RingSLIP: Ring Signatures from the Lattice Isomorphism Problem

Callum London, Daniel Gardham, and Constantin Cătălin Drăgan
Surrey Center for Cyber Security, University of Surrey, UK
{c.london, daniel.gardham, c.dragan}@surrey.ac.uk

Keywords: Ring Signature, Lattice Isomorphism Problem, Linkability, Online/Offline Signatures

Abstract: Ring signatures provide authentication over messages, whilst providing anonymity amongst a set of signer-defined public keys. They see active use in cryptocurrencies, e-voting and concurrent signature domains. However, post-quantum constructions typically rely on lattices, specifically utilising the Learning with Errors (LWE) and Short-Integer-Solution (SIS) problems, which cause inefficiencies when compared with classical constructions.

One promising route to circumvent the inherent challenges of these underlying assumptions is the Lattice Isomorphism Problem (LIP), which underpins the HAWK signature scheme by Ducas et. al, currently a second round candidate in the NIST standardisation project Post Quantum Cryptography: Additional Digital Signature Schemes. It offers significant performance improvements over standard lattice assumptions due to its improved decoding, however, the only known construction of a ring signature from LIP has been shown to not satisfy linkability nor correctness.

In this paper we propose RingSLIP, a secure *linkable* ring signature based on LIP, utilising the HAWK signature. The resulting ring signature is logarithmic in the number of ring members, and concretely has size 46KB when targeting 128 bits of security for 4096 ring members, which is competitive with other lattice-based schemes. Furthermore, we observe that our construction also benefits from online/offline computation, resulting in a signature with online signing and verification only requiring 8.54×10^4 and 1.48×10^5 CPU cycles respectively, compared to 1.35×10^{11} without these optimisations.

1 Introduction

Ring signatures provide crucial privacy properties in real-world applications ranging from anonymous cryptocurrency transactions, whistleblowing systems, secure voting protocols, to confidential authentication schemes. Standardised in ISO/IEC 20008, they allow a signer to create the signature that verifies against a set of public keys. Differentiating them from other anonymous signatures (e.g. group signatures), this ring is chosen by the signer, and requires no enrollment. Concretely, ring signatures asee use in *anonymous transactions* (e.g., Monero, Bytecoin), *anonymous communication* (e.g., for ring size two in designated-verifier signatures (Jakobsson et al., 1996), but also find practical uses in concurrent signatures (Chen et al., 2004)), *electronic voting* (Tsang and Wei, 2005), *medical data sharing* (Yang et al., 2025) and *vehicle to grid (V2G) communications* (Wang et al., 2026).

The two core security properties of any ring signature are *anonymity* and *unforgeability*. The aim of unforgeability is to ensure a signature is valid only if the issuer is a member of the ring. In essence, this is modeled by requiring the adversary to produce forgeries where all ring members are honest (the adversary doesn't know their secret keys). Based on whether the adversary can corrupt parties, we distinuish (Bender et al., 2009): *chosen-subring* where no corruption is allowed, and *insider-corruption* where other parties are corrupted (but not the ring members where the forgery is produced).

On the other hand, the anonymity property ensures that an adversary cannot distinguish which ring member created a signature. We distinguish two main types (Bender et al., 2009) based on the control the adversary has over the parties in the ring: *basic* (it knows only the public keys) and *adversary-chosen* (it can include adversary produced keys). Both variants rely on the existence of at least two honest parties in the ring (one that issues the signature and another), and the adversary is made to distinguish which one issued the signature. A stronger variant of anonymity, called *against full key exposure*, looks at having two

honest parties only during signing and only one honest party after.

The domains of cryptocurrency and e-voting have also motivated a richer feature for ring signatures: *linkability* (J. K. Liu and Wong, 2005), where signatures from the same issuer can be linked. This is often coupled with *non-frameability* where an adversary can't forge the existence of a link. Within linkable ring signatures, there are two flavours distinguished on whether the linking algorithm can be publicly executed (called here *non-selective*), or requires the original signing key, which is known as *selective linkability* (Fiore et al., 2022). This allows the signer to selectively disclose which, if any, of its signatures are linked through either: *implicit linking* where signers voluntarily link signatures using the same token (i.e., pseudonym) or *explicit linking* where only the signers can prove at any time two signatures are linked. An additional variant of explicit linkability considers *autonomous* where the witness for the proof is independent of the signing key. Explicit linkability is motivated by applications whereby a user may want to remain anonymous at time of signing, but later reveal a series of linked signature, e.g., linking at later stage to receive discounts in smart meters or connected vehicles. The property *linkable anonymity* ensures that signatures that are not intended to be linked (either selective or otherwise) cannot be associated, even if produced by the same signer.

In recent years, there has been increasing effort to realise practical constructions of ring signatures from post-quantum assumptions. Owing to its relative maturity compared to other post-quantum settings, most of these schemes are based on lattices, and in particular, from the LWE and SIS problems, e.g. Beullens et al. (2020), Z. Liu et al. (2019), Lu et al. (2019), and Zhang et al. (2018). However, signatures from these assumptions are typically unsatisfactory due to inherent loss in the sampling process, and this is carried over to ring signatures where state of the art schemes still reach 30KB for 32 ring members (compared to $\leqslant 2KB$ for group-based constructions, e.g. Hui and Chau (2024)). The Lattice Isomorphism Problem is a promising alternative for lattice-based signatures that bypasses such challenges, and is the basis of HAWK (Ducas et al., 2022), a finalist for the NIST standardisation project Post Quantum Cryptography: Additional Digital Signature Schemes process. The only known construction of a ring signature from LIP is presented by Khuc et al. (2023), however the linkable anonymity property has subsequently been found to be insecure (Budroni, Chi-Domínguez, and Franch, 2025b), and in fact does not satisfy correctness (Iribarnegaray et al., 2026).

While there are many blueprints and generic approaches to constructing ring signatures, all rely on application of zero-knowledge proofs, which can prove to be computationally intensive. A line of work, originating from Even et al. (1989), solves this problem by splitting the signing phase into an *offline* and *online* phase. The computationally heavy components can be computed in advance in a non-time sensitive environment, allowing for a comparatively short finalising step in the online phase. In the context of ring signatures, J. K. Liu et al. (2009) have proposed such a scheme based on the RSA assumption.

This leaves open the challenge of designing an

(offline/online) linkable ring signature based on LIP.

Contributions. We present three contributions in this paper:

1. We propose RingSLIP, the only *secure logarithmic linkable ring signature from LIP.* We give a construction based on efficient LIP signatures and zero-knowledge proofs, and prove that it satisfies *explicit linkability*, *anonymity*, and *non-frameability*, whilst achieving complexity logarithmic in the ring size. For linking, we employ a different strategy from Khuc et al. (2023), and thus are able to achieve linkability that is not susceptible to the attack by Budroni, Chi-Domínguez, and Franch (2025b).
2. We present *parameters and efficiency analysis.* When instantiated at standard NIST security levels, our scheme is comparable with state-of-the-art schemes from both LWE/SIS and LIP.
3. We extend our scheme to the *offline/online setting*. Due to the nature of our construction we can move most of the computation to an offline phase, while keeping for the online phase only the efficient HAWK signature. This results in an online signing/verification phase only requiring $56.54 \times 10^4 / 1.96 \times 10^5$ CPU cycles respectively.

1.1 Related Work

Zhang et al. (2018) proposed a logarithmic-size construction for ring signatures over ideal lattices using lattice-based signatures, however the resulting signature size is large and does not consider any notion of linkability. Lu et al. (2019) similarly proposed such a scheme by adapting the generic ring signature framework of Rivest et al. (2001) to the lattice setting. To improve suitability for cryptocurrency applications, Z. Liu et al. (2019) introduced a scheme incorporating stealth addresses, designed to handle

practical scenarios under adversarially chosen-key attacks.

Post-quantum state-of-the-art linkable ring signatures typically rely on module variations of LWE and SIS. Proposed by Beullens et al. (2020), Falafl achieves logarithmic complexity in the number of ring members, resulting in a signature size of 30KB for 32 members ($N = 32$) when targetting NIST level 1. They are also able to meet non-selective linkability, i.e. the linking algorithm is publically executable. DualRing-LB (Yuen et al., 2021) proposed a generic method to design non-linkable ring signatures from ideal lattices using a technique based on the Schnorr identification scheme. Their signatures obtain 6KB respectively for a ring of the same size, however these increase significantly as the size of the ring grows, reaching 106.6KB respectively for $N = 2^{12}$, as they do not enjoy the logarithmic advantage of other schemes such as Falafl (Beullens et al., 2020). Similarly, MatRiCT (Esgin et al., 2019) (also non-linkable) achieves 19KB respectively for a ring of size $N = 32$, growing to 59KB for $N = 2^{12}$.

As mentioned previously, Khuc et al. (2023) proposes linkable post-quantum ring signatures under the lattice-isomorphism problem (LIP), where Merkle Trees OR proof is combined with a signature of knowledge. However, Budroni, Chi-Domínguez, and Franch (2025a) have shown that Khuc et al. (2023) fails to meet linkable anonymity due to an implicit repetition of the secret isomorphism in the creation of both the signature and the tag. Iribarnegaray et al. (2026) further critique this work by showing the correctness and linkability property also do not hold. Khuc et al. (2023) do also present an unlinkable scheme, that only satisfies *unforgeability* and *anonymity* but does preserve the small keys and logarithmic signature size.

Xue et al. (2024) provide a general framework to build efficient linkable post-quantum ring signatures based on collision-resistant hash functions. Interestingly, this work also explores online/offline exploits to improve signature latency however this is for verification only, whereas our protocol benefits on both signing and verification phases.

We present a summary of selected related lattice-based schemes in Table 1, and compared to the contributions of this paper, for a selected ring size of 2^{12}.

Table 1: Comparison with lattice-based ring signatures for ring size $N = 2^{12}$. No amortisation has been included in these calculations.

Scheme	Sig. Size.	Hardness Assumptions	Linkability
RAPTOR (Lu et al., 2019)	5161KB	NTRU	Non-selective
MatRiCT (Esgin et al., 2019)	59KB	MLWE/MSIS	–
Falafl (Beullens et al., 2020)	35KB	MLWE/MSIS	Non-selective
DualRing-LB (Yuen et al., 2021)	106.6KB	MLWE/MSIS	–
Khuc et al. (2023)	45KB	LIP	Insecure
This Work	46KB	LIP	Selective

2 Preliminaries

We introduce basic lattice concepts and the lattice isomorphism problem.

Notation. For a lattice $S \subset \mathbb{R}^n$, we denote the set of all lattices isomorphic to S by

$$[S] := \{L \in \mathbb{R}^n : \exists O \in O_n(\mathbb{R}), L = O \cdot S\}.$$

We denote by $\otimes$ the Kronecker product and by G_S the gram matrix of B_S, given by ${B_S}^T B_S$. We denote by λ_k the kth successive minima of S, which is the minimum length l for which a set of k linearly independent lattice vectors $\{x_1, \ldots, x_k\}$ in S exists with $\|x_i\| \leqslant l \ \forall i \in [k]$. The Shortest Vector Problem (SVP) asks to find a vector of length λ_1 for a given lattice S, i.e. to find the shortest non-zero lattice vector.

Definition 1 (Gaussian Function and Distributions)**.** We consider $\rho_{r,\boldsymbol{c}} : \mathbb{R}^n \to \mathbb{R}$ the n-dimensional Gaussian function of width r and center $\boldsymbol{c}$. We obtain the n-dimensional spherical Gaussian distribution of width r by normalising the Gaussian function over $\mathbb{R}^n$ and denote the result by $\nu_{r,\boldsymbol{c}}$ where

$$\rho_{r,\boldsymbol{c}}(\boldsymbol{x}) := e^{-\pi\|\boldsymbol{x}-\boldsymbol{c}\|^2/r^2} \text{ and } \nu_{r,\boldsymbol{c}}(\boldsymbol{x}) := \rho_{r,\boldsymbol{c}}/r^n.$$

Given a discrete set $S \subset \mathbb{R}^n$, we obtain the n-dimensional discrete Gaussian distribution of width r supported by S via normalising the Gaussian function over S and we denote the result by $\mathcal{D}_{S,r,\boldsymbol{c}}$ as follows.

$$\mathcal{D}_{S,r,\boldsymbol{c}}(\boldsymbol{x}) := \begin{cases} \rho_{r,\boldsymbol{c}}(\boldsymbol{x})/\rho_{r,\boldsymbol{c}}(S), \text{ for } \boldsymbol{x} \in S \\ 0, \text{ otherwise} \end{cases}$$

Where a center $\boldsymbol{c}$ is omitted from any of the above notation we take $\boldsymbol{c} = \boldsymbol{0}$.

Definition 2 (Smoothing Parameter, (Micciancio and Regev, 2004, Definition 3.1))**.** Given a lattice L and a positive real number $0 < \varepsilon < 1$, the smoothing parameter $\eta_\varepsilon(L)$ is defined as the smallest real number $s > 0$ such that

$$\sum_{\boldsymbol{y} \in L^* \setminus \{\boldsymbol{0}\}} \rho_{1/s}(\boldsymbol{y}) \leqslant \varepsilon$$

Equivalently, $\eta_\varepsilon(L)$ is the smallest real number $s > 0$ such that for a random variable X distributed according to $\mathcal{D}_{L^*,1/s}$ we have $\Pr[X = \boldsymbol{0}] \geqslant 1/(1+\varepsilon)$.

2.1 Lattice Isomorphism Problem

The Lattice Isomorphism Problem formally considers two bases B_1, B_2 generating $\mathbb{R}^n$ and requires finding

two separate components that together form a map from one basis to the other. These components are an orthonormal matrix $O \in \mathcal{O}_n(\mathbb{R})$ and a unimodular matrix $U \in \mathcal{GL}_n(\mathbb{Z})$.

Definition 3 (LIP, Basis Perspective (Ducas and van Woerden, 2022))**.** Let $B_1, B_2 \subset \mathbb{R}^n$ be any two bases of $\mathbb{R}^n$. Then a solution to LIP_{B_1,B_2} is considered to be some pair $(O,U) \in \mathcal{O}_n(\mathbb{R}) \times \mathcal{GL}_n(\mathbb{Z})$ satisfying $B_2 = OB_1U$.

LIP is commonly expressed in one of the following two simpler forms.

Definition 4 (LIP, Lattice Perspective (Ducas and van Woerden, 2022))**.** Let $L_1, L_2 \subset \mathbb{R}^n$ be isomorphic n-dimensional lattices. Then a solution to LIP_{L_1,L_2} is considered to be some $O \in \mathcal{O}_n(\mathbb{R})$ satisfying $L_2 = O \cdot L_1$.

Definition 5 (LIP, Quadratic Form Perspective (Ducas and van Woerden, 2022))**.** Let $Q_1, Q_2 \in S_n^{>0}$ be two equivalent quadratic forms. Then a solution to LIP_{Q_1,Q_2} is considered to be some $U \in \mathcal{GL}_n(\mathbb{Z})$ satisfying $Q_2 = U^T Q_1 U$.

Both definitions complement each other in that each isolates a different component of the full basis transformation $B_2 = OB_1U$. This is achieved by considering both challenge bases within a quotient space: $\mathcal{GL}_n(\mathbb{R})/\mathcal{GL}_n(\mathbb{Z})$ for lattices and $\mathcal{O}_n(\mathbb{R})\backslash\mathcal{GL}_n(\mathbb{R})$ for quadratic forms.

Remark 1. For a particular basis B_L of some lattice L, the link with the associated quadratic form is given by the gram matrix $G_L := {B_L}^T B_L$. This matrix is symmetric and may be identified with the coefficients of a quadratic form as in the following example (here $n = 2$ for simplicity).

$$\begin{bmatrix} a & b \\ b & c \end{bmatrix} \leftrightarrow a \cdot x^2 + 2b \cdot xy + c \cdot y^2$$

From the defining property $O^T O = I$ for $O \in \mathcal{O}_n(\mathbb{R})$, it can be easily seen that isomorphisms of a lattice preserve the gram matrix:

$$\begin{aligned} (OB_L)^T (OB_L) &= ({B_L}^T O^T)(OB_L) \\ &= {B_L}^T (O^T O) B_L \\ &= {B_L}^T B_L. \end{aligned}$$

Hence the derived quadratic form varies only under a unimodular transform of the basis.

Decisional Variants. LIP has a decision variant dLIP which simply asks whether a given lattice L' belongs to particular class $[L]$, however the distinguish variant Δ-LIP is more commonly seen in cryptanalysis. In the distinguish variant a lattice is sampled from one of two possible classes $[L_0], [L_1]$ and the correct class must be identified. The difference from dLIP is that the domain of L' is narrowed, with the guarantee that exactly one of the classes is correct.

2.2 Algorithms

Sampling random isomorphisms on a real space $\mathbb{R}^n$ is a well-understood problem, with the methodology for doing so provided by Mezzadri (2007).

Lemma 1 (Isomorphism Sampling (Mezzadri, 2007, Theorem 4))**.** *There exists a* PPT *algorithm* IsoSample *that given a parameter n outputs a sample orthonormal matrix $O \in \mathcal{O}_n(\mathbb{R})$ uniformly at random.*

The generation of a unimodular component for a LIP instance may be performed via resampling the basis B_L of a given lattice L. This basis resampling is equivalent to sampling a random quadratic form from the equivalence class associated with L, a process for which Ducas and van Woerden (2022) provide a suitable algorithm.

Lemma 2 (Lattice Basis Resampling (Ducas and van Woerden, 2022, Algorithm 1))**.** *There exists a* PPT *algorithm* BasisSample *that given a basis B_L and a parameter $s > \max(\lambda_n(L), ||\widetilde{B_L}|| \cdot \sqrt{\ln(2n+4)/\pi})$ outputs a sample basis B'_L from $\mathcal{D}_s(L)$ together with a unimodular transform U satisfying $B'_L = B_L U$.*

We note that algorithm is independent of the choice of basis and therefore we may abuse notation and denote a call to the algorithm as $\mathsf{BasisSample}(L)$.

As part of BasisSample, and to instantiate lattice-based schemes in general, it is necessary to be able to sample from certain discrete Gaussian distributions. We rely on an exact sampling algorithm provided by Brakerski et al. (2013).

Lemma 3 (Discrete Gaussian Sampling (Brakerski et al., 2013))**.** *Consider an n-dimensional lattice L generated by some basis B. Denote by $||\widetilde{B}||$ the length of the longest vector in the Gram-Schmidt orthogonalisation of B. Then for $\boldsymbol{c} \in \mathbb{R}^n$ and $r \geqslant ||\widetilde{B}|| \sqrt{\ln(2n+4)/\pi}$ there exists a* PPT *algorithm* $\mathsf{DiscreteSample}_{L,r,\boldsymbol{c}}$ *that outputs a sample distributed according to $\mathcal{D}_{L,r,\boldsymbol{c}}$.*

2.3 Ring Signatures

In this section we formally define ring signatures and associated properties. As mentioned previously, linkability is a property that allows an observer to identify whether or not a set of signatures were produced by a single member of the ring. In such schemes, it is typical for this linking process to be public and hence any signatures of a given user can always be linked. However, Fiore et al. (2022) propose a variation of

this definition to allow for user-controlled linkability, that is, the signer can decide which, if any, of their signatures they want to be linked. This gives rise to two notions of user-controlled linkability:

Implicit Linkability: Each signature is created under an anonymous public key nym called the pseudonym, and a user can create multiple linked signatures under a single pseudonym. Security in this setting ensures it is difficult to produce signatures against a pre-existing pseudonym created by another user.

Explicit Linkability: A user can anonymously create a proof that a pair of signatures were both signed by them, even if they used a different pseudonym for each signature.

We adopt a definition for linkable ring signatures inspired by explicit linkability of Fiore et al., 2022, opting instead to allow for a dynamic linking secrets, and suppress the notion of *scope*, which has arisen from linkable DAA used in TPM attestation. The scope enables two signatures with the same pseudonym to be trivially linked, capturing implicit linkability. Such a definition allows the usage of different linking secrets thus ensuring that past signatures cannot be linked even if the signer is corrupted. Our modifications allow for simplified security definitions and ultimately a more concise and efficient construction, whilst retaining core linkable security properties. Whilst this paper considers mildly relaxed definitions, our scheme is readily adapted to capture the notion of scope, which we discuss further in Remark 4.

Definition 6 (RSSL - Ring Signature with Selective Linkability)**.** A ring signature with selective linkability is a PPT tuple of algorithms $(\mathsf{KGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Link}, \mathsf{VfLink})$ where

$\mathsf{KGen}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$**:** Generates a private-public key pair $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(m, R, \mathsf{sk}, \mathsf{ls}) \to (\sigma, \mathsf{nym}, \mathsf{ls})$**:** For a valid key pair $(\mathsf{sk}, \mathsf{pk}_i)$ output by KGen with $1 \leqslant i \leqslant n$, generates a pseudonym nym from pk_i under which a signature σ is created on the message m with respect to the ring $R := (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$. The linking secret ls is also updated with a token τ corresponding to the pseudonym nym. It is assumed that $n \geq 2$ and that the pk_i are distinct.

$\mathsf{Verify}(m, R, \sigma, \mathsf{nym}) \to \{0,1\}$**:** Decides whether the message-signature pair (m, σ) under the pseudonym nym is valid (returns 1) or invalid (returns 0) with respect to the ring R.

$\mathsf{Link}(\mathsf{ls}, \boldsymbol{\Sigma}) \to \pi_l$**:** If successful, produces a proof π_l linking all signatures $\Sigma_j \in \boldsymbol{\Sigma}$ to a common signer in possession of ls. Otherwise outputs an error $\perp$.

$\mathsf{VfLink}(\boldsymbol{\Sigma}, \pi_l) \to \{0,1\}$**:** Decides whether the linking proof π_l is a valid (returns 1) or invalid (returns 0) proof that all signature tuples $\Sigma_j \in \boldsymbol{\Sigma}$ were produced by a common signer.

Correctness. Correctness for ring signatures with linkability has two requirements. The first requirement is verification correctness, which is equivalent to the standard correctness definition for general ring signatures. In addition to this, it must also satisfy linking correctness, which captures the notion of correctness for the linking algorithms of the scheme, namely Link and VfLink.

Definition 7 (Verification Correctness)**.** We say that an RSSL scheme satisfies verification correctness if there exists a negligible function $\mathsf{negl}(\lambda)$ such that for any message m, security parameter λ, and set of key pairs $\{(\mathsf{sk}_i, \mathsf{pk}_i)\}_{i=1}^n \leftarrow \mathsf{KGen}(1^\lambda)$ with linking secrets $\{\mathsf{ls}_i\}_{i=1}^n$ we have for each $i \in [n]$ that $\Pr[\mathsf{Verify}(m, R, \mathsf{Sign}(m, R, \mathsf{sk}_i, \mathsf{ls}_i) \setminus \mathsf{ls}_i) = 1] \geqslant 1 - \mathsf{negl}(\lambda)$ where $R := (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$.

Definition 8 (Linking Correctness)**.** We say that a RSSL scheme satisfies linking correctness if there exists a negligible function $\mathsf{negl}(\lambda)$ such that for any messages $\{m_j\}_{j=1}^k$, security parameter λ, set of key pairs $\{(\mathsf{sk}_i, \mathsf{pk}_i)\}_{i=1}^n \leftarrow \mathsf{KGen}(1^\lambda)$ with linking secrets $\{\mathsf{ls}_i\}_{i=1}^n$, rings $\{R_j\}_{j=1}^k \subseteq (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$, and $i \in [n]$ we have that $\Pr[\mathsf{VfLink}(\boldsymbol{\Sigma}, \mathsf{Link}(\mathsf{ls}_i, \boldsymbol{\Sigma})) = 1] \geqslant 1 - \mathsf{negl}(\lambda)$ where $\boldsymbol{\Sigma} = \{\Sigma_j := (m_j, R_j, \mathsf{Sign}(m_j, R_j, \mathsf{sk}_i, \mathsf{ls}_i) \setminus \mathsf{ls}_i)\}_{j=1}^k$.

All ring signatures satisfy two key security properties, namely anonymity and unforgeability. We also state security properties to cover linking functionality of our scheme. The games are presented in Figure 1a and corresponding oracles in Figure 1b.

Anonymity. Our definition captures the strongest notion of anonymity: security against full key exposure. Intuitively, this property states that for a ring R containing at least two honest members, an adversary cannot tell which member produced a given signature σ on some chosen message m. The strongest form of anonymity is anonymity under full key exposure, which holds even when the adversary knows every user's secret key. For our definition, while we can allow the adversary to know all signing keys we must disallow the adversary from learning the linking secret of either of the two challenge members to prevent trivial wins. The formal definition is given in Definition 9.

Definition 9 (Anonymity under full signing-key exposure)**.** We say that an RSSL scheme

$(\mathsf{KGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Link}, \mathsf{VfLink})$ is anonymous under full signing-key exposure if any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ wins the anonymity game in Figure 1a with advantage negligbly close to $\frac{1}{2}$.

$$\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{A},RS}(\lambda) \;=\; \left|\Pr\left[\mathsf{Exp}^{\mathsf{anon}}_{\mathcal{A},RS} = 1\right] - \frac{1}{2}\right|$$

Unforgeability. Our unforgebaility notion is based on security against *insider corruption*. This property states that an adversary cannot create a forgery (R, m, σ) that is accepted by an honest verifier unless either, *i)* R contains a public key chosen by the adversary, or *ii)* a user in R has previously signed on m. The second condition is crucial to prevent the adversary from trivially winning the experiment by simply re-randomising a signature obtained from an honest member of R. We adopt the strongest form of unforgeability which allows for insider corruption, i.e. the adversary can corrupt any number of ring members to reveal their secret keys. Furthermore, the adversary is permitted to add additional public keys to the ring (which may be malformed). The formal definition is stated in Definition 10.

Definition 10 (Unforgeability w.r.t. insider corruption)**.** We say that an RSSL scheme $(\mathsf{KGen}, \mathsf{Sign}, \mathsf{Verify})$ is unforgeable w.r.t. insider corruption if any PPT adversary $\mathcal{A}$ wins the unforgeability game in Figure 1a with negligible probability. That is, the following advantage is negligible in λ.

$$\mathsf{Adv}^{\mathsf{unf}}_{\mathcal{A},RS}(\lambda) \;=\; \Pr\left[\mathsf{Exp}^{\mathsf{unf}}_{\mathcal{A},RS} = 1\right]$$

Linking Security. In addition to the standard security properties for ring signatures, we consider two further security properties related to the linking functions of an RSSL scheme. Link unforgeability corresponds directly to explicit linking, requiring that a valid linking proof for a pair of signatures can only be created by a signer that has produced both of them. Non-frameability ensures that a signature cannot be linked implicitly to a user that did not create it. More precisely, an adversary should require both the secret key and the linking secret that is associated with a particular pseudonym in order to forge signatures under that pseudonym. The formal definitions for both properties are as follows.

Definition 11 (Link Unforgeability)**.** We say that a RSSL scheme $(\mathsf{KGen}, \mathsf{Sign}, \mathsf{Verify})$ is link-unforgeable if any PPT adversary $\mathcal{A}$ wins the link-unforgeability game in Figure 1a with negligible probability. That is, the following advantage is negligible in λ.

$$\mathsf{Adv}^{\mathsf{l-unf}}_{\mathcal{A},RS}(\lambda) \;=\; \Pr\left[\mathsf{Exp}^{\mathsf{l-unf}}_{\mathcal{A},RS} = 1\right]$$

Definition 12 (Non-Frameability)**.** We say that a RSSL scheme $(\mathsf{KGen}, \mathsf{Sign}, \mathsf{Verify})$ is non-frameable if any PPT adversary $\mathcal{A}$ wins the non-frameability game in Figure 1a with negligible probability. That is, the following advantage is negligible in λ.

$$\mathsf{Adv}^{\mathsf{frame}}_{\mathcal{A},RS}(\lambda) \;=\; \Pr\left[\mathsf{Exp}^{\mathsf{frame}}_{\mathcal{A},RS} = 1\right]$$

Remark 2. The linking secret for a given pseudonym only needs to be unique with respect to a particular public key. It may be possible for two distinct public keys to map to a common pseudonym under different linking secrets. In this case non-frameability may still hold, provided that it is difficult to find such a collision.

3 Zero-Knowledge Protocol for Proof of Isomorphisms

Our construction will need to utilise zero-knowledge proofs for isomorphisms between lattices. We formally define and construct these proofs in this section.

For a given relation R, a language is defined as $L_R := \{x : \exists w \ (x, w) \in R\}$. A zero-knowledge proof system allows a prover to prove knowledge of a witness w to the relation R, without leaking any information about it. Formally, it is defined below.

Definition 13 (Non-Interactive Zero-Knowledge Proof System)**.** A NIZK proof system for a relation R is a tuple of algorithms $(\mathsf{Setup}, \mathsf{P}, \mathsf{V}, \mathsf{SimSetup}, \mathsf{Sim})$ defined as follows.

$\mathsf{Setup}(\lambda)$**:** Samples a common reference string ρ given the security parameter λ.

$\mathsf{P}(\rho, x, w)$**:** The prover creates a proof π of a statement x for a witness w, with respect to a common reference string ρ.

$\mathsf{V}(\rho, x, \pi)$**:** The verifier outputs 1 if π is a valid proof of the statement x with respect to the common reference string ρ, and 0 otherwise.

$\mathsf{SimSetup}(\lambda)$**:** Samples a common reference string ρ with a trapdoor τ given the security parameter λ.

$\mathsf{Sim}(\rho', \tau)$**:** Creates a simulated proof π' for statement x with respect to common reference string ρ'.

A NIZK proof system is *correct* if for any $\rho \leftarrow\!\!\$\ \mathsf{Setup}(\lambda)$ and $(x, w) \in R$ it follows that with overwhelming probability we have $1 \leftarrow \mathsf{V}(\rho, x, \mathsf{P}(\rho, x, w))$.

Definition 14 (Special Soundness)**.** We say that a NIZK proof system has the special soundness property if there exists a PPT algorithm Ext such that $w \leftarrow \mathsf{Ext}(\rho, x, \pi_1, \pi_2)$ where $\rho \leftarrow \mathsf{NIZK.Setup}(\lambda)$ and $\pi_i \leftarrow \mathsf{NIZK.P}(\rho, x, w)$ for some statement x.

$\mathsf{Exp}^{\text{anon}}_{\mathcal{A},RS}(1^\lambda)$ with $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

1 : **for** $i \in [N]$ **do** $(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$
2 : $(\mathsf{st}, id_0, id_1) \leftarrow \mathcal{A}_1((\mathsf{sk}_i, \mathsf{pk}_i) \mid \forall i)$
3 : $cPK \leftarrow \{\mathsf{pk}_{id_0}, \mathsf{pk}_{id_1}\}$; $\mathsf{ls}_{id_0}, \mathsf{ls}_{id_1} \leftarrow \emptyset$
4 : $b \leftarrow\$ \{0,1\}$
5 : $b' \leftarrow \mathcal{A}_2^{\mathsf{OChSign}, \mathsf{OChLink}}(\mathsf{st})$
6 : **return** $b' = b$

$\mathsf{Exp}^{\text{unf}}_{\mathcal{A},RS}(1^\lambda)$

1 : **for** $i \in [N]$ **do** $(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$; $ls_i \leftarrow \emptyset$
2 : $Q_{sig}, Q_l, C, L \leftarrow \emptyset$
3 : $(m^*, R^*, \sigma^*, \mathsf{nym}^*) \leftarrow\$ \mathcal{A}^{\mathsf{OSign},\mathsf{OLink},\mathsf{Corr},\mathsf{OLS}}(\mathsf{pk}_i \mid \forall i)$
4 : **if** $\mathsf{Verify}(m^*, R^*, \sigma^*, \mathsf{nym}^*) \stackrel{?}{=} 1 \wedge (\cdot, (m^*, R^*, \cdot, \cdot)) \notin Q_{sig}$
5 : $\quad \wedge R^* \subseteq \{(\mathsf{pk}_i)\}_{i=1}^N \setminus C$ **then return** 1
6 : **return** 0

$\mathsf{Exp}^{\text{l-unf}}_{\mathcal{A},RS}(1^\lambda)$

1 : **for** $i \in [N]$ **do** $(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$; $\mathsf{ls}_i \leftarrow \emptyset$
2 : $Q_{sig}, Q_l, C, L \leftarrow \emptyset$
3 : $(\mathbf{\Sigma}, \pi_l) \leftarrow \mathcal{A}^{\mathsf{OSign},\mathsf{OLink},\mathsf{Corr},\mathsf{OLS}}(\mathsf{pk}_i \mid \forall i)$
4 : **if** $\mathsf{VfLink}(\mathbf{\Sigma}, \pi_l) \stackrel{?}{=} 1 \wedge (\cdot, \mathbf{\Sigma}) \notin Q_l$
5 : $\quad \wedge \exists i \in [N], (m, R, \sigma, \mathsf{nym}) \in \mathbf{\Sigma} : (i, \mathsf{nym}) \notin L$
6 : $\quad \wedge (i, (m, R, \sigma, \mathsf{nym})) \in Q_{sig}$ **then return** 1
7 : **return** 0

$\mathsf{Exp}^{\text{frame}}_{\mathcal{A},RS}(1^\lambda)$

1 : **for** $i \in [N]$ **do** $(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$; $\mathsf{ls}_i \leftarrow \emptyset$
2 : $Q_{sig}, Q_l, C, L \leftarrow \emptyset$
3 : $(m, R, \sigma, \mathsf{nym}) \leftarrow \mathcal{A}^{\mathsf{OSign},\mathsf{OLink},\mathsf{Corr},\mathsf{OLS}}(\mathsf{pk}_i \mid \forall i)$
4 : **if** $\mathsf{Verify}(m, R, \sigma, \mathsf{nym}) \stackrel{?}{=} 1 \wedge \exists i \in [N] : (i, \mathsf{nym}) \notin L$
5 : $\quad \wedge (i, (m, R, \cdot, \mathsf{nym})) \notin Q_{sig}$
6 : $\quad \wedge (i, (\cdot, \cdot, \cdot, \mathsf{nym})) \in Q_{sig}$ **then return** 1
7 : **return** 0

(a) Formal experiments for unforgeability, linkable anonymity, link unforgeability and non-frameability.

$\mathsf{OSign}(i, m, R)$

1 : **if** $\mathsf{pk}_i \notin R$ **then return** $\perp$
2 : $(\sigma, \mathsf{nym}, \mathsf{ls}_i) \leftarrow\$ \mathsf{Sign}(m, R, \mathsf{sk}_i, \mathsf{ls}_i)$
3 : $Q_{sig} \leftarrow Q_{sig} \cup (i, (m, R, \sigma, \mathsf{nym}))$
4 : **return** (σ, nym)

$\mathsf{OChSign}(\beta, m, R)$ for $cPK \subseteq R$

1 : **if** $\beta = 1$ **then**
2 : $\quad (\sigma, \mathsf{nym}, \mathsf{ls}_{id_{1-b}}) \leftarrow\$ \mathsf{Sign}(m, R, \mathsf{sk}_{id_{1-b}}, \mathsf{ls}_{id_{1-b}})$
3 : **else** $(\sigma, \mathsf{nym}, \mathsf{ls}_{id_b}) \leftarrow\$ \mathsf{Sign}(m, R, \mathsf{sk}_{id_b}, \mathsf{ls}_{id_b})$
4 : **return** (σ, nym)

$\mathsf{Corr}(i)$

1 : $C \leftarrow C \cup \{\mathsf{pk}_i\}$
2 : **return** sk_i

$\mathsf{OLink}(i, \mathbf{\Sigma})$

1 : $\pi_l \leftarrow \mathsf{Link}(\mathsf{ls}_i, \mathbf{\Sigma})$
2 : $Q_l \leftarrow Q_l \cup (i, \mathbf{\Sigma})$
3 : **return** π_l

$\mathsf{OChLink}(\beta, \mathbf{\Sigma})$

1 : **if** $\beta = 1$ **then** $\pi_l \leftarrow \mathsf{Link}(\mathsf{ls}_{id_{1-b}}, \mathbf{\Sigma})$
2 : **else** $\pi_l \leftarrow \mathsf{Link}(\mathsf{ls}_{id_b}, \mathbf{\Sigma})$
3 : **return** π_l

$\mathsf{OLS}(i, \mathsf{nym})$

1 : **for** $(\mathsf{nym}', \tau) \in \mathsf{ls}_i$ **do if** $\mathsf{nym} \stackrel{?}{=} \mathsf{nym}'$ **then**
2 : $\quad L \leftarrow L \cup (i, \mathsf{nym})$; **return** τ
3 : **return** $\perp$

(b) Oracles for our security experiments.

Figure 1: Our security experiments and accompanying oracles for ring signature with user-controlled linkability.

Definition 15 (Zero Knowledge)**.** A NIZK proof system has the zero-knowledge property if the following advantage $\mathsf{Adv}^{\text{zk}}_{\mathcal{A},\mathsf{NIZK}}(\lambda)$ is negligible in the security parameter λ:

$$\left| \Pr\left[\rho \leftarrow \mathsf{NIZK.Setup}(\lambda) : \mathcal{A}^{\mathsf{NIZK.P}(\rho,\cdot,\cdot)}(\rho) = 1\right] - \Pr\left[(\rho', \tau) \leftarrow \mathsf{SimSetup}(\lambda) : \mathcal{A}^{\mathsf{Sim}(\rho',\tau,\cdot)}(\rho') = 1\right] \right|$$

3.1 Σ-Procotol

The formal description of the Σ-protocol for proving knowledge of isomophisms between lattices under zero-knowledge is shown in Figure 2, courtesy of Ducas and van Woerden (2022). Actually, this protocol applies to a pair of quadratic forms Q_0, Q_1 and the unimodular matrix $U \in \mathcal{GL}_n(\mathbb{Z})$ taking one to the other. Recalling Remark 1 these forms may be associated with two different bases $B_0, B_1 \in \mathbb{R}^n$ by considering the gram matrices ${B_0}^T B_0$ and ${B_1}^T B_1$. The relation for the protocol is then reformulated as follows.

$$R := \{(U; B_0, B_1) \in \mathcal{GL}_n(\mathbb{Z}) \times \mathcal{GL}_n(\mathbb{R}) \times \mathcal{GL}_n(\mathbb{R}) : {B_0}^T B_0 = U^T {B_1}^T B_1 U\}$$

Remark 3. Each basis B_i generates a lattice L_i, and $B_1 U$ also generates L_1 for any $U \in \mathcal{GL}_n(\mathbb{Z})$. The relation states that the gram matrix of the basis ${B_1}' := B_1 U$ is the same as that of B_0, implying that L_0, L_1 are isomorphic with ${B_1}' = O B_0$ for some $O \in O_n(\mathbb{R})$. Furthermore, knowledge of U allows the reconstruction of O from B_0, B_1 - we simply compute $B_1 U$ and compare with B_0. This means that the special soundness

of the proof system extends to knowledge of O. The converse is also true; knowledge of O allows for the efficient reconstruction of U via computing the basis OB_0. This implies that the zero-knowledge property of the proof system for U extends to that of O.

3.2 Extension to Set Membership Proof

Let $R := \{B_i\}_{i=1}^N$ be a ring containing N lattice bases each sampled from a given isomorphism class $[S]$, and consider some additional $L \in [S]$ with basis B_0. Consider further the set of isomorphisms $\{O \in O_n(\mathbb{R}) : O^{-1} \cdot L \in R\}$; one can construct a proof of knowledge of a member of this set from the Σ-protocol in Section 3.1 using an index-hiding Merkle tree as follows (Khuc et al., 2023).

Denote by I the index of the basis in R for which P knows the witness. For each $B_i \in R$, P samples (B_i', V_i) via $\mathsf{BasisSample}(B_i)$ and then builds the Merkle tree of depth $\log N$ with leaves $\{G_i'\}_{i=1}^N$ from a suitable hash function H, where each G_i' is the gram matrix of the corresponding B_i'. The Prover computes the root and sends it to V instead of G'. The Verifier replies with the challenge $b \in \{0,1\}$ and P proceeds according to b as follows.

- If $b = 0$, then P replies with $U_I^{-1} V_I$ together with the path corresponding to the leaf G_I'. The Verifier computes the leaf $G_I' = {V_I}^T {U_I^{-1}}^T {B_0}^T B_0 U_I^{-1} V_I$ and uses the path to reconstruct the root from G_I', checking that the result matches the initial commitment from P.
- If $b = 1$, then P simply provides all the leaves $\{G_i'\}_{i=1}^N$. The Verifier reconstructs the whole tree and computes the root, checking that the result matches the initial commitment from P.

Given that the Merkle tree is index-hiding, the value of I is not implicated to V by the leaf G_I for the case that $b = 0$. To meet some given soundness bound, the protocol must be repeated M times. Several optimisations can be made to the protocol, such as replacing all the B_i' in the response for $b = 1$ with the random seeds used by $\mathsf{BasisSample}$ on line 1. The challenge space can then be weighted to set $b = 1$ in a majority K of the repetitions, with $b = 0$ in the remaining $M - K$ repetitions resulting in a smaller overall proof size despite increasing the total number of repetitions. We refer to Khuc et al. (2023) for the details of the fully-optimised protocol.

4 Scheme Specification for RSSL

At a high level, our ring signature scheme composes the LIP signature under a randomised key with a zero-knowledge proof system. Since the underlying signature mechanism is based on Ducas and van Woerden (2022), we share the following public parameters.

- A decoding lattice S of dimension n. It is assumed that Gaussian sampling can be performed efficiently on S via Lemma 3.
- A width s for sampling public keys from $[S]$ as above, which must satisfy the following bound: $s \geq \max(\lambda_n(S), \|B_S^*\| \cdot \sqrt{\ln(2n+4)/\pi})$.
- A hash function $\mathsf{H} : \mathcal{M} \to \mathbb{T}_q^n$ where $q := \left\lceil \frac{s \cdot n}{\rho} \cdot \sqrt{\ln(2n+4)/\pi} \right\rceil$. This is modeled as a random oracle and may be reprogrammed.
- A signature verification bound ρ with $\rho/\sqrt{n} \geq \eta([S])$ above the smoothing parameter.

We start with the underlying idea behind LIP signatures. Sample a public key P in class $[S]$ along with an isomorphism O that can be used to map the ambient space of S to that of P. Messages are hashed into this space by H, and a signature is a point in P within a distance ρ of the hash. To sign the message m, one applies O^{-1} to $\mathsf{H}(m)$ and easily finds a nearby point in S using Gaussian sampling, then applies O to map this back to a close point in P. The security depends on the assumed hardness of finding O given arbitrary bases of each of P and S, which is the LIP problem.

Our approach to convert this into a ring signature is to modify the signing process such that the public key is re-randomised before signing. The signature must then include a proof linking the randomised key to a ring of possible static public keys. To perform this re-randomisation within Sign, we simply repeat the key generation process using the user's public key P as input in place of the decoding lattice S to obtain an ephemeral key Q, which acts as the pseudonym. This creates an extended path of lattice isomorphisms $Q \xrightarrow{T} P \xrightarrow{O} S$ to the decoding space S, and so the signer can still create signatures under the ephemeral signing key of composed isomorphisms TO. These signatures will verify against the randomised key Q.

To prove their membership in R, the signer can use the proof system described in Section 3 to construct a zero-knowledge proof of the isomorphism T linking Q to P. Actually, this system proves knowledge of a unimodular V against the gram matrices associated with B_Q and B_P. However by Remark 3 knowledge of V is equivalent to knowledge of T and both the special soundness and zero-knowledge properties hold with respect to T. To ensure anonymity, we extend this

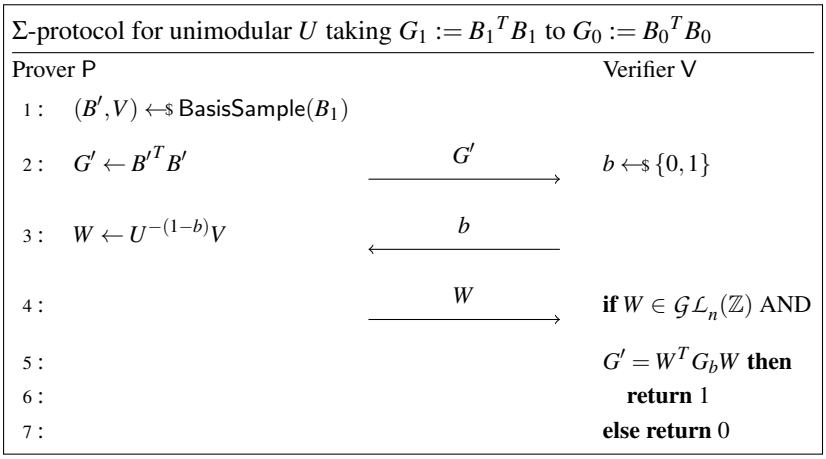

Figure 2: Σ-protocol behind the zero-knowledge proof of knowledge system NIZK_R.

proof to a set-membership proof Π_Q with witness P as an anonymous element of the ring R as detailed in Section 3.2.

Explicit linkability is also achieved through the use of the proof system of Section 3; to link two pseudonym lattices Q_1, Q_2 it is sufficient to prove knowledge of an isomorphism W between them. Such a map results by composing the corresponding T_1, T_2 (more precisely, exactly one must be inverted), which are stored in the user's linking secret at the time of signing.

Remark 4. To extend our scheme to incorporate the notion of *scope* as defined by Fiore et al. (2022), we can generate each T as follows.

- Sample a static linking secret $\mathsf{ls} \leftarrow \mathsf{IsoSample}$. This map remains secret to the user.
- For each signature, sample a scope $\mathsf{scp} \leftarrow \mathsf{IsoSample}$. This map is output as part of the signature.
- During signing, compute T as $\mathsf{ls} \cdot \mathsf{scp}$. It is important that ls is applied *after* scp, as then the non-commutativity of the composition of isomorphisms prevents the verifier from simply negating scp via its inverse.

However, the above changes increase overall complexity of the scheme. Our simplified definition does require the signer to record distinct linking secrets for each pseudonym, but on the other hand to satisfy these stronger notions it would have to store the equivalent information implicitly within each scope it uses. For this reason we believe that this is a realistic relaxation of the model, achieving comparable security guarantees and ultimately allowing for a more efficient construction, while more accurately reflecting how a real-world implementation of our scheme would look.

5 Security Analysis

In this section we prove the security of our construction, based on the lattice isomorphism problem.

Theorem 1 (Anonymity)**.** *The ring signature scheme* RingSLIP *is anonymous under full key exposure provided that the proof system* NIZK *is zero-knowledge. More precisely, an adversary $\mathcal{A}$ wins the anonymity game shown in Figure 1a with advantage:*

$$\mathsf{Adv}^{\mathsf{anon}}_{\mathcal{A},\mathsf{RingSLIP}}(\lambda) \leqslant \mathsf{Adv}^{\mathsf{zk}}_{\mathcal{A},\mathsf{NIZK}}(\lambda)$$

Proof. Consider an adversary $\mathcal{A}$ against the anonymity experiment for RLIP. Each query to its challenge oracle Chall_b is answered with some $\mathsf{Sig} = (\boldsymbol{\sigma}, \Pi_Q)$ where Π_Q is the output of the prover NIZK.P on some input witness V. Suppose this prover is replaced with a simulator NIZK.Sim that computes Π_Q without knowledge of the witness V. Then Π_Q depends only on the public information B_Q, R, while $\boldsymbol{\sigma}$ also depends only on B_Q.

It remains to show that the distribution of B_Q does not depend on B_P. Recall that the distribution of BasisSample does not depend on the choice of input basis and therefore B_Q is distributed equivalently to $B'_Q \leftarrow\$ \mathcal{D}_s(TOB_S)$. Now since each of $T, O \in O_n(\mathbb{R})$ are uniformly distributed so is $TO \in O_n(\mathbb{R})$. Hence the distribution of B_Q depends only on B_S which $\mathcal{A}$ already knows, meaning $\mathcal{A}$ must have zero advantage in the game with NIZK.Sim.

Now the advantage of $\mathcal{A}$ against the original anonymity game is bounded by the advantage of $\mathcal{A}$ against distinguishing NIZK.Sim from NIZK.P, which is negligible if NIZK has the zero-knowledge property. □

The proof of unforgeability of our ring signature follows a similar argument to that of Ducas and van

$\mathsf{KGen}(B_S)$

$O \leftarrow\!\$\ O_n(\mathbb{R})$
$(B_P, U) \leftarrow\!\$\ \mathcal{D}_s(OB_S)$
return (O, B_P)

$\mathsf{Link}(\mathsf{ls}, B_{Q_1}, B_{Q_2})$

for $i \in \{1,2\}$
  $V_i \leftarrow \perp$; **for** $(B_Q, V) \in \mathsf{ls}$
    if $B_Q \stackrel{?}{=} B_{Q_i}$ **then** $V_i \leftarrow V$
$W \leftarrow V_1^{-1} \cdot V_2$
return W

$\mathsf{Sign}(m, R, O, \mathsf{ls})$

$T \leftarrow\!\$\ O_n(\mathbb{R})$
$(B_Q, V) \leftarrow\!\$\ \mathcal{D}_s(TB_P)$
$\mathsf{ls} \leftarrow \mathsf{ls} \cup (B_Q, V)$
$\boldsymbol{t} \leftarrow B_Q \mathsf{H}(m \| R)$
$\boldsymbol{t}' \leftarrow O^{-1}T^{-1}\boldsymbol{t}$
$\boldsymbol{\sigma}' \leftarrow\!\$\ \mathcal{D}_{S,\rho/\sqrt{n},\boldsymbol{t}'}$
$\boldsymbol{\sigma} \leftarrow TO\boldsymbol{\sigma}'$
$\Pi_Q \leftarrow \mathsf{NIZK.P}(V; B_Q, R)$
$\mathsf{Sig} \leftarrow (\boldsymbol{\sigma}, \Pi_Q)$
return $(\mathsf{Sig}, B_Q, \mathsf{ls})$

$\mathsf{Verify}(m, R, \mathsf{Sig} = (\boldsymbol{\sigma}, \Pi_Q), B_Q)$

$\boldsymbol{t} \leftarrow B_Q \mathsf{H}(m \| R)$
$b \leftarrow \boldsymbol{\sigma} \in Q \wedge \|\boldsymbol{\sigma} - \boldsymbol{t}\| \leqslant \rho$
  $\wedge\, \mathsf{NIZK.V}((B_Q, R), \Pi_Q)$
return b

$\mathsf{VfLink}(B_{Q_1}, B_{Q_2}, W)$

$B_{Q'} \leftarrow B_{Q_1} W$
$G_{Q'} \leftarrow {B_{Q'}}^T B_{Q'}$; $G_{Q_2} \leftarrow {B_{Q_2}}^T B_{Q_2}$
$b \leftarrow G_{Q'} \stackrel{?}{=} G_{Q_2}$
return b

Figure 3: Our algorithms for RingSLIP.

Woerden (2022), which relies on the hardness of Δ-LIP for certain lattices containing sparse sublattices. The difference is that the re-randomisation of the public key introduced in our ring signature allows the adversary to chose any isomorphism of the public key to forge against, which widens their range of attack. We extend the proof to extract a short lattice vector from an adversary that is capable of such an attack, thus we additionally rely on the hardness of SVP.

Theorem 2 (Unforgeability)**.** *The ring signature scheme* RingSLIP *is selectively unforgeable w.r.t. insider corruption provided that the proof system* NIZK *has the special-soundness property. More precisely, an adversary $\mathcal{A}$ wins the unforgeability game shown in Figure 1a with advantage:*

$$\mathsf{Adv}^{\mathsf{unf}}_{\mathcal{A},\mathsf{RingSLIP}}(\lambda) \leqslant q_H \big(\mathsf{Adv}^{\mathsf{sp-sound}}_{\mathcal{A},\mathsf{NIZK}}(\lambda) + \mathsf{Adv}^{\Delta-\mathsf{lip}}_{\mathcal{A}}(\lambda) + \mathsf{Adv}^{\mathsf{svp}}_{\mathcal{A}}(\lambda)\big)$$

where q_H is the number of queries made to the random oracle during the experiment.

Proof. Let M be a lattice with $\det(M) = \det(S)$ and $\lambda_2(M) \gg 4\sqrt{n} \cdot \eta([S])$. Consider the following series of games:

$\mathsf{Unf}_1(\lambda)$**:** The original unforgeability game with our scheme as shown in Figure 1a.

$\mathsf{Unf}_2(\lambda)$**:** The secret key is no longer used to sign messages. Instead, the signature query for each message m is answered as follows. First, a random error vector $\boldsymbol{e}_{m,R}$ is sampled from $1/q \cdot D_{Q,q\rho/\sqrt{n}}$. The random oracle H is then reprogrammed such that $\mathsf{H}(m\|R) := B_P^{-1}\boldsymbol{e}_{m,R} \bmod B_Q$. Then $\boldsymbol{\sigma} := \mathsf{H}(m\|R) - \boldsymbol{e}_{m,R}$ is a valid signature for m and is returned.

$\mathsf{Unf}_3(\lambda)$**:** Let $q_\mathsf{H} = \mathsf{poly}(n)$ denote the number of times $\mathcal{A}$ queries the random oracle H. At the start of the game choose $i \in [q_\mathsf{H}]$ uniformly at random, then at the end of the adversarial phase abort the game unless the message m^* in the forgery submitted by the adversary matches the value m_i given as input to H on the ith query.

$\mathsf{Unf}_4(\lambda)$**:** Reprogram the output of H on the ith query as follows. Take $\boldsymbol{c} = (2q\rho, 0, \ldots, 0) \in \mathbb{R}^n$. Sample $\boldsymbol{e} \leftarrow\!\$\ 1/q \cdot \mathcal{D}_{Q,q\rho/\sqrt{n},\boldsymbol{c}}$. Then set $\mathsf{H}(m_i) = B_P^{-1}\boldsymbol{e} \bmod B_Q$ as the output of H.

$\mathsf{Unf}_5(\lambda)$**:** In KGen the public key P is now sampled from $D_s([M])$ instead of from $D_s([S])$.

We denote by ε_i^{unf} the probability $\Pr[\mathsf{Unf_i}]$ where $i \in [5]$, and have

$$\varepsilon_1^{\mathsf{unf}} := \mathsf{Adv}^{\mathsf{unf}}_{\mathcal{A},\mathsf{RLIP}}(\lambda).$$

Consider Unf_2. Because Q is sampled from $D_s[S]$, we have that the width $q\rho/\sqrt{n}$ is above the required bound $\|B_P^*\| \cdot \sqrt{\ln(2n+4)/\pi}$ for the Gaussian sampler to be able to sample from this distribution. Furthermore, the distribution of $\mathsf{H}(m\|R)$ is proportional to that of a Gaussian of width $p/\sqrt{n}$, which is above the smoothing parameter of S and therefore of Q. Hence, the distribution is statistically indistinguishable from uniform over the parallelepiped B_Q and the random oracle is simulated correctly. Furthermore, the distribution of $\boldsymbol{\sigma}$ conditioned on $\mathsf{H}(m\|R)$ appears as $D_{Q,\rho/\sqrt{n}}$ from the adversary's point of view, which is identical to that of Unf_1 and implies that

$$\varepsilon_1^{\mathsf{unf}} = \varepsilon_2^{\mathsf{unf}}.$$

In Unf_3 the probably that the game does not abort due to the change from Unf_2 is simply $1/q_\mathsf{H}$, so we have

$$\varepsilon_3^{\mathsf{unf}} = \frac{1}{q_\mathsf{H}} \cdot \varepsilon_2^{\mathsf{unf}}.$$

In Unf_4 the reprogramming of H is done in the same manner as Unf_2, so by the same argument as before the random oracle is corrected simulated for m_i and therefore Unf_4 is indistinguishable from Unf_3.

$$\varepsilon_4^{\mathsf{unf}} = \varepsilon_3^{\mathsf{unf}}.$$

In Unf_5 observe that it is not required to sample from M due to the changes inherited from Unf_2. Furthermore, the simulated signatures are created using only information already public to $\mathcal{A}$. An adversary that can distinguish Unf_4 from Unf_5 can be used to decide whether $P \in [M]$ or $P \in [S]$, thus breaking average-case distinguish LIP parametrised over M, S. Therefore we have

$$\left|\varepsilon_4^{\mathsf{unf}} - \varepsilon_5^{\mathsf{unf}}\right| \leqslant \mathsf{Adv}_{\mathcal{A}}^{\Delta-\mathsf{lip}}(\lambda).$$

Now consider an adversary $\mathcal{A}$ against Unf_5 that produces a valid forgery on m_i. By the special-soundness of the NIZK we may assume that $\mathcal{A}$ uses a pseudonym $Q \in [M]$, as otherwise there would not exist an isomorphism $M \to Q$ to extract. Now $\rho < \|\mathsf{H}(m_i)\| < 3\rho$, so any valid signature on m_i has length bounded by $\|\mathsf{H}(m_i)\| + \rho < 4\rho$ and is non-zero. Since $\lambda_2 > 4\rho$, the only vectors in M that satisfy this bound are of the form $k\boldsymbol{\lambda}_1$ with $k \in \mathbb{Z}\backslash\{0\}$ and so we receive such a vector from $\mathcal{A}$.

Now to recover $\pm\boldsymbol{\lambda}_1$ from $k\boldsymbol{\lambda}_1$ write the signature as a linear combination of the lattice basis vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ as follows.

$$\boldsymbol{\sigma} = \alpha_1\boldsymbol{b}_1 + \ldots + \alpha_n\boldsymbol{b}_n$$

This corresponds to a system of n linear equations in n variables that has a unique solution computable via Gaussian elimination. We know this vector has the form $\boldsymbol{\sigma} = k\boldsymbol{\lambda}_1$ and we have that $\boldsymbol{\lambda}_1$ itself can be written as a linear combination of $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ with integer coefficients. Therefore, we can compute $\beta = \gcd(\alpha_1, \ldots, \alpha_n)$ and deduce that $\boldsymbol{\lambda}_1 = \boldsymbol{\sigma}/\beta$ (up to the sign). This solves SVP in M and hence

$$\varepsilon_5^{\mathsf{unf}} \leqslant \mathsf{Adv}_{\mathcal{A},\mathsf{NIZK}}^{\mathsf{sp-sound}}(\lambda) + \mathsf{Adv}_{\mathcal{A}}^{\mathsf{svp}}(\lambda).$$

□

Theorem 3 (Link Unforgeability)**.** *The ring signature scheme* RingSLIP *is link-unforgeable provided that the proof system* NIZK *is zero-knowledge and has the special soundness property, and that the HAWK digital signature scheme is unforgeable. More precisely, an adversary $\mathcal{A}$ wins the link unforgeability game shown in Figure 1a with advantage:*

$$\mathsf{Adv}_{\mathcal{A},\mathsf{RingSLIP}}^{\mathsf{l-unf}}(\lambda) \leqslant q_S\big(\mathsf{Adv}_{\mathcal{A},\mathsf{NIZK}}^{\mathsf{zk}}(\lambda) + \mathsf{Adv}_{\mathcal{A},\mathsf{NIZK}}^{\mathsf{sp-sound}}(\lambda) + \mathsf{Adv}_{\mathcal{A},HAWK}^{\mathsf{euf\text{-}cma}}(\lambda)\big)$$

where q_S is the number of queries made to OSign *during the experiment.*

Proof. We prove the link unforgeability by a series of games. As the linking proofs in RingSLIP are constructed pairwise, we assume for simplicity that the $\boldsymbol{\Sigma}$ submitted by the adversary in each game consists of two signature tuples. For larger sets of tuples consider a hybrid argument. The games are as follows.

- $\mathsf{L-Unf}_1(\lambda)$**:** The original link unforgeability game with our scheme as shown in Figure 1a.
- $\mathsf{L-Unf}_2(\lambda)$**:** Let $q_S = \mathsf{poly}(n)$ denote the number of queries $\mathcal{A}$ makes to OSign. Choose a random $k \in [q_S]$ and modify lines 5-6 to require that $(\cdot, (m, R, \sigma, \mathsf{nym}))$ also corresponds to the output of the k-th query to OSign. If neither signature tuple passes this check, abort the game. Denote by i the identity provided in this query.
- $\mathsf{L-Unf}_3(\lambda)$**:** On the k-th query OSign generates the proof-of-knowledge Π_Q in the signature using a simulator NIZK.Sim in place of the prover NIZK.P.

We denote by ε_i^{l-unf} the probability $\Pr[\mathsf{L\text{-}Unf_i}]$ where $i \in [3]$, and have

$$\varepsilon_1^{\mathsf{l-unf}} := \mathsf{Adv}_{\mathcal{A},\mathsf{RingSLIP}}^{\mathsf{l-unf}}(\lambda).$$

In $\mathsf{L\text{-}Unf}_2$ the probability that the game does not abort due to the change from $\mathsf{L\text{-}Unf}_1$ is at least $1/q_S$, because at least one of the two given signature tuples already corresponds to an element of Q_{sig}. Hence we have

$$\varepsilon_2^{\mathsf{l-unf}} \geqslant \frac{1}{q_S}\varepsilon_1^{\mathsf{l-unf}}.$$

Furthermore, by the zero-knowledge property of the NIZK, we have that

$$\left|\varepsilon_2^{\mathsf{l-unf}} - \varepsilon_3^{\mathsf{l-unf}}\right| \leqslant \mathsf{Adv}_{\mathcal{A},\mathsf{NIZK}}^{\mathsf{zk}}(\lambda).$$

Finally we construct an adversary $\mathcal{B}$ against the unforgeability of HAWK from the adversary $\mathcal{A}$ against $\mathsf{L\text{-}Unf}_3$. Denote by P the public key of the HAWK game, and assume that parameters including S are shared with $\mathsf{L\text{-}Unf}_3$. We start by running $\mathsf{L\text{-}Unf}_3$ as normal, up until the k-th query to OSign is made with input (i, m, R) where $\mathsf{pk}_i \in R$. To respond to this query, $\mathcal{B}$ sets $\mathsf{nym} = B_P$ and obtains σ from the HAWK signing oracle on input $m||R$. Because of the change in $\mathsf{L\text{-}Unf}_3$, $\mathcal{B}$ can use the simulator NIZK.Sim to create the proof Π_P without knowledge of an isomorphism from pk_i to P (note that such an isomorphism does exist as all lattices considered are in $[S]$). Furthermore, by line 5 $\mathcal{B}$ loses $\mathsf{L\text{-}Unf}_3$ if they query $\mathsf{OSL}(i, \mathsf{nym})$ and so it does not matter that $\mathcal{B}$ can not respond to such a query. After the response to OSign is made, the rest of $\mathsf{L\text{-}Unf}_3$ is simulated as normal.

At the end of $\mathsf{L\text{-}Unf}_3$, if $\mathcal{A}$ is successful it outputs a linking proof $\pi_l = W$, along with a signature tuple

$(m', R', (\sigma', \Pi_Q), Q)$. By Remark 3 $\mathcal{B}$ can construct a map θ from $P = \mathsf{nym}$ to some other $Q = \mathsf{nym}'$ using W. Then by special soundness of the NIZK, $\mathcal{B}$ may run the extractor Ext on the proof Π_Q to obtain some τ mapping Q onto some $\mathsf{pk}_j \in R'$. By composing θ together with τ and sk_j, $\mathcal{B}$ is able to obtain an isomorphism $P \to S$ which is equivalent to the HAWK signing key. Thus $\mathcal{B}$ can use this key to create a forgery on any message of its choice and win the HAWK experiment.

$$\varepsilon_3^{\mathsf{l-unf}} \leqslant \mathsf{Adv}^{\mathsf{sp-sound}}_{\mathcal{A},\mathsf{NIZK}}(\lambda) + \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{A},HAWK}(\lambda).$$

□

Theorem 4 (Non-Frameability)**.** *The ring signature scheme* RingSLIP *is non-frameable provided that the proof system* NIZK *is zero-knowledge and the HAWK digital signature scheme is unforgeable. More precisely, an adversary $\mathcal{A}$ wins the non-frameability game shown in Figure 1a with advantage:*

$$\mathsf{Adv}^{\mathsf{l-unf}}_{\mathcal{A},\mathsf{RingSLIP}}(\lambda) \leqslant q_S\big(\mathsf{Adv}^{\mathsf{zk}}_{\mathcal{A},\mathsf{NIZK}}(\lambda) + \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{A},HAWK}(\lambda)\big)$$

where q_S is the number of queries made to OSign *during the experiment.*

Proof. $\mathsf{Frame}_1(\lambda)$**:** The original non-frameability game with our scheme as shown in Figure 1a.

$\mathsf{Frame}_2(\lambda)$**:** Let $q_S = \mathsf{poly}(n)$ denote the number of queries $\mathcal{A}$ makes to OSign. Choose a random $k \in [q_S]$ and modify lines 4-6 to require that $(\cdot, (\cdot, \cdot, \cdot, \mathsf{nym}))$ also corresponds to the output of the k-th query to OSign. If neither signature tuple passes this check, abort the game. Denote by i the identity provided in this query.

$\mathsf{Frame}_3(\lambda)$**:** On the k-th query OSign generates the proof-of-knowledge Π_Q in the signature using a simulator NIZK.Sim in place of the prover NIZK.P.

We denote by ε_i^{frame} the probability $\Pr[\mathsf{Frame}_i]$ where $i \in [3]$, and have

$$\varepsilon_1^{\mathsf{frame}} := \mathsf{Adv}^{\mathsf{frame}}_{\mathcal{A},\mathsf{RingSLIP}}(\lambda).$$

In Frame_2 the probability that the game does not abort due to the change from Frame_1 is at least $1/q_S$, because by line 6 we have that nym appears in at least one element of Q_{sig}. Hence we have

$$\varepsilon_2^{\mathsf{frame}} \geqslant \frac{1}{q_S}\varepsilon_1^{\mathsf{frame}}.$$

Furthermore, by the zero-knowledge property of the NIZK, we have that

$$\left|\varepsilon_2^{\mathsf{frame}} - \varepsilon_3^{\mathsf{frame}}\right| \leqslant \mathsf{Adv}^{\mathsf{zk}}_{\mathcal{A},\mathsf{NIZK}}(\lambda).$$

Finally we construct an adversary $\mathcal{B}$ against the unforgeability of HAWK from the adversary $\mathcal{A}$ against Frame_3. Denote by P the public key of the HAWK game, and assume that parameters including S are shared with Frame_3. We start by running Frame_3 as normal, up until the k-th query to OSign is made with input (i, m, R) where $\mathsf{pk}_i \in R$. To respond to this query, $\mathcal{B}$ sets $\mathsf{nym} = B_P$ and obtains σ from the HAWK signing oracle on input $m||R$. Because of the change in Frame_3, $\mathcal{B}$ can use the simulator NIZK.Sim to create the proof Π_P without knowledge of an isomorphism from pk_i to P (note that such an isomorphism does exist as all lattices considered are in $[S]$). Furthermore, by line 4 $\mathcal{B}$ loses Frame_3 if they query $\mathsf{OSL}(i, \mathsf{nym})$ and so it does not matter that $\mathcal{B}$ can not respond to such a query. After the response to OSign is made, the rest of Frame_3 is simulated as normal.

At the end of Frame_3, if $\mathcal{A}$ is successful it outputs a signature tuple $(m, R, (\sigma, \Pi), \mathsf{nym})$. By line 6 we have that $\mathsf{nym} = P$ and $\Pi = \Pi_P$, by line 4 we have that σ is a valid HAWK signature on $m||R$ against P, and by line 5 we have that $m||R$ was never queried to the HAWK signing oracle. Therefore $\mathcal{B}$ wins the HAWK experiment by submitting its forgery as $(m||R, \sigma)$.

$$\varepsilon_3^{\mathsf{frame}} \leqslant \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{A},HAWK}(\lambda).$$

□

5.1 Attack on Linkability

In this section we examine the attack by Budroni, Chi-Domínguez, and Franch (2025a) on the linkable ring signature by Khuc et al. (2023), and argue why our scheme is able to avoid this attack.

The scheme of Khuc et al. uses a tag system to achieve linkability. For each key-pair $(P, (O, U))$, the modified scheme generates the associated tag P' as the image of the decoding lattice S under the inverse of the secret key (O^{-1}, U^{-1}). A signature is then composed of a tag P'_i together with an extended OR-proof proving knowledge of O_i against both the ring $(P_1, \ldots, P_n)$ and the tag P'_i. Now the linkable anonymity properties require that the pair (P, P') formed by a public key P together with its matching tag P' is indistinguishable from an arbitrary pair of lattices (L, L'), each sampled independently from the isomorphism class $[S]$.

The attack exploits the fact that from each such pair two distinct lattice relations emerge with the same underlying transformation: $B_P = OB_SU$ on the side of the public key, and $B_S = OB_{P'}U$ on the side of the tag. Considering the gram matrices of the bases $B_P, B_{P'}, B_S$ leads to the following system of matrix

equations.

$$G_P \cdot X = Y \cdot G_S$$
$$G_S \cdot X = Y \cdot G_{P'}$$

Clearly we have that $X = U^{-1}$ and $Y = U^T$ is a solution to this system, but Budroni, Chi-Domínguez, and Franch (2025a) show that this expands to a family of solutions with dimension n generated by the centraliser of $G_S \cdot G_P^{-1}$. This means that the following matrix is reducible and has rank at most $2n^2 - n$.

$$A := \begin{bmatrix} G_P \otimes I_n & I_n \otimes (-G_S) \\ G_S \otimes I_n & I_n \otimes (-G_{P'}) \end{bmatrix} \in \mathbb{R}^{2n^2 \times 2n^2}$$

In contrast, Budroni, Chi-Domínguez, and Franch (2025a) also show that the matrix

$$A' := \begin{bmatrix} G_L \otimes I_n & I_n \otimes (-G_S) \\ G_S \otimes I_n & I_n \otimes (-G_{L'}) \end{bmatrix} \in \mathbb{R}^{2n^2 \times 2n^2}$$

has full rank $2n^2$ with overwhelming probability for general $L, L' \leftarrow\$\ \mathcal{D}_s(S)$, thus allowing one to distinguish the pairs of lattices by whether or not the matrix reduction is successful.

The key insight in the above attack is that the two lattice relations sharing a common transform U causes the test matrix A to be reducible. In our scheme, the two matrix relations that may be derived are $B_P = OB_SU$ and $B_Q = TB_PV$. Because we sample the salt transform (T,V) separately from the key (O,U), we ensure that our scheme is not vulnerable to the attack.

6 Instantiation and Optimisations

To instantiate our scheme at NIST Level 1 we opt to implement the LIP-based signature with HAWK (Ducas et al., 2022), and build out our zero-knowledge proof with the SHA-256 hash function (National Institute of Standards and Technology, 2015). For sake of comparison, we target NIST level 1 security. This gives rise to signer keys sizes of $184B$ and $1024B$ for sk and pk respectively. We also note that for each pseudonym that the signer intends to re-use, the linking secret grows by $184B$. Asymptotically, the signature is logarithmic in the number of ring members, i.e. $O(\log N)$. More concretely, this results in a signature size of between $36KB$ and $56KB$ as the size of the ring grows from 2^1 to 2^{12}. We present our results with comparisons to other lattice-based schemes in Table 3.

We now estimate the CPU cycles needed for our signatures based on a ring size of N. Based on Khuc et al. (2023), we choose the optimal value $M = 250$ commitments each requiring generating N HAWK keys and computing $2N-1$ SHA-256 hashes (for the Merkle Tree) and 1 more for the message $M||R$. For verification, we consider $M - K = 220$ reconstructions of Merkle Trees including HAWK key generation as above. We use benchmarks from the HAWK specification to estimate the overall proving and verification times. Note that we estimate the cycles needed to compute the proof by calculating the number of cycles for the HAWK key generations, as these dominate the computation time.

On Coffee Lake CPU architecture this amounts to the following numbers of clock cycles reported in the HAWK specification.

- $8.43 \times 10^6 \times N$ - HAWK key generations
- $8.9 \times 1024 \times N$ - SHA-256 hashes, 1024B inputs
- $232 \times 32 \times (N-1)$ - SHA-256 hashes, 32B inputs
- 1 - SHA-256 hash, $(N \times 1024B) + M$ input

We also note that other implementations exist that can optimise better for signing and verifying times, at the expense for more RAM usage. However we opt for the more balanced benchmarks, presented in Table 2.

6.1 Online/Offline and Amortisation

We observe that the only part of our signing algorithm that is dependent on the message is the final HAWK signature, using the fresh signing key TO corresponding to the public key Q. Thus, the zero-knowledge proof that proves knowledge of the isomorphism between a public key in the ring and Q can be computed in advance and offline, for a given ring R. Thus the only online costs are that of creating a HAWK signature, which comprise 56.54×10^4 cycles, which on commodity hardware takes around 10^{-6} seconds. Similarly, if the verifier is known in advance then it can also verify the proof and pre-validate the ephemeral key Q before the signature is even generated. This is also the most costly part of the verification phase, and now the online section only takes 1.96×10^5 cycles and 10^{-5}s. These results are presented in Table 2.

Another interesting optimisation is that our ring signature supports amortisation of signatures for a fixed ring and under the same pseudonym Q (i.e. implicitly linked signatures). Since Q will be verified as part of verifying the first signature received, it remains valid for all subsequent signatures also using Q. For applications where this could apply, this drastically improves efficiency and presents a amortised signature size of $555\text{B} + (40/M)\text{KB}$ for M signatures. For even 10 signatures, this gives an amortised signature

Table 2: Online/Offline comparison with LIP-based related work for $N = 2^6$ ring members, targeting NIST I security level. Sign and Verify values are estimated number of cycles from benchmarks on Coffee Lake architecture CPUs.

		Sign		Verify	
Scheme	**Sig Size**	**Offline**	**Online**	**Offline**	**Online**
Khuc et al. (2023)	40KB	–	1.35×10^{11}	–	1.19×10^{11}
This Work	40KB + 555B	1.35×10^{11}	56.54×10^{4}	1.19×10^{11}	1.96×10^{5}

Table 3: Comparison with lattice-based (claimed) linkable ring signatures with various ring sizes (N). All values in KB. No amortisation has been included in these calculations.

	N				
Scheme	2	2^3	2^6	2^{12}	2^{21}
RAPTOR (Lu et al., 2019)	2.5	10	81	5161	–
Falafl (Beullens et al., 2020)	29	30	32	35	39
Khuc et al. (2023)	35	37	40	45	54
This Work	36	38	41	46	55

size of $4.56KB$. This will have similar improvements for signing and verification times.

This could prove useful in privacy-preserving applications where potentially many signatures are expected for a single pseudonym, such as in medical data sharing (Yang et al., 2025) where pseudonyms link patients within a data set, and signatures could add information to the data set as tests and procedures are performed.

7 Conclusion

In this paper we have proposed the first secure linkable ring signature based on the lattice isomorphism problem. Our scheme satisfies strong notions of user controlled linkability, as well as standard unforgeability and anonymity properties. When instantiated with NIST candidate signatures and short proof techniques, our construction enjoys complexity logarithmic in the ring size, and more concretely for a ring of size, e.g. 64, is 41KB, which can be split into 40KB of offline computation and 555B of online computation. This optimisation is present in both signature generation and verification. Finally we observed that within implicitly linkable signatures, our scheme can also be amortised, reducing the signing and verification computation by a factor linear in the number signatures.

Acknowledgments

C. C. Drăgan is partially supported by TrustVote – EPSRC grant EP/Y020529/1, AP4L - EPSRC grant EP/W032473/1, CONNECT - Horizon Europe Guarantee 10043730 and EU Horizon grants 101069688, REWIRE - Horizon Europe Guarantee 10043743 and EU Horizon grants 101070627.

References

Bender, A., Katz, J., & Morselli, R. (2009). Ring signatures: Stronger definitions, and constructions without random oracles. *Journal of Cryptology*. https://doi.org/10.1007/s00145-007-9011-9

Beullens, W., Katsumata, S., & Pintore, F. (2020). Calamari and falafl: Logarithmic (linkable) ring signatures from isogenies and lattices. *ASIACRYPT 2020*. https://doi.org/10.1007/978-3-030-64834-3%5C_16

Brakerski, Z., Langlois, A., Peikert, C., Regev, O., & Stehlé, D. (2013). Classical hardness of learning with errors. *STOC 2013*. https://doi.org/10.1145/2488608.2488680

Budroni, A., Chi-Domínguez, J., & Franch, E. (2025a). Don't use it twice: Reloaded! on the lattice isomorphism group action. *IACR Communications in Cryptology*. https://doi.org/10.62056/ay76chdj

Budroni, A., Chi-Domínguez, J.-J., & Franch, E. (2025b). Don't use it twice: Reloaded! on the lattice isomorphism group action. *IACR Communications in Cryptology*.

Chen, L., Kudla, C., & Paterson, K. G. (2004). Concurrent signatures. *EUROCRYPT 2004*. https://doi.org/10.1007/978-3-540-24676-3%5C_18

Ducas, L., Postlethwaite, E. W., Pulles, L. N., & Woerden, W. v. (2022). Hawk: Module lip makes lattice signatures fast, compact and simple. *ASIACRYPT 2022*.

Ducas, L., & van Woerden, W. P. J. (2022). On the lattice isomorphism problem, quadratic forms, remarkable lattices, and cryptography. *EUROCRYPT 2022*. https://doi.org/10.1007/978-3-031-07082-2%5C_23

Esgin, M. F., Zhao, R. K., Steinfeld, R., Liu, J. K., & Liu, D. (2019). Matrict: Efficient, scalable

and post-quantum blockchain confidential transactions protocol. *CCS 2019*. https://api.semanticscholar.org/CorpusID:207957390

Even, S., Goldreich, O., & Micali, S. (1989). Online/off-line digital schemes. *CRYPTO 1989*. https://doi.org/10.1007/0-387-34805-0%5C_24

Fiore, D., Garms, L., Kolonelos, D., Soriente, C., & Tucker, I. (2022). Ring signatures with user-controlled linkability. *ESORICS 2022*. https://doi.org/10.1007/978-3-031-17146-8%5C_20

Hui, X., & Chau, S. C.-K. (2024). Llring: Logarithmic linkable ring signatures with transparent setup. *ESORICS 2024*.

Iribarnegaray, I. D., Gregor, V., & Leal, F. L. (2026). A note on the unsuitability of LIGA for linkable ring signatures: The perils of non-commutativity. https://eprint.iacr.org/2026/671

Jakobsson, M., Sako, K., & Impagliazzo, R. (1996). Designated verifier proofs and their applications. *EUROCRYPT 1996*. https://doi.org/10.1007/3-540-68339-9%5C_13

Khuc, T. X., Ta, T. A., Susilo, W., Duong, D. H., Guo, F., Fukushima, K., & Kiyomoto, S. (2023). Logarithmic-size (linkable) ring signatures from lattice isomorphism problems. *SPACE 2023*. https://doi.org/10.1007/978-3-031-51583-5%5C_13

Liu, J. K., Au, M. H., Susilo, W., & Zhou, J. (2009). Online/offline ring signature scheme. *Information and Communications Security*.

Liu, J. K., & Wong, D. S. (2005). Linkable ring signatures: Security models and new schemes. *ICCSA 2005*. https://doi.org/10.1007/11424826%5C_65

Liu, Z., Nguyen, K., Yang, G., Wang, H., & Wong, D. S. (2019). A lattice-based linkable ring signature supporting stealth addresses. *ESORICS 2019*. https://doi.org/10.1007/978-3-030-29959-0_35

Lu, X., Au, M. H., & Zhang, Z. (2019). Raptor: A practical lattice-based (linkable) ring signature. *ACNS 2019*. https://doi.org/10.1007/978-3-030-21568-2_6

Mezzadri, F. (2007). How to generate random matrices from the classical compact groups. *Notices of the American Mathematical Society*.

Micciancio, D., & Regev, O. (2004). Worst-case to average-case reductions based on gaussian measures. *FOCS 2004*. https://doi.org/10.1109/FOCS.2004.72

National Institute of Standards and Technology. (2015, August). *Secure hash standard (SHS)* (tech. rep.). NIST. https://doi.org

Rivest, R. L., Shamir, A., & Tauman, Y. (2001). How to leak a secret. *ASIACRYPT 2001*.

Tsang, P. P., & Wei, V. K. (2005). Short linkable ring signatures for e-voting, e-cash and attestation. *ISPEC 2005*. https://doi.org/10.1007/978-3-540-31979-5%5C_5

Wang, Z., Yang, P., Yang, D., Liu, X., & Pei, H. (2026). V2g-ringchain: Blockchain-enabled linkable ring signature scheme for vehicle-to-grid security and privacy. *Journal of Information Security and Applications*. https://www.sciencedirect.com/science/article/pii/S2214212626000013

Xue, Y., Lu, X., Au, M. H., & Zhang, C. (2024). Efficient linkable ring signatures: New framework and post-quantum instantiations. *ESORICS 2024*. https://doi.org/10.1007/978-3-031-70903-6%5C_22

Yang, Y., Liu, M., Wang, L., Pu, Y., Zheng, R., Wu, Q., & Guo, J. (2025). A ring signature scheme with linkability and traceability for blockchain-based medical data sharing system. *Peer-to-Peer Networking and Applications*.

Yuen, T. H., Esgin, M. F., Liu, J. K., Au, M. H., & Ding, Z. (2021). Dualring: Generic construction of ring signatures with efficient instantiations. *CRYPTO 2021*.

Zhang, H., Zhang, F., Tian, H., & Au, M. H. (2018). Anonymous post-quantum cryptocash. *Financial Cryptography and Data Security*.