}_0^T, 1) \in \mathcal{R}_{2q}^{1\times m}$.
2. The L2RS.Lift function is used to compute and return: $\mathbf{A} = (\mathbf{A}_0, \mathbf{a}_1) =$ L2RS.Lift$(\mathbf{A}'_0, \mathbf{a}'_1) = (2 \cdot \mathbf{A}'_0, -2 \cdot \mathbf{a}'_1 + q \bmod 2q) \in \mathcal{R}_{2q}^{1\times m}$.
3. In the private-key $\mathbf{S}^T = (\mathbf{S}_0^T, 1) \in \mathcal{R}_q^{1\times m}$, we consider $\mathbf{S}_0$ an element in $\mathcal{R}_{2q}$, so that this returns the private-key $\mathbf{S} \in \mathcal{R}_{2q}^{m\times 1}$.

Note that $\mathbf{A} \cdot \mathbf{S} = q \in \mathcal{R}_{2q}$. The list of the users' public-keys is defined as $L = \{\mathbf{A}_1, \ldots, \mathbf{A}_w\}$, where w is the number of users in the ring signature scheme. This KeyGen algorithm is described in the following **Algorithm 1**:

Algorithm 1 L2RS Algorithm - Key pair generation $(\mathbf{A}, \mathbf{S})$

Input: The public parameters Pub-Params: $\mathbf{A}'_0$ and $\mathbf{H}'_0$.
Output: $(\mathbf{A}, \mathbf{S})$, where $\mathbf{A}$ is the public-key and $\mathbf{S}$ is the private-key.

```
1: procedure L2RS.KeyGen(Pub-Params)
2:     Let $\mathbf{S}_0^T = (\mathbf{s}_{0,1}, \ldots, \mathbf{s}_{0,m-1}) \in \mathcal{R}_q^{1\times(m-1)}$, where $\mathbf{s}_{0,i} \leftarrow (-2^\gamma, 2^\gamma)^n$, for $1 \le i \le m-1$
3:     Let $\mathbf{S}^T = (\mathbf{S}_0^T, 1) \in \mathcal{R}_q^{1\times m}$.
4:     Compute $\mathbf{a}'_1 = \mathbf{A}'_0 \cdot \mathbf{S}_0 \bmod q \in \mathcal{R}_q$.
5:     Call function L2RS.Lift$(\mathbf{A}'_0, \mathbf{a}'_1)$, and it returns $\mathbf{A} = (\mathbf{A}_0, \mathbf{a}_1) = (2 \cdot \mathbf{A}'_0, -2 \cdot \mathbf{a}'_1 + q \bmod 2q) \in \mathcal{R}_{2q}^{1\times m}$
6:     Remark: $\mathbf{A} \cdot \mathbf{S} = q \in \mathcal{R}_{2q}$, where $\mathbf{S} \in \mathcal{R}_{2q}^{m\times 1}$.
7:     return $(\mathbf{A}, \mathbf{S})$.
```

5.3 Signature Generation - SigGen

The SigGen algorithm inputs the user's private-key $\mathbf{S}_\pi$, the message μ, the list of user's public-keys L, and will output the signature $\sigma_L(\mu)$. We call π the index in $\{1, \ldots, w\}$ of the user or signatory who wants to sign a message μ. For a message $\mu \in \{0,1\}^*$, the fixed list of public-keys L and the private-key $\mathbf{S}_\pi$ which corresponds to $\mathbf{A}_\pi$ with $1 \le \pi \le w$; the following computations are performed:

1. We define the linkability tag as $\mathbf{H} = \big(\mathbf{H}_0, \mathbf{h}_1\big)$, where $\mathbf{H}_0$ is a fixed public parameter for all users: $\mathbf{H}_0 = 2 \cdot \mathbf{H}'_0 \in \mathcal{R}_{2q}^{1\times(m-1)}$, and $\mathbf{h}_1 = -\mathbf{H}_0 \cdot \mathbf{S}_{\pi,0} + q \in \mathcal{R}_{2q}$, where $\mathbf{S}_\pi^T = \big(\mathbf{S}_{\pi,0}^T, 1\big) \in \mathcal{R}_{2q}^{1\times m}$, such that $\mathbf{H} \cdot \mathbf{S}_\pi = q \in \mathcal{R}_{2q}$.

2. By choosing a random vector $\mathbf{u}_\pi = (u_1, \ldots, u_m)^T$, where $u_i \leftarrow D_\sigma^n$, for $1 \leq i \leq m$, we calculate $\mathbf{c}_{\pi+1} = H_1\Big(L, \mathbf{H}, \mu, \mathbf{A}_\pi \cdot \mathbf{u}_\pi, \mathbf{H} \cdot \mathbf{u}_\pi\Big)$.
3. We choose random vector $\mathbf{t}_i = (t_{i,1}, \ldots, t_{i,m})^T$, where $t_{i,j} \leftarrow D_\sigma^n$, for $1 \leq j \leq m$, then for $(i = \pi+1, \ldots, w, 1, 2, \ldots, \pi-1)$, we compute $\mathbf{c}_{i+1} = H_1\Big(L, \mathbf{H}, \mu, \mathbf{A}_i \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i, \mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i\Big)$.
4. Select a random bit $b \in \{0, 1\}$ and finally compute $\mathbf{t}_\pi = \mathbf{u} + \mathbf{S}_\pi \cdot \mathbf{c}_\pi \cdot (-1)^b$ using rejection sampling (**Definition 2**).
5. Output the signature $\sigma_L(\mu) = \Big(\mathbf{c}_1, \mathbf{t}_1, \ldots, \mathbf{t}_w, \mathbf{H}\Big)$.

A formal description of this algorithm is shown in **Algorithm 2**.

Algorithm 2 L2RS Algorithm - Signature Generation $\sigma_L(\mu)$

```
Input: $\mathbf{S}_\pi, \mu, L$, where $L = \{\mathbf{A}_1, \ldots, \mathbf{A}_w\}$.
Output: $\sigma_L(\mu) = \Big(\mathbf{c}_1, \mathbf{t}_1, \ldots, \mathbf{t}_w, \mathbf{H}\Big)$
 1: procedure L2RS.SigGen($\mathbf{S}_\pi, \mu, L$)
 2:     Set $\mathbf{H} = (\mathbf{H}_0, \mathbf{h}_1)$, where $\mathbf{H}_0 = 2 \cdot \mathbf{H}'_0$ and $\mathbf{h}_1 = -\mathbf{H}_0 \cdot \mathbf{S}_{\pi,0} + q \bmod 2q$
 3:     Let $\mathbf{u} = (u_1, \ldots, u_m)^T$, where $u_i \leftarrow D_\sigma^n$, for $1 \leq i \leq m$.
 4:     Compute $\mathbf{c}_{\pi+1} = H_1\Big(L, \mathbf{H}, \mu, \mathbf{A}_\pi \cdot \mathbf{u}, \mathbf{H} \cdot \mathbf{u}\Big)$.
 5:     for $(i = \pi+1, \pi+2, \ldots, w, 1, 2, \ldots, \pi-1)$ do
 6:         Let $\mathbf{t}_i = (t_{i,1}, \ldots, t_{i,m})^T$, where $t_{i,j} \leftarrow D_\sigma^n$, for $1 \leq j \leq m$.
 7:         Compute $\mathbf{c}_{i+1} = H_1\Big(L, \mathbf{H}, \mu, \mathbf{A}_i \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i, \mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i\Big)$.
 8:     Choose $b \leftarrow \{0, 1\}$.
 9:     Let $\mathbf{t}_\pi \leftarrow \mathbf{u} + \mathbf{S}_\pi \cdot \mathbf{c}_\pi \cdot (-1)^b$.
10:     Continue with probability $\dfrac{1}{\left(M \exp\left(-\dfrac{\|\mathbf{S}_\pi \cdot \mathbf{c}_\pi\|^2}{2\sigma^2}\right) \cosh\left(\dfrac{\langle \mathbf{t}_\pi, \mathbf{S}_\pi \cdot \mathbf{c}_\pi \rangle}{\sigma^2}\right)\right)}$
        otherwise Restart.
11:     return $\sigma_L(\mu) = \Big(\mathbf{c}_1, \mathbf{t}_1, \ldots, \mathbf{t}_w, \mathbf{H}\Big)$.
```

5.4 Signature Verification - SigVer

The SigVer algorithm receives the signature $\sigma_L(\mu)$ along with the message μ and the fixed list L, and will output a decisional verification answer: whether accept or reject the signature (see **Algorithm 3**). The signature $\sigma_L(\mu)$ can be publicly validated by computing $\mathbf{H} = (\mathbf{H}_0, \mathbf{h}_1)$ in $\mathbf{c}_{i+1}$ for $(i = 1, \ldots, w)$, and it is verified and only accepted under the following four conditions: $\|\mathbf{t}_i\|_2 \leq B_2$ for $1 \leq i \leq w$, $\|\mathbf{t}_i\|_\infty < q/4$ for $1 \leq i \leq w$, $\mathbf{c}_1 = H_1\Big(L, \mathbf{H}, \mu, \mathbf{A}_w \cdot \mathbf{t}_w + q \cdot \mathbf{c}_w, \mathbf{H} \cdot \mathbf{t}_w + q \cdot \mathbf{c}_w\Big)$ and $\mathbf{H}_0 = 2 \cdot \mathbf{H}'_0$.

Algorithm 3 L2RS Algorithm - Signature Verification

```
Input: $\sigma_L(\mu) = \Big(\mathbf{c}_1, \mathbf{t}_1, \ldots, \mathbf{t}_w, \mathbf{H}\Big)$, $L$, $\mu$
Output: Accept or Reject
1: procedure L2RS.SigVer($\sigma_L(\mu)$)
2:     if $\mathbf{H} = (\mathbf{H}_0, \mathbf{h}_1)$ and $\mathbf{H}_0 = 2 \cdot \mathbf{H}_0'$ then Continue
3:         for $(i = 1, \ldots, w)$ do
4:             if $\mathbf{c}_{i+1} = H_1\Big(L, \mathbf{H}, \mu, \mathbf{A}_i \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i, \mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i\Big)$ then Continue
5:             else if $\|\mathbf{t}_i\|_2 \leq B_2$ then Continue
6:             else if $\|\mathbf{t}_i\|_\infty < q/4$ then Continue
7:     else if $\mathbf{c}_1 = H_1\Big(L, \mathbf{H}, \mu, \mathbf{A}_w \cdot \mathbf{t}_w + q \cdot \mathbf{c}_w, \mathbf{H} \cdot \mathbf{t}_w + q \cdot \mathbf{c}_w\Big)$ then Accept
8:     else Reject
9:     return Accept or Reject
```

Theorem 1. *Let $q > 2\eta\sqrt{m}\sigma$ and $\sigma_L(\mu) = \Big(\boldsymbol{c}_1, \boldsymbol{t}_1, \ldots, \boldsymbol{t}_w, \boldsymbol{H}\Big)$ be generated based on 2 such that $\|\boldsymbol{t}_i\|_\infty \leq q/4$, for $1 \leq i \leq m$. Then the output of **Algorithm 3** on input $\sigma_L(\mu)$ is **Accept** with probability $1 - 2^{-\lambda}$.*

Note that η is chosen such that $\|\mathbf{t}_i\| \leq q/2$ is verified with probability $1 - 2^{-\lambda}$. The proof of this theorem will be given in the full version.

5.5 Signature Linkability - SigLink

The SigLink algorithm, illustrated in **Algorithm 4**, takes two signatures as its input $\sigma_L(\mu_1)$ and $\sigma_L(\mu_2)$ and it outputs **Linked** if these signatures were generated by same signatory, it will output **Unlinked** otherwise. For a fixed list of public-keys L and given two signatures: $\sigma_L(\mu_1)$ and $\sigma_L(\mu_2)$, with the list L which can be described as: $\sigma_L(\mu_1) = \Big(\mathbf{c}_{1,\mu_1}, \mathbf{t}_{1,\mu_1}, \ldots, \mathbf{t}_{w,\mu_1}, \mathbf{H}_{\mu_1}\Big)$ and $\sigma_L(\mu_2) = \Big(\mathbf{c}_{1,\mu_2}, \mathbf{t}_{1,\mu_2}, \ldots, \mathbf{t}_{w,\mu_2}, \mathbf{H}_{\mu_2}\Big)$.

These two signatures must be successfully accepted by the SigVer algorithm, then one can verify that the linkability property can be achieved if the linkability tags ($\mathbf{H}_{\mu_1}$ and $\mathbf{H}_{\mu_2}$) of the above signatures $\sigma_L(\mu_1)$ and $\sigma_L(\mu_2)$ are equal.
The correctness proofs of L2RS.SigGen and L2RS.SigLink are given in Appendix A.

6 Security Analysis

Theorem 2 (One-Time Unforgeability). *Suppose $\sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$ is negligible in n and $\frac{1}{|\mathcal{S}_{n,\kappa}|}$ is negligible and $y = h$ is polynomial in n, where h denotes the number of queries to the random oracle H_1. If there is a PPT algorithm against one-time unforgeability of L2RS with non-negligible probability δ, then there exist a PPT algorithm that can extract a solution to the $\mathcal{R}$-$\mathbf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem (for*

Algorithm 4 L2RS Algorithm - Signature Linkability

```
Input: $\sigma_L(\mu_1)$ and $\sigma_L(\mu_2)$
Output: Linked or Unlinked
1: procedure L2RS.SigLink($\sigma_L(\mu_1), \sigma_L(\mu_2)$)
2:     if $\Big(\mathsf{L2RS.SigVer}(\sigma_L(\mu_1)) = \text{Accept}$ and $\mathsf{L2RS.SigVer}(\sigma_L(\mu_2)) = \text{Accept}\Big)$ then
   Continue [
3:     else if $\mathbf{H}_{\mu_1} = \mathbf{H}_{\mu_2}$ then Linked
4:     else Unlinked ]
5:     return Linked or Unlinked
```

$\beta = 2B_2$) with non-negligible probability $\left(\delta - \frac{1}{|\mathcal{S}_{n,\kappa}|}\right) \cdot \left(\frac{\delta - \frac{1}{|\mathcal{S}_{n,\kappa}|}}{y} - \frac{1}{|\mathcal{S}_{n,\kappa}|}\right) - \sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$.

Proof. The proof is given in Appendix B.

Theorem 3 (Anonymity). *Suppose $\sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$ is negligible in n with an attack against the unconditional anonymity that makes h queries to the random oracle H_1, where h, w are polynomial in n, then the L2RS scheme is unconditionally secure as defined in **Definition 4**.*

Proof. The proof is given in Appendix C.

Theorem 4 (Linkability). *The L2RS scheme is linkable in the random oracle model if the $\mathcal{R}$-$\mathbf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem is hard.*

Proof. The proof is given in Appendix D.

Theorem 5 (Non-Slanderability). *For any linkable ring signature, if it satisfies unforgeability and unlinkability, then it satisfies non-slanderability.*

Proof. The proof is given in Appendix E.

Corollary 1 (Non-Slanderability). *The L2RS scheme is non-slanderable under the assumptions of Theorem 2 and Theorem 4.*

7 Lattice RingCT v1.0 Protocol

This protocol is an extension of the original Ring CT protocol described in [48], and is constructed based on the L2RS scheme. Its algorithms are defined as follows:

- Setup: this PPT algorithm uses $\mathsf{L2RS.Setup}$ where it takes the security parameter λ and outputs the public parameters.
- KeyGen: this PPT algorithm uses $\mathsf{L2RS.KeyGen}$, it receives the public parameters and produces a pair of keys, the public-key and the private-key.

- Mint: a PPT algorithm that is used to generate new coins. This algorithm receives the public-key $\mathbf{A}$ and the amount a, and it outputs a coin $\mathbf{cn}$ along with its associated coin-key $\mathbf{ck}$. An account is formed using the public-key $\mathbf{A}$ and the coin $\mathbf{cn}$. Likewise, the private-key $\mathbf{S}$ along with the coin-key $\mathbf{ck}$ is used for the spending authorization.
- Spend: a PPT algorithm, which is used to generate the linkable ring signature, receives the fixed list of users' public-keys in the ring signature L, the Output Wallet OW and some transaction string $m \in \{0,1\}^*$, these three parameter constitute the transaction TX. This algorithm outputs the signature $\sigma_L(\mu)$ along with the TX.
- Verify: a deterministic PPT algorithm that takes as input the signature $\sigma_L(\mu)$ and the TX, it outputs either: **Accept** or **Reject**.

7.1 Scheme construction

Our Lattice RingCT scheme requires homomorphic commitment (Com) as an additional primitive. It is a cryptographic technique used to provide confidential transactions, in particular cryptocurrencies [6]. This primitive allows one party to commit to a chosen value while keeping it secret to other parties, then this committed value can be revealed later. This model is restricted to have an Input Wallet (IW) that will be spent into an Output Wallet (OW) only. We use the structure of the $\mathsf{L2RS.KeyGen}$ scheme **Algorithm 1**, where the public parameter $\mathbf{A}'_0 \in \mathcal{R}_q^{1\times(m-1)}$ is used to commit to a scalar message $\mathrm{m} \in \mathsf{Dom}_\mathrm{m} \subseteq \mathcal{R}_q$ with $\mathsf{Dom}_\mathrm{m} = [0, \dots, 2^{\ell-1}] \subseteq \mathbb{Z}$. This property is defined as $\mathsf{Com}_{\mathbf{A}'_0}(\mathrm{m}, \mathbf{S}_0) = \mathbf{A}'_0 \cdot \mathbf{S}_0 + \mathrm{m}$, where $\mathbf{S}_0 \in \mathsf{Dom}_{\mathbf{S}_0} \subseteq \mathcal{R}_q^{(m-1)\times 1}$ is the randomness. The properties of the homomorphic operations are also defined as:

$$\begin{aligned}\mathsf{Com}_{\mathbf{A}'_0}(\mathrm{m}_1, \mathbf{S}_0) \;\oplus\; \mathsf{Com}_{\mathbf{A}'_0}(\mathrm{m}_2, \mathbf{S}'_0) &\triangleq \mathsf{Com}_{\mathbf{A}'_0}(\mathrm{m}_1, \mathbf{S}_0) + \mathsf{Com}_{\mathbf{A}'_0}(\mathrm{m}_2, \mathbf{S}'_0) \bmod q \\ &= \mathsf{Com}_{\mathbf{A}'_0}(\mathrm{m}_1 + \mathrm{m}_2, \mathbf{S}_0 + \mathbf{S}'_0) \bmod q, \qquad (1)\end{aligned}$$

$$\begin{aligned}\mathsf{Com}_{\mathbf{A}'_0}(\mathrm{m}_1, \mathbf{S}_0) \;\ominus\; \mathsf{Com}_{\mathbf{A}'_0}(\mathrm{m}_2, \mathbf{S}'_0) &\triangleq \mathsf{Com}_{\mathbf{A}'_0}(\mathrm{m}_1, \mathbf{S}_0) - \mathsf{Com}_{\mathbf{A}'_0}(\mathrm{m}_2, \mathbf{S}'_0) \bmod q \\ &= \mathsf{Com}_{\mathbf{A}'_0}(\mathrm{m}_1 - \mathrm{m}_2, \mathbf{S}_0 - \mathbf{S}'_0) \bmod q, \qquad (2)\end{aligned}$$

where $\mathrm{m}_1, \mathrm{m}_2 \in \mathcal{R}_q$; and $\mathbf{S}_0, \mathbf{S}'_0 \in \mathcal{R}_q^{(m-1)\times 1}$. The integers $\mathrm{m}_1, \mathrm{m}_2 \in \mathbb{Z}$ are encoded in binary as coefficient vectors $\mathbf{m}_1 = (\mathrm{m}_{1,0}, \dots, \mathrm{m}_{1,\ell-1}, 0, \dots, 0) \in \{0,1\}^n$ and $\mathbf{m}_2 = (\mathrm{m}_{2,0}, \dots, \mathrm{m}_{2,\ell-1}, 0, \dots, 0) \in \{0,1\}^n$ where $\mathbf{m}_j = \sum_{i=0}^{\ell-1}(\mathrm{m}_{j,i} \cdot 2^i)$, with $\mathrm{m}_{j,i} \in \{0,1\}$ and $j \in \{0,1\}$, and $\mathbf{m} = \mathbf{m}_1 - \mathbf{m}_2 = (\mathrm{m}_{1,0} - \mathrm{m}_{2,0}, \dots, \mathrm{m}_{1,\ell-1} - \mathrm{m}_{2,\ell-1}, 0, \dots, 0) \in \{-1,0,1\}^n$. The difference between these vectors is zero $\in \mathcal{R}_q$ if $\mathbf{m}_1 = \mathbf{m}_2$, non-zero otherwise. Hence the commitment is done to bits.

The construction of the Lattice RingCT v1.0 algorithm has the following steps:

1. ($\mathsf{Pub\text{-}Params}$) $\leftarrow$ $\mathsf{Setup}(\lambda)$: On input security parameter λ, this algorithm calls $\mathsf{L2RS.Setup}$ and outputs the public parameters, $\mathbf{A}'_0$ and $\mathbf{H}'_0$.

2. $(\mathbf{A}_{in}, \mathbf{S}_{in}) \leftarrow \mathsf{KeyGen}(\mathsf{Pub\text{-}Params})$: Given the public parameters, we call $\mathsf{L2RS.KeyGen}$ to generate the pair of keys. Thus it outputs the IW pair of keys $(\mathbf{A}_{in}, \mathbf{S}_{in})$, where $\mathbf{A}_{in} \in \mathcal{R}_{2q}^{1\times m}$ is the public-key (or one-time address) and $\mathbf{S}_{in} = (\mathbf{S}_0, 1) \in \mathsf{Dom}_{\mathbf{S}_0} \subseteq \mathcal{R}_{2q}^{m\times 1}$ is the private-key. The commitment of the KeyGen is defined as $\mathbf{a}'_{1(in)} = \mathbf{A}'_0 \cdot \mathbf{S}_{0(in)} \bmod q \in \mathcal{R}_q = \mathsf{Com}_{\mathbf{A}'_0}\big(0, \mathbf{S}_{0(in)}\big)$.
3. $(\mathbf{cn}', \mathbf{ck}') \leftarrow \mathsf{Mint}(\mathbf{A}_{in}, a_{in})$: It receives a valid one-time address $\mathbf{A}_{in}$ as well as an input amount $a_{in} \in \mathbb{B}_w^n$, where $\mathbb{B} = \{0, 1\}$. Then, to create a coin $\mathbf{cn}'_{in}$, this algorithm chooses a coin-key $\mathbf{ck}'_{in} \in \mathsf{Dom}_{\mathbf{S}_0}$, where every component is chosen uniformly and independently with coefficients in $(-2^\gamma, 2^\gamma)$. Then, the commitment is computed as $\mathbf{cn}'_{in} = \mathsf{Com}_{\mathbf{A}'_0}(a_{in}, \mathbf{ck}'_{in})$ and it returns $(\mathbf{cn}'_{in}, \mathbf{ck}'_{in})$. An account constitutes $\big(\mathbf{a}'_{1(in)}, \mathbf{cn}'_{in}\big) \in \mathcal{R}_q \times \mathcal{R}_q$.
4. $(TX, \sigma_{L'}(\mu)) \leftarrow \mathsf{Spend}(\mu, OW)$: This algorithm follows the steps:
   (a) A new coin for the OW is created by the spender. It generates $\mathbf{ck}'_{out} \in \mathsf{Dom}_{\mathbf{S}_0}$, where every component is chosen uniformly and independently with coefficients in $(-2^\gamma, 2^\gamma)$, then it is computed $\mathbf{cn}'_{out} = \mathsf{Com}_{\mathbf{A}'_0}\big(a_{out}, \mathbf{ck}'_{out}\big)$. The new OW is set as $\big(\mathbf{a}'_{1(out)}, \mathbf{cn}'_{out}\big) \in \mathcal{R}_q \times \mathcal{R}_q$.
   (b) A transaction string $\mu \in \{0, 1\}^*$ defines the ring signature message.
   (c) The list of the ring signature is constructed as $L' = \big\{\big(\widehat{\mathbf{a}}'_{1(in),i}, \mathbf{cn}'_{in,i}\big)\big\} \in \mathcal{R}_q \times \mathcal{R}_q$ for $1 \le i \le w$ with w being the size of the ring signature, its components are produced as:
      – $\widehat{\mathbf{a}}'_{1(in),i} = \mathbf{a}'_{1(in),i} + \mathbf{cn}'_{in,i} - \mathbf{cn}'_{out,i} = \mathsf{Com}_{\mathbf{A}'_0}\big(a_{in,i} - a_{out}, \mathbf{S}_{0(in),i} + \mathbf{ck}'_{in,i} - \mathbf{ck}'_{out}\big)$.
      – $\mathbf{cn}'_{in,i} = \mathsf{Com}_{\mathbf{A}'_0}\big(a_{in,i}, \mathbf{ck}'_{in,i}\big)$.
   (d) We call the L2RS.Lift() function (**Definition 7**) to lift L' from $\mathcal{R}_q^{1\times m}$ to $\mathcal{R}_{2q}^{1\times m}$:
      – $L' = \big\{\big(\mathsf{L2RS.Lift}\big(\mathbf{A}'_0, \widehat{\mathbf{a}}'_{1(in),i}\big), \mathsf{L2RS.Lift}\big(\mathbf{A}'_0, \mathbf{cn}'_{in,i}\big)\big)\big\} = \big\{\big(\widehat{\mathbf{A}}_{1(in),i}, \mathbf{CN}_{in,i}\big)\big\} \in \mathcal{R}_{2q}^{1\times m} \times \mathcal{R}_{2q}^{1\times m}$, for $1 \le i \le w$.
      – The private-key of π is in the form of $\mathbf{S}''_{in,\pi} = \big(\mathbf{S}_{in,\pi}, \mathbf{CK}_{in,\pi}\big) \in \mathcal{R}_{2q}^{m\times 1} \times \mathcal{R}_{2q}^{m\times 1}$, where:
         • $\mathbf{S}_{in,\pi} = \big(\mathbf{S}_{0(in,\pi)} + \mathbf{ck}'_{in,\pi} - \mathbf{ck}'_{out,\pi}\big) \in \mathcal{R}_{2q}^{m\times 1}$.
         • $\mathbf{CK}_{in,\pi} = \big(\mathbf{ck}'_{in,\pi}, 1\big) \in \mathcal{R}_{2q}^{m\times 1}$.
   (e) By calling the $\mathsf{L2RS\text{-}DoubleSignGen}\big(\mathbf{S}''_{in,\pi}, L', \mu\big)$, **Algorithm 5**, we create the ring signature $\sigma_{L'}(\mu) = \left(\mathbf{c}_1, \begin{pmatrix} \mathbf{t}_1, \ldots, \mathbf{t}_w \\ \mathbf{t}'_1, \ldots, \mathbf{t}'_w \end{pmatrix}, \mathbf{H}\right)$.
   (f) We set the transaction $TX = (\mu, L', OW)$.
   (g) This algorithm ultimately outputs TX and $\sigma_{L'}(\mu)$.
5. (**Accept**/**Reject**) $\leftarrow \mathsf{Verify}\big(TX, \sigma_{L'}(\mu)\big)$: This algorithm calls $\mathsf{L2RS\text{-}DoubleSigVer}\big(\sigma_{L'}(\mu)\big)$, using **Algorithm 6** and will return either **Accept** or **Reject**.

This construction as stated supports one-IW to one-OW and thus in this case the range proof [6] is not needed. In the full version of this work, we will provide more details for the correctness and the security analysis of the hiding and binding property; this version will also extend the Lattice RingCT v1.0 scheme to support Multiple-Inputs to Multiple-Outputs (MIMO) wallets, and therefore a range proof will be given.

Algorithm 5 L2RS-DoubleSignGen Algorithm - Signature Generation $\sigma_{L'}(\mu)$

```
Input: $\mathbf{S}''_{in,\pi}, \mu, L'$, where $\mathbf{S}''_{in,\pi} = \big(\mathbf{S}_{in,\pi}, \mathbf{CK}_{in,\pi}\big)$ and $L' = \big\{\big(\widehat{\mathbf{A}}_{1(in),i}, \mathbf{CN}_{in,i}\big)\big\}_{i=1}^{w}$
Output: $\sigma_{L'}(\mu) = \left(\mathbf{c}_1, \begin{pmatrix}\mathbf{t}_1,\dots,\mathbf{t}_w \\ \mathbf{t}'_1,\dots,\mathbf{t}'_w\end{pmatrix}, \mathbf{H}\right)$
1: procedure L2RS.DoubleSignGen($\mathbf{S}''_{in,\pi}, \mu, L'$)
2:     Set $\mathbf{H} = (\mathbf{H}_0, \mathbf{h}_1)$, where $\mathbf{H}_0 = 2 \cdot \mathbf{H}'_0$ and $\mathbf{h}_1 = -\mathbf{H}_0 \cdot \mathbf{S}_{\pi,0} + q \bmod 2q$
3:     for $(1 \le i \le m)$ do
4:         Let $\mathbf{u} = (u_1,\dots,u_m)^T$, where $u_i \leftarrow D^n_\sigma$.
5:         Let $\mathbf{u}' = (u'_1,\dots,u'_m)^T$, where $u'_i \leftarrow D^n_\sigma$.
6:     Compute $\mathbf{c}_{\pi+1} = H_1\Big(L, \mathbf{H}, \mu, \widehat{\mathbf{A}}_{1(in),\pi} \cdot \mathbf{u}, \mathbf{CN}_{in,\pi} \cdot \mathbf{u}', \mathbf{H} \cdot \mathbf{u}\Big)$.
7:     for $(i = \pi+1, \pi+2, \dots, w, 1, 2, \dots, \pi-1)$ do
8:         for $(1 \le j \le m)$ do
9:             Let $\mathbf{t}_i = (t_{i,1},\dots,t_{i,m})^T$, where $t_{i,j} \leftarrow D^n_\sigma$.
10:            Let $\mathbf{t}'_i = (t'_{i,1},\dots,t'_{i,m})^T$, where $t'_{i,j} \leftarrow D^n_\sigma$.
11:        Compute $\mathbf{c}_{i+1} = H_1\Big(L, \mathbf{H}, \mu, \widehat{\mathbf{A}}_{1(in),i} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i, \mathbf{CN}_{in,i} \cdot \mathbf{t}'_i + q \cdot \mathbf{c}_i, \mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i\Big)$.
12:    Choose $b \leftarrow \{0,1\}$ and $b' \leftarrow \{0,1\}$.
13:    Let $\mathbf{t}_\pi \leftarrow \mathbf{u} + \mathbf{S}_{in,\pi} \cdot \mathbf{c}_\pi \cdot (-1)^b$.
14:    Continue with prob. $\dfrac{1}{M \exp\left(-\dfrac{\|\mathbf{S}_{in,\pi} \cdot \mathbf{c}_\pi\|^2}{2\sigma^2}\right) \cosh\left(\dfrac{\langle \mathbf{t}_\pi, \mathbf{S}_{in,\pi} \cdot \mathbf{c}_\pi \rangle}{\sigma^2}\right)}$ otherwise Restart.
15:    Let $\mathbf{t}'_\pi \leftarrow \mathbf{u}' + \mathbf{CK}_{in,\pi} \cdot \mathbf{c}_\pi \cdot (-1)^{b'}$.
16:    Continue with prob. $\dfrac{1}{M \exp\left(-\dfrac{\|\mathbf{CK}_{in,\pi} \cdot \mathbf{c}_\pi\|^2}{2\sigma^2}\right) \cosh\left(\dfrac{\langle \mathbf{t}'_\pi, \mathbf{CK}_{in,\pi} \cdot \mathbf{c}_\pi \rangle}{\sigma^2}\right)}$ otherwise Restart.
17:    return $\sigma_{L'}(\mu) = \left(\mathbf{c}_1, \begin{pmatrix}\mathbf{t}_1,\dots,\mathbf{t}_w \\ \mathbf{t}'_1,\dots,\mathbf{t}'_w\end{pmatrix}, \mathbf{H}\right)$.
```

8 Performance Analysis

We proposed a set of parameters (**Table 1**) to implement the L2RS and Lattice RingCT v1.0 schemes. They are secure against direct lattice attacks in terms of the BKZ algorithm Hermite factor δ, using the value of $\delta = 1.007$, based on the BKZ 2.0 complexity estimates with pruning enumeration-based Shortest Vector Problem (SVP) [49], this might give $90 - 100$ bits of security. We use the conditions stated in the L2RS.SigVer algorithm and in the security analysis (Section 6) with $\gamma = 0$ and $\alpha = 0.5$. This analysis turns out signatures sizes of

Algorithm 6 L2RS-DoubleSigVer Algorithm - Signature Verification

```
Input: $TX = (\mu, L', OW), \sigma_{L'}(\mu) = \left(\mathbf{c}_1, \begin{pmatrix} \mathbf{t}_1, \ldots, \mathbf{t}_w \\ \mathbf{t}'_1, \ldots, \mathbf{t}'_w \end{pmatrix}, \mathbf{H}\right)$, where $L' = \{(\widehat{\mathbf{A}}_{1(in),i}, \mathbf{CN}_{in,i})\}_{i=1}^{w}$
Output: Accept or Reject
1: procedure L2RS.DoubleSigVer($\sigma_{L'}(\mu)$)
2:     if $\mathbf{H} = (\mathbf{H}_0, \mathbf{h}_1)$ and $\mathbf{H}_0 = 2 \cdot \mathbf{H}'_0$ then Continue
3:         for $(i = 1, \ldots, w)$ do
4:             if $\mathbf{c}_{i+1} = H_1\left(L, \mathbf{H}, \mu, \widehat{\mathbf{A}}_{1(in),i} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i, \mathbf{CN}_{in,i} \cdot \mathbf{t}'_i + q \cdot \mathbf{c}_i, \mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i\right)$ then Continue
5:             else if $\|\mathbf{t}_i\|_2 \le B_2$ and $\|\mathbf{t}'_i\|_2 \le B_2$ then Continue
6:             else if $\|\mathbf{t}_i\|_\infty < q/4$ and $\|\mathbf{t}'_i\|_\infty < q/4$ then Continue
7:     else if $\mathbf{c}_1 = H_1\left(L, \mathbf{H}, \mu, \widehat{\mathbf{A}}_{1(in),i} \cdot \mathbf{t}_w + q \cdot \mathbf{c}_w, \mathbf{CN}_{in,w} \cdot \mathbf{t}'_w + q \cdot \mathbf{c}_w, \mathbf{H} \cdot \mathbf{t}_w + q \cdot \mathbf{c}_w\right)$ then Accept
8:     else Reject
9:     return Accept or Reject
```

53 KB and 60 KB for L2RS and Lattice RingCT v1.0, respectively, when the number of signers in a ring signature (w) is 1. The size of the pair of keys in L2RS is 0.592 KB (private-key) and 1.252 KB (public-key), whereas this size in Lattice RingCT v1.0 is 1.184 KB (private-key) and 1.12 KB (public-key).

Table 1. Selected parameters for L2RS and Lattice RingCT v1.0

Name of the Scheme	**L2RS**	**Lattice-RingCT v1.0**
Security parameter (λ)	100	100
n	128	128
κ	32	32
m	73	73
η	2.1	2.1
$\|\mathbf{Sc}\|$	546.8	546.8
σ	273.4	273.4
$\log(\beta)$	13.429	13.429
$\log(q)$	35	35
Signature size ($w = 1$)	51 KB	60 KB
Signature size ($w = 5$)	89 KB	136 KB
Signature size ($w = 10$)	136 KB	231 KB
Signature size ($w = 15$)	183 KB	325 KB
Private-key size	0.592 KB	1.184 KB
Public-key size	1.152 KB	1.12 KB

Acknowledgement. The work of Ron Steinfeld and Amin Sakzad was supported in part by ARC Discovery Project grant DP150100285. This work was also supported by the Monash-HKPU-Collinstar Blockchain Research Lab.

References

1. R. Rivest, A. Shamir, and Y. Tauman, “How to leak a secret,” in *ASIACRYPT*, pp. 552–565, Springer, 2001.
2. D. Chaum and E. Van Heyst, “Group signatures,” in *EUROCRYPT*, pp. 257–265, Springer, 1991.
3. J. K. Liu, V. K. Wei, and D. S. Wong, “Linkable spontaneous anonymous group signature for ad hoc groups,” in *ACISP*, pp. 325–335, Springer, 2004.
4. J. K. Liu, M. H. Au, X. Huang, W. Susilo, J. Zhou, and Y. Yu, “New Insight to Preserve Online Survey Accuracy and Privacy in Big Data Era,” in *ESORICS*, pp. 182–199, Springer, 2014.
5. P. P. Tsang and V. K. Wei, “Short linkable ring signatures for e-voting, e-cash and attestation,” in *ISPEC*, vol. 3439, pp. 48–60, Springer, 2005.
6. S. Noether, “Ring Signature Confidential Transactions for Monero,” *https://eprint.iacr.org/2015/1098*, vol. 2015, p. 1098, 2015.
7. T. ElGamal, “A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms,” in *Advances in Cryptology*, pp. 10–18, Springer, 1984.
8. T. Elgamal, “A public key cryptosystem and a signature scheme based on discrete logarithms,” *IEEE Transactions on Information Theory*, vol. 31, no. 4, pp. 469–472, 1985.
9. R. L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems,” *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.
10. P. W. Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer,” *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999.
11. D. Micciancio and O. Regev, “Lattice-based cryptography,” in *Post-quantum cryptography*, pp. 147–191, Springer, 2009.
12. L. Chen, L. Chen, S. Jordan, Y.-K. Liu, D. Moody, R. Peralta, R. Perlner, and D. Smith-Tone, *Report on post-quantum cryptography.* NIST, 2016.
13. L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky, “Lattice signatures and bimodal gaussians,” in *CRYPTO*, pp. 40–56, Springer, 2013.
14. J. K. Liu, M. H. Au, W. Susilo, and J. Zhou, “Linkable ring signature with unconditional anonymity,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 1, pp. 157–165, 2014.
15. M. H. Au, S. S. M. Chow, W. Susilo, and P. P. Tsang, “Short Linkable Ring Signatures Revisited,” in *EuroPKI*, pp. 101–115, Springer, 2006.
16. M. H. Au, J. K. Liu, W. Susilo, and T. H. Yuen, “Certificate Based (Linkable) Ring Signature,” in *ISPEC*, pp. 79–92, Springer, 2007.
17. M. H. Au, J. K. Liu, W. Susilo, and T. H. Yuen, “Secure ID-based linkable and revocable-iff-linked ring signature with constant-size construction,” in *INDOCRYPT*, vol. 4329, pp. 364–378, Springer, 2006.
18. J. K. Liu, D. S. Wong, J. K. Liu, and D. S. Wong, “Enhanced security models and a generic construction approach for linkable ring signature,” *Int. J. Found. of Comput. Sci.*, vol. 17, no. 6, pp. 1403–1422, 2006.

19. E. Fujisaki and K. Suzuki, “Traceable ring signature,” in *PKC*, vol. 4450, pp. 181–200, Springer, 2007.
20. E. Fujisaki, “Sub-linear Size Traceable Ring Signatures without Random Oracles,” in *CT-RSA*, vol. 11, pp. 393–415, Springer, 2011.
21. J. K. Liu and D. S. Wong, “Linkable Ring Signatures: Security Models and New Schemes,” in *ICCSA*, pp. 614–623, Springer, 2005.
22. J. K. Liu, W. Susilo, and D. S. Wong, “Ring Signature with Designated Linkability,” in *IWSEC*, pp. 104–119, Springer, 2006.
23. P. P. Tsang, M. H. Au, J. K. Liu, W. Susilo, and D. S. Wong, “A suite of non-pairing id-based threshold ring signature schemes with different levels of anonymity,” in *ProvSec*, vol. 6402, pp. 166–183, Springer, 2010.
24. T. H. Yuen, J. K. Liu, M. H. Au, W. Susilo, and J. Zhou, “Efficient Linkable and/or Threshold Ring Signature Without Random Oracles,” *The Computer Journal*, vol. 56, pp. 407–421, 4 2013.
25. D. Zheng, X. Li, K. Chen, and J. Li, “Linkable Ring Signatures from Linear Feedback Shift Register,” in *EUC*, pp. 716–727, Berlin, Heidelberg: Springer, 2007.
26. L. K. Grover and L. K., “A fast quantum mechanical algorithm for database search,” in *STOC*, pp. 212–219, ACM, 1996.
27. D. J. Bernstein and T. Lange, “Post-quantum cryptography,” *Nature*, vol. 549, pp. 188–194, 9 2017.
28. K. Lauter, “Postquantum Opportunities: Lattices, Homomorphic Encryption, and Supersingular Isogeny Graphs,” *IEEE Security & Privacy*, vol. 15, no. 4, pp. 22–27, 2017.
29. O. Goldreich, S. Goldwasser, and S. Halevi, “Public-key cryptosystems from lattice reduction problems,” in *CRYPTO*, p. 112, Springer, 1997.
30. J. Hoffstein, J. Pipher, and J. Silverman, “NSS: An NTRU lattice-based signature scheme,” in *EUROCRYPT*, pp. 211–228, Springer, 2001.
31. C. Gentry, C. Peikert, and V. Vaikuntanathan, “Trapdoors for hard lattices and new cryptographic constructions,” in *STOC*, p. 197, ACM, 2008.
32. V. Lyubashevsky, “Fiat-Shamir with Aborts: Applications to Lattice and Factoring-Based Signatures,” in *ASIACRYPT*, pp. 598–616, Springer, 2009.
33. A. Fiat and A. Shamir, “How To Prove Yourself: Practical Solutions to Identification and Signature Problems,” in *CRYPTO*, pp. 186–194, Springer, 1986.
34. M. Abdalla, J. An, M. Bellare, and C. Namprempre, “From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security,” in *EUROCRYPT*, pp. 418–433, Springer, 2002.
35. M. Ajtai, “Generating hard instances of lattice problems,” in *STOC*, pp. 99–108, ACM, 1996.
36. D. Micciancio, “Generalized compact knapsacks, cyclic lattices, and efficient one-way functions,” *Computational Complexity*, vol. 16, no. 4, pp. 365–411, 2007.
37. V. Lyubashevsky, C. Peikert, and O. Regev, “On ideal lattices and learning with errors over rings,” in *EUROCRYPT*, pp. 1–23, Springer, 2010.
38. Z. Brakerski and Y. T. Kalai, “A Framework for Efficient Signatures, Ring Signatures and Identity Based Encryption in the Standard Model,” *https://eprint.iacr.org/2010/086*, 2010.
39. P.-L. Cayrel, R. Lindner, M. Rückert, and R. Silva, “A Lattice-Based Threshold Ring Signature Scheme,” in *LATINCRYPT*, pp. 255–272, Springer, 2010.
40. C. Wang and H. Wang, “A New Ring Signature Scheme from NTRU Lattice,” in *ICCIS*, pp. 353–356, IEEE, 2012.

41. C. Aguilar Melchor, S. Bettaieb, X. Boyen, L. Fousse, and P. Gaborit, "Adapting Lyubashevskys Signature Schemes to the Ring Signature Setting," in *AFRICACRYPT*, pp. 1–25, Springer, 2013.
42. J. Wang and B. Sun, "Ring Signature Schemes from Lattice Basis Delegation," in *ICICS*, pp. 15–28, Springer, 2011.
43. B. Libert, S. Ling, K. Nguyen, and H. Wang, "Zero-Knowledge Arguments for Lattice-Based Accumulators: Logarithmic-Size Ring Signatures and Group Signatures Without Trapdoors," in *EUROCRYPT*, pp. 1–31, Springer, 2016.
44. R. Yang, M. Ho Au, J. Lai, Q. Xu, and Z. Yu, "Lattice-Based Techniques for Accountable Anonymity: Composition of Abstract Stern's Protocols and Weak PRF with Efficient Protocols from LWR," *https://eprint.iacr.org/2017/781*, 2017.
45. H. Zhang, F. Zhang, H. Tian, and M. H. Au, "Anonymous Post-Quantum Cryptocash (Full Version)," *https://eprint.iacr.org/2017/716*, 2017.
46. C. Baum, L. Huang, and O. Sabine, "Towards Practical Lattice-Based One-Time Linkable Ring Signatures," *https://eprint.iacr.org/2018/107*, 2018.
47. V. Lyubashevsky and G. Seiler, "Short, Invertible Elements in Partially Splitting Cyclotomic Rings and Applications to Lattice-Based Zero-Knowledge Proofs," in *EUROCRYPT*, pp. 204–224, Springer, 2018.
48. S.-F. Sun, M. H. Au, J. K. Liu, and T. H. Yuen, "RingCT 2.0: A Compact Accumulator-Based (Linkable Ring Signature) Protocol for Blockchain Cryptocurrency Monero," in *ESORICS*, pp. 456–474, Springer, 2017.
49. Y. Chen and P. Q. Nguyen, "BKZ 2.0: Better Lattice Security Estimates," in *ASIACRYPT*, pp. 1–20, Springer, 2011.
50. M. Bellare and G. Neven, "Multi-signatures in the plain public-Key model and a general forking lemma," in *CCS*, p. 390, ACM, 2006.
51. V. Shoup, "Sequences of games: a tool for taming complexity in security proofs," *https://eprint.iacr.org/2004/332*, 2004.

A L2RS - Correctness requirements

A.1 Correctness of SigGen

Beyond the required conditions of SigVer, we stated that if $\sigma_L(\mu_1) = (\mathbf{c}_1, \mathbf{t}_1, \ldots, \mathbf{t}_w, \mathbf{H})$ is the output of the SigGen algorithm on input $(\mu, L, \mathbf{S}_\pi)$, then the output of SigVer on input $(\mu, L, \sigma_L(\mu_1))$ should be accepted. We need to show that when SigVer computes $H_1(L, \mathbf{H}, \mu, \mathbf{A}_w \cdot \mathbf{t}_w + q \cdot \mathbf{c}_w, q \cdot \mathbf{t}_w + \mathbf{H} \cdot \mathbf{c}_w)$, the result is equal to $\mathbf{c}_1$. It is also illustrated that $H_1(L, \mathbf{H}, \mu, \mathbf{A}_i \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i, \mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i) = \mathbf{c}_{i+1}$ for $1 \leq i \leq w-1$ in SigVer. In this evaluation, we consider two scenarios, one when $i \neq \pi$ and $i = \pi$:

- For $i \neq \pi$, in SigGen we have $\mathbf{c}_{i+1} = H_1(L, \mathbf{H}, \mu, \mathbf{A}_i \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i, \mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i)$, while in SigVer we compute $\mathbf{c}_{i+1} = H_1(L, \mathbf{H}, \mu, \mathbf{A}_i \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i, \mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i)$. These are equal since $\mathbf{A}_i \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i$ (in SigGen) $= \mathbf{A}_i \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i$ (in SigVer); and $\mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i$ (in SigGen) $= \mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i$ (in SigVer).

- For $i = \pi$, in SigGen we have $\mathbf{c}_{\pi+1} = H_1(L, \mathbf{H}, \mu, \mathbf{A}_\pi \cdot \mathbf{u}, \mathbf{H} \cdot \mathbf{u})$, whereas in SigVer we calculate $\mathbf{c}_{\pi+1} = H_1(L, \mathbf{H}, \mu, \mathbf{A}_\pi \cdot \mathbf{t}_\pi + q \cdot \mathbf{c}_\pi, \mathbf{H} \cdot \mathbf{t}_\pi + q \cdot \mathbf{c}_\pi)$. In this case, we need to show that $\mathbf{c}_{\pi+1}$ (in SigGen) $= \mathbf{c}_{\pi+1}$ (in SigVer). In doing so, the following equalities need to be proved:

1. $\mathbf{A}_\pi \cdot \mathbf{u} = \mathbf{A}_\pi \cdot \mathbf{t}_\pi + q \cdot \mathbf{c}_\pi$, which is equivalent to $\mathbf{A}_\pi \cdot (\mathbf{u} - \mathbf{t}_\pi) = q \cdot \mathbf{c}_\pi$. Here, we replace $\mathbf{t}_\pi$ as defined in **Algorithm 2**, to obtain:

$$\mathbf{A}_\pi \cdot (\mathbf{u} - \mathbf{u} - \mathbf{S}_\pi \cdot \mathbf{c}_\pi \cdot (-1)^b) = q \cdot \mathbf{c}_\pi \iff$$
$$-\mathbf{A}_\pi \cdot \mathbf{S}_\pi \cdot \mathbf{c}_\pi \cdot (-1)^b = q \cdot \mathbf{c}_\pi \iff$$
$$-q \cdot \mathbf{c}_\pi \cdot (-1)^b = q \cdot \mathbf{c}_\pi$$

We distinguish two cases for b:
   - When b = 0, we verify that -$q \cdot \mathbf{c}_\pi = q \cdot \mathbf{c}_\pi \bmod 2q$.
   - When b = 1, we have $q \cdot \mathbf{c}_\pi = q \cdot \mathbf{c}_\pi \bmod 2q$.

2. $\mathbf{H} \cdot \mathbf{u} = \mathbf{H} \cdot \mathbf{t}_\pi + q \cdot \mathbf{c}_\pi$, which means that:

$$\mathbf{H} \cdot (\mathbf{u} - \mathbf{t}_\pi) = q \cdot \mathbf{c}_\pi \iff$$
$$\mathbf{H} \cdot (\mathbf{u} - \mathbf{u} - \mathbf{S}_\pi \cdot \mathbf{c}_\pi \cdot (-1)^b) = q \cdot \mathbf{c}_\pi \iff$$
$$-\mathbf{H} \cdot \mathbf{S}_\pi \cdot \mathbf{c}_\pi \cdot (-1)^b = q \cdot \mathbf{c}_\pi \iff$$
$$-q \cdot \mathbf{c}_\pi \cdot (-1)^b = q \cdot \mathbf{c}_\pi$$

We distinguish between two cases:
   - When b = 0, it is verified that $-q \cdot \mathbf{c}_\pi = q \cdot \mathbf{c}_\pi \bmod 2q$.
   - When b = 1, we have $q \cdot \mathbf{c}_\pi = q \cdot \mathbf{c}_\pi \bmod 2q$.

A.2 Correctness of SigLink

We show that an honest user π who signs two messages μ_1 and μ_2 in the L2RS with the list of public-keys L, will obtain a **Linked** answer when query the SigLink algorithm with overwhelming probability. As shown in **Algorithm 4**, two signatures $\sigma_L(\mu_1)$ and $\sigma_L(\mu_2)$ were created, these were also successfully verified by SigVer. Saying this, the linkability tags $\mathbf{H}_{\mu_1}$ and $\mathbf{H}_{\mu_2}$ must be equal. We know that:

$$\mathbf{H}_{\mu_1} = (\mathbf{H}_{0,\mu_1}, \mathbf{h}_{1,\mu_1}), \text{where } \mathbf{H}_{0,\mu_1} = 2 \cdot \mathbf{H}'_{0,\mu_1} \text{ and } \mathbf{h}_{1,\mu_1} = -\mathbf{H}_{0,\mu_1} \cdot \mathbf{S}_{\pi,0} + q$$
$$\mathbf{H}_{\mu_2} = (\mathbf{H}_{0,\mu_2}, \mathbf{h}_{1,\mu_2}), \text{where } \mathbf{H}_{0,\mu_2} = 2 \cdot \mathbf{H}'_{0,\mu_2} \text{ and } \mathbf{h}_{1,\mu_2} = -\mathbf{H}_{0,\mu_2} \cdot \mathbf{S}_{\pi,0} + q$$

The first part of the linkability tag will have the same output from the oracle as it uses the same list of public-keys L, thus:

$$Pr[\mathbf{H}_{0,\mu_1} = \mathbf{H}_{0,\mu_2}] = 1.$$

Ultimately, the second part uses the honest user's private-key $\mathbf{S}_\pi$ along with the first part of the linkability tag, so we conclude that:

$$Pr[-\mathbf{H}_{0,\mu_1} \cdot \mathbf{S}_{\pi,0} + q + \mathbf{H}_{0,\mu_2} \cdot \mathbf{S}_{\pi,0} - q = 0] = 1.$$

B Security Analysis - One-Time Unforgeability

As stated in [13], this L2RS scheme relies on the $\mathcal{R}\text{-}\mathbf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem to be secure against any existential forger. This means that a forgery algorithm succeeds with a negligible probability and so we conclude that under this probability, the attacker will also find a solution to the $\mathcal{R}\text{-}\mathbf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem. To prove this, we start replacing the SigGen algorithm with Hybrid 1 and Hybrid 2 algorithms that are used to simulate the creation of the L2RS signatures, until we obtain an algorithm that breaks the $\mathcal{R}\text{-}\mathbf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem. These Hybrid algorithms are illustrated in **Algorithm 7** and **Algorithm 8**, respectively.

The difference between SigGen and Hybrid 1 is that in Hybrid 1 the output of the random oracle H_1 is chosen at random from $\mathcal{S}_{n,\kappa} \subseteq \mathcal{R}_{2q}$ and then it is programmed, without checking the value of $\mathbf{A}_w \cdot \mathbf{u}$ and $\mathbf{H} \cdot \mathbf{u}$ being already set. This equality can be described as:

$$H_1(L, \mathbf{H}, \mu, \mathbf{A}_w \cdot \mathbf{t}_w + q \cdot \mathbf{c}_w, \mathbf{H} \cdot \mathbf{c}_w + q \cdot \mathbf{t}_w) = H_1(L, \mathbf{H}, \mu, \mathbf{A}_w \cdot \mathbf{u}, \mathbf{H} \cdot \mathbf{u})$$

Every time the Hybrid 1 is called, the probability of generating a $\mathbf{u}$ such that $\mathbf{A}_w \cdot \mathbf{u}$ and $\mathbf{H} \cdot \mathbf{u}$ are equal to one of the previous output that was queried is at most 2^{-n+1}. We define that the probability of getting a collusion each time is at most $h2^{-n+1}$, where "h" is the number of calls to the random oracle H_1, whereas the probability of occurring a collision after "o" queries to the Hybrid 1 is at most $o \cdot h2^{-n+1}$, which is negligible. (Based on [13], Lemma 3.4).

Algorithm 7 One-Time Unforgeability - Signature algorithm of Hybrid 1

```
Input: $\mathbf{S}_\pi, \mu, L$
Output: $(\mathbf{c}_1, \mathbf{t}_1, \ldots, \mathbf{t}_w, \mathbf{H})$
 1: procedure HYBRID-1($\mathbf{S}_\pi, \mu, L$)
 2:     Let $\mathbf{u} = (u_1, \ldots, u_m)^T$ where $u_i \leftarrow D^n_\sigma$.
 3:     $\boxed{\text{Choose at random } \mathbf{c}_{\pi+1} \leftarrow \mathcal{S}_{n,\kappa}}$
 4:     for $(i = \pi+1, \pi+2, \ldots, w, 1, 2, \ldots, \pi-1)$ do
 5:         Let $\mathbf{t}_i = (t_{i,1}, \ldots, t_{i,m})^T$, where $t_{i,j} \leftarrow D^n_\sigma$.
 6:         $\mathbf{c}_{i+1} \leftarrow H_1(L, \mathbf{H}, \mu, \mathbf{A}_i \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i, \mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i)$
 7:     Choose $b \leftarrow \{0, 1\}$
 8:     $\mathbf{t}_\pi \leftarrow \mathbf{u} + \mathbf{S}_\pi \cdot \mathbf{c}_\pi \cdot (-1)^b$
 9:     Continue with probability $\frac{1}{\left(M \exp\left(-\frac{\|\mathbf{S}_\pi \cdot \mathbf{c}_\pi\|^2}{2\sigma^2}\right) \cosh\left(\frac{\langle \mathbf{t}_\pi, \mathbf{S}_\pi \cdot \mathbf{c}_\pi \rangle}{\sigma^2}\right)\right)}$
    otherwise Restart.
10:     return $(\mathbf{c}_1, \mathbf{t}_1, \ldots, \mathbf{t}_w, \mathbf{H})$
```

After analyzing how $\mathbf{c}_1$ can be forged, we evaluate the $(\mathbf{t}_1, \ldots, \mathbf{t}_w)$ of the L2RS scheme. We claim that these are forgeable when an attacker finds a PPT algorithm $\mathcal{F}$ to solve the $\mathcal{R}\text{-}\mathbf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem. This attack can be simulated using the Hydrid 2 shown in **Algorithm 8**, where $\mathbf{t}_\pi$ is directly chosen from the distribution D_σ^n (Based on [13], Lemma 3.5).

The public-key $\mathbf{A} \in \mathcal{R}_{2q}^{1\times m}$ is generated such $\mathbf{A} \cdot \mathbf{S} = q \in \mathcal{R}_{2q}$ and finding a vector $\mathbf{v}$ such that $\mathbf{A}\mathbf{v} = 0 \bmod q$. We denote $y = h$ where y is the number of times the random oracle H_1 is programmed during this attack. Then this attack is performed as follows:

1. Random coins are selected for the forger ϕ and signer ψ.
2. The random oracle H_1 is called to generate the responses of the users in the L2RS scheme, $(\mathbf{c}_1, \ldots, \mathbf{c}_w) \leftarrow \mathcal{S}_{n,\kappa}$.
3. These create a *SubRoutine* that takes as input $(\mathbf{A}, \phi, \psi, \mathbf{c}_1, \ldots, \mathbf{c}_w)$.
4. $\mathcal{F}$ is initialized and run by providing the $\mathbf{A}$ and forger's random coins ϕ.
5. The *SubRoutine* signs the message μ using the signer's coins ψ in the Hydrid 2, this produces a signature $\sigma_L(\mu)$.
6. During the signing process, $\mathcal{F}$ will call the oracle H_1 and its answers are placed the list $(\mathbf{c}_1, \ldots, \mathbf{c}_w)$, it is also kept the queries in a table in the event that same queries are used in this oracle.
7. $\mathcal{F}$ is stopped and it outputs a forgery that is the *SubRoutine*'s result $(\mathbf{c}_1, \mathbf{t}_1, \ldots, \mathbf{t}_w, \mathbf{H})$, with negligible probability δ. This output has to be successfully accepted by the SigVer algorithm.

If the random oracle was not called using some input $\mathbf{A}_i \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i, \mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i$, then $\mathcal{F}$ has $1/|\mathcal{S}_{n,\kappa}|$ chance of producing a $\mathbf{c}$ such that $\mathbf{c} = H_1(L, \mathbf{H}, \mu, \mathbf{A} \cdot \mathbf{t} + q \cdot \mathbf{c}, \mathbf{H} \cdot \mathbf{t} + q \cdot \mathbf{c})$. This turns out that $\delta - 1/|\mathcal{S}_{n,\kappa}|$ be the probability that $\mathbf{c} = \mathbf{c}_j$ for some j.

Algorithm 8 One-Time Unforgeability - Signature algorithm of Hybrid 2 $\sigma_L(\mu)$

```
Input: $\mathbf{S}_\pi, \mu, L$
Output: $(\mathbf{c}_1, \mathbf{t}_1, \ldots, \mathbf{t}_w, \mathbf{H})$
 1: procedure HYBRID-2($\mathbf{S}_\pi, \mu, L$)
 2:     Let $\mathbf{u} = (u_1, \ldots, u_m)^T$, where $u_i \leftarrow D_\sigma^n$.
 3:     Choose at random $\mathbf{c}_{\pi+1} \leftarrow \mathcal{S}_{n,\kappa}$
 4:     for $(i = \pi + 1, \pi + 2, \ldots, w, 1, 2, \ldots, \pi - 1)$ do
 5:         Let $\mathbf{t}_i = (t_{i,1} \ldots t_{i,m})^T$, where $t_{i,j} \leftarrow D_\sigma^n$.
 6:         $\mathbf{c}_{i+1} \leftarrow H_1(L, \mathbf{H}, \mu, \mathbf{A}_i \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i, \mathbf{H} \cdot \mathbf{t}_i + q \cdot \mathbf{c}_i)$
 7:     Choose $b \leftarrow \{0, 1\}$
 8:     $\boxed{\text{Choose } \mathbf{t}_\pi \leftarrow D_\sigma^m}$
 9:     Continue with probability $\frac{1}{M}$ otherwise Restart.
10:     return $(\mathbf{c}_1, \mathbf{t}_1, \ldots, \mathbf{t}_w, \mathbf{H})$
```

FORGERY 1. Let's consider the situation that $\mathbf{c}_{j+1}$ is the result after using $\mathcal{F}$ which is $\mathbf{c}_{j+1} = H_1(L, \mathbf{H}, \mu', \mathbf{A} \cdot \mathbf{t}' + q \cdot \mathbf{c}_j, \mathbf{H} \cdot \mathbf{t}' + q \cdot \mathbf{c}_j)$. Then by comparing this with a legitimate signature, we have:

$$H_1(L, \mathbf{H}, \mu, \mathbf{A} \cdot \mathbf{t} + q \cdot \mathbf{c}_j, \mathbf{H} \cdot \mathbf{t} + q \cdot \mathbf{c}_j) = H_1(L, \mathbf{H}, \mu', \mathbf{A} \cdot \mathbf{t}' + q \cdot \mathbf{c}_j, \mathbf{H} \cdot \mathbf{t}' + q \cdot \mathbf{c}_j)$$

$\mathcal{F}$ will find a preimage of $\mathbf{c}_j$ if $\mu \neq \mu'$ or $\mathbf{A} \cdot \mathbf{t} + q \cdot \mathbf{c}_j \neq \mathbf{A} \cdot \mathbf{t}' + q \cdot \mathbf{c}_j$ or $\mathbf{H} \cdot \mathbf{t} + q \cdot \mathbf{c}_j \neq \mathbf{H} \cdot \mathbf{t}' + q \cdot \mathbf{c}_j$. Under this situation, we have with overwhelming probability that $\mu = \mu'$ and $\mathbf{A} \cdot \mathbf{t} + q \cdot \mathbf{c}_j = \mathbf{A} \cdot \mathbf{t}' + q \cdot \mathbf{c}_j$ and $\mathbf{H} \cdot \mathbf{t} + q \cdot \mathbf{c}_j = \mathbf{H} \cdot \mathbf{t}' + q \cdot \mathbf{c}_j$. These equalities will result in: $\mathbf{A}(\mathbf{t} - \mathbf{t}') = 0 \bmod 2q$ and $\mathbf{H}(\mathbf{t} - \mathbf{t}') = 0 \bmod 2q$. We assume that both $\mathbf{t}$ and $\mathbf{t}'$ are different and they met the SigVer conditions, so it yields $\mathbf{t} - \mathbf{t}' \neq 0 \bmod q$, and $\|\mathbf{t} - \mathbf{t}'\| \leq 2B_2$.

Forgery 2. In this scenario, we assume that the L2RS scheme can be forged by an attacker $\mathcal{F}$ as it was presented in the Forgery 1 and obtain $\mathbf{c}_j$, then another attacker can generate $(\mathbf{c}'_j, \ldots, \mathbf{c}'_w) \leftarrow \mathcal{S}_{n,\kappa}$ by replaying the first attack and using same message μ. We use the forking lemma [50] to show the probability of $\mathbf{c}_j = \mathbf{c}'_j$ and the forger uses an oracle response $\mathbf{c}'_j$ is at least:

$$\left(\delta - \frac{1}{|\mathcal{S}_{n,\kappa}|}\right) \cdot \left(\frac{\delta - \frac{1}{|\mathcal{S}_{n,\kappa}|}}{y} - \frac{1}{|\mathcal{S}_{n,\kappa}|}\right) \tag{3}$$

Therefore, with the probability (3), $\mathcal{F}$ creates a signature $\sigma_L(\mu) = (\mathbf{c}'_1, \mathbf{t}'_1, \ldots, \mathbf{t}'_w, \mathbf{H})$ where $\mathbf{A} \cdot \mathbf{t} + q \cdot \mathbf{c}_j = \mathbf{A} \cdot \mathbf{t}' + q \cdot \mathbf{c}'_j$ and $\mathbf{H} \cdot \mathbf{t} + q \cdot \mathbf{c}_j = \mathbf{H} \cdot \mathbf{t}' + q \cdot \mathbf{c}'_j$. We now obtained:

$\mathbf{A}(\mathbf{t} - \mathbf{t}') = \mathbf{q}(\mathbf{c}_j - \mathbf{c}'_j) \bmod 2q$ and $\mathbf{H}(\mathbf{t} - \mathbf{t}') = \mathbf{q}(\mathbf{c}_j - \mathbf{c}'_j) \bmod 2q$. Since $\mathbf{c}_j - \mathbf{c}'_j \neq 0 \bmod 2$, so in both cases ($\mathbf{A}$ and $\mathbf{H}$), we have $\mathbf{t} - \mathbf{t}' \neq 0 \bmod 2q$ where $\|\mathbf{t} - \mathbf{t}'\|_\infty < q/2$. By applying this reduction, we find a small non-zero vector $\mathbf{v} = \mathbf{t} - \mathbf{t}' \neq 0 \bmod q$. This $\mathbf{v}$ will compute $\mathbf{A}\mathbf{v} = 0 \bmod q$ with $\|\mathbf{v}\| \leq 2B_2$. Since $\mathbf{A} \bmod q = 2(\mathbf{A}'_0, -\mathbf{a}'_1) \bmod q$, we have $2(\mathbf{A}'_0, -\mathbf{a}'_1)\mathbf{v} = 0 \bmod q$, this implies that $(\mathbf{A}'_0, -\mathbf{a}'_1)\mathbf{v} = 0 \bmod q$, since q is odd. This vector $\mathbf{v}$ will be a solution to the $\mathcal{R}\text{-}\mathbf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem with $\beta = 2B_2$, with respect to $(\mathbf{A}'_0, -\mathbf{a}'_1)$ over $\mathcal{R}_q$. Notice that Hydrid 2 shown in **Algorithm 8** no longer uses the private-key $\mathbf{S}_\pi$, except for generating $\mathbf{A}$ to obtain the final $\mathcal{R}\text{-}\mathbf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ algorithm. We modified Hydrid 2 game to Hydrid 3 game shown in **Algorithm 9**, where it is changed the key generation of $\mathbf{A}$ to output an uniformly random $\mathbf{a}'_1 \leftarrow \mathcal{R}_q$ by the argument of the Leftover Hash Lemma (LHL) - **Lemma 1** and our assumption that $\sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$ is negligible in n. The success of probability of an attacker in Hydrid 3 game differs by a negligible amount from the success probability in Hydrid 2 and is thus non-negligible. Therefore, this vector $\mathbf{v}$ will be also ultimately a solution to the $\mathcal{R}\text{-}\mathbf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem with $\beta = 2B_2$ with non-negligible probability.

C Security Analysis - Anonymity

We prove the anonymity of this scheme using the sequence-of-games approach [51] where we make changes between successive games. In doing so, we use the "*transition based on indistinguishability*". We can start this analysis by:

Game 0: Suppose that an attacker $\mathcal{A}$ is given the list L, the signature $\sigma_L(\mu)$, message μ, and the random oracle models (H_1 and H_2). The key generation

Algorithm 9 Key pair generation of Hybrid 3 $(\mathbf{A}, \mathbf{S})$

```
Input: The public parameters Pub-Params: $\mathbf{A}'_0$ and $\mathbf{H}'_0$ .
Output: $(\mathbf{A}, \mathbf{S})$, where $\mathbf{A}$ is the public-key and $\mathbf{S}$ is the private-key.
1: procedure HYBRID-3(Pub-Params)
2:     Let $\mathbf{S}_0^T = (\mathbf{s}_{0,1}, \dots, \mathbf{s}_{0,m-1}) \in \mathcal{R}_q^{1\times(m-1)}$, where $\mathbf{s}_{0,i} \leftarrow (-2^\gamma, 2^\gamma)^n$, for $1 \le i \le m-1$
3:     Let $\mathbf{S}^T = (\mathbf{S}_0^T, 1) \in \mathcal{R}_q^{1\times m}$
4:     $\boxed{\text{Choose } \mathbf{a}'_1 \leftarrow \mathcal{R}_q}$
5:     Call function L2RS.Lift$(\mathbf{A}'_0, \mathbf{a}'_1)$, and it returns $\mathbf{A} = (\mathbf{A}_0, \mathbf{a}_1) = (2\cdot\mathbf{A}'_0, -2\cdot\mathbf{a}'_1 + q \bmod 2q) \in \mathcal{R}_{2q}^{1\times m}$
6:     Remark: $\mathbf{A}\cdot\mathbf{S} = q \in \mathcal{R}_{2q}$, where $\mathbf{S} \in \mathcal{R}_{2q}^{m\times 1}$
7:     return $(\mathbf{A}, \mathbf{S})$.
```

algorithm creates the pair of users' keys in the ring signature: Private-Keys $\leftarrow$ $(\mathbf{S}_1, \dots, \mathbf{S}_w)$ and the Public-Keys $\leftarrow (\mathbf{A}_1, \dots, \mathbf{A}_w)$; a user π is chosen uniformly at random from the ring signature: $\pi \leftarrow 1, \dots, w$, then the signature $\sigma_L(\mu)$ is generated. So in this **Game 0**, we first select $\pi \leftarrow \{0,1\}$, then we call the L2RS.SigGen$(\mathbf{S}_\pi, \mu, L)$. A PPT adversary $\mathcal{A}$ outputs a guess π' for signer's index π; thus in the event **Game 0**, $\mathcal{A}$ succeeds in breaking ambiguity **Game 0**$(\pi = \pi')$ if $\Pr[\textbf{Game 0}] \le \frac{1}{w} + non - negligible$; otherwise, the $\mathcal{A}$ is just randomly guessing.

Game 1: Changes in this game are made to the user π in the second part of the linkability tag $\mathbf{h}_1 = -\mathbf{H}_0 \cdot \mathbf{S}_{\pi,0} + q \bmod q$, in signature of user π, and public-key $\mathbf{a}_1 = (-2\cdot\mathbf{A}'_0\cdot\mathbf{S}_{\pi,0}) \in \mathcal{R}_q$ in the KeyGen algorithm. $\mathbf{h}_1$ and $\mathbf{a}_1$ are now randomly chosen from $\mathcal{R}_q$. Note that $\mathbf{h}_1 \bmod 2 = 1$ and $\mathbf{a}_1 \bmod 2 = 1$ It can be claimed that $|\Pr[\textbf{Game 0}] - \Pr[\textbf{Game 1}]| \le \epsilon_{LHL_{G1}}$.

Where $\epsilon_{LHL_{G1}}$ is the advantage of some efficient algorithm which is negligible. In these both cases $\mathbf{h}_1 = -\mathbf{H}_0 \cdot \mathbf{S}_{\pi,0} \bmod q$ and $\mathbf{a}_1 = (-2\cdot\mathbf{A}'_0\cdot\mathbf{S}_{\pi,0} \bmod q)$, we know that $\mathbf{H}_0$ and $\mathbf{A}'_0$ are uniform and $\mathbf{S}_{\pi,0}$ is chosen small and with coefficients in $(-2^\gamma, 2^\gamma)$. When $\mathbf{S}_{\pi,0}$ is multiplied by $\mathbf{H}_0$ and $\mathbf{A}'_0$ respectively, it gives $\mathbf{h}_1$ and $\mathbf{a}_1$ that are close to uniform over $\mathcal{R}_q$. By applying the Leftover Hash Lemma (LHL) - **Lemma 1**, the statistical distance between the distribution of ($\mathbf{h}_1 \bmod q$ and $\mathbf{a}_1 \bmod q$) and the uniform distribution on $\mathcal{R}_q \times \mathcal{R}_q$ is at most $n\cdot\frac{1}{2}\cdot\sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$. We conclude that in **Game 1**:

$$|\Pr[\textbf{Game 0}] - \Pr[\textbf{Game 1}]| \le n\cdot\frac{1}{2}\cdot\sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}. \tag{4}$$

Game 2: This time a change is made in the second part of the remaining public-keys $\mathbf{a}_{1,i}$ $(1 \le i \le w,\ i \ne \pi)$ which are in the ring signature list L. They are now randomly chosen as $\mathbf{a}_{1,i} \leftarrow \mathcal{R}_q$. It turns out that $|\Pr[\textbf{Game 1}] - \Pr[\textbf{Game 2}]| \le \epsilon_{LHL_{G2}}$.

Where $\epsilon_{LHL_{G2}}$ is the advantage of some efficient algorithm which is negligible. We consider that for ($i = 1$ to w where $i \neq \pi$), we know that $\mathbf{a}_{1,i} = (-2 \cdot \mathbf{A}'_{0,i} \cdot \mathbf{S}_{i,0} \bmod q)$ are uniform and all $\mathbf{S}_{i,0}$'s are chosen small with coefficients in $(-2^\gamma, 2^\gamma)$. When the $\mathbf{S}_{i,0}$'s are multiplied by $\mathbf{A}'_{0,i}$'s, it gives $\mathbf{a}_{1,i} \bmod q$'s that are close to uniform over $\mathcal{R}_q$. By applying the Leftover Hash Lemma (LHL) - **Lemma 1**, the statistical distance between the distribution of the $(-2 \cdot \mathbf{A}'_{0,i} \cdot \mathbf{S}_{i,0} \bmod q)$'s and the uniform distribution on $\mathcal{R}_q \times \mathcal{R}_q$ is at most $n \cdot \frac{1}{2} \cdot \sqrt{\frac{q^n}{2^{(\gamma+1)\cdot(m-1)\cdot n}}} \cdot (w-1)$. So in **Game 2**, we conclude that:

$$|\Pr[\textbf{Game 1}] - \Pr[\textbf{Game 2}]| \leq n \cdot \frac{1}{2} \cdot \sqrt{\frac{q^n}{2^{(\gamma+1)\cdot(m-1)\cdot n}}} \cdot (w-1). \tag{5}$$

Game 3: At this time, we make a change in $\mathbf{c}_{\pi+1}$. Instead of programming the oracle as $H_1(L, \mathbf{H}, \mu, \mathbf{A}_i \cdot \mathbf{u}, \mathbf{H} \cdot \mathbf{u})$, it is now randomly chosen $\mathbf{c}_{\pi+1} \leftarrow \mathcal{S}_{n,\kappa}$. We have that $|\Pr[\textbf{Game 2}] - \Pr[\textbf{Game 3}]| \leq \epsilon_{G3}$ where ϵ_{G3} is the advantage of some efficient algorithm which is negligible. This scenario outputs a signature $\sigma_L(\mu_1) = (\mathbf{c}_1, \mathbf{t}_1, \ldots, \mathbf{t}_w, \mathbf{H})$ and programs the oracle as $H_1(L, \mathbf{H}, \mu, \mathbf{A}_\pi \cdot \mathbf{t}_\pi + \mathbf{A}_\pi \cdot \mathbf{c}_\pi, \mathbf{H} \cdot \mathbf{t}_\pi + q \cdot \mathbf{c}_\pi) = \mathbf{c}_{\pi+1}$. Then, the adversary $\mathcal{A}$ makes h queries to H_1; so the distinguishing advantage of the signing algorithm and the one in **Game 2** is at most $h \cdot 2^{-n+1}$. We conclude that in **Game 3**:

$$|\Pr[\textbf{Game 2}] - \Pr[\textbf{Game 3}]| \leq h \cdot 2^{-n+1}. \tag{6}$$

Game 4: In this game a change is made in $\mathbf{t}_\pi$. Namely, instead of computing it as $\mathbf{u} + \mathbf{S}_\pi \cdot \mathbf{c}_\pi \cdot (-1)^{bit}$, it is now directly chosen from the Gaussian distribution D^n_σ. It is argued that $|\Pr[\textbf{Game 3}] - \Pr[\textbf{Game 4}]| \leq \epsilon_{RS_{G4}}$.

Where $\epsilon_{RS_{G4}}$ is the advantage of some efficient algorithm which is negligible. In previous Games, $\mathbf{t}_\pi$ is computed using rejection sampling - **Lemma 2**, thus it will always have a sample from the Gaussian distribution D^n_σ. In this Game, however, $\mathbf{t}_\pi$ is directly chosen from D^n_σ, this means that the advantage $\epsilon_{RS_{G4}}$ will be zero as in both **Game 3** and **Game 4**, $\mathbf{t}_\pi$ is having same distribution. In **Game 4**, we have:

$$|\Pr[\textbf{Game 3}] - \Pr[\textbf{Game 4}]| = 0. \tag{7}$$

Game 5: Finally, in the **Game 5**, a change is made in the index π. Namely, instead of choosing $\pi + 1$, it will be randomly chosen $(1, \ldots, w)$. We claim that $|\Pr[\textbf{Game 4}] - \Pr[\textbf{Game 5}]| \leq \epsilon_{G5}$ where ϵ_{G5} is the advantage of some efficient algorithm which is negligible. In this **Game 5**, we consider that when π is replaced by a fixed d, it might produce some collisions with previous queries to the oracle H_1; saying this, the adversary $\mathcal{A}$ may make h queries to H_1; therefore, the distinguishing advantage of the signing algorithm between **Game 4** and this **Game 5** is at most $h \cdot 2^{-n+1} \cdot w$. Finally, in **Game 5** we have:

$$|\Pr[\textbf{Game 4}] - \Pr[\textbf{Game 5}]| \leq h \cdot 2^{-n+1} \cdot w. \tag{8}$$

We also conclude that in **Game 5**, the adversary's view is statistical independent of π, thus $\Pr[\textbf{Game 5}] = \frac{1}{w}$.
Combining the probabilities of the above games (4), (5), (6), (7) and (8) we obtain:

$$|\Pr[\textbf{Game 5}] - \Pr[\textbf{Game 0}]| \leq |\Pr[\textbf{Game 1}] - \Pr[\textbf{Game 0}]| + |\Pr[\textbf{Game 2}] - \Pr[\textbf{Game 1}]| + |\Pr[\textbf{Game 3}] - \Pr[\textbf{Game 2}]| + |\Pr[\textbf{Game 4}] - \Pr[\textbf{Game 3}]| + |\Pr[\textbf{Game 5}] - \Pr[\textbf{Game 4}]|.$$

By replacing the resulting probabilities, we have:

$$|\Pr[\textbf{Game 5}] - \Pr[\textbf{Game 0}]| \leq \frac{1}{w} - \frac{1}{w} + \epsilon, \tag{9}$$

which means that $|\Pr[\textbf{Game 5}] - \Pr[\textbf{Game 0}]| \leq \epsilon$, which itself is smaller than

$$\frac{n \cdot (w-1)}{2} \cdot \left(\sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}} + \sqrt{\frac{q^{n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}} \right) + h \cdot 2^{-n+1} \cdot (1+w).$$

We notice that since h and w are polynomial in n, we get $h \cdot 2^{-n+1} \cdot (1+w)$ is negligible in n. In addition, we can say that $\left(\sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}} + \sqrt{\frac{q^{n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}} \right) \leq 2 \cdot \sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$, which is negligible by the assumption that $\sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$ is negligible. Hence we conclude that ϵ is negligible, meaning that $\Pr[\textbf{Game 0}] \leq \frac{1}{w} + \epsilon$.

D Security Analysis - Linkability

Proof. We construct the algorithm $\mathcal{B}$ for R-SIS. This algorithm runs the linkability attack game (**Definition 5**) as follows:

1. $\mathcal{B}$ generates using the L2RS.KeyGen algorithm all private-keys $\mathbf{S}_i$'s with the corresponding public-keys $\mathbf{A}_i$'s, then $\mathcal{B}$ gives $\mathbf{S}_\pi$ to the attacker $\mathcal{A}$ as a response to the attacker's $\mathcal{CO}$ query.
2. $\mathcal{A}$ outputs two signatures $\sigma_L(\mu_1)$ and $\sigma'_{L'}(\mu')$ along with their corresponding public-key's lists L and L' such that both signatures are successfully verified, but the linkability tags are different $\mathbf{h}_{1,\mu_1} \neq \mathbf{h}_{1,\mu'}$.
3. $\mathcal{B}$ computes $\mathbf{h}_{1,\mu_\pi} = -\mathbf{H}_0 \cdot \mathbf{S}_{\pi,0} + q \bmod 2q$, where π is the true signer's π linkability tag. This $\mathbf{h}_{1,\mu_\pi}$ tag can then be compared with the linkability tags $\mathbf{h}_{1,\mu_1}$and $\mathbf{h}_{1,\mu'}$, output by $\mathcal{A}$, in step 2, and one of them will be different.
4. Without loss of generality, suppose $\mathbf{h}_{1,\mu_1} \neq \mathbf{h}_{1,\mu_\pi} \bmod 2q$. Using the forking lemma [50], $\mathcal{B}$ rewinds the attacker $\mathcal{A}$ to the H_1 query corresponding to the L2RS.SigVer of the signature $\sigma_L(\mu_1)$. $\mathcal{B}$ reruns $\mathcal{A}$ with a different response of H_1 and ultimately gets another signature: $\sigma_L(\mu_2) =$

$(\mathbf{c}_{1,\mu_2}, \mathbf{t}_{1,\mu_2}, \ldots, \mathbf{t}_{w,\mu_2}, \mathbf{H}_{\mu_2})$. This second signature is used to extract a solution to the Ring-SIS problem, in case the $\mathcal{A}$ finds an efficient way to unlink these signatures, as shown in step 7.

5. The adversary $\mathcal{A}$ matches the challenge message of both signatures where $\mathbf{H}_{\mu_1}$ and $\mathbf{A}_{w,\mu_1}$ are kept. Thus we have:
   (a) $\mathbf{A}_{w,\mu_1} \cdot \mathbf{t}_{w,\mu_1} + q \cdot \mathbf{c}_{w,\mu_1} = \mathbf{A}_{w,\mu_1} \cdot \mathbf{t}_{w,\mu_2} + q \cdot \mathbf{c}_{w,\mu_2}$,
   (b) $\mathbf{H}_{\mu_1} \cdot \mathbf{t}_{w,\mu_1} + q \cdot \mathbf{c}_{w,\mu_1} = \mathbf{H}_{\mu_1} \cdot \mathbf{t}_{w,\mu_2} + q \cdot \mathbf{c}_{w,\mu_2}$.
   These expressions can be represented as:
   (a) $\mathbf{A}_{w,\mu_1} \cdot (\mathbf{t}_{w,\mu_1} - \mathbf{t}_{w,\mu_2}) = q \cdot (\mathbf{c}_{w,\mu_2} - \mathbf{c}_{w,\mu_1})$,
   (b) $\mathbf{H}_{\mu_1} \cdot (\mathbf{t}_{w,\mu_1} - \mathbf{t}_{w,\mu_2}) = q \cdot (\mathbf{c}_{w,\mu_2} - \mathbf{c}_{w,\mu_1})$.
   Reducing them mod$q$ we have (if $(\mathbf{c}_{w,\mu_2} - \mathbf{c}_{w,\mu_1}) \neq 0 \bmod 2$):
   (a) $\mathbf{A}_{w,\mu_1} \cdot (\mathbf{t}_{w,\mu_1} - \mathbf{t}_{w,\mu_2}) = 0 \bmod q$,
   (b) $\mathbf{H}_{\mu_1} \cdot (\mathbf{t}_{w,\mu_1} - \mathbf{t}_{w,\mu_2}) = 0 \bmod q$.
   We denote by $\mathbf{t}'_{w,\mu_1}$, the first $(m-1)$ ring elements in $\mathbf{t}_{w,\mu_1}$ and by $\mathbf{t}''_{w,\mu_1}$the m-th ring element in $\mathbf{t}_{w,\mu_1}$, i.e. $\mathbf{t}_{w,\mu_1} - \mathbf{t}_{w,\mu_2} = \begin{pmatrix} \mathbf{t}'_{w,\mu_1} - \mathbf{t}'_{w,\mu_2} \\ \mathbf{t}''_{w,\mu_1} - \mathbf{t}''_{w,\mu_2} \end{pmatrix} \in \mathcal{R}_q^m$, and using the public-key and linkability parts, we have:
   (a) $2 \cdot \mathbf{A}'_0 \cdot (\mathbf{t}'_{w,\mu_1} - \mathbf{t}'_{w,\mu_2}) = -2 \cdot \mathbf{a}'_1 \cdot (\mathbf{t}''_{w,\mu_1} - \mathbf{t}''_{w,\mu_2})$,
   (b) $2 \cdot \mathbf{H}'_0 \cdot (\mathbf{t}'_{w,\mu_1} - \mathbf{t}'_{w,\mu_2}) = -2 \cdot \mathbf{h}'_{1,\mu_1} \cdot (\mathbf{t}''_{w,\mu_1} - \mathbf{t}''_{w,\mu_2})$, where $\mathbf{h}'_{1,\mu_1} \triangleq \mathbf{H}'_0 \cdot \mathbf{S}_{\pi,0} \in \mathcal{R}_q$.
6. We let $\bar{\mathbf{S}}_0 = \frac{(\mathbf{t}'_{w,\mu_1} - \mathbf{t}'_{w,\mu_2})}{(\mathbf{t}''_{w,\mu_1} - \mathbf{t}''_{w,\mu_2})} \bmod q$ where $(\mathbf{t}''_{w,\mu_1} - \mathbf{t}''_{w,\mu_2}) \neq 0 \bmod q$. We distinguish two cases:
   (a) If $\bar{\mathbf{S}}_0 \neq \mathbf{S}_{\pi,0} \bmod q$, since we have $\mathbf{A}'_0 \cdot \bar{\mathbf{S}}_0 = \mathbf{A}'_0 \cdot \mathbf{S}_{\pi,0} = \mathbf{a}'_1 \bmod q$, then $(\bar{\mathbf{S}}_0 - \mathbf{S}_0)$ is a small non-zero vector SIS solution for $\mathbf{A}'_0 \in \mathcal{R}_q^{1\times(m-1)}$.
   (b) If $\bar{\mathbf{S}}_0 = \mathbf{S}_{\pi,0} \bmod q$, then $\mathbf{h}'_{1,\mu_1} = \mathbf{H}'_{0,\mu_1} \cdot \bar{\mathbf{S}}_0 \bmod q = \mathbf{H}_{0,\mu_1} \cdot \mathbf{S}_0 \bmod q$. The target is to show that $\mathbf{h}_{1,\mu_1} = \mathbf{h}_{1,\mu_\pi} \bmod 2$ and $\mathbf{h}_{1,\mu_1} = \mathbf{h}_{1,\mu_\pi} \bmod q$. If so, then we have $\mathbf{h}_{1,\mu_1} = \mathbf{h}_{1,\mu_\pi} \bmod 2q$, which is a contradiction with our assumption at step 4 of this proof. We now prove the first target:
   $$\mathbf{h}_{1,\mu_1} = -2 \cdot \mathbf{h}'_{1,\mu_1} + q = 1 \bmod 2 = -2\mathbf{H}'_0 \cdot \mathbf{S}_{\pi,0} + q = \mathbf{h}_{1,\mu_\pi},$$
   where the first and the last equalities follow from definition of $\mathbf{h}_1$ in second line of Algorithm 2. To show the second target, we have
   $$\mathbf{h}_{1,\mu_1} = -2 \cdot \mathbf{h}'_{1,\mu_1} + q = -2 \cdot \mathbf{h}'_{1,\mu_1} \bmod q$$
   $$= -2 \cdot \mathbf{H}'_{0,\mu_1} \cdot \bar{\mathbf{S}}_0 \bmod q = -2 \cdot \mathbf{H}'_{0,\mu_1} \cdot \mathbf{S}_{\pi,0} \bmod q = \mathbf{h}_{1,\mu_\pi},$$
   where the first and the last equalities follow from definition of $\mathbf{h}_1$ in second line of Algorithm 2 and the middle equality is true based on the argument at the beginning of step (6.b).
7. Since $(\mathbf{c}_{w,\mu_2} - \mathbf{c}_{w,\mu_1}) \neq 0 \bmod 2$, we have $(\mathbf{t}_{w,\mu_1} - \mathbf{t}_{w,\mu_2}) \neq 0 \bmod 2q$. In addition, we know that $\|\mathbf{t}_{w,\mu_2} - \mathbf{t}_{w,\mu_1}\|_\infty < q/2$, which implies that $(\mathbf{t}_{w,\mu_1} - \mathbf{t}_{w,\mu_2}) \neq 0 \bmod q$. Ultimately, we have $\mathbf{A} \cdot (\mathbf{t}_{w,\mu_1} - \mathbf{t}_{w,\mu_2}) = 0 \bmod q$ and $\|(\mathbf{t}_{w,\mu_1} - \mathbf{t}_{w,\mu_2}) \bmod q\| \leq 2B_2$. Therefore, this small non-zero vector $(\mathbf{t}_{w,\mu_1} - \mathbf{t}_{w,\mu_2})$ is the output of the algorithm $\mathcal{B}$, and this vector will be a solution to the $\mathcal{R}$-$\mathbf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem with $\beta = 2B_2$ for $\mathbf{a}_1 \in \mathcal{R}_q$.

E Security Analysis - Non-Slanderability

Let's suppose there is a non-slanderability adversary $\mathcal{A}_{Sland}$ who is given $\mathbf{pk}_i, \mathbf{sk}_i, i \neq \pi,$ and $i \in \{1, \ldots w\}$, and he produces a valid signature $\sigma'_L(\mu)$ with linkability tag $\mathbf{H}_{\sigma'_L(\mu)}$ which is equal to $\mathbf{H}_{\sigma_L(\mu)}$, $\sigma_L(\mu)$ being the legitimate signature generated with respect to $\mathbf{sk}_\pi$. This means that $\mathcal{A}_{Sland}$ can create a signature with the linkability tag $\mathbf{H}_{\sigma_L(\mu)}$ without knowing $\mathbf{sk}_\pi$. The adversary can also compute a valid $\sigma''_L(\mu)$ with $\mathbf{sk}_i$, $i \neq \pi,$ and $i \in \{1, \ldots w\}$ for which $\mathbf{H}_{\sigma''_L(\mu)} \neq \mathbf{H}_{\sigma'_L(\mu)}$. We give $(\sigma''_L(\mu), \sigma'_L(\mu))$ to the forger, which can turn it to an R-SIS solution. In particular, it will be computationally secure when two valid signatures created by different users are unlinked using the L2RS algorithms. An adversary $\mathcal{A}$ will break these properties with negligible probability as demonstrated in Theorems (2 and 4), and with these probabilities the $\mathcal{A}$ will find a $\mathcal{R}$-$\mathbf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ solution. Therefore, non-slanderability is implied by the definitions and security analysis of the unforgeability and linkability.