INTERNATIONAL JOURNAL OF INFORMATION SECURITY SCIENCE
M. S. Garjan et al., Vol.12, No.1, pp. 32-57.
https://doi.org/10.55859/ijiss.1187756

Supersingular Isogeny-based Ring Signature

Maryam Sheikhi Garjan[1], N. Gamze Orhon Kılıç[2], Murat Cenk[2]

[1]IT University of Copenhagen, Copenhagen, Denmark
[2]Middle East Technical University, Ankara, Turkey
Corresponding Author: mashe@itu.dk

Research Paper Received: 14.10.2022 Revised: 04.01.2023 Accepted: 29.01.2023

Abstract—The increasing demand for secure and anonymous transactions raises the popularity of ring signatures, which is a digital signature scheme that allows identifying a group of possible signers without revealing the identity of the actual signer. This paper presents efficient supersingular isogeny-based ring signature and linkable ring signature schemes that will find potential applications in post-quantum technologies. We develop the ring signature scheme by applying the Fiat-Shamir transform on the sigma protocol for a ring which we obtain from the supersingular isogeny-based interactive zero-knowledge identification scheme by adapting the scheme for a ring. We also extend our ring signature protocol with an additional parameter, i.e., a tag that provides to detect if a signer issues two signatures concerning the same ring by preserving anonymity and linkable anonymity. The signature size of our ring signature protocols increases logarithmically in the size of the ring thanks to the Merkle trees. We show the security proofs and efficiency analyses of the protocols offered. Moreover, we provide the implementation results of the supersingular isogeny-based ring signature, which offers small signature sizes for NIST post-quantum security levels.

Keywords—linkable ring signature, post-quantum cryptography, ring signature, supersingular isogeny

1. Introduction

Rivest, Shamir, and Kalai introduced the *ring signatures* at ASIACRYPT [1] in 2001. A ring signature is a digital signature scheme produced by a member of a *ring* (a group of people), which does not reveal the signer's identity. Ring signatures are very similar to group signatures. However, they differ from group signatures in some points, such that there are no group managers, coordination, setup, and revocation procedures in ring signatures. A signer can select a set of potential signers, including herself, and sign a message with her secret key and other signers' public keys. This scenario does not require the approval of the other signers.

Besides *correctness*, two main features must be satisfied in terms of security by a ring signature: *unforgeability* and *anonymity*. A ring signature scheme is unforgeable if that scheme does not allow anyone to generate a signature on behalf of an honest ring of signers without knowing the secret key of at least one member of the ring. For a given ring signature, anonymity is satisfied if no one can distinguish which member of the ring generated the signature, even with the information of all secret keys of the ring. Furthermore, there is no cooperation or group secret among the ring members in ring signature schemes. Therefore, choosing the ring members can be done in an ad-hoc way.

Whistleblowing was the original motivation of the ring signatures [1], where the leaking person's identity can be hidden by choosing a ring of people who have access to this specific leaked message while convincing the recipient about the authenticity of the leaked message. Recently, ring signatures have found many applications such as cryptocurrency technologies for secure and anonymous transactions and e-voting [2], [3]. For instance, in cryptocurrencies like *Monero*, known as a fungible currency, a user issues a ring signature on the transaction using a ring of public keys in the blockchain and generates a confidential transaction. A signer who generates a ring signature can hide her identity as an actual signer among the ring of public keys by ensuring that her identity is indistinguishable from other ring members' identities. In such applications, a public verification property that determines if the same signer has issued two signatures is crucial since it could prevent double-spending attacks and double-voting problems. *Linkable ring signatures* are introduced to address these problems. A linkable ring signature is an extension of the ring signatures that allows a verifier to determine whether the same signer has signed more than one message. In the e-voting schemes, as an application of linkable ring signatures, a voter can sign a vote concerning a ring of all eligible voters while using the linkability property. One can verify that each person in the ring has voted only once, and the voter's identity is protected by the anonymity property of the ring signatures.

Since 2001, a huge number of ring signature schemes on various hardness assumptions such as the integer factorization [1], [4], [5], discrete logarithm [2], [6], [7], [8], [9] and pairing-based [3], [10], [11], [12] have been proposed. The security of the pairing-based ring signatures could be proven without using a random oracle. Furthermore, efficient and short ring signatures that rely on pairing-based cryptography are introduced in [10], [13], [3]. In [6], [8], the signature size increases linearly in the size of the ring, and [11] gives a constant size ring signature, while the signature size in [7], [14] is logarithmic in the number of ring members. The ring signature sizes in [4], [5] based on RSA accumulators is independent of the ring size. Most recently, ring signatures that rely on the post-quantum assumptions like hash-based [15], [16] multivariate [17], [18] and (one-time) lattice-based [19], [20], [21], [22], [23] are introduced.

Recently, Beullens *et al.* presented linkable ring signature schemes in [20], based on logarithmic OR-proof with binary challenges for Commutative Supersingular Isogeny Diffie-Hellman (CSIDH) group action and Module Learning with Errors (MLWE) group action. The CSIDH group action is adapted from the Couveignes-Rostovtsev-Stolbunov scheme by substituting supersingular elliptic curves over $\mathbb{F}_p$ for ordinary elliptic curves to improve the efficiency of the scheme. The CSIDH group action is commutative since the subring of $\mathbb{F}_p$-rational endomorphisms is an order in an imaginary quadratic field. The security of the CSIDH-based linkable ring signature is based on the Group Action Inverse Problem (GAIP) and Squaring Decisional CSIDH (sdCSIDH) Problem. The best-known quantum algorithm to solve GAIP and its variants has subexponential complexity. Nevertheless, there is no ring signature scheme based on supersingular isogenies to the best of our knowledge. The design of the SIDH (Supersingular Isogeny Diffie-Hellman) scheme addressed the security weakness of the isogeny-based schemes by using supersingular elliptic curves defined over $\mathbb{F}_{p^2}$. The endomorphism rings of these curves are non-commutative and therefore provide exponential security. We should emphasize that SIDH is not similar to CSIDH in

security, construction, key size, and performance. SIDH has notable advantages over CSIDH by providing high security and computational efficiency.

This paper presents a post-quantum version of the sigma protocol for a ring that proves membership in the ring. In our sigma protocol, we apply the OR-proof with binary challenges for a group action proposed in [20] to the SIDH identification protocol given in [24], which does not follow the group action property. We give the proof of the correctness, *2-special soundness*, and *honest-verifier zero-knowledge (HVZK)* properties of the proposed protocol. Moreover, the fast-known quantum attacks against these assumptions are still exponential. Thus, we present a ring signature scheme based on the post-quantum assumptions, i.e., supersingular isogeny problems. The construction proposed in this paper provides a ring signature scheme, where the signature size grows logarithmically in the number of users in the ring. Also, we show that this scheme is correct, anonymous, and existentially unforgeable under an adaptive chosen message attack in the random oracle model. Furthermore, we extend our construction to the linkable ring signatures, where the ring signatures that are signed using the same secret signing key can be linked. We give the implementation results of the supersingular isogeny-based ring signature.

Supersingular isogeny-based ring signature (SIRS) protocol offered in this paper provides small signature sizes in most cases compared to other post-quantum ring signature schemes in the literature. We compare SIRS with the schemes in a parameter set that achieves NIST security levels. For a ring with $n = 2^3$, our ring signature size increases by 13.5 KB compared to 20 KB of the Falafl given in [20], transmission from the post-quantum security of NIST 1 to NIST 2. For the same ring size, namely $n = 2^3$ and $n = 2^6$, our scheme roughly saves 11 KB and 8 KB, respectively, in the security level NIST 2. For SIRS scheme, a signature for ring size $n = 1024$ is approximately 27% (17% Falafl) larger than a signature for ring size $n = 2^3$. However, our signature size for $n = 1024$ is 51 KB instead of 54 KB in the security level NIST 2. SIRS provides the smallest signature size for the security level NIST 5. Furthermore, SIRS offers efficient signature generation and verification, making it preferable for post-quantum applications.

The rest of the paper is organized as follows: In Section 2, we provide the background required by the offered schemes. In Section 3, we propose the supersingular isogeny-based sigma protocol and Merkle tree adaptation, followed by supersingular isogeny-based ring signatures in Section 4. We show the linkable version of the supersingular isogeny-based ring signature in Section 5. We present the efficiency analyzes and implementation results in Section 6 and conclude our paper in Section 7. In Appendices, we give the algorithms of supersingular isogeny-based sigma protocol for a ring, ring signature, and linkable ring signature.

2. Background

This section briefly provides some required information related to the elliptic curve isogenies [25], [24], [26], computational problems of supersingular isogenies [24], [27], [28], ring signatures [2], [10], [14], linkable ring signatures [2], [3], [9], and supersingular isogeny-based zero-knowledge proofs [27], [29], [30].

2.1. Elliptic Curve Isogenies

We consider the elliptic curves defined over a finite field $\mathbb{F}_q$ of characteristic $p > 3$. For an elliptic

curve $E : y^2 = x^3 + ax + b$ over $\mathbb{F}_q$, the j-invariant of E is denoted by

$$j(E) = 1728\frac{4a^3}{4a^3 + 27b^2}.$$

For a given $j \in \mathbb{F}_q$ with $j \neq 0$ and $j \neq 1728$, there is an elliptic curve,

$$y^2 = x^3 + \frac{3j}{1728 - j}x + \frac{2j}{1728 - j},$$

whose j-invariant is j. Two elliptic curves E and E' are isomorphic over $\overline{\mathbb{F}}_q$ if only if they have the same j-invariant. Isomorphism maps between elliptic curves are invertible algebraic maps over algebraic closure $\overline{\mathbb{F}}_q$ and can be efficiently computed.

The n-torsion group of E, denoted by $E[n]$, contains the set of all points $P \in E(\overline{\mathbb{F}}_q)$ such that $nP = \mathcal{O}_E$, where $\mathcal{O}_E$ is the identity element. For n, with $p \nmid n$, we have $E[n] \cong \mathbb{Z}/n\mathbb{Z} \oplus \mathbb{Z}/n\mathbb{Z}$.

The elliptic curves defined over a field of characteristic p can be classified according to the structure of their p-torsion group. The elliptic curves with $E[p] \simeq \mathbb{Z}/p\mathbb{Z}$ are called *ordinary* while the curves $E[p] \simeq \mathcal{O}$ are called *supersingular*.

An *isogeny* $\varphi : E \to E'$ is a non-constant morphism from E to E' that preserves the identity element. The *degree of an isogeny* is its degree as a morphism. If φ is separable, then $deg\varphi = \#ker(\varphi)$. The curves E and E' are *isogenous* if there is a separable isogeny between them. Due to Tate's theorem, E and E' are isogenous over $\mathbb{F}_q$ if and only if $\#E(\mathbb{F}_q) = \#E'(\mathbb{F}_q)$. The isogeny φ can be explicitly obtained by using Vélu's formulae [31]. An isogeny of degree d is called a d-isogeny. Every isogeny of smooth degree $d > 1$ can be computed as a composition of isogenies of prime degree $d = \prod_{i=1}^{m} \ell_i^{e_i}$ over $\overline{\mathbb{F}}_q$ where ℓ_i are small primes and e_i are integers.

An isogeny is a group homomorphism and can be uniquely identified with its kernel (up to isomorphism). Given $G \subseteq E$, there exists a unique curve E_G (up to isomorphism) and a unique separable isogeny (up to automorphism of E) $\varphi_G : E \to E_G \cong E/G$ such that $ker(\varphi_G) = G$. For a given prime ℓ, there exists exactly $\ell + 1$ cyclic subgroups of order ℓ that each defines different ℓ-isogenies. $\Phi_\ell(x, y) \in \mathbb{Z}[x, y]$ is a symmetric modular polynomial of degree $\ell + 1$ in both x and y, and $\Phi_\ell(j_1, j_2) = 0$ if and only if there is an ℓ-isogeny between two elliptic curves with j-invariants j_1 and j_2. Moreover, for a given j, the roots of the univariate equation $\Phi_\ell(x, j) = 0$ are the j-invariants of curves which are ℓ-isogenous to curves with j-invariant j. For each ℓ-isogeny $\varphi : E \to E'$, there exists a unique *dual* ℓ-isogeny $\widehat{\varphi} : E' \to E$ such that $\widehat{\varphi} o \varphi = [\ell]$ gives the multiplication-by-ℓ map on E and $\varphi o \widehat{\varphi} = [\ell]$ gives the multiplication-by-ℓ map on E'.

An *endomorphism* is an isogeny from E to itself. The set of all endomorphisms of the elliptic curve E, including the zero map, is denoted by $End(E)$. Moreover, it has a ring structure under point-wise addition and composition operations. The $End(E)$ over the algebraic closure field is isomorphic with an order in a quadratic imaginary field or a maximal order in a quaternion algebra. An elliptic curve whose $End(E)$ is an order in a quadratic imaginary field is called *ordinary*. The curve with $End(E)$ as a maximal order in a quaternion algebra is called the *supersingular* elliptic curve. Up to isomorphism, all supersingular elliptic curves over the finite field $\mathbb{F}_q$ of characteristic p can also be defined over $\mathbb{F}_{p^2}$. Indeed, the motivation for using the supersingular isogenies in cryptography is based on the hardness of computing the endomorphism of a randomly chosen supersingular elliptic curve. The best quantum algorithm to solve this problem has $O(p^{1/4})$ running time with only a quadratic improvement over classical algorithms.

2.2. Computational Problems of Supersingular Isogenies

Let $p = \ell_1^{e_1}\ell_2^{e_2}f \pm 1$ be a prime number where $\ell_1 \neq \ell_2$ are small primes, $\ell_1^{e_1} \approx \ell_2^{e_2}$, and f is an integer cofactor. Let E be a supersingular elliptic curve over $\mathbb{F}_{p^2}$. Fix $\{P_1, Q_1\}$ and $\{P_2, Q_2\}$ as bases of torsion groups $E[\ell_1^{e_1}]$ and $E[\ell_2^{e_2}]$, respectively. Computing the endomorphism ring of E is called the *endomorphism ring problem*. We state the computational problems that form security assumptions of the supersingular isogeny-based protocols given in [24], [27].

- Let m_1 and m_2 are randomly chosen integers modulo $\ell_1^{e_1}$, and not both divisible by ℓ_1. Let $\varphi : E \to E'$ be an $\ell_1^{e_1}$-isogeny whose kernel generated by $R = [m_1]P_1 + [m_2]Q_1$. For given $\{E', \varphi(P_2), \varphi(Q_2)\}$, *computational supersingular isogeny (CSSI)* problem is to compute a generator of the kernel φ.
- Let $\varphi : E \to E'$ and $\psi : E \to E''$ be secret isogenies whose kernels are generated by random points $R \in \langle P_1, Q_1\rangle$ and $S \in \langle P_2, Q_2\rangle$, respectively. *Supersingular Computational Diffie-Hellman (SSCDH)* problem is finding the j-invariant of $E/\langle R, S\rangle$ for given $\{E', E'', \varphi(P_2), \varphi(Q_2)\}, \{\psi(P_1), \psi(Q_1)\}$.
- Let $\varphi : E \to E'$ and $\psi : E \to E''$ be isogenies whose kernels are generated by random points $R \in E[\ell_1^{e_1}] = \langle P_1, Q_1\rangle$ and $S \in E[\ell_2^{e_2}] = \langle P_2, Q_2\rangle$, respectively. One of the following tuple is sampled with probability $1/2$:

  $$(E', E'', \{\varphi(P_2), \varphi(Q_2)\}, \{\psi(P_1), \psi(Q_1)\}, E/\langle R, S\rangle),$$
  $$(E', E'', \{\varphi(P_2), \varphi(Q_2)\}, \{\psi(P_1), \psi(Q_1)\}, E/\langle T\rangle)$$

  where $T \in E[\ell_1^{e_1}\ell_2^{e_2}]$ and is randomly chosen. *Supersingular Decision Diffie-Hellman (SSDDH)* problem is to determine from which distribution this tuple is sampled.
- Let $\varphi : E \to E'$ be an isogeny whose kernel is generated by a secret point $R \in E[\ell_1^{e_1}] = \langle P_1, Q_1\rangle$. Suppose that $E[\ell_2^{e_2}] = \langle P_2, Q_2\rangle$ and $(E, E', P_2, Q_2, \varphi(P_2), \varphi(Q_2))$ are given. Consider the following distributions of (E, E'):
  - (E_1, E_1'), where $E_1 = E/\langle S\rangle$ generated by $S \in E[\ell_2^{e_2}]$ and $E_1' = E'/\langle\varphi(S)\rangle$.
  - (E_1, E_1'), where E_1 is a random curve and isogenous to E, and E_1' is generated by a random point $R' \in E_1[\ell_1^{e_1}]$.

  Decisional Supersingular Product (DSSP) problem is to determine from which distribution the tuple (E_1, E_1') is sampled.

The following two problems were stated in [32], [33], which are the modified versions of CSSI and SSDDH, respectively. There are no auxiliary image points in these problems, but one of the secret kernels is given.

- *Modified Computational Supersingular Isogeny (MCSSI)* problem is to compute E_{RS} for the given curves E_S, E_R, and the kernel $\langle S\rangle$.
- *Modified Supersingular Decision Diffie-Hellman (MSSDDH)* problem is to determine whether $E_T = E_{RS}$ for given E_S, E_R, E_T, and $\langle S\rangle$.

2.3. Ring Signatures

A ring signature (RS) scheme for given public parameters $pp(\lambda)$ consists of a triple of *PPT (probabilistic polynomial-time)* algorithms $(\mathsf{Kgen}, \mathsf{Sig}_{\mathsf{RS}}, \mathsf{Ver}_{\mathsf{RS}})$, for generating keys, signing a message, and verifying the ring signature respectively.

- $\mathsf{Kgen}(pp, rnd)$: Outputs public and secret keys (pk_i, sk_i) of user i with respect to pp and a random number rnd.

- $\mathsf{Sig}_{\mathsf{RS}}(pp, L_{pk}, sk_r, m)$: Let $L_{pk} = \{pk_1, pk_2, \ldots\}$ be a list of public keys. $\mathsf{Sig}_{\mathsf{RS}}$ takes a message m, the secret key sk_r of the signer, and L_{pk}. It selects a ring $\mathcal{R}$ with n members, including the signer's public key $pk_r \in \mathcal{R}$, and outputs a signature σ on message m with respect to the ring $\mathcal{R}$.
- $\mathsf{Ver}_{\mathsf{RS}}(pp, \mathcal{R}, \sigma, m)$: Takes a signature σ, message m, and a ring $\mathcal{R} = \{pk_1, \ldots, pk_n\}$ as input, outputs 1 for accepting and 0 for rejecting.

2.3.1 Security of Ring Signatures

The conditions correcness, anonymity, and unforgeability must be satisfied by a ring signature scheme.

A ring signature σ is said to satisfy the correctness condition if for every public information pp, $n = \mathsf{poly}(\lambda)$, message m, $\mathcal{R} \subseteq L_{pk}$ where $(pk_i, sk_i) \leftarrow \mathsf{Kgen}(pp, rnd_i)$ for every $i \in \{1, 2, \ldots, n\}$, the signature $\sigma \leftarrow \mathsf{Sig}_{\mathsf{RS}}(pp, L_{pk}, sk_r, m)$ for $pk_r \in \mathcal{R}$, $1 \leq r \leq n$ always holds $Pr[\mathsf{Ver}_{\mathsf{RS}}(pp, \mathcal{R}, \sigma, m) = 1] = 1$.

$\mathsf{G}_{\mathsf{RS}_{\mathsf{ano}}}(\mathcal{A}, \mathcal{C})$:

- $\mathcal{C}$ runs $\{pk_i, sk_i\} \leftarrow \mathsf{Kgen}_{\mathsf{RS}}(pp, rnd_i)$ with the random coins rnd_i for $i = 1, \ldots, n$.
- $\mathcal{C}$ sends $L_{pk} = \{pk_i\}_{i=1}^{n}$ and $rnd = \{rnd_i\}_{i=1}^{n}$ to $\mathcal{A}$ so that $\mathcal{A}$ can reconstruct each sk_i.
- $\mathcal{A}$ requests a signature on the query $(pk_{a0}, pk_{a1}, \mathcal{R}, m)$ where $pk_{a0}, pk_{a1} \in \mathcal{R} \subseteq L_{pk}$.
- $\mathcal{C}$ flips a random bit $\mathsf{b} \in \{0, 1\}$, computes the signature $\sigma \leftarrow \mathsf{Sig}_{\mathsf{RS}}(pp, \mathcal{R}, S_{i\mathsf{b}}, m)$, and sends σ to $\mathcal{A}$.
- $\mathcal{A}$ makes a guess b' and wins if $\mathsf{b}' = \mathsf{b}$.

Figure 1. Anonimity game for a ring signature.

A ring signature σ is called anonymous if for every public parameter pp, message m, and $n = \mathsf{poly}(\lambda)$, any PPT adversary $\mathcal{A}$ has at most negligible advantage in the anonymity game $\mathsf{G}_{\mathsf{RS}_{\mathsf{ano}}}$ between $\mathcal{A}$ and a challenger $\mathcal{C}$. The game is given in Figure 1

$\mathsf{G}_{\mathsf{RS}_{\mathsf{unf}}}(\mathcal{A}, \mathcal{C})$:

- $\mathcal{C}$ generates the key pairs $\{pk_i, sk_i\} \leftarrow \mathsf{Kgen}_{\mathsf{RS}}(pp, rnd_i)$ with the random coins rnd_i for $i = 1, \ldots, q$.
- $\mathcal{C}$ sends $L_{pk} = \{pki_i\}_{i=1}^{q}$ to $\mathcal{A}$ and initializes the empty sets L_C and Σ.
- After $\mathcal{A}$ recieves L_{pk}, the following conversation between $\mathcal{A}$ and $\mathcal{C}$ can start:
  - $\mathcal{A}$: makes the $(\mathsf{sign}, i, \mathcal{R}, m)$ request to get a signature of m for pk_i with respect to the ring $\mathcal{R}$.
  - $\mathcal{C}$: if $pk_i \in L_{pk}$, computes the signature $\sigma \leftarrow \mathsf{Sig}_{\mathsf{RS}}(pp, \mathcal{R}, sk_i, m)$, and returns it. $\mathcal{C}$ adds σ to Σ.
  - $\mathcal{A}$: requests the randomness generating pk_i with the query $(\mathsf{corrupt}, i)$.
  - $\mathcal{C}$: returns rnd_i to $\mathcal{A}$ and saves pk_i to L_C.
- Finally, $\mathcal{A}$ outputs the transcript $(\sigma^*, \mathcal{R}^*, m^*)$.
- $\mathcal{C}$ checks the following:
  - $\mathcal{R}^* \subseteq L_{pk} \backslash L_C$,
  - $(\mathsf{sign}, ., \mathcal{R}^*, m^*) \in \Sigma$,
  - $\mathsf{Ver}_{\mathsf{RS}}(pp, \sigma^*, m^*) = 1$.
- If all checks are successful, $\mathcal{C}$ returns 1.

Figure 2. Unforgeability game for a ring signature.

and shows that even in the case of full key exposure, a ring signature must leak no information about the identity of the exact signer.

A ring signature σ is called unforgeable under insider corruption if for every public parameter pp and $n = \mathsf{poly}(\lambda)$, any PPT adversary $\mathcal{A}$ has at most a negligible advantage in the unforgeability game $\mathsf{G}_{\mathsf{RS}_{\mathsf{unf}}}$ against a challenger $\mathcal{C}$ given in Figure 2. The advantage of $\mathcal{A}$ in the unforgeability game is denoted as $\xi_{\mathsf{unf}} = Pr[\mathcal{A}\ wins]$.

2.4. Linkable Ring Signatures

An extended version of the ring signatures with one additional property of having linkability is called a linkable ring signature (LRS). Linkability provides to determine whether the same ring member signs two signatures issued within the same ring without revealing the signer's identity.

A linkable ring signature consists of a PPT algorithm ($\mathsf{Lver}_{\mathsf{LRS}}$) which provides a linkable verifica-

tion and also three PPT algorithms that a ring signature scheme consists, as ($\mathsf{Kgen_{LRS}}$, $\mathsf{Sig_{LRS}}$, $\mathsf{Ver_{LRS}}$). For given two signatures σ_0 and σ_1, the algorithm $1/0 \leftarrow \mathsf{Lver_{LRS}}(\sigma_0, \sigma_1)$ outputs 1 if they are signed by the same secret key and 0 if not.

A linkable ring signature must satisfy correctness, linkability, *linkable anonymity*, *non-frameability* and unforgeability properties. Note that unforgeability can be obtained from linkability and non-frameability properties.

Let $n \geq 1$, $pp(\lambda)$ be the public keys, L_{pk} be the list of public parameters then for all message $m \in \{0,1\}^*$, ring $\mathcal{R} \subseteq L_{pk}$, and key pair (sk_r, pk_r) such that $pk_r \in \mathcal{R}$, the linkable ring signature satisfies correctness since $Pr\left[\mathsf{Ver_{LRS}}(pp, \mathcal{R}, \sigma, m) = 0 \mid \sigma \leftarrow \mathsf{Sig_{LRS}}(pp, L_{pk}, sk_r m)\right] \leq negl(\lambda)$.

Let $\mathcal{A}$ be any PPT algorithm, and pp and L_{pk} are given. $\mathcal{A}$ outputs a set $\{(\sigma_i, \mathcal{R}_i, m_i)\}_{i=1}^{n+1}$ such that for all $1 \leq i \leq n+1$, $\mathcal{R} \subseteq L_{pk}$ and $\mathsf{Ver_{LRS}}$ $(pp, \mathcal{R}, \sigma_i, m_i) = 1$. The linkable ring signature provides linkability since for all $i, j \in [1, n+1]$ with $i \neq j$, $Pr[\mathsf{Lver_{LRS}}(\sigma_i, \sigma_j) = 0 \mid \sigma_i, \sigma_j \in \{(\sigma_i, \mathcal{R}_i, m_i)\}_{i=1}^{n+1}] \leq negl(\lambda)$.

Let $\mathcal{A}$ be any PPT algorithm, pp, pk, $L = \{(pk_i, sk_i) \mid pk_i \in \mathcal{R} - \{pk_0, pk_1\}\}$ be given, and let $\sigma \leftarrow \mathsf{Sig_{LRS}}(pp, L_{pk}, sk_r, m)$ where $pk_r \in \{pk_0, pk_1\} \subseteq R$. Linkable ring signatures provide linkable anonymity since $|Pr\left[\mathcal{A}(m, \mathcal{R}, L, \sigma) = r\right] - 1/2| \leq negl(\lambda)$.

Let $\mathcal{A}$ be any PPT algorithm and is given pp, L_{pk}. Also, $\mathcal{A}$ has access to Squeri and Cqueri queries. Let $L_{\mathsf{Squeri}} = \{\sigma \leftarrow \mathsf{Squeri}(i, \mathcal{R}, m) \mid pk_i \in \mathcal{R}\}$ be a list of signing query and $L_{\mathsf{Cqueri}} = \{sk_i \leftarrow \mathsf{Cqueri}(i, pk_i)\}$ be a list of corruption query made by $\mathcal{A}$. Then, $\mathcal{A}$ outputs $(\sigma^*, \mathcal{R}^*, m^*)$ such that $\mathsf{Ver_{LRS}}(\sigma^*, \mathcal{R}^*, m^*) = 1$ and $(., \mathcal{R}^*, m^*) \notin L_{\mathsf{Squeri}}$. The linkable ring signature satisfies the non-frameability property since $Pr\left[\mathsf{Lver_{LRS}}(\sigma^*, \sigma) = 1 \mid \forall \sigma_{(i,\mathcal{R},m)} \in L_{\mathsf{Squeri}}, sk_i \notin L_{\mathsf{Cqueri}}\right] \leq negl(\lambda)$.

2.5. Supersingular Isogeny-Based Zero-Knowledge Proof

A supersingular isogeny-based zero-knowledge proof of identity is presented in [27]. This protocol is computationally zero-knowledge and works as follows: Assume that Peggy (prover) wants to prove to Victor (verifier) that she knows the secret kernel $\langle S \rangle$ of the isogeny $\varphi : E \rightarrow E_S$ without revealing it. Let $\ell_p = \ell_1^{e_1}$, $\ell_v = \ell_2^{e_2}$, f be a small integer, p be a prime such that $p = \ell_p \ell_v f \pm 1$, $E(\mathbb{F}_{p^2})$ be a supersingular elliptic curve. The points P, Q, S are on the curve E such that $E[\ell_v] = \langle P, Q \rangle$ and $S \in E[\ell_p]$. Let $\{p, E, E_S, P, Q, \varphi(P), \varphi(Q)\}$ be publicly known, and S be the secret information. Peggy selects a random cyclic subgroup $V \in E[\ell_v]$, computes isogenies $\psi : E \rightarrow E_V$ and $\psi' : E_S \rightarrow E_{SV}$, whose kernels are generated by V and $\varphi(V)$, respectively. Peggy then publishes E_V and E_{SV} as commitment. Victor chooses a random challenge $\mathsf{b} \in \{0,1\}$ and sends it to Peggy. Peggy responds with $\{V, \varphi(V)\}$ upon receiving the challenge $\mathsf{b} = 0$, or responds with $\psi(S)$ for challenge $\mathsf{b} = 1$. Victor accepts if the response generates the isogenies that connect the corresponding curves. For λ-bit security, this interactive process should be run λ times, and Peggy successfully proves her knowledge of the secret kernel S if the verifier accepts the responses of all λ times of interaction. An interactive zero-knowledge proof can be transformed into a non-interactive signature scheme as given in [29], [30].

3. Supersingular Isogeny-Based Sigma Protocol for a Ring

In this section, we propose a supersingular isogeny-based sigma protocol for a ring that forms the basis of the supersingular isogeny-based ring signature scheme given in Section 4. The proposed sigma protocol is derived from the interactive zero-knowledge proof of identity proposed by De Feo, Jao, and Plût [24]. This section presents the proposed sigma protocol in detail, proves its security, and provides a Merkle tree application.

3.1. Sigma Protocol for a Ring

Let $\mathcal{R}$ be a ring chosen by Peggy with n members and r be an integer with $1 \leq r \leq n$. Peggy wants to convince Victor that she knows a secret key $\langle S_r \rangle$ that generates one of the public keys (i.e., E_{S_r}) in $\mathcal{R}$, without revealing the secret key and the particular public key in the ring $\mathcal{R}$. A supersingular isogeny-based interactive zero-knowledge proof takes over $\mathcal{R}$ as follows:

1 For a security parameter λ, let the public parameters be a prime number $p = \ell_p \ell_v f \pm 1$ where $\ell_p \approx \ell_v$ are smooth numbers, a supersingular elliptic curve $E(\mathbb{F}_{p^2})$, two points P and Q that are the generators of the ℓ_v-torsion group $E[\ell_v]$.

2 Every user in the system has public and secret keys for given security parameter λ. For the i^{th} user, S_i is the secret key and (E_{S_i}, P_i, Q_i) is the public key where $S_i \in E[\ell_p]$, generating the kernel of a secret ℓ_p-isogeny $\alpha_i : E \to E_{S_i}$, and $P_i = \alpha_i(P)$, $Q_i = \alpha_i(Q)$ as the images of ℓ_v-torsion generators $\langle P, Q \rangle$.

3 Peggy picks a ring $\mathcal{R} = \{(E_{S_i}, P_i, Q_i)\}_{i=1}^{n}$ of n public keys where Peggy's public key $pk_r \in \mathcal{R}$. She chooses a random integer $\omega \in \mathbb{Z}/\ell_v\mathbb{Z}$, then computes $V = P + [\omega]Q \in E[\ell_v]$ and $\alpha_i(V) = P_i + [\omega]Q_i$ defining the kernels of the isogenies given in Figure 3. In this scheme, $\beta : E \to E/\langle V \rangle = E_V$ and $\beta_i : E_{S_i} \to E_{S_i}/\langle \alpha_i(V) \rangle = E_{S_iV}$ are ℓ_v-isogenies defined by V and $\alpha_i(V)$, respectively. Peggy applies a random permutation τ on $\big[j(E_V), j(E_{S_1V}), \ldots, j(E_{S_nV})\big]$ and obtains the commitment $X = [j_{i_1}, j_{i_2}, \ldots, j_{i_{n+1}}]$. She sends the commitment X to Victor.

4 Victor sends a challenge $\mathsf{b} \in \{0, 1\}$ to Peggy.

5 Peggy reveals the response resp, based on the challenge. If $\mathsf{b} = 1$ then, $\mathsf{resp} = (\omega, \tau)$. If $\mathsf{b} = 0$ then $\mathsf{resp} = (j(E_V), \beta(S_r))$ where $\langle \beta(S_r) \rangle$ is the kernel of the isogeny $\alpha'_r : E_V \to E_V/\langle \beta(S_r) \rangle = E_{VS_r}$.

6 If $\mathsf{b} = 1$ and $\mathsf{resp} = (\omega, \tau)$, Victor verifies whether ω generates the elliptic curve points of order ℓ_v that define the kernels of the isogenies (shown in Figure 4) $E \to E_{V'}$, and $E_{S_i} \to E_{S_iV'}$, for $1 \leq i \leq n$, respectively. Victor applies τ on $\big[j(E_{V'}), j(E_{S_1V'}), \ldots, j(E_{S_nV'})\big]$ and obtains $X' = [j'_{i_1}, j'_{i_2}, \ldots, j'_{i_{n+1}}]$. He accepts if $X' = X$, otherwise rejects. If $\mathsf{b} = 0$, Victor checks whether $\beta(S_r)$ has order ℓ_p and generates the isogeny illustrated in Figure 5, $E_V \to E_V/\langle \beta(S_r) \rangle = E_{VS'_r}$ and then accepts if $j(E_V), j(E_{VS'_r}) \in X$. He rejects otherwise. Note that $E_{S_rV} \simeq E/\langle S_r, V \rangle \simeq E/\langle S_r \rangle/\langle \alpha_r(V) \rangle \simeq E/\langle V \rangle/\langle \beta(S_r) \rangle$.

The sigma protocol does not leak any information about (E_{S_r}, r). The prover uses a permutation map, which hides the index of the elements in the commitment. Moreover, when the verifier sends $\mathsf{b} = 1$, the prover's response allows the verifier to compute all the commitments, and therefore, there is no leak of anonymity. When the verifier sends the challenge $\mathsf{b} = 0$, the prover's answer is an isogeny between two arbitrary curves (E_V, E_{S_iV}) in the commitment. Since the verifier does not know the isogeny that

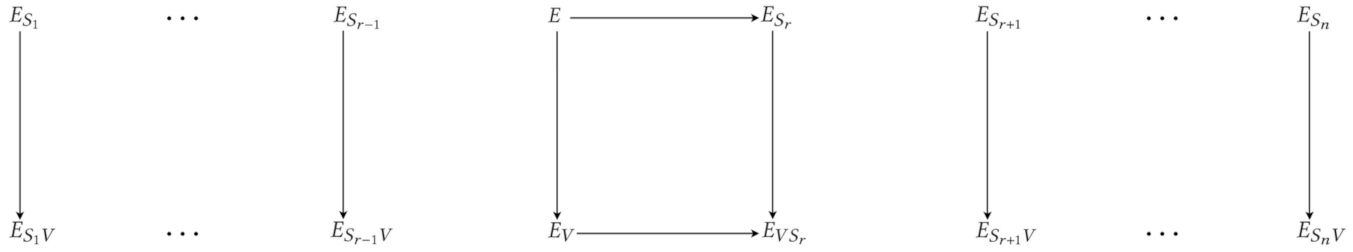

Figure 3. The isogenies that generate the curves of a commitment for supersingular isogeny-based sigma protocol for a ring.

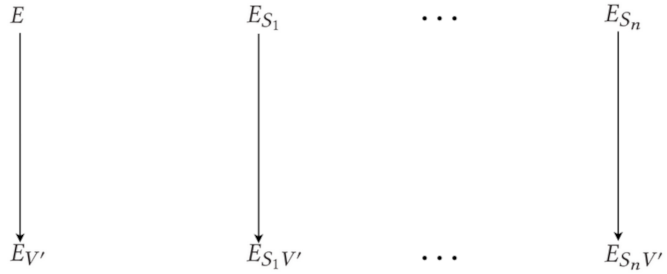

Figure 4. The isogenies that verifier computes to verify the commitment for $\mathsf{b} = 1$ of a supersingular isogeny-based sigma protocol for a ring.

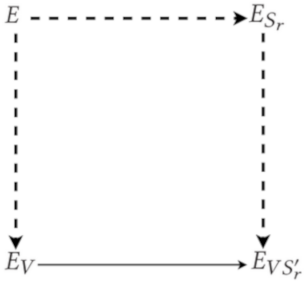

Figure 5. The isogenies that verifier computes to verify the commitment for $\mathsf{b} = 0$ of a supersingular isogeny-based sigma protocol for a ring.

connects these two curves to public curves in the ring, the response to this challenge is independent of the knowledge of (E_{S_r}, r).

Theorem 1 *The supersingular isogeny-based sigma protocol for a ring is complete, honest-verifier zero-knowledge (HVZK), and it satisfies 2-special soundness if the supersingular isogeny problems — DSSP and CSSI problems — are computationally hard.*

Proof: It is trivial to check the completeness. We shall prove that the scheme is HVZK, which means that one can simulate a real execution of the identification protocol for a given public key and a challenge without the knowledge of the secret key. To see this, consider the algorithm $(\mathsf{comm}, \mathsf{b}, \mathsf{resp}) \leftarrow \mathsf{Sim}(\mathcal{R}, \mathsf{b})$. For a given $\mathcal{R}$ and a challenge b, Sim works as follows: If $\mathsf{b} = 1$, choose random ω', τ' and compute the corresponding isogeny maps of degree ℓ_v with the public keys in $\mathcal{R}$. X' stores the j-invariants of the image curves. Sim outputs the transcript $(\mathsf{comm}, \mathsf{b}, \mathsf{resp}) = (X', 1, (\omega', \tau'))$. In this case, the output transcript is simulated correctly. If $\mathsf{b} = 0$, choose a curve E' (isogenous to E) and a random point $S' \in E'[\ell_p]$ where $E'' = E'/\langle S' \rangle$. X' holds the j-invariants of E', E'', and $n-1$ randomly chosen curves isogenous to E. Sim outputs the transcript $(\mathsf{comm}, \mathsf{b}, \mathsf{resp}) = (X', 0, (E', S'))$, however, in this case, X' is not distributed as a real execution. The computational assumption of DSSP implies that it is computationally hard to distinguish whether a transcript is simulated or is a transcript of a real

execution. Therefore, the scheme has computational zero-knowledge. 2-Special soundness follows from the following observation: For given two valid transcripts $(\mathsf{comm}, \mathsf{b}, \mathsf{resp}) = (X, 1, (\omega, \tau))$ and $(\mathsf{comm}, \mathsf{b}', \mathsf{resp}') = (X, 0, (E', S'))$ with respect to $\mathcal{R}$, it is possible to extract the secret key. Let $\beta : E \to E' = E/\langle V \rangle$ be the isogeny generated by the kernel $V = P + [\omega]Q$ and $\alpha' : E' \to E'' = E'/\langle S' \rangle$ be the isogeny generated by S'. With the knowledge of these two transcripts, one can compute $\beta'(S')$ that generates a secret kernel for one of the curves in the ring. Suppose that $\mathcal{A}$ is an adversary that can correctly respond for both $\mathsf{b} = 0$ and $\mathsf{b} = 1$ corresponding with X, then $\mathcal{A}$ can solve an instance of the CSSI problem. □

Assume that Peggy does not know any S_i that generates one of the public keys in $\mathcal{R}$ and tries to cheat Victor into that she is a member of $\mathcal{R}$. She can select a random number $\omega \in \mathbb{Z}/\ell_v\mathbb{Z}$ and obtains a commitment X, only with the knowledge of public parameters. In this scenario, if Victor sends $\mathsf{b} = 1$, then Peggy can send a valid response, but if Victor sends $\mathsf{b} = 0$ since she does not know any of the secret keys, she cannot compute a valid kernel $\langle S_i \rangle$ that generates one of the curves in commitment X. Conversely, Peggy can choose a random number $\omega \in \mathbb{Z}/\ell_v\mathbb{Z}$, compute $V = P + [\omega]Q$ and E_V. Then, she selects random points on E_V to generate X. If Victor sends $\mathsf{b} = 0$, Peggy's response $j(E_V), \beta(S_i)$ will convince Victor since X includes $E_V/\langle \beta(S_i) \rangle$. If Victor sends $\mathsf{b} = 1$, then Peggy has to send ω, which generates the kernels of the curves in X; however, the commitment does not include the curves generated by the public keys in $\mathcal{R}$. Therefore, Victor does not accept Peggy's response. For both of these scenarios, Peggy can cheat with the probability of $1/2$. So, the sigma protocol should be repeated until Victor is convinced that Peggy is honest.

3.2. Reducing the Size of Commitment Using Merkle Tree

The size of the commitment in Section 3.1 is large. To reduce the size of the commitment, we apply the Merkle tree to the commitment set X in each iteration of the sigma protocol for $\mathcal{R}$. We set a Merkle tree on commitment $X = [j_{i_1}, j_{i_2}, \ldots, j_{i_{n+1}}]$ whose leaf nodes are $\left[\mathsf{H}(j_{i_1}), \mathsf{H}(j_{i_2}), \ldots, \mathsf{H}(j_{i_{n+1}})\right]$ where H is a hash function. Internal nodes in the tree are hash values of a concatenation of two hashes (their two children). The root of the Merkle tree (the top hash) contains the hash of the entire tree. In order to prove that j_{i_*} is a leaf node of the Merkle tree M for a given $\mathsf{Root}(M)$, an ordered path that contains the sibling node of j_{i_*} and other internal nodes are needed. This path has a logarithmic size in the number of leaf nodes.

As an example, Figure 6 illustrates the construction of a Merkle tree where $X = [j_{i_1}, j_{i_2}, \ldots, j_{i_8}]$ is the permuted j-invariants of the curves $E_V, E_{S_1V}, \ldots, E_{S_7V}$. One can obtain the path of a single node by following the shortest path from the root node to the specific node. For instance, $\mathsf{Path}(j_{i_6}) = (h_5, h_{78}, h_{1234})$.

We slightly modify the sigma protocol for $\mathcal{R}$ so that the prover only reveals a Merkle tree root of X as a commitment. The changes in each step of the sigma protocol are as follows: Peggy applies the step 3 of sigma protocol given in Section 3.1, and generates a Merkle tree M on $X = [j_{i_1}, j_{i_2}, \ldots, j_{i_{n+1}}]$. Then, sends $\mathsf{Root}(M)$ to Victor. Victor sends a challenge $\mathsf{b} \in \{0, 1\}$ to Peggy. If $\mathsf{b} = 1$, Peggy reveals the response $\mathsf{resp} = (\omega, \tau)$. Victor applies τ on $\left[j(E_{V'}), j(E_{S_1V'}), \ldots, j(E_{S_nV'})\right]$ generated by ω and gets $X' = [j'_{i_1}, j'_{i_2}, \ldots, j'_{i_{n+1}}]$. He constructs the Merkle tree M' of X', and obtains $\mathsf{Root}(M')$. He accepts if $\mathsf{Root}(M') = \mathsf{Root}(M)$. If $\mathsf{b} = 0$, the response is modified as $\mathsf{resp} =$

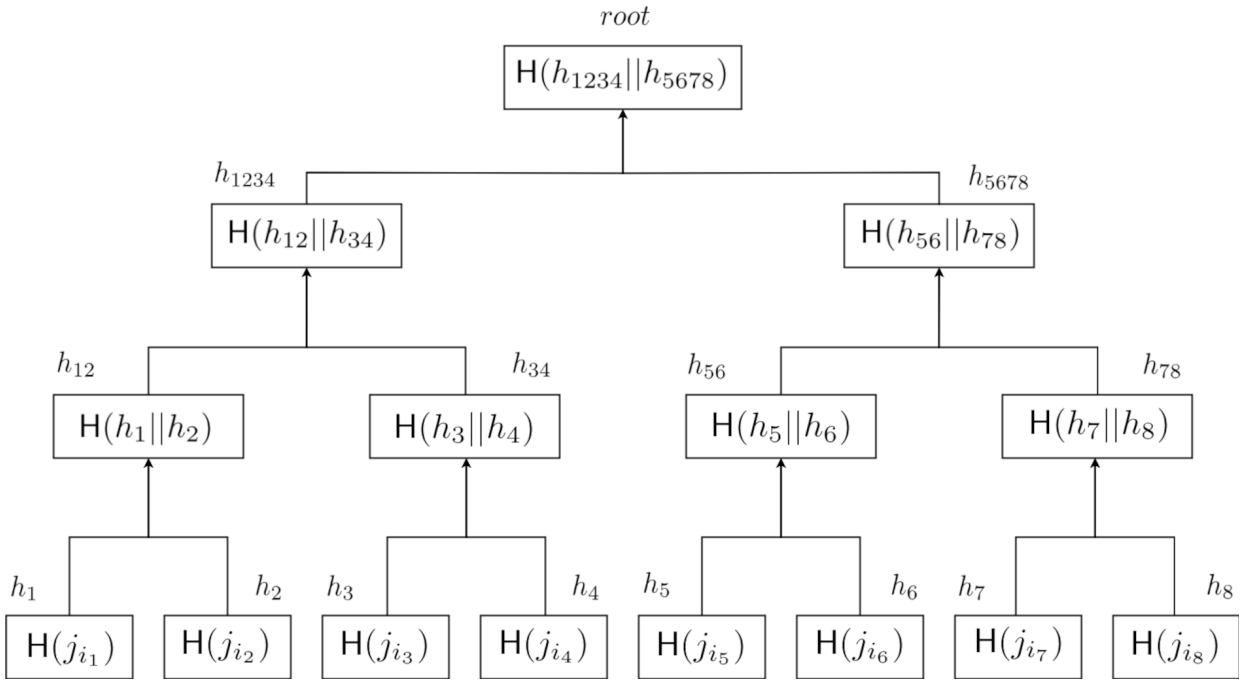

Figure 6. A Merkle tree constructed on a commitment list with 8 j-invariants.

$(j(E_V), \beta(S_r), \mathsf{Path}(j_r))$ where j_r denotes the j-invariant of $E_{S_r}V$. Victor first computes $E_{VS'_r}$ from the knowledge $(j(E_V), \beta(S_r))$. Then, by using the given $\mathsf{Path}(j_r)$, $\mathsf{H}(j(E_{VS'_r}))$ recovers $\mathsf{Root}(M')$. Victor accepts it if $\mathsf{Root}(M') = \mathsf{Root}(M)$.

4. Supersingular Isogeny-Based Ring Signature

In this section, we describe a supersingular isogeny-based ring signature which is obtained by applying Fiat-Shamir transform to the sigma protocol given in Section 3. We give the proof of correctness, anonimity, and unforgeability of our signature scheme which is the non-interactive version of the sigma protocol; where we use random oracle to substitue verifier's random challenges. Therefore, the proofs are shown in random oracle model.

Let the set of public parameters be $pp = \{p, E, P, Q, \mathsf{H}\}$ for given security parameter λ, where H is a hash function with output size $q = O(\lambda)$. The supersingular isogeny-based ring signature protocol uses the same key generation given in Section 3.1 and the rest works as follows:

- $\mathsf{Sig}_{\mathsf{RS}}(pp, L_{pk}, S_r, m)$: Let L_{pk} be the list of public keys, S_r be Peggy's secret key defining her public key (E_{S_r}, P_r, Q_r), and let Peggy choose a ring $\mathcal{R} = \{(E_{S_i}, P_i, Q_i)\}_{i=1}^{n} \subseteq L_{pk}$ such that $(E_{S_r}, P_r, Q_r) \in \mathcal{R}$. Peggy generates the ring signature by running the sigma protocol for a ring q times. The k^{th} iteration of the protocol is as follows:
  - Select a random integer $\omega_k \in \mathbb{Z}/\ell_v\mathbb{Z}$ and the random permutation $\tau_k(\lambda)$.
  - Compute $V_k = P + [\omega_k]Q$ and the corresponding ℓ_v-isogeny $\beta_k : E \to E_{V_k}$.
  - By using the public keys $\mathcal{R}$, compute the isogenies $\beta_{k1}, \beta_{k2}, \ldots, \beta_{kn}$ where $\beta_{ki} : E_{S_i} \to E_{S_iV_k}$, generated by $\alpha_i(V_k) = P_i + [\omega_k]Q_i$ of degree ℓ_v, for $i = 1, 2, \ldots, n$.
  - Obtain the permuted list $X_k = [j_{i_1}, \ldots j_{i_{n+1}}]$

by gathering the j-invariants of the commitment curves (i.e., E_{V_k} and $\{E_{S_iV_k}\}_{i=1}^{n}$) and applying τ_k on it.

– Generate the Merkle tree M_k on X_k and set $\mathsf{root}_k = \mathsf{Root}(M_k)$.

After collecting all root_k values for $k = 1, \ldots, q$, Peggy computes $h = \mathsf{H}(m, \mathsf{root}_1, \mathsf{root}_2, \ldots, \mathsf{root}_q)$ where $m \in \{0,1\}^*$ is the message and $h \in \{0,1\}^q$ is the output of H which holds the challenges of the ring signature. Let z be the verification key and z_k be the k^{th} element of z, for $k = 1, \ldots, q$. If $h_k = 1$, Peggy sets $z_k = (\omega_k, \tau_k)$, otherwise $z_k = (j(E_{V_k}), \beta_k(S_r), \mathsf{Path}(j_{rk}))$ where $j_{rk} = j(E_{S_rV_k})$. The signature is $\sigma = (z, \mathcal{R})$.

- $\mathsf{Ver}_{\mathsf{RS}}(pp, \sigma, m)$: Victor can recover each root_k by using the information given by z_k, for $1 \leq k \leq q$. First, he extracts the bits of $h = \mathsf{H}(m, \mathsf{root}_1, \mathsf{root}_2, \ldots, \mathsf{root}_q)$ according to the size of z_k, since the size of z_k differs for $h_k = 1$ and $h_k = 0$. If Victor obtains $z_k = (\omega_k, \tau_k)$ and so $h_k = 1$, he computes $V'_k = P + [\omega_k]Q$, $V'_{ki} = P_i + [\omega_k]Q_i$ and the isogenies $\beta'_k : E \to E_{V'_k}$, $\beta'_{ki} : E \to E_{S_iV'_k}$ with the kernels $\langle V'_k \rangle$ and $\langle V'_{ki} \rangle$. Victor obtains $X'_k = [j'_{i_1}, \ldots, j'_{i_{n+1}}]$, by applying τ_k on the commitment list including the j-invariants of $E_{V'_k}$, $\{E_{S_iV'_k}\}_{i=1}^{n}$. He constructs the Merkle tree M'_k of X'_k, and finds $\mathsf{root}'_k = \mathsf{Root}(M'_k)$. In the case $h_k = 0$, z_k contains a kernel $\beta_k(S_r)$ of an ℓ_p-isogeny and $j(E_{V_k})$, so Victor can compute $E_{V_k} \to E_{V_kS'_r} = E_{V_k}/\langle \beta_k(S_r) \rangle$. $\mathsf{Path}(j_{rk})$ is also included in z_k, thus Victor obtains root'_k using $\mathsf{H}(j(E_{V_kS'_r}))$ and $\mathsf{Path}(j_{rk})$. After collecting each root'_k, he computes $h' = \mathsf{H}(m, \mathsf{root}'_1, \mathsf{root}'_2, \ldots, \mathsf{root}'_q)$ then, compares h and h'. Victor accepts if $h = h'$.

Theorem 2 *The supersingular isogeny-based ring signature is correct, anonymous, and existentially unforgeable under an adaptive chosen message attack in the random oracle model if the problems CSSI and DSSP are computationally hard, and the sigma-protocol for a ring given in Section 3 is correct, 2-special sound, and honest-verifier zero-knowledge.*

Proof: The correctness of the ring signature produced by a signer who knows a secret key in the ring follows from the correctness of the sigma protocol for a ring since we run it in q parallel times, and the commitments are reconstructed from the verification keys of the signature.

We prove the anonimity and unforgeability by showing that a PPT adversary $\mathcal{A}$ has at most a negligible advantage against a challenger $\mathcal{C}$ in the games given in Figure 1 and Figure 2.

Let $pp = \{p, E, P, Q, \mathsf{H}, q\}$ be the public parameters for given security parameter λ where $q = O(\lambda)$ for the games $\mathsf{G}_{\mathsf{RS}_{\mathsf{ano}}}(\mathcal{A}, \mathcal{C})$ and $\mathsf{G}_{\mathsf{RS}_{\mathsf{unf}}}(\mathcal{A}, \mathcal{C})$.

In the game $\mathsf{G}_{\mathsf{RS}_{\mathsf{ano}}}(\mathcal{A}, \mathcal{C})$ the adversary recieves a signature $\sigma = (z, \mathcal{R})$ including a ring $\mathcal{R}$ and a verification key z of q responses which assumed to be generated with the secret information that $\mathcal{A}$ has full access. So, one may think that it is trivial to obtain the signing key of σ for $\mathcal{A}$. Assume that $\mathcal{C}$ simulated σ in the random oracle, i.e., none of the secret keys in $\mathcal{R}$ is the signing key of the signature. Let $\sigma' = (z', \mathcal{R})$ be the simulated signature where $z'_k = (\omega'_k, \tau'_k)$ for $h_k = 1$ and $z'_k = (j(E_k), \beta(S_c), \mathsf{Path}(j_{ck}))$ for $h_k = 0$. And let $\sigma = (z, \mathcal{R})$ be the real signature where $z_k = (\omega_k, \tau_k)$ for $h_k = 1$ and $z_k = (j(E_{V_k}), \beta_k(S_{i\mathsf{b}}), \mathsf{Path}(j_{i\mathsf{b}k}))$ for $h_k = 0$. These two distributions are indistinguishable due to the computational assumption of DSSP; thus, the knowledge of the secret signing keys do not provide any advantage to adversary. Furthermore, $\mathcal{A}$ has to make guesses for each iteration of the signature which comprises $q = O(\lambda)$

transcripts. The winning probability of $\mathcal{A}$ for each transacript is $1/2$, independent of the knowledge of secret information. Consequently, let ξ_{ano} be the advantage of $\mathcal{A}$ in the game $\mathsf{G}_{\mathsf{RS}_{\mathsf{ano}}}(\mathcal{A}, \mathcal{C})$ and is defined as:

$$\xi_{\mathsf{ano}} = 1/2 \cdot |Pr[\mathcal{A}\ wins = 1] - 1/2|$$

which is negligible for λ. Hence, the zero-knowledge property of the ring signature is independent of the knowledge of the secret keys, which preserves the anonymity of the proposed scheme even against the full key exposure.

In the game $\mathsf{G}_{\mathsf{RS}_{\mathsf{unf}}}(\mathcal{A}, \mathcal{C})$, the case that the adversary outputs a signature σ^* of m^* with respect to the ring $\mathcal{R}^*$, where a signature for never queried for $m^*, \mathcal{R}^*$ and $\mathcal{R}^*$ does not include any corrupted public key, σ^* is indistinguishable from a signature generated in the random oracle assumption. Then, we can say that if $\mathcal{A}$ succeeds, it means that either a collision found or two valid transcripts results in a secret key in L_{pk} due to the security assumptions of supersingular isogenies.

□

5. Supersingular Isogeny-Based Linkable Ring Signature

In this section, we present the linkable version of the supersingular isogeny-based ring signature given in Section 4.

The *supersingular isogeny-based linkable ring signature* contains a *tag* which is the second public key of the signer and can only be generated by the signer's secret key. When there are no auxiliary image points, the supersingular isogeny maps of different degrees are non-commutative. Hence, only Peggy can compute her tag. Besides, an adversary cannot produce a valid signature linked with the signature that Peggy issued (assuming Peggy is an honest ring member), i.e., non-frameability. Peggy generates a proof (a signature verification key) that verifies both the tag and other public keys in the ring to obtain the linkability property. To decide if the same secret key creates two signatures concerning the same ring, one can compare the tags of given two signatures.

Supersingular isogeny-based linkable ring signature scheme consists of four algorithms such that $\mathsf{Kgen}_{\mathsf{LRS}}$, $\mathsf{Sig}_{\mathsf{LRS}}$, $\mathsf{Ver}_{\mathsf{LRS}}$, and $\mathsf{Lver}_{\mathsf{LRS}}$. Let $p = \ell_p \ell_v \ell_t f \pm 1$ be a prime number where $\ell_p \approx \ell_v \approx \ell_t$ for given security parameter λ. $E(\mathbb{F}_{p^2})$ is a supersingular elliptic curve. The points P and Q are the generators of the ℓ_v-torsion group such that $E[\ell_v] = \langle P, Q \rangle$ and the point T is an element of ℓ_t-torsion group such that $T \in E[\ell_t]$. One can compute the elliptic curve E_T by the ℓ_t-degree isogeny $\theta : E \to E_T$ and the corresponding image points $(P_T, Q_T) = (\theta(P), \theta(Q))$. So, the set of public parameters is obtained as $pp = \{p, E, E_T, P, Q, T, P_T, Q_T, \mathsf{H}\}$ where H is a hash function with output size $O(\lambda) = q$.

- $\mathsf{Kgen}_{\mathsf{LRS}}(pp)$: For the i^{th} user, the point $S_i \in E[\ell_p]$ which generates the kernel of the secret isogeny $\alpha_i : E \to E_{S_i}$, is the secret key, (E_{S_i}, P_i, Q_i) is the public key where $(P_i, Q_i) = (\alpha_i(P), \alpha_i(Q))$. Using the map θ_i that defines an isogeny from E to E_{TS_i} with the kernel $\langle T, S_i \rangle$, the tag of the i^{th} user is generated. The tag consists of the image curve E_{TS_i} and two image points $(P_{T_i}, Q_{T_i}) = (\theta_i(P), \theta_i(Q))$. So, $(sk_i, pk_i, Tag_i) = (S_i, (E_{S_i}, P_i, Q_i), (E_{TS_i}, P_{T_i}, Q_{T_i}))$ is obtained. Note that the signer generates the tag only if she will issue a linkable signature, thus, the list of public keys L_{pk} do not include the tags of the users.
- $\mathsf{Sig}_{\mathsf{LRS}}(pp, L_{pk}, S_r, Tag_r, m)$: Let S_r, (E_{S_r}, P_r, Q_r), and $Tag_r = (E_{TS_r}, P_{T_r}, Q_{T_r})$

be Peggy's secret key, public key, and tag, respectively. Assume that Peggy wants to generate a linkable signature on behalf of a ring $\mathcal{R} = \{(E_{S_i}, P_i, Q_i)\}_{i=1}^{n}$ including her public key. Peggy implements the following sequence in order to issue a supersingular isogeny-based linkable ring signature. The signature isogenies of proposed scheme are illustrated in Figure 7.

For $k = 1, 2, \ldots, q$:

- Select a random permutation $\tau_k(\lambda)$ and a random number $\omega_k \in \mathbb{Z}/\ell_v\mathbb{Z}$.
- Calculate the kernels $\langle V_k \rangle = P + [\omega_k]Q$, $\langle V_{kT} \rangle = P_T + [\omega_k]Q_T$, $\langle V_{kT_r} \rangle = P_{T_r} + [\omega_k]Q_{T_r}$, $\langle V_{ki} \rangle = P_i + [\omega_k]Q_i$ of the isogenies with the domains E, E_T, E_{TS_r}, and E_{S_i} for $i = 1, 2, \ldots, n$, respectively.
- Compute the isogenies $\beta_k : E \to E_{V_k}$, $\gamma_k : E_T \to E_{TV_k}$, and $\delta_k : E_{TS_r} \to E_{TS_rV_k}$.
- For each user i in $\mathcal{R}$, compute the isogenies $\beta_{ki} : E_{S_i} \to E_{S_iV_k}$.
  Obtain the permuted list $X_k = [j_{i_1}, \ldots, j_{i_{n+1}}]$ by gathering the j-invariants of the commitment curves and applying τ_k on it.
- Generate the Merkle tree M_k on X_k and set $\mathsf{root}_k = \mathsf{Root}(M_k)$.
- Set the tag, $\mathsf{tag}_k = (j(E_{TV_k}), j(E_{TS_rV_k}))$.
- The signature for iteration k is $\sigma_k = \mathsf{H}(\mathsf{root}_k, \mathsf{tag}_k)$.

When Peggy collects all σ_k values, she computes $h = \mathsf{H}(m, \sigma_1, \ldots, \sigma_q)$. Let z be a set that contains the verification keys matching each bit of h. If $h_k = 1$, Peggy sets $z_k = (\omega_k, \tau_k)$ and if $h_k = 0$, she sets $z_k = (j(E_{V_k}), \beta_k(S_r), \beta_k(T), \mathsf{Path}(j_{rk}))$ where j_{rk} denotes the j-invariant of the curve $E_{S_rV_k}$. The signature is obtained as $\sigma = (z, Tag_r, \mathcal{R})$.

- $\mathsf{Ver}_{\mathsf{LRS}}(pp, \sigma, m)$: After receiving the message m and the signature $\sigma = (z, Tag_r, \mathcal{R})$, Victor reconstructs the bits of h as explained in Section 4. For the case $z_k = (\omega_k, \tau_k)$, by using pp and ω_k, the image curves $E_{V'_k}$ and $E_{TV'_k}$ can be obtained. Also, for each member of $\mathcal{R}$, $E_{S_iV'_k}$ can be generated. Victor obtains the permuted list $X'_k = [j'_{i_1}, \ldots, j'_{i_{n+1}}]$ by applying τ_k on the commitment list. After constructing the Merkle tree M'_k of X'_k, he sets $\mathsf{root}'_k = \mathsf{Root}(M'_k)$. He can also define the curve $E_{TS'_rV'_k}$ with the kernel $V'_{kT_r} = P_{T_r} + [\omega_k]Q_{T_r}$ since Peggy sent the Tag_r in σ. Finally, he sets $\mathsf{root}'_k = (j(E_{TV'_k}), j(E_{TS'_rV'_k}))$, and computes $\sigma'_k = \mathsf{H}(\mathsf{root}'_k, \mathsf{tag}'_k)$. In the case that $h_k = 0$ and $z_k = (j(E_{V_k}), \beta_k(S_r), \beta_k(T), \mathsf{Path}(j_{rk}))$, Victor computes the isogeny $E_{V_k} \to E_{V_kS'_r}$ with the kernel $\beta_k(S_r)$. Since $\mathsf{Path}(j_{rk})$ is included in z_k, the $\mathsf{Root}(M'_k)$ can be found. He also obtains $E_{V_kT'S'_r} = E_{V_k}/\langle \beta_k(T), \beta_k(S_r) \rangle$ and $E_{V_kT'} = E_{V_k}/\langle \beta_k(T) \rangle$. So, Victor sets $\mathsf{tag}'_k = (j(E_{V_kT'}), j(E_{V_kT'S'_r}))$ and computes $\sigma'_k = \mathsf{H}(\mathsf{root}'_k, \mathsf{tag}'_k)$. Finally, he collects each σ'_k and computes $h' = \mathsf{H}(m, \sigma'_1, \ldots, \sigma'_n)$. If $h' = h$ he accepts, otherwise rejects.
- $\mathsf{Lver}_{\mathsf{LRS}}(\sigma_1, \sigma_2)$: Let σ_1 and σ_2 be two signatures with respect to the same $\mathcal{R}$, the tag of σ_1 be Tag_1, and the tag of σ_2 be Tag_2. Victor accepts if $Tag_1 \neq Tag_2$, and rejects otherwise.

Theorem 3 *The supersingular isogeny-based linkable ring signature scheme is correct, linkable, linkable anonymous, and non-frameable in the random oracle model, providing that CSSI, SSDDH, and their modified versions MCSSI, MSSDDH are hard problems.*

Proof: In the linkable version of the ring signature, the signer with the index i generates a second public key $Tag = (E_{TS_i}, P_{T_i}, Q_{T_i})$ from her secret key $S_i \in E[\ell_p]$ and the given curve E_T. The curves (E_T, E_{TS_i}) are used for generating signatures

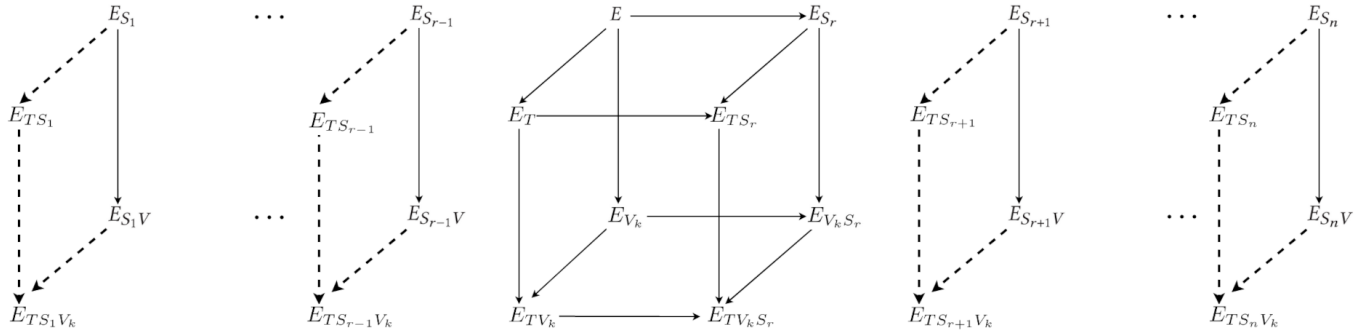

Figure 7. The illustration of the isogenies computed in signature generation of supersingular isogeny-based linkable ring signature protocol.

similar to the public keys in the ring $\mathcal{R}$. Therefore, proving the correctness of the linkable ring signature is based on the correctness of the sigma protocol for the ring $\mathcal{R}$ and (E_T, E_{TS_i}) in q parallel executions.

We prove the linkable anonymity of this scheme by providing games that a PPT adversary $\mathcal{A}$ plays against the linkable anonymity property. We show that the adversary has a negligible advantage to win the following games against a challenger because of the hardness of the isogeny assumptions. Let $\xi = Pr(\mathcal{A}(G_i))$ be the winning probability of the adversary in each game G_i. The games are as follows:

G) In the real game between a challenger and $\mathcal{A}$, the challenger runs $\{(sk_i, pk_i, Tag_i)\}_{i=1}^{n} \leftarrow \mathsf{Kgen}_{\mathsf{LRS}}(pp)$, and selects a random bit $\mathsf{b} \in \{0,1\}$. Then it creates a list $L = \{sk_i\}_{i=1}^{n} - \{sk_{c_0}, sk_{c_1}\}$ where $\{sk_{c_0}, sk_{c_1}\} \in \{sk_i\}_{i=1}^{n}$ and randomly chosen. Challenger provides $pp, L_{pk} = \{pk_i, \ldots, pk_n\}$, and L to adversary. $\mathcal{A}$ selects $\{pk_{a_0}, pk_{a_1}\} \in L_{pk}$ and sets a ring $\mathcal{R}$ such that $\{pk_{a_0}, pk_{a_1}\} \in \mathcal{R}$. $\mathcal{A}$ requests a signature σ for $(pk_{\mathcal{A}}, \mathcal{R}, m)$ where $pk_{\mathcal{A}} \in \{pk_{a_0}, pk_{a_1}\}$.
$\mathcal{A}$ outputs a guess b^* and wins if $\mathsf{b} = \mathsf{b}^*$. The advantage of $\mathcal{A}$ is:

$$\xi = \left| Pr(\mathcal{A}(G)) - \frac{1}{2} \right|.$$

G_1) In this version of the game, the challenger's response to $\mathcal{A}$'s signature query $(pk_{\mathcal{A}}, \mathcal{R}, m)$ is generated by a simulator Sim. The challenger computes $Tag = E_{TS_c}$ using a secret key $sk_c = S_c$ corresponding with $pk_c = E_{S_c}$ and generates binary random challenges h. Then it runs the zero-knowledge Sim to return the signature $\sigma \leftarrow \mathsf{Sim}(pp, \mathcal{R}, Tag, h, m)$. The simulator program uses the random oracle to adjust the challenges h with the inputs to generate a signature. In this case, Sim generates the transcripts without the knowledge of a secret key. The advantage of $\mathcal{A}$ in G_1 is:

$$|Pr(\mathcal{A}(G_1)) - Pr(\mathcal{A}(G))| = negl(\lambda).$$

G_2) In this modified game, the challenger first makes a guess on the public keys that $\mathcal{A}$ selected. Let challenger's guess be $\{pk_{c_0}, pk_{c_1}\} \in pk$. Then, it computes their corresponding tags Tag_{c_0}, Tag_{c_1} using the secret keys sk_{c_0} and sk_{c_1}, respectively. If $pk_{\mathcal{A}} \in \{pk_{c_0}, pk_{c_1}\}$, the challenger returns $\sigma \leftarrow \mathsf{Sim}(pp, \mathcal{R}, Tag_{c_{\mathcal{A}}}, h, m)$ as in game G_1 where

$Tag_{c_{\mathcal{A}}}$ is the tag of $\mathcal{A}$ corresponding with pk_{c_i}. Otherwise, it aborts the game. In this case, $\mathcal{A}$'s advantage is:

$$\big|Pr(\mathcal{A}(G_2))\big| = \frac{1}{n^2}\big|Pr(\mathcal{A}(G_1))\big|.$$

G_3) In this game, the challenger first generates binary random challenges h and samples $i_0, i_1 \in \{1, 2, \ldots, n\}$. Then, it generates random $\{pk_{i_0}, pk_{i_1}\}$ and $\{Tag_{i_0}, Tag_{i_1}\}$ which are isogeneous to E and E_T, respectively. The challenger adds the new public keys to L_{pk} such that $L_{pk} = \{pk_1, \ldots, pk_n\} \bigcup \{pk_{i_0}, pk_{i_1}\}$. For the query $(pk_{\mathcal{A}}, \mathcal{R}, m)$, challenger runs the simulator $\sigma \leftarrow \mathsf{Sim}(pp, \mathcal{R}, Tag_{i_{\mathcal{A}}}, h, m)$ if $pk_{\mathcal{A}} \in \{pk_{i_0}, pk_{i_1}\}$. Otherwise, it aborts the game. Thus, the signature simulated in this way is computationally indistinguishable from the game G_2 (SSDDH and MSSDDH problems). Therefore,

$$|Pr(\mathcal{A}(G_3)) - Pr(\mathcal{A}(G_2))| = negl(\lambda).$$

Moreover, unlike the above games, $\{pk_{i_0}, pk_{i_1}\}$ and $\{Tag_{i_0}, Tag_{i_1}\}$ are not dependent on a secret key. Also, by knowing that the signature is generated by a zero-knowledge simulator, we obtain

$$\xi = |Pr(\mathcal{A}(G_3))| = \frac{1}{2}.$$

Finally, applying the result of G_3 to the previous games shows that $\mathcal{A}$'s advantage in G is negligible.

Here, we briefly provide proof of linkability; for more details, please refer to [20]. Suppose that $\mathcal{A}$ is an efficient adversary against the linkability, that runs $\{(sk_i, pk_i, Tag_i)\}_{i=1}^{n} \leftarrow \mathsf{Kgen}_{\mathsf{LRS}}(pp)$ and generates a set $\{(\sigma_i, \mathcal{R}_i, m_i)\}_{i=1}^{n+1}$ of valid signatures with different tags, which means $\mathsf{Lver}_{\mathsf{LRS}}(\sigma_j, \sigma_k) = 0$ for all $(j, k) \in [1, n+1]$ where $j \neq k$.

One can construct an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ as a black box and programs a random oracle to refresh the randomness on certain points. If $\mathcal{A}$ wins the linkability game, for each $1 \leq j \leq n+1$, then $\mathcal{B}$ can extract the secret key sk_j. To achieve this, $\mathcal{B}$ reruns $\mathcal{A}$ by changing the randomness (of the query j) to get signatures with different challenges and verification keys, i.e., $(\mathsf{comm}, h, z)_j$ and $(\mathsf{comm}, h', z')_j$. $\mathcal{B}$ obtains sk_j based on the 2-special soundness property of the sigma protocol. Assume that $\mathcal{A}$ wins the linkability game, then $\mathcal{B}$ will extract $n+1$ secret keys $\{sk_1, \ldots, sk_{n+1}\}$. Therefore, either two of the following cases might happen:

- One of the secret keys provides a collision for the composite hash roots $\sigma_k = \mathsf{H}(\mathsf{root}_k, \mathsf{tag}_k)$.
- There exist two secret keys that generate one public key but two different tags. In other words, since we have n public keys, then there exist two secret keys $sk_k = S_k$ and $sk_j = S_j$ such that $E/\langle S_k \rangle = E/\langle S_j \rangle$ and $(E_{TS_k} = E/\langle T, S_k \rangle) \neq (E/\langle T, S_j \rangle = E_{TS_j})$.

The advantage of the adversary in the first case is negligible. For the second case, we know that an isogeny is uniquely determined by its kernel up to isomorphism, so the adversary's success is zero.

We use a similar technique that we used in the proof of linkability to prove non-frameability. Let $\mathcal{A}$ be an efficient adversary against the non-frameability property and let $\mathcal{B}$ be an algorithm that uses $\mathcal{A}$ that can program a random oracle to extract the secret key (which is not corrupted before) for given (pk_j, Tag_j). This algorithm runs $\{(sk_i, pk_i, Tag_i)\}_{i=1}^{n+1} \leftarrow \mathsf{Kgen}_{\mathsf{LRS}}(pp)$ (where $(sk_j, pk_j) \neq (sk_k, pk_k)$ for $j \neq k$), generates a simulated signature $\sigma \leftarrow \mathsf{Sim}(pp, \mathcal{R}, pk_j, Tag_j, m)$, and some signatures by querying random oracle.

$\mathcal{B}$ runs $\mathcal{A}$ for $pk_k \in \mathcal{R}$. If $\mathcal{A}$ wins the game, it provides a linkable signature σ^* on $(Tag_j, \mathcal{R}^*, m^*)$ (i.e., two signatures with the same tag implying that $\mathsf{Lver}_{\mathsf{LRS}}(\sigma, \sigma^*) = 1$) which means that $\mathcal{A}$ has queried the random oracle on a certain point. Therefore $\mathcal{B}$

can run $\mathcal{A}$ with new randomness and retrieve the secret key sk_j. So, sk_j is either a collision for the hash function or an answer to the supersingular isogeny computation problem. The advantage of $\mathcal{A}$ in both cases is negligible. □

6. Efficiency and Implementation

This section provides efficiency analyses of the schemes introduced in Section 3, 4, and 5 and implementation results of the supersingular isogeny-based ring signature. Note that the method given in *Lemma 2.* of [29] is used for the efficiency analyses.

6.1. Efficiency Analyses

The best known classical and quantum attacks of supersingular isogeny assumptions of smooth degree $\ell_p \approx \ell_v$ have roughly $O(\sqrt{\ell_p})$ and $O(\sqrt[3]{\ell_p})$ heuristic running times, respectively. Thus, for a given security parameter λ, we have $\log \ell_p = 2\lambda$ for the classical security and $\log \ell_p = 3\lambda$ for the quantum security.

We assume that H is a secure hash function with the output $\{0,1\}^q$ where $q = O(\lambda)$ and the ring $\mathcal{R}$ consists of n public keys. $pk_i = (j(E_i), x(P_i), x(Q_i)) \in \mathcal{R}$ is the public key of a ring member where $j(E_i), x(P_i), x(Q_i) \in \mathbb{F}_{p^2}$. The secret key of the i^{th} user is an ℓ_p-torsion point. Assume that $\langle P_s, Q_s \rangle = E[\ell_p]$, $sk_i \in \mathbb{Z}/\ell_p\mathbb{Z}$ and sk_i is relatively prime to the smooth base (i.e., if $\ell_p = 2^a$ then $gcd(sk_i, 2) = 1$), then the secret key of the i^{th} user is defined as $S_i = P_s + [sk_i]Q_s$. Therefore, it is enough to represent the secret key with the integer sk_i.

We present the efficiency analysis of the supersingular isogeny-based sigma protocol for a ring. $\mathcal{R}$ consists of n public keys $pk_i = (j(E_i), x(P_i), x(Q_i))$, where one of these public keys corresponds with the prover's secret key sk_r. The size of the ring is $|\mathcal{R}| = 6n \log p$, where the size of a public key is $|pk_i| = 6 \log p$, and the secret key size is $|sk_i| = 1/2 \log p$, providing that the generators of the torsion group $E[\ell_p]$ are given as public information. The prover sends a commitment $\mathsf{comm} = [j_{i_1}, j_{i_2}, \ldots, j_{i_{n+1}}]$ consisting the j-invariants of $n+1$ curves that are computed using the ℓ_v-isogeny maps from E and the curves in $\mathcal{R}$. In this case, the size of the commitment is $|\mathsf{comm}| = 2(n+1) \log p$ where $j_i \in \mathbb{F}_{p^2}$. The prover's response is either $\mathsf{resp} = (\omega, \tau)$ or $\mathsf{resp} = (j(E_V), x(\beta(S_r)))$ based on challenge $\mathsf{b} = 1$ and $\mathsf{b} = 0$, respectively. On average, the size of the response

$$|\mathsf{resp}| = \frac{1}{2}\left(1/2 \log p + \log \tau + \left[2 \log p + 1/2 \log p\right]\right)$$

where $|\omega| = 1/2 \log p$, $|\tau| = \log \tau$, $|x(\beta(S_r))| = 1/2 \log p$, and $|j(E_V)| = 2 \log p$. With the Merkle tree implementation, the size of the prover's response can be changed to

$$|\mathsf{resp}| = 3 \log p + q \log n + \log \tau$$

where $q \log n$ is the Merkle tree path size from a leaf node to root. The computation of the supersingular isogeny map is the main operation in the proposed sigma protocol. The prover computes $n+1$ isogenies to generate a commitment. In the verification phase, the verifier computes $n+1$ isogenies if $\mathsf{b} = 1$ and one isogeny if $\mathsf{b} = 0$.

Efficiency analysis of the supersingular isogeny-based ring signature can be explained as follows: The key sizes and the ring size are the same as the sigma protocol. The verification key includes q elements, and the size of each element depends on the hash output. The hash function H with output h of q bits where the number of $h_k = 0$ and $h_k = 1$ are roughly equal. So, the size of z is calculated as follows: In the case that $h_k = 1$,

$|z_k| = 1/2 \log p + \log \tau$. If $h_k = 0$, $|z_k| = 5/2 \log p + q \log n$ where $|j(E_{V_k})| + |x(\beta_k(S_r))| = 5/2 \log p$ and $|\mathsf{Path}(j_{rk})| = q \log n$. Consequently, $|z| = q/2(1/2 \log p + \log \tau + (5/2 \log p + q \log n))$. When we put them all together, we come up with the size of the signature on average:

$$|\sigma| = 6n \log p + \frac{q}{2}\left(3 \log p + q \log n + \log \tau\right).$$

In the proposed ring signature, the signer computes $q(n+1)$ isogenies to generate the signature, and the verifier computes $q/2(n+1)$ isogenies on average to verify the signature.

If we have an ordered set of public keys, instead of including a ring of public keys as a part of the signature, which increases the total size of signature $6n \log p$, the signer can provide a seed and an integer as part of the signature. The seed generates n random integers. The signer then finds an integer such that the addition of the random numbers and integer modulo n will generate the indices of n public keys, including the signer's public key from the ordered public key list. This optimization saves approximately $6n \log p$ in the signature size.

In the construction of the linkable ring signature proposed in Section 5, the signer needs to provide another public key related to her secret key as a tag. So, for a given security parameter λ, a prime number $p = \ell_p \ell_v \ell_t f \pm 1$ of size $\log p \approx 6\lambda$ for classical and $\log p \approx 9\lambda$ for quantum security are required. The public key size is $|pk_i| = 6 \log p$, and the secret key size is $|sk_i| = 1/2 \log p$. The size of the linkable ring signature $\sigma = (z, Tag_r, \mathcal{R})$ differs from the ring signature in Tag_r and z. Since a tag contains three $\mathbb{F}_{p^2}$ elements, the size is represented as $|Tag_r| = 6 \log p$. The size calculation of z changes only when $h_k = 0$. In this case, $z_k = (j(E_{V_k}), x(\beta_k(S_r)), x(\beta_k(T)), \mathsf{Path}(j_{rk}))$, therefore the size is computed as follows: $|z_k| = 6 \log p + q \log n$ where $|x(\beta_k(S_r))| + |x(\beta_k(T))| = 4 \log p$ in canonical representation. Hence, the size of the linkable ring signature $\sigma = (z, Tag_r, \mathcal{R})$ on average is:

$$|\sigma| = (6n+6) \log p + \frac{q}{2}\left(7/2 \log p + q \log n + \log \tau\right).$$

Signature generation requires $q(n+3)$ and verification requires $q/2(n+6)$ isogeny computations on average.

6.2. Implementation Results

Here we share the benchmark test results of SIRS. We implement the protocol by using the isogeny functions of SIDH library[1] that provides SIDH and SIKE implementations for different levels of security.

Our implementation calculates the average running cycles of SIRSp434, SIRSp503, SIRSp610, and SIRSp751 that provide post-quantum security of AES128 (NIST 1), SHA3-256 (NIST 2), AES192 (NIST 3), and AES256 (NIST 5), respectively as in SIDH library. We run the tests on a 64-bit platform; however, the library supports both 32-bit and 64-bit architectures. We compiled the library using clang[2] with optimization level FAST. For more details, please refer to [25]. For each level of security, we gathered the signature sizes. Merkle tree construction is done using the hash algorithm SHAKE256 with the output sizes[3] required by the relevant quantum security levels.

Recall the verification key size calculation given in Section 6.1 of the ring signature protocol:

$$\frac{q}{2}\big(|\omega| + |\tau| + |j(E_V)| + |x(\beta(S_r))| + |\mathsf{Path}((j_r)|\big).$$

The signature σ includes the verification key z

1. See https://github .com/Microsoft/PQCrypto-SIDH.

2. A compiler front end for some programming languages like C, as well as some frameworks such as OpenMP.

3. The hash output is taken as 2λ for λ-bit quantum security.

Table 1.
The the signature sizes in kilobytes.

n	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}
SIRSp434	25.6	26.6	27.6	28.6	29.6	30.6	31.6	32.6
SIRSp503	39.1	40.9	42.6	44.3	46	47.7	49.5	51.2
SIRSp610	54.1	56.3	58.6	60.8	63.1	65.3	67.6	69.8
SIRSp751	88.9	92.9	96.9	100.9	104.9	108.9	112.9	116.9

and the ring $\mathcal{R}$; however, the size of a signature is taken as the size of z since providing a seed instead of $\mathcal{R}$ is enough to recover $\mathcal{R}$.

We give the signature sizes (z) in Table 1 that SIRS provides. Figure 8 shows the number of cycles to generate and verify a signature of SIRS. Note that the supersingular isogeny-based ring signature benchmark tests were done on a *3.1GHz AMD Ryzen 3 1200 Quad-Core Processor* running on *Debian GNU/Linux 11 (bullseye).*

7. Conclusion

In this paper, we have presented a post-quantum sigma protocol for a ring based on supersingular isogenies. We have proved the correctness, 2-special soundness, and honest-verifier zero-knowledge properties of this supersingular isogeny-based sigma protocol for a ring. We have also proposed a supersingular isogeny-based ring signature obtained by applying Fiat-Shamir transform to the supersingular isogeny-based sigma protocol for a ring. The correctness, anonymity, and existential unforgeability properties of this ring signature scheme have been provided. Furthermore, we have added the linkability property to the ring signature scheme, which offers the ability to determine if the same signer has issued two signatures and could prevent the problems such as double-spending attacks on crypto-currencies and double-voting attacks on e-voting protocols. We have shown that the supersingular isogeny-based linkable ring signature scheme is correct, linkable, linkable anonymous, and non-frameable in the random oracle model. We have applied the Merkle tree to our constructions to improve the efficiency of the proposed protocols. We have provided the efficiency analyses of the given protocols. In the proposed ring signature, the signature size grows logarithmically in the size of the ring where Merkle tree paths or roots have formed a part of the verification keys. Finally, we have shared the benchmark test results and signature sizes of supersingular isogeny-based ring signature for post-quantum security levels of NIST. SIRS protocol provides small signature sizes and efficient implementation; thus, it is a strong candidate for post-quantum applications.

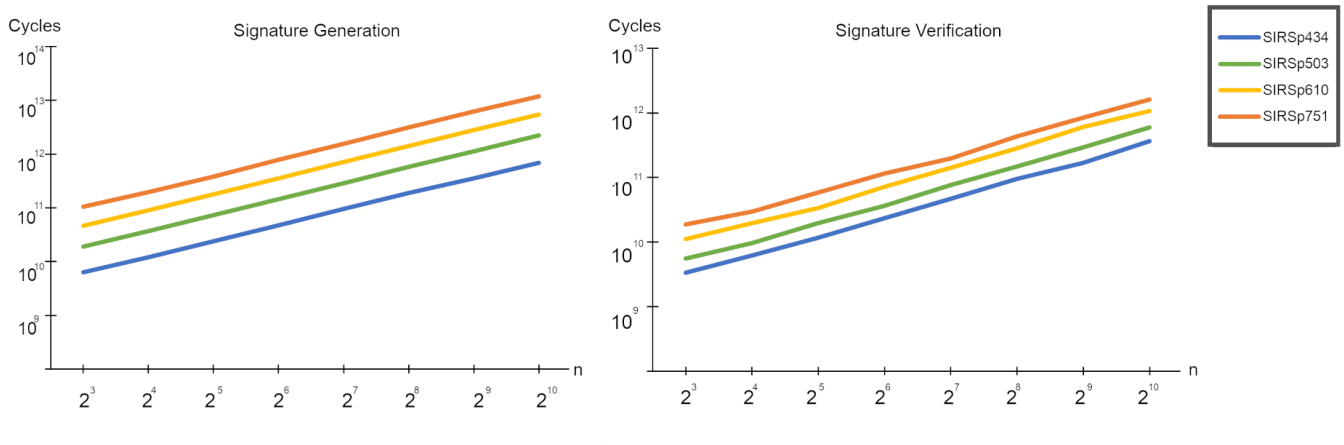

Figure 8. Number of cycles (in logarithmic scale) needed for signature generation and verification of SIRS.

Appendix A
Algorithms

Here, we provide the algorithms of supersingular isogeny-based sigma protocol for a ring $(\mathsf{Kgen}, \mathsf{Comm}, \mathsf{Resp}, \mathsf{Ver})$, ring signature $(\mathsf{Sig}_{\mathsf{RS}}, \mathsf{Ver}_{\mathsf{RS}})$, and linkable ring signature $(\mathsf{Kgen}_{\mathsf{LRS}}, \mathsf{Sig}_{\mathsf{LRS}}, \mathsf{Ver}_{\mathsf{LRS}}, \mathsf{Lver}_{\mathsf{LRS}})$ in Section A.1, A.2, A.3, respectively. The following are the generic parameters and functions used in the algorithms we give in this Appendix.

- λ is the security parameter,
- $\mathcal{R} = [(E_{S_i}, P_i, Q_i)]_{i=0}^{n-1}$ is the ring with n members,
- X denotes the commitment list that includes the j-invariants of the commitment curves,
- M denotes the Merkle tree,
- H is the function with the output size $q = O(\lambda)$ that returns the hash of the given input,
- $\mathsf{IsogenyMap}$ is a function that returns the isogeny map of the given domain curve and the kernel,
- $\mathsf{IsogenyImage}$ is a function that returns the image curve of the given isogeny map,
- Rand is a function that outputs a random element in the given range,
- $\mathsf{Permute}$ is the function that permutes the given list with given τ,
- $\mathsf{MerkleTree}$ function builds the Merkle tree for given list of commitments,
- Root function returns the root of given Merkle tree,
- Path is the function that returns the path of given node to the root of given Merkle tree,
- $\mathsf{FindRoot}$ function returns the root of a Merkle tree with the given path of given node hash,
- $\mathsf{RetrieveCurve}$ denotes the function that retrieves the elliptic curve equation with given j-invariant.

Sigma protocol and ring signature are defined on the same prime p (i.e., $\log p = 6\lambda$ for quantum security); however, linkable ring signature requires a different setting since it uses a larger prime ($\log p = 9\lambda$) to handle the tag. Here we define the parameters specific for the algorithms given in Sections A.1 and A.2:

- $p = \ell_p \ell_v f \pm 1$ is the prime number such that $\ell_p \approx \ell_v$,

- $E(\mathbb{F}_{p^2})$ is the supersingular elliptic curve,
- P and Q are the points such that $E[\ell_v] = \langle P, Q\rangle$,
- $S_r \in E[\ell_p]$ denotes the secret key of the signer,
- $(E_{S_r}, P_r, Q_r) \in \mathcal{R}$ is the public key of the signer,
- $pp = \{\lambda, p, E(\mathbb{F}_{p^2}), P, Q, \mathsf{H}\}$ is the list of public parameters.

The algorithms of linkable ring signature, given in Section A.3 use the following parameters:

- $p = \ell_p\ell_v\ell_t f \pm 1$ is the prime number such that $\ell_p \approx \ell_v \approx \ell_t$,
- $E(\mathbb{F}_{p^2})$ is the supersingular elliptic curve,
- P, Q, T are the points such that $E[\ell_v] = \langle P, Q\rangle$ and $T \in E[\ell_t]$,
- E_T is the curve which is defined as $\theta : E \to E_T$,
- $S_r \in E[\ell_p]$ denotes the secret key of the signer,
- $(E_{S_r}, P_r, Q_r) \in \mathcal{R}$ is the public key of the signer,
- $Tag_r = (E_{TS_r}, P_{Tr}, Q_{Tr})$ is the tag of the signer,
- $pp = \{\lambda, p, E(\mathbb{F}_{p^2}), P, Q, T, E_T, P_T, Q_T, \mathsf{H}\}$ is the list of public parameters.

Note that we assume Peggy generates the ring before commitment and signature generation; thus, the algorithms Comm, $\mathsf{Sig_{RS}}$, and $\mathsf{Sig_{LRS}}$ use $\mathcal{R}$ as input and so do not return it. Similarly, Victor retrieves the ring before verification; therefore Ver, $\mathsf{Ver_{RS}}$, and $\mathsf{Ver_{LRS}}$ take $\mathcal{R}$ as input.

A1. Supersingular Isogeny-based Sigma Protocol for a Ring

In this section, we provide the algorithms for key generation (Kgen), computing the commitment (Comm), setting the response (Resp), and verification (Ver) of sigma protocol. (Kgen) given in Algorithm 1 takes the list of public parameters as input and returns the secret and public keys of the i^{th} user. Algorithm 2 (Comm) takes pp, $\mathcal{R}$ with size n that includes (E_{S_r}, P_r, Q_r), and S_r as input. It computes and returns the commitment comm. Algorithm 3 (Resp) takes pp, b, M, $j(E_V)$, $j(E_{S_rV})$, β, ω, τ, and S_r as input where b is the challenge that verifier sends to prover after receiving the commitment. It returns the response resp with respect to the challenge b. Algorithm 4 (Ver) takes pp, $\mathcal{R}$, comm, b, and resp as input. It returns 1 for accepting the response and 0 for rejecting it.

Algorithm 1 Kgen

Input: pp.
Output: $S_i, (E_{S_i}, P_i, Q_i)$.
1: $S_i \leftarrow \mathsf{Rand}(E[\ell_p])$.
2: $\alpha_i \leftarrow \mathsf{IsogenyMap}(E, \langle S_i\rangle)$.
3: $E_{S_i} \leftarrow \mathsf{IsogenyImage}(\alpha_i)$.
4: $P_i, Q_i \leftarrow \alpha_i(P), \alpha_i(Q)$.
5: **return** $S_i, (E_{S_i}, P_i, Q_i)$.

Algorithm 2 Comm

Input: pp, $\mathcal{R}$, and S_r.
Output: The commitment comm.
1: $\omega \leftarrow \mathsf{Rand}(\mathbb{Z}/\ell_v\mathbb{Z})$.
2: $V \leftarrow P + [\omega]Q$.
3: $\beta \leftarrow \mathsf{IsogenyMap}(E, \langle V\rangle)$.
4: $E_V \leftarrow \mathsf{IsogenyImage}(\beta)$.
5: $X[0] \leftarrow j(E_V)$.
6: **for** $i = 0$ to $n - 1$ **do**
7: $V_i \leftarrow P_i + [\omega]Q_i$.
8: $\beta_i \leftarrow \mathsf{IsogenyMap}(E_{S_i}, \langle V_i\rangle)$.
9: $E_{S_iV} \leftarrow \mathsf{IsogenyImage}(\beta_i)$.
10: $X[i + 1] \leftarrow j(E_{S_iV})$.
11: **end for**
12: $\tau \leftarrow \mathsf{Rand}(\lambda)$.
13: $X \leftarrow \mathsf{Permute}(X, \tau)$.
14: $M \leftarrow \mathsf{MerkleTree}(X)$.
15: comm $\leftarrow \mathsf{Root}(M)$.
16: **return** comm.

Algorithm 3 Resp

Input: pp, M, b, ω, τ, $j(E_V)$, $j(E_{S_rV})$, and $\beta(S_r)$.
Output: The response resp.

if (b $==1$) **then**
  resp $\leftarrow (\omega, \tau)$.
else
  path $\leftarrow \mathsf{Path}(M, \mathsf{H}(j(E_{S_rV})))$.
  resp $\leftarrow (j(E_V), \beta(S_r), \mathsf{path})$.
end if
return resp.

Algorithm 4 Ver

Input: pp, b, $\mathcal{R}$, comm, and resp.
Output: $1/0$.

if (b $==1$) **then**
  $\omega, \tau \leftarrow \mathsf{resp}[0], \mathsf{resp}[1]$.
  $V' \leftarrow P + [\omega]Q$.
  $\beta' \leftarrow \mathsf{IsogenyMap}(E, \langle V' \rangle)$.
  $E_{V'} \leftarrow \mathsf{IsogenyImage}(\beta')$.
  $X'[0] \leftarrow j(E_{V'})$.
  for $i = 0$ to $n-1$ **do**
    $V'_i \leftarrow P_i + [\omega]Q_i$.
    $\beta'_i \leftarrow \mathsf{IsogenyMap}(E_{S_i}, \langle V'_i \rangle)$.
    $E_{S_iV'} \leftarrow \mathsf{IsogenyImage}(\beta'_i)$.
    $X'[i+1] \leftarrow j(E_{S_iV'})$.
  end for
  $X' \leftarrow \mathsf{Permute}(X', \tau)$.
  $M' \leftarrow \mathsf{MerkleTree}(X')$.
  $\mathsf{comm}' \leftarrow \mathsf{Root}(M')$.
else
  $\mathsf{j}, \mathsf{ker}, \mathsf{path} \leftarrow \mathsf{resp}[0], \mathsf{resp}[1], \mathsf{resp}[2]$.
  $E_{V'} \leftarrow \mathsf{RetrieveCurve(j)}$.
  $\alpha'_r \leftarrow \mathsf{IsogenyMap}(E_{V'}, \langle \mathsf{ker} \rangle)$.
  $E_{V'S'_r} \leftarrow \mathsf{IsogenyImage}(\alpha'_r)$.
  $\mathsf{comm}' \leftarrow \mathsf{FindRoot}(\mathsf{H}(j(E_{V'S'_r})), \mathsf{path})$.
end if
if $\mathsf{comm}' == \mathsf{comm}$ **then**
  return 1.
else
  return 0.
end if

A2. Supersingular Isogeny-based Ring Signature

The algorithms $\mathsf{Sig_{RS}}$ and $\mathsf{Ver_{RS}}$ used in ring signature are given in this section. The supersingular isogeny-based ring signature protocol uses Algorithm 1 for key generation. Algorithm 5 ($\mathsf{Sig_{RS}}$) takes pp, $\mathcal{R}$ with size n that includes (E_{S_r}, P_r, Q_r), and a message m as input. It returns the signature σ for m. Algorithm 6 ($\mathsf{Ver_{RS}}$) takes pp, $\mathcal{R}$, σ, and m as input and returns 1 for accepting the signature, 0 for rejecting it.

Algorithm 5 $\mathsf{Sig_{RS}}$

Input: pp, $\mathcal{R}$, S_r, and m.
Output: The signature σ .

for $k = 0$ to $q-1$ **do**
  $\omega_k \leftarrow \mathsf{Rand}(\mathbb{Z}/\ell_v\mathbb{Z})$.
  $V_k \leftarrow P + [\omega_k]Q$.
  $\beta_k \leftarrow \mathsf{IsogenyMap}(E, \langle V_k \rangle)$.
  $E_{V_k} \leftarrow \mathsf{IsogenyImage}(\beta_k)$.
  $X_k[0] \leftarrow j(E_{V_k})$.
  for $i = 0$ to $n-1$ **do**
    $V_{ki} \leftarrow P_i + [\omega_k]Q_i$.
    $\beta_{ki} \leftarrow \mathsf{IsogenyMap}(E_{S_i}, \langle V_{ki} \rangle)$.
    $E_{S_iV_k} \leftarrow \mathsf{IsogenyImage}(\beta_{ki})$.
    $X_k[i+1] \leftarrow j(E_{S_iV_k})$.
  end for
  $\tau_k \leftarrow \mathsf{Rand}(\lambda)$.
  $X_k \leftarrow \mathsf{Permute}(X_k, \tau_k)$.
  $M_k \leftarrow \mathsf{MerkleTree}(X_k)$.
  $\mathsf{path}_k \leftarrow \mathsf{Path}(M_k, \mathsf{H}(j(E_{V_kS_r})))$.
  $\mathsf{root}_k \leftarrow \mathsf{Root}(M_k)$.
end for
$h \leftarrow \mathsf{H}(m, \mathsf{root}_0, \ldots, \mathsf{root}_{q-1})$.
for $k = 0$ to $q-1$ **do**
  if $h[k] == 1$ **then**
    $\sigma[k] \leftarrow (\omega_k, \tau_k)$.
  else
    $\sigma[k] \leftarrow (j(E_{V_k}), \beta_k(S_r), \mathsf{path}_k)$.
  end if
end for
return σ.

Algorithm 6 $\mathsf{Ver_{RS}}$

Input: pp, $\mathcal{R}$, σ, and m.
Output: $1/0$.

Define h.
for $k = 0$ to $q-1$ **do**
  if $\sigma_k == (\omega_k, \tau_k)$ **then**
    $h \leftarrow h \parallel 1$.
    $\omega_k, \tau_k \leftarrow \sigma_k[0], \sigma_k[1]$.
    $V'_k \leftarrow P + [\omega_k]Q$.
    $\beta'_k \leftarrow \mathsf{IsogenyMap}(E, \langle V'_k \rangle)$.
    $E_{V'_k} \leftarrow \mathsf{IsogenyImage}(\beta'_k)$.
    $X'_k[0] \leftarrow j(E_{V'_k})$.
    for $i = 0$ to $n-1$ **do**
      $V'_{ki} \leftarrow P_i + [\omega_k]Q_i$.
      $\beta'_{ki} \leftarrow \mathsf{IsogenyMap}(E_{S_i}, \langle V'_{ki} \rangle)$.
      $E_{S_i V'_k} \leftarrow \mathsf{IsogenyImage}(\beta'_{ki})$.
      $X'_k[i+1] \leftarrow j(E_{S_i V'_k})$.
    end for
    $X'_k \leftarrow \mathsf{Permute}(X'_k, \tau_k)$.
    $M'_k \leftarrow \mathsf{MerkleTree}(X'_k)$.
    $\mathsf{root}'_k \leftarrow \mathsf{Root}(M'_k)$.
  else
    $h \leftarrow h \parallel 0$.
    $\mathsf{j}_k, \mathsf{ker}_k, \mathsf{path}_k \leftarrow \sigma_k[0], \sigma_k[1], \sigma_k[2]$.
    $E_{V'_k} \leftarrow \mathsf{RetrieveCurve}(\mathsf{j}_k)$.
    $\alpha'_{kr} \leftarrow \mathsf{IsogenyMap}(E_{V'_k}, \langle \mathsf{ker}_k \rangle)$.
    $E_{V'_k S'_r} \leftarrow \mathsf{IsogenyImage}(\alpha'_{kr})$.
    $\mathsf{root}'_k \leftarrow \mathsf{FindRoot}(\mathsf{H}(j(E_{V'_k S'_r})), \mathsf{path}_k)$.
  end if
end for
$h' \leftarrow \mathsf{H}(m, \mathsf{root}'_0, \ldots, \mathsf{root}'_{q-1})$.
if $h == h'$ **then**
  return 1.
else
  return 0.
end if

A3. Supersingular Isogeny-based Linkable Ring Signature

In this section we give the algorithms $\mathsf{Kgen_{LRS}}$, $\mathsf{Sig_{LRS}}$, $\mathsf{Ver_{LRS}}$, and $\mathsf{Lver_{LRS}}$ used in linkable ring signature. Algorithm 7 ($\mathsf{Kgen_{LRS}}$) takes pp as input and returns the secret key S_i, the public key (E_{S_i}, P_i, Q_i), and the tag $(E_{TS_i}, P_{T_i}, Q_{T_i})$ of the i^{th} user. Note that, a signer generates the tag only if she issues a linkable ring signature. Algorithm 9 ($\mathsf{Sig_{LRS}}$) takes pp, $\mathcal{R}$ with size n that includes (E_{S_r}, P_r, Q_r), S_r, Tag_r, and m as input and returns a linkable ring signature σ for the message m. Algorithm 10 ($\mathsf{Ver_{LRS}}$) takes pp, σ, $\mathcal{R}$, and m as input and returns 1 for accepting the linkable ring signature and 0 for rejecting it. Algorithm 8 (LverLRS) takes σ_1,σ_2 as input and returns 1 if the tags of given two signatures are different and 0 otherwise.

Algorithm 7 $\mathsf{Kgen_{LRS}}$

Input: pp.
Output: $S_i, (E_{S_i}, P_i, Q_i)$, and $(E_{TS_i}, P_{T_i}, Q_{T_i})$.

$S_i \leftarrow \mathsf{Rand}(E[\ell_p])$.
$\alpha_i \leftarrow \mathsf{IsogenyMap}(E, \langle S_i \rangle)$.
$\theta_i \leftarrow \mathsf{IsogenyMap}(E, \langle T, S_i \rangle)$.
$E_{S_i} \leftarrow \mathsf{IsogenyImage}(\alpha_i)$.
$E_{TS_i} \leftarrow \mathsf{IsogenyImage}(\theta_i)$.
$P_i, Q_i \leftarrow \alpha_i(P), \alpha_i(Q)$.
$P_{T_i}, Q_{T_i} \leftarrow \theta_i(P), \theta_i(Q)$.
return $S_i, (E_{S_i}, P_i, Q_i), (E_{TS_i}, P_{T_i}, Q_{T_i})$.

Algorithm 8 $\mathsf{LVer_{LRS}}$

Input: σ_1,σ_2.
Output: $1/0$.

$Tag_1, Tag_2 \leftarrow \sigma_1[1], \sigma_2[1]$
if $Tag_1 == Tag_2$ **then**
  return 0
else
  return 1
end if

INTERNATIONAL JOURNAL OF INFORMATION SECURITY SCIENCE
M. S. Garjan et al., Vol.12, No.1, pp. 32-57.
https://doi.org/10.55859/ijiss.1187756

Algorithm 9 $\mathsf{Sig}_{\mathsf{LRS}}$

Input: pp, $\mathcal{R}$, S_r, Tag_r, and m.
Output: The signature σ.

for $k = 0$ to $q-1$ **do**
  $\omega_k \leftarrow \mathsf{Rand}(\mathbb{Z}/\ell_v\mathbb{Z})$.
  $V_k \leftarrow P + [\omega_k]Q$.
  $V_{kT} \leftarrow P_T + [\omega_k]Q_T$.
  $V_{kT_r} \leftarrow P_{T_r} + [\omega_k]Q_{T_r}$.
  $\beta_k \leftarrow \mathsf{IsogenyMap}(E, \langle V_k \rangle)$.
  $\gamma_k \leftarrow \mathsf{IsogenyMap}(E_T, \langle V_{kT} \rangle)$.
  $\delta_k \leftarrow \mathsf{IsogenyMap}(E_{TS_r}, \langle V_{kT_r} \rangle)$.
  $E_{V_k} \leftarrow \mathsf{IsogenyImage}(\beta_k)$.
  $E_{TV_k} \leftarrow \mathsf{IsogenyImage}(\gamma_k)$.
  $E_{TS_rV_k} \leftarrow \mathsf{IsogenyImage}(\delta_k)$.
  $X_k[0] \leftarrow j(E_{V_k})$.
  for $i = 0$ to $n-1$ **do**
    $V_{ki} \leftarrow P_i + [\omega_k]Q_i$.
    $\beta_{ki} \leftarrow \mathsf{IsogenyMap}(E_{S_i}, \langle V_{ki} \rangle)$.
    $E_{S_iV_k} \leftarrow \mathsf{IsogenyImage}(\beta_{ki})$.
    $X_k[i+1] \leftarrow j(E_{S_iV_k})$.
  end for
  $\tau_k \leftarrow \mathsf{Rand}(\lambda)$.
  $X_k \leftarrow \mathsf{Permute}(X_k, \tau_k)$.
  $M_k \leftarrow \mathsf{MerkleTree}(X_k)$.
  $\mathsf{path}_k \leftarrow \mathsf{Path}(M_k, \mathsf{H}(j(E_{V_kS_r})))$.
  $\mathsf{root}_k \leftarrow \mathsf{Root}(M_k)$.
  $\mathsf{tag}_k \leftarrow (j(E_{TV_k}), j(E_{TS_rV_k}))$.
  $\sigma_k \leftarrow (\mathsf{root}_k, \mathsf{tag}_k)$.
end for
$h \leftarrow \mathsf{H}(m, \sigma_0, \ldots, \sigma_{q-1})$.
for $k = 0$ to $q-1$ **do**
  if $h[k] == 1$ **then**
    $z[k] \leftarrow (\omega_k, \tau_k)$.
  else
    $z[k] \leftarrow (j(E_{V_k}), \beta_k(S_r), \beta_k(T), \mathsf{path}_k)$.
  end if
end for
$\sigma \leftarrow (z, Tag_r)$.
return σ.

Algorithm 10 $\mathsf{Ver}_{\mathsf{LRS}}$

Input: pp, $\mathcal{R}$, σ, and m.
Output: $1/0$.

Define h.
for $k = 0$ to $q-1$ **do**
  if $z_k == (\omega_k, \tau_k)$ **then**
    $h \leftarrow h \parallel 1$.
    $\omega_k, \tau_k \leftarrow z_k[0], z_k[1]$.
    $V'_k \leftarrow P + [\omega_k]Q$.
    $V'_{kT} \leftarrow P_T + [\omega_k]Q_T$.
    $V'_{kT_r} \leftarrow P_{T_r} + [\omega_k]Q_{T_r}$.
    $\beta'_k \leftarrow \mathsf{IsogenyMap}(E, \langle V'_k \rangle)$.
    $\gamma'_k \leftarrow \mathsf{IsogenyMap}(E_T, \langle V'_{kT} \rangle)$.
    $\delta'_k \leftarrow \mathsf{IsogenyMap}(E_{TS_r}, \langle V'_{kT_r} \rangle)$.
    $E_{V'_k} \leftarrow \mathsf{IsogenyImage}(\beta'_k)$.
    $E_{TV'_k} \leftarrow \mathsf{IsogenyImage}(\gamma'_k)$.
    $E_{TS_rV'_k} \leftarrow \mathsf{IsogenyImage}(\delta'_k)$.
    $X'_k[0] \leftarrow j(E_{V'_k})$.
    for $i = 0$ to $n-1$ **do**
      $V'_{ki} \leftarrow P_i + [\omega_k]Q_i$.
      $\beta'_{ki} \leftarrow \mathsf{IsogenyMap}(E_{S_i}, \langle V'_{ki} \rangle)$.
      $E_{S_iV'_k} \leftarrow \mathsf{IsogenyImage}(\beta'_{ki})$.
      $X'_k[i+1] \leftarrow j(E_{S_iV'_k})$.
    end for
    $X'_k \leftarrow \mathsf{Permute}(X'_k, \tau_k)$.
    $M'_k \leftarrow \mathsf{MerkleTree}(X'_k)$.
    $\mathsf{root}'_k \leftarrow \mathsf{Root}(M'_k)$.
    $\mathsf{tag}'_k \leftarrow (j(E_{TV'_k}), j(E_{TS_rV'_k}))$.
  else
    $h \leftarrow h \parallel 0$.
    $\mathsf{j}_k \leftarrow z_k[0]$.
    $\ker_{S_k} \leftarrow z_k[1]$.
    $\ker_{T_k} \leftarrow z_k[2]$.
    $\mathsf{path}_k \leftarrow z_k[3]$.
    $E_{V'_k} \leftarrow \mathsf{RetrieveCurve}(\mathsf{j}_k)$.
    $\alpha'_{kr} \leftarrow \mathsf{IsogenyMap}(E_{V'_k}, \langle \ker_{S_k} \rangle)$.
    $\theta'_k \leftarrow \mathsf{IsogenyMap}(E_{V'_k}, \langle \ker_{T_k} \rangle)$.
    $\theta'_{kr} \leftarrow \mathsf{IsogenyMap}(E_{V'_k}, \langle \ker_{T_k}, \ker_{S_k} \rangle)$.
    $E_{V'_kS'_r} \leftarrow \mathsf{IsogenyImage}(\alpha'_{kr})$.
    $E_{V'_kT} \leftarrow \mathsf{IsogenyImage}(\theta'_k)$.
    $E_{V'_kTS'_r} \leftarrow \mathsf{IsogenyImage}(\theta'_{kr})$.
    $\mathsf{root}'_k \leftarrow \mathsf{FindRoot}(\mathsf{H}(j(E_{V'_kS'_r})), \mathsf{path}'_k)$.
    $\mathsf{tag}'_k \leftarrow (j(E_{V'_kT}), j(E_{V'_kTS'_r}))$.
  end if
  $\sigma'_k \leftarrow (\mathsf{root}'_k, \mathsf{tag}'_k)$.
end for
$h' \leftarrow \mathsf{H}(m, \sigma'_0, \ldots, \sigma'_{q-1})$.
if $h == h'$ **then**
  return 1.
else
  return 0.
end if

Acknowledgment

This work was supported by TÜBİTAK under grant no 120E065. M. Sheikhi Garjan was a post-doctoral researcher at Middle East Technical University during a period of this research and would like to thank the Institute of Applied Mathematics Cryptography Department for the hospitality. N. G. Orhon Kılıç was supported by the Council of Higher Education (YÖK) 100/2000 CoHE Ph.D. Scholarship and would like to thank the Council of Higher Education. A part of this paper was written while M. Cenk was visiting the University of Waterloo and would like to thank the Department of Combinatorics & Optimization for the hospitality.

References

[1] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2001, pp. 552–565.

[2] J. K. Liu, V. K. Wei, and D. S. Wong, "Linkable spontaneous anonymous group signature for ad hoc groups," in *Australasian Conference on Information Security and Privacy*. Springer, 2004, pp. 325–335.

[3] P. P. Tsang and V. K. Wei, "Short linkable ring signatures for e-voting, e-cash and attestation," in *International Conference on Information Security Practice and Experience*. Springer, 2005, pp. 48–60.

[4] M. Chase and A. Lysyanskaya, "On signatures of knowledge," in *Annual International Cryptology Conference*. Springer, 2006, pp. 78–96.

[5] Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup, "Anonymous identification in ad hoc groups," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2004, pp. 609–626.

[6] M. Abe, M. Ohkubo, and K. Suzuki, "1-out-of-n signatures from a variety of keys," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2002, pp. 415–432.

[7] J. Groth and M. Kohlweiss, "One-out-of-many proofs: Or how to leak a secret and spend a coin," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2015, pp. 253–280.

[8] J. Herranz and G. Sáez, "Forking lemmas for ring signature schemes," in *International Conference on Cryptology in India*. Springer, 2003, pp. 266–279.

[9] J. K. Liu and D. S. Wong, "Linkable ring signatures: Security models and new schemes," in *International Conference on Computational Science and Its Applications*. Springer, 2005, pp. 614–623.

[10] A. Bender, J. Katz, and R. Morselli, "Ring signatures: Stronger definitions, and constructions without random oracles," in *Theory of Cryptography Conference*. Springer, 2006, pp. 60–79.

[11] L. Nguyen, "Accumulators from bilinear pairings and applications," in *Cryptographers' track at the RSA conference*. Springer, 2005, pp. 275–292.

[12] H. Shacham and B. Waters, "Efficient ring signatures without random oracles," in *International Workshop on Public Key Cryptography*. Springer, 2007, pp. 166–180.

[13] S. S. Chow, S.-M. Yiu, and L. C. Hui, "Efficient identity based ring signature," in *International Conference on Applied Cryptography and Network Security*. Springer, 2005, pp. 499–512.

[14] M. Backes, N. Döttling, L. Hanzlik, K. Kluczniak, and J. Schneider, "Ring signatures: Logarithmic-size, no setup—from standard assumptions," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2019, pp. 281–311.

[15] D. Derler, S. Ramacher, and D. Slamanig, "Post-quantum zero-knowledge proofs for accumulators with applications to ring signatures from symmetric-key primitives," in *International Conference on Post-Quantum Cryptography*. Springer, 2018, pp. 419–440.

[16] J. Katz, V. Kolesnikov, and X. Wang, "Improved non-interactive zero knowledge with applications to post-quantum signatures," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 525–537.

[17] D. H. Duong, H. T. Tran, W. Susilo *et al.*, "An efficient multivariate threshold ring signature scheme," *Computer Standards & Interfaces*, vol. 74, p. 103489, 2020.

[18] M. S. E. Mohamed and A. Petzoldt, "Ringrainbow–an efficient multivariate ring signature scheme," in *International Conference on Cryptology in Africa*. Springer, 2017, pp. 3–20.

[19] C. Baum, H. Lin, and S. Oechsner, "Towards practical lattice-based one-time linkable ring signatures," in *International Conference on Information and Communications Security*. Springer, 2018, pp. 303–322.

[20] W. Beullens, S. Katsumata, and F. Pintore, "Calamari and falafl: Logarithmic (linkable) ring signatures from isogenies and lattices," 2020.

[21] M. F. Esgin, R. K. Zhao, R. Steinfeld, J. K. Liu, and D. Liu, "Matrict: efficient, scalable and post-quantum blockchain confidential transactions protocol," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 567–584.

[22] B. Libert, S. Ling, K. Nguyen, and H. Wang, "Zero-knowledge arguments for lattice-based accumulators: logarithmic-size ring signatures and group signatures without trapdoors," in *Annual*

International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 2016, pp. 1–31.

[23] W. A. A. Torres, R. Steinfeld, A. Sakzad, J. K. Liu, V. Kuchta, N. Bhattacharjee, M. H. Au, and J. Cheng, "Post-quantum one-time linkable ring signature and application to ring confidential transactions in blockchain (lattice ringct v1. 0)," in *Australasian Conference on Information Security and Privacy*. Springer, 2018, pp. 558–576.

[24] L. De Feo, D. Jao, and J. Plût, "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies," *Journal of Mathematical Cryptology*, vol. 8, no. 3, pp. 209–247, 2014.

[25] C. Costello, P. Longa, and M. Naehrig, "Efficient algorithms for supersingular isogeny diffie-hellman," in *Annual International Cryptology Conference*. Springer, 2016, pp. 572–601.

[26] J. H. Silverman, *The arithmetic of elliptic curves*. Springer Science & Business Media, 2009, vol. 106.

[27] D. Jao and L. De Feo, "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies," in *International Workshop on Post-Quantum Cryptography*. Springer, 2011, pp. 19–34.

[28] C. D. de Saint Guilhem, P. Kutas, C. Petit, and J. Silva, "Séta: Supersingular encryption from torsion attacks," 2019.

[29] S. D. Galbraith, C. Petit, and J. Silva, "Identification protocols and signature schemes based on supersingular isogeny problems," *Journal of Cryptology*, vol. 33, no. 1, pp. 130–175, 2020.

[30] Y. Yoo, R. Azarderakhsh, A. Jalali, D. Jao, and V. Soukharev, "A post-quantum digital signature scheme based on supersingular isogenies," in *International Conference on Financial Cryptography and Data Security*. Springer, 2017, pp. 163–181.

[31] J. Vélu, "Isogenies entre courbes elliptiques," *Communications de lÁcademie royale des Sciences de Paris*, vol. 273, p. 238–241, 1971.

[32] D. Jao and V. Soukharev, "Isogeny-based quantum-resistant undeniable signatures," in *International Workshop on Post-Quantum Cryptography*. Springer, 2014, pp. 160–179.

[33] M. S. Srinath and V. Chandrasekaran, "Isogeny-based quantum-resistant undeniable blind signature scheme." *IACR Cryptology ePrint Archive*, vol. 2016, p. 148, 2016.