Efficient and Post-Quantum Zero-Knowledge Proofs for Blockchain Confidential Transaction Protocols*

Shang GAO[1], Tianyu ZHENG[1], Yu GUO[2], and Bin XIAO[1]

[1] The Hong Kong Polytechnic University, Hong Kong, China
[2] SECBIT Labs, Suzhou, China
{shanggao,tianzheng,csbxiao}@polyu.edu.hk
yu.guo@secbit.io

Abstract. We propose new zero-knowledge proofs for efficient and post-quantum ring confidential transaction (RingCT) protocols based on lattice assumptions in Blockchain systems. First, we introduce an inner-product based linear equation satisfiability approach for balance proofs with a wide range (e.g. 64-bit precision). Unlike existing balance proofs that require additional proofs for some "corrector values" [CCS'19], our approach avoids the corrector values for better efficiency. Furthermore, we design a ring signature scheme to efficiently hide a user's identity in large anonymity sets. Different from existing approaches that adopt a one-out-of-many proof [CCS'19, Crypto'19], we show that a linear sum proof suffices in ring signatures which could avoid the costly binary proof part. We further use the idea of "unbalanced" relations to build a logarithmic-size ring signature scheme. Finally, we show how to adopt these techniques in RingCT protocols and implement a prototype to compare the performance with existing approaches. The results show our solutions can reduce about 25% proof size of Crypto'19, and up to 70% proof size, 30% proving time, and 20% verification time of CCS'19. We also believe our techniques are of independent interest for other privacy-preserving applications such as secure e-voting and are applicable in a generic setting.

Keywords: Lattice-based cryptography, zero-knowledge proof, balance proof, ring signature, RingCT, blockchain

1 Introduction

Cryptocurrencies adopt the blockchain technique where each participant maintains a ledger of all transactions to avoid any tampering attempts from minority attackers. In private/anonymous cryptocurrencies, the amount[3] stored in each

* This is not the final version. The experiment part needs to be changed due to some major changes.

[3] In this paper, the "amount" refers to "account balance". We avoid using balance here as it conflicts with balance proofs.

account and the user's identity need to be hidden from the outside world. Meanwhile, it also requires public verification to ensure each transaction is valid. Existing solutions such as Monero [32] and Zcash [34] adopt zero-knowledge proofs (ZKPs) to prove useful statements without leaking any private information. For instance, in Monero, a ring confidential transaction (RingCT) protocol is used with a *range proof* to show all amounts are non-negative and the difference between outputs and inputs is zero (balance property), and a *ring signature*-like approach to hide the identity of a spender with one-out-of-many proofs [27]. However, as the security of these implementations is mainly based on discrete logarithm assumptions, they are at risk of potential attacks from quantum computers.

This deficiency has impelled the development of "post-quantum" solutions. Among all approaches, lattice-based cryptography is one of the most promising candidates based on computational lattice problems. Unfortunately, the costs of lattice-based solutions increase significantly in comparison with those in discrete logarithm settings. Taking the range proof in Crypto'19 [14] as an example, a single proof costs nearly 200KB size while the Bulletproofs protocol [11] costs less than 1KB. Even worse, as the amounts in a RingCT protocol need to be committed separately, the efficient aggregation approach in [14] cannot be adopted. MatRiCT (CCS'19 [17]) is the first practical lattice-based RingCT protocol to optimize the proof size in a blockchain environment and is currently applied in Hcash [31]. By using a novel balance proof with hashed-message commitments (HMC) to show a transaction is valid, MatRiCT reduces the size of commitments and allows proofs on a wide range. Furthermore, it adopts techniques such as batched commitments and rejection sampling for secrets with a fixed Hamming weight in one-out-of-many proofs to improve the efficiency of the ring signature. MatRiCT+ [16] further improves the performance of MatRiCT by optimizing the underlying cyclotomic rings. However, both MatRiCT and MatRiCT+ require some "corrector values" in balance proofs. Proving corrector values are correct imposes a prohibitive cost.

1.1 Our Contribution

The main goal of this paper is to propose efficient, scalable, and practical ZKPs for existing post-quantum[4] anonymous cryptocurrencies such as Hcash [31]. We focus on some key problems in lattice-based RingCT protocols (e.g., MatRiCT) and significantly reduce the proof size and proving/verification time with our new ZKP techniques. Besides, as our approaches optimize the high-level ZKP relations, which are independent from the cyclotomic ring improvements in [16], our techniques can also be applied in MatRiCT+ to achieve more efficient RingCT protocols (here we focus on MatRiCT as our improvements are based on the techniques proposed in MatRiCT). To achieve the high efficiency of our approach, we

[4] The post-quantum security referred in this paper relies on the hardness of "post-quantum" lattice assumptions and does not necessarily involve security proofs in quantum random oracle model [21]. Instead, we use ROM for Fiat-Shamir transformation in the security analysis, as with [14–17].

propose two novel techniques, linear equation satisfiability (Section 4.1 and 5.1) and unbalanced linear sum proof (Section 4.2 and 5.2). The former technique implies balance relations in RingCT protocols. The latter one proves a weaker but still secure relation to replace the one-out-of-many relation in ring signatures. Since both of them do not rely on lattice settings, we believe our results are of independent interest for ZKPs in a generic setting and other applications (Section 9).

We conclude our revelations as follows:

- *"Corrector values" are unnecessary in a balance proof.* We analyze the balance proof in MatRiCT and find the corrector values can be reduced without sacrificing the security of the proof (Section 3.1).
- *Verification of multiple accounts can be batched in one.* To reduce the cost in the verification of multiple accounts, we propose a partial amortization for binary proofs to batch multiple relations (Section 3.2).
- *Inner-product relation is efficient in proving linear equation satisfiability.* Based on our observation, we generalize balance proofs to a linear equation satisfiability (Section 4.1 and 5.1). Specifically, we solve the overflow problem in inner-product relations under lattice settings, i.e., ensuring $-\sum_{j=1}^{k-1} 2^j f_j \mod q$ is small under a small q (more details in Section 4.1). Finally, we build a more efficient balance proof for RingCT protocol (Section 6.1).
- *The binary proof is redundant in ring signatures.* We analyze existing ring signatures (one-out-of-many proofs) and show the binary proof used in these approaches requires a larger parameter set. Furthermore, we prove it is sufficient to use a linear sum proof for ring signatures without the binary proof part in a one-out-of-many proof (Section 3.3).
- *Unbalanced linear sum proof is secure and efficient for ring signatures.* To propose an efficient ring signature scheme, we leverage the idea of relaxed relation and build our linear sum proof with an "unbalanced" relation (Section 4.2 and 5.2). Furthermore, we design an efficient ring signature scheme based on the unbalanced linear sum proof (Section 6.2). To apply our proposed techniques on RingCT protocols, we need to solve some additional problems such as double-spending. We describe how to address these issues and build a practical RingCT protocol for real-world post-quantum anonymous cryptocurrencies (Section 8).

1.2 Related Work

In anonymous cryptocurrencies, RingCT protocols [17, 27, 35, 38] adopt *range proofs* to show transaction amounts are valid and *ring signature*-like approaches to hide a spender's identity. We describe existing work in these two directions.

Range proofs. To guarantee the amount of each account in a confidential transaction is valid, range proofs [11, 28, 30] are used in RingCT protocols. By encapsulating the amounts in homomorphic commitments, the prover proves that 1) all the inputs and outputs are non-negative and 2) the sum of inputs equals outputs. The proofs can be succinct and efficient with a trusted setup [6, 18, 19,

29], but will undermine the decentralized property of blockchain systems at the same time where no particular trusted authority should be involved. Though the trusted setup can be replaced by a secure multi-party computation, the process is costly and may not be reusable when the application (i.e., circuit) is updated [6, 19]. Currently, the smallest proof without a trusted setup is the Bulletproofs protocol [11], which leverages the vector compression idea in [8]. However, these approaches fail to address quantum attacks as they are proposed based on discrete logarithm assumptions.

One of the most promising post-quantum cryptography candidates is lattice-based cryptography. Esgin et al. propose new range proofs in lattice settings based on the unbounded-message commitment (UMC) scheme and further adopt a new packing technique for efficient batch processing [14]. Unfortunately, the size of a UMC commitment is linear to the message size which is not suitable for large values such as amounts of different accounts. Besides, the batch processing in [14] is only efficient when the amounts of all accounts are committed together in a single commitment, while the amounts are usually committed separately in a RingCT protocol. The first *practical* lattice-based RingCT approach is MatRiCT [17] (applied in Hcash [31]). Instead of using UMC to commit to an amount directly, MatRiCT commits to the bits of an amount with HMC [15] and further adopts a balance proof with some "corrector values" to show the sums of inputs and outputs are equal. MatRiCT+ [16] further reduces the proof size and running time of MatRiCT by optimizing the underlying cyclotomic rings. Here we focus on MatRiCT since our improvements are based on the techniques proposed in MatRiCT, which are quite independent from the improvements in MatRiCT+. Though the efficiency has been improved compared with [14], a subtle issue prevents the use of MatRiCT and MatRiCT+ in general cases: the corrector values require additional range proofs when dealing with multiple input and output accounts (more details in Section 3.1).

Ring signatures. To hide the identity of a signer, ring signatures (one-out-of-many proofs) allow one to prove the knowledge of a secret key corresponding to an element in a set of public keys. The idea of the ring signature is proposed by Rivest, Shamir, and Tauman [33]. In discrete logarithm settings, logarithmic-size ring signatures [7, 20] have been used in different applications. Most of current anonymous cryptocurrencies are implemented based on discrete logarithm assumptions which cannot provide post-quantum security.

On the side of lattice settings, linear-size ring signatures have been proposed [25, 36], but these approaches are inefficient for large anonymous groups. Libert et al. [24] design a Merkle tree based accumulator and build a ZKP system for this accumulator. With these tools, logarithmic-size ring and group signatures are proposed. Furthermore, a linkable version of [24] (signatures created by the same signer can be linked) is introduced in [37]. Though the signature size of [24, 37] is logarithmic, the zero-knowledge arguments applied in the accumulator require multiple protocol iterations (multi-shot proofs) to get a negligible soundness error. Esgin et al. [15] introduce new tools for ZKPs to extend the discrete logarithm proof techniques in [20] to lattice settings. Logarithmic-size ring

signatures can be easily achieved with these new techniques. A further improvement in [14] makes the underlying ZKPs achieve a negligible soundness error at a single protocol iteration (i.e., the one-shot proof in Appendix A) and reduces the signature size accordingly. Following the blueprint of [14], MatRiCT [17] batches commitments in binary proofs and improves the rejection sampling to build a more efficient ring signature scheme. Besides, MatRiCT uses two sets of compatible parameters for the ring signature to reduce the size (discussed in Section 3.3).

2 Preliminaries

2.1 Notations

We use $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ to denote the ring of integers modulo q represented by the range $[-\frac{q-1}{2}, \frac{q-1}{2}]$. The rings are defined by $R = \mathbb{Z}[X]/(X^d + 1)$ and $R_q = \mathbb{Z}_q[X]/(X^d + 1)$ where $d > 1$ is a power of 2. Bold-face lower-case letters such as $\boldsymbol{a}$ and bold-face capital letters such as $\boldsymbol{A}$ are used to denote column vectors and matrices respectively. Commitments are denoted by capital letters such as C even though they may be vectors. We use $(\boldsymbol{a}, \boldsymbol{b})$ to denote appending vector $\boldsymbol{a}$ to $\boldsymbol{b}$. For a vector $\boldsymbol{a} = (a_0, \cdots, a_{k-1})$, the norms are defined as $\|\boldsymbol{a}\| = \sqrt{\sum_{i=0}^{k-1} a_i^2}$, $\|\boldsymbol{a}\|_1 = \sum_{i=0}^{k-1} |a_i|$, and $\|\boldsymbol{a}\|_\infty = \max_i |a_i|$. The norms of a polynomial are defined in a similar way as a vector. Suppose $x \in \mathbb{Z}_q$, we denote $\boldsymbol{x}^k = (1, x, x^2, \cdots, x^{k-1})$. Furthermore, the inner-product of two k-dimensional vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ is denoted as $\langle \boldsymbol{a}, \boldsymbol{b} \rangle = \sum_{i=0}^{k-1} a_i b_i$ and the Hadamard product is denoted as $\boldsymbol{a} \circ \boldsymbol{b} = (a_0 \cdot b_0, \cdots, a_{k-1} \cdot b_{k-1})$. The Kronecker's delta is denoted as $\delta_{j,i}$ such that $\delta_{j,i} = 1$ when $j = i$ and otherwise $\delta_{j,i} = 0$. $HW(x)$ denotes the Hamming weight of the coefficient vector of $x \in R$. Uniform distribution on a set S is denoted by $\mathcal{U}(S)$, and $a \leftarrow \mathcal{S}$ denotes sampling a from a distribution $\mathcal{S}$, or uniformly sampling from a set $\mathcal{S}$. $\mathcal{S}^{md}$ indicates that totally md coefficients are sampled to generate m polynomials in R of degree d. $\mathfrak{U}_\mathcal{B}$ denotes the set of polynomials in R with infinity norm at most $\mathcal{B} \in \mathbb{Z}^*$.

The challenge space in a Σ-protocol is defined as follows:

$$\mathcal{C} = \{x \in R : \deg(x) = d - 1 \wedge HW(x) = w \wedge \|x\|_\infty = p\}. \tag{1}$$

Clearly, we can observe $\|x\|_1 \leq pw$ and $|\mathcal{C}| = \binom{d}{w} \cdot (2p)^w$. We denote all non-zero challenges as $\mathcal{C}^*$.

Lemma 1. ***(Lemma 3 in [14])*** *For any $y_1, \cdots, y_n \in \mathcal{C}^*$, we have $\| \prod_{i=1}^n y_i \|_\infty \leq (2p)^n \cdot w^{n-1}$ and $\| \prod_{i=1}^n y_i \| \leq \sqrt{d} \cdot (2p)^n \cdot w^{n-1}$.*

2.2 Rejection Sampling

In Σ-protocols, a prover needs to encode its witness $\boldsymbol{b}$ as $\boldsymbol{f}$ with a challenge x and a masking vector $\boldsymbol{a}$, $\boldsymbol{f} = x\boldsymbol{b} + \boldsymbol{a}$. In the M-SIS assumption (described in

Section 2.3), it is important to hide the distribution of $\boldsymbol{b}$ from the distribution of $(x, \boldsymbol{f})$. The most commonly used approach is the rejection sampling to restrict the distribution of $(x, \boldsymbol{f})$ being independent of $\boldsymbol{b}$ by rejecting $\boldsymbol{f}$ which are out of bounds [26]. We summarize rejection sampling in Algorithm 1, where $T = \|\boldsymbol{b}\|$ and ϕ is a positive value to control the deviation of the normal distribution. Returning 1 means $\boldsymbol{f}$ passes the rejection sampling.

Algorithm 1 Rejection Sampling

Rej$(\boldsymbol{f}, \boldsymbol{b}, \phi, T)$

1: $\sigma = \phi T$; $\mu(\phi) = \exp(\frac{12}{\phi} + \frac{1}{2\phi^2})$; $u \leftarrow [0, 1)$
2: **if** $u > \frac{1}{\mu(\phi)} \cdot \exp(\frac{-2\langle \boldsymbol{f}, \boldsymbol{b}\rangle + \|\boldsymbol{b}\|^2}{2\sigma^2})$ **then**
3: Return $\perp$
4: **end if**
5: Return 1

2.3 M-SIS and M-LWE Problems

We define the two well-known lattice problems [22], module short integer solution (M-SIS) and module learning with errors (M-LWE), which our schemes' security relies on.

Definition 1. *M-SIS*(n, m, q, γ). *Given* $\boldsymbol{A} \leftarrow R_q^{n \times m}$, *the goal of the problem is to find* $\boldsymbol{z} \in R_q^m$ *such that* $\boldsymbol{A}\boldsymbol{z} = \boldsymbol{0} \mod q$ *and* $0 < \|\boldsymbol{z}\| \leqslant \gamma$.

Definition 2. *M-LWE*$(n, m, q, \mathcal{B})$. *Given* $\mathfrak{U}_{\mathcal{B}}$ *be a distribution over* R_q *and* $\boldsymbol{s} \leftarrow \mathfrak{U}_{\mathcal{B}}^n$ *be a secret key. Define* $\mathsf{LWE}(q, \boldsymbol{s})$ *as the distribution obtained by sampling* $\boldsymbol{a} \leftarrow R_q^n$, $e \leftarrow \mathfrak{U}_{\mathcal{B}}$ *and outputting* $(\boldsymbol{a}, \langle \boldsymbol{a}, \boldsymbol{s}\rangle + e)$. *The goal of the problem is to distinguish between* m *given samples from either* $\mathsf{LWE}(q, \boldsymbol{s})$ *or* $\mathcal{U}(R_q^n, R_q)$.

2.4 Hashed-Message Commitment

Let $n, m, \mathcal{B}, q$ be positive integers with $m > n$. Suppose a prover commits to v-dimensional vectors over R_q for $v \geqslant 1$. The instantiation of the hashed-message commitment (HMC) scheme [5, 17] is as follows:

- $\mathsf{CKeygen}(1^\lambda)$: Sample $\boldsymbol{G}_r \leftarrow R_q^{n \times m}$ and $\boldsymbol{G}_m \leftarrow R_q^{n \times v}$. Output $ck = \boldsymbol{G} = (\boldsymbol{G}_r, \boldsymbol{G}_m) \in R_q^{n \times (m+v)}$.
- $\mathsf{Commit}_{ck}(\boldsymbol{m})$: Sample $\boldsymbol{r} \leftarrow \{-\mathcal{B}, \cdots, \mathcal{B}\}^{md}$. Output $\boldsymbol{r}$ and $\mathrm{Com}_{ck}(\boldsymbol{m}, \boldsymbol{r}) = \boldsymbol{G} \cdot (\boldsymbol{r}, \boldsymbol{m}) = \boldsymbol{G}_r \cdot \boldsymbol{r} + \boldsymbol{G}_m \cdot \boldsymbol{m}$.
- $\mathsf{COpen}_{ck}(C, (y, \boldsymbol{m}', \boldsymbol{r}'))$: If $\|(\boldsymbol{m}', \boldsymbol{r}')\| \leqslant \gamma$ and $yC = \mathrm{Com}_{ck}(\boldsymbol{m}', \boldsymbol{r}')$ return 1, otherwise return 0.

Remarks. As all operations are conducted on R_q, the prover needs to sample md-many $\mathbb{Z}_q$ elements to build an m-dimensional R_q vector in Commit. Furthermore, the opening algorithm COpen does not simply check $C \stackrel{?}{=} \mathrm{Com}_{ck}(\boldsymbol{m}', \boldsymbol{r}')$ in common lattice-based schemes [12], but with a relaxation factor $y \in R_q$ as in [9, 14, 17]. This is due to the straightforward soundness proofs under lattice assumptions do not work. Thus, we use "relaxed relations" by relaxing the verification relation to overcome the complications. Besides, the verifier also needs to check the norm of the openings, $\|(\boldsymbol{m}', \boldsymbol{r}')\| \leqslant \gamma$, to ensure the hardness of M-SIS problem in Defination 1.

Lemma 2. ***(Lemma 2.3 in [17])*** *For a (large) set of appropriately chosen parameters, if M-LWE$(m-n, m, q, \mathcal{B})$ problem is hard then the HMC defined above is computationally hiding. If M-SIS$(n, m+v, q, 2\gamma)$ is hard, then the HMC defined above is computationally strong γ-binding to the same relaxation factor y.*

2.5 Vandermonde Matrix and One-Shot Proof [14]

A $(k+1)$-dimensional Vandermonde matrix $\boldsymbol{V}$ is defined as follows for some $x_0, \cdots, x_k \in R$:

$$\boldsymbol{V} = \begin{pmatrix} 1 & x_0 & \cdots & x_0^k \\ 1 & x_1 & \cdots & x_1^k \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_k & \cdots & x_k^k \end{pmatrix}. \tag{2}$$

Let $\mathrm{adj}(\boldsymbol{V})$ denotes adjugate matrix of $\boldsymbol{V}$ and $\det(\boldsymbol{V})$ denotes the determinant of $\boldsymbol{V}$. Considering the property $\mathrm{adj}(\boldsymbol{V}) \cdot \boldsymbol{V} = \det(\boldsymbol{V}) \cdot \boldsymbol{I}_{k+1}$, we have

$$\det(\boldsymbol{V}) = \prod_{0 \leq i < j \leq k} (x_j - x_i). \tag{3}$$

Let $(\Gamma_0, \cdots, \Gamma_k)$ be the last row of $\mathrm{adj}(\boldsymbol{V})$. Then

$$\Gamma_i = (-1)^{i+k} \prod_{\substack{0 \leq s < j \leq k \\ s, j \neq i}} (x_j - x_s). \tag{4}$$

Lemma 3. ***(Lemma 4 in [14])*** *Let $\kappa = \frac{k(k+1)}{2}$, we have $\|\det(\boldsymbol{V})\|_\infty \leq (2p)^\kappa w^{\kappa-1}$ when using the challenge space in Equation (1).*

The one-shot proof is a technique proposed in [14] to efficiently prove non-linear polynomial relations. Consider a k-degree polynomial relation with commitments $C_0 = \mathrm{Com}(\boldsymbol{m}_0; \boldsymbol{r}_0), \cdots, C_k = \mathrm{Com}(\boldsymbol{m}_k; \boldsymbol{r}_k)$. The prover encodes the message as $(\boldsymbol{f}, \boldsymbol{z}) \leftarrow (\sum_{i=0}^k x^i \boldsymbol{m}_i, \sum_{i=0}^k x^i \boldsymbol{r}_i)$ with a challenge x. The verifier checks the norms of $\boldsymbol{f}, \boldsymbol{z}$ and $\sum_{i=0}^k x^i C_i \stackrel{?}{=} \mathrm{Com}(\boldsymbol{f}; \boldsymbol{z})$. This protocol has $(k+1)$-special soundness as we can extract a witness in one shot with the following approach.

Considering $(k+1)$ accepted transactions with distinct challenges x_i's and responses $(\boldsymbol{f}_i, \boldsymbol{z}_i)$'s where $i \in [0,k]$ (C_i's are the same). We have the following relation

$$\begin{pmatrix} 1 & x_0 & \cdots & x_0^k \\ 1 & x_1 & \cdots & x_1^k \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_k & \cdots & x_k^k \end{pmatrix} \cdot \begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_k \end{pmatrix} = \begin{pmatrix} \mathrm{Com}(\boldsymbol{f}_0; \boldsymbol{z}_k) \\ \mathrm{Com}(\boldsymbol{f}_1; \boldsymbol{z}_k) \\ \vdots \\ \mathrm{Com}(\boldsymbol{f}_k; \boldsymbol{z}_k) \end{pmatrix}. \tag{5}$$

Let the Vandermonde matrix in Equation (5) be $\boldsymbol{V}$. By multiplying both sides of Equation (5) by $\mathrm{adj}(\boldsymbol{V})$, based on Equation (4), its last row becomes

$$\begin{aligned} \det(\boldsymbol{V}) \cdot C_k &= \sum_{i=0}^{k} \Gamma_i \mathrm{Com}(\boldsymbol{f}_i; \boldsymbol{z}_i) \\ = \mathrm{Com}(\sum_{i=0}^{k} \Gamma_i \boldsymbol{f}_i; \sum_{i=0}^{k} \Gamma_i \boldsymbol{z}_i) &:= \mathrm{Com}(\widehat{\boldsymbol{m}}_k; \widehat{\boldsymbol{r}}_k). \end{aligned} \tag{6}$$

Therefore, $(\widehat{\boldsymbol{m}}_k, \widehat{\boldsymbol{r}}_k)$ is an exact opening of yC_k with a relaxation factor $y = \det(\boldsymbol{V})$.

Lemma 4. ***(Lemma 5 in [14])*** *In Equation (6), the following holds for* $\kappa' = \frac{k(k-1)}{2}$*:*

$$\begin{aligned} \|\widehat{\boldsymbol{m}}_k\| &\leq d(k+1)(2p)^{\kappa'} w^{\kappa'-1} \cdot \max_i \|\boldsymbol{f}_i\|, \\ \|\widehat{\boldsymbol{r}}_k\| &\leq d(k+1)(2p)^{\kappa'} w^{\kappa'-1} \cdot \max_i \|\boldsymbol{z}_i\|. \end{aligned} \tag{7}$$

Note that the i-th row of the Vandermonde matrix ($i \neq k$) is much different from the form in Equation (4), which involves a function of x_i's in the $(-1)^{i+k}$ part. This hinders us from using the one-shot proof to derive the relaxed opening of C_i directly.

2.6 Amortized Relation [3]

The amortization technique to open *multiple* linear forms for essentially the price of *one*. In [3], Attema et al. describe two amortization techniques in discrete logarithm settings, amortized exponentiations and amortized homomorphisms. Here we focus on the first one and ignore randomness for simplicity. Consider the following relation

$$\mathcal{R}_{\mathrm{AmorExp}} = \left\{ \begin{array}{c} ((ck, (B_i, P_i)_{i=1}^k), ((\boldsymbol{b}_i)_{i=1}^k) : \\ (\mathrm{Com}_{ck}(\boldsymbol{b}_i) = B_i, g(\boldsymbol{b}_i) = P_i)_{i=1}^k) \end{array} \right\}, \tag{8}$$

where $g(\cdot)$ is a homomorphic function, it is equivalent to prove[5] $\mathrm{Com}_{ck}(\sum_{i=1}^{k} x^i \boldsymbol{b}_i) = \sum_{i=1}^{k} x^i B_i$ and $g(\sum_{i=1}^{k} x^i \boldsymbol{b}_i) = \sum_{i=1}^{k} x^i P_i$ with a challenge x. The prover can further uses one vector $\boldsymbol{a}$ to generate the response as $\boldsymbol{f} = \boldsymbol{a} + \sum_{i=1}^{k} x^i \boldsymbol{b}_i$ as a standard Σ-protocol.

Attema et al. proved the completeness, special soundness, and SHVZK prosperities of the above amortized Σ-protocol in discrete logarithm settings (Theorem 4 and Theorem 5 in [3]). In this paper, we leverage this idea in the balance proof and extent it to lattice settings.

3 Observations and Techniques

We first analyze the balance proof and ring signature scheme in MatRiCT [17]. Then we show the performance of these approaches can be further improved with our new techniques.

3.1 Corrector Values in Balance Proofs

In existing RingCT protocols, to prove a transaction is valid, a spender (prover) needs to show 1) all the inputs and outputs are non-negative and 2) the difference between inputs and outputs is zero. The former relation can be checked in a range proof while the latter one is quite simple as $\mathrm{Com}(a_1; *) + \mathrm{Com}(a_2; *) = \mathrm{Com}(a_1 + a_2; *)$ holds under a homomorphic commitment scheme. In lattice settings, some approaches use UMC to commit to an unbounded secret like amount [5, 14]. However, as the size of a UMC commitment grows linearly with the secret size, using a range proof directly is not practical in lattice-based RingCT protocols.

MatRiCT [17] commits to bits of each amount with HMC to avoid the cost of UMC. Thus, the former relation can be proved in a binary proof. For the latter one, it requires "corrector values" to ensure $\mathrm{Bits}(a_1) + \mathrm{Bits}(a_2)$ equals to $\mathrm{Bits}(a_1 + a_2)$ after some corrections. For instance, suppose a prover wants to prove that the following relations hold for M inputs and S outputs:

$$a_i \geq 0,\ \forall i \in [0, M); \quad \wedge \quad b_i \geq 0,\ \forall i \in [0, S); \tag{9}$$

$$\sum_{i=0}^{M-1} a_i = \sum_{i=0}^{S-1} b_i; \tag{10}$$

where a_i's are input accounts and b_i's are output accounts. A balance proof first converts each account into bits, $\boldsymbol{a}_i = (a_{i,0}, \cdots, a_{i,k-1}) \leftarrow \mathrm{Bits}(a_i)$ and $\boldsymbol{b}_i = (b_{i,0}, \cdots, b_{i,k-1}) \leftarrow \mathrm{Bits}(b_i)$, and commits to each $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$. Then, the prover shows 1) $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$ are binary vectors for Equation (9) and 2) Equation

[5] In discrete logarithm settings, it should be $\prod_{i=1}^{k} B_i^{x_i}$ and $\prod_{i=1}^{k} P_i^{x_i}$. Here we express in a additive group since it is more suitable under lattice settings.

(10) holds such that:

$$\sum_{i=0}^{M-1} a_i = \sum_{i=0}^{S-1} b_i \Longleftrightarrow \sum_{i=0}^{M-1}\sum_{j=0}^{k-1} 2^j a_{i,j} = \sum_{i=0}^{S-1}\sum_{j=0}^{k-1} 2^j b_{i,j}$$
$$\Longleftrightarrow \sum_{i=0}^{S-1} b_{i,j} - \sum_{i=0}^{M-1} a_{i,j} + \tau_j - 2\tau_{j+1} = 0, \quad \forall j \in [0,k),$$

where τ_j's are correct values to ensure $\sum_{i=0}^{S-1} b_{i,j} - \sum_{i=0}^{M-1} a_{i,j} + \tau_j - 2\tau_{j+1} = 0$ holds for all $j \in [0,k)$ and $\tau_0 = \tau_k = 0$.

The balance proof requires additional work to ensure τ_j's are properly generated. In general, the prover needs to ensure $\tau_j \in [-M+1, S-1]$ (Lemma 4.1 in [17]) with standard range proofs[6]. It is acceptable to embed τ_j's in the binary proof of $b_{i,j}$'s when $M = 1$ and $S \leq 2$, as with the Algorithm 8 and 9 in [17]. However, in other cases, the cost of range proofs is not negligible. Taking the state-of-the-art range proof in [14] as an example, the additional range proof requires 3 UMCs (the commitment of τ_j's needs to be included), a $(k-1)\log(S+M-1)(d/s)$-size vector (i.e., $f_j^{(i)}$'s in [14] where s refers to the number of packing slots in [14]), and a $3md$-size randomness (i.e., $\boldsymbol{z}$ in [14]). Even for a small range, the proof size is prohibitively large as the UMCs and HMCs cannot be batched together. More specifically, under the settings of [17], the range proof costs nearly 200KB, while other parts only cost about 100KB when $M = 2$ and $S = 3$.

One observation is that the *corrector values* $(\tau_0, \cdots, \tau_k)$ are unnecessary for balance proofs. To prove Equation (10) holds, one can simply prove

$$\begin{aligned} &\sum_{i=0}^{M-1}\sum_{j=0}^{k-1} 2^j a_{i,j} = \sum_{i=0}^{S-1}\sum_{j=0}^{k-1} 2^j b_{i,j} \\ \Longleftrightarrow &\sum_{j=0}^{k-1} 2^j \Big(\sum_{i=0}^{S-1} b_{i,j} - \sum_{i=0}^{M-1} a_{i,j} \Big) = 0. \end{aligned} \tag{11}$$

Let $c_j = \sum_{i=0}^{S-1} b_{i,j} - \sum_{i=0}^{M-1} a_{i,j}$. We can rewrite Equation (11) as $\sum_{j=0}^{k-1} 2^j c_j = 0$. The fact behind this idea is that though $\mathrm{Bits}(a_1) + \mathrm{Bits}(a_2) \neq \mathrm{Bits}(a_1 + a_2)$, we have $\langle \mathrm{Bits}(a_1), \mathbf{2}^k \rangle + \langle \mathrm{Bits}(a_2), \mathbf{2}^k \rangle = \langle \mathrm{Bits}(a_1 + a_2), \mathbf{2}^k \rangle$. Accordingly, we can fully remove the range proofs of τ_j's as well as the commitments and responses to τ_j's.

Additionally, the prover can also avoid sending the commitment of c_j's as it can be derived from the following equation:

$$\mathrm{Com}(c_0, \cdots, c_{k-1}; *) = \sum_{i=0}^{S-1} \mathrm{Com}(\boldsymbol{b}_i; *) - \sum_{i=0}^{M-1} \mathrm{Com}(\boldsymbol{a}_i; *). \tag{12}$$

[6] Esgin points out that the range proof can be replaced by an alternative approach (described in Appendix F).

Meanwhile, the range proofs of c_i's can be avoided when $\boldsymbol{a}_i$'s and $\boldsymbol{b}_i$'s are binary vectors since Equation (12) implies $c_i \in [-M, S]$.

When using the inner-product relation in lattice settings, a serious problem arises at the same time: after encoding c_j as $f_j = xc_j + d_j$ (d_j is a masking value and x is a challenge), $\sum_{j=0}^{k-1} 2^j f_j$ can be greater than q, i.e., $(\sum_{j=0}^{k-1} 2^j f_j \bmod q) \neq \sum_{j=0}^{k-1} 2^j f_j$. Accordingly, verifying $\sum_{j=0}^{k-1} 2^j f_j = x \sum_{j=0}^{k-1} 2^j c_j + \sum_{j=0}^{k-1} 2^j d_j$ on R_q may *not* imply $\sum_{j=0}^{k-1} 2^j c_j = 0$. A straightforward solution is to use a large q to avoid overflowing. However, such a solution will result in a large proof size, making it impractical for real-world applications. In this paper, we solve this problem with a new approach to find proper d_j's to ensure both f_j's and $\sum_{j=0}^{k-1} 2^j f_j$ are short at the same time. More details are given in Section 4.1.

The idea of using an inner-product equation to prove balance relations can be generalized to prove the satisfiability of a linear equation (Section 4.1 and 5.1). Besides, we also observe the range of corrector values can be limited to $\{-1, 0, 1\}$. As we do not adopt this approach in this paper, we only discuss it in Appendix E.

3.2 Randomness of masking values

In MatRiCT (as well as other RingCT-based cryptocurrencies such Monero), a spender needs to use a masking vector hide the amount of money in each account in a transaction. For example, when dealing with N accounts $(\boldsymbol{b}_i, \boldsymbol{r}_{b,i}, B_i)_{i=0}^{N-1}$ such that $B_i = \mathrm{Com}(\boldsymbol{b}_i; \boldsymbol{r}_{b,i})$, the prover needs to use N vectors, $(\boldsymbol{t}_i, \boldsymbol{r}_{t,i})_{i=0}^{N-1}$ to generate the responses $\boldsymbol{g}_i = x\boldsymbol{b}_i + \boldsymbol{t}_i$ and $\boldsymbol{z}_{b,i} = x\boldsymbol{r}_{b,i} + \boldsymbol{r}_{t,i}$ with a challenge x. Accordingly, the verifier needs to check $\mathrm{Com}(\boldsymbol{g}_i; \boldsymbol{z}_{b,i}) = xB_i + G_i$ holds for all $i \in [0, S)$. Since the verification is conducted separately for each i, all $\boldsymbol{z}_{b,i}$'s must be included in the proof, which increases the proof size when dealing with multiple accounts.

Our observation is it is possible to batch the verification of $\mathrm{Com}(\boldsymbol{g}_i; \boldsymbol{z}_{b,i}) = xB_i + G_i$ with the amortized technique in [3]. Unfortunately, since the binary relation $\boldsymbol{b}_i \circ (\boldsymbol{1} - \boldsymbol{b}_i) = 0$ is not homomorphic, it cannot be regarded as the $g(\cdot)$ in Equation (8). This brings us the idea of *partial amortization*: only using the batched verification for $\mathrm{Com}(\boldsymbol{g}_i; \boldsymbol{z}_{b,i}) = xB_i + G_i$ and leaving the binary relation part unchanged. Specifically, we show proving $\mathrm{Com}(\sum_{i=0}^{N-1} \zeta^i \boldsymbol{b}_i, \sum_{i=0}^{N-1} \zeta^i \boldsymbol{r}_{b,i}) = \sum_{i=0}^{N-1} \zeta^i B_i$ implies $\mathrm{Com}(\hat{\boldsymbol{b}}_i; \hat{\boldsymbol{r}}_{b,i}) = yB_i$ for a challenge ζ and a relaxation factor y in lattice settings. Since the non-homomorphic binary relation does not involve $\boldsymbol{r}_{b,i}$ (i.e., B_i), the prover can batch $\boldsymbol{r}_{b,i}$'s and only send one element $\boldsymbol{z}_b = \sum_{i=0}^{N-1} \zeta^i \boldsymbol{z}_{b,i}$.

Note that it is important to keep the masking form as $\boldsymbol{g}_i = x\boldsymbol{b}_i + \boldsymbol{t}_i$ for the non-homomorphic binary relation, which hinders us from using $\boldsymbol{g} = \boldsymbol{t} + \sum_{i=0}^{N-1} x^{i+1} \boldsymbol{b}_i$ in [3]. Therefore, the prover needs to send a commitment for the batched masking values $G = \mathrm{Com}(\sum_{i=0}^{N-1} \zeta^i \boldsymbol{t}_i; \sum_{i=0}^{N-1} \zeta^i \boldsymbol{r}_{t,i})$ to allow the verifier to check $\mathrm{Com}(\sum_{i=0}^{N-1} \zeta^i \boldsymbol{g}_i; \boldsymbol{z}_b) = x \sum_{i=0}^{S-1} \zeta^i B_i + G$. As a result, the proof size can

be reduced when dealing with multiple accounts since only one element, $\boldsymbol{z}_b$, is needed in the proof.

3.3 Binary Proof in Ring Signatures

In most of existing ring signatures [14, 15, 17], a one-out-of-many proof is used to show a prover (signer) knows an opening of a public key P_l in a public key set $(P_0, \cdots, P_{N-1})$. The idea for this proof is regarding a public key as a commitment to zero. Thus, by constructing a secret binary sequence $\boldsymbol{\delta} = (\delta_{l,0}, \cdots, \delta_{l,N-1})$ with Hamming weight 1, a prover proves 1) $\boldsymbol{\delta}$ is well-formed and 2) $\sum_{i=0}^{N-1} \delta_{l,i} P_i = P_l$ is a commitment to 0. A straightforward solution for the former relation is to use a binary proof to show $\boldsymbol{\delta}$ is a binary vector and $\sum_{i=0}^{N-1} f_i = \sum_{i=0}^{N-1} (x\delta_i + a_i) = x$ for a challenge x and some masking values a_i's where $\sum_{i=0}^{N-1} a_i = 0$. However, this approach is inefficient as the proof size is $O(N)$ due to the size of $\boldsymbol{\delta}$.

The efficient logarithmic-size ring signatures "compress" $\boldsymbol{\delta}$ to several shorter delta vectors and allow the verifier to "reconstruct" $\boldsymbol{\delta}$ with these vectors [14, 15, 17]. Suppose $N = \beta^k$. Write $l = (l_0, \cdots, l_{k-1})$ and $i = (i_0, \cdots, i_{k-1})$ as the representations in base β such that $\delta_{l,i} = \prod_{j=0}^{k-1} \delta_{l_j, i_j}$. Instead of proving that an N-size vector $\boldsymbol{\delta}$ is well-formed, the prover only needs to prove k-many β-size vectors, $(\delta_{l_j,0}, \cdots, \delta_{l_j,\beta-1})_{j=0}^{k-1}$, are well-formed, which reduces the proof size to $O(k\beta)$.

We have two observations. First, to ensure security, the binary proof requires a larger parameter set than other parts of the proof. This is due to 1) the hardness of the M-SIS problem and 2) $b(1-b) = 0$ may not hold in R_q for a smaller q [17]. Though the binary proof is simple, its larger parameters indicate a larger proof size. Motivated by this, we analyze ring signatures and find proving $\boldsymbol{\delta}$ being a *binary* sequence is redundant. For example, a signer can prove knowing the opening to $2P_l$ instead of P_l without sacrificing security. Generally speaking, it is sufficient to relax the one-out-of-many proof by proving the knowledge of an opening to $\sum_{i=0}^{N-1} b_i P_i$ in ring signatures, where b_i's are short and not all b_i's are 0. While reducing binary proof is nice in itself, we would like to highlight that it is particularly important for ring signatures. As "*the binary proof requires a much bigger modulus than (other parts of) the one-out-of-many proof*" [17], avoiding the binary proof fully releases ring signatures from the burden of large parameters. Therefore, instead of running a full one-out-of-many proof, ring signatures can use a much more efficient *linear sum* proof with a small modulus.

Our second observation is the linear sum proof may be difficult to adopt the "compressing" technique in [14,15,17] to achieve logarithmic-size ring signatures as there may not exist $(b_{j,0}, \cdots, b_{j,\beta-1})$ such that $b_i = \prod_{j=0}^{k-1} b_{j,i_j}$ for all $i \in [0, N)$ and finding such a solution can be very inefficient. This brings us to the idea of adopting "unbalanced" relations as in relaxed proofs: using a stricter relation in proving, but checking the original relation in verifying. For instance, as a linear sum relation is sound for ring signatures and a one-out-of-many relation is stricter than the linear sum relation, a prover can use $b_i = \delta_{l,i}$ in the one-out-

of-many relation to generate a proof. The verifier checks the linear sum relation of the proof instead of the one-out-of-many relation.

Though our "unbalanced" relation is derived from relaxed relations, the motivations behind are different. In our approach, we start from the *verifier's* side and show verifying a linear sum proof suffices in ring signatures. To improve the efficiency, we restrict the prover's relation and require the prover to run under a one-out-of-many relation. The key idea is to find a *strict* and *efficient* relation for provers. On the other hand, existing relaxed proofs start from the *prover's* side and find straightforward soundness proofs do not work. They need to relax the relation on the verifier's side to overcome the complications. The key is to find a *relaxed* but *sound* relation for verifiers. Thus we use the term "unbalanced relations" to distinguish with relaxed relations. We describe the unbalance linear sum proof in Section 4.2 and 5.2

Notice that in the linkable version of our ring signatures, the verifier can further ensure a one-out-of-many relation with two additional checks: 1) only one correct serial number is included, which implies exactly one b_i is non-zero; and 2) $\sum_{i=0}^{N-1} b_i = 1$, which ensures the non-zero b_i is 1 (step 23 in Protocol 4). We show more details in Section 8.

4 New Techniques for RingCT Protocols

Based on the ideas in Section 3, we propose two general techniques. The RingCT protocol can be regarded as an application of these techniques.

4.1 Linear Equation Satisfiability

Definition. Let N be a positive integer and $\omega_0, \cdots, \omega_{N-1}$ be (public) integers. The linear function is defined as:

$$F(X_0, \cdots, X_{N-1}) = \sum_{i=0}^{N-1} \omega_i X_i. \tag{13}$$

The linear equation satisfiability is to prove the knowledge of the witness $(b_i)_{i=0}^{N-1}$ such that $F(b_0, \cdots, b_{N-1}) = 0$.

To support b_i's with a wide range in lattice settings, we also use the HMC to commit to the bits of b_i's with $B_i = \mathrm{Com}(\boldsymbol{b}_i; *)$, where $\boldsymbol{b}_i$ is the binary representation of b_i. Thus, $F(b_0, \cdots, b_{N-1})$ can be rewritten as:

$$F'(\boldsymbol{b}_0, \cdots, \boldsymbol{b}_{N-1}) = \sum_{i=0}^{N-1} \Big(\omega_i \cdot \langle \boldsymbol{2}^k, \boldsymbol{b}_i \rangle\Big). \tag{14}$$

Definition 3. *The following defines the linear equation relations, proving $\mathcal{R}_{LE}$ and relaxed opening $\mathcal{R}'_{LE}$:*

$$\mathcal{R}_{LE}(\mathcal{T}) = \left\{ \begin{array}{c} ((ck, (\omega_i, B_i)_{i=0}^{N-1}), (\boldsymbol{b}_i, \boldsymbol{r}_{b,i})_{i=0}^{N-1}) : \boldsymbol{b}_i \in \{0,1\}^k \wedge \|\boldsymbol{r}_{b,i}\| \leq \mathcal{T} \\ \wedge B_i = \mathrm{Com}_{ck}(\boldsymbol{b}_i; \boldsymbol{r}_{b,i}) \wedge F'(\boldsymbol{b}_0, \cdots, \boldsymbol{b}_{N-1}) = 0 \end{array} \right\},$$

$$\mathcal{R}'_{LE}(\widehat{\mathcal{T}}) = \left\{ \begin{array}{c} ((ck, (\omega_i, B_i)_{i=0}^{N-1}), (y, (\boldsymbol{b}_i, \widehat{\boldsymbol{r}}_{b,i})_{i=0}^{N-1}) : \boldsymbol{b}_i \in \{0,1\}^k \wedge \|\widehat{\boldsymbol{r}}_{b,i}\| \leq \widehat{\mathcal{T}} \\ \wedge y \in \mathcal{C}^* \wedge yB_i = \mathrm{Com}_{ck}(y\boldsymbol{b}_i; \widehat{\boldsymbol{r}}_{b,i}) \wedge F'(\boldsymbol{b}_0, \cdots, \boldsymbol{b}_{N-1}) = 0 \end{array} \right\}.$$

where $\mathcal{T}$ and $\widehat{\mathcal{T}}$ are norm bounds of $\boldsymbol{r}_{b,i}$ and $\widehat{\boldsymbol{r}}_{b,i}$ respectively.

Inner-product based proof. Based on the idea discussed in Section 3.1, we propose an inner-product based proof for the linear equation satisfiability. The $\mathcal{R}_{LE}$ indicates two important relations: 1) B_i's are commitments to bits and 2) $F'(\boldsymbol{b}_0, \cdots, \boldsymbol{b}_{N-1}) = 0$. The former one can be proved in a binary proof. For the second relation we can rewrite Equation (14) as

$$\begin{aligned} F'(\boldsymbol{b}_0, \cdots, \boldsymbol{b}_{N-1}) = 0 &\Longleftrightarrow \sum_{i=0}^{N-1} \left(\omega_i \sum_{j=0}^{k-1} 2^j b_{i,j}\right) = 0 \\ \Longleftrightarrow \sum_{j=0}^{k-1} \left(2^j \cdot \sum_{i=0}^{N-1} \omega_i b_{i,j}\right) &= \sum_{j=0}^{k-1} 2^j c_j = 0, \end{aligned} \tag{15}$$

where $b_{i,j}$ is the j-th element of $\boldsymbol{b}_i$ and $c_j = \sum_{i=0}^{N-1} \omega_i b_{i,j}$. Denote $\boldsymbol{c} = (c_0, \cdots, c_{k-1})$. The verifier can compute the commitment of $\boldsymbol{c}$ with ω_i's and B_i's: $C = \mathrm{Com}(\boldsymbol{c}; *) = \sum_{i=0}^{N-1} \omega_i B_i$. Let $\boldsymbol{f} = x\boldsymbol{c} + \boldsymbol{d}$ with some masking values $\boldsymbol{d} = (d_0, \cdots, d_{k-1})$ and a challenge x, $D = \mathrm{Com}(\boldsymbol{d}; *)$, $d_{sum} = \langle \boldsymbol{d}, \boldsymbol{2}^k \rangle$. We have

$$\begin{aligned} &\mathrm{Com}(\boldsymbol{f}; *) = \mathrm{Com}(x\boldsymbol{c} + \boldsymbol{d}; *) = xC + D, \\ &\langle \boldsymbol{f}, \boldsymbol{2}^k \rangle = \langle x\boldsymbol{c} + \boldsymbol{d}, \boldsymbol{2}^k \rangle = x\langle \boldsymbol{c}, \boldsymbol{2}^k \rangle + \langle \boldsymbol{d}, \boldsymbol{2}^k \rangle = d_{sum}, \end{aligned} \tag{16}$$

which ensure $F'(\boldsymbol{b}_0, \cdots, \boldsymbol{b}_{N-1}) = 0$ holds. Note that the prover can avoid sending f_0. The verifier computes $f_0 = d_{sum} - \sum_{j=1}^{k-1} 2^j f_j$ and only checks the first equation in (16).

Overflow issue. One important issue for the second equation is that it may not imply $\langle \boldsymbol{c}, \boldsymbol{2}^k \rangle = 0$ when verifying on R_q for a smaller q, i.e., $q < 2^k$. Here we propose two solutions.

- Our first approach is simple and straightforward: the prover computes and sends d_{sum} on R to avoid the overflow problem. Accordingly, the verifier computes f_0 on R and checks $f_0 \in R_q$.
- Our second approach can avoid sending d_{sum} by finding short d_j's while ensuring $\langle \boldsymbol{d}, \boldsymbol{2}^k \rangle = 0$. Specifically, the prover samples $(d'_j)_{j=1}^{k-1}$ and sets $d'_0 = d'_k = 0$. By setting $d_j = d'_j - 2d'_{j+1}$, we have $\langle \boldsymbol{d}, \boldsymbol{2}^k \rangle = \sum_{j=0}^{k-1} 2^j d_j - \sum_{j=1}^{k} 2^j d_j = d_0 - 2^k d_k = 0$. Therefore, the prove can avoid sending d_{sum} and compute $f_0 = -\sum_{j=1}^{k-1} 2^j f_j$. Notice that the norms of d_j will be bigger than the first approach (but still acceptable) which indicates a less strict soundness. In our second approach, though the form of d_j's is very similar to the form of corrector values in Equation (11), the intuitions behind are different. MatRiCT uses τ_j's to prove a stronger relation that each bit is 0 in Equation (11), i.e., $c_j + \tau_j - 2\tau_{j+1} = 0$. It further needs masking values to encode $a_{i,j}$'s, $b_{i,j}$'s, and τ_j's respectively. In our approach, we prove the original relation $\sum_{j=0}^{k-1} c_j = 0$. d_j's are used for masking c_j's instead of correcting each bit. No corrector values are required and the additional range proofs can be avoided.

Partial amortization for binary proofs. As discussed in Section 3.1, we propose partially amortized binary proofs to show B_i's are commitments to bits. The binary relation can be written as

$$\mathrm{Com}(\boldsymbol{b}_i) = B_i \quad \wedge \quad \boldsymbol{b}_i \circ (\mathbf{1} - \boldsymbol{b}_i) = \mathbf{0}, \quad \forall i \in [0, N). \tag{17}$$

For the latter relation, the prover encodes $\boldsymbol{b}_i$ as $\boldsymbol{g}_i = x\boldsymbol{b}_i + \boldsymbol{t}_i$ with a challenge x and a masking vector $\boldsymbol{t}_i$, which further allows the prover to check $\mathrm{Com}((\boldsymbol{g}_i \circ (x \cdot \mathbf{1} - \boldsymbol{g}_i))_{i=0}^{N-1}; *) = xE + F$, where $E = \mathrm{Com}((\boldsymbol{t}_i \circ (\mathbf{1} - 2\boldsymbol{b}_i))_{i=0}^{N-1}; *)$ and $F = \mathrm{Com}((-\boldsymbol{t}_i \circ \boldsymbol{t}_i)_{i=0}^{N-1}; *)$. This works the same as a standard binary proof.

The former relation is equivalent to $\mathrm{Com}(\sum_{i=0}^{N-1} \zeta^i \boldsymbol{b}_i; *) = \sum_{i=0}^{N-1} \zeta^i B_i$ with a challenge ζ. Since $\boldsymbol{g}_i = x\boldsymbol{b}_i + \boldsymbol{t}_i$, the prover needs to send the commitment of the batched masking vectors, $G = \mathrm{Com}(\sum_{i=0}^{N-1} \zeta^i \boldsymbol{t}_i; *)$, to allow the verifier to check $\mathrm{Com}(\sum_{i=0}^{N-1} \zeta^i \boldsymbol{g}_i) = x \sum_{i=0}^{N-1} \zeta^i B_i + G$. We briefly describe the partially amortized binary proof in Protocol 2.

Protocol 2 Partially Amortized Binary Proof (Sketch)

$\mathcal{P}((B_i)_{i=0}^{N-1}, (\boldsymbol{b}_i)_{i=0}^{N-1})$		$\mathcal{V}((B_i)_{i=0}^{N-1})$
	$\xleftarrow{\quad\zeta\quad}$	$\zeta \leftarrow \mathcal{C}$
Sample $\boldsymbol{t}_i$'s		
$E = \mathrm{Com}((\boldsymbol{t}_i \circ (\mathbf{1} - 2\boldsymbol{b}_i))_{i=0}^{N-1}; *)$		
$F = \mathrm{Com}((-\boldsymbol{t}_i \circ \boldsymbol{t}_i)_{i=0}^{N-1}; *)$		
$G = \mathrm{Com}(\sum_{i=0}^{S-1} \zeta^i \boldsymbol{t}_i; *)$		
	$\xrightarrow{\quad E, F, G \quad}$	
	$\xleftarrow{\quad x \quad}$	$x \leftarrow \mathcal{C}$
$\boldsymbol{g}_i = x\boldsymbol{b}_i + \boldsymbol{t}_i$		
	$\xrightarrow{\quad \boldsymbol{g}_i \quad}$	
		$xE + F \stackrel{?}{=} \mathrm{Com}((\boldsymbol{g}_i \circ (x \cdot \mathbf{1} - \boldsymbol{g}_i))_{i=0}^{N-1}; *)$
		$x \sum_{i=0}^{N-1} \zeta^i B_i + G \stackrel{?}{=} \mathrm{Com}(\sum_{i=0}^{N-1} \zeta^i \boldsymbol{g}_i; *)$

Note that in our binary proof, we do not batch the commitments as $\mathrm{Com}((\boldsymbol{b}_i)_{i=0}^{N-1}, (\boldsymbol{t}_i \circ (\mathbf{1} - 2\boldsymbol{b}_i))_{i=0}^{N-1}; *)$ and $\mathrm{Com}((\boldsymbol{t}_i)_{i=0}^{N-1}, (-\boldsymbol{t}_i \circ \boldsymbol{t}_i)_{i=0}^{N-1}; *)$ as [17] since verifying $\mathrm{Com}((\boldsymbol{f}_i)_{i=0}^{N-1}; *) = x\mathrm{Com}((\boldsymbol{b}_i)_{i=0}^{N-1}; *) + \mathrm{Com}((\boldsymbol{t}_i)_{i=0}^{N-1}; *)$ is redundant (it has been checked on B_i and G as the last step of verification).

Remarks. In the security analysis (soundness), if we use the one-shot proof [14] directly to extract the relaxed opening of B_i's, we will end up with a painful process to compute the i-th row elements in the Vandermonde matrix for the norm bounds. Here we use a trick to swap the i-th row with the last one to get a new Vandermonde matrix. Note the determinants of the two matrices are only different in sign, which enables us to use the same relaxation value for all B_i's. More details are presented in Appendix B.

4.2 Ring Signature

Definition. Let $\boldsymbol{r}$ be a private key and P_l be the corresponding public key in a public key set $\boldsymbol{P} = (P_0, \cdots, P_{N-1})$ for some $N \geq 1$ and $0 \leq l < N$. The goal of ring signatures is to show the knowledge of a secret key(s) corresponding to a public key(s) in $\boldsymbol{P}$. Based on the idea in Section 3.3, we show that proving the knowledge of an opening of a *short non-zero linear sum* relation of the public keys suffices for ring signatures, i.e., knowing some bounded b_i's and an opening to $\sum_{i=0}^{N-1} b_i P_i$ where at least one b_i is not zero. This is formally given in Lemma 5.

Lemma 5. *In ring signatures, if the commitment scheme is computational hiding and γ-binding, then it is hard to efficiently extract $(b_i)_{i=0}^{N-1}$ and an opening to $\sum_{i=0}^{N-1} b_i P_i$ with non-negligible probability, such that $b_i \in [-\mathcal{B}_{LS}, \mathcal{B}_{LS}]$ and at least one b_i is not zero, with respect to insider corruption*[7] *in the random oracle model.*

Proof. Assume there exists a PPT adversary $\mathcal{F}$ that can efficiently extract b_i's and a valid opening $(\mathbf{0}, \boldsymbol{s})$ of $\sum_{i=0}^{N-1} b_i P_i$ with non-negligible probability, then we have a collision finder $\mathcal{A}$ which can break the binding property of the HMC commitment scheme, and solve the M-SIS problem accordingly.

Specifically, $\mathcal{A}$ samples $\boldsymbol{r} \leftarrow \{-\mathcal{B}, \cdots, \mathcal{B}\}$ and computes an invalid public key $P_l = \mathrm{Com}_{ck}(1, 0, \cdots, 0; \boldsymbol{r})$. Due to the hiding probability of the commitment scheme, $\mathcal{F}$ cannot distinguish P_l with other public keys. By calling $\mathcal{F}$, $\mathcal{A}$ gets $(b_i)_{i=0}^{N-1}$ and a valid opening $(\mathbf{0}, \boldsymbol{s})$ of $\sum_{i=0}^{N-1} b_i P_i$. With non-negligible probability, we have $b_l \neq 0$ since $\mathcal{F}$ can only make polynomially many registration queries to $\mathcal{A}$ (calling RKeygen). Then, $\mathcal{A}$ uses all private keys but $\boldsymbol{r}_l$ to compute $\boldsymbol{s}' = \boldsymbol{s} - \sum_{i \neq l} b_i \boldsymbol{r}_i$. Since $b_l \neq 0$, we have a binding collision for the commitment scheme, $((b_l, 0, \cdots, 0), b_l \boldsymbol{r})$ and $(\mathbf{0}, \boldsymbol{s}')$. More details about the security reduction is presented in Appendix D.

Remarks. The adversary $\mathcal{A}$ interacts as a collision finder with the HMC challenger and as a ring signature challenger with the $\mathcal{F}$. Thus, $\mathcal{A}$ can access all private keys by calling RKeygen as these key pairs will not help to find a collision without a signature forgery [14, 17, 20]. Besides, since b_i's are important to compute the HMC collision, we require $\mathcal{F}$ also provide b_i's along with the forgery $\boldsymbol{s}$ (i.e., $\mathcal{F}$ here is not exactly as a ring signature forger). More details of $\mathcal{F}$ extracting b_i's are presented in the proof of Theorem 4 (here $\mathcal{F}$ works as the adversary $\mathcal{A}$ in Theorem 4).

Unbalanced Linear Sum Proof. We further leverage the idea of unbalanced relations in Section 3.3 to propose a logarithmic-size unbalanced linear sum proof, i.e., the prover uses a one-out-of-many relation to run the protocol by setting $b_i = \delta_{l,i}$ and the verifier checks under a linear sum relation. To ensure

[7] The attacker can obtain private keys to some public keys with corruption queries. Accordingly, the signature forgery should not include these "corrupted" public keys in its ring.

at least one b_i is not zero, the verifier checks whether $\|\boldsymbol{b}\| > 0$ in the opening. The unbalanced linear sum relations are defined as follows:

Definition 4. *The following defines the unbalanced relations for our unbalanced linear sum proof, proving $\mathcal{R}_{LS}$ and relaxed opening $\mathcal{R}'_{LS}$:*

$$\mathcal{R}_{LS}(\mathcal{T}) = \left\{ \begin{array}{c} ((ck, \boldsymbol{P}), (l, \boldsymbol{r})) : \\ l \in [0, N) \wedge \|\boldsymbol{r}\| \leq \mathcal{T} \wedge P_l = \mathrm{Com}_{ck}(\boldsymbol{0}; \boldsymbol{r}) \end{array} \right\},$$

$$\mathcal{R}'_{LS}(\widehat{\mathcal{T}}_b, \widehat{\mathcal{T}}_r) = \left\{ \begin{array}{l} ((ck, \boldsymbol{P}), (y, \boldsymbol{b}, \widehat{\boldsymbol{r}})) : \|\boldsymbol{b}\| > 0 \wedge \|b_i\| \leq \widehat{\mathcal{T}}_b \wedge \|\widehat{\boldsymbol{r}}\| \leq \widehat{\mathcal{T}}_r \wedge \\ y \sum_{i=0}^{N-1} b_i P_i = \mathrm{Com}_{ck}(\boldsymbol{0}; \widehat{\boldsymbol{r}}) \wedge y \text{ is a product of } x_i \in \mathcal{C}^* \end{array} \right\},$$

where $\mathcal{T}$, $\widehat{\mathcal{T}}_b$, and $\widehat{\mathcal{T}}_r$ are norm bounds of $\boldsymbol{r}$, b_i, and $\widehat{\boldsymbol{r}}$ respectively.

In our unbalanced linear sum proofs, a prover can directly apply the "compressing" technique in [14,15,17] to achieve a logarithmic-size proof. Specifically, the prover first finds and commits to k-many sequences $(\delta_{l_j,0}, \cdots, \delta_{l_j,\beta-1})_{j=0}^{k-1}$ which allow the verifier to reconstruct $\boldsymbol{\delta}$ base on $\delta_{l,i} = \prod_{j=0}^{k-1} \delta_{l_j,i_j}$ under a one-out-of-many relation. After receiving a challenge x, the prover's response contains $f_{j,i} = x\delta_{l_j,i} + a_{j,i}$ with some masking values $a_{j,i}$'s. Let $\boldsymbol{\delta'} = (\delta_{l_0,0}, \cdots, \delta_{l_{k-1},\beta-1})$, $\boldsymbol{a} = (a_{0,0}, \cdots, a_{k-1,\beta-1})$, and $\boldsymbol{f} = (f_{0,0}, \cdots, f_{k-1,\beta-1})$. To ensure $\boldsymbol{\delta'}$ is well-formed, the prover shows the following equations hold:

$$\begin{aligned} &\mathrm{Com}(\boldsymbol{f}; *) = \mathrm{Com}(x\boldsymbol{\delta'} + \boldsymbol{a}; *) = xB + A; \\ &\sum_{i=0}^{\beta-1} f_{j,i} = x \sum_{i=0}^{\beta-1} \delta_{l_j,i} + \sum_{i=0}^{\beta-1} a_{j,i} = x + \sum_{i=0}^{\beta-1} a_{j,i}, \quad \forall j \in [0, k). \end{aligned} \tag{18}$$

The second equation ensures at least one element in $(\delta_{l_j,0}, \cdots, \delta_{l_j,\beta-1})$ is not 0 for all j's as $\sum_{i=0}^{\beta-1} f_{j,i} = x + \sum_{i=0}^{\beta-1} a_{j,i}$ implies $\sum_{i=0}^{\beta-1} \delta_{l_j,i} = 1$. Moreover, proving $\boldsymbol{\delta'}$ being "short" is done in the norm check of HMC (presented later in steps 24 and 25 of Protocol 4). Besides, the binary proof for $\boldsymbol{\delta'}$ is avoided here as we do not require the reconstructed $\boldsymbol{\delta}$ being a binary vector under the linear sum relation. Furthermore, the second equation is not a necessary condition for the linear sum relation. However, based on the unbalanced relations in Section 3.3, the prover can efficiently show the second equation holds with a one-out-of-many relation. Other steps such as reconstructing $\boldsymbol{\delta}$ and checking $\sum_{i=0}^{N-1} \delta_{l,i} P_i$ being a commitment to zero are exactly same as the one-out-of-many proofs in [14, 15, 17], which are presented in Section 5.2.

We can further adopt some techniques to reduce the proof size. First, choosing $a_{j,i}$'s such that $\sum_{i=0}^{\beta-1} a_{j,i} = 0$ can avoid the cost of sending $\sum_{i=0}^{\beta-1} a_{j,i}$ in Equation (18). Moreover, the prover only needs to send $(a_{j,i})_{i=1}^{\beta-1}$ which allows the verifier to rebuild $a_{j,0}$'s with $a_{j,0} = -\sum_{i=1}^{\beta-1} a_{j,i}$ for all $j \in [0, k)$ without further checking the second equation in (18).

5 Lattice-based Proofs

We formally describe our balance proof and ring signature for RingCT protocols in lattice settings. In this paper, we separate the two protocols for a clear expression.

5.1 Linear Equation Satisfiability

We formally present our linear equation satisfiability protocol in Protocol 3 (we describe with our first approach to solve the overflow issue in Section 4.1). Specifically, steps 8 to 12 generate and commit to masking values $t_{i,j}$'s for the binary proof of $\boldsymbol{b}$. c_j's in Equation (15) are derived in step 12 and their masking values, d_j's, are generated in step 14. Note that in steps 16 and 17, we do not batch the binary proof commitments as $\mathrm{Com}(\boldsymbol{b}, \boldsymbol{t} \circ (\boldsymbol{1} - 2\boldsymbol{b}); *)$ and $\mathrm{Com}(\boldsymbol{t}, -\boldsymbol{t} \circ \boldsymbol{t}; *)$.

After receiving the challenge x, the prover generates the responses based on steps 18 to 27. As $\langle \boldsymbol{f}, \boldsymbol{2}^k \rangle = d_{sum}$ holds, the prover can avoid sending f_0 in steps 20 and 21. In step 23, the randomness for $\boldsymbol{c}$ (i.e., $\boldsymbol{r}_c$) is derived based on $\boldsymbol{r}_{b,i}$'s since $c_j = \sum_{i=0}^{N-1} \omega_i b_{i,j}$.

Finally, the verifier generates f_0 in step 32 to ensure $\langle \boldsymbol{f}, \boldsymbol{2}^k \rangle = d_{sum}$ holds (i.e., $F'(\boldsymbol{b}_0, \cdots, \boldsymbol{b}_{N-1}) = 0$). Here he also needs to run on R instead of R_q to avoid the overflow problem and returns false if f_0 is not in R_q (step 35). The commitment of $\boldsymbol{c}$ (step 33) is derived based on B_i's. Step 39 ensures f_i's are properly generated from c_i's and the last two steps ensure $\boldsymbol{b}_i$'s are binary vectors.

Theorem 1. *Let* $\kappa = N(N-1)/2$ *and* $\gamma_{LE} = 2^{\kappa+1} N\mathcal{B}p^{\kappa} w^{\kappa-1} md^2 \phi_3 (md(\|\boldsymbol{\omega}\|_1^2 + N + 1))^{1/2} \sum_{i=0}^{N-1} (wp)^i$ *and the HMC is hiding and* γ_{LE}*-binding. Protocol 3 has* $(3, N+1)$*-special soundness for relations* $\mathcal{R}_{LE}(\mathcal{B}\sqrt{md})$ *and* $\mathcal{R}'_{LE}(\gamma_{LE})$ *with a completeness error* $1 - 1/(\mu(\phi_1)\mu(\phi_2)\mu(\phi_3))$ *defined in Lemma 8.*

The proof for Theorem 1 is given in Appendix B.

We can easily switch to our second approach to solve the overflow issue. Specifically, in step 14, the prover needs to sample $(d'_j)_{j=1}^{k-1}$ from $D^d_{\phi_2 T_2}$, sets $d'_0 = d'_k = 0$.and computes $d_j = d'_j - 2d'_{j+1}$. As $\langle \boldsymbol{d}, \boldsymbol{2}^k \rangle = 0$, d_{sum} is no longer needed in the rest of the protocol (by regarding $d_{sum} = 0$). Since $d_j = d'_j - 2d'_{j+1}$ and d'_j and d'_{j+1} are sampled from $D^d_{\phi_2 T_2}$, except with negligible probability, we have $\|f_j\| \leq 6\phi_2 T_2 \sqrt{d}$ based on Lemma 7. Accordingly, we need to loose the norm bound to $6\phi_2 T_2 \sqrt{d}$.

We can also prove the security prosperities of the above protocol in a similar way as Theorem 1. Note that the only difference is SHVZK of f_j's. Here we need to sample $(f'_j)_{j=1}^{k-1}$ from $D^d_{\phi_2 T_2}$ and set $f'_0 = f'_k = 0$. By writing $f_j = f'_j - 2f'_{j+1}$, we can get $\sum_{j=0}^{k-1} 2^j f_j = 0$. Meanwhile, since $f_j = f'_j - 2f'_{j+1}$, the distribution of $(f_1, \cdots, f_{k-1})$ is statistically close to the real distributions based on Lemma 7.

Protocol 3 Linear Equation Satisfiability

	$\mathcal{P}_{LE}(ck, (\omega_i, B_i)_{i=0}^{N-1}, (\boldsymbol{b}_i, \boldsymbol{r}_{b,i})_{i=0}^{N-1})$		$\mathcal{V}_{LE}(ck, (\omega_i, B_i)_{i=0}^{N-1})$
1:		$\xleftarrow{\quad\zeta\quad}$	$\zeta \leftarrow \mathcal{C}$
2:	$\boldsymbol{\omega} = (\omega_0, \cdots, \omega_{N-1})$		
3:	$T_1 = p\sqrt{wkN}$		
4:	$T_2 = \max(-\sum_{\omega_i<0}\omega_i, \sum_{\omega_i>0}\omega_i)p\sqrt{wk}$		
5:	$T_3 = \mathcal{B}wp\sqrt{md(\Vert\boldsymbol{\omega}\Vert_1^2 + N + 1)}$		
6:	$\boldsymbol{r}_b \leftarrow \{-\mathcal{B}, \cdots, \mathcal{B}\}^{md}$		
7:	$\boldsymbol{r}_t, \boldsymbol{r}_d \leftarrow D_{\phi_3 T_3}^{md}$		
8:	**for** $i = 0$ to $N-1$ **do**		
9:	$\quad \boldsymbol{t}_i \leftarrow D_{\phi_1 T_1}^{kd}, \boldsymbol{r}_{t,i} \leftarrow D_{\phi_3 T_3}^{md}$		
10:	$\quad G_i = \mathrm{Com}_{ck}(\boldsymbol{t}_i; \boldsymbol{r}_{t,i})$		
11:	**end for**		
12:	$c_j = \sum_{i=0}^{N-1}\omega_i b_{i,j}, \forall j \in [0, k)$		
13:	$\boldsymbol{b} = (\boldsymbol{b}_i)_{i=0}^{N-1}, \boldsymbol{t} = (\boldsymbol{t}_i)_{i=0}^{N-1}$		
14:	$\boldsymbol{d} \leftarrow D_{\phi_2 T_2}^{d}, d_{sum} = \langle \boldsymbol{d}, \boldsymbol{2}^k \rangle \in R$		
15:	$D = \mathrm{Com}_{ck}(\boldsymbol{d}; \boldsymbol{r}_d), G = \sum_{i=0}^{N-1}\zeta^i G_i$		
16:	$E = \mathrm{Com}_{ck}(\boldsymbol{t} \circ (\boldsymbol{1} - 2\boldsymbol{b}); \boldsymbol{r}_b)$		
17:	$F = \mathrm{Com}_{ck}(-\boldsymbol{t} \circ \boldsymbol{t}; \boldsymbol{r}_t)$		
		$\xrightarrow{\quad d_{sum}, D, E, F, G\quad}$	
		$\xleftarrow{\quad x\quad}$	$x \leftarrow \mathcal{C}$
18:	$\boldsymbol{g} = x\boldsymbol{b} + \boldsymbol{t}$		
19:	$\mathrm{Rej}(\boldsymbol{g}, x\boldsymbol{b}, \phi_1, T_1)$		
20:	$\boldsymbol{c}_1 = (c_j)_{j=1}^{k-1}, \boldsymbol{d}_1 = (d_j)_{j=1}^{k-1}$		
21:	$\boldsymbol{f}_1 = x\boldsymbol{c}_1 + \boldsymbol{d}_1$		
22:	$\mathrm{Rej}(\boldsymbol{f}_1, x\boldsymbol{c}_1, \phi_2, T_2)$		
23:	$\boldsymbol{r}_c = \sum_{i=0}^{N-1}\omega_i \boldsymbol{r}_{b,i}$		
24:	$\boldsymbol{z} = x\boldsymbol{r}_c + \boldsymbol{r}_d, \boldsymbol{z}_g = x\boldsymbol{r}_b + \boldsymbol{r}_t$		
25:	$\boldsymbol{z}_{b,i} = x\boldsymbol{r}_{b,i} + \boldsymbol{r}_{t,i} \forall i \in [0, N)$		
26:	$\boldsymbol{z}_b = \sum_{i=0}^{N-1}\zeta^i \boldsymbol{z}_{b,i}$		
27:	$\mathrm{Rej}((\boldsymbol{z}, \boldsymbol{z}_g, (\boldsymbol{z}_{b,i})_{i=0}^{N-1}), x(\boldsymbol{r}_c, \boldsymbol{r}_b, (\boldsymbol{r}_{b,i})_{i=0}^{N-1}), \phi_3, T_3)$		
		$\xrightarrow{\quad \boldsymbol{f}_1, \boldsymbol{g}, \boldsymbol{z}, \boldsymbol{z}_g, \boldsymbol{z}_b\quad}$	
28:		$\boldsymbol{g} = (g_{0,0}, \cdots, g_{N-1,k-1})$	
29:		$\boldsymbol{g}_i = (g_{i,0}, \cdots, g_{i,k-1}), \forall i$	
30:		$\boldsymbol{h} = (g_{i,j}(x - g_{i,j}))_{i=0,j=0}^{N-1,k-1}$	
31:		$\boldsymbol{f}_1 = (f_1, \cdots, f_{k-1})$	
32:		$f_0 = d_{sum} - \sum_{j=1}^{k-1} 2^j \cdot f_j \in R$	
33:		$C = \sum_{i=0}^{N-1}\omega_i B_i$	
34:		$\Vert g_{i,j}\Vert \overset{?}{\leq} 2\phi_1 T_1\sqrt{d}, \quad \forall i, j$	
35:		$f_0 \overset{?}{\in} R_q$	
36:		$\Vert f_j\Vert \overset{?}{\leq} 2\phi_2 T_2\sqrt{d}, \quad \forall j$	
37:		$\Vert \boldsymbol{z}\Vert, \Vert \boldsymbol{z}_g\Vert \overset{?}{\leq} 2\phi_3 T_3\sqrt{md}$	
38:		$\Vert \boldsymbol{z}_b\Vert \overset{?}{\leq} 2md\phi_3 T_3 \sum_{i=0}^{N-1}(wp)^i$	
39:		$xC + D \overset{?}{=} \mathrm{Com}_{ck}((f_0, \cdots, f_{k-1}); \boldsymbol{z})$	
40:		$xE + F \overset{?}{=} \mathrm{Com}_{ck}(\boldsymbol{h}; \boldsymbol{z}_g)$	
41:		$x\sum_{i=0}^{N-1}\zeta^i B_i + G \overset{?}{=} \mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta^i \boldsymbol{g}_i; \boldsymbol{z}_b)$	

5.2 Unbalanced Linear Sum Proof

We formally describe our unbalanced linear sum proof protocol. Based on the description of Section 4.2, the prover needs to show 1) $(\delta_{l_j,0}, \cdots, \delta_{l_j,\beta-1})$'s are short non-zero vectors and are properly committed and 2) $\delta_{l,i}$'s can be constructed with $\delta_{l,i} = \prod_{j=0}^{k-1} \delta_{l_j,i_j}$ such that $\sum_{i=0}^{N-1} \delta_{l,i} P_i$ being a commitment to zero. The first relation is discussed in Section 4.2 which implies $\boldsymbol{\delta} = (\delta_{l,0}, \cdots, \delta_{l,N-1})$ being a short and non-zero vector. Here we briefly describe the second relation, which follows the same process of [14, 17, 20].

After receiving a challenge x, the prover's response contains $f_{j,i} = x\delta_{l_j,i} + a_{j,i}$ with some masking values $a_{j,i}$'s. To rebuild $\delta_{l,i}$'s, the verifier computes the product $p_i(x) = \prod_{j=0}^{k-1} f_{j,i_j}$:

$$\begin{aligned} p_i(x) &= \prod_{j=0}^{k-1} f_{j,i_j} = \prod_{j=0}^{k-1} (x\delta_{l_j,i_j} + a_{j,i_j}) \\ &= x^k \cdot \prod_{j=0}^{k-1} \delta_{l_j,i_j} + \sum_{j=0}^{k-1} p_{i,j} \cdot x^j = \delta_{l,i} x^k + \sum_{j=0}^{k-1} p_{i,j} x^j, \end{aligned} \tag{19}$$

where $p_{i,j}$'s are functions of δ_{l_j,i_j}'s (i.e., l) and $a_{j,i}$'s. Equation (19) holds for all $i \in [0, N)$. As $p_{i,j}$'s are independent of x, the prover can pre-compute $p_{i,j}$'s and send $E_j = \sum_{i=0}^{N-1} p_{i,j} P_i$ to allow the verifier to cancel out the coefficients of the terms $1, x, \cdots, x^{k-1}$ before receiving x (the randomness is omitted here for simplicity). For x^k part, it can be set to the prover's public key P_l with $\sum_{i=0}^{N-1} \delta_{l,i} P_i$. Our unbalanced linear sum proof is formally described in Protocol 4.

In Protocol 4, steps 4 to 7 generate the masking values $a_{j,i}$'s for $\delta_{l_j,i}$'s and ensure $\sum_{i=0}^{\beta-1} a_{j,i} = 0$ (which further ensures $\sum_{i=0}^{\beta-1} (x\delta_{l_j,i} + a_{j,i}) = x$). The $p_{i,j}$'s in steps 11 and 14 are derived based on Equation (19).

Upon receiving the challenge x, the prover generates the responses $\boldsymbol{f}_1$, $\boldsymbol{z}_b$, and $\boldsymbol{z}_r$. For $\boldsymbol{f}$, the prover can avoid sending $f_{j,0}$'s as $\sum_{i=0}^{\beta-1} f_{j,i} = x$ holds. In step 21, $\boldsymbol{z}_r$ is the response to randomness in P_l and E_j's based on Equation (19). As $\boldsymbol{\rho}_0$ is sampled from $D_{\phi_2 T_2}^{md}$ and other $\boldsymbol{\rho}_j$'s and $\boldsymbol{r}$ are sampled from $\{-\mathcal{B}, \cdots, \mathcal{B}\}^{md}$, $\boldsymbol{\rho}_0$ is the masking vector for $x^k \boldsymbol{r} - \sum_{j=1}^{k-1} x^j \boldsymbol{\rho}_j$ (step 21).

Finally, the verifier computes $f_{j,0} = x - \sum_{i=1}^{\beta-1} f_{j,i}$ for all $j \in [0, k)$ as $\sum_{i=0}^{\beta-1} f_{j,i} = x$, which ensures $\sum_{i=0}^{\beta-1} \delta_{l_j,i} = 1$ (and further ensures at least one element in $(\delta_{l_j,0}, \cdots, \delta_{l_j,\beta-1})$ is not 0 for all j's). The last two steps ensure that $\sum_{i=0}^{N-1} \delta_{l,i} P_i$ is a commitment to 0 as discussed in Section 4.2.

Theorem 2. *Let* $\gamma_{LS} = (4\phi_1\sqrt{k\beta})^k d^{k-\frac{1}{2}}$, $\gamma'_{LS} = (k+1)2^{\kappa'+2}\sqrt{2}\phi_2 \mathcal{B} m d^2 w^\kappa p^{\kappa+1}$ *for* $\kappa = k(k+1)/2$ *and* $\kappa' = k(k-1)/2$, *the HMC is hiding and* γ_{LS}*-binding. Protocol 4 has* $(k+1)$*-special soundness for relations* $\mathcal{R}_{LS}(\mathcal{B}\sqrt{md})$ *and* $\mathcal{R}'_{LS}(\gamma_{LS}, \gamma'_{LS})$ *with a completeness error* $1 - 1/(\mu(\phi_1)\mu(\phi_2))$ *defined in Lemma 8.*

The proof for Theorem 2 is given in Appendix C.

Protocol 4 Unbalanced Linear Sum Proof

$\mathcal{P}_{LS}(ck, \boldsymbol{P}, (l, \boldsymbol{r}))$		$\mathcal{V}_{LS}(ck, \boldsymbol{P})$
$T_1 = p\sqrt{kw},\ T_2 = (wp)^k \mathcal{B}\sqrt{2md}$		
$\boldsymbol{r}_a \leftarrow D^{md}_{\phi_2 T_2}$		
$\boldsymbol{r}_b \leftarrow \{-\mathcal{B}, \cdots, \mathcal{B}\}^{md}$		
for $j = 0$ to $k-1$ **do**		
$\quad a_{j,1}, \cdots, a_{j,\beta-1} \leftarrow D^{d}_{\phi_1 T_1}$		
$\quad a_{j,0} = -\sum_{i=1}^{\beta-1} a_{j,i}$		
end for		
$\boldsymbol{\delta} = (\delta_{l_0,0}, \cdots, \delta_{l_{k-1},\beta-1})$		
$\boldsymbol{a} = (a_{0,0}, \cdots, a_{k-1,\beta-1})$		
$\boldsymbol{\rho}_0 \leftarrow D^{md}_{\phi_2 T_2}$		
$E_0 = \sum_{i=0}^{N-1} p_{i,0} P_i + \mathrm{Com}_{ck}(\boldsymbol{0}; \boldsymbol{\rho}_0)$		
for $j = 1$ to $k-1$ **do**		
$\quad \boldsymbol{\rho}_j \leftarrow \{-\mathcal{B}, \cdots, \mathcal{B}\}^{md}$		
$\quad E_j = \sum_{i=0}^{N-1} p_{i,j} P_i + \mathrm{Com}_{ck}(\boldsymbol{0}; \boldsymbol{\rho}_j)$		
end for		
$B = \mathrm{Com}_{ck}(\boldsymbol{\delta}; \boldsymbol{r}_b),\ A = \mathrm{Com}_{ck}(\boldsymbol{a}; \boldsymbol{r}_a)$		
	$\xrightarrow{A, B, (E_j)_{j=0}^{k-1}}$	
	$\xleftarrow{x}$	$x \leftarrow \mathcal{C}$
$\boldsymbol{\delta}_1 = (\delta_{l_0,1}, \cdots, \delta_{l_{k-1},\beta-1})$		
$\boldsymbol{a}_1 = (a_{0,1}, \cdots, a_{k-1,\beta-1})$		
$\boldsymbol{f}_1 = x\boldsymbol{\delta}_1 + \boldsymbol{a}_1$		
$\mathrm{Rej}(\boldsymbol{f}_1, x\boldsymbol{\delta}_1, \phi_1, T_1)$		
$\boldsymbol{z}_b = x\boldsymbol{r}_b + \boldsymbol{r}_a,\ \boldsymbol{z}_r = x^k \boldsymbol{r} - \sum_{j=0}^{k-1} x^j \boldsymbol{\rho}_j$		
$\mathrm{Rej}((\boldsymbol{z}_b, \boldsymbol{z}_r), (x\boldsymbol{r}_b, x^k\boldsymbol{r} - \sum_{j=1}^{k-1} x^j \boldsymbol{\rho}_j), \phi_2, T_2)$		
	$\xrightarrow{\boldsymbol{f}_1, \boldsymbol{z}_b, \boldsymbol{z}_r}$	
		$f_{j,0} = x - \sum_{i=1}^{\beta-1} f_{j,i}, \forall j \in [0, k)$
		$\|f_{j,i}\| \overset{?}{\leq} 2\phi_1 T_1 \sqrt{d}, \quad \forall j, \forall i \neq 0$
		$\|f_{j,0}\| \overset{?}{\leq} 2\phi_1 T_1 \sqrt{\beta d}, \forall j \in [0, k)$
		$\|\boldsymbol{z}_b\|, \|\boldsymbol{z}_r\| \overset{?}{\leq} 2\phi_2 T_2 \sqrt{md}$
		$\boldsymbol{f} = (f_{0,0}, \cdots, f_{k-1,\beta-1})$
		$xB + A \overset{?}{=} \mathrm{Com}_{ck}(\boldsymbol{f}; \boldsymbol{z}_b)$
		$\sum_{i=0}^{N-1} (\prod_{j=0}^{k-1} f_{j,i_j}) P_i - \sum_{j=0}^{k-1} E_j x^j \overset{?}{=} \mathrm{Com}_{ck}(\boldsymbol{0}; \boldsymbol{z}_r)$

6 Efficient ZKPs for RingCT

As applications of our techniques, we show how to build balance proofs and ring signatures for RingCT protocols.

6.1 Non-Interactive Balance Proof

Based on Protocol 3, we design an efficient non-interactive balance proof for RingCT protocols. Consider the case in Section 3.1 with M input accounts $(a_i)_{i=0}^{M-1}$ and S output accounts $(b_i)_{i=0}^{S-1}$. The balance proof is a special case of linear equation satisfiability, where $N = S + M$, $(\omega_0, \cdots, \omega_{S-1}) = (1, \cdots, 1)$, and $(\omega_S, \cdots, \omega_{S+M-1}) = (-1, \cdots, -1)$. Accordingly, Equation (13) can be expressed as $F(a_0, \cdots, a_{M-1}, b_0, \cdots, b_{S-1}) = \sum_{i=0}^{S-1} b_i - \sum_{i=0}^{M-1} a_i$.

Let $\mathtt{CNK}_{in} = (\boldsymbol{r}_{a,i})_{i=0}^{M-1}$ and $\mathtt{CNK}_{out} = (\boldsymbol{r}_{b,i})_{i=0}^{S-1}$ be the sets of input and output coin keys respectively (i.e. randomness), $\mathtt{CN}_{in} = (A_i)_{i=0}^{M-1}$ and $\mathtt{CN}_{out} = (B_i)_{i=0}^{M-1}$ be the sets of input and output coins (commitments to $\boldsymbol{a}_i$'s and $\boldsymbol{b}_i$'s, i.e., $A_i = \mathrm{Com}_{ck}(\boldsymbol{a}_i; \boldsymbol{r}_{a,i})$ and $B_i = \mathrm{Com}_{ck}(\boldsymbol{b}_i; \boldsymbol{r}_{b,i})$). Denote the initial commitments in Protocol 3 as $\mathsf{CMT} = (D, E, F, (G_i)_{i=0}^{S-1})$, the prover's response as $\mathsf{RSP} = (\boldsymbol{f}_1, \boldsymbol{g}, \boldsymbol{z}, \boldsymbol{z}_g, (\boldsymbol{z}_{b,i})_{i=0}^{S-1})$, and $\mathsf{CMT}^* = E$. We omit descriptions of the full set of algorithms and detail the key ones here.

• **Setup**(1^λ): Run $\boldsymbol{G} \leftarrow \mathsf{CKeygen}$ and set $ck = \boldsymbol{G}$. Choose a hash function $H : \{0,1\}^* \to \mathcal{C}$. Return $pp = (ck, H)$.

• **Mint**(pp, v): Sample $\boldsymbol{r} \leftarrow \{-\mathcal{B}, \cdots, \mathcal{B}\}^{md}$ and compute $(v_0, \cdots, v_{k-1}) \leftarrow \mathrm{Bits}(v)$, $B = \mathrm{Com}_{ck}(\boldsymbol{v}; \boldsymbol{r})$. Return $(\mathtt{cn}, \mathtt{cnk}) = (B, \boldsymbol{r})$.

• **Spend**$((a_i)_{i=0}^{M-1}, (b_i)_{i=0}^{S-1}, pp, \mathtt{CN}_{in}, \mathtt{CNK}_{in})$: Parse $\mathtt{CN}_{in} = (A_i)_{i=0}^{M-1}$ and $\mathtt{CNK}_{in} = (\boldsymbol{r}_{a,i})_{i=0}^{M-1}$. Set $\boldsymbol{a}_i \leftarrow \mathrm{Bits}(a_i)$ for $i \in [0, M)$ and $\boldsymbol{b}_i \leftarrow \mathrm{Bits}(b_i)$ for $i \in [0, S)$. Call **Mint**$(pp, b_i) = (\mathtt{cn}_i, \mathtt{cnk}_i) = (B_i, \boldsymbol{r}_{b,i})$ for $i \in [0, S)$ to mint coins for output accounts. Set $\mathtt{CN}_{out} = (\mathtt{cn}_i)_{i=0}^{S-1}$ and $\mathtt{CNK}_{out} = (\mathtt{cnk}_i)_{i=0}^{S-1}$. Proceed as follows:

1. Run $\mathcal{P}_{LE}(ck, ((1, B_i)_{i=0}^{S-1}, (-1, A_i)_{i=0}^{M-1}), ((\boldsymbol{b}_i, \boldsymbol{r}_{b,i})_{i=0}^{S-1}, (\boldsymbol{a}_i, \boldsymbol{r}_{a,i})_{i=0}^{S-1}))$ to generate CMT based on the first 18 steps[8] of Protocol 3.
2. Compute $x = H(ck, (A_i)_{i=0}^{M-1}, (B_i)_{i=0}^{S-1}, \mathsf{CMT})$.
3. Compute RSP by running the remaining steps of $\mathcal{P}_{LE}$.
4. Return $\mathtt{CN}_{out}$ and $\pi = (\mathsf{CMT}^*, x, \mathsf{RSP})$.

• **Verify**$(\mathtt{CN}_{in}, \mathtt{CN}_{out}, \pi, pp)$: Parse $\mathtt{CN}_{in} = (A_i)_{i=0}^{M-1}$, $\mathtt{CN}_{out} = (B_i)_{i=0}^{S-1}$, $\pi = (\mathsf{CMT}^*, x, \mathsf{RSP})$. Proceed as follows:

1. Compute C, D, F, and G based on step 33, 39, 40, and 41 of Protocol 3 and set $\mathsf{CMT} = (D, E, F, G)$.
2. Return 0 if $x \neq H(ck, (A_i)_{i=0}^{M-1}, (B_i)_{i=0}^{S-1}, \mathsf{CMT})$.
3. Return the output of $\mathcal{V}_{LE}(ck, ((1, B_i)_{i=0}^{S-1}, (-1, A_i)_{i=0}^{M-1}))$ with $(\mathsf{CMT}, x, \mathsf{RSP})$.

Notice that this non-interactive balance proof does not ensure anonymity. It can be extended to an anonymous RingCT protocol with the linkable ring signature scheme (described in Section 8).

[8] In existing anonymous cryptocurrency implementations, binary proofs for inputs $\boldsymbol{a}_i$'s can be reduced as they have been verified as output accounts in previous transactions.

Theorem 3. *Let* $\gamma_{LE} = 2^{\kappa+1}N\mathcal{B}p^{\kappa}w^{\kappa-1}md^2\phi_3(S+M+1)\sum_{i=0}^{N-1}(wp)^i$*, and the HMC is hiding and* γ_{LE}*-binding. The balance proof has* $(3, N+1)$*-special soundness for relations* $\mathcal{R}_{LE}(\mathcal{B}\sqrt{md})$ *and* $\mathcal{R}'_{LE}(\gamma_{LE})$ *with a completeness error* $1 - 1/(\mu(\phi_1)\mu(\phi_2)\mu(\phi_3))$ *defined in Lemma 8.*

Proof. Considering $N = S+M$, $(\omega_0, \cdots, \omega_{S-1}) = (1, \cdots, 1)$, and $(\omega_S, \cdots, \omega_{S+M-1}) = (-1, \cdots, -1)$, we have $\|\boldsymbol{\omega}\|_1 = S + M \geq 1$. Equation (24) can be further simplified as:

$$\mathcal{B}wp\sqrt{md(\|\boldsymbol{\omega}\|_1^2 + N + 1)} \leq \mathcal{B}wp(S+M+1)\sqrt{md} = T_3. \tag{20}$$

Thus, we have $\gamma_{LE} = 2^{\kappa+1}N\mathcal{B}p^{\kappa}w^{\kappa-1}md^2\phi_3(S+M+1)\sum_{i=0}^{N-1}(wp)^i$.

Other parts can be derived directly from Theorem 1.

6.2 Ring Signature

Protocol 4 can be used to construct an efficient ring signature scheme. Let M be the message to be signed, the initial commitment in Protocol 4 be $\mathsf{CMT} = (A, B, (E_j)_{j=0}^{k-1})$, and the prover's response be $\mathsf{RSP} = (\boldsymbol{f}_1, \boldsymbol{z}_b, \boldsymbol{z}_r)$. Denote $\mathsf{CMT}^* = (B, (E_j)_{j=1}^{k-1})$. The ring signature is defined as follows:

• **RSetup**(1^λ): Run $\boldsymbol{G} \leftarrow \mathsf{CKeygen}$ and set $ck = \boldsymbol{G}$. Choose a hash function $H : \{0,1\}^* \rightarrow \mathcal{C}$. Return $pp = (ck, H)$.

• **RKeygen**(pp): Sample $\boldsymbol{r} \leftarrow \{-\mathcal{B}, \cdots, \mathcal{B}\}^{md}$ and compute $P = \mathrm{Com}_{ck}(\boldsymbol{0}; \boldsymbol{r})$. Return $(pk, sk) = (P, \boldsymbol{r})$.

• **RSign**$(M, \boldsymbol{P}, pp, sk)$: Parse $\boldsymbol{P} = (P_0, \cdots, P_{N-1})$ and $P_l = \mathrm{Com}_{ck}(\boldsymbol{0}; sk)$ for $l \in [0, N)$. Proceed as follows:

1. Generate CMT by running $\mathcal{P}_{LS}(ck, \boldsymbol{P}, (l, \boldsymbol{r}))$, step 1 to 16 in Protocol 4.
2. Compute $x = H(ck, M, \boldsymbol{P}, \mathsf{CMT})$.
3. Compute RSP by running the remaining steps of $\mathcal{P}_{LS}$.
4. Return $\pi = (\mathsf{CMT}^*, x, \mathsf{RSP})$.

• **RVerify**$(M, \boldsymbol{P}, \pi, pp)$: Parse $\boldsymbol{P} = (P_0, \cdots, P_{N-1})$, $\pi = (\mathsf{CMT}^*, x, \mathsf{RSP})$, and $\mathsf{CMT}^* = (B, (E_j)_{j=1}^{k-1})$. Proceed as follows:

1. Compute A and E_0 based on step 28 and 29 of Protocol 4 and set $\mathsf{CMT} = (A, B, (E_j)_{j=0}^{k-1})$.
2. Return 0 if $x \neq H(ck, M, \boldsymbol{P}, \mathsf{CMT})$.
3. Return the output of $\mathcal{V}_{LS}(ck, \boldsymbol{P})$ with $(\mathsf{CMT}, x, \mathsf{RSP})$.

The correctness and anonymity of the ring signature can be derived directly from the completeness and SHVZK of Protocol 4. The unforgeability of the ring signature is formally described as follows:

Theorem 4. *If the commitment scheme is computational hiding and* γ*-binding, then the ring signature scheme described above is unforgeable with respect to insider corruption in the random oracle model.*

Proof. Assume there exists a PPT adversary $\mathcal{F}$ that can efficiently forge a ring signature with non-negligible probability, we have an adversary $\mathcal{A}$ which can break the binding property of the commitment scheme, and solve the M-SIS problem accordingly.

$\mathcal{A}$ samples $\boldsymbol{r} \leftarrow \{-\mathcal{B}, \cdots, \mathcal{B}\}$ and computes an invalid public key $pk_l = \mathrm{Com}_{ck}(1, 0, \cdots, 0; \boldsymbol{r})$. Due to the hiding probability of the commitment scheme, $\mathcal{F}$ cannot distinguish pk_l with other public keys. Then, $\mathcal{A}$ runs $\mathcal{F}$ for $(k+1)$ times to get $(k+1)$ forgeries with distinct challenges and a same CMT^* part based on the forking lemma (pk_l is not corrupted). Furthermore, $\mathcal{A}$ reconstructs CMT and runs the extractor of Protocol 4 with the $(k+1)$ forgeries to get valid $b'_i = y^k b_i$ for $i \in [0, N)$ and a valid opening $(\mathbf{0}, \boldsymbol{s})$ of $y \sum_{i=0}^{N-1} b_i \cdot pk_i$ for some public keys. Thus, we have proper b'_i's and a valid opening $(\mathbf{0}, y^{k-1}\boldsymbol{s})$ of $\sum_{i=0}^{N-1} b'_i \cdot pk_i$. Based on Lemma 5, we have a collision for the commitment scheme, $((b'_l, 0, \cdots, 0), b'_l \boldsymbol{r})$ and $(\mathbf{0}, y^{k-1}\boldsymbol{s} - \sum_{i \neq l} b'_i \boldsymbol{r}_i)$ as $(b'_l, 0, \cdots, 0) \neq \mathbf{0}$ ($\boldsymbol{r}_i$'s are the private keys as the output of Corrupt(i) in the proof of Lemma 5). More details about the security reduction is presented in Appendix D.

7 Evaluation

Implementation. To evaluate the performance of the proposed proofs, we give a reference implementation of both MatRiCT [17] and our approaches in Golang [2]. The underlaying polynomial ring operations are implemented with LaGo [23][9]. For the linear equation satisfiability, we only implement the balance proof version (i.e., ω_i's are fixed in our code) to compare with the balance proof in MatRiCT. The code of MatRiCT and our work is published in [2]. All experiments are performed on a personal laptop equipped with Intel i7-8750H 2.20GHz CPU and 8GB memory.

Proof size: balance proof. We first evaluate the performance of our balance proof. Referring to [17], we consider the scenario that requires 64-bit precision for amounts (i.e., $k = 64$) and set the parameters as: $\mathcal{B} = 1$, $(d, w, p) = (64, 56, 8)$, $q = (2^{27} - 2^{21} + 1) \cdot (2^{26} - 2^{12} + 1) \approx 2^{53}$, $(n, m) = (32, 65)$, and $\phi_1 = \phi_2 = \phi_3 = 15$. These parameters are chosen based on a "root Hermite factor" of $\delta \approx 1.0045$ for both M-LWE and M-SIS, and ensure 128-bit security based on the "LWE estimator" [1]. More details are discussed in [14, 17].

As the Algorithm 8 and 9 in [17] only deal with $M = 1$ and $S \leq 2$, we need to include an additional range proof in MatRiCT for other cases. However, MatRiCT does no specified range proof approaches in [17]. According to the discussion in Section 3.1, we use the state-of-the-art range proof in [14] to evaluate the performance of MatRiCT (the number of slot in CRT packing is set to 16 as with [14]). For $M = 1$ and $S \leq 2$ cases, we follow algorithms 8 and 9 in [17] as the implementation of MatRiCT.

First, we show the balance proof size growth with the number of input accounts in Figure 1. Different from the result in [17], our result shows the proof

[9] We also found a bug in MulPoly function of LaGo. Please refer to our repository [2] for more details.

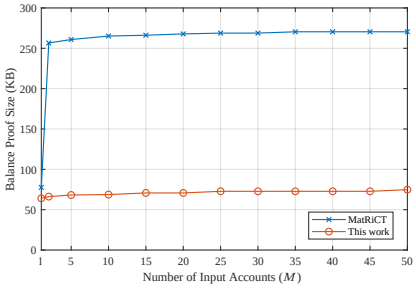

Fig. 1: Balance proof size growth with the number of input accounts (M) when $S = 1$.

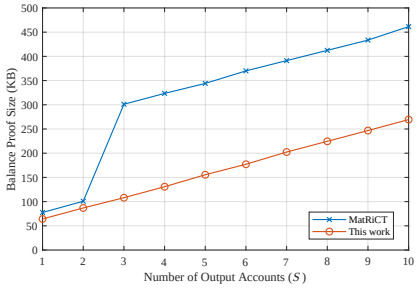

Fig. 2: Balance proof size growth with the number of output accounts (S) when $M = 1$.

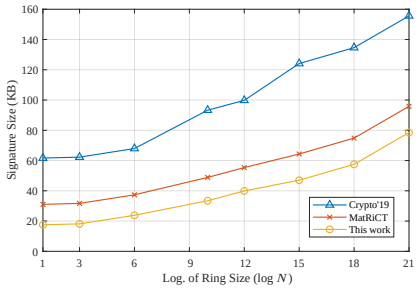

Fig. 3: Ring signature size. The size of MatRiCT does not include the cost of serial numbers for fairness.

size is relatively small and does not scale linearly with M. It is because [17] takes NM input accounts for anonymity (N is the set size as in ring signatures). Thus, other $(N-1)M$ hiding accounts contribute to a great part of the proof size, which scales linearly with the size in [17]. While in our experiment, we do not consider anonymity in balance proofs. Furthermore, there is a clear burst in MatRiCT when $M = 2$. It is due to the additional range proofs for corrector values. The burst also indicates the cost of range proofs is prohibitively large for real-world implementations. Besides, the proof size does not increase much with M except for $M = 1$. For some M, e.g. $M \in [30, 40]$, the size remains the same. This is an expected result as M contributes to the size of each element in $\boldsymbol{g}$, $\boldsymbol{z}_b$, and $\boldsymbol{z}_r$, instead of the length of these vectors. Finally, our balance proof can reduce about 15% size of MatriCT when $M = 1$ and more than 70% in other cases.

Second, we present the balance proof size growth with the number of output accounts in Figure 2. As discussed above, the burst of MatRiCT is caused by range proofs. When $S > 2$, the proof size scales linearly with S due to the cost of $(\boldsymbol{z}_{b,i})_{i=0}^{S-1}$. As S also contributes to the size of vector elements, the growth is not exactly linear. Generally speaking, our approach can reduce 15% proof size of MatRiCT when $S < 2$ and up to 60% for other cases.

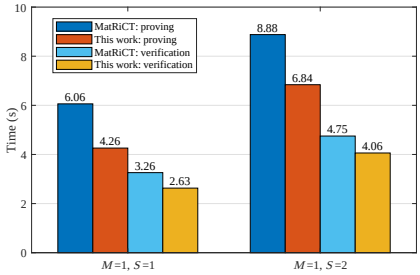

Fig. 4: Time cost of balance proofs.

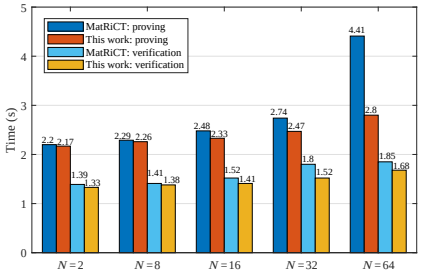

Fig. 5: Time cost of ring signatures.

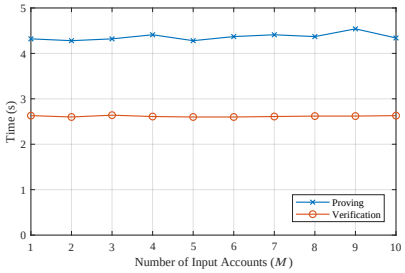

Fig. 6: Time cost of balance proofs with various inputs M when $S = 1$.

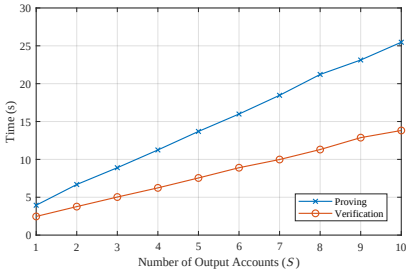

Fig. 7: Time cost of balance proofs with various outputs S when $M = 1$.

Proof size: ring signature. We further evaluate the performance of our ring signature, and compare with Crypto'19 [14] and MatRiCT [17]. We use the same settings in [17] with two sets of parameters: $(\hat{n}, \hat{m}) = (32, 65)$, $\hat{q} = (2^{27}-2^{21}+1)\cdot(2^{26}-2^{12}+1) \approx 2^{53}$ for the binary proof part; and $(n, m) = (18, 38)$, $q = 2^{31} - 2^{18} + 2^3 + 1 \approx 2^{31}$ for other parts. Other settings for the two parts are same (i.e., $(d, w, p) = (64, 56, 8)$). Please note that as our unbalanced linear sum proof avoids the binary proofs, only the smaller parameters, n, m, and q, affect the performance.

The signature size growth with the logarithmic ring size $\log(N)$ is depicted in Figure 3. The result of Crypto'19 is not as "smooth" as that in [14]. It is caused by the parameters we used are different from short infinity norm challenges in [14]. Besides, our result also shows that MatRiCT does not improve much of Crypto'19, which is different from the results in [17]. This is mainly caused by the parameter settings, as d and q in [14] are much larger than those in [17]. Since MatRiCT uses the same blueprint in Crypto'19, Crypto'19 is also improved under MatRiCT settings. Moreover, in our ring signature approach, as we avoid the cost of a binary proof, the sizes of a commitment and $\boldsymbol{z}_b, \boldsymbol{z}_r$ elements are much smaller. A further observation is that all approaches do not scale logarithmically in N. This is due to the growth of element size. Comparing with existing state-of-the-art approaches, our ring signature is the most efficient which can reduce about 50% and 20% of the signature size in Crypto'19 and MatRiCT respectively.

Proving/verification time. Finally, we compare the proving and verification time of our approaches with MatRiCT [17]. As we explained earlier, the balance proof in MatRiCT only works when $M = 1$ and $S \leq 2$, we only compare the performance in two cases, $(M, S) = (1, 1)$ and $(M, S) = (1, 2)$. The results are depicted in Figure 4. Our inner-product based approach reduces nearly 30% proving time of the MatRiCT, as we do not involve c_i's in binary proofs. Besides, since the commitment of c_i's is derived from A_i's and B_i's, our approach also reduce the time of committing to corrector values. Furthermore, our approach reduces nearly 20% verification time of the MatRiCT. The main reason is verifying the inner-product relation (step 35 of Protocol 3) is much more efficient than the balance relation with corrector values. The efficiency of binary verification is also improved when removing the corrector values.

The performances of ring signatures are depicted in Figure 5. Our unbalanced linear sum approach can reduce nearly 35% time of the one-out-of-many approach when $N = 64$. This is mainly contributed by avoiding the binary proof parts in our approach. When N is small, the binary proof/verification cost is only a small portion of the whole cost and thus the improvement is less significant. Besides, one interesting result is that the verification time does not increase much when $N = 64$. It is because we set $\beta = 4$ and $k = 3$ in this case, while in other cases $\beta = N$ and $k = 1$. Though the verification time is greatly reduced under these settings, the proof size increases accordingly. Thus, for other cases, we keep $\beta = N$ and $k = 1$ as in [14]. Nevertheless, our approaches outperform MatRiCT in all settings.

Additionally, we show the proving and verification time of a balance proof for various inputs (M) and outputs (S) in Figure 6 and Figure 7 respectively. As a prover does not need to verify input accounts (input accounts have been verifier in previous transactions as output accounts), there is almost no change in proving/verification time (similar as the result of proof size in Figure 1). Furthermore, the proving and verification time scales linearly with S since the prover needs to generate proofs for output accounts and the verifier needs to check these proofs accordingly. As stated in [14], "the most common cases for the number of input/output accounts are $(M, S) = (1, 2)$ and $(M, S) = (2, 2)$", the time cost of our approaches are acceptable in most scenarios.

8 RingCT Protocol

Though the unbalanced linear sum proof is secure for ring signatures, it cannot be applied in RingCT directly. Besides, we also need to avoid double-spending. We show how these issues are addressed to apply our techniques in RingCT.

Combining two proofs. In a RingCT protocol, a spender needs to prove a transaction is valid and hides the identity of input accounts simultaneously. This can be achieved by adding hiding accounts into inputs and proving the balance and linear sum relations in one proof. Consider NM inputs $(\mathtt{CN}_{in}^{(j)})_{j=0}^{N-1} = (\mathsf{IN}_i^{(j)})_{i=0,j=0}^{M-1,N-1}$, which have M spender's accounts (the spender owns the amount values and coin keys at index $j = l$, $(\mathsf{in}_i^{(l)}, \mathsf{ink}_i^{(l)})_{i=0}^{M-1}$, such that $\mathsf{IN}_i^{(l)} = \mathrm{Com}(\mathsf{in}_i^{(l)}, \mathsf{ink}_i^{(l)})$),

and $(N-1)M$ hiding accounts, $\mathtt{CN}_{in}^{(j)}$ where $j \neq l$. To transfer funds to S output accounts $(\mathsf{OUT}_i = \mathrm{Com}(\mathsf{out}_i, \mathsf{outk}_i))_{i=0}^{S-1}$, the spender needs to send an additional commitment $\mathrm{Com}(\boldsymbol{c}; \boldsymbol{r}'_c)$ and compute public keys $P_j = \sum_{i=0}^{S-1} \mathsf{OUT}_i - \sum_{i=0}^{M-1} \mathsf{IN}_i^{(j)} - \mathrm{Com}(\boldsymbol{c}; \boldsymbol{r}'_c)$ for $j \in [0, N)$.

Ideally, we regard P_l as a commitment to zero with the private key (randomness) $\boldsymbol{r} = \sum_{i=0}^{S-1} \mathsf{outk}_i - \sum_{i=0}^{M-1} \mathsf{ink}_i^{(l)} - \boldsymbol{r}'_c$. The spender can further show P_l is a commitment to zero as in our ring signature scheme, which proves the amount balance and hides the identity at the same time. Unfortunatelly, as linear sum proof only ensures $\sum_{i=0}^{N-1} b_i P_i$ is a commitment to zero, the above approach will incur an *unbalancing problem*. For instance, let $M = S = 1$. If the spender owns *two* input accounts at indices s and t with $\mathsf{in}^{(s)} = 2$ and $\mathsf{in}^{(t)} = 1$, she can mint $\mathsf{out} = 4$ coins (more than the sum of all inputs) by setting $b_s = 3$ and $b_t = -2$. As P_s is a commitment to $\mathsf{out} - \mathsf{in}^{(s)}$ (i.e., 2) and P_t is a commitment to $\mathsf{out} - \mathsf{in}^{(t)}$ (i.e., 3), $b_s P_s + b_t P_t$ is a commitment to 0 (here we use the amounts directly instead of their bits to simplify the example). This is due to the security proof of our ring signatures relies on P_i's being correctly generated (i.e., commitments to 0), which may not be true as P_i's are derived from different accounts. We latter describe our solution.

Avoid double-spending. To avoid double-spending, we extend our ring signature (Protocol 4) to provide linkability by checking the serial number of each input account to ensure it is not included in previous transactions. This could be done by following the blueprint of MatRiCT [17]. Consider a new commitment key $\boldsymbol{H}$. A serial number SN is a public commitment to zero under $\boldsymbol{H}$ with $\boldsymbol{r}$ as the randomness, i.e., $\mathsf{SN} = \boldsymbol{H} \cdot \boldsymbol{r}$. At step 12 of Protocol 4, the prover also needs to compute $F_j = \boldsymbol{H} \cdot \boldsymbol{\rho}_j$ for all $j \in [0, k)$. In the verification, the verifier can 1) link the proof with previous ones based on SN and 2) check SN is correct with $x^k \cdot \mathsf{SN} - \sum_{j=0}^{k-1} x^j F_j = \boldsymbol{H} \cdot \boldsymbol{z}_b$.

In a RingCT protocol, each account has an additional account key pair, $(\mathsf{pk}, \mathsf{sk})$ such that $\mathsf{pk} = \mathrm{Com}_{ck}(\boldsymbol{0}, \mathsf{sk})$. For each input account, $i \in [0, S)$, the spender places it at index l of an N-size ring $\boldsymbol{PK}_i$ and runs the above linkable version of Protocol 4, $\mathcal{P}_{LS}(ck, \boldsymbol{PK}_i, \mathsf{SN}_i, (l, \mathsf{sk}_i))$. As all accounts share the same index l, $\boldsymbol{f}_1$ and $\boldsymbol{z}_b$ in Protocol 4 will be the same in different proofs (no need to retransmit). To avoid double-spending, the verifier checks the serial numbers are distinct and not included in previous transactions.

The security proof of serial numbers is the same as MatRiCT (Lemma 5.7 in [17]). Besides, the unbalancing problem does not exist here since assuming pk_i's are properly generated is acceptable as with [17].

Avoiding unbalancing problem. To address the unbalancing problem described above, we show a simple and efficient approach to ensure the spender can only use *one* P_i to run the linear sum proof. Recall the linkable version of our unbalanced linear sum proof. The serial number check ensures a spender cannot 1) avoid sending any serial number of her real account and 2) include the serial numbers of other's accounts. Thus, the serial number set of a valid transaction must be the serial numbers of all real input accounts. Accordingly, the number of serial numbers should be $HW(\boldsymbol{b}) \cdot S$, which shows how many accounts are

used as real inputs in our unbalanced linear sum proof. Therefore, to ensure *one* P_i out of an N-size public list, the verifier checks the number of serial numbers being S.

9 Discussion

Compatible with other techniques. As we improve the underlying ZKPs of a RingCT protocol, our approaches preserve all distinguishing features of MatRiCT, such as being compatible with efficient rejection sampling and extractable commitment techniques. The former one can be adopted in our unbalanced linear sum proof to improve the acceptance probability of rejection sampling. For a secret bit $b \in \{0, 1\}$ and a challenge $x \in [-p, p]$, the prover can sample the masking value from a uniform distribution, $a \leftarrow [-\mathcal{B}_a, \mathcal{B}_a]$ when $b = 1$, or $a \leftarrow [-(\mathcal{B}_a - p), \mathcal{B}_a - p]$ when $b = 0$. As a will never be rejected when $b = 0$, this approach improves the efficiency and avoids leaking side-channel information when dealing binary secrets with a fixed Hamming weight. The latter technique allows one to design an auditable RingCT protocol by placing a "mini trapdoor" in HMC. Setting $\boldsymbol{G}_r = \left[\boldsymbol{A}, \boldsymbol{t}^\top\right]^\top$ as an LWE matrix where $\boldsymbol{t} = \boldsymbol{A}^\top \boldsymbol{s} + \boldsymbol{e}$ for some secret $\boldsymbol{s}$ (i.e., secret key for auditing) and error $\boldsymbol{e}$, an extractor can extract a message from a commitment $C = \boldsymbol{G}_r \boldsymbol{r} + \boldsymbol{G}_m \boldsymbol{m}$ by computing $\langle (\boldsymbol{s}, -1), C \rangle = -\langle \boldsymbol{e}, \boldsymbol{r} \rangle + \langle (\boldsymbol{s}, -1)^\top \boldsymbol{G}_m, \boldsymbol{m} \rangle$. Based on the fact that the norm of $\langle (\boldsymbol{s}, -1), C \rangle - \langle (\boldsymbol{s}, -1)^\top \boldsymbol{G}_m, \boldsymbol{m} \rangle$ is small, the extractor enumerates all possible values to recover $\boldsymbol{m}$.

Besides MatRiCT, other techniques in MatRiCT+ [16] to optimize the underlying cyclotomic rings can also be applied in our approaches. Specifically, a new CRT-packing technique is proposed in power-of-2 cyclotomic rings to reduce the modulus with binary CRT slots (and reduce the commitment size accordingly). Furthermore, MatRiCT+ optimizes challenges in cyclotomic rings to reduce their Hamming weights [16]. As both techniques are *"general and of independent interest for lattice-based proof systems"* [16], our approaches can regard them as optimized settings to further improve efficiency. Besides, since corrector values are avoided in our solution, balance proofs can directly use these settings without mapping under Galois automorphisms for corrector values [16].

Applications in discrete logarithm settings. Since our techniques do not rely on lattice settings, the results are believed to be of independent interest for RingCT protocols in a generic setting. The unbalanced linear sum proof can be applied in discrete logarithm settings directly to improve the performance of ring signatures by removing the binary proof part. Note that under the discrete logarithm assumption, b_i's do not necessarily have to be short as the constraint of "short b_i's" is only used to ensure the hiding and binding properties of HMC. Thus, steps 24 and 25 in Protocol 4 can be avoided accordingly (in fact, all norm checks can be avoided). However, the improvement of our unbalanced linear sum proof may be less significant as the binary proof does not require a larger parameter set under discrete logarithm assumptions. For the linear equation satisfiability, it is compatible with bit-based commitments with Equation

(14) (commit to the bits of the secret instead of its value directly). Note that bit-based commitments bring some advantages in existing RingCT protocols: a binary proof implies a range proof relation directly. With our linear equation satisfiability, we can build the balance and range relations in a different way. Furthermore, the linear equation satisfiability has a wider application in anonymous DeFi applications (decentralized finance smart contracts on Ethereum). By settings ω_i's as the exchange rates of different pools, we can enable confidential multi-pool transactions (inputs and outputs are from distinct pools) for today's anonymous DeFi such as Zether [10].

General-purpose lattice-based proof systems. Since we do not exploit any special property of the commitment scheme other than the standard hiding and binding properties, other approaches with intriguing properties in general-purpose lattice-based proof systems [4, 9] may be applied in our scenarios. In standard SIS commitment schemes, the witness is a $(v \times l)$-size matrix, $S \in \mathbb{Z}_q^{v \times l}$. With a $(r \times v)$-size matrix $A \in \mathbb{Z}_q^{r \times v}$, the commitment works as $\mathrm{Com}(S) = A \cdot S = T \in \mathbb{Z}_q^{r \times l}$. Based on an $(l \times n)$-size challenge $C \in \{0,1\}^{l \times n}$, the prover encodes S as $Z = S \cdot C + Y$ with $Y \in D_\sigma^{v \times n}$.

First, an $O(\sqrt{N})$-size commitment scheme is proposed in [4] by encoding N-many secrets into S where $v = l = O(\sqrt{N})$. Unfortunately, when adopting this approach, the $\langle \boldsymbol{f}, \boldsymbol{2}^k \rangle$ in Equation (16) cannot be calculated directly as Z will "batch" some f_i's when computing $S \cdot C$. For instance, consider the first element in Z, $z_{0,0} = \sum_{i=0}^{l-1} s_{0,i} \cdot c_{i,0} + y_{0,0}$. As $f_0 = s_{0,0} \cdot c_{0,0} + y_{0,0}$, we have $2^0 \cdot z_{0,0} = 2^0 \cdot f_0 + 2^0 \cdot (\sum_{i=1}^{l-1} s_{0,i} \cdot c_{i,0}) = 2^0 \cdot f_0 + 2^0 \cdot e_0$. Therefore, it is important to allow the verifier to cancel out $\sum_{i=0}^{k-1} 2^i \cdot e_i$ without leaking any information when computing $\langle \boldsymbol{f}, \boldsymbol{2}^k \rangle$. The same issue occurs in ring signatures when computing $\sum_{i=0}^{\beta-1} f_{j,i}$ in Equation (18) and $\prod_{j=0}^{k-1} f_{l_j,i_j}$ in Equation (19). The latter one is a much thornier problem when using $z_{i,j}$'s to compute $\prod_{j=0}^{k-1} f_{j,i_j}$. Second, levelled commitments and Bulletproofs folding are proposed in [9]. The proof size can be reduced to $O(N^{\frac{1}{d+1}})$ with d-levelled commitments or $O(\log^2(N))$ with Bulletproofs folding. Though the result is promising, we find it is hard to be applied in our approaches due to the same reasons above. Besides, the sizes of the extracted solutions (denoted by "slack" in [9]) also increase. Generally speaking, we cannot directly apply these approaches if the response batches f_i's.

Open problems. Our ring signature approach avoids most of the binary proof in existing approaches based on the fact that a one-out-of-many relation is not a *necessary* condition for ring signatures. An interesting question is finding the *sufficient and necessary* condition for ring signatures. We may further avoid unnecessary parts of our linear sum proof to improve the efficiency of ring signatures. Another interesting problem is to remove unnecessary parts in range proofs or balance proofs. For instance, it is sufficient to prove the balance based on Equation (11) even when $\boldsymbol{a}_i$'s and $\boldsymbol{b}_i$'s are not binary vectors. However, this requires additional proofs to show $\langle \boldsymbol{a}_i, \boldsymbol{2}^k \rangle \geq 0$ and $\langle \boldsymbol{b}_i, \boldsymbol{2}^k \rangle \geq 0$, which may not be efficient. Furthermore, the linear sum relation yields a "many-out-of-many" relation [13] which can reduce the anonymity set in RingCT. Unlike [13] which

generates *many* public key index from a *single* secret l by permutations and a linear mapping, the linear sum relation directly maps b_i's to P_i's which may be more efficient. Thus, the logarithmic-size linear sum proof seems to be a promising solution. Finally, supporting other commitment schemes in general-purpose lattice-based proof systems [4, 9] is also promising.

10 Acknowledgment

We gratefully acknowledge Dr. Xingye LU from the University of Hong Kong for the helpful technical discussions about lattice-based cryptography, Dr. Zuoxia Yu from the University of Wollongong and Dr. Aoning HU from the Southeast University for the discussion about ring signatures, as well as Dr. Muhammed Esgin from Monash University for the discussion of MatRiCT and pointing out some misleading parts. We would also like to thank the reviewers of CCS'21, Oakland'21, and Oakland'22 for their valuable comments. This research is partially supported by HK RGC GRF PolyU 15216721/Q86A and HK PolyU Start-Up ZVUE.

References

1. Albrecht, M.R., Player, R., Scott, S.: On the Concrete Hardness of Learning With Errors. In: Journal of Mathematical Cryptology (2015)
2. Anonymous: RingCT Implementation. `https://github.com/Anonymous-000/RingCT` (2022)
3. Attema, T., Cramer, R., Fehr, S.: Compressing Proofs of k-out-of-N Partial Knowledge. In: Proc. of the Annual International Cryptology Conference (CRYPTO). Springer (2021)
4. Baum, C., Bootle, J., Cerulli, A., Del Pino, R., Groth, J., Lyubashevsky, V.: Sublinear Lattice-based Zero-knowledge Arguments for Arithmetic Circuits. In: Proc. of the Annual International Cryptology Conference (CRYPTO). Springer (2018)
5. Baum, C., Damgård, I., Lyubashevsky, V., Oechsner, S., Peikert, C.: More Efficient Commitments from Structured Lattice Assumptions. In: Proc. of the International Conference on Security and Cryptography for Networks (SCN). Springer (2018)
6. Ben-Sasson, E., Chiesa, A., Genkin, D., Tromer, E., Virza, M.: SNARKs for C: Verifying Program Executions Succinctly and in Zero Knowledge. In: Proc. of the Annual International Cryptology Conference (CRYPTO). Springer (2013)
7. Bootle, J., Cerulli, A., Chaidos, P., Ghadafi, E., Groth, J., Petit, C.: Short Accountable Ring Signatures Based on DDH. In: Proc. of the European Symposium on Research in Computer Security (ESORICS). Springer (2015)
8. Bootle, J., Cerulli, A., Chaidos, P., Groth, J., Petit, C.: Efficient Zero-Knowledge Arguments for Arithmetic Circuits in the Discrete Log Setting. In: Proc. of the Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT). Springer (2016)
9. Bootle, J., Lyubashevsky, V., Nguyen, N.K., Seiler, G.: A Non-PCP Approach to Succinct Quantum-safe Zero-knowledge. In: Proc. of the Annual International Cryptology Conference (CRYPTO). Springer (2020)

10. Bünz, B., Agrawal, S., Zamani, M., Boneh, D.: Zether: Towards Privacy in a Smart Contract World. In: Proc. of the International Conference on Financial Cryptography and Data Security (FC). Springer (2020)
11. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short Proofs for Confidential Transactions and More. In: Proc. of the IEEE Symposium on Security and Privacy (Oakland). IEEE (2018)
12. Del Pino, R., Lyubashevsky, V., Seiler, G.: Lattice-Based Group Signatures and Zero-Knowledge Proofs of Automorphism Stability. In: Proc. of the ACM Conference on Computer & Communications Security (CCS). ACM (2018)
13. Diamond, B.E.: "Many-out-of-Many" Proofs with Applications to Anonymous Zether. In: Proc. of the IEEE Symposium on Security and Privacy (Oakland). IEEE (2020)
14. Esgin, M.F., Steinfeld, R., Liu, J.K., Liu, D.: Lattice-Based Zero-Knowledge Proofs: New Techniques for Shorter and Faster Constructions and Applications. In: Proc. of the Annual International Cryptology Conference (CRYPTO). Springer (2019)
15. Esgin, M.F., Steinfeld, R., Sakzad, A., Liu, J.K., Liu, D.: Short Lattice-Based One-Out-of-Many Proofs and Applications to Ring Signatures. In: Proc. of the International Conference on Applied Cryptography and Network Security (ACNS). Springer (2019)
16. Esgin, M.F., Steinfeld, R., Zhao, R.K.: MatRiCT+: More Efficient Post-Quantum Private Blockchain Payments. In: Proc. of the IEEE Symposium on Security and Privacy (Oakland) (2022)
17. Esgin, M.F., Zhao, R.K., Steinfeld, R., Liu, J.K., Liu, D.: MatRiCT: Efficient, Scalable and Post-Quantum Blockchain Confidential Transactions Protocol. In: Proc. of the ACM Conference on Computer & Communications Security (CCS). ACM (2019)
18. Gabizon, A., Williamson, Z.J., Ciobotaru, O.: PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive Arguments of Knowledge. In: IACR Cryptology ePrint Archive (2019)
19. Groth, J.: On the Size of Pairing-based Non-Interactive Arguments. In: Proc. of the Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT). Springer (2016)
20. Groth, J., Kohlweiss, M.: One-Out-of-Many Proofs: Or How to Leak a Secret and Spend a Coin. In: Proc. of the Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT). Springer (2015)
21. Kiltz, E., Lyubashevsky, V., Schaffner, C.: A Concrete Treatment of Fiat-Shamir Signatures in the Quantum Random-Oracle Model. In: Proc. of the Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT). Springer (2018)
22. Langlois, A., Stehlé, D.: Worst-Case to Average-Case Reductions for Module Lattices. In: Designs, Codes and Cryptography. Springer (2015)
23. Li, S., Jovanovic, P., ChristianMct: LAGO. `https://github.com/dedis/lago` (2022)
24. Libert, B., Ling, S., Nguyen, K., Wang, H.: Zero-Knowledge Arguments for Lattice-Based Accumulators: Logarithmic-Size Ring Signatures and Group Signatures without Trapdoors. In: Proc. of the Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT). Springer (2016)
25. Lu, X., Au, M.H., Zhang, Z.: Raptor: A Practical Lattice-Based (Linkable) Ring Signature. In: Proc. of the International Conference on Applied Cryptography and Network Security (ACNS). Springer (2019)

26. Lyubashevsky, V.: Lattice Signatures Without Trapdoors. In: Proc. of the Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT). Springer (2012)
27. Noether, S.: Ring Signature Confidential Transactions for Monero. In: IACR Cryptology ePrint Archive (2015)
28. Noether, S., Mackenzie, A., et al.: Ring Confidential Transactions. In: Ledger (2016)
29. Parno, B., Howell, J., Gentry, C., Raykova, M.: Pinocchio: Nearly Practical Verifiable Computation. In: Proc. of the IEEE Symposium on Security and Privacy (Oakland). IEEE (2013)
30. Poelstra, A., Back, A., Friedenbach, M., Maxwell, G., Wuille, P.: Confidential Assets. In: Proc. of the International Conference on Financial Cryptography and Data Security (FC). Springer (2018)
31. Hash Team: Hcash. `https://h.cash/` (2022)
32. Monero Project: Monero. `https://www.getmonero.org/` (2022)
33. Rivest, R.L., Shamir, A., Tauman, Y.: How to Leak a Secret. In: Proc. of the Annual International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT). Springer (2001)
34. Sasson, E.B., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M.: Zerocash: Decentralized Anonymous Payments from Bitcoin. In: Proc. of the IEEE Symposium on Security and Privacy (Oakland). IEEE (2014)
35. Sun, S.F., Au, M.H., Liu, J.K., Yuen, T.H.: RingCT 2.0: A Compact Accumulator-Based (Linkable Ring Signature) Protocol for Blockchain Cryptocurrency Monero. In: Proc. of the European Symposium on Research in Computer Security (ESORICS). Springer (2017)
36. Torres, W.A.A., Steinfeld, R., Sakzad, A., Liu, J.K., Kuchta, V., Bhattacharjee, N., Au, M.H., Cheng, J.: Post-Quantum One-Time Linkable Ring Signature and Application to Ring Confidential Transactions in Blockchain (Lattice RingCT v1. 0). In: Proc. of the Australasian Conference on Information Security and Privacy (ACISP). Springer (2018)
37. Yang, R., Au, M.H., Lai, J., Xu, Q., Yu, Z.: Lattice-Based Techniques for Accountable Anonymity: Composition of Abstract Stern's Protocols and Weak PRF with Efficient Protocols from LWR. In: IACR Cryptology ePrint Archive (2017)
38. Yuen, T.H., Sun, S.f., Liu, J.K., Au, M.H., Esgin, M.F., Zhang, Q., Gu, D.: RingCT 3.0 for Blockchain Confidential Transaction: Shorter Size and Stronger Security. In: Cryptology ePrint Archive (2019)

A Additional Lemmas

Lemma 6. ***(Lemma 8 in [14])*** *For any $f, g \in R = \mathbb{Z}[X]/(X^d + 1)$, we have the following relations:*

- $\|f\| \leq \sqrt{d} \cdot \|f\|_\infty$,
- $\|f\| \leq \|f\|_1 \leq \sqrt{d} \cdot \|f\|$,
- $\|f \cdot g\| \leq \sqrt{d} \cdot \|f\| \cdot \|g\|$,
- $\|f \cdot g\|_\infty \leq \|f\| \cdot \|g\|$,
- $\|f \cdot g\|_\infty \leq \|f\|_1 \cdot \|g\|_\infty$,
- $\|\prod_{i=1}^{n} f_i\|_\infty \leq (\prod_{i=1}^{n-1} \|f_i\|_1) \cdot \|f_n\|_\infty$.

Lemma 7. ***(Theorem 4.4 in [26])***

- *For any $k > 0$, $Pr[|z| > k\sigma; z \leftarrow D_\sigma] \leq 2e^{-k^2/2}$.*
- *For any $k > 1$, $Pr[\|\boldsymbol{z}\| > k\sigma\sqrt{s}; \boldsymbol{z} \leftarrow D_\sigma^s] < k^s e^{s(1-k^2)/2}$.*

Lemma 8. ***(Theorem 4.6 in [26])*** *Let h be a probability distribution over $V \in \mathbb{Z}^s$ where $s \geq 1$ and the norm of all elements is less than T. Let $\boldsymbol{c} \leftarrow h$ and $\phi > 0$. Considering an algorithm that samples $\boldsymbol{y} \leftarrow D_\sigma^s$ and outputs $Rej(\boldsymbol{z}, \boldsymbol{c}, \phi, T)$ for $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{c}$. The probability that the algorithm outputs 1 is within 2^{-100} of $1/\mu(\phi)$ where $\mu(\phi) = e^{12/\phi + 1/(2\phi^2)}$. When the output is 1, the statistical distance between the distribution of $\boldsymbol{z}$ and D_σ^s is at most 2^{-100}.*

Lemma 9. ***(Lemma 6 in [14])*** *For $f, g \in R$ and $k \in \mathbb{Z}^*$, if $f \cdot g^k = 0$ in R_q, then $f \cdot g = 0$ in R_q.*

Lemma 10. *Considering independent vectors $\boldsymbol{y}_1, \cdots, \boldsymbol{y}_s$ with distributions $D_{\sigma_1}^d, \cdots, D_{\sigma_s}^d$ for $d \geq 1$. If $\sigma_i \geq \tau(\mathbb{Z}^d)/\sqrt{\pi}$ for all $i \in [1, s]$ where $\tau(\mathbb{Z}^d)$ is a smoothing parameter of $\mathbb{Z}^d$, the distribution of $\sum_{i=1}^{s} \boldsymbol{y}_i$ is statistically close to $D^d_{\sqrt{\sum_{i=1}^{s} \sigma_i^2}}$.*

In particular, we have the distribution of $\sum_{i=1}^{s} \boldsymbol{y}_i$ is statistically close to $D^d_{\sigma\sqrt{s}}$ if $D^d_{\sigma_i} = D^d_\sigma$ **(Lemma 9 in [14])** and the distribution of $\sum_{i=1}^{s} \boldsymbol{y}_i$ is statistically close to $D^d_{\sigma\sqrt{\sum_{i=1}^{s} 2^i}}$ if $D^d_{\sigma_i} = D^d_{\sigma\sqrt{2^i}}$ for all $i \in [1, s]$.

B Proof of Theorem 1

Proof. **Completeness:** Based on Lemma 8, the prover responds with probability $1/(\mu(\phi_1)\mu(\phi_2)\mu(\phi_3))$. As there are at most kN-many 1's in $\boldsymbol{b}$ and $HW(x) = w$, we have at most wkN non-zero elements in $x\boldsymbol{b}$. Since $\|x\|_\infty = p$, we have,

$$\|x\boldsymbol{b}\| \leq p\sqrt{wkN} = T_1. \tag{21}$$

Furthermore, we have $\|\boldsymbol{c}\|_\infty \leq \max(-\sum_{\omega_i<0}\omega_i, \sum_{\omega_i>0}\omega_i)$ based on $c_j = \sum_{i=0}^{N-1}\omega_i b_{i,j} \in [\sum_{\omega_i<0}\omega_i, \sum_{\omega_i>0}\omega_i]$. Since there are at most k non-zero elements in $\boldsymbol{c}$ and $HW(x) = w$, $x\boldsymbol{c}$ has at most wk non-zero elements. Thus,

$$\|x\boldsymbol{c}\| \leq \max(-\sum_{\omega_i<0}\omega_i, \sum_{\omega_i>0}\omega_i)\cdot p\sqrt{wk} = T_2. \tag{22}$$

Let $\boldsymbol{\omega} = (\omega_0, \cdots, \omega_{N-1})$. Based on Lemma 6, we have

$$\begin{aligned}\|\boldsymbol{r}_c\| = \left\|\sum_{i=0}^{N-1}\omega_i\boldsymbol{r}_{b,i}\right\| \leq \sum_{i=0}^{N-1}\left(|\omega_i|\cdot\|\boldsymbol{r}_{b,i}\|\right)\\ \leq\sqrt{md}\cdot\|\boldsymbol{r}_{b,i}\|_\infty\cdot\|\boldsymbol{\omega}\|_1 \leq \mathcal{B}\sqrt{md}\|\boldsymbol{\omega}\|_1,\end{aligned} \tag{23}$$

and thus,

$$\begin{aligned}&\|x(\boldsymbol{r}_c, \boldsymbol{r}_b, \boldsymbol{r}_{b,0}, \cdots, \boldsymbol{r}_{b,N-1})\|\\ =&(\|x\boldsymbol{r}_c\|^2 + \|x(\boldsymbol{r}_b, \boldsymbol{r}_{b,0}, \cdots, \boldsymbol{r}_{b,N-1})\|^2)^{1/2}\\ \leq&\mathcal{B}wp\sqrt{md(\|\boldsymbol{\omega}\|_1^2 + N + 1)} = T_3.\end{aligned} \tag{24}$$

Therefore, based on Lemma 8, the distributions of $g_{i,j}$'s, f_j's, and $\boldsymbol{z}, \boldsymbol{z}_g, (\boldsymbol{z}_{b,i})_{i=0}^{N-1}$ are statistically close to $D^d_{\phi_1 T_1}$, $D^d_{\phi_2 T_2}$, and $D^{md}_{\phi_3 T_3}$ respectively. Except with negligible probability, we have the following relations based on Lemma 7:

$$\begin{aligned}&\|g_{i,j}\| \leq 2(\phi_1 T_1)\sqrt{d}, \quad \forall i \in [0, N), j \in [0, k),\\ &\|f_j\| \leq 2(\phi_2 T_2)\sqrt{d}, \quad \forall j \in [0, k),\\ &\|\boldsymbol{z}\|, \|\boldsymbol{z}_g\|, (\|\boldsymbol{z}_{b,i}\|)_{i=0}^{N-1} \leq 2(\phi_3 T_3)\sqrt{md},\end{aligned} \tag{25}$$

which satisfy steps 38, 40, and 41 of verification.

Finally, since $\boldsymbol{z}_b = \sum_{i=0}^{N-1}\zeta^i\boldsymbol{z}_{b,i}$, we have

$$\begin{aligned}\|\boldsymbol{z}_b\| = \left\|\sum_{i=0}^{N-1}\zeta^i\boldsymbol{z}_{b,i}\right\| \leq \sum_{i=0}^{N-1}\|\zeta^i\boldsymbol{z}_{b,i}\|\\ \leq\sqrt{md}\Big(\sum_{i=0}^{N-1}\|\zeta\|^i\cdot\|\boldsymbol{z}_{b,i}\|\Big) \leq 2md\phi_3 T_3\sum_{i=0}^{N-1}(wp)^i.\end{aligned} \tag{26}$$

$(3, N+1)$-special soundness:

Linear equation relation. We first prove $F'(\boldsymbol{b}_0, \cdots, \boldsymbol{b}_{N-1}) = 0$ relation in $\mathcal{R}'_{LE}$. Given 3 distinct challenges, (x, x', x''), we have 3 accepted responses $(\boldsymbol{f}_1, \boldsymbol{g}, \boldsymbol{z}, \boldsymbol{z}_g, (\boldsymbol{z}_{b,i})_{i=0}^{N-1})$, $(\boldsymbol{f}'_1, \boldsymbol{g}', \boldsymbol{z}', \boldsymbol{z}'_g, (\boldsymbol{z}'_{b,i})_{i=0}^{N-1})$, and $(\boldsymbol{f}''_1, \boldsymbol{g}'', \boldsymbol{z}'', \boldsymbol{z}''_g, (\boldsymbol{z}''_{b,i})_{i=0}^{N-1})$ with the same inputs and commitments $d_{sum}, (B_i)_{i=0}^{N-1}, D, E, F, (G_i)_{i=0}^{N-1}$. Set $C = \sum_{i=0}^{N-1}\omega_i B_i$. For each transcript, compute $f_0 = d_{sum} - \sum_{j=1}^{k-1}2^j\cdot f_j$ on R

and rebuild $\boldsymbol{f} = (f_0, \cdots f_{k-1})$. Obviously, we have $\langle \boldsymbol{f}, \mathbf{2}^k \rangle = d_{sum}$ (so do $\boldsymbol{f}'$ and $\boldsymbol{f}''$). Taking $(\boldsymbol{f}, \boldsymbol{z})$, $(\boldsymbol{f}', \boldsymbol{z}')$, and $(\boldsymbol{f}'', \boldsymbol{z}'')$, we have

$$xC + D = \mathrm{Com}_{ck}(\boldsymbol{f}; \boldsymbol{z}), \tag{27}$$

$$x'C + D = \mathrm{Com}_{ck}(\boldsymbol{f}'; \boldsymbol{z}'), \tag{28}$$

$$x''C + D = \mathrm{Com}_{ck}(\boldsymbol{f}''; \boldsymbol{z}''). \tag{29}$$

Subtracting Equation (28) from Equation (27), we get

$$(x - x')C = \mathrm{Com}_{ck}(\boldsymbol{f} - \boldsymbol{f}'; \boldsymbol{z} - \boldsymbol{z}') := \mathrm{Com}_{ck}(\widehat{\boldsymbol{c}}; \widehat{\boldsymbol{r}}_c). \tag{30}$$

Setting $y = x - x'$ as a relaxation factor, we extract a valid opening $(\widehat{\boldsymbol{c}}; \widehat{\boldsymbol{r}}_c)$ to yC and prove the claimed bound for $\mathcal{R}'_{LE}$.

Taking Equation (30) and (27), we have

$$\begin{aligned} yD &= y(xC + D) - xyC = \mathrm{Com}_{ck}(y\boldsymbol{f} - x\widehat{\boldsymbol{c}}; y\boldsymbol{z} - x\widehat{\boldsymbol{r}}_c) \\ &= \mathrm{Com}_{ck}(x\boldsymbol{f}' - x'\boldsymbol{f}; x\boldsymbol{z}' - x'\boldsymbol{z}) := \mathrm{Com}_{ck}(\widehat{\boldsymbol{d}}; \widehat{\boldsymbol{r}}_d). \end{aligned} \tag{31}$$

Obviously, based on the definition of $\widehat{\boldsymbol{c}}$ and $\widehat{\boldsymbol{d}}$, we have $y\boldsymbol{f} = x\widehat{\boldsymbol{c}} + \widehat{\boldsymbol{d}}$.

As we conduct step 35 on R instead of R_q, we have $\langle \boldsymbol{f}, \mathbf{2}^k \rangle = d_{sum}$, and thus $x\langle \widehat{\boldsymbol{c}}, \mathbf{2}^k \rangle + \langle \widehat{\boldsymbol{d}}, \mathbf{2}^k \rangle = d_{sum}$. Based on the γ_{LE}-binding property of the commitment scheme, the PPT prover cannot extract a new valid opening of yC and yD with non-negligible probability. Thus, we also have $x'\langle \widehat{\boldsymbol{c}}, \mathbf{2}^k \rangle + \langle \widehat{\boldsymbol{d}}, \mathbf{2}^k \rangle = d_{sum}$, which implies $\langle \widehat{\boldsymbol{c}}, \mathbf{2}^k \rangle = 0$ for distinct challenges. Considering the definition of C, we have

$$C = \sum_{i=0}^{N-1} \omega_i B_i = \mathrm{Com}_{ck}(\sum_{i=0}^{N-1} \omega_i \boldsymbol{b}_i; \sum_{i=0}^{N-1} \omega_i \boldsymbol{r}_{b,i}), \tag{32}$$

and thus

$$\begin{aligned} & y\langle \sum_{i=0}^{N-1} \omega_i \boldsymbol{b}_i, \mathbf{2}^k \rangle = \langle \widehat{\boldsymbol{c}}, \mathbf{2}^k \rangle = 0 \\ \Longrightarrow & \sum_{i=0}^{N-1} \Big(\omega_i \cdot \langle \mathbf{2}^k, \boldsymbol{b}_i \rangle \Big) = F'(\boldsymbol{b}_0, \cdots, \boldsymbol{b}_{N-1}) = 0, \end{aligned} \tag{33}$$

which proves the $F'(\boldsymbol{b}_0, \cdots, \boldsymbol{b}_{N-1}) = 0$ relation in $\mathcal{R}'_{LE}$.

Binary relation. We first consider the relation of $x\sum_{i=0}^{N-1} \zeta^i B_i + G = \mathrm{Com}_{ck}(\sum_{i=0}^{N-1} \zeta^i \boldsymbol{g}_i; \boldsymbol{z}_b)$. Given 2 distinct challenges, (x, x'), we have 2 accepted responses, $(\sum_{i=0}^{N-1} \zeta^i \boldsymbol{g}_i, \boldsymbol{z}_b)$ and $(\sum_{i=0}^{N-1} \zeta^i \boldsymbol{g}'_i, \boldsymbol{z}'_b)$ with the same ζ, inputs, and commitments. Accordingly, we have

$$x \sum_{i=0}^{N-1} \zeta^i B_i + G = \mathrm{Com}_{ck}(\sum_{i=0}^{N-1} \zeta^i \boldsymbol{g}_i; \boldsymbol{z}_b), \tag{34}$$

$$x' \sum_{i=0}^{N-1} \zeta^i B_i + G = \mathrm{Com}_{ck}(\sum_{i=0}^{N-1} \zeta^i \boldsymbol{g}'_i; \boldsymbol{z}'_b) \tag{35}$$

Subtracting Equation (35) from Equation (34), we get

$$(x-x')\sum_{i=0}^{N-1}\zeta^i B_i = \mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta^i(\boldsymbol{g}_i-\boldsymbol{g}'_i);\boldsymbol{z}_b-\boldsymbol{z}'_b) := \mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta^i\widetilde{\boldsymbol{b}}_i;\widetilde{\boldsymbol{r}}_b).$$

Setting $y = x-x'$, we extract an opening $(\sum_{i=0}^{N-1}\zeta^i\widetilde{\boldsymbol{b}}_i;\widetilde{\boldsymbol{r}}_b)$ to $y\sum_{i=0}^{N-1}\zeta^i B_i$, where $\|\widetilde{\boldsymbol{r}}_b\| \leq 4md\phi_3 T_3\sum_{i=0}^{N-1}(wp)^i$. Taking the opening to Equation (34), we have

$$\begin{aligned} yG &= y\mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta^i\boldsymbol{g}_i;\boldsymbol{z}_b) - x\sum_{i=0}^{N-1}\zeta^i yB_i \\ &= \mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta^i(y\boldsymbol{g}_i - x\widetilde{\boldsymbol{b}}_i); y\boldsymbol{z}_b - x\widetilde{\boldsymbol{r}}_b) := \mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta^i\hat{\boldsymbol{t}}_i;\hat{\boldsymbol{r}}_g), \end{aligned} \tag{36}$$

which gives an opening $(\sum_{i=0}^{N-1}\zeta^i\hat{\boldsymbol{t}}_i;\hat{\boldsymbol{r}}_g)$ to yG.

For $\sum_{i=0}^{N-1}\zeta^i yB_i$ part, given N distinct challenges, $(\zeta_s)_{s=0}^{N-1}$, we have N accepted responses, $(\sum_{i=0}^{N-1}\zeta_s^i\widetilde{\boldsymbol{b}}_i^{(s)};\widetilde{\boldsymbol{r}}_b^{(s)})_{s=0}^{N-1}$ with the same inputs and commitments. Accordingly, we have

$$\begin{pmatrix} 1 & \zeta_0 & \cdots & \zeta_0^{N-1} \\ 1 & \zeta_1 & \cdots & \zeta_1^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta_{N-1} & \cdots & \zeta_{N-1}^{N-1} \end{pmatrix} \cdot \begin{pmatrix} yB_0 \\ yB_1 \\ \vdots \\ yB_{N-1} \end{pmatrix} = \begin{pmatrix} \mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta_0^i\widetilde{\boldsymbol{b}}_i^{(0)};\widetilde{\boldsymbol{r}}_b^{(0)}) \\ \mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta_1^i\widetilde{\boldsymbol{b}}_i^{(1)};\widetilde{\boldsymbol{r}}_b^{(1)}) \\ \vdots \\ \mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta_{N-1}^i\widetilde{\boldsymbol{b}}_i^{(N-1)};\widetilde{\boldsymbol{r}}_b^{(N-1)}) \end{pmatrix}. \tag{37}$$

Let the Vandermonde matrix on the left hand side be $\boldsymbol{V}_{N-1}$. We can obtain $(\hat{\boldsymbol{b}}_{N-1};\hat{\boldsymbol{r}}_{b,N-1})$ as the opening of $y_\zeta yB_{N-1}$ with another relaxation factor $y_\zeta = \det(\boldsymbol{V}_{N-1})$. Specifically, $\widetilde{\boldsymbol{b}}_{N-1}, \widetilde{\boldsymbol{r}}_{b,N-1}$ can be derived based on Equation (4).

Note that directly compute the relaxed opening of yB_s's requires to compute the s-th row of $\mathrm{adj}(\boldsymbol{V}_{N-1})$, which will be extremal complicated to derive the norms of the opening. To avoid this problem, we swap the s-th row with the last row:

$$\begin{pmatrix} 1 & \zeta_0 & \cdots & \zeta_0^{N-1} \\ 1 & \zeta_1 & \cdots & \zeta_1^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta_{N-1} & \cdots & \zeta_{N-1}^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta_s & \cdots & \zeta_s^{N-1} \end{pmatrix} \cdot \begin{pmatrix} yB_0 \\ yB_1 \\ \vdots \\ yB_{N-1} \\ \vdots \\ yB_s \end{pmatrix} = \begin{pmatrix} \mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta_0^i\widetilde{\boldsymbol{b}}_i^{(0)};\widetilde{\boldsymbol{r}}_b^{(0)}) \\ \mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta_1^i\widetilde{\boldsymbol{b}}_i^{(1)};\widetilde{\boldsymbol{r}}_b^{(1)}) \\ \vdots \\ \mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta_{N-1}^i\widetilde{\boldsymbol{b}}_i^{(N-1)};\widetilde{\boldsymbol{r}}_b^{(N-1)}) \\ \vdots \\ \mathrm{Com}_{ck}(\sum_{i=0}^{N-1}\zeta_s^i\widetilde{\boldsymbol{b}}_i^{(s)};\widetilde{\boldsymbol{r}}_b^{(s)}) \end{pmatrix}. \tag{38}$$

Let the Vandermonde matrix on the left hand side be $\boldsymbol{V}_s$. Since $\boldsymbol{V}_s$ is derived by swapping two rows of $\boldsymbol{V}_{N-1}$, we have $\det(\boldsymbol{V}_s) = -\det(\boldsymbol{V}_{N-1}) = -y_\zeta$ for all

$s \neq N-1$. Similar to the previous approach, we can obtain an opening $(\hat{\boldsymbol{b}}_s; \hat{\boldsymbol{r}}_{b,s})$ of $y_\zeta y B_s$ based on Equation (4), where $\hat{\boldsymbol{r}}_{b,s} = \sum_{i=0}^{N-1} \Gamma_{i,s} \widetilde{\boldsymbol{r}}_b^{(s)}$.

Let $\kappa = N(N-1)/2$ and $\kappa' = (N-1)(N-2)/2$. Based on Lemma 1 and Lemma 4, we have the bound of $\|\hat{\boldsymbol{r}}_{b,s}\|$:

$$\begin{aligned} \|\hat{\boldsymbol{r}}_{b,s}\| \leq & Nd(2p)^{\kappa'} w^{\kappa'-1} \cdot 4md\phi_3 T_3 \sum_{i=0}^{N-1} (wp)^i \\ \leq & 2^{\kappa+1} N\mathcal{B} p^\kappa w^{\kappa-1} md^2 \phi_3 \sqrt{md(\|\boldsymbol{\omega}\|_1^2 + N + 1)} \sum_{i=0}^{N-1} (wp)^i = \gamma_{LE}. \end{aligned} \tag{39}$$

Taking the opening $(\sum_{i=0}^{N-1} \zeta^i \hat{\boldsymbol{t}}_i; \hat{\boldsymbol{r}}_g)$ to yG, based on the last step of verification, we have

$$y_\zeta y \sum_{i=0}^{N-1} \zeta^i \boldsymbol{g}_i = x \sum_{i=0}^{N-1} \zeta^i \hat{\boldsymbol{b}}_i + \sum_{i=0}^{N-1} \zeta^i \widetilde{\boldsymbol{t}}_i = \sum_{i=0}^{N-1} \zeta^i (x\hat{\boldsymbol{b}}_i + y_\zeta \hat{\boldsymbol{t}}_i). \tag{40}$$

Based on the γ_{LE}-binding property of the commitment scheme, the PPT prover cannot extract a new valid opening of $y_\zeta y B_s$'s. Meanwhile, though different challenges generate different G's, the γ_{LE}-binding property of the commitment scheme avoids extracting different $\hat{\boldsymbol{t}}_i$'s such that $\mathrm{Com}_{ck}(\sum_{i=0}^{N-1} \zeta^i \hat{\boldsymbol{t}}_i; \hat{\boldsymbol{r}}_g)$ (different $\hat{\boldsymbol{t}}_i$'s result in different $\hat{\boldsymbol{b}}_i$'s and break the γ_{LE}-binding property). Thus, the same form in Equation (40) holds for $N+1$ challenges $(\zeta_s)_{s=0}^N$, which indicates

$$y_\zeta y \boldsymbol{g}_i = x\hat{\boldsymbol{b}}_i + y_\zeta \hat{\boldsymbol{t}}_i. \tag{41}$$

For the relation of $xE + F = \mathrm{Com}_{ck}(\boldsymbol{h}; \boldsymbol{z}_g)$, given 3 distinct challenges, (x, x', x''), we have 3 accepted responses, $(\boldsymbol{h}; \boldsymbol{z}_b)$, $(\boldsymbol{h}'; \boldsymbol{z}_b')$, and $(\boldsymbol{h}''; \boldsymbol{z}_b'')$ with the same inputs and commitments. Accordingly, we have

$$xE + F = \mathrm{Com}_{ck}(\boldsymbol{h}; \boldsymbol{z}_b), \tag{42}$$

$$x'E + F = \mathrm{Com}_{ck}(\boldsymbol{h}'; \boldsymbol{z}_b'), \tag{43}$$

$$x''E + F = \mathrm{Com}_{ck}(\boldsymbol{h}''; \boldsymbol{z}_b''). \tag{44}$$

Using the same extraction approach, we can derive the opening $(\widehat{\boldsymbol{e}}; \widehat{\boldsymbol{r}}_b)$ to yE and $(\widehat{\boldsymbol{s}}; \widehat{\boldsymbol{r}}_t)$ to yF such that $y\boldsymbol{h} = x\widehat{\boldsymbol{e}} + \widehat{\boldsymbol{s}}$. For each element of $\boldsymbol{h}$, we have the following relations for all $i \in [0, N)$ and $j \in [0, k)$:

$$yh_{i,j} = y(g_{i,j}(x - g_{i,j})) = x\widehat{e}_{i,j} + \widehat{s}_{i,j}. \tag{45}$$

Based on the γ_{LE}-binding property of the commitment scheme, the responses to x'' will have the same form in Equation (45), which indicates Equation (45) holds for x, x', x''.

Based on Equation (45), we have the following equation:

$$\begin{aligned} & y_\zeta^2 y(x\widehat{e}_{i,j} + \widehat{s}_{i,j}) = y_\zeta^2 y(yg_{i,j}(x - g_{i,j})) \\ = & y_\zeta y g_{i,j}(xy_\zeta y - y_\zeta y g_{i,j}) = (x\widehat{b}_{i,j} + y_\zeta \widehat{t}_{i,j})(xy_\zeta y - x\widehat{b}_{i,j} - y_\zeta \widehat{t}_{i,j}) \\ = & x^2\big(\widehat{b}_{i,j}(y_\zeta y - \widehat{b}_{i,j})\big) + x\big(\widehat{t}_{i,j}(y_\zeta y - 2\widehat{b}_{i,j})\big) - y_\zeta^2 \widehat{t}_{i,j}^2, \end{aligned} \tag{46}$$

and thus

$$x^2\big(\widehat{b}_{i,j}(y_\zeta y-\widehat{b}_{i,j})\big)+x\big(\widehat{t}_{i,j}(y_\zeta y-2\widehat{b}_{i,j})-y_\zeta^2 y\widehat{e}_{i,j}\big)+\big(-y_\zeta^2\widehat{t}_{i,j}^2-y_\zeta^2 y\widehat{s}_{i,j}\big)=0, \tag{47}$$

Since Equation (47) holds for x, x', and x'', we have the following system:

$$\begin{pmatrix}1 & x & x^2\\ 1 & x' & x'^2\\ 1 & x'' & x''^2\end{pmatrix}\cdot\begin{pmatrix}-y_\zeta^2\widehat{t}_{i,j}^2-y_\zeta^2 y\widehat{s}_{i,j}\\ \widehat{t}_{i,j}(y_\zeta y-2\widehat{b}_{i,j})-y_\zeta^2 y\widehat{e}_{i,j}\\ \widehat{b}_{i,j}(y_\zeta y-\widehat{b}_{i,j})\end{pmatrix}=\mathbf{0}. \tag{48}$$

As all operations are conducted on a field R_q, the Vandermonde matrix on the left is invertible for distinct challenges. We have $\widehat{b}_{i,j}(y_\zeta y-\widehat{b}_{i,j})=0$, which implies $\widehat{b}_{i,j}=0$ or $\widehat{b}_{i,j}=y_\zeta y$, i.e., $\widehat{b}_{i,j}=y_\zeta y b_{i,j}$ for $b_{i,j}\in\{0,1\}$. Thus, all $\boldsymbol{b}_i$'s are binary vectors.

SHVZK: Assume the protocol is not aborted. The simulator samples $\boldsymbol{r}_b\leftarrow\{-\mathcal{B},\cdots,\mathcal{B}\}^{md}$, $g_{i,j}\leftarrow D^d_{\phi_1 T_1}$ for all $i\in[0,N)$ and $j\in[0,k)$, $f_j\leftarrow D^d_{\phi_2 T_2}$ for all $j\in(0,k)$, $\boldsymbol{z},\boldsymbol{z}_g,(\boldsymbol{z}_{b,i})_{i=0}^{N-1}\leftarrow D^{md}_{\phi_3 T_3}$, and sets $C=\sum_{i=0}^{N-1}\omega_i B_i$, $E=\mathrm{Com}_{ck}(\mathbf{0};\boldsymbol{r}_b)$, $\boldsymbol{f}_1=(f_1,\cdots,f_{k-1})$, $\boldsymbol{f}=(f_0,\cdots,f_{k-1})$, $d_{sum}=\langle\boldsymbol{f},\mathbf{2}^k\rangle$, $\boldsymbol{g}=(g_{0,0},\cdots,g_{s-1,k-1})$, $\boldsymbol{g}_i=(g_{i,0},\cdots,g_{i,k-1})$ for all $i\in[0,N)$. Then, given x, it computes $\boldsymbol{h}=(g_{i,j}(x-g_{i,j}))_{i=0,j=0}^{N-1,k-1}$, $D=\mathrm{Com}_{ck}(\boldsymbol{f};\boldsymbol{z})-xC$, $F=\mathrm{Com}_{ck}(\boldsymbol{g},\boldsymbol{h};\boldsymbol{z}_g)-xE$, and $G_i=\mathrm{Com}_{ck}(\boldsymbol{g}_i;\boldsymbol{z}_{b,i})-xB_i$ for all $i\in[0,N)$. Obviously, the simulated transcript $\big((d_{sum},D,E,F,(G_i)_{i=0}^{N-1}),x,(\boldsymbol{f}_1,\boldsymbol{g},\boldsymbol{z},\boldsymbol{z}_g,(\boldsymbol{z}_{b,i})_{i=0}^{N-1})\big)$ will be an accepted transcript.

Based on Lemma 7, the distributions of $\boldsymbol{f}_1,\boldsymbol{g},\boldsymbol{z},\boldsymbol{z}_g,(\boldsymbol{z}_{b,i})_{i=0}^{N-1}$ are statistically close to the real distributions. The simulated distributions of $d_{sum},D,F,(G_i)_{i=0}^{N-1}$ are the same as the real distributions. Finally, due to the hiding property of the commitment scheme, the distribution of simulated E is computationally indistinguishable from the real case based on the M-LWE assumption.

C Proof of Theorem 2

Proof. **Completeness:** Based on Lemma 8, the prover responds with probability $1/(\mu(\phi_1)\mu(\phi_2))$. As there are at most k-many 1's in $\boldsymbol{\delta}$ and $HW(x)=w$, we have at most wk-many non-zero elements in $x\boldsymbol{\delta}$. Since $\|x\|_\infty=p$, we have,

$$\|x\boldsymbol{\delta}\|\le p\sqrt{wk}=T_1. \tag{49}$$

Furthermore, based on Lemma 6, we have

$$\begin{aligned}
&\left\|x^k\boldsymbol{r}-\sum_{j=1}^{k-1}x^j\boldsymbol{\rho}_j\right\| \le \|x^k\boldsymbol{r}\|+\sum_{j=1}^{k-1}\|x^j\boldsymbol{\rho}_j\| \\
\le&\sqrt{md}\Big(\|x^k\boldsymbol{r}\|_\infty+\sum_{j=1}^{k-1}\|x^j\boldsymbol{\rho}_j\|_\infty\Big) \\
\le&\sqrt{md}\Big(\|x\|_1^k\cdot\|\boldsymbol{r}\|_\infty+\sum_{j=1}^{k-1}\|x\|^j\cdot\|\boldsymbol{\rho}_j\|_\infty\Big) \\
\le&\sqrt{md}\Big((wp)^k\mathcal{B}+\mathcal{B}\sum_{j=1}^{k-1}(wp)^j\Big)=\mathcal{B}\sqrt{md}\sum_{j=1}^{k}(wp)^j.
\end{aligned} \tag{50}$$

Denote $\boldsymbol{r}'=x^k\boldsymbol{r}-\sum_{j=1}^{k-1}x^j\boldsymbol{\rho}_j$, we have

$$\begin{aligned}
&\|x\boldsymbol{r}_b,\boldsymbol{r}'\|=(\|x\boldsymbol{r}_b\|^2+\|\boldsymbol{r}'\|^2)^{1/2} \\
\le&\mathcal{B}\sqrt{md}\Big(w^2p^2+(\sum_{j=1}^{k}w^jp^j)^2\Big)^{1/2}\le\mathcal{B}(wp)^k\sqrt{2md}=T_2.
\end{aligned} \tag{51}$$

Considering each element of $\boldsymbol{f}_1$. Based on Lemma 10, the sum of discrete normal variables behaves as its continuous counterpart. Thus, for all $j\in[0,k)$, we have the distribution of $\sum_{i=1}^{\beta-1}f_{j,i}$ is statistically close to $D^d_{\phi_1T_1\sqrt{\beta-1}}$. Therefore, for all $j\in[0,k)$ and $i\in(0,\beta)$, we have

$$\begin{aligned}
&\|f_{j,i}\|\le 2(\phi_1T_1)\sqrt{d}, \\
&\|f_{j,0}\|\le\Big\|x-\sum\nolimits_{i=1}^{\beta-1}f_{j,i}\Big\|\le\|x\|+\Big\|\sum\nolimits_{i=1}^{\beta-1}f_{j,i}\Big\| \\
&\qquad\le\sqrt{w}+2(\phi_1T_1)\sqrt{d(\beta-1)}\approx 2\phi_1p\sqrt{kwd\beta}, \\
&\|\boldsymbol{z}_b\|,\|\boldsymbol{z}_r\|\le 2(\phi_2T_2)\sqrt{md}.
\end{aligned} \tag{52}$$

$(k+1)$-special soundness: Given $(k+1)$ distinct challenges $(x_s)_{s=0}^k$, we have $(k+1)$ accepted responses $(\boldsymbol{f}_1^{(s)},\boldsymbol{z}_b^{(s)})_{s=0}^k$ with the same commitments $A,B,(E_j)_{j=0}^{k-1}$. For each transcript, compute $\boldsymbol{f}_{j,0}^{(s)}=x_s-\sum_{i=1}^{\beta-1}f_{i,j}^{(s)}$ for all $j\in[0,k)$, and rebuild $\boldsymbol{f}^{(s)}=(f_{0,0}^{(s)},\cdots f_{k-1,\beta-1}^{(s)})$. Taking $(\boldsymbol{f}^{(0)},\boldsymbol{z}_b^{(0)})$ and $(\boldsymbol{f}^{(1)},\boldsymbol{z}_b^{(1)})$, we have

$$x_0B+A=\mathrm{Com}_{ck}(\boldsymbol{f}^{(0)},\boldsymbol{z}_b^{(0)}), \tag{53}$$

$$x_1B+A=\mathrm{Com}_{ck}(\boldsymbol{f}^{(1)},\boldsymbol{z}_b^{(1)}). \tag{54}$$

Subtracting (54) from (53), we get $(x_0-x_1)B=\mathrm{Com}_{ck}(\boldsymbol{f}^{(0)}-\boldsymbol{f}^{(1)},\boldsymbol{z}_b^{(0)}-\boldsymbol{z}_b^{(1)})$, which gives us an opening of yB with a relaxation factor $y=(x_0-x_1)$:

$$yB=\mathrm{Com}_{ck}(\boldsymbol{f}^{(0)}-\boldsymbol{f}^{(1)},\boldsymbol{z}_b^{(0)}-\boldsymbol{z}_b^{(1)}):=\mathrm{Com}_{ck}(\widehat{\boldsymbol{b}},\widehat{\boldsymbol{r}}_b). \tag{55}$$

Subtracting x_0 times of (55) from y times of (53), we have:

$$\begin{aligned} yA &= \mathrm{Com}_{ck}(y\boldsymbol{f}^{(0)} - x_0\widehat{\boldsymbol{b}};\quad y\boldsymbol{z}_b^{(0)} - x_0\widehat{\boldsymbol{r}}_b) \\ &= \mathrm{Com}_{ck}(x_0\boldsymbol{f}^{(1)} - x_1\boldsymbol{f}^{(0)};\quad x_0\boldsymbol{z}_b^{(1)} - x_1\boldsymbol{z}_b^{(0)}) \\ &:= \mathrm{Com}_{ck}(\widehat{\boldsymbol{a}};\widehat{\boldsymbol{r}}_a). \end{aligned} \tag{56}$$

Obviously, we have $x_s\widehat{\boldsymbol{b}} + \widehat{\boldsymbol{a}} = y\boldsymbol{f}^{(s)}$ for $s = \{0,1\}$. Taking each element in $\widehat{\boldsymbol{b}} = (\widehat{b}_{j,i})_{i=0,j=0}^{\beta-1,k-1}$ and $\widehat{\boldsymbol{a}} = (\widehat{a}_{j,i})_{i=0,j=0}^{\beta-1,k-1}$, we have the following system for all $i \in [0,\beta), j \in [0,k)$:

$$x_s\widehat{b}_{j,i} + \widehat{a}_{j,i} = yf_{j,i}^{(s)}. \tag{57}$$

Taking $\sum_{i=0}^{\beta-1} f_{j,i}^{(s)} = x_s$ and Equation (57), we have

$$yx_s = \sum_{i=0}^{\beta-1} yf_{j,i}^{(s)} = \sum_{i=0}^{\beta-1} x_s\widehat{b}_{j,i} + \sum_{i=0}^{\beta-1} \widehat{a}_{j,i}, \tag{58}$$

and thus

$$0 = x_s(\sum_{i=0}^{\beta-1} \widehat{b}_{j,i} - y) + \sum_{i=0}^{\beta-1} \widehat{a}_{j,i}. \tag{59}$$

Based on the γ-binding property of the commitment scheme, Equation (59) holds for $s = \{0,1\}$. Therefore, we have $\sum_{i=0}^{\beta-1} \widehat{b}_{j,i} - y = 0$ and $\sum_{i=0}^{\beta-1} \widehat{a}_{j,i} = 0$, and thus $\sum_{i=0}^{\beta-1} \widehat{b}_{j,i} = y$, i.e., $\sum_{i=0}^{\beta-1} \widehat{b}_{j,i} = y\sum_{i=0}^{\beta-1} b_{j,i}$ for $\sum_{i=0}^{\beta-1} b_{j,i} = 1$. Based on step 23, we have $b_{j,0} = 1 - \sum_{i=1}^{\beta-1} b_{j,i}$. Thus, $\|b_{j,0}\| \le 1 + 4\phi_1\sqrt{kd(\beta-1)} \approx 4\phi_1\sqrt{kd\beta}$.

Now we construct b_i's for all $i \in [0,N)$ with $b_i = \prod_{j=0}^{k-1} b_{j,i_j}$, where i_j's are the digits of representation of i in base β such that $i = (i_0, \cdots, i_{k-1})$. Clearly, $b_i \neq 0$ if and only if $b_{j,i_j} \neq 0$ for all $j \in [0,k)$.

Based on Equation (55) and Lemma 6, we have

$$\begin{aligned} \|b_i\| &= \left\| \prod_{j=0}^{k-1} b_{j,i_j} \right\| \le d^{\frac{k-1}{2}} \prod_{j=0}^{k-1} \|b_{j,i_j}\| \\ &\le d^{\frac{k-1}{2}} \|b_{j,0}\|^k \le (4\phi_1\sqrt{k\beta})^k d^{k-\frac{1}{2}} = \gamma_{LS}. \end{aligned} \tag{60}$$

Considering the γ-binding property of the commitment scheme, the following equation holds for $(k+1)$ challenges

$$yf_{j,i}^{(s)} = x_s\widehat{b}_{j,i} + \widehat{a}_{j,i} = yx_sb_{j,i} + \widehat{a}_{j,i}, \quad s \in [0,k]. \tag{61}$$

We compute $\widehat{p}_i(x_s) = y^k \prod_{j=0}^{k-1} f_{j,i_j}^{(s)} = \prod_{j=0}^{k-1}(yx_sb_{j,i_j} + \widehat{a}_{j,i_j})$ for each $i \in [0,N)$. Obviously, for all $s \in [0,k]$, if $\widehat{p}_i(x_s)$ is a polynomial of degree k, then $b_{j,i_j} \neq 0$ for all $j \in [0,k)$, which indicates $b_i \neq 0$. Thus, for all $i \in [0,\beta)$, we have at least one b_i is not zero, i.e., $\|\boldsymbol{b}\| > 0$.

As the last verification step holds, we multiply both sides of the equation by y^k:

$$\begin{aligned}&\sum_{i=0}^{N-1}\widehat{p}_i(x_s)\cdot P_i-\sum_{j=0}^{k-1}y^kE_jx_s^j\\=&x_s^ky^k\sum_{i=0}^{N-1}P_i+\sum_{j=0}^{k-1}E'_jx_s^j=\mathrm{Com}_{ck}(\mathbf{0};y^k\boldsymbol{z}_r^{(s)}),\end{aligned}\tag{62}$$

where E'_j's are the terms multiplied by the monomials x_s^j's of degree at most $(k-1)$ and are independent from x_s. Taking all $(k+1)$ transcripts, we have

$$\begin{pmatrix}1&x_0&\cdots&x_0^k\\1&x_1&\cdots&x_1^k\\\vdots&\vdots&\ddots&\vdots\\1&x_k&\cdots&x_k^k\end{pmatrix}\cdot\begin{pmatrix}E'_0\\E'_1\\\vdots\\y^k\sum\limits_{i=0}^{N-1}b_iP_i\end{pmatrix}=\begin{pmatrix}\mathrm{Com}_{ck}(\mathbf{0};y^k\boldsymbol{z}_r^{(0)})\\\mathrm{Com}_{ck}(\mathbf{0};y^k\boldsymbol{z}_r^{(1)})\\\vdots\\\mathrm{Com}_{ck}(\mathbf{0};y^k\boldsymbol{z}_r^{(k)})\end{pmatrix}.\tag{63}$$

Let the Vandermonde matrix on the left hand side of Equation (63) be $\boldsymbol{V}$. Based on Equation (6), we can obtain $(\mathbf{0},y^k\widehat{\boldsymbol{r}})$ as the opening of $\det(\boldsymbol{V})\cdot y^k\sum_{i=0}^{N-1}b_iP_i$, where $\widehat{\boldsymbol{r}}=\sum_{i=0}^{N-1}\Gamma_i\boldsymbol{z}_r^{(i)}$ (Γ_i is defined in Equation (4)). Based on Lemma 9, we have

$$\begin{aligned}&\det(\boldsymbol{V})\cdot y^k\sum\nolimits_{i=0}^{N-1}b_iP_i=\mathrm{Com}_{ck}(\mathbf{0};y^k\widehat{\boldsymbol{r}})\\\Longrightarrow&y^k\Big(\det(\boldsymbol{V})\cdot\sum\nolimits_{i=0}^{N-1}b_iP_i-\mathrm{Com}_{ck}(\mathbf{0};\widehat{\boldsymbol{r}})\Big)=0\\\Longrightarrow&y\Big(\det(\boldsymbol{V})\cdot\sum\nolimits_{i=0}^{N-1}b_iP_i-\mathrm{Com}_{ck}(\mathbf{0};\widehat{\boldsymbol{r}})\Big)=0\\\Longrightarrow&\det(\boldsymbol{V})\cdot y\sum\nolimits_{i=0}^{N-1}b_iP_i=\mathrm{Com}_{ck}(\mathbf{0};y\widehat{\boldsymbol{r}})=0.\end{aligned}\tag{64}$$

Thus we extract an opening of $\det(\boldsymbol{V})\cdot y\sum_{i=0}^{N-1}b_iP_i$ as $(\mathbf{0},y\widehat{\boldsymbol{r}})$. Let $\kappa=k(k+1)/2$ and $\kappa'=k(k-1)/2$. Based on Lemma 1 and Lemma 4, we have the bound of $\|y\widehat{\boldsymbol{r}}\|$:

$$\begin{aligned}&\|y\widehat{\boldsymbol{r}}\|\le(k+1)d(2p)^{\kappa'+1}w^{\kappa'}\cdot2\phi_2T_2\sqrt{md}\\\le&(k+1)2^{\kappa'+2}\sqrt{2}\phi_2\mathcal{B}md^2w^{\kappa}p^{\kappa+1}=\gamma'_{LS}.\end{aligned}\tag{65}$$

SHVZK: Assume that the protocol is not aborted. The simulator samples $\boldsymbol{r}\leftarrow\{-\mathcal{B},\cdots,\mathcal{B}\}^{md}$, $f_{j,i}\leftarrow D^d_{\phi_1T_1}$ for all $i\in(0,\beta)$ and $j\in[0,k)$, $\boldsymbol{z}_b,\boldsymbol{z}_r\leftarrow D^{md}_{\phi_2T_2}$, $E_j\leftarrow\mathcal{U}(R_q^n)$ for all $j\in(0,k)$, and sets $B=\mathrm{Com}_{ck}(\mathbf{0};\boldsymbol{r})$. Then, given x, it computes $f_{j,0}=x-\sum_{i=1}^{\beta-1}f_{j,i}$ for all $j\in[0,k)$ and sets $\boldsymbol{f}=(f_{0,0},\cdots,f_{k-1,\beta-1})$, $A=\mathrm{Com}_{ck}(\boldsymbol{f};\boldsymbol{z}_b)-xB$, and $E_0=\sum_{i=0}^{N-1}(\prod_{j=0}^{k-1}f_{j,i_j})P_i-\mathrm{Com}_{ck}(\mathbf{0};\boldsymbol{z}_r)-\sum_{j=1}^{k-1}E_jx^j$. Obviously, the simulated transcript $((A,B,(E_j)_{j=0}^{k-1}),x,(\boldsymbol{f},\boldsymbol{z}_b,\boldsymbol{z}_r))$ will be an accepted transcript.

Based on Lemma 7, the distributions of $\boldsymbol{f}$, $\boldsymbol{z}_b$, $\boldsymbol{z}_v$ are statistically close to the real distributions. The simulated distributions of A and E_0 are the same as the real ones. Due to the hiding property of the commitment scheme, the distribution of simulated B is computationally indistinguishable from the real case. Finally, the simulated $E_1, \cdots, E_{k-1}$ are computationally indistinguishable from the real cases based on the M-LWE assumption.

Considering M inputs $(a_i)_{i=0}^{M-1}$. Let $a = \sum_{i=0}^{M-1} a_i$, $k = \lceil \log(a) \rceil$, and $a[j]$ be the j-th bit of a where $0 \leq j < k$. First, taking $\sum_{i=0}^{M-1} a_i[0]$ and writing it as the binary representation $(\tau_0'^{(0)}, \cdots, \tau_{k-1}'^{(0)})$ such that $\sum_{i=0}^{M-1} a_i[0] = \sum_{t=0}^{k-1} (\tau_t'^{(0)} \cdot 2^t)$, we have $a[0] = \tau_0'^{(0)}$. Furthermore, taking $\tau_1'^{(0)}$ and $\sum_{i=0}^{M-1} a_i[1]$, we can write the sum of them as the binary representation $(\tau_0'^{(1)}, \cdots, \tau_{k-1}'^{(1)})$ and derive $a[1] = \tau_0'^{(1)}$. In this way, we can observe that in order to derive $a[j]$, the j-th element consists of two parts: $\sum_{i=0}^{M-1} a_i[j]$ and $\sum_{t=1}^{j} \tau_t'^{(j-t)}$ (the latter one is the sum of all carries from $a[j-1], \cdots a[0]$). Thus, for all $j \in [0, k)$, writing $\sum_{i=0}^{M-1} a_i[j] + \sum_{t=1}^{j} \tau_t'^{(j-t)}$ as the binary representation $(\tau_0'^{(j)}, \cdots, \tau_{k-1}'^{(j)})$, we have

$$\sum_{i=0}^{M-1} a_i[j] + \sum_{t=1}^{j} \tau_t'^{(j-t)} = \sum_{t=0}^{k-1} (\tau_t'^{(j)} \cdot 2^t). \tag{66}$$

Obviously, $a[j] = \tau_0'^{(j)}$. Thus, we have

$$a[j] = \sum_{i=0}^{M-1} a_i[j] + \sum_{t=1}^{j} \tau_t'^{(j-t)} - \sum_{t=1}^{k-1} (\tau_t'^{(j)} \cdot 2^t). \tag{67}$$

Similarly, let $b = \sum_{i=0}^{S-1} b_i$, we can derive the following equation for S outputs

$$b[j] = \sum_{i=0}^{S-1} b_i[j] + \sum_{t=1}^{j} \tau_t''^{(j-t)} - \sum_{t=1}^{k-1} (\tau_t''^{(j)} \cdot 2^t), \tag{68}$$

when the balance property holds such that $k = \lceil \log(a) \rceil = \lceil \log(b) \rceil$. Taking Equation (71) and (72), for all $j \in [0, k)$, we have

$$\begin{aligned} b[j] - a[j] &= \sum_{i=0}^{S-1} b_i[j] - \sum_{i=0}^{M-1} a_i[j] \\ &+ \sum_{t=1}^{j} (\tau_t''^{(j-t)} - \tau_t'^{(j-t)}) - \sum_{t=1}^{k-1} \left((\tau_t''^{(j)} - \tau_t'^{(j)}) \cdot 2^t \right) \\ &= \sum_{i=0}^{S-1} b_i[j] - \sum_{i=0}^{M-1} a_i[j] + \sum_{t=1}^{j} \tau_t^{(j-t)} - \sum_{t=1}^{k-1} (\tau_t^{(j)} \cdot 2^t), \end{aligned} \tag{69}$$

where $\tau_t^{(j)} = \tau_t''^{(j)} - \tau_t'^{(j)}$. Obviously, the corrector value in [17] is a special case of Equation (73) under $k = 2$ by regarding $\tau_1^{(j-1)} = \tau_j$, $\tau_1^{(j)} = \tau_{j+1}$, and $\tau_t^{(j-t)} = \tau_t^{(j)} = 0$ when $t > 1$.

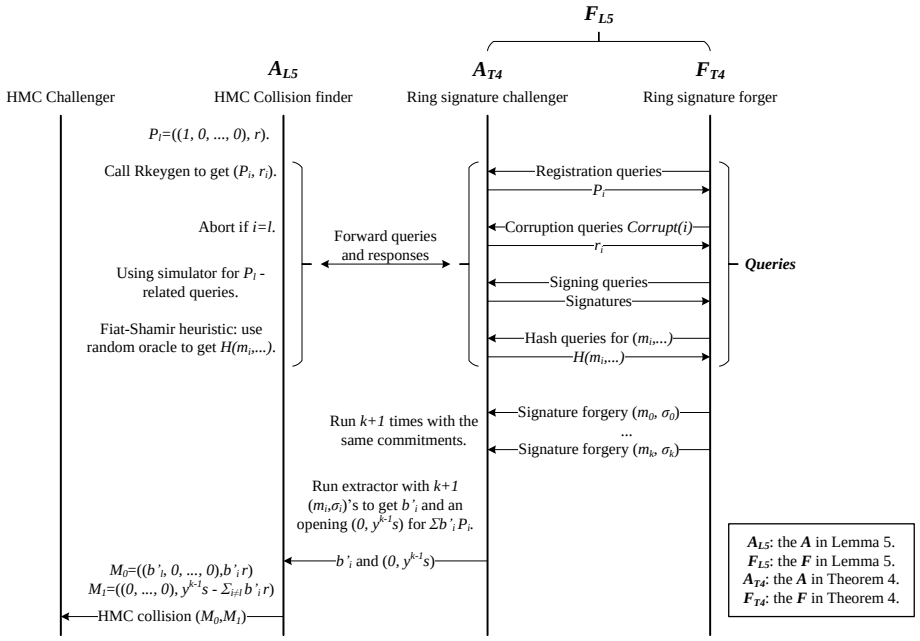

Fig. 8: Reduction from "forging a ring signature" to "finding an HMC collision" for ring signature unforgeability.

As $\tau_t''^{(j)}, \tau_t'^{(j)} \in \{0,1\}$, we have $\tau_t^{(j)} \in \{-1,0,1\}$, which narrows down the range of corrector values in [17] (but requires more corrector values).

D Security Reduction (Sketch) of Lemma 5 and Theorem 4

Lemma 5 and Theorem 4 reduces "forging a ring signature" to "finding an HMC collision" for ring signature unforgeability. The security game can be illustrated in Figure 8.

Oracle Simulation. The ring signature challenger $\mathcal{A}_{T4}$ will forward queries and responses between the HMC collision finder $\mathcal{A}_{L5}$ and the ring signature forger $\mathcal{F}_{T4}$. $\mathcal{A}_{L5}$ simulates the oracles as follows:

Registration query. Assume $\mathcal{F}_{T4}$ can only query N times ($N \geq 1$). $\mathcal{A}_{L5}$ randomly picks $l \leftarrow \{0,1,\cdots,N-1\}$. For index l, $\mathcal{A}_{L5}$ sets $P_l = \mathrm{Com}_{ck}(1,0,\cdots,0;\boldsymbol{r})$. For other indices $j \neq l$, $\mathcal{A}_{L5}$ calls the RKeyGen algorithm to generate a public/private key pair (P_j, r_j). Upon the $(i+1)$-th query, $\mathcal{A}_{L5}$ returns the corresponding public key P_i.

Corruption query. On input a public key P_i, $\mathcal{A}_{L5}$ aborts when $i = l$. Otherwise, $\mathcal{A}_{L5}$ returns the corresponding private key r_i.

Signing query. When $\mathcal{F}_{T4}$ queries to sign on message m_j with a signer P_i in a public key list $\boldsymbol{P}_i = \{P_j | j \in \boldsymbol{t}_j\}$ where $\boldsymbol{t}_j$ indicates the indices of the public keys in $\{P_0, \cdots, P_{N-1}\}$, i.e., $\boldsymbol{t}_i \subset \{1, \cdots, N-1\}$, $\mathcal{A}_{L5}$ processes as follows:

- If $i \neq l$, $\mathcal{A}_{L5}$ calls RSign algorithm directly to get the corresponding signature since he has r_i and programs the Hash function (random oracle) if needed.
- If $i = l$, $\mathcal{A}_{L5}$ runs the simulator in the proof of Theorem 2 (SHVZK prosperity) to get the signature and programs the Hash function (random oracle) so that $H(ck, m_i, \boldsymbol{P}_i, A, B, (E_j)_{j=0}^{k-1}) = x$.

Hash query. For queries with inputs that have already been programmed, $\mathcal{A}_{L5}$ returns the corresponding output. Otherwise, $\mathcal{A}_{L5}$ chooses x at random from the set $\mathcal{C} \backslash \{x_0, \cdots, x_{m-1}\}$ where x_i's are the outputs of the Hash function that have been programmed. The output of the Hash function for this input is programmed to x.

Signature Forgery. At a given point, $\mathcal{F}_{T4}$ finishes running and outputs a forgery (m_i, σ_i). Since P_l cannot be distinguished from other P_i's due to the hiding property of the HMC commiement scheme and $\mathcal{F}_{T4}$ can only make N times of registration queries to $\mathcal{A}_{L5}$, we have the signature is signed by the signer P_l with non-negligible probability[10], i.e., $b_l \neq 0$.

Output. After collecting $k+1$ signature forgeries with the same commitments (this can be done in polynomial time using the forking lemma), $\mathcal{A}_{T4}$ computes b_i''s and an opening $(\boldsymbol{0}, y^{k-1}\boldsymbol{s})$ to $\sum_{i=0}^{N-1} b_i' P_i$ by running the extractor in the proof of Theorem 2 ($(k+1)$-special soundness) and forwards b_i''s and $(\boldsymbol{0}, y^{k-1}\boldsymbol{s})$ to $\mathcal{A}_{L5}$. Accordingly, $\mathcal{A}_{L5}$ can find a collision for the HMC commitment scheme, $M_0 = ((b_l', 0, \cdots, 0), b_l'\boldsymbol{r})$ and $M_1 = (\boldsymbol{0}, y^{k-1}\boldsymbol{s} - \sum_{i \neq l} b_i' \boldsymbol{r}_i)$ since $(b_l', 0, \cdots, 0) \neq \boldsymbol{0}$ ($\boldsymbol{r}_i$'s are the private keys of other users in the ring).

E An Alternative Approach to Avoid the Range Proofs of Corrector Values

We present an alternative approach to reduce the cost of corrector values in [17] by narrowing down the range of corrector values to $\{-1, 0, 1\}$. Based on the discussion in Section 9, we can avoid the cost of UMC in range proofs.

Considering M inputs $(a_i)_{i=0}^{M-1}$. Let $a = \sum_{i=0}^{M-1} a_i$, $k = \lceil \log(a) \rceil$, and $a_i[j]$ be the j-th bit of a_i where $0 \leq j < k$. First, taking $\sum_{i=0}^{M-1} a_i[0]$ and writing it as the binary representation $(\tau_0'^{(0)}, \cdots, \tau_{k-1}'^{(0)})$ such that $\sum_{i=0}^{M-1} a_i[0] = \sum_{t=0}^{k-1} (\tau_t'^{(0)} \cdot 2^t)$, we have $a[0] = \tau_0'^{(0)}$. Furthermore, taking $\tau_1'^{(0)}$ and $\sum_{i=0}^{M-1} a_i[1]$, we can write the sum of them as the binary representation $(\tau_0'^{(1)}, \cdots, \tau_{k-1}'^{(1)})$ and derive $a[1] = \tau_0'^{(1)}$. In this way, we can observe that in order to derive $a[j]$ (the j-th bit of $a = \sum_{i=0}^{M-1} a_i$), the j-th element consists of two parts: $\sum_{i=0}^{M-1} a_i[j]$ and $\sum_{t=1}^{j} \tau_t'^{(j-t)}$ (the latter one is the sum of all carries from $a[j-1], \cdots a[0]$). Thus,

[10] Other signers may also be involved such that $b_i \neq 0$ for some $i \neq l$.

for all $j \in [0, k)$, writing $\sum_{i=0}^{M-1} a_i[j] + \sum_{t=1}^{j} \tau_t'^{(j-t)}$ as the binary representation $(\tau_0'^{(j)}, \cdots, \tau_{k-1}'^{(j)})$, we have

$$\sum_{i=0}^{M-1} a_i[j] + \sum_{t=1}^{j} \tau_t'^{(j-t)} = \sum_{t=0}^{k-1} (\tau_t'^{(j)} \cdot 2^t). \tag{70}$$

Obviously, $a[j] = \tau_0'^{(j)}$. Thus, we have

$$a[j] = \sum_{i=0}^{M-1} a_i[j] + \sum_{t=1}^{j} \tau_t'^{(j-t)} - \sum_{t=1}^{k-1} (\tau_t'^{(j)} \cdot 2^t). \tag{71}$$

Similarly, let $b = \sum_{i=0}^{S-1} b_i$, we can derive the following equation for S outputs

$$b[j] = \sum_{i=0}^{S-1} b_i[j] + \sum_{t=1}^{j} \tau_t''^{(j-t)} - \sum_{t=1}^{k-1} (\tau_t''^{(j)} \cdot 2^t), \tag{72}$$

when the balance property holds such that $k = \lceil \log(a) \rceil = \lceil \log(b) \rceil$. Taking Equation (71) and (72), for all $j \in [0, k)$, we have

$$\begin{aligned} b[j] - a[j] &= \sum_{i=0}^{S-1} b_i[j] - \sum_{i=0}^{M-1} a_i[j] \\ &+ \sum_{t=1}^{j} (\tau_t''^{(j-t)} - \tau_t'^{(j-t)}) - \sum_{t=1}^{k-1} \left((\tau_t''^{(j)} - \tau_t'^{(j)}) \cdot 2^t \right) \\ &= \sum_{i=0}^{S-1} b_i[j] - \sum_{i=0}^{M-1} a_i[j] + \sum_{t=1}^{j} \tau_t^{(j-t)} - \sum_{t=1}^{k-1} (\tau_t^{(j)} \cdot 2^t), \end{aligned} \tag{73}$$

where $\tau_t^{(j)} = \tau_t''^{(j)} - \tau_t'^{(j)}$. Obviously, the corrector value in [17] is a special case of Equation (73) under $k = 2$ by regarding $\tau_1^{(j-1)} = \tau_j$, $\tau_1^{(j)} = \tau_{j+1}$, and $\tau_t^{(j-t)} = \tau_t^{(j)} = 0$ when $t > 1$.

As $\tau_t''^{(j)}, \tau_t'^{(j)} \in \{0, 1\}$, we have $\tau_t^{(j)} \in \{-1, 0, 1\}$, which narrows down the range of corrector values in [17] (but requires more corrector values).

F Reducing the Cost of Range Proofs in MatRiCT

Esgin points out that running a full range proof (as in [14]) is unnecessary since $\tau_0 - 2\tau_1, \cdots, \tau_k - 2\tau_{k+1}$ have already been committed in $C = \mathrm{Com}(\tau_0 - 2\tau_1, \cdots, \tau_k - 2\tau_{k+1})$. Consider one correct value τ_j that falls in the range $[-(M-1), S-1]$. A prover shifts it to $\tau_j' = \tau_j + (M-1)$ and proves $\tau_j' \in [0, S+M-2]$. Specifically, the prover writes τ_j' in the binary representation, $(\tau_{j,0}', \cdots, \tau_{j,l-1}') \leftarrow \mathrm{Bits}(\tau_j')$, where $l = \log(S+M-1)$. By running a binary proof (can be aggregated in the binary proof part of output accounts), the prover convinces the verifier

that $\tau'_{j,i}$'s are bits after sending $f_{j,i} = x \cdot \tau'_{j,i} + a_{j,i}$ for all $j \in [1, k)$'s and $i \in [0, l)$'s with a challenge x and some masking values $a_{j,i}$'s. Additionally, the verifier reconstructs the masked τ'_j, f_j, with

$$f_j = \sum_{i=0}^{l-1} 2^i f_{j,i} = x \cdot \tau'_j + \sum_{i=0}^{l-1} 2^i a_{j,i}. \tag{74}$$

With f_j's as the masked corrector values, the verifier further checks

$$\mathrm{Com}(f_0 - 2f_1, \cdots, f_k - 2f_{k+1}) = xC + A_{sum}, \tag{75}$$

where $A_{sum} = \mathrm{Com}(\sum_{i=0}^{l-1} 2^i a_{0,i} - 2\sum_{i=0}^{l-1} 2^i a_{1,i}, \cdots, \sum_{i=0}^{l-1} 2^i a_{k,i} - 2\sum_{i=0}^{l-1} 2^i a_{k+1,i})$ and $f_0 = f_k = 0$. Besides, A_{sum} and Equation (75) can also be aggregated into the binary proof verification, the prover only needs to include a $(k-1)l$-size vector, $(f_{j,i})_{j=1,i=0}^{k-1,l-1}$, in the proof.