Lattice RingCT v2.0 with Multiple Input and Multiple Output Wallets

Wilson Alberto Torres[1], Veronika Kuchta[1], Ron Steinfeld[1], Amin Sakzad[1], Joseph K. Liu[1] and Jacob Cheng[2]

[1] Faculty of IT, Monash University, Melbourne, Australia
{Wilson.Torres,Veronika.Kuchta,Ron.Steinfeld,
Amin.Sakzad,Joseph.Liu}@monash.edu
[2] Collinstar Capital, Melbourne, Australia
jacob@collinstar.com

Abstract. This paper presents the Lattice-based Ring Confidential Transactions "*Lattice RingCT v2.0*" protocol. Unlike the previous Lattice RingCT v1.0 (LRCT v1.0) protocol, the new protocol supports Multiple-Input and Multiple-Output (MIMO) wallets in transactions, and it is a fully functional protocol construction for cryptocurrency applications such as *Hcash*. Since the MIMO cryptocurrency setting introduces new balance security requirements (and in particular, security against *out-of-range* amount attacks), we give a refined balance security model to capture such attacks, as well as a refined anonymity model to capture amount privacy attacks. Our protocol extends a previously proposed ring signature scheme in the LRCT v1.0 protocol, to support the MIMO requirements while preserving the post-quantum security guarantees, and uses a lattice-based zero-knowledge range proof to achieve security against *out-of-range* attacks. Preliminary parameter estimates and signature sizes are proposed as a point of reference for future studies.

Keywords: Cryptocurrencies, Lattice-Based Cryptography, Post-Quantum Cryptography, RingCT.

1 Introduction

In the current digital age, cryptocurrencies are applications that use virtual assets and cryptographic mechanisms to conduct e-commerce operations such as electronic payments or money transfers. Those payments can be carried out among accounts or wallets, independently of a central party [11]. Cryptocurrencies lead to some advantages like lower transaction fees, theft resistance and anonymous transactions. Bitcoin [26] is by far the most widely known and decentralised cryptocurrency to date, having its three underlying building blocks: transactions, blockchain and consensus protocol. Contrary to the traditional banking model, Bitcoin allows electronic financial operations in a decentralised Peer-to-Peer (P2P) network. Although Bitcoin was intended to achieve the security properties of privacy and anonymity by using pseudonyms, some analyses [30, 17] show that these security properties can be compromised, therefore information about the payers, payees and transactions can be revealed. Thus Bitcoin is only a pseudo-anonymous cryptocurrency.

Nonetheless, since its creation, Bitcoin has revolutionised the field of digital currency and motivated the invention of new cryptocurrencies, also known as alcoins. As an example, CryptoNote [34] was proposed to address the privacy weaknesses of Bitcoin, as it also offers a framework that can be extended by other cryptocurrencies such Bytecoin [7] and Monero [25]. CryptoNote uses *traceable ring signatures* [16] as a fundamental component to achieve true anonymity, where any member of the ring (or group) can create a signature, but it is infeasible by a verifier to identify the real signer. This type of signature hides information about the sender and receiver, and it also has a *linking tag* to prevent double spending coins. Further enhancements to this framework have resulted in an extended protocol called Ring Confidential Transactions "RingCT" [27]. The RingCT protocol uses three techniques: a new type of ring signature *Linkable Ring Signatures* [19], a *homomorphic commitment* and a *range proof*, to preserve the privacy of the sender and the receiver as well as the transaction amounts.

However, the security of this RingCT protocol relies on classical number-theory assumptions, such as the hardness of discrete logarithms [14]. As a consequence, this protocol will be vulnerable in the

event of powerful quantum computers [31]. This situation has motivated researchers in the area of post-quantum cryptography to construct secure approaches against quantum attacks. Among the alternatives, lattice-based cryptography has attracted attention due to its distinguishing features and robust security guarantees [24, 9].

To the best of our knowledge, the first *post-quantum RingCT* scheme using Lattice-based cryptography was proposed in [1]. However, this proposal is limited. Firstly, it only enables transfers from a single input wallet to a single output wallet (SISO). In the RingCT model, signatures are *one-time*, then if one needs to receive change after making a payment or transfer, a new output wallet is required, so this points out the importance of supporting multiple input and output wallets. Secondly, having more than one output wallet also introduces a new security problem like the negative output amount (or out-of-range) attack [6], where an adversary is capable of creating extra coins. This attack is addressed in the previous RingCT [27] by using a range proof technique; however, this technique is not post-quantum secure.

1.1 Contributions

- We construct the Lattice-based Ring Confidential Transactions (LRCT) for Multiple-Input and Multiple-Output wallets (MIMO). This construction is a generalisation of the SISO.LRCT scheme in [1] where we changed its underlying framework (L2RS signature) to be compatible. Our MIMO.LRCT inherits the post-quantum security guarantees, like the hardness of lattice mathematical assumptions as well as unconditional anonymity.
- We improve the MIMO.LRCT's security model, in particular, the balance and anonymity properties. We explicitly define a balance model that considers out-of-range attacks [6], and we prove the security of our protocol which previous RingCT's proposals [1, 33] did not address. User anonymity is only addressed in [33], while we include the analysis of both user anonymity and amount privacy.
- We show how to incorporate a lattice-based range proof into our MIMO.LRCT protocol, which was a missing ingredient in former proposals [1, 33]. To begin with, our protocol deals with the difficulties of the imperfection of lattice-based zero-knowledge proofs, Section 5.1 discusses more on this. In particular, range proofs follow the approach based on 1-of-2 OR-proofs, but our analysis shows that directly applying lattice-based OR-proofs from [12] does not provide soundness for the range proof. This argument leads us to carefully select the challenge space as we describe in Lemma 3. Although these challenges are smaller (in norm) than the ones used in the OR-proofs, they are still larger than the challenges in [18]. In this framework, we achieve lower soundness error than the previous lattice-based range proof [18]. We also provide a thorough concrete analysis of the MIMO.LRCT protocol by including this range proof analysis.
- We apply our concrete bounds to derive preliminary scheme parameters for regular RingCT transactions that support 64-bit amounts along with fewer Multiple Input and Output wallets. This analysis serves as a benchmark for future practical implementations.

The organisation of this work is as follows. Section 1.2 presents CryptoNote and RingCT protocols literature. After introducing the notation and concepts used in our work in Section 2, we define the MIMO.LRCT as well as its security model in Section 3. Section 4.1 involves the concrete construction of the homomorphic commitment and the MIMO.L2RS signature schemes, then Section 5 illustrates the construction of MIMO.LRCT. Section 6 and 7 point out the MIMO.LRCT's security and performance analyses, respectively.

1.2 Related Work

Evaluations [22, 28] of CryptoNote have discovered serious vulnerabilities which impact the privacy of the involved parties in the transactions. Therefore, the Ring Confidential Transactions RingCT [27] protocol was devised to address these issues. The RingCT extends the CryptoNote scheme by using a new class of linkable ring signature called *Multi-layered Linkable Spontaneous Anonymous Group Signature* (MLSAG) [19]. This signature is spontaneous (or *ad-hoc*), which removes the dependency of a trusted third party and group members are unaware of belonging to a determined group, thereby enhancing the *anonymity* property. It is also multilayered, meaning that it enables multiple input and output wallets in transactions. The security of RingCT is ameliorated by introducing the Confidential Transactions [23], which enables

amounts to be hidden by using the *Pedersen Commitment* [29] technique. This cryptographic primitive enables a party to commit to a chosen secret value while keeping it hidden to other parties, where this commitment can later be opened. Such a primitive offers homomorphic properties allowing parties to prove the account balance by computing homomorphically input and output accounts to show that their result is zero. RingCT added another verification mechanism for the committed output amounts which was called *range proof*, guaranteeing that this amount lies in a range of *non-negative* values and avoiding the creation of *free money*. Bulletproofs [6] is an efficient technique for this range preservation.

RingCT v2.0 [33] was later proposed. It provided sound security analysis of the (RingCT) protocol as well as improved the size of the signature by using one-way accumulators [4] along with signatures of knowledge "SoK" [8]. However, it requires a trusted setup for its accumulator to achieve the signature constant size. The first post-quantum RingCT protocol was proposed in [1], where the authors named it *Lattice RingCT v1.0*. This construction uses lattice-based cryptography to design a new *Linkable Ring Signature*, which is called *Lattice-based Linkable Ring Signature* (L2RS). The L2RS follows the well known Fiat-Shamir [15] transformation signature: *Bimodal Lattice Signature Scheme* (BLISS) [13], a practical and secure lattice-based signature scheme. The L2RS offers computational security as per the hardness of lattice assumptions for *unforgeability*, *linkability* and *non-slanderability*, it also achieves unconditional *anonymity*. However, the proposed *Lattice RingCT v1.0* showed no security definition or proofs, and transactions were restricted to Single Input and Single Output wallets.

2 Preliminaries

The polynomial ring $\mathcal{R} = \mathbb{Z}[x]/f(x)$, where $f(x) = x^n + 1$ with n being a power of 2. 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 $[-q/2, q/2]$. The challenge space $\mathcal{S}_{n,\kappa}$, is the set of all binary vectors of length n and weight κ. A hash function modeled as Random Oracle Model (ROM), H_1 with range $\mathcal{S}_{n,\kappa} \subseteq \mathcal{R}_{2q}$. When we use $x \leftarrow D$ for a distribution D, it means that x is sampled from D, and when we use $x \leftarrow S$ for a set S, it means that x is uniformly sampled at random from S. If we use $\mathbf{r}_{||}$, it means $\mathbf{r}$ is concatenated such $\left(\mathbf{r}^{(1)}, \ldots, \mathbf{r}^{(\theta)}\right)$ for any θ. The discrete Gaussian distribution over $\mathbb{Z}^m$ with standard deviation $\sigma \in \mathbb{R}$ and center at zero, is defined by $D^m_\sigma(\mathbf{x}) = \rho_\sigma(\mathbf{x})/\rho_\sigma(\mathbb{Z}^m)$, where ρ_σ is the m-dimensional Gaussian function $\rho_\sigma(\mathbf{x}) = \exp(-\|\mathbf{x}\|^2/(2\sigma^2))$. Vector transposition is denoted by $\mathbf{v}^T$. The hardness assumption of this work is the Module-SIS (Short Integer Solution) problem and is defined as follows.

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

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

Lemma 2 (Rejection Sampling-Gaussian Distribution).

Lemma 3. (Based on [5]) *Let $\mathcal{R} = \mathbb{Z}[X]/(X^n + 1)$ where $n > 1$ is a power of 2 and $0 < i, j < 2n - 1$. Then all the coefficients of $2(X^i - X^j)^{-1} \in \mathcal{R}$ are in $\{-1, 0, 1\}$. This implies that $\|2(X^i - X^j)^{-1}\| \leq \sqrt{n}$.*

Lemma 4. *For $a, b \in \mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$ the following relations hold $\|a\| \leq \sqrt{n}\|a\|_\infty$, $\|a \cdot b\| \leq \sqrt{n}\|a\|_\infty \cdot \|b\|_\infty$, $\|a \cdot b\|_\infty \leq \|a\| \cdot \|b\|$.*

Lemma 5 (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 $H(\mathbf{S}) = \mathbf{A} \cdot \mathbf{S} \in \mathcal{R}_q$, with $\mathbf{S} \in \mathsf{Dom}_{\mathbf{S}}$, where each function is indexed by $\mathbf{A} \in \mathcal{R}_q^{1\times(m-1)}$ and $\mathsf{Dom}_{\mathbf{S}} \subseteq \mathcal{R}_q^{1\times(m-1)}$ consists of vectors of $\mathcal{R}_q$ elements with coefficients in $\Gamma \triangleq (-2^\gamma, 2^\gamma)$. This is a universal hash family if for all $\mathbf{S} \neq \mathbf{S}'$, we have

$$\Pr\left[\mathbf{A} \cdot \mathbf{S} = \mathbf{A} \cdot \mathbf{S}'\right] = \frac{1}{|\mathcal{R}_q|}.$$

This is a universal hash family if there exists $1 \leq i \leq m-1$ such that $s_i - s_i'$ is invertible in $\mathcal{R}_q$ with $s_i, s_i' \in \Gamma^n$. This can be guaranteed by appropriate choice of q, e.g. as shown in ([21], Corollary 1.2), it is sufficient to use q such that $f(x) = x^n + 1$ factors into k irreducible factors modulo q and $2^\gamma < \frac{1}{\sqrt{k}} \cdot q^{1/k}$. We assume that $\mathcal{R}_q$ is chosen to satisfy this condition.

2.1 Homomorphic Commitment Definition

This is a cryptographic technique that is used to provide confidential transactions, in particular cryptocurrencies [27]. 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. The definition of such technique, which is based on [2], has three algorithms: (KeyGen, Com, Open), such that:

- $\mathsf{Pub\text{-}Params} \leftarrow \mathsf{KeyGen}(1^\lambda)$: A PPT algorithm that produces a public commitment parameter Pub-Params after receiving the security parameter (λ).
- $\mathsf{c} \leftarrow \mathsf{Com}$: A PPT algorithm that receives the Pub-Params, the randomness r and the message m. This algorithm generates the commitment c.
- $\mathrm{m}' \leftarrow \mathsf{Open}$: A PPT algorithm that receives the commitment c along with the randomness r, and it outputs m'. A valid commitment c is opened if $(\mathrm{m}' = \mathrm{m})$.

The security properties of this non-interactive homomorphic commitment scheme are defined as:

Definition 2 (Hiding). *This property ensures that the commitment* $\mathsf{Com}(\mathrm{m}, \mathsf{r})$ *does not leak information on* m*, that is, for any PPT adversary* $\mathcal{A}$*, it holds that:*

$$\left|\Pr\left[\mathcal{A}(\mathsf{c}_b) = b : \begin{array}{l}\mathsf{Pub\text{-}Params} \leftarrow \mathsf{KeyGen}(1^\lambda); \mathsf{r} \leftarrow \mathsf{RandGen}(\mathsf{Pub\text{-}Params});\\ (\mathrm{m}, \mathrm{m}') \leftarrow \mathcal{A}(\mathsf{Pub\text{-}Params}); b \leftarrow \{0,1\}; \mathsf{c}_b \leftarrow \mathsf{Com}(\mathsf{r}, \mathrm{m}_b)\end{array}\right] - \frac{1}{2}\right|,$$

is $negl(\lambda)$.

Definition 3 ($\beta-$Binding). *This property ensures that the commitment* $\mathsf{Com}(\mathrm{m}, \mathsf{r})$ *can only be opened in one way, that is, for any PPT adversary* $\mathcal{A}$*, it holds that:*

$$\Pr\left[\begin{array}{l}\mathsf{r} \neq \mathsf{r}' \wedge\\ \mathrm{m} \neq \mathrm{m}' \wedge\\ \mathsf{Com}(\mathrm{m}, \mathsf{r}) = \mathsf{Com}(\mathrm{m}', \mathsf{r}')\end{array} : \begin{array}{l}\mathsf{Pub\text{-}Params} \leftarrow \mathsf{KeyGen}(1^\lambda);\\ \mathsf{r} \leftarrow \mathsf{RandGen}(\mathsf{Pub\text{-}Params});\\ (\mathrm{m}, \mathsf{r}, \mathrm{m}', \mathsf{r}') \leftarrow \mathcal{A}(\mathsf{r})\end{array}\right] \leq negl(\lambda),$$

where $\|\mathsf{r}\|, \|\mathsf{r}'\| \leq \beta$.

2.2 Fiat-Shamir Non-Interactive Zero-Knowledge Proofs in the Random Oracle Model

Zero-knowledge proof of knowledge (ZKPoK) is a two party protocol between the *prover* and the *verifier*, which allows the *prover* to convince the *verifier* that he knows some information, without revealing anything about the secret apart from what the claim itself already reveals [5].

Definition 4. *Let be* $\mathcal{L} \subseteq \{0,1\}^*$ *the language that has witness relationship* $R \subseteq \{0,1\}^* \times \{0,1\}^*$ *if* $x \in \mathcal{L} \leftrightarrow \exists(x, w) \in R$*. We call* w *a witness for* $x \in \mathcal{L}$*. Let* $(\mathcal{P}, \mathcal{V})$ *be a two-party protocol where* $\mathcal{P}$ *(prover) and* $\mathcal{V}$ *(verifier) are PPT algorithms, and* $\mathcal{L}, \mathcal{L}'$ *be languages with witness relations* R, R' *with* $R \subset R'$*. Then* $(\mathcal{P}, \mathcal{V})$ *has a proof* σ *with completeness error* α*, public input* x *and private input* w*, if the following conditions are satisfied:*

- *The protocol uses a hash function H modeled as a random oracle which is called by both $\mathcal{P}$ and $\mathcal{V}$. This protocol has the following form: on input (x, w), $\mathcal{P}$ outputs a proof σ that is sent to $\mathcal{V}$. On input x, the verifier $\mathcal{V}$ accepts or rejects σ.*
- *Completeness: whenever $(x, w) \in R$, the honest verifier accepts the proof σ with probability at least $1 - \alpha$.*
- *Soundness: given a dishonest prover $\mathcal{A}$ with input x, it outputs a valid proof σ with non-negligible probability, then there there exists a PPT algorithm $\mathcal{E}$ (the knowledge extractor) that extracts a witness w' satisfying $(x, w') \in R'$.*
- *Special honest-verifier zero-knowledgeness (HVZK): there exists two PPT algorithms $\mathcal{S}$ (the simulator) and $\mathcal{S}_H$ (random oracle simulator) that take $x \in \mathcal{L}$, and output the proofs $\sigma_{sim} = \mathcal{S}(x)$ and $\mathcal{S}_H(x, \cdot)$ such that is computationally indistinguishable from $\sigma = \mathcal{P}(x, w)$ and $H(\cdot)$ generated by a real protocol.*

3 Ring Confidential Transaction Protocol (RCT)

The RCT protocol is defined based on the former RingCT 2.0 protocol in [33].

Definition 5 (Account or wallet). *A wallet has a public component "act" and a private component "ask". The act is composed of the user's pk (or a valid address) and the coin **cn**, while the ask is formed of the user's sk along with the coin-key **ck**.*

The RCT protocol has five PPT algorithms (RCT.Setup, RCT.KeyGen, RCT.Mint, RCT.Spend, RCT.Verify) as well as the correctness (RCT.Correctness). The RCT's algorithms are defined as follows:

- RCT.Setup: this PPT algorithm takes the security parameter λ and outputs the public parameters Pub-Params.
- RCT.KeyGen: this PPT algorithm uses the Pub-Params to produce a pair of keys, the public-key pk and the private-key sk.
- RCT.Mint: a PPT algorithm generating new coins by receiving Pub-Params and the amount \$. This algorithm outputs a coin **cn** and a coin-key **ck**.
- RCT.Spend: a PPT algorithm that receives the Pub-Params, a set of input wallets $\{IW_i\}_{i\in[w]}$ with w being the size of the ring, a user π's input wallets IW_π along with its set of secret keys K_π, a set of output addresses OA, some transaction string $\mu \in \{0,1\}^*$, the output amount \$ and the set of output wallets OW. Then, this algorithm outputs: the transaction $TX = (\mu, IW, OW)$, the signature sig and a set of transaction/serial numbers TN, which is used to prevent the double spending coins.
- RCT.Verify: a deterministic PPT algorithm that takes as input the Pub-Params, the signature sig, the TX, and the TN and verifies if the transaction was legitimately generated and outputs either: Accept or Reject.

Transaction Correctness requirements: RCT.Correctness ensures that an honest user (payer) is able to spend or transfer any of his accounts (wallets) into a group of destination accounts (payee), where this transaction is accepted with overwhelming probability by a verifier. Thus the correctness of RCT is guaranteed if for all PPT adversaries $\mathcal{A}$, it holds that:

$$\Pr\left[\begin{array}{l}\mathsf{LRCT.Verify}\big(TX, \mathsf{sig}, TN\big) = 1 : \\ \end{array}\begin{array}{l}\mathsf{Pub\text{-}Params} \leftarrow \mathsf{LRCT.Setup}(1^\lambda); \\ (\mu, IW, OA) \leftarrow \mathcal{A}(\mathsf{Pub\text{-}Params}, IW_\pi, K_\pi) \\ \text{with } (IW_\pi, K_\pi) \text{ as in Table 1;} \\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{LRCT.KeyGen}(\mathsf{Pub\text{-}Params}); \\ (\mathbf{cn}, \mathbf{ck}) \leftarrow \mathsf{LRCT.Mint}(\mathsf{Pub\text{-}Params}, \$); \\ (TX, \mathsf{sig}, TN) \leftarrow \mathsf{LRCT.Spend}(\mu, \\ \mathsf{Pub\text{-}Params}, IW_\pi, K_\pi, IW, OA, \$_{(out)}).\end{array}\right] = 1.$$

3.1 Oracles for adversaries

We now list all the adversarial oracles used in RCT, and we define them as:

- AddGen(i): on input a query number i, this oracle picks randomness τ_i, runs algorithm $\big(\mathsf{pk}_i, \mathsf{sk}_i\big) \leftarrow \mathsf{RCT.KeyGen}(\mathsf{Pub\text{-}Params}, \tau_i)$, and returns the public-key or one-time address pk_i.

- $\mathsf{ActGen}(i, \$_i)$: on input a query number i and an amount $\$_i$, it runs $(\mathbf{cn}_i, \mathbf{ck}_i) \leftarrow \mathsf{RCT.Mint}\big(\mathsf{Pub\text{-}Params}, \$_i\big)$. Then, ActGen adds i and the account $act_i = \big(\mathsf{pk}_i, \mathbf{cn}_i\big)$ to empty lists $\mathcal{I}$ and IW, respectively. ActGen outputs $(act_i, \mathbf{ck}_i)$ for the one-time address pk_i, where these addresses are added to a list $\mathcal{PK}$. The associated secret key with account act_i is defined as $ask_i \triangleq (\mathsf{sk}_i, \mathbf{ck}_i)$. With this ask_i, the challenger calls $\mathsf{MIMO.L2RS.SigGen}(\mathsf{sk}_i, \cdot, \cdot, \cdot)$ to determine the transaction number TN_i of act_i and adds it to a list $\mathcal{TN}$.
- $\mathsf{O\text{-}Spend}(\mu, IW, IW_\pi, OA, \$_{(out)}, \mathsf{Pub\text{-}Params})$: on input the transaction string μ, input accounts (wallets) IW containing IW_π and output addresses OA, it runs $(TX, \mathsf{sig}, TN) \leftarrow \mathsf{RCT.Spend}(\mu, K_\pi, IW, IW_\pi, OA, \$_{(out)}, \mathsf{Pub\text{-}Params})$ and adds the outputs to $\mathcal{T}$, where $IW_\pi \in IW$. We assume that at least one account/address in IW_π has not been corrupted. We define the set of transaction numbers in the $\mathsf{RCT.Spend}$ queries as $\mathcal{TN}^*$.
- $\mathsf{Corrupt}(i)$: on input query number $i \in \mathcal{I}$, uses account key ask_i to determine the transaction/serial number TN_i of account act_i with address pk_i, then adds TN_i and $(TN_i, \$_i)$ to lists $\mathcal{C}$ and $\mathcal{B}$ respectively and finally returns τ_i.

3.2 Threat Model

The protocol RCT is modeled in terms of *balance*, *anonymity* and *non-slanderability* for security analysis purposes, which are defined as follows.

Definition 6 (Balance). *This property requires that any adversary cannot spend any account without her control and cannot spend her own accounts with a larger output amount. This security property is guaranteed if for all PPT adversaries $\mathcal{A}$, it holds that:*

$$\Pr\left[\mathcal{A}\ wins: \begin{array}{l} \mathit{Pub\text{-}Params} \leftarrow \mathit{LRCT.Setup}(1^\lambda); \\ (\{IW_i^{(k)}\}_{i\in[w],k\in[N_{in}]}, \mathcal{T}) \leftarrow \mathcal{A}^{\mathit{AddGen},\mathit{ActGen},\mathit{O\text{-}Spend},\mathit{Corrupt}}(\mathit{Pub\text{-}Params}) \end{array}\right],$$

is $negl(\lambda)$, where adversaries' oracles are defined in Section 3.1. We have that $IW_i^{(k)} = \{\mathsf{pk}_{(in),i}^{(k)}, \boldsymbol{cn}_{(in),i}^{(k)}\}_{i\in[w],k\in[N_{in}]}$ and $\mathcal{T} = (TX, \mathsf{sig}, TN)$. These spends can be transferred to the challenger with the account address $\mathsf{pk}_{(out)} = \{\mathsf{pk}_{(out)}^{(j)}\}_{j\in[N_{out}]}$, where we assume not all of them are corrupted, and at least one of them is honest. This $\mathsf{pk}_{(out)}$ has been created by the AddGen oracle, so the challenger knows all balances of the spent accounts and output accounts involved in the adversarial spends $\mathcal{T}$. This means that $TX = (\mu, IW, OW)$ with $OW = \{OW^{(j)}\}_{j\in[N_{out}]} = \{\mathsf{pk}_{(out)}^{(j)}, \boldsymbol{cn}_{(out)}^{(j)}\}_{j\in[N_{out}]}$ being the output wallet corresponding to output account $\mathsf{pk}_{(out)}$. The adversary $\mathcal{A}$ wins this experiment if her outputs satisfy the following conditions:

1. $\mathit{RCT.Verify}(TX, \mathsf{sig}, TN) = 1$.
2. $\sum_{k\in E_{(in)}} \$_{(in),\pi}^{(k)} < \sum_{j\in G_{(out)}} \$_{(out)}^{(j)}$, *where we let $\pi \in [w]$ s.t. π's row $\{\mathsf{pk}_{(in),\pi}^{(1)}, \ldots, \mathsf{pk}_{(in),\pi}^{(N_{in})}\}$ are the ones that have $\{TN_\pi^{(1)}, \ldots, TN_\pi^{(N_{in})}\}$ which are found in ActGen, $E_{(in)}$ are the corrupted inputs, and $G_{(out)}$ are the not corrupted outputs in $\mathcal{T}$. For each $TN^{(k)}$ let $\$_{(in)}^{(k)}$ be the amount queried to ActGen at the index query i such $TN \subseteq \mathcal{TN}$. $\$_{(in)}^{(k)}$ is also defined as equal to zero if $IW_i^{(k)}$ is equal to some input wallet IW queried to $\mathsf{O\text{-}Spend}$, using same TN, which means that $IW_i^{(k)}$ has been spent.*
3. *TN cannot be the output of previous queries to the $\mathsf{O\text{-}Spend}(\cdot)$ (i.e. $TN \cap \mathcal{TN}^* = \emptyset$).*
4. *pk_π is queried to $\mathsf{O\text{-}Spend}$ oracle only once.*
5. *$PK \subseteq \mathcal{PK}$, where $PK \triangleq \{\mathsf{pk}_{(in),i}^{(k)}\}_{i\in[w],k\in[N_{in}]}$.*

Our extended *anonymity* property captures two types of attacks (compared to one type in [33]) that depend on the adversary's choices for users $\pi_0, \pi_1 \in [w]$ and output amounts $\$_{(out),0}, \$_{(out),1}$. It starts with the user anonymity attack where the adversary selects $\pi_0 \neq \pi_1$ with $\$_{(out),0} = \$_{(out),1}$, while in the amount privacy attack this adversary chooses $\pi_0 = \pi_1$ with $\$_{(out),0} \neq \$_{(out),1}$. We formally define this property as:

Definition 7 (Anonymity). *This property requires that two proofs of knowledge with the same transaction string μ, input accounts IW, output addresses OA, distinct both output amounts $(\$_{(out),0}, \$_{(out),1})$ and spent accounts $IW_{\pi_0}, IW_{\pi_1} \in IW$ are indistinguishable, meaning that the spender's accounts and amounts are successfully hidden among all the honestly generated accounts. The protocol RCT is called anonymous if for all PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, it holds that:*

$$\left| \Pr \left[b' = b : \begin{array}{l} \mathit{Pub\text{-}Params} \leftarrow \mathit{Setup}(1^\lambda); \\ (\mu, IW_{\pi_0}, IW_{\pi_1}, IW, OA, \$_{(out),0}, \$_{(out),1}) \leftarrow \mathcal{A}_1^{\mathit{AddGen}, \mathit{ActGen}, \mathit{O\text{-}Spend}, \mathit{Corrupt}}(\mathit{Pub\text{-}Params}); \\ b \leftarrow \{0,1\}; \\ (TX^*, \mathsf{sig}_b^*, TN^*) \leftarrow \mathit{LRCT.Spend}(\mu, K_{\pi_b}, IW_{\pi_b}, IW, OA, \$_{(out),b}, \mathit{Pub\text{-}Params}); \\ b' \leftarrow \mathcal{A}_2^{\mathit{O\text{-}Spend}, \mathit{Corrupt}}(\mathit{Pub\text{-}Params}, (TX^*, \mathsf{sig}_b^*, TN^*)) \end{array} \right] - \frac{1}{2} \right|,$$

is $negl(\lambda)$, where adversaries' oracles are defined in Section 3.1. In addition, the following restrictions should be satisfied:

1. *For all $b \in \{0,1\}$, any account in IW_{π_i} has not been corrupted.*
2. *Any query in the form of $(\cdot, IW_\pi, \cdot, \cdot)$, such that $IW_\pi \cap IW_{\pi_i} \neq \emptyset$ has not been issued to O-Spend oracle.*

Definition 8 (Non-Slanderability). *This property requires that a malicious user cannot slander any honest user after observing an honestly generated spending. That is, it is infeasible for any malicious user to produce a valid spending that shares at least one transaction/serial number with a previously generated honest spending. The protocol RCT is non-slanderable if for all PPT adversaries $\mathcal{A}$, it holds that:*

$$\Pr \left[\mathcal{A} \text{ wins} : \begin{array}{l} \mathit{Pub\text{-}Params} \leftarrow \mathit{LRCT.Setup}(1^\lambda); \\ \big((TX, \mathsf{sig}, TN), (TX^*, \mathsf{sig}^*, TN^*)\big) \leftarrow \mathcal{A}^{\mathit{AddGen}, \mathit{ActGen}, \mathit{O\text{-}Spend}, \mathit{Corrupt}}(\mathit{Pub\text{-}Params}) \end{array} \right],$$

is $negl(\lambda)$, where adversaries' oracles are defined in Section 3.1, and (TX, sig, TN) is one output of the oracle O-Spend for some (μ, IW_π, IW, OA). We say $\mathcal{A}$ succeeds if the output satisfies:

1. $\mathit{RCT.Verify}(TX^*, \mathsf{sig}^*, TN^*) = 1$,
2. $(TX^*, \mathsf{sig}^*, TN^*) \notin \mathcal{T}$, *and*
3. $TN \cap \mathcal{C} = \emptyset$ *but* $TN \cap TN^* \neq \emptyset$.

4 Building Blocks Construction

In this section, we summarize the underlying lattice-based primitives that are used in the construction of MIMO.LRCT. This includes a lattice-based homomorphic commitment scheme and a MIMO version of L2RS signatures, specified in Appendix C.

4.1 Lattice-based Commitment Construction

The MIMO.LRCT protocol requires a non-interactive homomorphic commitment (Com) as an essential primitive. We construct the three algorithms: (KeyGen, Com, Open), using the MIMO.L2RS scheme (Appendix C):

- $\mathbf{A} \leftarrow \mathsf{KeyGen}(1^\lambda)$: A PPT algorithm that produces a public commitment parameter $\mathbf{A} \in \mathcal{R}_q^{2\times(m-1)}$ after receiving the security parameter (λ). In doing so, we call the MIMO.L2RS.Setup (Appendix C) to generate $\mathbf{A} \in \mathcal{R}_q^{2\times(m-1)}$.
- $\mathbf{c} \leftarrow \mathsf{Com}_{\mathbf{A}}(\mathrm{m}, \mathsf{sk})$: A PPT algorithm that receives the public parameter $\mathbf{A}$ (from KeyGen), the randomness sk and the message formed as $\overline{\mathrm{m}} = (0, \mathrm{m})^T \in \mathcal{R}_q^{1\times 2}$. This algorithm generates the commitment $\mathbf{c} \in \mathcal{R}_q^2$. The randomness $\mathsf{sk} \in \mathsf{Dom}_{\mathsf{sk}} \subseteq \mathcal{R}_q^{(m-1)\times 1}$ with every component chosen uniformly and independently with coefficients in $(-2^\gamma, 2^\gamma)$, is produced by calling the MIMO.L2RS.KeyGen (Algorithm 1) and the message $\mathrm{m} \in \mathsf{Dom}_{\mathrm{m}} = \mathcal{R}_q$, then the commitment $\mathbf{c} = \mathsf{Com}_{\mathbf{A}}(\mathrm{m}, \mathsf{sk}) = \mathbf{A} \cdot \mathsf{sk} + \overline{\mathrm{m}} \in \mathcal{R}_q^2$.

– $\mathrm{m}' \leftarrow \mathsf{Open}_{\mathbf{A}}(\mathbf{c}, \mathsf{sk})$: A PPT algorithm receiving commitment $\mathbf{c}$ and randomness sk, and it outputs m'. A valid $\mathbf{c}$ is opened if $(\mathrm{m}' = \mathrm{m})$. This algorithm computes $\overline{\mathrm{m}}' = (0, \mathrm{m}')^T = \mathsf{Open}_{\mathbf{A}}(\mathbf{c}, \mathsf{sk}) = \mathbf{c} - \mathbf{A} \cdot \mathsf{sk}$.

Remark 2. $\mathsf{Dom}_{\mathrm{m}}$ is full and not a small subset $\mathcal{R}_q$, whereas $\mathsf{Dom}_{\mathsf{sk}}$ is only a small domain versus q. These adjustments help us to obtain better parameters than SISO.LRCT and security against out-of-range attacks.

This homomorphic commitment scheme performs the following operations:

$$\begin{aligned} \mathsf{Com}_{\mathbf{A}}(\mathrm{m}, \mathsf{sk}) \boxed{\pm} \mathsf{Com}_{\mathbf{A}}(\mathrm{m}', \mathsf{sk}') &\triangleq \mathsf{Com}_{\mathbf{A}}(\mathrm{m}, \mathsf{sk}) \pm \mathsf{Com}_{\mathbf{A}}(\mathrm{m}', \mathsf{sk}') \bmod q \\ &\triangleq \mathsf{Com}_{\mathbf{A}}(\mathrm{m} \pm \mathrm{m}', \mathsf{sk} \pm \mathsf{sk}') \bmod q. \end{aligned} \tag{1}$$

Theorem 1 (Hiding). *If* $\frac{1}{2}\sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$ *is negligible in security parameter* λ*, then the above* Com *is information theoretically hiding.*

Proof. Suppose that a PPT adversary $\mathcal{A}$ is given two messages $(\mathrm{m}, \mathrm{m}')$, the public parameter $\mathbf{A} \in \mathcal{R}_q^{2\times(m-1)}$ and the randomness sk. A bit b is chosen uniformly at random from $b = \{0, 1\}$, and the commitment is generated as $\mathbf{c}_b \leftarrow \mathsf{Com}_{\mathbf{A}}(\mathrm{m}_b, \mathsf{sk}) = \mathbf{A} \cdot \mathsf{sk} + \overline{\mathrm{m}}_b$. This adversary $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$, where $\mathcal{A}$ succeeds in breaking the hiding property when $(b = b')$. We now analyze the generated commitment $\mathbf{c}_b$ with a uniformly random element from $\mathcal{R}_q^2$. We know that sk is chosen small with coefficients in $(-2^\gamma, 2^\gamma)$. By applying the Leftover Hash Lemma (Lemma 5), we argue that the statistical distance between the distribution of $\mathbf{c}$ and the uniform distribution on $\mathcal{R}_q^2$ is at most $\left(\frac{1}{2} \cdot \sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}\right)$, which is negligible in (λ). □

Theorem 2 ($\beta-$Binding). *The described Commitment Scheme is computationally $\beta-$binding if the* $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,2\beta}$ *problem is hard.*

Proof. Suppose that an adversary $\mathcal{A}$ generates $(\mathbf{c}, \mathsf{sk}, \mathsf{sk}')$ such that $\overline{\mathrm{m}} = \mathsf{Open}_{\mathbf{A}}(\mathbf{c}, \mathsf{sk})$ and $\overline{\mathrm{m}}' = \mathsf{Open}_{\mathbf{A}}(\mathbf{c}, \mathsf{sk}')$ with $\overline{\mathrm{m}} = (0, \mathrm{m})^T \in \mathcal{R}_q^{1\times 2}$ and $\overline{\mathrm{m}}' = (0, \mathrm{m}')^T \in \mathcal{R}_q^{1\times 2}$ being valid messages and $\mathrm{m} \neq \mathrm{m}'$. Using the Open algorithm, we have $\mathbf{A} \cdot (\mathsf{sk} - \mathsf{sk}') = (\overline{\mathrm{m}} - \overline{\mathrm{m}}') = (0, \mathrm{m} - \mathrm{m}')^T \neq 0$, where we find a small non-zero vector $\mathbf{v} = \left(\mathsf{sk} - \mathsf{sk}'\right)^T$ with respect to the first row $\mathbf{A}_1$ of the public commitment parameter $\mathbf{A}$, such that $\mathbf{A}_1 \cdot \mathbf{v} = 0 \bmod q$, with $\|\mathbf{v}\| \leq 2\beta$. Therefore, this vector $\mathbf{v}$ gives a solution to the $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,2\beta}$ problem. □

4.2 Multiple-Input Multiple-Output Wallets L2RS (MIMO.L2RS)

We adapt all the notations from [1] into our MIMO.L2RS. The MIMO.L2RS signs a signature for multiple wallets, which means that it signs N_{in} L2RS signatures in parallel. This MIMO.L2RS is an extension of the single-input and single-output proposal from [1]. In such extension, we needed to modify the Lattice-based Linkable Ring Signature (L2RS) to be capable of signing multiple wallets. Precisely, we adjusted the key generation, the signature generation and the verification algorithms to sign the total number of input wallets that a user wants to transfer to some output wallets. We call these algorithms: MIMO.L2RS.KeyGen, MIMO.L2RS.SigGen and MIMO.L2RS.SigVer, and we describe them in Algorithms 1 2 and 3, respectively.

Algorithm 1 MIMO.L2RS.KeyGen - Key-pair Generation $(\mathbf{a}, \mathbf{S})$

Input: Pub-Param: $\mathbf{A} \in \mathcal{R}_q^{2\times(m-1)}$.
Output: $(\mathbf{a}, \mathbf{S})$, being the public-key and the private-key, respectively.
1: **procedure** MIMO.L2RS.KeyGen($\mathbf{A}$)
2: Let $\mathbf{S}^T = (\mathbf{s}_1, \ldots, \mathbf{s}_{m-1}) \in \mathcal{R}_q^{1\times(m-1)}$, where $\mathbf{s}_i \leftarrow (-2^\gamma, 2^\gamma)^n$, for $1 \leq i \leq m-1$
3: Compute $\mathbf{a} = (\mathbf{a}_1, \mathbf{a}_2)^T = \mathbf{A} \cdot \mathbf{S} \bmod q \in \mathcal{R}_q^2$.
4: **return** $(\mathbf{a}, \mathbf{S})$.

Algorithm 2 MIMO.L2RS.SigGen - MIMO Signature Generation $\sigma_{L'}(\mu)$

Input: $\{\mathbf{S}^{(k)}_{(in),\pi}\}_{k\in[N_{in}+1]}, \mu$, L' as in (4), and Pub-Params.

Output: $\sigma_{L'}(\mu) = \Big(\mathbf{c}_1, \{\mathbf{t}_1^{(k)}, \ldots, \mathbf{t}_w^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{h}^{(k)}\}_{k\in[N_{in}]}\Big)$

procedure MIMO.L2RS.SigGen($\mathbf{S}^{(k)}_{(in),\pi}, \mu, L'$, Pub-Params)
  for $(1 \le k \le N_{in}+1)$ **do**
    Set $\mathbf{H}^{(k)}_{2q} = \big(2\cdot\mathbf{H}, -2\cdot\mathbf{h}^{(k)} + \mathbf{q}\big) \in \mathcal{R}^{2\times m}_{2q}$, where $\mathbf{h}^{(k)} = \mathbf{H}\cdot\mathbf{S}^{(k)}_{(in),\pi} \in \mathcal{R}^2_q$.
    Call MIMO.L2RS.Lift$(\mathbf{A}, \mathbf{a}^{(k)}_{(in),\pi})$ to obtain $\mathbf{A}^{(k)}_{2q,\pi} = (2\cdot\mathbf{A}, -2\cdot\mathbf{a}^{(k)}_{(in),\pi} + \mathbf{q}) \in \mathcal{R}^{2\times m}_{2q}$.
    Let $\mathbf{u}^{(k)} = (u_1, \ldots, u_m)^T$, where $u_i \leftarrow D^n_\sigma$, for $1 \le i \le m$.
  Compute $\mathbf{c}_{\pi+1} = H_1\Big(L', \{\mathbf{H}^{(k)}_{2q}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}^{(k)}_{2q,\pi}\cdot\mathbf{u}^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{H}^{(k)}_{2q}\cdot\mathbf{u}^{(k)}\}_{k\in[N_{in}+1]}\Big)$.
  for $(i = \pi+1, \pi+2, \ldots, w, 1, 2, \ldots, \pi-1)$ **do**
    for $(1 \le k \le N_{in}+1)$ **do**
      Call MIMO.L2RS.Lift$(\mathbf{A}, \mathbf{a}^{(k)}_{(in),i})$ to obtain $\mathbf{A}^{(k)}_{2q,i} = (2\cdot\mathbf{A}, -2\cdot\mathbf{a}^{(k)}_{(in),i} + \mathbf{q}) \in \mathcal{R}^{2\times m}_{2q}$.
      Let $\mathbf{t}^{(k)}_i = (t_{i,1}, \ldots, t_{i,m})^T$, where $t_{i,j} \leftarrow D^n_\sigma$, for $1 \le j \le m$.
    Compute $\mathbf{c}_{i+1} = H_1\Big(L', \{\mathbf{H}^{(k)}_{2q}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}^{(k)}_{2q,i}\cdot\mathbf{t}^{(k)}_i + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}, \{\mathbf{H}^{(k)}_{2q}\cdot\mathbf{t}^{(k)}_i + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}\Big)$.
  for $(1 \le k \le N_{in}+1)$ **do**
    Choose $b^{(k)} \leftarrow \{0,1\}$.
    Let $\mathbf{t}^{(k)}_\pi \leftarrow \mathbf{u}^{(k)} + \mathbf{S}^{(k)}_{2q,\pi}\cdot\mathbf{c}_\pi\cdot(-1)^{b^{(k)}}$, where $\mathbf{S}^{(k)}_{2q,\pi} = [(\mathbf{S}^{(k)}_\pi)^T, 1]^T$.
    Continue with prob. $\left(M\exp\left(-\frac{\|\mathbf{S}^{(k)}_{2q,\pi}\cdot\mathbf{c}_\pi\|^2}{2\sigma^2}\right)\cosh\left(\frac{\langle\mathbf{t}^{(k)}_\pi, \mathbf{S}^{(k)}_{2q,\pi}\cdot\mathbf{c}_\pi\rangle}{\sigma^2}\right)\right)^{-1}$ otherwise **Restart**.
  return $\sigma_{L'}(\mu) = \Big(\mathbf{c}_1, \{\mathbf{t}_1^{(k)}, \ldots, \mathbf{t}_w^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{h}^{(k)}\}_{k\in[N_{in}]}\Big)$.

Algorithm 3 MIMO.L2RS.SigVer - MIMO Signature Verification

Input: $\sigma_{L'}(\mu)$ as in (7), L' as in (4), μ, and Pub-Params.
Output: Accept or Reject

procedure MIMO.L2RS.SigVer($\sigma_{L'}(\mu)$, L', Pub-Params)
  for $(1 \le k \le N_{in}+1)$ **do**
    if $\mathbf{H}^{(k)}_{2q} = \big(2\cdot\mathbf{H}, -2\cdot\mathbf{h}^{(k)} + \mathbf{q}\big) \in \mathcal{R}^{2\times m}_{2q}$ **then** Continue
  for $(i = 1, \ldots, w)$ **do**
    for $(1 \le k \le N_{in}+1)$ **do**
      Call MIMO.L2RS.Lift$(\mathbf{A}, \mathbf{a}^{(k)}_{(in),i})$ to obtain $\mathbf{A}^{(k)}_{2q,i} = (2\cdot\mathbf{A}, -2\cdot\mathbf{a}^{(k)}_i + \mathbf{q}) \in \mathcal{R}^{2\times m}_{2q}$.
      if $\mathbf{c}_{i+1} = H_1\Big(L', \{\mathbf{H}^{(k)}_{2q}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}^{(k)}_{2q,i}\cdot\mathbf{t}^{(k)}_i + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}, \{\mathbf{H}^{(k)}_{2q}\cdot\mathbf{t}^{(k)}_i + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}\Big)$ **then** Continue
      else if $\|\mathbf{t}^{(k)}_i\|_2 \le \beta_v$ (the acceptance bound based on [13]) **then** Continue
      else if $\|\mathbf{t}^{(k)}_i\|_\infty < q/4$ **then** Continue
  if $\mathbf{c}_1 = H_1\Big(L', \{\mathbf{H}^{(k)}_{2q}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}^{(k)}_{2q,w}\cdot\mathbf{t}^{(k)}_i + \mathbf{q}\cdot\mathbf{c}_w\}_{k\in[N_{in}+1]}, \{\mathbf{h}^{(k)}_{2q}\cdot\mathbf{t}^{(k)}_w + \mathbf{q}\cdot\mathbf{c}_w\}_{k\in[N_{in}+1]}\Big)$ **then** Accept
  else Reject
  return Accept or Reject

4.3 MIMO.L2RS security properties

The security properties of the MIMO.L2RS are inherited from the L2RS' security analysis. By appropriately modifying these analysis, we can obtain the same results for unforgeability, anonymity, linkability and non-slanderability, which are shown in Theorems (2, 3, 4, 5 from [1]), respectively. The following proposition summarises these inherited properties:

Proposition 1. *If* $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,\beta}$ *problem (with* $\beta = 2\beta_v$*) is hard and* $\sqrt{\frac{q^{4n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$ *is negligible in n, then the MIMO.L2RS achieves one-time unforgeability, anonymity, linkability and non-slanderability as in Definitions (3, 4, 5, 6 from [1]).*

We also use the MIMO.L2RS signature scheme as a Proof of Knowledge (PoK) to accomplish, in part, the MIMO.LRCT's balance property. This proof is formalised, namely as:

Proposition 2. *The MIMO.L2RS.SigGen and MIMO.L2RS.SigVer which are described in Algorithms 2 and 3, respectively, are a Fiat-Shamir Non-Interactive Proof of Knowledge in the Random Oracle Model (Section 2.2) for the relations* R_{PoK} *and* R'_{PoK} *that we represent as:*

$$R_{PoK} \triangleq \left\{ \begin{array}{c} \{\mathbf{a}^{(k)}_{(in),i}, \mathbf{cn}^{(k)}_{(in),i}, \mathbf{cn}^{(j)}_{(out)}, \mu\}; \{\mathbf{S}^{(k)}_{(in),i}, \mathbf{ck}^{(k)}_{(in),i}, \mathbf{ck}^{(j)}_{(out)}, \$_{in}, \$_{out}\} : \\ \exists i \in [w] \ s.t. \ \mathbf{a}^{(N_{in}+1)}_{(in),i} = \mathsf{Com}_{\mathbf{A}}(0, \mathbf{S}^{(N_{in}+1)}_{(in),i}); \|\mathbf{S}^{(N_{in}+1)}_{(in),i}\| \leq \beta_{wit} \end{array} \right\}$$

$$R'_{PoK} \triangleq \left\{ \begin{array}{c} \{\mathbf{a}^{(k)}_{(in),i}, \mathbf{cn}^{(k)}_{(in),i}, \mathbf{cn}^{(j)}_{(out)}, \mu'\}; \{\mathbf{S}^{(k)}_{(in),i}, \mathbf{ck}^{(k)}_{(in),i}, \mathbf{ck}^{(j)}_{(out)}, \$_{in}, \$_{out}\} : \\ \exists z \in [w] \ s.t. \ \mathbf{v}^{(N_{in}+1)}_{z} = (\mathbf{v}^{(N_{in}+1)}_{z,(1)}, \mathbf{v}^{(N_{in}+1)}_{z,(2)})^T; \\ \mathbf{a}^{(N_{in}+1)}_{(in),z} \cdot \mathbf{v}^{N_{in}+1}_{z,(2)} = \mathsf{Com}_{\mathbf{A}}(0, \mathbf{v}^{(N_{in}+1)}_{z,(1)}); \left\|\mathbf{v}^{(N_{in}+1)}_{z}\right\| \leq \beta'_{wit} \end{array} \right\}$$

where $\beta_{wit} = 3 \cdot 2^{\gamma}$ *is said to be the honest prover's witness norm and* $\beta'_{wit} = 2 \cdot \beta_v$ *being the extracted malicious prover's witness norm.* β_v *is the acceptance bound of* $\boldsymbol{t}$ *from Algorithm 3 and* $\mathbf{a}^{(N_{in}+1)}_{(in),i}$ *is defined in* (5).

Proof.
Completeness: Since MIMO.L2RS runs parallel L2RS signatures, we said that the MIMO.L2RS's correctness (Appendix D.3) allows to achieve completeness in the MIMO.L2RS signature scheme.
Soundness: We show that for all PPT adversaries $\mathcal{A}$ of MIMO.L2RS, there is a PPT algorithm Ext, which extracts a valid witness of MIMO.L2RS. We perform a first run $(\mathbf{c}_i, \ldots, \mathbf{c}_w) \leftarrow \mathcal{S}_{n,\kappa}$ where we assume that $\mathbf{c}_i = H_1\Big(L', \big\{\mathbf{H}^{(k)}_{2q}\big\}_{k\in[N_{in}+1]}, \mu, \big\{\mathbf{A}^{(k)}_{2q,i-1} \cdot \mathbf{t}^{(k)}_{i-1} + \mathbf{q} \cdot \mathbf{c}_{i-1}\big\}_{k\in[N_{in}+1]}, \big\{\mathbf{H}^{(k)}_{2q} \cdot \mathbf{t}^{(k)}_{i-1} + \mathbf{q} \cdot \mathbf{c}_{i-1}\big\}_{k\in[N_{in}+1]}\Big)$ was a response to a random oracle H_1 (collision resistance) query made by $\mathcal{A}$. When $\mathcal{A}$ rewinds (second run) by responding with $\mathbf{c}_i \neq \mathbf{c}'_i$, we obtain another proof $(\mathbf{t}'^{(k)}_1, \ldots, \mathbf{t}'^{(k)}_w)$ and the corresponding hash values $(\mathbf{c}'_i, \ldots, \mathbf{c}'_w)$. Then, we verify around the ring signature loop (going backwards) to find a collision in the input of H_1, so for $k = N_{in}+1$, such that $\mathbf{A}^{(N_{in}+1)}_{2q,i-1} \cdot \mathbf{t}^{(N_{in}+1)}_{i-1} + \mathbf{q} \cdot \mathbf{c}_{i-1} = \mathbf{A}^{(N_{in}+1)}_{2q,i-1} \cdot \mathbf{t}'^{(N_{in}+1)}_{i-1} + \mathbf{q} \cdot \mathbf{c}'_{i-1} \bmod 2q$. In each stage, we analyze two cases. If $\mathbf{c}_{i-1} \neq \mathbf{c}'_{i-1}$ (case 1), then we use this collision to extract a witness; otherwise, if $\mathbf{c}_{i-1} = \mathbf{c}'_{i-1}$ (case 2) then we move backwards (-1) until the first case holds. Once this condition is met, we set the index $z = i - x$ where x is the number that decreases if case 2 holds. Subsequently, the following equality is built based on this collision, we said that $\mathbf{A}^{(N_{in}+1)}_{2q,z} \cdot \mathbf{t}^{(N_{in}+1)}_{z} + \mathbf{q} \cdot \mathbf{c}_z = \mathbf{A}^{(N_{in}+1)}_{2q,z} \cdot \mathbf{t}'^{(N_{in}+1)}_{z} + \mathbf{q} \cdot \mathbf{c}'_z \bmod 2q$ with $\mathbf{c}_{z+1} = \mathbf{c}'_{z+1}$. We reorganise this equality as $\mathbf{A}^{(N_{in}+1)}_{2q,z} \cdot (\mathbf{t}^{(N_{in}+1)}_{z} - \mathbf{t}'^{(N_{in}+1)}_{z}) = \mathbf{q} \cdot (\mathbf{c}_z - \mathbf{c}'_z) \bmod 2q$, when this is reduced mod$q$, we have $\mathbf{A}^{(N_{in}+1)}_{2q,z} \cdot (\mathbf{t}^{(N_{in}+1)}_{z} - \mathbf{t}'^{(N_{in}+1)}_{z}) = 0 \bmod q$. Since $\mathbf{c}_z - \mathbf{c}'_z \neq 0 \bmod 2$, so we have $\mathbf{t}^{(N_{in}+1)}_{z} - \mathbf{t}'^{(N_{in}+1)}_{z} \neq 0 \bmod 2q$ where $\|\mathbf{t}^{(N_{in}+1)}_{z} - \mathbf{t}'^{(N_{in}+1)}_{z}\|_{\infty} < q/2$. By reducing mod$q$, we find a small non-zero vector $\mathbf{v}^{(N_{in}+1)}_{z} \triangleq \mathbf{t}^{(N_{in}+1)}_{z} - \mathbf{t}'^{(N_{in}+1)}_{z} \neq 0 \bmod q$ with $\|\mathbf{v}^{(N_{in}+1)}_{z}\| \leq 2 \cdot \beta_{wit}$. This $\mathbf{v}^{(N_{in}+1)}_{z}$ will compute $\mathbf{A}^{(N_{in}+1)}_{2q,z} \cdot \mathbf{v}^{(N_{in}+1)}_{z} = 0 \bmod q$. Since $\mathbf{A}^{(N_{in}+1)}_{2q,z} \bmod q = 2 \cdot (\mathbf{A}, -\mathbf{a}) \bmod q$, we have $2 \cdot (\mathbf{A}, -\mathbf{a}^{(N_{in}+1)}_{(in),z}) \cdot \mathbf{v}^{(N_{in}+1)}_{z} = 0 \bmod q$, this implies that $(\mathbf{A}, -\mathbf{a}^{(N_{in}+1)}_{(in),z}) \cdot \mathbf{v}^{(N_{in}+1)}_{z} = 0 \bmod q$, since q is odd. Then, we consider $\mathbf{v}^{(N_{in}+1)}_{z} = (\mathbf{v}^{(N_{in}+1)}_{z,(1)}, \mathbf{v}^{(N_{in}+1)}_{z,(2)})^T$ where $\mathbf{a}^{(N_{in}+1)}_{(in),z} \cdot \mathbf{v}^{(N_{in}+1)}_{z,(2)} = \mathbf{A} \cdot \mathbf{v}^{(N_{in}+1)}_{z,(1)} \bmod q$.

Finally, we extract the witness as $\mathbf{a}_{(in),z}^{(N_{in}+1)} \cdot \mathbf{v}_{z,(2)}^{(N_{in}+1)} = \mathsf{Com}_{\mathbf{A}}\big(0, \mathbf{v}_{z,(1)}^{(N_{in}+1)}\big)$ with $(\mathbf{v}_{z,(1)}^{(N_{in}+1)}, \mathbf{v}_{z,(2)}^{(N_{in}+1)})^T \neq 0 \bmod q$.

HVZK: This property is guaranteed by the MIMO.L2RS's anonymity (in Theorem 10) as well as the hiding property of the homomorphic commitment scheme which was proved in Theorem 1. □

5 MIMO Lattice-based RingCT Construction

In this section, we construct the MIMO Lattice-based RingCT (MIMO.LRCT) protocol (Table 1 shows the MIMO.LRCT's notations), where one is allowed to have multiple (IW) and to spend them into multiple (OW). Furthermore, two sub-protocols are needed to support the MIMO.LRCT's threat model, which are: MIMO.L2RS security properties (subsection 4.3) and range preservation (subsection 5.1). The

Table 1: Notation of the Lattice RingCT v2.0

Notation	Description
act	Account or Wallet "Public part" = $\big(\mathsf{pk}, \mathbf{cn}\big) \in \mathcal{R}_q^2 \times \mathcal{R}_q^2$.
ask	Account or Wallet "Private part" = $\big(\mathsf{sk}, \mathbf{ck}\big) \in \mathcal{R}_q^2 \times \mathcal{R}_q^2$.
$\mathcal{S}_{n,\kappa}$	Binary vectors of length n of weight κ.
\$	Amount $\in \mathcal{S}_{n,\kappa}$.
$\$_{(in)}$	Group of input amounts $\$_{(in)}^{(k)}$ for $k \in [N_{in}]$.
$\$_{(out)}$	Group of output amounts $\$_{(out)}^{(j)}$ for $j \in [N_{out}]$.
$\ell_\$$	The bit-length of \$.
w	Number of users in the ring.
N_{in}	Number of input wallets of a user[3].
IW_i	Input wallet of the i-th user $act_i = \left\{\mathsf{pk}_{(in),i}^{(k)}, \mathbf{cn}_{(in),i}^{(k)}\right\}_{k\in[N_{in}]}$.
IW	Set of input wallet = $\{IW_i\}_{i\in[w]}$.
IW_π	Input wallet of user $\pi = \left\{\mathsf{pk}_{(in),\pi}^{(k)}, \mathbf{cn}_{(in),\pi}^{(k)}\right\}_{k\in[N_{in}]}$.
K_π	User π's private-keys = $ask_\pi = \left\{\mathsf{sk}_{(in),\pi}^{(k)}, \mathbf{ck}_{(in),\pi}^{(k)}\right\}_{k\in[N_{in}]}$.
N_{out}	Number of output wallets.
OW	Set of output wallet = $\{OW^{(j)}\}_{j\in[N_{out}]} = \{\mathsf{pk}_{(out)}^{(j)}, \mathbf{cn}_{(out)}^{(j)}\}_{j\in[N_{out}]}$.
OA	Set of output addresses = $\left\{\mathsf{pk}_{(out)}^{(j)}\right\}_{j\in[N_{out}]}$.
TX	Transaction = (μ, IW, OW).
TN	Set of serial/transaction numbers (linking tag).

MIMO scheme works using a set of algorithms MIMO.LRCT = (MIMO.LRCT.Setup, MIMO.LRCT.KeyGen, MIMO.LRCT.Mint, MIMO.LRCT.Spend, MIMO.LRCT.Verify) and they are listed as:

1. (Pub-Params) ← MIMO.LRCT.Setup(λ): On input the security parameter λ, this algorithm calls MIMO.L2RS.Setup (Appendix C) and outputs the public parameters $\mathbf{A} \in \mathcal{R}_q^{2\times(m-1)}$ and $\mathbf{H} \in \mathcal{R}_q^{2\times(m-1)}$.
2. $(\mathbf{a}, \mathbf{S})$ ← MIMO.LRCT.KeyGen($\mathbf{A}$): Given the public parameter $\mathbf{A} \in \mathcal{R}_q^{2\times(m-1)}$, it runs MIMO.L2RS.KeyGen (Algorithm 1) and outputs a pair of keys, the public-key or one-time address pk as $\mathbf{a} \in \mathcal{R}_q^2$ and the private-key sk as $\mathbf{S} \in \mathcal{R}_q^{(m-1)\times 1}$. A homomorphic commitment is generated as $\mathbf{a} = \mathsf{Com}_{\mathbf{A}}(0, \mathbf{S}) = \mathbf{A} \cdot \mathbf{S} + \overline{0} \bmod q \in \mathcal{R}_q^2$.
3. $(\mathbf{cn}, \mathbf{ck})$ ← MIMO.LRCT.Mint($\mathbf{A}$, \$): It receives the public parameter $\mathbf{A}$ and input amount \$ $\in [0, 2^{\ell_\$ - 1}]$. It computes a coin $\mathbf{cn}$, by choosing a coin-key $\mathbf{ck} \in \mathsf{Dom}_{\mathbf{S}}$, where every component of $\mathbf{ck}$ is chosen uniformly and independently, then compute $\mathbf{cn}$ (as Algorithm 4) and this algorithm returns $(\mathbf{cn}, \mathbf{ck})$.

[3] In this work, we consider that all users have a fixed number of input wallets N_{in}.

Algorithm 4 MIMO.LRCT.Mint

Input: $\left(\mathbf{A} \in \mathcal{R}_q^{2\times(m-1)}, \$ \in [0, 2^{\ell_\$ - 1}]\right)$, being the public parameter $\mathbf{A}$ and the amount \$.
Output: $(\mathbf{cn}, \mathbf{ck})$, where they are the coin and the coin key, respectively.
1: **procedure** MIMO.LRCT.MINT($\mathbf{A}, \$$)
2: Let $\mathbf{ck}^T = (\mathbf{ck}_1, \ldots, \mathbf{ck}_{m-1}) \in \mathcal{R}_q^{1\times(m-1)}$ with $\mathbf{ck}_i \leftarrow (-2^\gamma, 2^\gamma)^n$, for $1 \leq i \leq m-1$
3: $\mathbf{cn} = \mathsf{Com}_{\mathbf{A}}(\$, \mathbf{ck}) = \mathbf{A} \cdot \mathbf{ck} + \overline{\$} \bmod q \in \mathcal{R}_q^2$ with $\overline{\$} = (0, \$)^T \in \mathcal{R}_q^{1\times 2}$
4: **return** $(\mathbf{cn}, \mathbf{ck})$

4. $(TX, \mathsf{sig}, TN) \leftarrow \mathsf{MIMO.LRCT.Spend}(\mu, IW, IW_\pi, K_\pi, OA, \$_{(out)}^{(j)}, \mathsf{Pub\text{-}Params})$: This algorithm spends/transfers amounts from the user π's input wallets to some output wallets. We denote the user π who successfully created its input wallets IW_π, based on determine amounts $\$_{(in)}$. Note that notation of these parameters are defined in Table 1, and this spend algorithm is briefly described in Algorithm 5. Then, π selects the recipients' valid public keys or output addresses OA where π wants to spend his/her amount. To do so π performs the following steps:
   (a) π receives $\{\$_{(out)}^{(j)}\}_{j\in[N_{out}]}$, with $\$_{(out)}^{(j)} \in [0, \ldots, 2^{\ell_\$ - 1}]$, such balance satisfies, we call this condition *amount preservation*. This checks that input amounts are equal to output amounts, by checking if the following equality holds:

$$\sum_{k=1}^{N_{in}} \$_{(in),\pi}^{(k)} = \sum_{j=1}^{N_{out}} \$_{(out)}^{(j)}. \tag{2}$$

   π then runs $\mathsf{MIMO.LRCT.Mint}(\mathbf{A}, \$_{(out)}^{(j)})$ for $j \in [N_{out}]$ and obtain $(\mathbf{cn}_{(out)}^{(j)}, \mathbf{ck}_{(out)}^{(j)})_{j\in[N_{out}]}$, which define the output wallets as

$$OW = \{OW^{(j)}\}_{j\in[N_{out}]} = \{\mathbf{a}_{(out)}^{(j)}, \mathbf{cn}_{(out)}^{(j)}\}_{j\in[N_{out}]}. \tag{3}$$

   Then, the output coin-keys and amounts $\{\mathbf{ck}_{(out)}^{(j)}, \$_{(out)}^{(j)}\}_{j\in[N_{out}]}$ are securely sent to users with valid $OA^j = \{\mathbf{a}_{(out)}^{(j)}\}_{j\in[N_{out}]}$.
   (b) User π selects $(w-1)$ input wallets from the blockchain which he/she uses to anonymously transfer her/his input wallets $\{IW_\pi^{(k)}\}_{k\in[N_{in}]}$. Then, a preliminary ring signature list is built as $IW = \{IW_i\}_{i\in[w]} = \{\mathbf{a}_{(in),i}^{(k)}, \mathbf{cn}_{(in),i}^{(k)}\}_{i\in[w],k\in[N_{in}]}$.
   (c) π adds a record to IW_i in order to homomorphically compute and verify the *amount preservation*; this uses the homomorphic commitment scheme (defined in Section 4.1). The result of this computation is a commitment to zero, where the user π is only able to obtain since he/she knows both IW_π and OW. This new record is placed in the position $(N_{in}+1)$ and then a list L' is defined as:

$$L' = \left\{\mathbf{a}_{(in),i}^{(k)}\right\}_{i\in[w],k\in[N_{in}+1]}, \tag{4}$$

   with $\mathbf{a}_{(in),i}^{(N_{in}+1)} \triangleq \mathsf{Com}_{\mathbf{A}}\left(\sum_{k=1}^{N_{in}} \$_{(in),i}^{(k)} - \sum_{j=1}^{N_{out}} \$_{(out)}^{(j)}, \mathbf{S}_{(in),i}^{(N_{in}+1)}\right)$, where $\mathbf{S}_{(in),i}^{(N_{in}+1)} \triangleq \sum_{k=1}^{N_{in}} \mathbf{S}_{(in),i}^{(k)} + \mathbf{ck}_{(in),i}^{(k)} - \sum_{j=1}^{N_{out}} \mathbf{ck}_{(out)}^{(j)} \in \mathcal{R}_q^{(m-1)\times 1}$. This implies that

$$\mathbf{a}_{(in),i}^{(N_{in}+1)} = \sum_{k=1}^{N_{in}} \mathbf{a}_{(in),i}^{(k)} + \mathbf{cn}_{(in),i}^{(k)} - \sum_{j=1}^{N_{out}} \mathbf{cn}_{(out)}^{(j)}. \tag{5}$$

   Note that if the *amount preservation* conditions (2) and (6) (for every $k \in [N_{in}]$) are achieved, then $\mathbf{a}_{(in),i}^{(N_{in}+1)} = \mathsf{Com}_{\mathbf{A}}(0, \mathbf{S}_{(in),i}^{(N_{in}+1)})$.

$$\mathbf{a}_{(in),i}^{(k)} = \mathsf{Com}_{\mathbf{A}}(0, \mathbf{S}_{(in),i}^{(k)}) = \mathbf{A} \cdot \mathbf{S}_{(in),i}^{(k)} + \overline{0} \mod q \in \mathcal{R}_q^2. \tag{6}$$

(d) To sign the transaction, we use the π's private-keys: $\{\mathbf{S}^{(k)}_{(in),\pi}\}_{k\in[N_{in}+1]}$, the list L' and a transaction string $\mu \in \{0,1\}^*$. Then, we run MIMO.L2RS.SigGen (Algorithm 2) which outputs:

$$\sigma_{L'}(\mu) = \left(\mathbf{c}_1, \{\mathbf{t}_1^{(k)}, \ldots, \mathbf{t}_w^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{h}^{(k)}\}_{k\in[N_{in}]}\right). \tag{7}$$

(e) We show that the output amount $\$^{(j)}_{(out)}$ lies in a non-zero range value, by running our range proof (Algorithm 8) $\Pi_{Range}.\mathcal{P}_{Range}\left(\left\{\mathbf{cn}^{(j)}_{(out)}, \mathbf{ck}^{(j)}_{(out)}, \$^{(j)}_{(out)}, \mathbf{A}\right\}_{j\in[N_{out}]}\right)$. This proof outputs: $\left\{PoK^{(j)}_{Range}\right\}_{j\in[N_{out}]}$.

(f) We set the transaction TX as (μ, IW, OW) and $TN = \{\mathbf{h}^{(k)}\}_{k\in[N_{in}]}$. This algorithm outputs TX, TN, $\mathsf{sig}_\pi = \left(\sigma_{L'}(\mu), \left\{PoK^{(j)}_{Range}\right\}_{j\in[N_{out}]}\right)$.

5. $(Accept/Reject) \leftarrow$ MIMO.LRCT.Verify$(TX, \mathsf{sig}_\pi, TN)$: This algorithm calls MIMO.L2RS.SigVer$(\mathsf{sig}_{\pi,1}, L', \mathsf{Pub\text{-}Params})$ (Algorithm 3) with $\mathsf{sig}_{\pi,1} = \sigma_{L'}(\mu)$, and on input $\mathsf{sig}_{\pi,2} = \left\{PoK^{(j)}_{Range}\right\}_{j\in[N_{out}]}$, it runs (Algorithm 9) $\Pi_{Range}.\mathcal{V}_{Range}\left(\left\{PoK^{(j)}_{Range}, \mathbf{cn}^{(j)}_{(out)}, \mathbf{A}\right\}_{j\in[N_{out}]}\right)$. This MIMO.LRCT.Verify outputs *Accept* if both MIMO.L2RS.SigVer$(\cdot)$ and $\Pi_{Range}.\mathcal{V}_{Range}(\cdot)$ output *Accept*, else it outputs *Reject*.

Algorithm 5 MIMO.LRCT.Spend

Input: $(\mu, IW, IW_\pi, K_\pi, OA, \$^{(j)}_{(out)}, \mathsf{Pub\text{-}Params})$, being the message, the input wallets, π's input wallet, π's private keys, the output addresses, the output amount and the public parameter, respectively.
Output: $(TX, \sigma_{L'}(\mu), TN)$

procedure MIMO.LRCT.Spend$(\mu, IW, IW_\pi, K_\pi, OA, \$^{(j)}_{(out)}, \mathsf{Pub\text{-}Params})$
  User π selects $\{\$^{(j)}_{(out)}\}_{j\in[N_{out}]}$ such that (2) is satisfied.
  User π runs MIMO.LRCT.Mint$(\mathbf{A}, \$^{(j)}_{(out)})$ for $j \in [N_{out}]$ to generate $(\mathbf{cn}^{(j)}_{(out)}, \mathbf{ck}^{(j)}_{(out)})$ and sets OW as in (3).
  User π sends securely coin-keys and amounts $\{\mathbf{ck}^{(j)}_{(out)}, \$^{(j)}_{(out)}\}_{j\in[N_{out}]}$ to user's $OA^j = \mathbf{a}^{(j)}_{(out)}$ for $j \in [N_{out}]$.
  Create the list of input wallets $IW = \{IW_i\}_{i\in[w]} = \{\mathbf{a}^{(k)}_{(in),i}, \mathbf{cn}^{(k)}_{(in),i}\}_{i\in[w],k\in[N_{in}]}$.
  Let $L' = \{\mathbf{a}^{(k)}_{(in),i}\}_{i\in[w],k\in[N_{in}+1]}$, where $\mathbf{a}^{(k)}_{(in),i}$ are defined in (6) and (5) for $1 \le k \le N_{in}$ and $k = N_{in}+1$, respectively.
  Call MIMO.L2RS.SigGen$\left(\left\{\mathbf{S}^{(k)}_{(in),\pi}\right\}_{k\in[N_{in}+1]}, L', \mu, \mathsf{Pub\text{-}Params}\right)$ and obtain $\sigma_{L'}(\mu)$ as in (7).
  Run $\Pi_{Range}.\mathcal{P}_{Range}\left(\left\{\mathbf{cn}^{(j)}_{(out)}, \mathbf{ck}^{(j)}_{(out)}, \$^{(j)}_{(out)}, \mathbf{A}\right\}_{j\in[N_{out}]}\right)$ for $j \in [N_{out}]$, it outputs $\left\{PoK^{(j)}_{Range}\right\}_{j\in[N_{out}]}$.
  Set $\mathsf{sig}_\pi = (\sigma_{L'}(\mu), \left\{PoK^{(j)}_{Range}\right\}_{j\in[N_{out}]})$.
  Let $TX = (\mu, IW, OW)$ and $TN = \{\mathbf{h}^{(k)}\}_{k\in[N_{in}+1]}$.
  return $(TX, \mathsf{sig}_\pi, TN)$

5.1 Range Preservation

This section presents the techniques that we use to preserve the range preservation; that is, these techniques prevent the negative output amount $\$_{(out)}$ attack, also known as out-of-range attack. Since $\$_{(out)}$ is decomposed into binary representation such $\$_{(out)} = \sum_{i=0}^{\ell_\$-1} 2^i b_i$, where $b_i \in \{0,1\}$, we start proving that each bit b_i is binary, using the OR-Proof technique from [12]. We adjust this OR-Proof technique to our commitment scheme, which is constructed in Section 4.1. Thereafter, we use a range-proof technique to show that this amount lies in a range of *non-negative* values, so we prove that each committed output amount is within a range which cannot overflow (e.g. $[0, 2^{64})$).

Binary Proof. We define a protocol for the OR-Proof for our homomorphic commitments as Π_{OR} with the prover $\mathcal{P}_{OR}$ and the verifier $\mathcal{V}_{OR}$. This protocol is a zero knowledge proof where the commitment of the bit $\mathbf{cb}$ opens to have a value $b \in \{0,1\}$. The proof is approximate *or relaxed* by using a relaxation

factor f. Meaning that this proof only proves knowledge of a randomness of the bit $\mathbf{rb}$ such that $f \cdot \mathbf{cb}$ opens to a message $f \cdot b$ with $f \in \mathcal{R}_q$ with respect to $\mathbf{rb}$. We use the public parameter $\mathbf{A} \in \mathcal{R}_q^{2\times(m-1)}$ to define our two binary relations such $R_{OR} = R_0 \vee R_1$:

$$R_0 \triangleq \{(\mathbf{cb}, \mathbf{rb}) \in \mathcal{R}_q^2 \times \mathcal{R}_q^{(m-1)\times 1}, \mathbf{cb} = \mathsf{Com}_{\mathbf{A}}(0, \mathbf{rb}) = \mathbf{A} \cdot \mathbf{rb} + \overline{0}, \|\mathbf{rb}\| \leq B_{OR}\}$$
$$R_1 \triangleq \{(\mathbf{cb}, \mathbf{rb}) \in \mathcal{R}_q^2 \times \mathcal{R}_q^{(m-1)\times 1}, \mathbf{cb} = \mathsf{Com}_{\mathbf{A}}(1, \mathbf{rb}) = \mathbf{A} \cdot \mathbf{rb} + \overline{1}, \|\mathbf{rb}\| \leq B_{OR}\}$$

Two *relaxed* binary relations (i.e. $R'_{OR} = R'_0 \vee R'_1$) are also defined where a witness will be recovered by the soundness extractor:

$$R'_0 \triangleq \{(\mathbf{cb}, \mathbf{rb}), 2 \cdot \mathbf{cb} = \mathsf{Com}_{\mathbf{A}}(2 \cdot 0, \mathbf{rb}) = \mathbf{A} \cdot \mathbf{rb} + 2 \cdot \overline{0}, \|\mathbf{rb}\| \leq B'_{OR}\}$$
$$R'_1 \triangleq \{(\mathbf{cb}, \mathbf{rb}), 2 \cdot \mathbf{cb} = \mathsf{Com}_{\mathbf{A}}(2 \cdot 1, \mathbf{rb}) = \mathbf{A} \cdot \mathbf{rb} + 2 \cdot \overline{1}, \|\mathbf{rb}\| \leq B'_{OR}\}, \text{where } B'_{OR} > B_{OR}$$

We now define below a challenge set of monomials such that for any distinct pair of monomials $X^i, X^j \in \mathcal{C}_0$ with all the coefficients of $2 \cdot (X^i - X^j)^{-1}$ are in $\{-1, 0, 1\}$ according to Lemma 3. Since we are using monomials, the relaxation factor is defined as $f = 2$.

$$\mathcal{C}_0 \triangleq \{X^i \in \mathcal{R}_q, i = 0, \ldots, 2n-1\}$$

Remark 3. The main difference between our OR-proof and the OR-proof from [12] is the size of the challenges. As we cannot achieve soundness of our range proof using the same challenge space as in [12], we adapt their protocol to another challenge space which we call $\mathcal{C}_0$ (this space was introduced in [5]), which consists of monomials in $\mathcal{R}_q$. With this new space $\mathcal{C}_0$ we are now able to prove the soundness of our relaxed proof to the relaxed relation R'_{OR} with relaxation factor $f = 2$. However, because the relatively small size of the challenge space $\mathcal{C}_0$ these relatively small challenges, Π_{OR} needs to repeat the basic protocol θ times in parallel, where the rejection sampling (defined in Lemma 1) returns something after $\theta - 1$ repetitions. In practice, we only need a relatively small $\theta < 20$, whereas previous lattice based range proofs (i.e. [18]) need much larger $\theta > 100$ for the same soundness level.

The challenge space $\mathfrak{P}$ consists of the set of all permutations of dimension n, $\mathtt{Perm}(n)$, and a bit $c \in \{0, 1\}$, i.e. $\mathfrak{P} \triangleq \{p = (s, c) \in \mathtt{Perm}(n) \times \{0, 1\}\}$. Each $p \in \mathfrak{P}$ permutes the exponent of a polynomial in $\mathcal{C}_0$ according to the permutation s. Let $f, g \in \mathcal{C}_0$ be two monomials, if $f = X^{i_f}, g = (-1)^c \cdot X^{i_g}$ and $s(i_f) = i_g$, then we define $p(f) = g$. It holds that $\Pr[p(f) = g] = 1/|\mathcal{C}_0|$ for a uniformly random $p \in \mathfrak{P}$, for any two fixed elements $f, g \in \mathcal{C}_0$. σ_{OR} is defined to be a positive real parameter, whereas B_{OR} is a positive real bound. We also need a cryptographic hash function H modeled as random oracle, which maps arbitrary inputs to the uniform distribution over the challenge space $\mathfrak{P}^\theta$. Our OR-Proof protocol Π_{OR} is defined in two algorithms (Alg. 6 and 7), for proving if one bit is binary.

Range Proof. We define a range proof Π_{range}, having two algorithms, one for the prover $\mathcal{P}_{range}$, and one for the verifier $\mathcal{V}_{range}$. $\mathcal{P}_{range}$ receives from $\mathsf{MIMO.LRCT.Spend}(\cdot)$, Section 5, the parameters $\{\$^{(j)}_{(out)}, \mathbf{cn}^{(j)}_{(out)}, \mathbf{ck}^{(j)}_{(out)}\}_{j\in[N_{out}]} \in [0, \ldots, 2^{\ell_\$ - 1}] \times \mathcal{R}_q^2 \times \mathcal{R}_q^{(m-1)}$. We define the first relation R_{range} for this protocol:

$$R_{range} \triangleq \left\{ \begin{array}{c} \{\mathbf{cn}^{(j)}_{(out)}\}, \{\$^{(j)}_{(out)}, \mathbf{ck}^{(j)}_{(out)}\} : \forall j \ , \ \mathbf{cn}^{(j)}_{(out)} = \mathsf{Com}_{\mathbf{A}}(\$^{(j)}_{(out)}, \mathbf{ck}^{(j)}_{(out)}) = \\ \mathbf{A} \cdot \mathbf{ck}^{(j)}_{(out)} + \overline{\$}^{(j)}_{(out)}, \|\mathbf{ck}^{(j)}_{(out)}\| \leq 2\beta, \$^{(j)}_{(out)} \in [0, \ldots, 2^{\ell_\$ - 1}] \end{array} \right\}$$

A *relaxed* relation (R'_{range}) is also defined by using f where the corresponding witness will be recovered by its soundness extractor:

$$R'_{range} \triangleq \left\{ \begin{array}{c} \{\mathbf{cn}^{(j)}_{(out)}\}, \{\$^{(j)}_{(out)}, \mathbf{ck}^{(j)}_{(out)}\} : \forall j \ s.t. \ f \cdot \mathbf{cn}^{(j)}_{(out)} = \mathsf{Com}_{\mathbf{A}}(f \cdot \$^{(j)}_{(out)}, \mathbf{ck}^{(j)}_{(out)}) = \\ \mathbf{A} \cdot \mathbf{ck}^{(j)}_{(out)} + f \cdot \overline{\$}^{(j)}_{(out)}, \|\mathbf{ck}^{(j)}_{(out)}\| \leq 2\beta, \$^{(j)}_{(out)} \in [0, \ldots, 2^{\ell_\$ - 1}], f = 4 \end{array} \right\}$$

$\mathcal{P}_{range}$ and $\mathcal{V}_{range}$ are described below in algorithms (8 and 9):

Algorithm 6 OR-Proof Protocol Π_{OR}, prover's algorithm $\mathcal{P}_{OR}$

Input: $(\mathbf{cb} = \mathbf{A} \cdot \mathbf{rb} + \overline{b}, \mathbf{A}, \mathbf{rb}, b \in \{0,1\})$
Output: $\left\{f_0^{(t)}, f_1^{(t)}, \mathbf{r}_0^{(t)}, \mathbf{r}_1^{(t)}\right\}_{t=1}^{\theta}$

procedure $\Pi_{OR}.\mathcal{P}_{OR}(\mathbf{cb}, \mathbf{A}, \mathbf{rb}, b)$
  for $(1 \leq t \leq \theta)$ **do**
    Let $f_{1-b}^{(t)} \leftarrow \mathcal{C}_0$
    Let $\mathbf{u}^{(t)} \leftarrow D_{\sigma_{OR}}^{n(m-1)}$
    $\mathbf{r}_{1-b}^{(t)} \leftarrow D_{\sigma_{OR}}^{n(m-1)}$
    $\mathbf{a}_{1-b}^{(t)} \coloneqq \mathbf{A} \cdot \mathbf{r}_{1-b}^{(t)} + f_{1-b}^{(t)} \cdot \overline{(1-b)} - f_{1-b}^{(t)} \cdot \mathbf{cb}$
    $\mathbf{a}_b^{(t)} \coloneqq \mathbf{A} \cdot \mathbf{u}^{(t)}$
  Concatenate $(\mathbf{a}_{1-b})_{||} \coloneqq \left(\mathbf{a}_{1-b}^{(1)}, \ldots, \mathbf{a}_{1-b}^{(\theta)}\right)$
  Concatenate $(\mathbf{a}_b)_{||} \coloneqq \left(\mathbf{a}_b^{(1)}, \ldots, \mathbf{a}_b^{(\theta)}\right)$
  $(p^{(1)}, \ldots, p^{(\theta)}) \coloneqq H\left(\mathbf{cb}, (\mathbf{a}_{1-b}^{(t)})_{||}, (\mathbf{a}_b^{(t)})_{||}\right) \leftarrow \mathfrak{P}^{\theta}$
  for $(1 \leq t \leq \theta)$ **do**
    $f_b^{(t)} = (p^{(t)})^{2b-1}(f_{1-b}^{(t)})$
    $\mathbf{r}_b^{(t)} = \mathbf{u}^{(t)} + f_b^{(t)} \cdot \mathbf{rb}$
  $\mathbf{r}_{b||} \coloneqq \left(\mathbf{r}_b^{(1)}, \ldots, \mathbf{r}_b^{(\theta)}\right)$
  $(f_b \cdot \mathbf{rb})_{||} \coloneqq \left(f_b^{(1)} \cdot \mathbf{rb}, \ldots, f_b^{(\theta)} \cdot \mathbf{rb}\right)$
  Continue with probability $1 - \min\left\{\frac{D_{\sigma_{OR}}^{n(m-1)\theta}(\mathbf{r}_{b||})}{3 \cdot D_{(f_b \cdot \mathbf{rb})_{||}, \sigma_{OR}}^{n(m-1)\theta}(\mathbf{r}_{b||})}, 1\right\}$ otherwise **Restart**
  return $PoK_{OR} = \left\{f_0^{(t)}, f_1^{(t)}, \mathbf{r}_0^{(t)}, \mathbf{r}_1^{(t)}\right\}_{t=1}^{\theta}$

Algorithm 7 OR-Proof Protocol Π_{OR}, verifier's algorithm $\mathcal{V}_{OR}$

Input: $(\mathbf{cb}, \mathbf{A}, PoK_{OR})$ with $PoK_{OR} = \left\{f_0^{(t)}, f_1^{(t)}, \mathbf{r}_0^{(t)}, \mathbf{r}_1^{(t)}\right\}_{t=1}^{\theta}$
Output: Accept or Reject

procedure $\Pi_{OR}.\mathcal{V}_{OR}(\mathbf{cb}, \mathbf{A}, PoK_{OR})$
  for $(1 \leq t \leq \theta)$ **do**
    Let $\mathbf{a}_0^{(t)} \coloneqq \mathbf{A} \cdot \mathbf{r}_0^{(t)} - f_0^{(t)} \cdot \mathbf{cb}$
    Let $\mathbf{a}_1^{(t)} \coloneqq \mathbf{A} \cdot \mathbf{r}_1^{(t)} + f_1^{(t)} \cdot \overline{1} - f_1^{(t)} \cdot \mathbf{cb}$
  Concatenate $(\mathbf{a}_0)_{||} \coloneqq \left(\mathbf{a}_0^{(1)}, \ldots, \mathbf{a}_0^{(\theta)}\right)$
  Concatenate $(\mathbf{a}_1)_{||} \coloneqq \left(\mathbf{a}_1^{(1)}, \ldots, \mathbf{a}_1^{(\theta)}\right)$
  $(p^{(1)}, \ldots, p^{(\theta)}) \coloneqq H\left(\mathbf{cb}, (\mathbf{a}_0)_{||}, (\mathbf{a}_1)_{||}\right) \leftarrow \mathfrak{P}^{\theta}$
  for $(1 \leq t \leq \theta)$ **do**
    if $\|\mathbf{r}_0^{(t)}\| \leq B'_{OR} \wedge \|\mathbf{r}_1^{(t)}\| \leq B'_{OR}$ **then** Continue; otherwise Reject
    else if $f_0^{(t)} \in \mathcal{C}_0 \wedge f_1^{(t)} = p^{(t)}(f_0^{(t)})$ **then** Continue; otherwise Reject
  return Accept

Algorithm 8 Range-Proof Protocol Π_{Range}, prover's algorithm $\mathcal{P}_{Range}$

Input: $\left(\left\{\mathbf{cn}_{(out)}^{(j)}, \mathbf{ck}_{(out)}^{(j)}, \$_{(out)}^{(j)}, \mathbf{A}\right\}_{j\in[N_{out}]}\right)$

Output: $\left\{PoK_{Range}^{(j)}\right\}_{j\in[N_{out}]}$

procedure $\Pi_{Range}.\mathcal{P}_{Range}(\left\{\mathbf{cn}_{(out)}^{(j)}, \mathbf{ck}_{(out)}^{(j)}, \$_{(out)}^{(j)}, \mathbf{A}\right\}_{j\in[N_{out}]})$
  for $(1 \le j \le N_{out})$ **do**
    Decompose in binary $\$_{(out)}^{(j)} = \left\{b_i^{(j)}\right\}_{i=0}^{\ell_\$ - 1}$
    Compute commitment to each bit $\mathbf{cb}_i^{(j)} = \mathsf{Com}_{\mathbf{A}}(b_i^{(j)}, \mathbf{rb}_i^{(j)}) = \mathbf{A}\cdot\mathbf{rb}_i^{(j)} + \bar{b}_i^{(j)}$, as defined in Section 4.1
    for $(0 \le i \le \ell_\$ - 1)$ **do**
      Calls binary proof $PoK_{OR,i}^{(j)} = \Pi_{OR}.\mathcal{P}_{OR}(\mathbf{cb}_i^{(j)}, \mathbf{A}, \mathbf{rb}_i^{(j)}, b_i^{(j)})$
    Compute $\mathbf{prck}^{(j)} = \sum_{i=0}^{\ell_\$ - 1} 2^i \mathbf{rb}_i^{(j)} - \mathbf{ck}_{(out)}^{(j)}$
    Compute $\mathbf{prcn}^{(j)} = \sum_{i=0}^{\ell_\$ - 1} 2^i \mathbf{cb}_i^{(j)} - \mathbf{cn}_{(out)}^{(j)}$, it should be equal to $\mathsf{Com}_{\mathbf{A}}(0, \mathbf{prck}^{(j)})$
    $\mathbf{y}^{(j)} \leftarrow D_{\sigma_{Range}}^{n(m-1)}$
    $\mathbf{c}^{(j)} \leftarrow H(\mathbf{prcn}^{(j)} \cdot \mathbf{y}^{(j)})$
    $\mathbf{z}^{(j)} \leftarrow \mathbf{prck}^{(j)} \cdot \mathbf{c}^{(j)} + \mathbf{y}^{(j)}$, this is a PoK of $\mathsf{Com}_{\mathbf{A}}(0, \mathbf{prck}^{(j)})$
  $\mathbf{z}_{||} = (\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(N_{out})})$
  $(\mathbf{prck}\cdot\mathbf{c})_{||} = \left(\mathbf{prck}^{(1)}\cdot\mathbf{c}^{(1)}, \ldots, \mathbf{prck}^{(N_{out})}\cdot\mathbf{c}^{(N_{out})}\right)$
  Continue with probability $1 - \min\left\{\frac{D_{\sigma_{Range}}^{n(m-1)}(\mathbf{z}_{||})}{M\cdot D_{(\mathbf{prck}\cdot\mathbf{c})_{||}, \sigma_{Range}}^{n(m-1)}(\mathbf{z}_{||})}, 1\right\}$ otherwise **Restart**
  return $PoK_{Range}^{(j)} = \left\{PoK_{OR,i}^{(j)}, \mathbf{cb}_i^{(j)}, \mathbf{z}^{(j)}, \mathbf{c}^{(j)}\right\}_{i\in[0,\ell_\$-1], j\in[N_{out}]}$

Algorithm 9 Range-Proof Protocol Π_{Range}, verifier's algorithm $\mathcal{V}_{Range}$

Input: $\left(\left\{PoK_{Range}^{(j)}\right\}_{j\in[N_{out}]}\right)$, where $PoK_{Range}^{(j)} = \left\{PoK_{OR,i}^{(j)}, \mathbf{cb}_i^{(j)}, \mathbf{z}^{(j)}, \mathbf{c}^{(j)}\right\}_{i\in[0,\ell_\$-1], j\in[N_{out}]}$.

Output: Accept or Reject

procedure $\Pi_{Range}.\mathcal{V}_{Range}(\left\{PoK_{Range}^{(j)}\right\}_{j\in[N_{out}]})$
  for $(1 \le j \le N_{out})$ **do**
    for $(0 \le i \le \ell_\$ - 1)$ **do**
      Accept $\stackrel{?}{=} \Pi_{OR}.\mathcal{V}_{OR}(\mathbf{cb}_i^{(j)}, \mathbf{A}, PoK_{OR,i}^{(j)})$, this checks the binary proof for $\mathbf{cb}_i^{(j)}$
    Compute $\mathbf{vrcn}^{(j)} = \sum_{i=0}^{\ell_\$ - 1} 2^i \mathbf{cb}_i^{(j)} - \mathbf{cn}_{(out)}^{(j)}$
    Check $\mathbf{c}^{(j)} \stackrel{?}{=} H(\mathbf{A}\cdot\mathbf{z}^{(j)} - \mathbf{vrcn}^{(j)}\cdot\mathbf{c}^{(j)})$, otherwise Reject; this checks the range proof of zero commitment
  return Accept or Reject

Theorem 3 (Binary Proof). *If* $\sigma_{OR} \geq 22\sqrt{\kappa}B_{OR}$ *and* $B'_{OR} \geq 2\sqrt{n}\sigma_{OR}$*, then the protocol* $\Pi_{OR}(\mathcal{P}_{OR}, \mathcal{V}_{OR})$ *is a* R'_b*-Protocol complete for relation* R_{OR} *and sound for relation* R'_{OR}*.*

Proof. The proof of Theorem 3 is given in Appendix A. □

Theorem 4 (Range Proof). *The protocol described in Step 2 of the range proof is a proof of knowledge (from [20]) complete for relation* R_{range} *and sound for relation* R'_{range} *with* $\beta_{range} = 2^{\ell_\$+2}n\sqrt{\kappa n(m-1)}\sigma_{OR} + 2^2\sqrt{n}\beta_v$*.*

Proof. The proof Theorem 4 are given in Appendix A. □

6 Security Analysis

Theorem 5 (Balance). *If* MIMO.L2RS *with parameter* β_v *is unforgeable, linkable and* $\mathsf{Com}_{\boldsymbol{A}}$ *is* $\beta-$*binding with* $\beta = 4\sqrt{\kappa(2\beta_v)^2 + \kappa(2\beta_v)^2 n(m-1)((2N_{in}+N_{out})2^\gamma)^2} + 2\beta_v N_{out}(2^{\ell_\$+2}n\sqrt{\kappa n(m-1)}\sigma_{OR} + 2^2\sqrt{n}\beta_v)$*, then* MIMO.LRCT *satisfies balance.*

Proof. The proof is given in Appendix B.1. □

Corollary 1 (Balance). *The Balance of* MIMO.LRCT *is satisfied if* $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,\beta_{Balance}}$ *is hard where* $\beta_{balance} = \max(\beta_{case1.1}, \beta_{case1.2}, \beta_{case2})$*.*

Proof. By combining Theorem 5 (Balance), along with Theorem 2 ($\beta-$Binding), Theorem 9 (MIMO.L2RS Unforgeability) and Theorem 11 (Linkability), this analysis concludes that the $\beta_{balance} = \max(\beta_{case1.1}, \beta_{case1.2}, \beta_{case2})$. $\beta_{case1.2}$ is seen as the maximum, then we said that $\beta_{balance} = 4\sqrt{\kappa(2\beta_v)^2 + \kappa(2\beta_v)^2 n(m-1)((2N_{in}+N_{out})2^\gamma)^2} + 2\beta_v N_{out}(2^{\ell_\$+2}n\sqrt{\kappa n(m-1)}\sigma_{OR} + 2^2\sqrt{n}\beta_v)$. □

Remark 4. In the balance proof, we only need zero-time unforgeability, meaning that in the reduction the attacker produces a forgery without seeing any signatures. Secondly, we do not need the message part of the signature, and thus this is treated as a Proof of Knowledge.

Theorem 6 (LRCT-Anonymity). *Suppose* $n \cdot \frac{1}{2} \cdot \sqrt{\frac{q^{4n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$ *and* $o \cdot h \cdot 2^{-n+1}$ *are negligible in* n *with an attack against the unconditional anonymity that makes* h *queries to the random oracle* H_1*, then the* MIMO.LRCT *scheme is unconditionally secure for anonymity and amount privacy as defined in Def. 7.*

Proof. The proof is given in Appendix B.2. □

Theorem 7 (LRCT-Non-Slanderability). *If* MIMO.LRCT *satisfies balance, then it satisfies non-slanderability as in Def. 8. In addition, the non-slanderability of* MIMO.LRCT *can be reduced to the non-slanderability of* MIMO.L2RS*.*

7 Performance Analysis

In this section, we propose a set of parameters for the MIMO.LRCT scheme. This construction is 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) [10]. We let $n = 1024$, $m = 132$, $\log q = 196$, $\kappa = 14$, $\eta = 1.1$, $\alpha = 0.5$, $\sigma = 22010$, $\sigma_{OR} = 277350$ and $\ell_\$ = 64$ to achieve the security parameter $\lambda = 100$, with α being the rejection sampling parameter determined in ([13] Section 3.2). Signature sizes of this analysis are illustrated in Table 2, where regular numbers for N_{in} and N_{out} were taken from Monero blockchain network[4].

Acknowledgement. This research project was supported by the Monash-HKPU(Hong Kong Polytechnic University)-Collinstar Blockchain Research Lab, whereas the work of Ron Steinfeld and Amin Sakzad was supported in part by ARC Discovery Project grant DP150100285. The work of Ron Steinfeld and Joseph K. Liu were also supported in part by ARC Discovery Project grant DP180102199.

[4] https://moneroblocks.info/

Table 2: Size estimation for MIMO.LRCT

MIMO.LRCT	$(N_{in}, N_{out}) = (1, 2)$	$(N_{in}, N_{out}) = (2, 2)$	$(N_{in}, N_{out}) = (3, 2)$
$\log(\beta)$ (Theorem 5)	≈ 126.3	≈ 126.3	≈ 126.3
Signature size ($w = 1$)	≈ 4.8 MB	≈ 5.1 MB	≈ 5.4 MB
Signature size ($w = 5$)	≈ 6.7 MB	≈ 8 MB	≈ 9.2 MB
Private-key size	≈ 49 KB	≈ 73 KB	≈ 98 KB
Public-key size	≈ 97 KB	≈ 146 KB	≈ 195 KB

References

1. W. A. Alberto Torres, R. Steinfeld, A. Sakzad, J. K. Liu, V. Kuchta, N. Bhattacharjee, M. H. Au, and J. Cheng. Post-Quantum One-Time Linkable Ring Signature and Application to Ring Confidential Transactions in Blockchain (Lattice RingCT v1.0). In *ACISP*, pages 558–576. Springer, 2018.
2. C. Baum, I. Damgård, V. Lyubashevsky, S. Oechsner, and C. Peikert. More Efficient Commitments from Structured Lattice Assumptions. In *SCN*, pages 368–385. Springer, 2018.
3. M. Bellare and G. Neven. Multi-signatures in the plain public-key model and a general forking lemma. In *CCS*, page 390. ACM, 2006.
4. J. Benaloh and M. de Mare. One-Way Accumulators: A Decentralized Alternative to Digital Signatures. In *EUROCRYPT*, pages 274–285. Springer, 1993.
5. F. Benhamouda, J. Camenisch, S. Krenn, V. Lyubashevsky, and G. Neven. Better Zero-Knowledge Proofs for Lattice Encryption and Their Application to Group Signatures. In *ASIACRYPT*, pages 551–572. Springer, 2014.
6. B. Bunz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. Bulletproofs: Short Proofs for Confidential Transactions and More. In *IEEE Symposium on Security and Privacy*. IEEE, 2018.
7. Bytecoin Team. Aggregate Addresses in CryptoNote: Towards Efficient Privacy. *https://bytecoin.org/static/files/docs/aggregate-addresses.pdf*, 2015.
8. M. Chase and A. Lysyanskaya. On Signatures of Knowledge. In *CRYPTO*. Springer, 2006.
9. 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.
10. Y. Chen and P. Q. Nguyen. BKZ 2.0: Better Lattice Security Estimates. In *ASIACRYPT*, pages 1–20. Springer, 2011.
11. M. Conti, S. K. E, C. Lal, and S. Ruj. A Survey on Security and Privacy Issues of Bitcoin. *IEEE Communications Surveys and Tutorials*, (IEEE), 2018.
12. R. del Pino, V. Lyubashevsky, G. Neven, and G. Seiler. Practical Quantum-Safe Voting from Lattices. In *CCS*, pages 1565–1581. ACM Press, 2017.
13. L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky. Lattice signatures and bimodal gaussians. In *CRYPTO*, pages 40–56. Springer, 2013.
14. T. Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.
15. A. Fiat and A. Shamir. How To Prove Yourself: Practical Solutions to Identification and Signature Problems. In *CRYPTO*, pages 186–194. Springer, 1986.
16. E. Fujisaki and K. Suzuki. Traceable ring signature. In *PKC*, volume 4450, pages 181–200. Springer, 2007.
17. P. Koshy, D. Koshy, and P. McDaniel. An Analysis of Anonymity in Bitcoin Using P2P Network Traffic. In *Financial Cryptography*, pages 469–485. Springer, 2014.
18. B. Libert, S. Ling, K. Nguyen, and H. Wang. Lattice-Based Zero-Knowledge Arguments for Integer Relations. In *CRYPTO*. Springer, 2018.
19. J. K. Liu, V. K. Wei, and D. S. Wong. Linkable spontaneous anonymous group signature for ad hoc groups. In *ACISP*, pages 325–335. Springer, 2004.
20. V. Lyubashevsky. Lattice Signatures without Trapdoors. In *EUROCRYPT*. Springer, 2012.
21. V. Lyubashevsky and G. Seiler. Short, Invertible Elements in Partially Splitting Cyclotomic Rings and Applications to Lattice-Based Zero-Knowledge Proofs. In *EUROCRYPT*, pages 204–224. Springer, 2018.
22. S. M. C. T. Mackenzie, Adam; Noether. Improving Obfuscation in the CryptoNote Protocol. *https://lab.getmonero.org/pubs/MRL-0004.pdf*, 2015.
23. G. Maxwell. Confidential Transactions. *https://xiph.org/confidential_values.txt*, 2015.
24. D. Micciancio and O. Regev. Lattice-based cryptography. In *Post-quantum cryptography*, pages 147–191. Springer, 2009.

25. Monero. About Monero — Monero - secure, private, untraceable. *https://getmonero.org/resources/about/*, 2014.
26. S. Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. *https://bitcoin.org/bitcoin.pdf*, 2009.
27. S. Noether. Ring Signature Confidential Transactions for Monero. *https://eprint.iacr.org/2015/1098*, 2015.
28. S. Noether, S. Noether, and A. Mackenzie. A Note on Chain Reactions in Traceability in CryptoNote 2.0. *https://lab.getmonero.org/pubs/MRL-0001.pdf*, 2014.
29. T. P. Pedersen. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In *CRYPTO*. Springer, 1991.
30. D. Ron and A. Shamir. Quantitative Analysis of the Full Bitcoin Transaction Graph. In *Financial Cryptography*, pages 6–24. Springer, 2013.
31. P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.
32. V. Shoup. Sequences of games: a tool for taming complexity in security proofs. *https://eprint.iacr.org/2004/332*, 2004.
33. 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*, pages 456–474. Springer, 2017.
34. N. Van Saberhagen. CryptoNote v 2.0. *https://cryptonote.org/whitepaper.pdf*, 2013.

A Proofs of Range Proof

Proof of Theorem 3 (OR-Proof):

Proof.
Correctness We show that $\mathbf{A} \cdot \mathbf{r}_0^{(t)} - f_0^{(t)} \cdot \mathbf{cb} = \mathbf{A} \cdot \mathbf{u}^{(t)}$ so:

$$\begin{aligned}
\mathbf{a}_0^{(t)} &= \mathbf{A} \cdot \mathbf{r}_0^{(t)} - f_0^{(t)} \cdot \mathbf{cb} \iff \\
&= \mathbf{A} \cdot (\mathbf{u}^{(t)} + f_b^{(t)} \cdot \mathbf{rb}) - f_0^{(t)} \cdot \mathbf{cb} \iff \\
&= \mathbf{A} \cdot \mathbf{u}^{(t)} + f_b^{(t)} \cdot (\mathbf{A} \cdot \mathbf{rb} - \mathbf{cb}) \iff \\
&= \mathbf{A} \cdot \mathbf{u}^{(t)} + f_b^{(t)} \cdot (\overline{b}) \iff
\end{aligned}$$

Since $\overline{b} = 0$, then this condition holds.
In the case when $\overline{b} = 1$, we refer to the line 12 of $\mathcal{P}_{OR}$ (Algorithm 6), we have $f_1^{(t)} = (p^{(t)})^1(f_0^{(t)})$, then line 9 of $\mathcal{V}_{OR}$ (Algorithm 7) is satisfied. When $\overline{b} = 0$, we have $f_0^{(t)} = (p^{(t)})^{-1}(f_1^{(t)}) \iff (p^{(t)})f_0^{(t)} = (f_1^{(t)})$, this also shown that the condition holds. We also check the bound of $(\mathbf{r}_0^{(t)}, \mathbf{r}_1^{(t)})$, where the rejection sampling lemma (to include the new lemma in the definition) is used to show that the distribution of $\mathbf{r}_{b||}$ (from Algorithm 6, line 14) is statistically close to $D_{\sigma_{OR}}^{n(m-1)\theta}$. Therefore, and each component $\mathbf{r}_b^{(t)}$ is statistically close to $D_{\sigma_{OR}}^{n(m-1)}$. A condition is needed $\sigma_{OR} \geq \|f_b \cdot \mathbf{rb})_{||}\|$. By the tail bound lemma (to be included) $\|\mathbf{r}_b^{(t)}\|_2 \leq \sqrt{n(m-1)} \cdot \sigma_{OR} = B'_{OR}$, except with probability $2^{n(m-1)}$.

Soundness Let $(\mathbf{cn}, \mathbf{r}_b) \in R_0 \vee R_1$. Let $\mathcal{P}_{OR}$ be a deterministic prover, who queries H on the same input. Therefore, her success probability depends on the output of H only. Let $p_0 = 1/|\mathcal{C}_0| + \epsilon$ be the success probability of the prover $\mathcal{P}_{OR}$. We need to construct an extractor $\mathcal{E}$ to extract the values $\mathbf{r}''_b$ and f''_b while making $poly(|(\mathbf{cn}, \mathbf{r}_b)|)/\epsilon$ times queries to H. It holds that $(\mathbf{cn}, \mathbf{r}''_b, f''_b) \in R'_0 \vee R'_1$. Extractor $\mathcal{E}$ runs $\mathcal{P}_{OR}(\mathbf{cn})$ on a challenge $p \leftarrow \mathfrak{P}$ and outputs a valid proof $(\mathbf{cn}, (\mathbf{r}_0, \mathbf{r}_1, f_0, f_1))$. Then, $\mathcal{E}$ runs $\mathcal{P}_{OR}(\mathbf{cn})$ on random challenges and outputs a proof $(\mathbf{cn}, (\mathbf{r}'_0, \mathbf{r}'_1, f'_0, f'_1))$ such that $f_0 \neq f'_0$ or $f_1 \neq f'_1$. Let $\alpha \in \{0, 1\}$ be a bit such that $f_\alpha \neq f'_\alpha$. Let $(\mathbf{cn}, \mathbf{a}_0, \mathbf{a}_1)$ be the hash query by $\mathcal{P}_{OR}(\mathbf{cn})$. Since both proofs verify, we have $\mathbf{a}_\alpha = \mathbf{A} \cdot \mathbf{r}_\alpha + f_\alpha \alpha - f_\alpha \cdot \mathbf{cn}$ and $\mathbf{a}_\alpha = \mathbf{A} \cdot \mathbf{r}'_\alpha + f'_\alpha \alpha - f'_\alpha \cdot \mathbf{cn}$. Subtracting these two equations results into:

$$(f_\alpha - f'_\alpha) \cdot \mathbf{cn} = \mathbf{A} \cdot (\mathbf{r}_\alpha - \mathbf{r}'_\alpha) + \alpha(f_\alpha - f'_\alpha)$$

Set $\mathbf{r}''_\alpha = \mathbf{r}_\alpha - \mathbf{r}'_\alpha$ and $f''_\alpha = f_\alpha - f'_\alpha$. It follows that $(\mathbf{cn}, \mathbf{r}''_\alpha, f''_\alpha) \in \mathcal{R}'_0 \vee \mathcal{R}'_1$. Finally, we show that $\mathcal{P}_{OR}(\mathbf{cn})$ outputs a proof such that $f_\alpha \neq f'_\alpha$ with at least negligible probability ϵ.

$$\begin{aligned}\Pr[\mathcal{P}_{OR}\text{ succ.} \quad &\wedge (f_0 = f'_0 \vee f_1 = f'_1)] \\ &= \Pr[\mathcal{P}_{OR}\text{ succ.}] - \Pr[\mathcal{P}_{OR}\text{ succ.} \wedge (f_0 = f'_0 \wedge f_1 = f'_1)] \\ &= p_0 - \Pr[\mathcal{P}_{OR}\text{ succ.} \wedge (f_0 = f'_0 \wedge p(f_0) = p(f'_0))] \\ &\geq p_0 - \Pr[p(f_0) = p(f'_0)] = \epsilon. \end{aligned} \tag{8}$$

HVZK: To prove special honest-verifier zero-knowledgeness, we have to show that the honest-verifier distribution and simulated distribution are identical. We show how to construct a simulator $\mathcal{S}$. For $(\mathbf{cn}, \mathbf{r}_b) \in R_0 \vee R_1$ and a challenge $p \in \mathfrak{P}$ the simulator $\mathcal{S}$ does the following:

1. $f_0 \leftarrow \mathcal{C}_0$
2. $f_1 = p(f_0)$
3. For $\alpha \in \{0, 1\}$, sample $\mathbf{r}_\alpha \leftarrow D^{n(m-1)}_{\sigma_{OR}}$
4. For $\alpha \in \{0, 1\}$, compute $\mathbf{a}_\alpha = \mathbf{A} \cdot \mathbf{r}_\alpha + f_\alpha \cdot \alpha - f_\alpha \cdot \mathbf{cn}$
5. Abort with probability $1 - 1/M$
6. Output $(\mathbf{r}_0, \mathbf{r}_1, f_0, f_1)$

Using the rejection sampling bounds, the distribution of the output of $\mathcal{S}$ is identical to the honest one. □

Proof of Theorem 4 (Range-Proof):

Proof. We prove the three security of a zero-knowledge proof of knowledge as defined in Definition 2.2.

Completeness: If the prover follows the protocol, then the following equation holds:

$$H(\mathbf{A}\overline{\mathbf{r}} - f'D, \mu) = H\Big(\mathbf{A} \cdot (f'\mathbf{r} + \mathbf{r}_0) - f'D, \mu\Big) \tag{9}$$

From (12) follows that $f'D = \mathsf{Com}_{\mathbf{A}}(0, \mathbf{r}) = \mathbf{A} \cdot \mathbf{r}$. Then, (9) is equivalent to:

$$\begin{aligned} H(\mathbf{A}\overline{\mathbf{r}} - f'D, \mu) &= H\Big(\mathbf{A} \cdot (f'\mathbf{r} + \mathbf{r}_0) - f'D, \mu\Big) \\ &= H\Big(f'\mathbf{A}\mathbf{r} + \mathbf{A}\mathbf{r}_0 - f'\mathbf{A} \cdot \mathbf{r}, \mu\Big) \\ &= H(\mathbf{A}\mathbf{r}_0, \mu) = f', \end{aligned}$$

where the last equation satisfies the verification. By the rejection sampling (Definition 1 the prover responds with probability $1/M^2$. The distribution of $\overline{\mathbf{r}}$ is statistically close to $D^{n(m-1)}_{12n\sqrt{n(m-1)}}$ since $\|f'\mathbf{r}\| \leq n\sqrt{n(m-1)}$ within the statistical distance 2^{-100}.

Soundness: To prove the soundness, we need to extract a witness $(f, \$, \mathbf{r})$ s.t. $f \cdot \mathbf{cn} = \mathsf{Com}_{\mathbf{A}}(f \cdot \$, \mathbf{r})$ with $\$ \in [0, 2^{\ell_\$} - 1]$.

From the OR proof witness extraction in Theorem 3, we first extract $(f''_i, b_i, \mathbf{r}_i)$ with $b_i \in \{0, 1\}$ such that for all $i \in [\ell_\$]$ the following relation holds:

$$f''_i \cdot \mathbf{cn}_i = \mathsf{Com}_{\mathbf{A}}(f''_i \cdot b_i, \mathbf{r}_i). \tag{10}$$

Let $f''_i = f_i - f'_i$ bet the difference between two challenges f_i and f'_i. According to Lemma 3 it holds that f'' is invertible in $\mathcal{R}_q$. Consequently we can multiply (10) by $(f''_i)^{-1}$ and get:

$$\mathbf{cn}_i = \mathsf{Com}_{\mathbf{A}}\left(b_i, (f''_i)^{-1} \cdot \mathbf{r}_i\right), \tag{11}$$

for all $i \in [\ell_\$]$. We now extract an opening $(f', \tilde{\mathbf{r}})$ of a commitment to 0 in the last step of the range proof protocol such that:

$$f' \cdot \left(\sum_{i=0}^{\ell_\$ - 1} 2^i \cdot \mathbf{cn}_i - \mathbf{cn} \right) = \mathsf{Com}_{\mathbf{A}}(0, \mathbf{r}) \tag{12}$$
$$\iff f' \cdot \sum_{i=0}^{\ell_\$ - 1} 2^i \cdot \mathbf{cn}_i - f' \cdot \mathbf{cn} = \mathsf{Com}_{\mathbf{A}}(0, \mathbf{r})$$

holds. Note that we use the PoK^* protocol from [20] which we adapt to our setting using $D := \left(\sum_{i=0}^{\ell_\$ - 1} 2^i \cdot \mathbf{cn}_i - \mathbf{cn} \right)$ and present in the Table 3:

Table 3: Π_{PoK^*} protocol [20]

$\mathcal{P}_{PoK}(\mathbf{r}, \mu, D)$		$\mathcal{V}_{PoK}(D)$
Pick $\mathbf{r}_0 \in D_{\sigma_0}^{m(n-1)}$		
Compute $U := \mathbf{A} \cdot \mathbf{r}_0$		
Set $f' := H(\mathbf{A}\mathbf{r}_0, \mu)$		
Compute $\overline{\mathbf{r}} := f'\mathbf{r} + \mathbf{r}_0$		
Abort with prob. ρ_0 from (13)		
	$\xrightarrow{f', \overline{\mathbf{r}}}$	
		Check $f' := H(\mathbf{A}\overline{\mathbf{r}} - f'D, \mu)$

where

$$\rho_0 := 1 - \min \left\{ \frac{D_{\sigma_0}^{n(m-1)}(\overline{\mathbf{r}})}{M \cdot D_{(f' \cdot \mathbf{r}), \sigma_0}^{n(m-1)}(\overline{\mathbf{r}})}, 1 \right\} \tag{13}$$

and $\sigma_0 = 12n\sqrt{n(m-1)}$. Using (11), it follows that:

$$f' \cdot \sum_{i=0}^{\ell_\$ - 1} 2^i \cdot \mathbf{cn}_i = f' \cdot \mathsf{Com}_{\mathbf{A}} \left(\sum_{i=0}^{\ell_\$ - 1} 2^i \cdot b_i, \sum_{i=0}^{\ell_\$ - 1} 2^i (f_i'')^{-1} \cdot \mathbf{r}_i \right). \tag{14}$$

After inserting the definition of $\mathbf{cn}$ into (12), we obtain:

$$f' \cdot \mathbf{cn} = f' \cdot \sum_{i=0}^{\ell_\$ - 1} 2^i \cdot \mathbf{cn}_i - \mathsf{Com}_{\mathbf{A}}(0, \mathbf{r}) \tag{15}$$

Next, we insert (14) into (15) to get:

$$\begin{aligned} f' \cdot \mathbf{cn} &= f' \cdot \mathsf{Com}_{\mathbf{A}} \left(\sum_{i=0}^{\ell_\$ - 1} 2^i \cdot b_i, \sum_{i=0}^{\ell_\$ - 1} 2^i (f_i'')^{-1} \cdot \mathbf{r}_i \right) - \mathsf{Com}_{\mathbf{A}}(0, \mathbf{r}) \\ &= \mathsf{Com}_{\mathbf{A}} \left(f' \cdot \$, f' \cdot \sum_{i=0}^{\ell_\$ - 1} 2^i (f_i'')^{-1} \cdot \mathbf{r}_i - \mathbf{r} \right), \end{aligned} \tag{16}$$

where we set $\$ = \sum_i 2^i b_i$ (note $\$ \in [0, 2^{\ell_\$} - 1]$). Now, we would have liked to show that $f \cdot (f_i'')^{-1} \cdot \mathbf{r}_i$ is 'small', but it is not. Instead, assume, there is a small and invertible $g \in \mathcal{R}_q$, such that $g \cdot (f_i'')^{-1} = h_i$ is small. Since f_i'' is a non-zero difference of monomials from $\mathcal{C}_0$, by Lemma 3, we can take $g = 2$ as it is small and invertible in $\mathcal{R}_q$. Multiplying the right hand-side of (16) by g yields:

$$(g \cdot f') \cdot \mathbf{cn} = \mathsf{Com}_{\mathbf{A}} \left(g \cdot f' \cdot \$, f' \cdot \sum_{i=0}^{\ell_\$ - 1} 2^i h_i \cdot \mathbf{r}_i - g \cdot \mathbf{r} \right). \tag{17}$$

We get the desired 'small' range proof witness $\Big(f = g \cdot f', \$, \mathbf{r}' = f' \cdot \sum_{i=0}^{\ell_\$ - 1} 2^i h_i \cdot \mathbf{r}_i - g \cdot \mathbf{r}\Big)$, where $\|\mathbf{r}'\| \leq \beta_{range}$ and using estimations from Lemma 4,

$$\begin{aligned}
\beta_{range} &= \left\| f \cdot \sum_{i=0}^{\ell_\$ - 1} h_i \cdot \mathbf{r}_i - g \cdot \mathbf{r} \right\|_2 \leq \left\| f \cdot \sum_{i=0}^{\ell_\$ - 1} h_i \cdot \mathbf{r}_i \right\|_2 + \|g \cdot \mathbf{r}\|_2 \\
&\leq \sqrt{n} \cdot \|f\|_\infty \cdot \left\| \sum_{i=0}^{\ell_\$ - 1} 2^i h_i \cdot \mathbf{r}_i \right\|_\infty + \sqrt{n} \cdot \|g\|_\infty \cdot \|\mathbf{r}\|_\infty \\
&\leq \sqrt{n} \cdot 2\sqrt{\kappa} \cdot \sqrt{n} \cdot 2^{\ell_\$} \cdot 2\sqrt{n(m-1)}\sigma_{OR} + \sqrt{n} \cdot 2\sqrt{\kappa} \cdot 2\beta_v \\
&\leq 2^{\ell_\$ + 2} n \sqrt{\kappa n (m-1)}\sigma_{OR} + 2^2 \sqrt{n\kappa}\beta_v. \qquad (18)
\end{aligned}$$

SHVZK: Here we have to show that our range proof from **??** satisfies the requirement of perfect simulation. Since the underlying OR proof is perfectly simulatable as showed in the last proof of Theorem 3, we only need to show that the underlying proof of knowledge from Table 3 is simulatable too. Given a challenge f', the simulator aborts with probability $1 - 1/M^2$. Otherwise, we have to show the PoK is zero-knowledge. To do so the simulator picks $\overline{\mathbf{r}} \leftarrow D^{n(m-1)}_{12n\sqrt{n(m-1)}}$ and computes $\mathbf{A} \cdot \overline{\mathbf{r}} - f'D$ to guarantee that the verification equation $f' = H(\mathbf{A} \cdot \overline{\mathbf{r}} - f'D)$ is satisfied. The simulator outputs simulated transcript $\overline{\mathbf{r}}, f'$, which is indistinguishable by rejection sampling (1) and by hiding property declared in Theorem 1 of our commitment scheme. □

B LRCT - Proofs of the Security Analysis

B.1 LRCT - Balance (Theorem 5)

Proof. By definition of successful balance attack (Def. 6), $\exists i \in [w]$ such that $\sum_{k \in E^{i^*}_{(in)}} \$^{(k)}_{(in),i^*} < \sum_{j \in G^{i^*}_{(out)}} \$^{(j)}_{(out),i^*}$, being i^* a dishonest transaction. For this analysis we consider three cases, case 1.1, case 1.2 and case 2:

- **Case 1 -** $TN \in \mathcal{TN}$ **from ActGen**: we consider two sub-cases, the outsider and insider attacks which are described as follows:
  - **Case 1.1 - The outsider attack**: $\forall i \in [w]\ \exists k^* \in [N_{in}]$ such that $IW_i^{(k^*)}$ is not corrupted, this means that not all inputs to $\mathcal{T}_{i^*}$ are corrupted. We show that given any PPT MIMO.LRCT adversary, we can construct a MIMO.L2RS adversary, which has equal advantage. In doing this reduction, we firstly define the entities interacting to prove LRCT-Unforgeability. We use a challenger, MIMO.L2RS.Challenger, and two adversaries MIMO.L2RS($\mathcal{B}$) and MIMO.LRCT($\mathcal{A}$). This experiment begins with the challenger who generates the Pub-Params $\leftarrow$ MIMO.L2RS.Setup(λ), and these Pub-Params are given to the adversary $\mathcal{B}$. This adversary then runs $\mathcal{A}$, by simulating $\mathcal{A}$'s oracle answers (Def. 6). We assume that $\mathcal{A}$ makes at most q_{ad}, q_{ac}, q_{co} and q_{spend} queries to AddGen, ActGen, Corrupt and O-Spend respectively. This simulation runs as follows:
    - AddGen(i): on input a query number i, $\mathcal{B}$ forwards the query to its own $\mathcal{JO}$ and obtains the public-key(s) $\mathsf{pk}_i^{(k)}$. $\mathcal{B}$ returns these to $\mathcal{A}$.
    - ActGen($i, \$_i$): on input address index i and an amount $\$_i$, $\mathcal{B}$ runs algorithm MIMO.LRCT.Mint(Pub-Params, $\$_i$) and returns the account $IW_i = \big(\mathsf{pk}_i^{(k)}, \mathbf{cn}_i^{(k)}\big)$ and its corresponding $\mathbf{ck}_i^{(k)}$ to $\mathcal{A}$.
    - O-Spend($\mu, IW, IW_i, OA, \$^{(j)}_{(out)}$, Pub-Params): on input the transaction strings μ, input wallet IW containing IW_i, output addresses OA, and Pub-Params, $\mathcal{B}$ creates a signature by calling its signing oracle as: $\sigma_{L'}(\mu)_i \leftarrow \mathcal{SO}(w, IW, \mathsf{pk}_i^{(k)}, \mu)$, then $\mathcal{B}$ builds the MIMO.LRCT.Spend output as $(TX_i, \mathsf{sig}_i, TN_i)$, where $TX = (\mu, IW, OA)$, TN_i is the linking tag, and $\mathsf{sig}_i = (\sigma_{L'}(\mu)_i, \{\sigma^{(j)}_{range}\}_{j \in [N_{out}]})$. These outputs are returned to $\mathcal{A}$.

$*$ $\mathsf{Corrupt}(i)$: on input query number i, $\mathcal{B}$ calls its corruption oracle to obtain the private key $\mathsf{sk}_i^{(k)} \leftarrow \mathcal{CO}(\mathsf{pk}_i^{(k)})$. This private-key is returned to $\mathcal{A}$.

$\mathcal{A}$ outputs a forgery transaction $(TX^*, \mathsf{sig}_\pi^*, TN^*)$ such $\mathsf{MIMO.LRCT.Verify}(TX^*, \mathsf{sig}_\pi^*, TN^*) = 1$ where $\mathsf{sig}_\pi^* = (\sigma_{L'}(\mu)_\pi^*, \{\sigma_{range}^{(j)}\}_{j\in[N_{out}]}^*)$. $\mathcal{B}$ also outputs its forgery $\sigma_{L'}(\mu)_\pi^*$ and IW^*, where IW^* is the input list in TX^*. We show that the advantage of $\mathsf{MIMO.L2RS}(\mathcal{B})$ adversary is equal as the advantage of $\mathsf{MIMO.LRCT}(\mathcal{A})$ to break the unforgeability property. In this simulation, $\mathcal{A}$'s view is perfectly simulated by $\mathcal{B}$ as in the real balance game. Moreover, in the event where $\mathcal{A}$ wins the game and case 1.1 occurs, then $\mathcal{B}$ also wins its game. This forgery meets the conditions of both definitions, the MIMO.L2RS one-time unforgeability (Def. 9) and balance (Def. 6), which we summarise below:

1. In both views $\mathsf{MIMO.LRCT.Verify}(\cdot) = 1$ (Def. 6 Cond. 1) and $\mathsf{MIMO.L2RS.SigVer}(\cdot) = 1$ (Def. 9, Cond. 1), transaction signatures must be valid.
2. The $\mathsf{pk}_i^{(k)}$ of the list accounts were generated during the simulation by AddGen oracle (Def. 6 Cond. 2) and this oracle forwarded queries to the MIMO.L2RS's oracle $\mathcal{JO}(\cdot)$ (Def. 9, Cond. 2).
3. The forgeries sig_π^* and $\sigma_{L'}(\mu)_\pi^*$ are not the output of the $\mathsf{O\text{-}Spend}(\cdot)$ (Def. 6 Cond. 3) and $\mathcal{SO}(\cdot)$ (Def. 9, Cond. 3) oracles, respectively.
4. $\mathsf{pk}_\pi^{(k)}$ was only queried to $\mathsf{O\text{-}Spend}(\cdot)$ oracle once (Def. 6 Cond. 4), and thus only a query was forwarded to $\mathcal{SO}(\cdot)$ (Def. 9, Cond. 4).
5. The condition of this case 1.1 ($\forall i \in [w]\ \exists k^* \in [N_{in}]$ such that $IW_i^{(k^*)}$) implies that $\exists k^*$ *s.t.* $\mathsf{pk}_i^{(k^*)}$ is not corrupted (Def. 9, Cond. 5). Therefore, this also meets the condition of the MIMO.L2RS.

To sum up, if the outsider adversary breaks this case 1.1 attack, then we refer to the Theorem 9 (Unforgeability) where the security analysis reduces to the $\mathbf{MSIS}_{q,m,k,\beta}^{\mathcal{K}}$ problem with $\beta = 2\beta_v$. Thus we said that $\beta_{case1.1} = 2\beta_v$.

- **Case 1.2 - The insider attack**: $\exists i \in [w]\ \forall k^* \in [N_{in}]$ such that $IW_i^{(k^*)}$ is corrupted, meaning that all inputs to $\mathcal{T}_{i^*}$ are corrupted. We start this case by running the extractor of the MIMO.L2RS's proof of knowledge (in Proposition 2) so we can extract the witness of this signature relation as $\mathbf{a}_{(in),i^*}^{(N_{in}+1)} \cdot \mathbf{v}_{i^*,(2)}^{(N_{in}+1)} = \mathsf{Com}_{\mathbf{A}}\big(0, \mathbf{v}_{i^*,(1)}^{(N_{in}+1)}\big)$ with $(\mathbf{v}_{i^*,(1)}^{(N_{in}+1)}, \mathbf{v}_{i^*,(2)}^{(N_{in}+1)})^T \neq \mathbf{0} \bmod q$. For simplicity, we define $\mathbf{g}_{L2RS} \triangleq \mathbf{v}_{i^*,(2)}^{(N_{in}+1)}$ and $\mathbf{r} \triangleq \mathbf{v}_{i^*,(1)}^{(N_{in}+1)}$. Then, we have $\mathbf{a}_{(in),i^*}^{(N_{in}+1)} = \mathbf{g}_{L2RS}^{-1} \cdot \mathsf{Com}_{\mathbf{A}}\big(0, \mathbf{r}\big) = \sum_{k=1}^{N_{in}} \mathbf{a}_{(in),i^*}^{(k)} + \mathbf{cn}_{(in),i^*}^{(k)} - \sum_{j=1}^{N_{out}} \mathbf{cn}_{(out),i^*}^{(j)}$ from definition of $\mathbf{a}_{(in)}^{(N_{in}+1)}$ in (5) [from Section 5]. We said that $\sum_{j=1}^{N_{out}} \mathbf{cn}_{(out),i^*}^{(j)} \triangleq \sum_{j\in G_{(out)}^{i^*}} \mathbf{cn}_{(out),i^*}^{(j)} + \sum_{j\in E_{(out)}^{i^*}} \mathbf{cn}_{(out),i^*}^{(j)}$ where G_{out} and E_{out} are $\mathcal{T}_{i^*}$'s not corrupted and corrupted outputs, respectively. Then, replacing with this definition, we have $\mathbf{g}_{L2RS}^{-1} \cdot \mathsf{Com}_{\mathbf{A}}\big(0, \mathbf{r}\big) = \mathsf{Com}_{\mathbf{A}}\left(\sum_{k=1}^{N_{in}} \$_{(in),i^*}^{(k)} - \sum_{j\in G_{(out)}^{i^*}} \$_{(out),i^*}^{(j)}, \mathbf{S}_{(in),i^*}^{(N_{in}+1)}\right) + \sum_{j\in E_{(out)}^{i^*}} \mathbf{cn}_{(out),i^*}^{(j)}$. The latter equation is equivalent to $\sum_{j\in E_{(out)}^{i^*}} \mathbf{cn}_{(out),i^*}^{(j)} = \mathbf{g}_{L2RS}^{-1} \cdot \mathsf{Com}_{\mathbf{A}}\left(\sum_{k=1}^{N_{in}} \$_{(in),i^*}^{(k)} - \sum_{j\in G_{(out)}^{i^*}} \$_{(out),i^*}^{(j)}, \mathbf{r} - \mathbf{S}_{(in),i^*}^{(N_{in}+1)}\right)$. Afterwards, we multiply both sides by $\mathbf{g}_{L2RS}$ and it results in:

$$\mathbf{g}_{L2RS} \cdot \sum_{j\in E_{(out)}^{i^*}} \mathbf{cn}_{(out),i^*}^{(j)} = \mathsf{Com}_{\mathbf{A}}\left(\mathbf{g}_{L2RS} \cdot \Delta, \overline{\mathbf{r}}\right), \tag{19}$$

where $\overline{\mathbf{r}} \triangleq \mathbf{g}_{L2RS} \cdot (\mathbf{r} - \mathbf{S}_{(in),i^*}^{(N_{in}+1)})$ and $\Delta \triangleq \sum_{k=1}^{N_{in}} \$_{(in),i^*}^{(k)} - \sum_{j\in G_{(out)}^{i^*}} \$_{(out),i^*}^{(j)}$. Since $\$_{(in),i^*}^{(k)} \in [0, 2^{\ell_\$} - 1]$ and $\max(N_{in}, N_{out}) \leq N$, then $\sum_{k=1}^{N_{in}} \$_{(in),i^*}^{(k)} \in [0, N \cdot 2^{\ell_\$} - 1]$ and $\sum_{j\in G_{(out)}^{i^*}} \$_{(out),i^*}^{(j)} \in [0, N \cdot 2^{\ell_\$} - 1]$ where $N_{in} \leq N$ and $N_{out} \leq N$, respectively. We have, $\left|\$_{(in),i^*}^{(k)} - \$_{(out),i^*}^{(j)}\right| \leq$

$N \cdot (2^{\ell_\$} - 1)$ and $\$^{(k)}_{(in),i^*} - \$^{(j)}_{(out),i^*} < 0$, which is less than $q/2$ by the choice of q. Therefore, $\Delta \bmod q = \Delta \in [-N \cdot (2^{\ell_\$} - 1), -1]$. We now run the proof of knowledge extractor of parallel range proofs from Theorem 4, $\forall j \in [E_{out}]$ $\mathbf{cn}^{(j)}_{(out),i^*}$. We then obtain:

$$\mathbf{g}_{Range} \cdot \sum_{j \in E^{i^*}_{(out)}} \mathbf{cn}^{(j)}_{(out),i^*} = \mathsf{Com}_{\mathbf{A}} \left(\mathbf{g}_{Range} \cdot \overline{\$}, \overline{\mathbf{ck}} \right), \tag{20}$$

where $\mathbf{g}_{Range} = f, \overline{\$} \triangleq \sum_{j \in E^{i^*}_{(out)}} \overline{\$}^{(j)}_{(out)}$ and the randomness $\overline{\mathbf{ck}} \triangleq \sum_{j \in E^{i^*}_{(out)}} \overline{\mathbf{ck}}^{(j)}_{(out)}$, as per in (17). If we multiply and subtract both equations (19) and (20) by $\mathbf{g}_{Range}$ and $\mathbf{g}_{L2RS}$, respectively, it results to $0 = \mathsf{Com}_{\mathbf{A}} \left(\mathbf{g}_{Range} \cdot \mathbf{g}_{L2RS} \cdot (\Delta - \overline{\$}), \mathbf{g}_{Range} \cdot \overline{\mathbf{r}} - \mathbf{g}_{L2RS} \cdot \overline{\mathbf{ck}} \right)$. Assuming that $\|\mathbf{g}_{Range} \cdot \mathbf{g}_{L2RS}\| < \frac{1}{\sqrt{k}} \cdot q^{1/k}$ where k denotes the number of irreducible factors mod$q$ of $x^n + 1$, then by [Corollary 1.2 from 2.[21]], $\mathbf{g}_{Range} \cdot \mathbf{g}_{L2RS}$ is invertible in $\mathcal{R}_q$. This implies that $\mathbf{g}_{Range} \cdot \mathbf{g}_{L2RS} \cdot (\Delta - \overline{\$}) \neq 0 \bmod q$, using the fact that $\Delta - \overline{\$} \neq 0 \bmod q$. Therefore, we obtain a $\beta-$binding collision for $\mathsf{Com}_{\mathbf{A}}$ with $\beta-$binding $= \left\| \mathbf{g}_{Range} \cdot \overline{\mathbf{r}} - \mathbf{g}_{L2RS} \cdot \overline{\mathbf{ck}} \right\| \leq \beta_{case1.2}$. By replacing this $\beta-$binding with the results of the witness extraction, it turns out that $\beta_{case2.2} \geq 4\sqrt{\kappa(2\beta_v)^2 + \kappa(2\beta_v)^2 n(m-1)((2N_{in} + N_{out})2^\gamma)^2} + 2\beta_v N_{out}(2^{\ell_\$ + 2} n\sqrt{\kappa n(m-1)}\sigma_{OR} + 2^2\sqrt{n}\beta_v)$.

– **Case 2 -** $TN \notin \mathcal{TN}$ **from ActGen (Linkability Attack)**: $\exists k^* \in [N_{in}]$ *s.t.* IW_i^{k*} with $i \in [w]$ was queried to O-Spend, where k^*'th is the real input account in the forgery transaction with TN, and $TN \not\subseteq \mathcal{TN}$.
In this proof, we show that any PPT MIMO.LRCT adversary has equal advantage to the corresponding MIMO.L2RS adversary. In doing this reduction, we firstly define the entities interacting to prove the LRCT-Linkability. We use a challenger, MIMO.L2RS.Challenger, and two adversaries MIMO.L2RS($\mathcal{B}$) and MIMO.LRCT($\mathcal{A}$). This experiment begins with the challenger who generates the Pub-Params $\leftarrow$ MIMO.L2RS.Setup(λ), and these are given to the adversary $\mathcal{B}$. This adversary then runs $\mathcal{A}$, by simulating $\mathcal{A}$'s oracle answers (see Section 3.1). We assume that $\mathcal{A}$ makes at most q_{ad}, q_{ac} queries to AddGen and ActGen respectively, then by querying the oracle O-Spend, it will generate a signature or PoK. This simulation runs as follows:

- AddGen(i): on input a query number i, $\mathcal{B}$ forwards the query to its own $\mathcal{JO}$ and obtains the public-key(s) $\mathsf{pk}^{(k)}_i$. $\mathcal{B}$ returns these to $\mathcal{A}$.
- ActGen($i, \$_i$): on input address index i and an amount $\$_i$, $\mathcal{B}$ runs algorithm LRCT.Mint(Pub-Params, $\$_i$) and returns the account $IW_i = \left(\mathsf{pk}^{(k)}_i, \mathbf{cn}^{(k)}_i\right)$ and its corresponding $\mathbf{ck}^{(k)}_i$ to $\mathcal{A}$.
- O-Spend($\mu, IW, IW_i, OA, \$^{(j)}_{(out)}$, Pub-Params): on input the transaction strings μ, input wallet IW containing IW_i, output addresses OA, and Pub-Params, $\mathcal{B}$ creates a signature by calling its signing oracle as: $\sigma_{L'}(\mu)_i \leftarrow \mathcal{SO}(w, IW, \mathsf{pk}^{(k)}_i, \mu)$, then $\mathcal{B}$ builds the MIMO.LRCT.Spend output as $(TX_i, \mathsf{sig}_i, TN_i)$, where $TX = (\mu, IW, OA)$, TN_i is the linking tag, and $\mathsf{sig}_i = (\sigma_{L'}(\mu)_i, \{\sigma^{(j)}_{range}\}_{j \in [N_{out}]})$. These outputs are returned to $\mathcal{A}$.
- Corrupt(i): on input query number i, $\mathcal{B}$ calls its corruption oracle to obtain the private key $\mathsf{sk}^{(k)}_i \leftarrow \mathcal{CO}(\mathsf{pk}^{(k)}_i)$, and this private-key is returned to $\mathcal{A}$.

$\mathcal{A}$ outputs two transaction forgeries $(TX^*, \mathsf{sig}^*_\pi, TN^*)$ and $(TX'^{,*}, \mathsf{sig}'^{,*}_\pi, TN'^{,*})$, whereas $\mathcal{B}$ outputs two signature forgeries $\sigma_{L^*}(\mu)^*_\pi$ and $\sigma_{L^*}(\mu)'^{,*}_\pi$ with their corresponding IW^* and $IW'^{,*}$ which were taken from TX^* and $TX'^{,*}$, respectively. These forgeries meet the conditions of the balance MIMO.LRCT (Def. 6) and the MIMO.L2RS linkability definition (Def. 11), and we summarise these as:

1. In both views MIMO.LRCT.Verify($\cdot$) $= 1$ (Def. 6 Cond. 1) and MIMO.L2RS.SigVer($\cdot$) $= 1$ (Def. 11, Cond. 1), transaction signatures must be valid.

2. The $\mathsf{pk}_i^{(k)}$ of the list accounts were generated during the simulation by AddGen oracle (Def. 6 Cond. 2) and this oracle forwarded queries to the MIMO.L2RS's oracle $\mathcal{JO}(\cdot)$ (Def. 11, Cond. 2).
3. Condition 3 of (Def. 6) implies $\mathsf{MIMO.L2RS.SigLink}(\sigma_{L^*}(\mu)^*_\pi, \sigma_{L^*}(\mu)'^*_\pi) = Unlinked$ (Def. 11, Cond. 3).

We showed that the advantage of $\mathsf{MIMO.L2RS}(\mathcal{B})$ adversary is equal as $\mathsf{MIMO.LRCT}(\mathcal{A})$ to break the linkability property. Then, we refer to the Theorem 11 (Linkability) where the security analysis reduces to the $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,\beta}$ problem with $\beta = 2\beta_v$. Thus we said that $\beta_{case2} = 2\beta_v$. □

B.2 LRCT - Anonymity - Proof

Proof. We prove the anonymity of this scheme using the sequence-of-games approach. We begin our analysis by:

Game 0 - Real Game : We firstly define the entities interacting to prove this LRCT-Anonymity property. We use a challenger, MIMO.LTCR.Challenger, and two adversaries, $\mathsf{MIMO.LRCT}(\mathcal{A}_1)$ and $\mathsf{MIMO.LRCT}(\mathcal{A}_2)$. This experiment begins with the challenger who generates the $\mathsf{Pub\text{-}Params} \leftarrow \mathsf{MIMO.L2RS.Setup}(\lambda)$, and this output is given to the adversary $\mathcal{A}_1$. Then, $\mathcal{A}_1$ runs the oracles, which were defined in Def. 7. We assume that $\mathcal{A}_1$ makes at most q_{ad}, q_{ac}, and q_{co} queries to AddGen, ActGen, and Corrupt, respectively. This simulation runs as follows:

- $\mathsf{AddGen}(i)$: on input a query number i, it returns the public-key(s) $\mathsf{pk}_i^{(k)}$.
- $\mathsf{ActGen}(i, \$_i)$: on input address index i and an amount $\$_i$, $\mathcal{A}_1$ runs algorithm $\mathsf{LRCT.Mint}(\mathsf{Pub\text{-}Params}, \$_i)$ and returns the account $IW_i = \left(\mathsf{pk}_i^{(k)}, \mathbf{cn}_i^{(k)}\right)$ and its corresponding $\mathbf{ck}_i^{(k)}$.
- $\mathsf{O\text{-}Spend}(\mu, IW, IW_i, OA, \{\$^{(j)}_{(out),i}\}_{j\in[N_{out}]}, \mathsf{Pub\text{-}Params})$, and it outputs $(TX, \mathsf{sig}_i, TN_i)$.
- $\mathsf{Corrupt}(i)$: on input query number i, it outputs $(\mathsf{sk}_i^{(k)}, \mathsf{ck}_i^{(k)})$.

Now $\mathcal{A}_1$ construct IW with w accounts from the ActGen's queries q_{ac}, then it selects two elements π_0 and π_1 from IW, such $IW_{\pi_0} = \{\mathsf{pk}^{(k)}_{\pi,0}, \mathbf{cn}^{(k)}_{\pi,0}\}_{k\in[N_{in}]}$ and $IW_{\pi_1} = \{\mathsf{pk}^{(k)}_{\pi,1}, \mathbf{cn}^{(k)}_{\pi,1}\}_{k\in[N_{in}]}$, with $\mathsf{pk}^{(k)}_{\pi,0} = \mathsf{Com}_{\mathbf{A}}(\mathbf{0}, \mathbf{sk}^{(k)}_{\pi,0})$, $\mathsf{pk}^{(k)}_{\pi,1} = \mathsf{Com}_{\mathbf{A}}(\mathbf{0}, \mathbf{sk}^{(k)}_{\pi,1})$, $\mathbf{cn}^{(k)}_{\pi,0} = \mathsf{Com}_{\mathbf{A}}(\$^{(k)}_{(in),0}, \mathbf{ck}^{(k)}_{\pi,0})$ and $\mathbf{cn}^{(k)}_{\pi,1} = \mathsf{Com}_{\mathbf{A}}(\$^{(k)}_{(in),1}, \mathbf{ck}^{(k)}_{\pi,1})$. After this $\mathcal{A}_1$ outputs $(\mu, IW_{\pi_0}, IW_{\pi_1}, IW, OA, \$^{(j)}_{(out),0}, \$^{(j)}_{(out),1})$, where $\sum_{k=1}^{N_{in}} \$^{(k)}_{(in),0} = \sum_{j=1}^{N_{out}} \$^{(j)}_{(out),0}$ and $\sum_{k=1}^{N_{in}} \$^{(k)}_{(in),1} = \sum_{j=1}^{N_{out}} \$^{(j)}_{(out),1}$. The MIMO.LRCT.Challenger picks at random $b = \{0, 1\}$ and returns $(TX^*, \mathsf{sig}^*_b, TN^*_b) \leftarrow \mathsf{RCT.Spend}(\mu, K_{\pi_b}, IW_{\pi_b}, IW, OA, \$_{(out)_b}\mathsf{Pub\text{-}Params})$ where $IW_{\pi_b} = \{\mathbf{pk}^{(k)}_{\pi_b}, \mathbf{cn}^{(k)}_{\pi_b}\}$ and $\mathbf{cn}^{(k)}_{\pi_b} = \mathsf{Com}_{\mathbf{A}}(\$^{(k)}_{(in),\pi_b}, \mathbf{ck}^{(k)}_{\pi_b})$ to $\mathcal{A}_2$. The adversary $\mathcal{A}_2$ runs the oracles as (Def. 7):

- $\mathsf{O\text{-}Spend}(\mu, IW', IW'_\pi, OA, \$^{(j)}_{(out)}, \mathsf{Pub\text{-}Params})$ with $IW' \neq IW$ and $IW'_\pi \neq W_{\pi_0}, W_{\pi_1}$. This outputs $(TX^{*\prime}, \mathsf{sig}^{*\prime}_{b'}, TN^{*\prime}_{b'})$, with $TN^{*\prime}_{b'} = \mathsf{Com}_{\mathbf{H}}(\mathbf{0}, \mathbf{sk}^{(k)\prime}_{\pi,b'})$
- $\mathsf{Corrupt}(i)$: on input query number i, it returns $(\mathsf{sk}_i^{(k)}, \mathsf{ck}_i^{(k)})$.

The adversary $\mathcal{A}_2$ outputs b'. If we define the S_0 to be the event that $b = b'$, then the $\mathcal{A}_2$'s advantage is $|\Pr[S_0] - \frac{1}{2}|$.

Game 1 - Signature : In this game, we perform changes in the signature $\mathsf{sig}^*_b = (\sigma_{L'}(\mu)_b, \{\sigma^{(j)}_{range_b}\}_{j\in[N_{out}]})$, in particular $\sigma_{L'}(\mu)_b$. Instead of generating this real signature with MIMO.L2RS.SigGen (Algorithm 2), we use the hybrids MIMO.L2RS.Hybrid-1 and MIMO.L2RS.Hybrid-2, Algorithms 11 and 12, respectively; based on our security analysis in Appendix (MIMO.L2RS unforgeability). In the transition from the real signature to hybrid 1, the $(\mathbf{c}_{\pi+1})_b$ is chosen at random. This transition concluded that the statistical distance between $\mathbf{c}_{\pi+1}$ and $\mathcal{S}_{n,\kappa} \subseteq \mathcal{R}_{2q}$ will be at most $\epsilon_{Game1} = o \cdot h \cdot 2^{-n+1}$, which is negligible (Based on [13], Lemma 3.4), where h and o are the number of queries to H_1 and the

hybrid 1, respectively. We now consider the transition from hybrid 1 to hybrid 2. The output of both hybrids follows the same distribution due to the rejection sampling (Lemma 1). This means that choosing $\mathbf{t}_\pi$ at random, will not have any effect in the output of both hybrids. Let S_0 be the event that $b = b'$ in Game 1. We claim that the view of the adversary in Game 0 and Game 1 is:

$$|\Pr[S_0] - \Pr[S_1]| \leq \epsilon_{Game1}. \tag{21}$$

Game 2 - User Anonymity ($\boldsymbol{\pi_0 \neq \pi_1}$ with $\boldsymbol{\$_{(out),0} = \$_{(out),1}}$) : Changes in this game are made to TN_b^* and $\mathbf{pk}_{\pi_b}$. TN_b^* is now randomly chosen from $\mathcal{R}_q^2$. When $b' = 0$, then $\mathbf{pk}_{\pi_0} \leftarrow \mathcal{R}_q^2$ whereas $\mathsf{pk}_{\pi,1} = \mathsf{Com}_{\mathbf{A}}(\mathbf{0}, \mathbf{sk}_{\pi,1})$. When $b' = 1$ then $\mathbf{pk}_{\pi_1} \leftarrow \mathcal{R}_q^2$ whereas $\mathsf{pk}_{\pi,0} = \mathsf{Com}_{\mathbf{A}}(\mathbf{0}, \mathbf{sk}_{\pi,0})$. When $\mathbf{sk}_{\pi,b}$ is multiplied by $\mathbf{H}$ and $\mathbf{A}$ respectively, it gives TN_b^* and $\mathbf{pk}_{\pi_b}$ that are close to uniform over $\mathcal{R}_q^2$. By applying the Leftover Hash Lemma (LHL) - **Lemma 5**, the statistical distance between the distribution of (TN_b^* mod q and $\mathbf{pk}_{\pi_b}$ mod q) and the uniform distribution on $\mathcal{R}_q^2 \times \mathcal{R}_q^2$ is at most $\left(n \cdot \frac{1}{2} \cdot \sqrt{\frac{q^{4n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}\right)$, which is negligible. Let S_2 be the event that $b = b'$ in Game 2. We claim that

$$|\Pr[S_1] - \Pr[S_2]| \leq \epsilon_{Game2}. \tag{22}$$

Game 3 - User Anonymity ($\boldsymbol{\pi_0 \neq \pi_1}$ with $\boldsymbol{\$_{(out),0} = \$_{(out),1}}$) : We now transform Game 1 into Game 2, where we choose $\mathbf{pk}_{\pi_{1-b}}$ at random. This means that when $b' = 0$, then $\mathbf{pk}_{\pi_1} \leftarrow \mathcal{R}_q^2$ and when $b' = 1$ then $\mathbf{pk}_{\pi_0} \leftarrow \mathcal{R}_q^2$. We conclude that by applying the Leftover Hash Lemma (LHL) - **Lemma 5**, the statistical distance between the distribution of ($\mathbf{pk}_{\pi_b}$ mod q) and the uniform distribution on $\mathcal{R}_q^2$ is at most $\left(n \cdot \frac{1}{2} \cdot \sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}\right)$ which is negligible. Let S_3 be the event that $b = b'$ in Game 3. We claim that

$$|\Pr[S_2] - \Pr[S_3]| \leq \epsilon_{Game3}. \tag{23}$$

Game 4 - Amount Privacy ($\boldsymbol{\pi_0 = \pi_1}$ with $\boldsymbol{\$_{(out),0} \neq \$_{(out),1}}$) : In this transitional Game, we choose $\mathbf{cn}_{\pi_b}$ at random, instead of computing $\mathbf{cn}_{\pi_b} = \mathsf{Com}_{\mathbf{A}}(\$_{(out),\pi_b}, \mathbf{ck}_{\pi_b})$. We use the result of the Theorem 1 (Homomorphic Commitment Hiding), to show that by applying the Leftover Hash Lemma (Lemma 5), we argue that the statistical distance between the distribution of $\mathbf{cn}_{\pi_b}$ and the uniform distribution on $\mathcal{R}_q^2$ is at most $\left(\frac{1}{2} \cdot \sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}\right)$ which is negligible. Let S_4 be the event that $b = b'$ in Game 4, then we claim that

$$|\Pr[S_3] - \Pr[S_4]| \leq \epsilon_{Game4}. \tag{24}$$

Combining (21), (22), (23) and (24), we obtain

$$\left|\Pr[S_0] - \frac{1}{2}\right| \leq \epsilon_{Game1} + \epsilon_{Game2} + \epsilon_{Game3} + \epsilon_{Game4},$$

and this is negligible. □

C MIMO.L2RS Security model

C.1 MIMO.L2RS Definitions

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

- MIMO.L2RS.Setup: a PPT algorithm that takes the security parameter λ and produces the Public Parameters (Pub-Params).

- MIMO.L2RS.KeyGen: a PPT algorithm that by taking the Pub-Params, it produces a pair of keys: the public-key pk and the private-key sk.
- MIMO.L2RS.SigGen: a PPT algorithm that receives the Pub-Params, a singer π's sk, a message μ and the list L of users' pk's in the ring signature, and outputs a signature $\sigma_L(\mu)$. w is defined as the size of the ring and N_{in} as the number of input wallets (used in the MIMO.LRCT protocol).

$$L \triangleq \left\{ \mathsf{pk}_i^{(k)} \right\}_{i\in[w],k\in[N_{in}]} \tag{25}$$

- MIMO.L2RS.SigVer: a PPT algorithm that takes Pub-Params, a signature $\sigma_L(\mu)$, a list L of pk's and the message μ, and it verifies if this signature was legitimately created, this algorithm outputs either: *Accept* or *Reject*.
- MIMO.L2RS.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:

- MIMO.L2RS.SigGen Correctness: this guarantees that valid signatures signed by honest signers will be accepted by a verifier with overwhelming probability.
- MIMO.L2RS.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.

C.2 Oracles for adversaries

The following oracles are available to any adversary who tries to break the security of an MIMO.L2RS scheme:

- $\mathsf{pk}_i^{(k)} \leftarrow \mathcal{JO}(\perp)$. The *Joining Oracle*, on request, adds new user(s) to the system. It returns the public-key(s) $\mathsf{pk}_i^{(k)}$.
- $\mathsf{sk}_i^{(k)} \leftarrow \mathcal{CO}(\mathsf{pk}_i^{(k)})$. The *Corruption Oracle*, on input a $\mathsf{pk}_i^{(k)}$ that is a query output of $\mathcal{JO}$, returns the corresponding $\mathsf{sk}_i^{(k)}$.
- $\sigma'_L(\mu) \leftarrow \mathcal{SO}(w, L, \mathsf{pk}_\pi^{(k)}, \mu)$. The *Signing Oracle*, on input a group size w, a set L of w $\mathsf{pk}^{(k)}$'s, the signer's $\mathsf{pk}_\pi^{(k)}$, and a message μ, this oracle returns a valid signature $\sigma'_L(\mu)$.

C.3 Threat Model for MIMO.L2RS

- One-time Unforgeability. One time unforgeability for the MIMO.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:
  - $\mathcal{S}$ generates and gives the list L of $\mathsf{pk}^{(k)}$'s to $\mathcal{A}$.
  - $\mathcal{A}$ may query the oracles according to any adaptive strategy.
  - $\mathcal{A}$ gives $\mathcal{S}$ a ring signature size w, a set L of w $\mathsf{pk}^{(k)}$'s, a message μ and a signature $\sigma_L(\mu)$.

  $\mathcal{A}$ wins the game if:
  1. MIMO.L2RS.SigVer$(\sigma_L(\mu))$=**Accept**.
  2. $\mathsf{pk}^{(k)}$'s in the L are outputs from $\mathcal{JO}$ oracle.
  3. $\sigma_L(\mu)$ is not an output of $\mathcal{SO}$.
  4. No signing key $\mathsf{pk}_\pi^{(k)}$ was queried more than once to $\mathcal{SO}$.
  5. $\forall i \in [w]\ \exists k \in [N_{in}]$ *s.t.* $\mathsf{pk}_i^{(k)}$ is not corrupted.

  The advantage of the one-time unforgeability in the MIMO.L2RS scheme is denoted by

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

Definition 9 (One-Time Unforgeability). *The MIMO.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 be infeasible for an adversary $\mathcal{A}$ to distinguish a signer's $\mathsf{pk}^{(k)}$ with probability 1/2, even if the adversary has unlimited computing resources. This property for MIMO.L2RS schemes is defined in the following game between a simulator $\mathcal{S}$ and an unbounded adversary $\mathcal{A}$.
  - $\mathcal{A}$ may query $\mathcal{JO}$ according to any adaptive strategy.
  - $\mathcal{A}$ gives $\mathcal{S}$ the $L = \{\mathsf{pk}_0^{(k)}, \mathsf{pk}_1^{(k)}\}_{k\in[N_{in}]}$, which is the output of the $\mathcal{JO}$, and a message μ.
  - $\mathcal{S}$ flips a coin $b = \{0,1\}$, then $\mathcal{S}$ computes the signature $\sigma_b = \mathsf{MIMO.L2RS.SigGen}(L, \mathsf{sk}_b^{(k)}, \mu, \mathsf{Pub\text{-}Params})$. This signature is given to $\mathcal{A}$.
  - $\mathcal{A}$ outputs a bit b'.
  - The output of this experiment is defined to be 1 if $b = b'$, or 0 "zero" otherwise.

  $\mathcal{A}$ wins the game if:
  1. $\mathsf{pk}_0^{(k)}$ and $\mathsf{pk}_1^{(k)}$ cannot be used by $\mathcal{CO}$ and $\mathcal{SO}$.
  2. Outputs 1, where $b = b'$, with Pr = 1/2.

  The unconditional anonymity advantage of the MIMO.L2RS scheme is denoted by

$$\mathbf{Advantage}_{\mathcal{A}}^{Anon}(\lambda) = \left| \Pr[b = b'] - \frac{1}{2} \right|.$$

  Definition 10 (Unconditional Anonymity). *The MIMO.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 an adversary $\mathcal{A}$ to **unlinked** two valid MIMO.L2RS signatures which were correctly generated with same sk_π. To describe this, we use the interaction between a simulator $\mathcal{S}$ and an adversary $\mathcal{A}$:
  - The $\mathcal{A}$ queries the $\mathcal{JO}$ multiple times.
  - The $\mathcal{A}$ outputs two signatures $\sigma_L(\mu)$ and $\sigma'_{L'}(\mu')$ and two lists L and L' of $\mathsf{pk}^{(k)}$'s.

$$L' \triangleq \left\{ \mathsf{pk}_i'^{(k)} \right\}_{i\in[w],k\in[N_{in}]} \tag{26}$$

  $\mathcal{A}$ wins the game if:
  1. By calling MIMO.L2RS.SigVer on input $\sigma_L(\mu)$ and $\sigma'_{L'}(\mu')$, it outputs **Accept** on both inputs.
  2. The $\mathsf{pk}^{(k)}$'s in L and L' are outputs of $\mathcal{JO}$.
  3. Finally, it gets **Unlinked**, when calling MIMO.L2RS.SigLink on input $\sigma_L(\mu)$ and $\sigma'_{L'}(\mu')$.

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

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

  Definition 11 (Linkability). *The MIMO.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 $\mathcal{A}$ to **linked** two valid MIMO.L2RS signatures which were correctly generated with different $\mathsf{sk}^{(k)}$'s. 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 MIMO.L2RS.SigLink algorithm outputs **Linked**. To describe this, we use the interaction between a simulator $\mathcal{S}$ and an adversary $\mathcal{A}$:
  - The $\mathcal{S}$ generates and gives the list L of $\mathsf{pk}^{(k)}$'s to $\mathcal{A}$.
  - The $\mathcal{A}$ queries the $\mathcal{JO}$ and $\mathcal{CO}$ to obtain $\mathsf{pk}_\pi^{(k)}$ and $\mathsf{sk}_\pi^{(k)}$, respectively.
  - $\mathcal{A}$ gives the generated parameters to $\mathcal{S}$.
  - $\mathcal{S}$ uses the $\mathsf{sk}_\pi^{(k)}$ and calls the $\mathcal{SO}$ to output a valid signature $\sigma_L(\mu)$, which is given to $\mathcal{A}$.
  - 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.

  $\mathcal{A}$ wins the game if:
  1. The MIMO.L2RS.SigVer, on input $\sigma_L(\mu)$ and $\sigma'_L(\mu)$, outputs **Accept**.
  2. The keys $\mathsf{pk}_\pi^{(k)}$ and $\mathsf{sk}_\pi^{(k)}$ were not used to generated the second signature $\sigma'_L(\mu)$.
  3. When calling the MIMO.L2RS.SigLink on input $\sigma_L(\mu)$ and $\sigma'_L(\mu)$, it outputs **Linked**.

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

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

  Definition 12 (Non-Slanderability). *The MIMO.L2RS scheme is* non-slanderable *if for all PPT adversary $\mathcal{A}$,* $\mathbf{Advantage}_{\mathcal{A}}^{NS}$ *is negligible.*

D MIMO.L2RS Scheme construction

The scheme MIMO.L2RS = (MIMO.L2RS.Setup, MIMO.L2RS.KeyGen, MIMO.L2RS.SigGen, MIMO.L2RS.SigVer, MIMO.L2RS.SigLink) works as follows.

D.1 MIMO.L2RS.Setup

By receiving the security parameter λ, this MIMO.L2RS.Setup algorithm randomly chooses $\mathbf{A} \leftarrow \mathcal{R}_q^{2\times(m-1)}$ and $\mathbf{H} \leftarrow \mathcal{R}_q^{2\times(m-1)}$. This outputs the public parameters (Pub-Params): $\mathbf{A}$ and $\mathbf{H}$.

Remark 5. To prevent malicious attack, MIMO.L2RS.Setup incorporates a trapdoor in $\mathbf{A}$ or $\mathbf{H}$, in practice MIMO.L2RS.Setup would generate $\mathbf{A}$ and $\mathbf{H}$ based on the cryptographic Hash function H_2 evaluated at two distinct and fixed constants.

Definition 13 (Function MIMO.L2RS.Lift). *This function maps $\mathcal{R}_q^2$ to $\mathcal{R}_{2q}$ with respect to a public parameter $\boldsymbol{A} \in \mathcal{R}_q^{2\times(m-1)}$. Given $\boldsymbol{a} \in \mathcal{R}_q^2$, we let MIMO.L2RS.Lift$(\boldsymbol{A}, \boldsymbol{a}) \triangleq (2\cdot\boldsymbol{A}, -2\cdot\boldsymbol{a}+\boldsymbol{q}) \in \mathcal{R}_{2q}^{2\times m}$ with $\boldsymbol{q} = q\cdot(1,1)^T$.*

D.2 Key Generation - MIMO.L2RS.KeyGen

This algorithm receives the public parameter Pub-Param: $\mathbf{A} \in \mathcal{R}_q^{2\times(m-1)}$, then it generates a key pair in $\mathcal{R}_q^2$, we:

- Pick $(\mathbf{s}_1, \ldots, \mathbf{s}_{m-1})$ with every component chosen uniformly and independently with coefficients in $(-2^\gamma, 2^\gamma)$.
- Define $\mathbf{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_{m-1})^T \in \mathcal{R}_q^{1\times(m-1)}$.
- Compute $\mathbf{a} = (\mathbf{a}_1, \mathbf{a}_2)^T = \mathbf{A}\cdot\mathbf{S} \bmod q \in \mathcal{R}_q^2$. The $\mathbf{a}$ and $\mathbf{S}$ are the public-key pk and the private-key sk, respectively.

This MIMO.L2RS.KeyGen algorithm is described in the following Algorithm 1.

D.3 Signature Generation - MIMO.L2RS.SigGen

The MIMO.L2RS.SigGen algorithm inputs the user's private-key $\mathbf{S}_{(in),\pi}^{(k)}$, the message μ, the list of user's public-keys L' and the public parameters Pub-Params: $\mathbf{H} \in \mathcal{R}_q^{1\times(m-1)}$ and $\mathbf{A} \in \mathcal{R}_q^{1\times(m-1)}$. This algorithm outputs 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 $L = \{\mathbf{a}_{(in),1}^{(k)}, \ldots, \mathbf{a}_{(in),w}^{(k)}\}$ and the private-key $\mathbf{S}_{(in),\pi}^{(k)}$ which corresponds to $\mathbf{a}_{(in),\pi}^{(k)}$ with $1 \le \pi \le w$ and $k \in [1, N_{in}+1]$; the following computations are performed:

1. We define the linkability tag as $\mathbf{H}_{2q}^{(k)} = \big(2\cdot\mathbf{H}, -2\cdot\mathbf{h}^{(k)} + \mathbf{q}\big) \in \mathcal{R}_{2q}^{2\times m}$, where $\mathbf{H}$ is the fixed public parameter for all users, and $\mathbf{h}^{(k)} = \mathbf{H}\cdot\mathbf{S}_{(in),\pi}^{(k)} \in \mathcal{R}_q^2$. We consider $\mathbf{S}_{(in),\pi}^{(k),T} \in \mathcal{R}_q^{1\times(m-1)}$ as an element in $\mathcal{R}_{2q}$ and let $\mathbf{S}_{(in),2q,\pi}^{(k),T} = \big(\mathbf{S}_{(in),\pi}^{(k),T}, 1\big) \in \mathcal{R}_{2q}^{1\times m}$, such that $\mathbf{H}_{2q}^{(k)}\cdot\mathbf{S}_{(in),\pi}^{(k),T} = q \in \mathcal{R}_{2q}$.
2. The π's public-key is lifted from $\mathcal{R}_q^{1\times m}$ to $\mathcal{R}_{2q}^{1\times m}$, so by calling the lift function MIMO.L2RS.Lift$(\mathbf{A}, \mathbf{a}_{(in),\pi}^{(k)})$, we get $\mathbf{A}_{2q,\pi}^{(k)} = (2\cdot\mathbf{A}, -2\cdot\mathbf{a}_{(in),\pi}^{(k)} + \mathbf{q}) \in \mathcal{R}_{2q}^{2\times m}$.
3. Note that $\mathbf{A}_{2q,\pi}^{(k)}\cdot\mathbf{S}_{(in),\pi}^{(k),T} = q \in \mathcal{R}_{2q}$
4. By choosing a random vector $\mathbf{u}^{(k)} = (u_1, \ldots, u_m)^T$, where $u_i \leftarrow D_\sigma^n$, for $1 \le i \le m$, we calculate $\mathbf{c}_{\pi+1} = H_1\Big(L', \big\{\mathbf{H}_{2q}^{(k)}\big\}_{k\in[N_{in}+1]}, \mu, \big\{\mathbf{A}_{2q,\pi}^{(k)}\cdot\mathbf{u}^{(k)}\big\}_{k\in[N_{in}+1]}, \big\{\mathbf{H}_{2q}^{(k)}\cdot\mathbf{u}^{(k)}\big\}_{k\in[N_{in}+1]}\Big)$.

5. We choose random vector $\mathbf{t}_i^{(k)} = (t_{i,1}, \ldots, t_{i,m})^T$, where $t_{i,j} \leftarrow D_\sigma^n$, for $1 \leq j \leq m$, then for $(i = \pi + 1, \ldots, w, 1, 2, \ldots, \pi - 1)$, after lifting from $\mathcal{R}_q^{1\times m}$ to $\mathcal{R}_{2q}^{1\times m}$, using MIMO.L2RS.Lift$(\mathbf{A}, \mathbf{a}_{(in),i}^{(k)})$, we obtain $\mathbf{A}_{2q,i}^{(k)} = (2\cdot\mathbf{A}, -2\cdot\mathbf{a}_{(in),i}^{(k)} + \mathbf{q}) \in \mathcal{R}_{2q}^{2\times m}$. Then, we compute $\mathbf{c}_{i+1} = H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}_{2q,i}^{(k)} \cdot \mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)}\cdot\mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}\Big)$.
6. Select a random bit $b \in \{0, 1\}$ and finally compute $\mathbf{t}_\pi^{(k)} \leftarrow \mathbf{u}^{(k)} + \mathbf{S}_{2q,\pi}^{(k)} \cdot \mathbf{c}_\pi \cdot (-1)^{b^{(k)}}$ using rejection sampling (Definition 1).
7. Output the signature $\sigma_{L'}(\mu) = \Big(\mathbf{c}_1, \{\mathbf{t}_1^{(k)}, \ldots, \mathbf{t}_w^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{h}^{(k)}\}_{k\in[N_{in}]}\Big)$.

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

Correctness of MIMO.L2RS.SigGen

Proof. Beyond the required conditions of MIMO.L2RS.SigVer, we claim that if $\sigma_{L'}(\mu) = \Big(\mathbf{c}_1, \{\mathbf{t}_1^{(k)}, \ldots, \mathbf{t}_w^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{h}^{(k)}\}_{k\in[N_{in}]}\Big)$ is the output of the MIMO.L2RS.SigGen algorithm on input $(\mu, L, \mathbf{S}_\pi, \text{Pub-Params})$, then the output of MIMO.L2RS.SigVer on input $(\mu, L, \sigma_L(\mu))$ should be accepted. We need to show that when MIMO.L2RS.SigVer computes $H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}_{2q,i}^{(k)} \cdot \mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)}\cdot\mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}\Big)$, the result is equal to $\mathbf{c}_1$. We also show that this $H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}_{2q,i}^{(k)}\cdot\mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)}\cdot\mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}\Big) = \mathbf{c}_{i+1}$ for $1 \leq i \leq w-1$ in MIMO.L2RS.SigVer. In this evaluation, we consider two scenarios, one when $i \neq \pi$ and $i = \pi$:

- For $i \neq \pi$, in MIMO.L2RS.SigGen we have $\mathbf{c}_{i+1} = H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}_{2q,i}^{(k)}\cdot\mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)}\cdot\mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}\Big)$, while in MIMO.L2RS.SigVer we compute $\mathbf{c}_{i+1} = H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}_{2q,i}^{(k)}\cdot\mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)}\cdot\mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}\Big)$. These are equal since $\mathbf{A}_{2q,i}^{(k)}\cdot\mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i$ (in MIMO.L2RS.SigGen) $= \mathbf{A}_{2q,i}^{(k)}\cdot\mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i$ (in MIMO.L2RS.SigVer); and $\mathbf{H}_{2q}^{(k)}\cdot\mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}$ (in MIMO.L2RS.SigGen) $= \mathbf{H}_{2q}^{(k)}\cdot\mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}$ (in MIMO.L2RS.SigVer).
- For $i = \pi$, in MIMO.L2RS.SigGen we have $\mathbf{c}_{\pi+1} = H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}_{2q,\pi}^{(k)}\cdot\mathbf{u}^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)}\cdot\mathbf{u}^{(k)}\}_{k\in[N_{in}+1]}\Big)$, whereas in MIMO.L2RS.SigVer we calculate $\mathbf{c}_{\pi+1} = H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}_{2q,\pi}^{(k)}\cdot\mathbf{t}_\pi^{(k)} + \mathbf{q}\cdot\mathbf{c}_\pi\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)}\cdot\mathbf{t}_\pi^{(k)} + \mathbf{q}\cdot\mathbf{c}_\pi\}_{k\in[N_{in}+1]}\Big)$. In this case, we need to show that $\mathbf{c}_{\pi+1}$ (in MIMO.L2RS.SigGen) $= \mathbf{c}_{\pi+1}$ (in MIMO.L2RS.SigVer). In doing so, the following equalities need to be proved:
  1. $\mathbf{A}_{2q,\pi}^{(k)}\cdot\mathbf{u}^{(k)} = \mathbf{A}_{2q,\pi}^{(k)}\cdot\mathbf{t}_\pi^{(k)} + \mathbf{q}\cdot\mathbf{c}_\pi$, which is equivalent to $\mathbf{A}_{2q,\pi}^{(k)}\cdot(\mathbf{u}^{(k)} - \mathbf{t}_\pi^{(k)}) = \mathbf{q}\cdot\mathbf{c}_\pi$. Here, we replace $\mathbf{t}_\pi^{(k)}$ as defined in Algorithm 2, to obtain:

$$\begin{gathered}\mathbf{A}_{2q,\pi}^{(k)}\cdot(\mathbf{u}^{(k)} - \mathbf{u}^{(k)} + \mathbf{S}_{2q,\pi}^{(k)}\cdot\mathbf{c}_\pi\cdot(-1)^{b^{(k)}}) = \mathbf{q}\cdot\mathbf{c}_\pi \iff \\ -\mathbf{A}_{2q,\pi}^{(k)}\cdot\mathbf{S}_{2q,\pi}^{(k)}\cdot\mathbf{c}_\pi\cdot(-1)^{b^{(k)}} = \mathbf{q}\cdot\mathbf{c}_\pi \iff \\ -\mathbf{q}\cdot\mathbf{c}_\pi\cdot(-1)^b = \mathbf{q}\cdot\mathbf{c}_\pi\end{gathered}$$

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

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

$$\begin{gathered}\mathbf{H}_{2q}^{(k)} \cdot (\mathbf{u}^{(k)} - \mathbf{t}_{\pi}^{(k)}) = \mathbf{q} \cdot \mathbf{c}_{\pi} \iff \\ \mathbf{H}_{2q}^{(k)} \cdot (\mathbf{u}^{(k)} - \mathbf{u}^{(k)} + \mathbf{S}_{2q,\pi}^{(k)} \cdot \mathbf{c}_{\pi} \cdot (-1)^{b^{(k)}}) = \mathbf{q} \cdot \mathbf{c}_{\pi} \iff \\ -\mathbf{H}_{2q}^{(k)} \cdot \mathbf{S}_{2q,\pi}^{(k)} \cdot \mathbf{c}_{\pi} \cdot (-1)^{b} = \mathbf{q} \cdot \mathbf{c}_{\pi} \iff \\ -\mathbf{q} \cdot \mathbf{c}_{\pi} \cdot (-1)^{b} = \mathbf{q} \cdot \mathbf{c}_{\pi}\end{gathered}$$

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

□

D.4 Signature Verification - MIMO.L2RS.SigVer

This is described in Algorithm 3. Furthermore, in the following theorem, we show the bound of β_v which is used in this verification algorithm (MIMO.L2RS.SigVer).

Theorem 8. *Let* $\beta_v = \eta\sigma\sqrt{nm}$ *and* $q/4 > \left(\sqrt{2(\lambda+1)\ln 2 + 2\ln(nm)}\right)\sigma$ *and* $\sigma_{L'}(\mu) = \left(\boldsymbol{c}_1, \left\{\boldsymbol{t}_1^{(k)}, \ldots, \boldsymbol{t}_w^{(k)}\right\}_{k\in[N_{in}+1]}, \left\{\boldsymbol{h}^{(k)}\right\}_{k\in[N_{in}]}\right)$ *be generated based on Algorithm 2. Then the output of Algorithm 3 on input* $\sigma_{L'}(\mu)$ *is accepted with probability* $1 - 2^{-\lambda}$.

Proof. In this proof, we start mentioning that in BLISS [13], for a desired expected rejection and repetition M, if we take the definition of α where $M = e^{\frac{1}{2\alpha^2}}$, then $\mathbf{t}_{\pi}^{(k)}$ will be indistinguishable from D_{σ} if $\sigma \geq \alpha \cdot \|\mathbf{S}_{2q,\pi}^{(k)} \cdot \mathbf{c}_{\pi}\|$ [Section 3.2 in [13]]. We also use [lemma 4.4, parts 1 and 3, in [20]]. The part 3 of this lemma shows that the bound on Euclidean norm $\beta_v = \eta\sigma\sqrt{nm}$, for a given $\eta > 1$, has a probability $\Pr[\|\mathbf{t}_i^{(k)}\|_2 > \eta\sigma\sqrt{nm}] \geq 1 - 2^{\lambda}$. In addition, the bound on infinity norm ($\|\mathbf{t}_i\|_{\infty} < q/4$) is analysed in part 1 of this lemma where its union bound is also considered. It turns out that η is required such $q/4 > \eta\sigma > \left(\sqrt{2(\lambda+1)\ln 2 + 2\ln(nm)}\right)\sigma$, except with probability of $2^{-\lambda}$. □

D.5 Signature Linkability - MIMO.L2RS.SigLink

The MIMO.L2RS.SigLink algorithm, illustrated in Algorithm 10, takes two signatures as input: $\sigma_L(\mu_1)$ and $\sigma'_{L'}(\mu_2)$, and it outputs either *Linked* if these signatures were generated by same signatory, or *Unlinked*, otherwise. Given public-keys' lists L and L', and two signatures: $\sigma_L(\mu_1)$ and $\sigma'_{L'}(\mu_2)$, which can be described as: $\sigma_L(\mu_1) = \left(\mathbf{c}_{1,\mu_1}, \left\{\mathbf{t}_{1,\mu_1}^{(k)}, \ldots, \mathbf{t}_{w,\mu_1}^{(k)}\right\}_{k\in[N_{in}+1]}, \left\{\mathbf{h}_{\mu_1}^{(k)}\right\}_{k\in[N_{in}]}\right)$ and $\sigma'_{L'}(\mu_2) = \left(\mathbf{c}_{1,\mu_2}, \left\{\mathbf{t}_{1,\mu_2}^{(k)}, \ldots, \mathbf{t}_{w,\mu_2}^{(k)}\right\}_{k\in[N_{in}+1]}, \left\{\mathbf{h}_{\mu_2}^{(k)}\right\}_{k\in[N_{in}]}\right)$.

These two signatures must be successfully accepted by the MIMO.L2RS.SigVer algorithm, then one can verify that the linkability property is achieved if the linkability tags ($\mathbf{h}_{\mu_1}^{(k)}$ and $\mathbf{h}_{\mu_2}^{(k)}$) of the above signatures $\sigma_L(\mu_1)$ and $\sigma'_{L'}(\mu_2)$ are equal.

Algorithm 10 L2RS.SigLink - Signature Linkability

Input: $\sigma_L(\mu_1)$ and $\sigma'_{L'}(\mu_2)$
Output: Linked or Unlinked
1: **procedure** MIMO.L2RS.SigLink($\sigma_L(\mu_1), \sigma'_{L'}(\mu_2)$)
2: **if** $\left(\text{MIMO.L2RS.SigVer}(\sigma_L(\mu_1)) = \text{Accept}$ **and** $\text{MIMO.L2RS.SigVer}(\sigma'_{L'}(\mu_2)) = \text{Accept}\right)$ **then** Continue [
3: **else if** $\mathbf{h}_{\mu_1}^{(k)} = \mathbf{h}_{\mu_2}^{(k)}$ **then** Linked
4: **else** Unlinked]
5: **return** Linked or Unlinked

Correctness of MIMO.L2RS.SigLink

Proof. We show that an honest user π who signs two messages μ_1 and μ_2 in the MIMO.L2RS scheme with the list of public-keys L, obtains a *Linked* output from MIMO.L2RS.SigLink algorithm with overwhelming probability. As shown in Algorithm 10, two signatures $\sigma_L(\mu_1)$ and $\sigma_L(\mu_2)$ were created, and then successfully verified by MIMO.L2RS.SigVer. Therefore, the linkability tags $\mathbf{h}_{\mu_1}^{(k)}$ and $\mathbf{h}_{\mu_2}^{(k)}$ must be equal. To prove this, we show that:

$$\begin{gathered}\mathbf{H}_{2q,\mu_1}^{(k)} = \big(2\cdot\mathbf{H}, -2\cdot\mathbf{h}_{\mu_1}^{(k)} + \mathbf{q}\big) \in \mathcal{R}_{2q}^{2\times m}, \text{where} \\ \mathbf{H} = \text{Pub-Param and } \mathbf{h}_{\mu_1}^{(k)} = (\mathbf{H}\cdot\mathbf{S}_{(in),\pi}^{(k)} + \mathbf{q}) \in \mathcal{R}_q^2 \\ \mathbf{H}_{2q,\mu_2}^{(k)} = \big(2\cdot\mathbf{H}, -2\cdot\mathbf{h}_{\mu_2}^{(k)} + \mathbf{q}\big) \in \mathcal{R}_{2q}^{2\times m}, \text{where} \\ \mathbf{H} = \text{Pub-Param and } \mathbf{h}_{\mu_2}^{(k)} = (\mathbf{H}\cdot\mathbf{S}_{(in),\pi}^{(k)} + \mathbf{q}) \in \mathcal{R}_q^2\end{gathered}$$

The first parts of the linkability tag in both MIMO.L2RS signatures have same equality with following probability:

$$Pr\big[2\cdot\mathbf{H} = 2\cdot\mathbf{H}\big] = 1.$$

Ultimately, the second part uses the honest user's private-key $\mathbf{S}_{(in),\pi}^{(k)}$ is used, so we conclude that:

$$Pr\big[-2\cdot\mathbf{h}_{\mu_1}^{(k)} + \mathbf{q} + 2\cdot\mathbf{h}_{\mu_2}^{(k)} - \mathbf{q} = 0\big] = 1.$$

□

E MIMO.L2RS - Security Analysis

Theorem 9 (One-Time Unforgeability). *Suppose* $\sqrt{\frac{q^{4n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$ *is negligible in* n, $\frac{1}{|\mathcal{S}_{n,\kappa}|}$ *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 MIMO.L2RS with non-negligible probability* δ*, then there exist a PPT algorithm that can extract a solution to the* $\mathbf{MSIS}_{q,m,k,\beta}^{\mathcal{K}}$ *problem (with* $\beta = 2\beta_v$*) with non-negligible probability* $\left(\delta - \frac{1}{|\mathcal{S}_{n,\kappa}|}\right)\cdot\left(\frac{\delta - \frac{1}{|\mathcal{S}_{n,\kappa}|}}{y} - \frac{1}{|\mathcal{S}_{n,\kappa}|}\right) - \sqrt{\frac{q^{4n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$.

Proof. The proof is given in Appendix F. □

Theorem 10 (Anonymity). *Suppose* $\sqrt{\frac{q^{4n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$ *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 MIMO.L2RS scheme is unconditionally secure for anonymity as defined in Def. 10.*

Proof. The proof is given in AppendixG. □

Theorem 11 (Linkability). *The MIMO.L2RS scheme with parameter* β_v *is linkable in the random oracle model if the* $\mathbf{MSIS}_{q,m,k,\beta}^{\mathcal{K}}$ *problem (with* $\beta = 2\beta_v$*) is hard.*

Proof. The proof is given in Appendix H. □

Theorem 12 (Non-Slanderability). *For any linkable ring signature, if it satisfies unforgeability and linkability, then it satisfies non-slanderability.*

Proof. The proof is given in Appendix I. □

Corollary 2 (Non-Slanderability). *The MIMO.L2RS scheme is non-slanderable under the assumptions of Theorem 9 and Theorem 11.*

F MIMO.L2RS - Security Analysis - One-Time Unforgeability

Proof. As stated in [13], this MIMO.L2RS scheme relies on the $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,\beta}$ 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 $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,\beta}$ problem. To prove this, we start replacing the MIMO.L2RS.SigGen algorithm with MIMO.L2RS.Hybrid-1 and MIMO.L2RS.Hybrid-2 algorithms that are used to simulate the creation of the signatures, until we obtain an algorithm that breaks the $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,\beta}$ problem. These Hybrid algorithms are illustrated in Algorithm 11 and Algorithm 12, respectively.

In MIMO.L2RS.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 $\mathbf{A}^{(k)}_{2q,\pi} \cdot \mathbf{u}^{(k)}$ and $\mathbf{H}^{(k)}_{2q} \cdot \mathbf{u}^{(k)}$ being already set. This equality can be described as:

$$H_1\Big(L', \big\{\mathbf{H}^{(k)}_{2q}\big\}_{k\in[N_{in}+1]}, \mu, \big\{\mathbf{A}^{(k)}_{2q,w} \cdot \mathbf{t}^{(k)}_i + \mathbf{q}\cdot\mathbf{c}_w\big\}_{k\in[N_{in}+1]}, \big\{\mathbf{h}^{(k)}_{2q} \cdot \mathbf{t}^{(k)}_w + \mathbf{q}\cdot\mathbf{c}_w\big\}_{k\in[N_{in}+1]}\Big) =$$
$$H_1\Big(L', \big\{\mathbf{H}^{(k)}_{2q}\big\}_{k\in[N_{in}+1]}, \mu, \big\{\mathbf{A}^{(k)}_{2q,\pi} \cdot \mathbf{u}^{(k)}\big\}_{k\in[N_{in}+1]}, \big\{\mathbf{H}^{(k)}_{2q} \cdot \mathbf{u}^{(k)}\big\}_{k\in[N_{in}+1]}\Big)$$

Every time the MIMO.L2RS.Hybrid-1 is called, the probability of generating $\mathbf{u}$, (such that $\mathbf{A}^{(k)}_{2q,\pi} \cdot \mathbf{u}^{(k)}$ and $\mathbf{H}^{(k)}_{2q} \cdot \mathbf{u}^{(k)}$ 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 $h \cdot 2^{-n+1}$, 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 MIMO.L2RS.Hybrid-1 is at most $o \cdot h \cdot 2^{-n+1}$, which is negligible (Based on [13], Lemma 3.4).

Algorithm 11 MIMO.L2RS.Hybrid-1

Input: $\{\mathbf{S}^{(k)}_{(in),\pi}\}_{k\in[N_{in}+1]}, \mu$, L' as in (4), and Pub-Params.
Output: $\sigma_{L'}(\mu) = \Big(\mathbf{c}_1, \{\mathbf{t}^{(k)}_1, \ldots, \mathbf{t}^{(k)}_w\}_{k\in[N_{in}+1]}, \{\mathbf{h}^{(k)}\}_{k\in[N_{in}]}\Big)$

procedure MIMO.L2RS.HYBRID-1($\mathbf{S}^{(k)}_{(in),\pi}, \mu, L'$, Pub-Params)
  for $(1 \le k \le N_{in}+1)$ **do**
    Set $\mathbf{H}^{(k)}_{2q} = \big(2\cdot\mathbf{H}, -2\cdot\mathbf{h}^{(k)} + \mathbf{q}\big) \in \mathcal{R}^{2\times m}_{2q}$, where $\mathbf{h}^{(k)} = \mathbf{H}\cdot\mathbf{S}^{(k)}_{(in),\pi} \in \mathcal{R}^2_q$.
    Call L2RS.Lift($\mathbf{A}, \mathbf{a}^{(k)}_{(in),\pi}$) to obtain $\mathbf{A}^{(k)}_{2q,\pi} = (2\cdot\mathbf{A}, -2\cdot\mathbf{a}^{(k)}_{(in),\pi} + \mathbf{q}) \in \mathcal{R}^{2\times m}_{2q}$.
    Let $\mathbf{u}^{(k)} = (u_1, \ldots, u_m)^T$, where $u_i \leftarrow D^n_\sigma$, for $1 \le i \le m$.
    Choose at random $\mathbf{c}_{\pi+1} \leftarrow \mathcal{S}_{n,\kappa}$
  for $(i = \pi+1, \pi+2, \ldots, w, 1, 2, \ldots, \pi-1)$ **do**
    for $(1 \le k \le N_{in}+1)$ **do**
      Call L2RS.Lift($\mathbf{A}, \mathbf{a}^{(k)}_{(in),i}$) to obtain $\mathbf{A}^{(k)}_{2q,i} = (2\cdot\mathbf{A}, -2\cdot\mathbf{a}^{(k)}_{(in),i} + \mathbf{q}) \in \mathcal{R}^{2\times m}_{2q}$.
      Let $\mathbf{t}^{(k)}_i = (t_{i,1}, \ldots, t_{i,m})^T$, where $t_{i,j} \leftarrow D^n_\sigma$, for $1 \le j \le m$.
    Compute $\mathbf{c}_{i+1} = H_1\Big(L', \{\mathbf{H}^{(k)}_{2q}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}^{(k)}_{2q,i}\cdot\mathbf{t}^{(k)}_i + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}, \{\mathbf{H}^{(k)}_{2q}\cdot\mathbf{t}^{(k)}_i + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}\Big)$.
  for $(1 \le k \le N_{in}+1)$ **do**
    Choose $b^{(k)} \leftarrow \{0,1\}$.
    Let $\mathbf{t}^{(k)}_\pi \leftarrow \mathbf{u}^{(k)} + \mathbf{S}^{(k)}_{2q,\pi}\cdot\mathbf{c}_\pi\cdot(-1)^{b^{(k)}}$, where $\mathbf{S}^{(k)}_{2q,\pi} = [(\mathbf{S}^{(k)}_\pi)^T, 1]^T$.
    Continue with prob. $\Big(M\exp\Big(-\frac{\|\mathbf{S}^{(k)}_{2q,\pi}\cdot\mathbf{c}_\pi\|^2}{2\sigma^2}\Big)\cosh\Big(\frac{\langle\mathbf{t}^{(k)}_\pi, \mathbf{S}^{(k)}_{2q,\pi}\cdot\mathbf{c}_\pi\rangle}{\sigma^2}\Big)\Big)^{-1}$ otherwise **Restart**.
  return $\sigma_{L'}(\mu) = \Big(\mathbf{c}_1, \{\mathbf{t}^{(k)}_1, \ldots, \mathbf{t}^{(k)}_w\}_{k\in[N_{in}+1]}, \{\mathbf{h}^{(k)}\}_{k\in[N_{in}]}\Big)$.

After analyzing how $\mathbf{c}_1$ can be forged, we evaluate the $\mathbf{t}^{(k)}_1, \ldots, \mathbf{t}^{(k)}_w$ of the MIMO.L2RS scheme. We claim that these are forgeable when an attacker finds a PPT algorithm $\mathcal{F}$ to solve the $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,\beta}$ problem. This attack can be simulated using the MIMO.L2RS.Hybrid-2 shown in Algorithm 12, where $\mathbf{t}_\pi$ is directly chosen from the distribution D^n_σ (Based on [13], Lemma 3.5).

The public-key $\mathbf{A}_{2q,\pi}^{(k)} \in \mathcal{R}_{2q}^{2\times m}$ is generated such $\mathbf{A}_{2q,\pi}^{(k)} \cdot \mathbf{S}_{(in),\pi}^{(k),T} = q \in \mathcal{R}_{2q}^{2}$, so finding a vector $\mathbf{v}$ such that $\mathbf{A}_{2q,\pi}^{(k)} \cdot \mathbf{v} = \mathbf{0} \bmod q$ with $\mathbf{0} = (0,0)^T$. 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 $(\mathbf{A}_{2q,\pi}^{(k)}, \phi, \psi, \mathbf{c}_1, \ldots, \mathbf{c}_w)$.
4. $\mathcal{F}$ is initialized and run by providing the $\mathbf{A}_{2q,\pi}^{(k)}$ and forger's random coins ϕ.
5. The SubRoutine signs the message μ using the signer's coins ψ in the MIMO.L2RS.Hybrid-2, this produces a signature $\sigma_L(\mu)$.
6. During the signing process, $\mathcal{F}$ calls the oracle H_1 and answers are placed in the list $(\mathbf{c}_1, \ldots, \mathbf{c}_w)$, the queries are kept 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 $(\mathbf{c}_1, \{\mathbf{t}_1^{(k)}, \ldots, \mathbf{t}_w^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{h}^{(k)}\}_{k\in[N_{in}]})$, with negligible probability δ. This output has to be successfully accepted by the MIMO.L2RS.SigVer algorithm.

If the random oracle was not called using some input $\{\mathbf{A}_{2q,i}^{(k)} \cdot \mathbf{t}_i^{(k)} + \mathbf{q} \cdot \mathbf{c}_i\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)} \cdot \mathbf{t}_i^{(k)} + \mathbf{q} \cdot \mathbf{c}_i\}_{k\in[N_{in}+1]}$, then $\mathcal{F}$ has $1/|\mathcal{S}_{n,\kappa}|$ chances of producing a $\mathbf{c}$ such that $\mathbf{c} = H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}_{2q}^{(k)} \cdot \mathbf{t}^{(k)} + \mathbf{q}\cdot\mathbf{c}\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)} \cdot \mathbf{t}^{(k)} + \mathbf{q}\cdot\mathbf{c}\}_{k\in[N_{in}+1]}\Big)$. This turns out that $\delta - 1/|\mathcal{S}_{n,\kappa}|$ be the probability that $\mathbf{c} = \mathbf{c}_j$ for some j.

Algorithm 12 MIMO.L2RS.Hybrid-2

Input: $\{\mathbf{S}_{(in),\pi}^{(k)}\}_{k\in[N_{in}+1]}, \mu, L'$ as in (4), and Pub-Params.
Output: $\sigma_{L'}(\mu) = \Big(\mathbf{c}_1, \{\mathbf{t}_1^{(k)}, \ldots, \mathbf{t}_w^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{h}^{(k)}\}_{k\in[N_{in}]}\Big)$

procedure MIMO.L2RS.HYBRID-2($\mathbf{S}_{(in),\pi}^{(k)}, \mu, L'$, Pub-Params)
  for $(1 \le k \le N_{in}+1)$ **do**
    Set $\mathbf{H}_{2q}^{(k)} = \big(2\cdot\mathbf{H}, -2\cdot\mathbf{h}^{(k)} + \mathbf{q}\big) \in \mathcal{R}_{2q}^{2\times m}$, where $\mathbf{h}^{(k)} = \mathbf{H}\cdot\mathbf{S}_{(in),\pi}^{(k)} \in \mathcal{R}_q^2$.
    Call L2RS.Lift$(\mathbf{A}, \mathbf{a}_{(in),\pi}^{(k)})$ to obtain $\mathbf{A}_{2q,\pi}^{(k)} = (2\cdot\mathbf{A}, -2\cdot\mathbf{a}_{(in),\pi}^{(k)} + \mathbf{q}) \in \mathcal{R}_{2q}^{2\times m}$.
    Let $\mathbf{u}^{(k)} = (u_1, \ldots, u_m)^T$, where $u_i \leftarrow D_\sigma^n$, for $1 \le i \le m$.
  Choose at random $\mathbf{c}_{\pi+1} \leftarrow \mathcal{S}_{n,\kappa}$.
  for $(i = \pi+1, \pi+2, \ldots, w, 1, 2, \ldots, \pi-1)$ **do**
    for $(1 \le k \le N_{in}+1)$ **do**
      Call L2RS.Lift$(\mathbf{A}, \mathbf{a}_{(in),i}^{(k)})$ to obtain $\mathbf{A}_{2q,i}^{(k)} = (2\cdot\mathbf{A}, -2\cdot\mathbf{a}_{(in),i}^{(k)} + \mathbf{q}) \in \mathcal{R}_{2q}^{2\times m}$.
      Let $\mathbf{t}_i^{(k)} = (t_{i,1}, \ldots, t_{i,m})^T$, where $t_{i,j} \leftarrow D_\sigma^n$, for $1 \le j \le m$.
    Compute $\mathbf{c}_{i+1} = H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}_{2q,i}^{(k)} \cdot \mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)} \cdot \mathbf{t}_i^{(k)} + \mathbf{q}\cdot\mathbf{c}_i\}_{k\in[N_{in}+1]}\Big)$.
  for $(1 \le k \le N_{in}+1)$ **do**
    Choose $b^{(k)} \leftarrow \{0,1\}$.
    Choose $\mathbf{t}_\pi^{(k)} \leftarrow D_\sigma^{n\times m}$
    Continue with probability $\frac{1}{M}$ otherwise **Restart**.
  return $\sigma_{L'}(\mu) = \Big(\mathbf{c}_1, \{\mathbf{t}_1^{(k)}, \ldots, \mathbf{t}_w^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{h}^{(k)}\}_{k\in[N_{in}]}\Big)$.

FORGERY 1. Let's consider the situation that $\mathbf{c}_{j+1}$ is the result after using $\mathcal{F}$ which is $\mathbf{c}_{j+1} = H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu', \{\mathbf{A}_{2q}^{(k)} \cdot \mathbf{t}'^{(k)} + \mathbf{q}\cdot\mathbf{c}_j\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)} \cdot \mathbf{t}'^{(k)} + \mathbf{q}\cdot\mathbf{c}_j\}_{k\in[N_{in}+1]}\Big)$. Then by comparing this with a legitimate signature, we have:

$$H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}_{2q}^{(k)} \cdot \mathbf{t}^{(k)} + \mathbf{q}\cdot\mathbf{c}_j\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)} \cdot \mathbf{t}^{(k)} + \mathbf{q}\cdot\mathbf{c}_j\}_{k\in[N_{in}+1]}\Big) = H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu', \{\mathbf{A}_{2q}^{(k)} \cdot \mathbf{t}'^{(k)} + \mathbf{q}\cdot\mathbf{c}_j\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)} \cdot \mathbf{t}'^{(k)} + \mathbf{q}\cdot\mathbf{c}_j\}_{k\in[N_{in}+1]}\Big)$$

$\mathcal{F}$ will find a preimage of $\mathbf{c}_j$ if $\mu \neq \mu'$ or $\mathbf{A}_{2q}^{(k)} \cdot \mathbf{t}^{(k)} + \mathbf{q}\cdot\mathbf{c}_j \neq \mathbf{A}_{2q}^{(k)} \cdot \mathbf{t}'^{(k)} + \mathbf{q}\cdot\mathbf{c}_j$ or $\mathbf{H}_{2q}^{(k)} \cdot \mathbf{t}^{(k)} + \mathbf{q}\cdot\mathbf{c}_j \neq \mathbf{H}_{2q}^{(k)} \cdot \mathbf{t}'^{(k)} + \mathbf{q}\cdot\mathbf{c}_j$. Then, we have with overwhelming probability that $\mu = \mu'$ and

$\mathbf{A}_{2q}^{(k)} \cdot \mathbf{t}^{(k)} + \mathbf{q} \cdot \mathbf{c}_j = \mathbf{A}_{2q}^{(k)} \cdot \mathbf{t}'^{(k)} + \mathbf{q} \cdot \mathbf{c}_j$ and $\mathbf{H}_{2q}^{(k)} \cdot \mathbf{t}^{(k)} + \mathbf{q} \cdot \mathbf{c}_j = \mathbf{H}_{2q}^{(k)} \cdot \mathbf{t}'^{(k)} + \mathbf{q} \cdot \mathbf{c}_j$. These equalities will result in: $\mathbf{A}_{2q}^{(k)}(\mathbf{t}^{(k)} - \mathbf{t}'^{(k)}) = 0 \bmod q$ and $\mathbf{H}_{2q}^{(k)}(\mathbf{t}^{(k)} - \mathbf{t}'^{(k)}) = 0 \bmod q$. We assume that both $\mathbf{t}$ and $\mathbf{t}'$ are different and they met the MIMO.L2RS.SigVer conditions, so it yields $\mathbf{t} - \mathbf{t}' \neq 0 \bmod q$, and $\|\mathbf{t} - \mathbf{t}'\| \leq 2\beta_v$.

Forgery 2. In this scenario, we assume that the MIMO.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 [3] to show the probability of $\mathbf{c}_j = \mathbf{c}'_j$ and the forger uses an oracle response $\mathbf{c}'_j$ is at least:

$$\left(\delta - \frac{1}{|\mathcal{S}_{n,\kappa}|}\right) \cdot \left(\frac{\delta - \frac{1}{|\mathcal{S}_{n,\kappa}|}}{y} - \frac{1}{|\mathcal{S}_{n,\kappa}|}\right) \tag{27}$$

Therefore, with the probability (27), $\mathcal{F}$ creates a signature $\sigma_L(\mu) = \left(\mathbf{c}'_1, \left\{\mathbf{t}_1^{'(k)}, \ldots, \mathbf{t}_w^{'(k)}\right\}_{k\in[N_{in}+1]}, \left\{\mathbf{h}^{(k)}\right\}_{k\in[N_{in}]}\right)$ where $\mathbf{A}_{2q}^{(k)} \cdot \mathbf{t}^{(k)} + \mathbf{q} \cdot \mathbf{c}_j = \mathbf{A}_{2q}^{(k)} \cdot \mathbf{t}'^{(k)} + \mathbf{q} \cdot \mathbf{c}_j$ and $\mathbf{H}_{2q}^{(k)} \cdot \mathbf{t}^{(k)} + \mathbf{q} \cdot \mathbf{c}_j = \mathbf{H}_{2q}^{(k)} \cdot \mathbf{t}'^{(k)} + \mathbf{q} \cdot \mathbf{c}_j$. We now obtained: $\mathbf{A}_{2q}^{(k)} \cdot (\mathbf{t}^{(k)} - \mathbf{t}^{'(k)}) = \mathbf{q}(\mathbf{c}_j - \mathbf{c}'_j) \bmod 2q$ and $\mathbf{H}_{2q}^{(k)} \cdot (\mathbf{t}^{(k)} - \mathbf{t}^{'(k)}) = \mathbf{q}(\mathbf{c}_j - \mathbf{c}'_j) \bmod 2q$. Since $\mathbf{c}_j - \mathbf{c}'_j \neq 0 \bmod 2$, so in both equations, we have $\mathbf{t}^{(k)} - \mathbf{t}^{'(k)} \neq 0 \bmod 2q$ where $\|\mathbf{t}^{(k)} - \mathbf{t}^{'(k)}\|_\infty < q/2$. By applying mod$q$ reduction, we find a small non-zero vector $\mathbf{v}^{(k)} = \mathbf{t}^{(k)} - \mathbf{t}^{'(k)} \neq 0 \bmod q$. This $\mathbf{v}^{(k)}$ will compute $\mathbf{A}_{2q}^{(k)} \cdot \mathbf{v}^{(k)} = \mathbf{0} \bmod q$ and $\mathbf{H}_{2q}^{(k)} \cdot \mathbf{v}^{(k)} = \mathbf{0} \bmod q$ with $\|\mathbf{v}^{(k)}\| \leq 2\beta_v$. Since $\mathbf{v}^{(k)}$ is same for both $\mathbf{A}_{2q}^{(k)}$ and $\mathbf{H}_{2q}^{(k)}$, we only use the former to continue this analysis. We say that $\mathbf{A}_{2q}^{(k)} \bmod q = 2(\mathbf{A}, -\mathbf{a}^{(k)}) \bmod q$, then $2(\mathbf{A}, -\mathbf{a}^{(k)})\mathbf{v}^{(k)} = \mathbf{0} \bmod q$, this implies that $(\mathbf{A}, -\mathbf{a}^{(k)})\mathbf{v}^{(k)} = \mathbf{0} \bmod q$, since q is odd. The probability of success of an attacker in MIMO.L2RS.Hydrid-3 differs by a negligible amount from the success probability in MIMO.L2RS.KeyGen and is thus non-negligible. Therefore, this vector $\mathbf{v}$ will be a solution to the $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,\beta}$ problem, where $\beta = 2\beta_v$, with non-negligible probability and with respect to $(\mathbf{A}, -\mathbf{a}^{(k)})$ over $\mathcal{R}_q^2$. Furthermore, notice that MIMO.L2RS.Hybrid-2 shown in Algorithm 12 no longer uses the private-key $\mathbf{S}_\pi^{(k)}$, except for generating $\mathbf{A}_{2q,\pi}^{(k)}$ and $\mathbf{H}_{2q}^{(k)}$ to obtain the final $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,\beta}$ solution. For $\mathbf{A}_{2q,\pi}^{(k)}$, we modified the MIMO.L2RS.KeyGen algorithm with the MIMO.L2RS.Hydrid-3 game shown in Algorithm 13, where the public-key $\mathbf{a}^{(k)}$ is uniformly and randomly taken as $\mathbf{a}^{(k)} \leftarrow \mathcal{R}_q^2$. On the other hand, for $\mathbf{H}_{2q}^{(k)}$, we chose the linking taq uniformly and randomly as $\mathbf{h}^{(k)} \leftarrow \mathcal{R}_q^2$. By the argument of the Leftover Hash Lemma (LHL) - Lemma 5 and our assumption that $\sqrt{\frac{q^{4n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$ is negligible in n.

Algorithm 13 MIMO.L2RS.Hybrid-3 $(\mathbf{a}, \mathbf{S})$

Input: Pub-Param: $\mathbf{A}$.
Output: $(\mathbf{a}, \mathbf{S})$, being the public-key and the private-key, respectively.
1: **procedure** MIMO.L2RS.HYBRID-3($\mathbf{A}$)
2: Let $\mathbf{S}^T = (\mathbf{s}_1, \ldots, \mathbf{s}_{m-1}) \in \mathcal{R}_q^{1\times(m-1)}$, where $\mathbf{s}_i \leftarrow (-2^\gamma, 2^\gamma)^n$, for $1 \leq i \leq m-1$
3: Choose $\mathbf{a} \leftarrow \mathcal{R}_q^2$
4: **return** $(\mathbf{a}, \mathbf{S})$.

□

G MIMO.L2RS - Security Analysis - Anonymity

Proof. We prove the anonymity of this scheme using the sequence-of-games approach [32] 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 of pk's $L = \{\mathbf{a}_0^{(k)}, \mathbf{a}_1^{(k)}\}_{k\in[N_{in}+1]}$, the signature $\sigma_L(\mu)$, message μ, and the random oracle model (H_1). The key generation algorithm creates the pair of users' keys in this ring signature: Private-Keys $\leftarrow \{\mathbf{S}_0^{(k)}, \mathbf{S}_1^{(k)}\}_{k\in[N_{in}+1]}$ and the Public-Keys $\leftarrow (\mathbf{a}_0^{(k)}, \mathbf{a}_1^{(k)})$; a user b is chosen uniformly at random from the list $L = \{\mathbf{a}_0^{(k)}, \mathbf{a}_1^{(k)}\}$, then the signature $\sigma_L(\mu) = \mathsf{MIMO.L2RS.SigGen}(\mathbf{S}_b^{(k)}, \mu, L, \mathsf{Pub\text{-}Param})$ is generated. So in **Game 0**, a PPT adversary $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$; thus in the event **Game 0**, $\mathcal{A}$ succeeds in breaking ambiguity **Game 0**$(b = b')$ if $\Pr[\textbf{Game 0}] \leq \frac{1}{2} + non - negl(\lambda)$.

Game 1: Changes in this game are made to the user π in the second part of the linkability tag $\mathbf{h}^{(k)} = (\mathbf{H} \cdot \mathbf{S}^{(k)}) \in \mathcal{R}_q^2$, in signature of user π, and public-key $\mathbf{a}^{(k)} = (\mathbf{A} \cdot \mathbf{S}^{(k)}) \in \mathcal{R}_q^2$ in the $\mathsf{MIMO.L2RS.KeyGen}$ algorithm. The $\mathbf{h}^{(k)}$ and $\mathbf{a}^{(k)}$ are now randomly chosen from $\mathcal{R}_q^2$. We claim that $|\Pr[\textbf{Game 0}] - \Pr[\textbf{Game 1}]| \leq \epsilon_{LHL_{G1}}$.

Where $\epsilon_{LHL_{G1}}$ is the advantage of some efficient algorithm which is negligible. In both cases $\mathbf{h}^{(k)} = (\mathbf{H} \cdot \mathbf{S}^{(k)}) \in \mathcal{R}_q^2$ and $\mathbf{a}^{(k)} = (\mathbf{A} \cdot \mathbf{S}^{(k)}) \in \mathcal{R}_q^2$, we know that $\mathbf{H}$ and $\mathbf{A}$ are uniform and $\mathbf{S}^{(k)}$ is chosen small and with coefficients in $(-2^\gamma, 2^\gamma)$. When $\mathbf{S}^{(k)}$ is multiplied by $\mathbf{H}$ and $\mathbf{A}$ respectively, it gives $\mathbf{h}^{(k)}$ and $\mathbf{a}^{(k)}$ that are close to uniform over $\mathcal{R}_q^2$. By applying the Leftover Hash Lemma (LHL) - **Lemma 5**, the statistical distance between the distribution of ($\mathbf{h}^{(k)}$ mod q and $\mathbf{a}^{(k)}$ mod q) and the uniform distribution on $\mathcal{R}_q^2 \times \mathcal{R}_q^2$ is at most $n \cdot \frac{1}{2} \cdot \sqrt{\frac{q^{4n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$. We conclude that in **Game 1**:

$$|\Pr[\textbf{Game 0}] - \Pr[\textbf{Game 1}]| \leq n \cdot \tfrac{1}{2} \cdot \sqrt{\tfrac{q^{4n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}. \tag{28}$$

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

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 $\mathbf{a}_i^{(k)} = (\mathbf{A} \cdot \mathbf{S}_i^{(k)} \bmod q)$ are uniform and all $\mathbf{S}_i^{(k)}$'s are chosen small with coefficients in $(-2^\gamma, 2^\gamma)$. When the $\mathbf{S}_i^{(k)}$'s are multiplied by $\mathbf{A}_i$'s, it gives ($\mathbf{a}_i^{(k)} \bmod q$)'s that are close to uniform over $\mathcal{R}_q^2$. By applying the Leftover Hash Lemma (LHL) - **Lemma 5**, the statistical distance between the distribution of the $(\mathbf{A} \cdot \mathbf{S}_i^{(k)} \bmod q)$'s and the uniform distribution on $\mathcal{R}_q^2 \times \mathcal{R}_q^2$ is at most $n \cdot \frac{1}{2} \cdot \sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}} \cdot (w-1)$. So in **Game 2**, we conclude that:

$$|\Pr[\textbf{Game 1}] - \Pr[\textbf{Game 2}]| \leq n \cdot \tfrac{1}{2} \cdot \sqrt{\tfrac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}} \cdot (w-1). \tag{29}$$

Game 3: At this time, we make a change in $\mathbf{c}_{\pi+1}$. Instead of programming the oracle as $H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}_{2q,\pi}^{(k)} \cdot \mathbf{u}^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)} \cdot \mathbf{u}^{(k)}\}_{k\in[N_{in}+1]}\Big)$, 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 $\sigma_{L'}(\mu) = \Big(\mathbf{c}_1, \{\mathbf{t}_1^{(k)}, \ldots, \mathbf{t}_w^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{h}^{(k)}\}_{k\in[N_{in}]}\Big)$ and programs the oracle as $H_1\Big(L', \{\mathbf{H}_{2q}^{(k)}\}_{k\in[N_{in}+1]}, \mu, \{\mathbf{A}_{2q,\pi}^{(k)} \cdot \mathbf{u}^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{H}_{2q}^{(k)} \cdot \mathbf{u}^{(k)}\}_{k\in[N_{in}+1]}\Big) = \mathbf{c}_{\pi+1}$. 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**:

$$|\Pr[\textbf{Game 2}] - \Pr[\textbf{Game 3}]| \leq h \cdot 2^{-n+1}. \tag{30}$$

Game 4: In this game a change is made in $\mathbf{t}_\pi^{(k)}$. Namely, instead of computing it as $\mathbf{u}^{(k)} + \mathbf{S}_{2q,\pi}^{(k)} \cdot \mathbf{c}_\pi \cdot (-1)^{bit}$, it is now directly chosen from the Gaussian distribution D_σ^n. 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, $\mathbf{t}_{\pi}^{(k)}$ is computed using rejection sampling - **Lemma 1**, thus it is always sample from the Gaussian distribution D_{σ}^{n}. In this Game, however, $\mathbf{t}_{\pi}^{(k)}$ is directly chosen from D_{σ}^{n}, this means that the advantage $\epsilon_{RS_{G4}}$ will be zero as in both **Game 3** and **Game 4**, $\mathbf{t}_{\pi}^{(k)}$ is having same distribution. In **Game 4**, we have:

$$|\Pr[\textbf{Game 3}] - \Pr[\textbf{Game 4}]| = 0. \tag{31}$$

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:

$$|\Pr[\textbf{Game 4}] - \Pr[\textbf{Game 5}]| \leq h \cdot 2^{-n+1} \cdot w. \tag{32}$$

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 (28), (29), (30), (31) and (32) we obtain:

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

By replacing the resulting probabilities, we have:

$$|\Pr[\textbf{Game 5}] - \Pr[\textbf{Game 0}]| \leq \frac{1}{w} - \frac{1}{2} + \epsilon, \tag{33}$$

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

$$\frac{n \cdot (w-1)}{2} \cdot \left(\sqrt{\frac{q^{4n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}} + \sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}} \right) + h \cdot 2^{-n+1} \cdot (1+w).$$

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 $\left(\sqrt{\frac{q^{4n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}} + \sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}} \right) \leq 2 \cdot \sqrt{\frac{q^{4n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$, which is negligible by the assumption that $\sqrt{\frac{q^{2n}}{2^{(\gamma+1)\cdot(m-1)\cdot n}}}$ is also negligible. Hence we conclude that ϵ is negligible, meaning that $\Pr[\textbf{Game 0}] \leq \frac{1}{2} + \epsilon$. □

H MIMO.L2RS - Security Analysis - Linkability

Proof. We construct the algorithm $\mathcal{B}$ for the $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,\beta}$ problem. This algorithm runs the linkability attack game (Def. 11) as follows:

1. $\mathcal{B}$ generates using the MIMO.L2RS.KeyGen algorithm all private-keys $\mathbf{S}_i^{(k)}$'s with the corresponding public-keys $\mathbf{a}_i^{(k)}$'s, then $\mathcal{B}$ gives $\mathbf{S}_{\pi}^{(k)}$ 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 lists L and L' such that both signatures are successfully verified by MIMO.L2RS.SigVer, but the linkability tags are different $\mathbf{h}_{\mu_1}^{(k)} \neq \mathbf{h}_{\mu'}^{(k)}$ with $k \in [N_{in}]$.

3. $\mathcal{B}$ computes $\mathbf{h}_{\mu_\pi}^{(k)} = \mathbf{H} \cdot \mathbf{S}_\pi^{(k)} \bmod q$, where π is the true signer's π linkability tag. This $\mathbf{h}_{\mu_\pi}^{(k)}$ tag can then be compared with the linkability tags $\mathbf{h}_{\mu_1}^{(k)}$ and $\mathbf{h}_{\mu'}^{(k)}$, output by $\mathcal{A}$, in step 2, and one of them will be different.
4. Without loss of generality, suppose $\mathbf{h}_{\mu_1}^{(k)} \neq \mathbf{h}_{\mu_\pi}^{(k)} \bmod q$. Using the forking lemma [3], $\mathcal{B}$ rewinds the attacker $\mathcal{A}$ to the H_1 query corresponding to the MIMO.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: $\sigma_L(\mu_2) = \left(\mathbf{c}_{1,\mu_2}, \{\mathbf{t}_{1,\mu_2}^{(k)}, \ldots, \mathbf{t}_{w,\mu_2}^{(k)}\}_{k\in[N_{in}+1]}, \{\mathbf{h}_{\mu_2}^{(k)}\}_{k\in[N_{in}]}\right)$. This second signature is used to extract a solution to the $\mathbf{MSIS}_{q,m,k,\beta}^{\mathcal{K}}$ 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 $\mathbf{H}_{2q,\mu_1}^{(k)}$ and $\mathbf{A}_{2q,w,\mu_1}^{(k)}$ are kept. Thus we have:
   (a) $\mathbf{A}_{2q,w,\mu_1}^{(k)} \cdot \mathbf{t}_{w,\mu_1}^{(k)} + \mathbf{q} \cdot \mathbf{c}_{w,\mu_1} = \mathbf{A}_{2q,w,\mu_1}^{(k)} \cdot \mathbf{t}_{w,\mu_2}^{(k)} + \mathbf{q} \cdot \mathbf{c}_{w,\mu_2}$,
   (b) $\mathbf{H}_{2q,\mu_1}^{(k)} \cdot \mathbf{t}_{w,\mu_1}^{(k)} + \mathbf{q} \cdot \mathbf{c}_{w,\mu_1} = \mathbf{H}_{2q,\mu_1}^{(k)} \cdot \mathbf{t}_{w,\mu_2}^{(k)} + \mathbf{q} \cdot \mathbf{c}_{w,\mu_2}$.
   These expressions can be represented as:
   (a) $\mathbf{A}_{2q,w,\mu_1}^{(k)} \cdot (\mathbf{t}_{w,\mu_1}^{(k)} - \mathbf{t}_{w,\mu_2}^{(k)}) = \mathbf{q} \cdot (\mathbf{c}_{w,\mu_2} - \mathbf{c}_{w,\mu_1})$,
   (b) $\mathbf{H}_{2q,\mu_1}^{(k)} \cdot (\mathbf{t}_{w,\mu_1}^{(k)} - \mathbf{t}_{w,\mu_2}^{(k)}) = \mathbf{q} \cdot (\mathbf{c}_{w,\mu_2} - \mathbf{c}_{w,\mu_1})$.
   Reducing them mod$q$ we have (if $(\mathbf{c}_{w,\mu_2} - \mathbf{c}_{w,\mu_1}) \neq \mathbf{0} \bmod 2$):
   (a) $\mathbf{A}_{2q,w,\mu_1}^{(k)} \cdot (\mathbf{t}_{w,\mu_1}^{(k)} - \mathbf{t}_{w,\mu_2}^{(k)}) = \mathbf{0} \bmod q$,
   (b) $\mathbf{H}_{2q,\mu_1}^{(k)} \cdot (\mathbf{t}_{w,\mu_1}^{(k)} - \mathbf{t}_{w,\mu_2}^{(k)}) = \mathbf{0} \bmod q$.
   We denote by $\mathbf{t}_{w,\mu_1}^{\prime(k)}$, the first $(m-1)$ ring elements in $\mathbf{t}_{w,\mu_1}^{(k)}$ and by $\mathbf{t}_{w,\mu_1}^{\prime\prime(k)}$the m-th ring element in $\mathbf{t}_{w,\mu_1}^{(k)}$, i.e. $\mathbf{t}_{w,\mu_1}^{(k)} - \mathbf{t}_{w,\mu_2}^{(k)} = \begin{pmatrix} \mathbf{t}_{w,\mu_1}^{\prime(k)} - \mathbf{t}_{w,\mu_2}^{\prime(k)} \\ \mathbf{t}_{w,\mu_1}^{\prime\prime(k)} - \mathbf{t}_{w,\mu_2}^{\prime\prime(k)} \end{pmatrix} \in \mathcal{R}_q^m$, and using the public-key and linkability parts, we have:
   (a) $2 \cdot \mathbf{A} \cdot (\mathbf{t}_{w,\mu_1}^{\prime(k)} - \mathbf{t}_{w,\mu_2}^{\prime(k)}) = -2 \cdot \mathbf{a}^{(k)} \cdot (\mathbf{t}_{w,\mu_1}^{\prime\prime(k)} - \mathbf{t}_{w,\mu_2}^{\prime\prime(k)})$,
   (b) $2 \cdot \mathbf{H} \cdot (\mathbf{t}_{w,\mu_1}^{\prime(k)} - \mathbf{t}_{w,\mu_2}^{\prime(k)}) = -2 \cdot \mathbf{h}_{\mu_1}^{(k)} \cdot (\mathbf{t}_{w,\mu_1}^{\prime\prime(k)} - \mathbf{t}_{w,\mu_2}^{\prime\prime(k)})$, where $\mathbf{h}_{\mu_1}^{(k)} \triangleq \mathbf{H} \cdot \mathbf{S}_\pi^{(k)} \in \mathcal{R}_q^2$.
6. We let $\bar{\mathbf{S}}^{(k)} = \frac{(\mathbf{t}_{w,\mu_1}^{\prime(k)} - \mathbf{t}_{w,\mu_2}^{\prime(k)})}{(\mathbf{t}_{w,\mu_1}^{\prime\prime(k)} - \mathbf{t}_{w,\mu_2}^{\prime\prime(k)})} \bmod q$ where $(\mathbf{t}_{w,\mu_1}^{\prime\prime(k)} - \mathbf{t}_{w,\mu_2}^{\prime\prime(k)}) \neq \mathbf{0} \bmod q$. We distinguish two cases:
   (a) If $\bar{\mathbf{S}}^{(k)} \neq \mathbf{S}_\pi^{(k)} \bmod q$, since we have $\mathbf{A} \cdot \bar{\mathbf{S}}^{(k)} = \mathbf{A} \cdot \mathbf{S}_\pi^{(k)} = \mathbf{a}^{(k)} \bmod q$, then $(\bar{\mathbf{S}}^{(k)} - \mathbf{S}^{(k)})$ is a small non-zero vector $\mathbf{MSIS}_{q,m,k,\beta}^{\mathcal{K}}$ solution for $\mathbf{A} \in \mathcal{R}_q^{2\times(m-1)}$.
   (b) If $\bar{\mathbf{S}}^{(k)} = \mathbf{S}_\pi^{(k)} \bmod q$, then $\mathbf{h}_{\mu_1}^{(k)} = \mathbf{H} \cdot \bar{\mathbf{S}}^{(k)} \bmod q = \mathbf{H} \cdot \mathbf{S}_\pi^{(k)} \bmod q$. The target is to show that $\mathbf{h}_{\mu_1}^{(k)} = \mathbf{h}_{\mu_\pi}^{(k)} \bmod 2$ and $\mathbf{h}_{\mu_1}^{(k)} = \mathbf{h}_{\mu_\pi}^{(k)} \bmod q$. If so, then we have $\mathbf{h}_{\mu_1}^{(k)} = \mathbf{h}_{\mu_\pi}^{(k)} \bmod 2q$, which is a contradiction with our assumption at step 4 of this proof. We now prove the first target:

$$\mathbf{h}_{\mu_1}^{(k)} = -2 \cdot \mathbf{h}_{\mu_1}^{\prime(k)} + \mathbf{q} = \mathbf{1} \bmod 2 = -2 \cdot \mathbf{H} \cdot \mathbf{S}_\pi^{(k)} + \mathbf{q} = \mathbf{h}_{\mu_\pi}^{(k)},$$

   where the first and the last equalities follow from definition of $\mathbf{h}^{(k)}$ in second line of Algorithm 2. To show the second target, we have

$$\mathbf{h}_{\mu_1}^{(k)} = -2 \cdot \mathbf{h}_{\mu_1}^{(k)} + \mathbf{q} = -2 \cdot \mathbf{h}_{\mu_1}^{(k)} \bmod q$$

$$= -2 \cdot \mathbf{H} \cdot \bar{\mathbf{S}}^{(k)} \bmod q = -2 \cdot \mathbf{H} \cdot \mathbf{S}_\pi^{(k)} \bmod q = \mathbf{h}_{\mu_\pi}^{(k)},$$

   where the first and the last equalities follow from definition of $\mathbf{h}^{(k)}$ 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 $(\mathbf{c}_{w,\mu_2} - \mathbf{c}_{w,\mu_1}) \neq \mathbf{0} \bmod 2$, we have $(\mathbf{t}_{w,\mu_1}^{(k)} - \mathbf{t}_{w,\mu_2}^{(k)}) \neq \mathbf{0} \bmod 2q$. In addition, we know that $\|\mathbf{t}_{w,\mu_1}^{(k)} - \mathbf{t}_{w,\mu_2}^{(k)}\|_\infty < q/2$, which implies that $(\mathbf{t}_{w,\mu_1}^{(k)} - \mathbf{t}_{w,\mu_2}^{(k)}) \neq \mathbf{0} \bmod q$. Ultimately, we have $\mathbf{A} \cdot (\mathbf{t}_{w,\mu_1}^{(k)} - \mathbf{t}_{w,\mu_2}^{(k)}) = \mathbf{0} \bmod q$ and $\|(\mathbf{t}_{w,\mu_1}^{(k)} - \mathbf{t}_{w,\mu_2}^{(k)}) \bmod q\| \leq 2\beta_v$. Therefore, this small non-zero vector $(\mathbf{t}_{w,\mu_1}^{(k)} - \mathbf{t}_{w,\mu_2}^{(k)})$ is the output of the algorithm $\mathcal{B}$, and this vector is a solution to the $\mathbf{MSIS}_{q,m,k,\beta}^{\mathcal{K}}$ problem with $\beta = 2\beta_v$ for $\mathbf{a}^{(k)} \in \mathcal{R}_q^2$.

□

I MIMO.L2RS - Security Analysis - Non-Slanderability

Proof. 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 $i \in \{1, \dots w\}$, 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 $i \in \{1, \dots w\}$ 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 $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,\beta}$ 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 (9 and 11), and with these probabilities the $\mathcal{A}$ will find a $\mathbf{MSIS}^{\mathcal{K}}_{q,m,k,\beta}$ solution. Therefore, non-slanderability is implied by the definitions of the unforgeability (Def. 9) and linkability (Def. 11), and security analysis, (Appendix F) and (Appendix H), respectively. □