LLRing: Logarithmic Linkable Ring Signatures with Transparent Setup

Xiangyu Hui and Sid Chi-Kin Chau

CSIRO Data61, Australia
xiangyu.hui@data61.csiro.au, sid.chau@acm.org

March 11, 2024

Abstract. Linkable ring signatures are an important cryptographic primitive for anonymized applications, such as e-voting, e-cash and confidential transactions. To eliminate backdoor and overhead in trusted setup, transparent setup in the discrete logarithm or pairing settings has received considerable attention in practice. Recent advances have improved the proof sizes and verification efficiency of linkable ring signatures with transparent setup to achieve logarithmic bounds. Omniring (CCS '19) and RingCT 3.0 (FC '20) proposed linkable ring signatures in the discrete logarithm setting with logarithmic proof sizes with respect to the ring size, whereas DualDory (ESORICS '22) achieves logarithmic verifiability in the pairing setting. We make three novel contributions in this paper to improve the efficiency and soundness of logarithmic linkable ring signatures: (1) We report an attack on DualDory that breaks its linkability. (2) To eliminate such attacks, we present a new linkable ring signature scheme in the pairing setting with logarithmic verifiability. (3) We improve the verification efficiency of linkable ring signatures in the discrete logarithm setting, by a technique of reducing the number of group exponentiations for verification in Omniring by 50%. Furthermore, our technique is applicable to general inner-product relation proofs, which might be of independent interest. Finally, we empirically evaluate our schemes and compare them with the extant linkable ring signatures in concrete implementation.

Keywords: Zero-knowledge Proofs, Ring Signatures, Linkability, e-Voting, Confidential Transactions, Logarithmic Verificability, Pairing

1 Introduction

Ring signatures [RST01] ensure the anonymity of a signer with respect to an ad hoc group of public keys, which enables anonymous applications like anonymous whistle-blowing. To strengthen ring signatures, linkable ring signatures further guarantee that each anonymous signature can only be used at most once to eliminate duplication. For example, linkable ring signatures can prevent double voting in e-voting and double spending in confidential transactions. Nowadays, linkable ring signatures have been extensively utilized on blockchain platforms to enable confidential transactions. For example, Zerocash [BsCG+14], Monero [Noe15] and Tornado cash, use linkable ring signatures to unlink the sender and recipient in a transaction, while using a tag (or nullifier) that is unique to each public key to prevent double spending by the same key.

In the past, trusted setup, requiring a trusted third-party (or a group of appointed distributed parties) to generate structured reference strings for setup, has been employed in ring signatures [AOS02, ACJT00, BBS04, DKNS04]. Such generation will entail a security backdoor or incur cumbersome overhead. Transitioning into a scheme with a transparent setup without entrusting a third-party is critical for enabling decentralized trustless applications of e-voting, e-cash and blockchains.

Recently, new linkable ring signature schemes with a transparent setup have been developed in the discrete logarithmic setting ([LRR+19, YSL+20]). Further improvements have been made by pairing-friendly finite groups [BEHM22]. Particularly, certain functions of pairing have been provisioned through pre-compiled contracts on blockchain platforms [eth17]. Pairing-friendly finite groups are becoming more commonly utilized in real-world blockchain applications. In this paper, we focus on the improvement on *verification efficiency*. In decentralized systems, such as blockchain platforms, the computational overhead of verification will present a significant performance bottleneck. In particular, inefficient verification computation of linkable ring signatures will incur costly gas fees on miners.

Hence, improving verification efficiency is paramount to the successful deployment of linkable ring signatures in practice.

Regarding to the context of our work, we highlight several recent linkable ring signature schemes as follows:

- *Discrete Logarithm Setting*: Omniring [LRR+19] and RingCT 3.0 [YSL+20] proposed linkable ring signatures in the discrete logarithm setting with logarithmic proof sizes in terms of the ring size. Both schemes are based on Bulletproofs [BBB+18] (a popular recursive proof system for inner-product relations that compresses a linearly sized proof to a logarithmically sized one).
- *Pairing Setting*: DualDory [BEHM22] improved the efficiency of ring signatures in the pairing setting with logarithmic verifiability. DualDory is based on DualRing [YEL+21] and Dory [Lee21] (a recent recursive proof system for inner-product relations with logarithmic verifiability by leveraging precomputation).

Note that post-quantum ring signature schemes [YEL+21, LLNW16] based on lattice problems were also proposed in the literature, but they normally require a considerably larger proof size and higher overhead. In this paper, we focus on the discrete logarithm or pairing settings for the sake of practical efficiency and implementability on today's decentralized systems and blockchain platforms.

Our Contributions. In this paper, we present two novel linkable ring signature schemes (`LLRing–P` and `LLRing–DL`). First, we discovered an attack on the linkability of DualDory. The linkability of DualDory is attained by normalizing the public keys with a commitment of a known secret key. We show that an attacker can exploit the knowledge of more than two secret keys to enable a malleability attack to pass verification with an unrelated secret key.

To eliminate such attacks, we present a new linakble ring signature scheme `LLRing-P` in the pairing setting. Our idea is to restrict the selection of a known secret key by a unit basis vector in a similar manner as Omniring, but replacing Bulletproofs by Dory in the pairing setting. However, we need to overcome some obstacles with Dory, as it requires pre-defined generators for precomputation, unlike the ones with Bulletproofs. We remark that our technique for logarithmically verifiable linkable ring signatures also applies to logarithmically verifiable range proofs, which may be of independent interest.

Furthermore, we present `LLRing-DL` in the discrete logarithm setting that improves the verification efficiency of Bulletproofs-based linkable ring signatures (e.g., Omniring, RingCT 3.0). The verification time of Bulletproofs critically depends on the number of performed group exponentiations. We present a general technique to halve the number of group exponentiations in Bulletproofs at the expense of double proof sizes. The effect of our technique is on par with SwiftRange [WCL24], but is applicable to general inner-product relation proofs, which might be of independent interest. We then leverage precomputation to reduce the number of group exponentiations for verification.

We summarize our contributions in this paper as follows:

- ▶ We report an attack on Dualdory to break its linkability.
- ▶ We present `LLRing-P`, a linkable ring signature scheme in the pairing setting with logarithmic verifiability.
- ▶ We present `LLRing-DL` to improve the verification efficiency of linkable ring signatures in the discrete logarithm setting, by reducing the number of group exponentiations for verification in Omniring by 50%.

Note that both `LLRing-DL` and `LLRing-P` are able to improve verification efficiency by leveraging precomputation, which is ring-dependent but signature-independent.

Comparison to Related Work. This work provides significant improvements over the past studies of linkable ring signatures. Early linkable ring signatures rely on trusted setups acting on fixed rings [AOS02, ACJT00, BBS04, DKNS04]. Recent linkable ring signatures can be set up transparently with updatable rings. The state-of-the-art studies are on optimizing the space and computational efficiency of linkable ring signatures with transparent setup.

In particular, we compare several extant ring signature schemes in Table 1. One-out-of-many [GK15] is the first transparent scheme that achieves a logarithmic proof size, albeit with a super-linear proving time. [ACF21] utilizes compressed Σ-protocol to enable k-out-of-n proofs. Their approach is not indicated to support linkability. Omniring [LRR+19] and RingCT 3.0 [YSL+20] are two similar Bulletproofs-based schemes with logarithmic proof sizes, but the verification takes $2n$ group

Table 1: A comparison of various linkable/non-linkable ring signature schemes.

Scheme	Setting	Proof Size	Verification Time	Proving Time	Linkable
1-out-of-Many [GK15]	DL	$3\log n\|\mathbb{G}\|$	n $\mathbb{G}$ Exps	$n\log n$ $\mathbb{G}$ Exps	✓
Compressed k-out-of-n [ACF21]	DL	$2\log n\|\mathbb{G}\|$	$3n$ $\mathbb{G}$ Exps	$4n$ $\mathbb{G}$ Exps	✗
RingCT 3.0 [YSL+20]	DL	$2\log n\|\mathbb{G}\|$	$2n$ $\mathbb{G}$ Exps	$10n$ $\mathbb{G}$ Exps	✓
Omniring [LRR+19]	DL	$2\log n\|\mathbb{G}\|$	$2n$ $\mathbb{G}$ Exps	$10n$ $\mathbb{G}$ Exps	✓
DualRing-EC [YEL+21]	DL	$2\log n\|\mathbb{G}\|$	n $\mathbb{G}$ Exps	$4n$ $\mathbb{G}$ Exps	✗
DualDory [BEHM22]	SXDH	$6\log n\|\mathbb{G}_T\|$	$10\log n$ $\mathbb{G}_T$ Exps (PreComp[b]: $2n$ $\mathbb{P}$)	$10n$ $\mathbb{P}+5n$ $\mathbb{G}$ Exps	✗[a]
LLRing-DL (This work)	DL	$4\log n\|\mathbb{G}\|$	n $\mathbb{G}$ Exps (PreComp: n $\mathbb{G}$ Exps)	$10n$ $\mathbb{G}$ Exps	✓
LLRing-P (This work)	SXDH	$6\log n\|\mathbb{G}_T\|$	$10\log n$ $\mathbb{G}_T$ Exps (PreComp: $2n$ $\mathbb{P}+n$ $\mathbb{G}$ Exps)	$10n$ $\mathbb{P}+4n$ $\mathbb{G}$ Exps	✓

[a] We report an attack on the linkability of DualDory in this paper. [b] In addition to the verification for each signature, the verifier also performs ring-dependent (but signature-independent) precomputation in advance.

Note: In our performance estimation, we only state the most significant terms: $\mathbb{G}$ stands for group exponentiations and $\mathbb{P}$ for pairing operations.

exponentiations. DualRing [YEL+21] is a space-efficient but unlinkable scheme. DualDory [BEHM22] builds on DualRing and Dory to achieve logarithmic verifiability. However, it suffers from an attack on the linkability, as reported in this paper. Our LLRing schemes improve upon the previous schemes with faster verification time (which takes n group exponentiations - the fastest in discrete logarithm setting, and attains $O(\log n)$ verification runtime in the pairing setting).

Paper Organization. Sec. 2 provides a technical overview of our results. Sec. 3 presents the preliminaries and formal models of linkable ring signatures. Sec. 4 presents an attack on DualDory. Sec. 5 explains the Omniring scheme. Secs. 6 and 7 present LLRing schemes in discrete logarithm and pairing settings, respectively. Sec. 8 empirically evaluates our schemes and compares them with Omniring in concrete implementation. Sec. 9 concludes this paper. Some technical proofs are deferred to the Appendix.

2 Technical Overview

First, we define some basic notations, before we provide a technical overview of our results.

Vectors. Denote a cyclic group of prime order p by $\mathbb{G}$, and a ring of integers modulo p by $\mathbb{Z}_p$. Let $\mathbb{Z}_p^* \triangleq \mathbb{Z}_p \backslash \{0\}$. We denote a vector by bold font. For example, $\vec{\mathbf{a}} \triangleq (a_1, ..., a_n) \in \mathbb{Z}_p^n$ denotes a scalar vector, and $\vec{\mathbf{G}} \triangleq (G_1, ..., G_n) \in \mathbb{G}^n$ denotes a vector of generators from a group.

Define the following basic vector operations:

- $\vec{\mathbf{a}} + \vec{\mathbf{b}} \triangleq (a_1 + b_1, ..., a_n + b_n) \in \mathbb{Z}_p^n$
- $\vec{\mathbf{a}} \circ \vec{\mathbf{b}} \triangleq (a_1 \cdot b_1, ..., a_n \cdot b_n) \in \mathbb{Z}_p^n$
- $\vec{\mathbf{G}} \circ \vec{\mathbf{H}} \triangleq (G_1 \cdot H_1, ..., G_n \cdot H_n) \in \mathbb{G}^n$
- $\vec{\mathbf{G}}^{\circ \vec{\mathbf{a}}} \triangleq (G_1^{a_1}, ..., G_n^{a_n}) \in \mathbb{G}^n$
- $\vec{\mathbf{G}}^{\vec{\mathbf{a}}} \triangleq \prod_{i \in [n]} G_i^{a_i} \in \mathbb{G}$

Denote $\vec{\mathbf{G}}_L \triangleq (G_1, ..., G_{\frac{n}{2}})$ and $\vec{\mathbf{G}}_R \triangleq (G_{\frac{n}{2}+1}, ..., G_n)$ as the left-half and right-half sub-vectors of $\vec{\mathbf{G}}$.

Bilinear Pairing. A *bilinear pairing* is a mapping $\mathsf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ between two groups $\mathbb{G}_1, \mathbb{G}_2$ and a target group $\mathbb{G}_T$ (all of prime order p), satisfying the following:

- *Bilinearity*: For any $P, Q \in \mathbb{G}_1$, $R, S \in \mathbb{G}_2$, $a, b \in \mathbb{Z}_p$:

$$\mathsf{e}(P^a \cdot Q^b, R) = \mathsf{e}(P, R)^a \cdot \mathsf{e}(Q, R)^b, \quad \mathsf{e}(P, R^a \cdot S^b) = \mathsf{e}(P, R)^a \cdot \mathsf{e}(P, S)^b$$

- *Non-degeneracy*: If P is a generator of $\mathbb{G}_1$ and R is a generator of $\mathbb{G}_2$, then $\mathsf{e}(P, R)$ is a generator of $\mathbb{G}_T$.
- *Computability*: There exists an efficient algorithm to compute $\mathsf{e}(P, R)$ for any $P \in \mathbb{G}_1$, $R \in \mathbb{G}_2$.

We define an inner-product relation via bilinear pairing for given pair $(\vec{\mathbf{\Omega}} \in \mathbb{G}_1^n, \vec{\mathbf{\Theta}} \in \mathbb{G}_2^n, \vec{\mathbf{c}} \in \mathbb{Z}_p^n)$ by:

$$\mathsf{e}(\vec{\mathbf{\Omega}}, \vec{\mathbf{\Theta}}) \triangleq \prod_{i\in[n]} \mathsf{e}(\Omega_i, \Theta_i), \quad \mathsf{e}(\vec{\mathbf{\Omega}}, \vec{\mathbf{\Theta}})^{\vec{\mathbf{c}}} \triangleq \prod_{i\in[n]} \mathsf{e}(\Omega_i, \Theta_i)^{c_i} \tag{1}$$

Ring Signature. A *ring signature* scheme enables a member of an ad hoc group to sign a message anonymously within the group. A public key is constructed by $\mathsf{pk} = P^{\mathsf{sk}} \in \mathbb{G}$ from a secret key $\mathsf{sk} \in \mathbb{Z}_p$. Let n be the size of a ring of public keys. We denote $\vec{\mathbf{pk}}$ as a set of public keys. Given $\vec{\mathbf{pk}} = (\mathsf{pk}_i)_{i\in[n]}$, the prover aims to prove that he knows $j \in [n]$ and $\mathsf{sk} \in \mathbb{Z}_p$, such that $\mathsf{pk}_j = P^{\mathsf{sk}}$.

In the following, we briefly explain the high-level ideas of our results and linkable ring signature schemes.

2.1 Attack on DualDory

The basic idea of DualDory is that a prover first commits in advance, $\mathtt{Cm}[\mathsf{sk}]$ and $X \triangleq (\frac{\vec{\mathtt{Cm}}[\mathsf{sk}]}{\vec{\mathbf{pk}}})^{-\vec{\mathbf{c}}}$ (where $c_j = 0$ when $\mathsf{pk}_j = P^{\mathsf{sk}}$, otherwise c_i is a random number). We ignore zero knowledge for the moment. Next, the verifier issues a challenge requiring that $\tilde{c} = \sum_{i\in[n]} c_i$. The prover needs to set $c_j \triangleq \tilde{c} - \sum_{i\neq j} c_i$ and reveal $\vec{\mathbf{c}}$ to the verifier. Then, the prover can provide a proof-of-knowledge of a secret key in $\vec{\mathbf{pk}}$ by a Schnorr proof-of-knowledge on $(\frac{\vec{\mathtt{Cm}}[\mathsf{sk}]}{\vec{\mathbf{pk}}})^{\vec{\mathbf{c}}} \cdot X$.

However, if an attacker knows more than one secret key in $\vec{\mathbf{pk}}$, he has more than one degree of freedom in setting $\vec{\mathbf{c}}$ for the summation constraint of $\tilde{c}$. This enables a malleability attack on $(\frac{\vec{\mathtt{Cm}}[\mathsf{sk}]}{\vec{\mathbf{pk}}})^{\vec{\mathbf{c}}} \cdot X$, which can pass the verification of Schnorr proof-of-knowledge with arbitrary sk in $\mathtt{Cm}[\mathsf{sk}]$. Note that the possibility of an attacker knowing more than one secret key in the ring is very common, because any public keys can be provided by the users and the users may collude to share secret keys.

2.2 LLRing-P Linkable Ring Signature Scheme

To prevent an attacker's exploitation of more than one known secret key in $\vec{\mathbf{pk}}$, one can restrict $\vec{\mathbf{c}}$ to be a unit basis vector, such that $c_j = 1$, when $\mathsf{pk}_j = P^{\mathsf{sk}}$, otherwise $c_i = 0$. This idea has been utilized in Omniring, which applies Bulletproofs to check if $\vec{\mathbf{c}}$ is a unit basis vector.

In this paper, we seek to replace Bulletproofs by Dory to realize logarithmic verifiability. However, there are obstacles in adapting the Bulletproofs approach to Dory. Particularly, Bulletproofs checks the conformity of a unit basis vector by $\vec{\mathbf{G}}^{\vec{\mathbf{l}}} \cdot (\vec{\mathbf{H}}(y))^{\vec{\mathbf{r}}} \cdot K^{\langle \vec{\mathbf{l}}, \vec{\mathbf{r}} \rangle} \stackrel{?}{=} Z$, where $(\vec{\mathbf{l}}, \vec{\mathbf{r}})$ encode $\vec{\mathbf{c}}$ and its complement, and $\vec{\mathbf{H}}(y)$ are generators that depend on verifier-supplied challenge y. But Dory relies on precomputation with only pre-defined signature-independent generators. The same inner-product relation can not be adopted in Dory.

Hence, we devise a new approach in LLRing-P, without the adapting Bulletproofs approach. First, LLRing-P checks the conformity of a unit basis vector by $\langle \vec{\mathbf{c}}, \vec{\mathbf{1}} \rangle \stackrel{?}{=} 1$ and $\vec{\mathbf{c}} \circ (\vec{\mathbf{c}} - \vec{\mathbf{1}}) \stackrel{?}{=} \vec{\mathbf{0}}$, which can be checked via bilinear pairing: $\mathsf{e}(\vec{L'}, \vec{L})^{\vec{\mathbf{c}}} \stackrel{?}{=} \mathsf{e}(L', L)$ and $\mathsf{e}(\vec{\mathbf{G}}, \vec{\mathbf{H}})^{\vec{\mathbf{c}} \circ \vec{\mathbf{c}}} \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{G}}, \vec{\mathbf{H}})^{\vec{\mathbf{c}}}$, where $L', L, \vec{\mathbf{G}}, \vec{\mathbf{H}}$ are randomly selected generators. Then, we carefully integrate these bilinear pairing equations with a proof-of-knowledge of a secret key in the ring, which can be checked by Dory with logarithmic verifiability.

2.3 LLRing-DL Linkable Ring Signature Scheme

We also improve the verification efficiency of Omniring by a variant of Bulletproofs. In typical Bulletproofs, the generators $(\vec{\mathbf{G}}, \vec{\mathbf{H}})$ are distinct to prevent non-trivial logarithmic relations in $\vec{\mathbf{G}}^{\vec{\mathbf{l}}} \cdot \vec{\mathbf{H}}^{\vec{\mathbf{r}}}$, which incurs $2n$ group exponentiations in the single multi-exponentiation. We observe that an inner-product relation can be separately checked by $\vec{\mathbf{G}}^{\vec{\mathbf{l}}} \stackrel{?}{=} Z_1$ and $\vec{\mathbf{H}}^{\vec{\mathbf{r}}} \cdot K^{\langle \vec{\mathbf{l}}, \vec{\mathbf{r}} \rangle} \stackrel{?}{=} Z_2$. As a result, we can set identical generators $\vec{\mathbf{G}} = \vec{\mathbf{H}}$. Using random linear combination, the two separate relations can be checked together in a single multi-exponentiation, which only takes n group exponentiations, at the expense of doubling the proof size.

However, when we apply this variant of Bulletproofs to Omniring, we also need to account for pre-processing generators $\vec{\mathbf{G}} \triangleq \vec{\mathbf{pk}}^w \circ \vec{\mathbf{R}}$, where w is a verifier-supplied challenge. The pre-processing

of generators $\vec{\mathbf{G}}$ also takes n group exponentiations, if not handled in the single multi-exponentiation. In LLRing-DL, we are able to precompute $\vec{\mathbf{G}}$ in advance for each ring by different generators without a verifier-supplied challenge. Hence, LLRing-DL only takes n group exponentiations for verification and n group exponentiations for (ring-dependent but signature-independent) precomputation.

3 Preliminaries and Models

In this section, we present the preliminaries and define the formal models of linkable ring signatures. Let λ be the security level parameter and $\mathrm{negl}(\lambda)$ be a negligible function of λ. PPT denotes "probabilistic polynomial time". "$\xleftarrow{\$}$" denotes a uniformly random selection from a set.

Commitment Schemes. A commitment scheme is a mapping $\mathtt{Cm} : \mathcal{M}^n \times \mathcal{R} \to \mathcal{C}$ from a (vector) message space $\mathcal{M}^n$ and a random mask space $\mathcal{R}$ to a commitment space $\mathcal{C}$. A commitment scheme is *homomorphic*, if for any $\vec{\mathsf{m}}_1, \vec{\mathsf{m}}_2 \in \mathcal{M}^n, \mathsf{r}_1, \mathsf{r}_2 \in \mathcal{R}$:

$$\mathtt{Cm}(\vec{\mathsf{m}}_1; \mathsf{r}_1) \cdot \mathtt{Cm}(\vec{\mathsf{m}}_2; \mathsf{r}_2) = \mathtt{Cm}(\vec{\mathsf{m}}_1 + \vec{\mathsf{m}}_2; \mathsf{r}_1 + \mathsf{r}_2)$$

Definition 1 (Computationally Hiding). *A commitment scheme is* computationally hiding*, if a commitment does not reveal the message for any PPT adversary $\mathcal{A}$:*

$$\left| \mathtt{Pr}\left[b' = b \;\middle|\; \begin{array}{c} (\vec{\mathsf{m}}_1, \vec{\mathsf{m}}_2) \leftarrow \mathcal{A}, \\ \mathsf{r} \xleftarrow{\$} \mathcal{R}, b \xleftarrow{\$} \{1,2\}, \\ \mathsf{c} \leftarrow \mathtt{Cm}(\vec{\mathsf{m}}_b; \mathsf{r}), b' \leftarrow \mathcal{A}[\mathsf{c}] \end{array} \right] - \tfrac{1}{2} \right| \leq \mathrm{negl}(\lambda)$$

A commitment scheme is perfectly hiding*, if* $\mathrm{negl}(\lambda) = 0$.

Definition 2 (Computationally Binding). *A commitment scheme is* computationally binding*, if a commitment opens to only one message for any PPT adversary $\mathcal{A}$:*

$$\mathtt{Pr}\left[\begin{array}{l} \big(\mathtt{Cm}(\vec{\mathsf{m}}_1; \mathsf{r}_1) = \mathtt{Cm}(\vec{\mathsf{m}}_2; \mathsf{r}_2)\big) \\ \wedge\ \vec{\mathsf{m}}_1 \neq \vec{\mathsf{m}}_2 \end{array} \;\middle|\; \begin{array}{c} \vec{\mathsf{m}}_1 \xleftarrow{\$} \mathcal{M}, \vec{\mathsf{r}}_1 \xleftarrow{\$} \mathcal{R}, \\ (\vec{\mathsf{m}}_2, \mathsf{r}_2) \leftarrow \mathcal{A}[\vec{\mathsf{m}}_1, \mathsf{r}_1] \end{array} \right] \leq \mathrm{negl}(\lambda)$$

A commitment scheme is perfectly binding*, if* $\mathrm{negl}(\lambda) = 0$.

Pedersen commitment and AFGHO commitment are two homomorphic commitment schemes that are perfectly hiding and computationally binding.

Definition 3 (Pedersen Commitment). *Let $\mathcal{M} = \mathbb{Z}_p^n$, $\mathcal{R} = \mathbb{Z}_p^*$ and $\mathcal{C} = \mathbb{G}$ of order p. Let $\vec{\mathbf{G}} \xleftarrow{\$} \mathbb{G}^n, Q \xleftarrow{\$} \mathbb{G}$ be randomly selected generators. Define Pedersen commitment by*

$$\mathtt{Cm}(\vec{\mathsf{m}}; \mathsf{r}) \triangleq \vec{\mathbf{G}}^{\vec{\mathsf{m}}} \cdot Q^{\mathsf{r}} = \Big(\prod_{i \in [n]} G_i^{\mathsf{m}_i} \Big) \cdot Q^{\mathsf{r}}$$

Definition 4 (AFGHO Commitment). *Let $\mathcal{M} = \mathbb{Z}_p^n$, $\mathcal{R} = \mathbb{Z}_p^*$ and $\mathcal{C} = \mathbb{G}_T$ of order p. Let $\vec{\mathbf{G}} \xleftarrow{\$} \mathbb{G}_1^n, \vec{\mathbf{\Lambda}} \xleftarrow{\$} \mathbb{G}_2^n, Q_1 \xleftarrow{\$} \mathbb{G}_1, Q_2 \xleftarrow{\$} \mathbb{G}_2$ be randomly selected generators. Let $\mathtt{Q} \triangleq \mathtt{e}(Q_1, Q_2)$. Define AFGHO commitment by*

$$\mathtt{Cm}(\vec{\mathsf{m}}; \mathsf{r}) \triangleq \mathtt{e}(\vec{\mathbf{G}}, \vec{\mathbf{\Lambda}})^{\vec{\mathsf{m}}} \cdot \mathtt{Q}^{\mathsf{r}} = \Big(\prod_{i \in [n]} \mathtt{e}(G_i, \Lambda_i)^{\mathsf{m}_i} \Big) \cdot \mathtt{Q}^{\mathsf{r}}$$

Cryptographic Assumptions. We define several cryptographic assumptions as follows. The DDH assumption implies the CDH assumption, which implies the DLog assumption. The SXDH assumption implies the DPair and DDH assumptions.

Definition 5 (Discrete Logarithm (DLog)). *The DLog assumption holds for any PPT adversary $\mathcal{A}$:*

$$\mathtt{Pr}\left[\vec{\mathbf{x}} \leftarrow \mathcal{A}[\vec{\mathbf{G}}], \;\middle|\; \begin{array}{c} \mathbb{G} \leftarrow \mathtt{Setup}[1^\lambda], \\ \vec{\mathbf{G}} \xleftarrow{\$} \mathbb{G} \end{array} \right] \leq \mathrm{negl}(\lambda)$$

As a result of the DLog assumption, non-trivial discrete logarithm relations among random generators $\vec{\mathbf{G}}$ cannot be discovered by any PPT adversary.

Definition 6 (Computational Diffie–Hellman (CDH)). *Given a random generator $G \xleftarrow{\$} \mathbb{G}$ and a tuple (G^a, G^b), where $(a, b) \xleftarrow{\$} \mathbb{Z}_p^{*2}$ are selected at random, the CDH assumption holds, if G^{ab} is computationally hard for any PPT adversary.*

Definition 7 (Decisional Diffie–Hellman (DDH)). *Given a random generator $G \xleftarrow{\$} \mathbb{G}$ and a tuple (G^a, G^b, G^c), where $(a, b, c) \xleftarrow{\$} \mathbb{Z}_p^{*3}$ are selected at random, the DDH assumption holds, if G^c is computationally indistinguishable from G^{ab} for any PPT adversary.*

Definition 8 (Symmetric External Diffie-Hellman (SXDH)). *Given a bilinear pairing $\mathsf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ and random generators $G \xleftarrow{\$} \mathbb{G}_1, H \xleftarrow{\$} \mathbb{G}_2$, the SXDH assumption holds, if the DDH assumption holds for $\mathbb{G}_1$ and $\mathbb{G}_2$, and the following distributions are computationally indistinguishable for any PPT adversary:*

1. *Tuple $(G, G^a, H, H^b, \mathsf{e}(G, H)^{ab})$ where $(a, b) \xleftarrow{\$} \mathbb{Z}_p^{*2}$*
2. *Tuple (G, G^a, H, H^b, T) where $(a, b) \xleftarrow{\$} \mathbb{Z}_p^{*2}, T \xleftarrow{\$} \mathbb{G}_T$*

Definition 9 (Double Pairing (DPair)). *Given a bilinear pairing $\mathsf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ and a random element vector $\vec{\mathbf{G}} \xleftarrow{\$} \mathbb{G}_1^n$, the DPair assumption holds, if it is computationally hard to produce $\vec{\mathbf{H}} \in \mathbb{G}_2^n$ for any PPT adversary, such that $\mathsf{e}(\vec{\mathbf{G}}, \vec{\mathbf{H}}) = 1$.*

Zero-Knowledge Arguments of Knowledge. An argument system is consisted of three PPT algorithms $(\mathcal{G}, \mathcal{P}, \mathcal{V})$, where $\mathcal{G}$ is the setup algorithm for public parameters pp, $\mathcal{P}$ and $\mathcal{V}$ are the prover and verifier algorithms. Denote the communication transcript between the prover and verifier by $\mathsf{tr} \leftarrow \langle \mathcal{P}(\cdot), \mathcal{V}(\cdot) \rangle$. At the end, the transcript will produce a binary decision: $\mathtt{Accept}[\mathsf{tr}] \in \{0, 1\}$.

Denote a polynomial-time decidable tertiary relation by $\mathscr{R} \subset \{0,1\}^{*3}$. A language dependent on pp is defined as $\mathscr{L}_{\mathscr{R}}^{\mathsf{pp}} \triangleq \{x \mid \exists \omega : (\mathsf{pp}, x, \omega) \in \mathscr{R}\}$, where ω is a witness for a statement x in the relation $(\mathsf{pp}, x, \omega) \in \mathscr{R}$. A key class of argument systems are *zero-knowledge arguments of knowledge* (e.g., ring signatures).

Definition 10 (Argument of Knowledge). *An argument system $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ is called an* argument of knowledge *for relation $\mathscr{R}$, if it satisfies the perfect completeness (Definition (11)) and CWE (Definition (12)).*

Definition 11 (Completeness). *An argument system $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ satisfies* completeness*, if for any PPT adversary $\mathcal{A}$:*

$$\Pr\left[\begin{array}{c}\mathtt{Accept}[\mathsf{tr}] \\ = 1\end{array}\middle|\begin{array}{c}\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), \\ (\mathsf{pp}, x, \omega) \in \mathscr{R}, \\ \mathsf{tr} \leftarrow \langle \mathcal{P}(\mathsf{pp}, x, \omega), \mathcal{V}(\mathsf{pp}, x) \rangle\end{array}\right] \geq 1 - \mathrm{negl}(\lambda)$$

An argument system satisfies perfect completeness*, if* $\mathrm{negl}(\lambda) = 0$.

We are interested in *knowledge sound* arguments. Informally, if a prover always passes the verification, then there is a way to extract the witness out of the transcripts, possibly by the idea of rewinding, as captured by the notion of CWE.

Definition 12 (Computational Witness-Extended Emulation (CWE)). *An argument system $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ satisfies CWE, if there exists an expected polynomial-time emulator $\mathcal{E}$, such that for any adversaries $\mathcal{A}_1, \mathcal{A}_2$:*

$$\left|\Pr\left[\mathcal{A}_1[\mathsf{tr}] = 1 \middle| \begin{array}{c}\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), \\ (x, \tilde{w}, \tilde{\mathcal{P}}) \leftarrow \mathcal{A}_2[\mathsf{pp}], \\ \mathsf{tr} \leftarrow \langle \tilde{\mathcal{P}}(\mathsf{pp}, x, \tilde{w}), \mathcal{V}(\mathsf{pp}, x) \rangle\end{array}\right] - \Pr\left[\begin{array}{c}\mathcal{A}_1[\mathsf{tr}'] = 1 \\ \wedge \big(\mathtt{Accept}[\mathsf{tr}'] = 1 \ \Rightarrow \\ (\mathsf{pp}, x, w') \in \mathscr{R}\big)\end{array}\middle| \begin{array}{c}\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), \\ (x, \tilde{w}, \tilde{\mathcal{P}}) \leftarrow \mathcal{A}_2[\mathsf{pp}], \\ (\mathsf{tr}', w') \leftarrow \mathcal{E}^{\mathcal{O}}[\mathsf{pp}, x]\end{array}\right]\right| \leq \mathrm{negl}(\lambda)$$

where $\tilde{\mathcal{P}}$ is a deterministic polynomial-time algorithm, $\mathcal{A}_1[\mathsf{tr}]$ recognizes the transcripts that are produced by $\tilde{\mathcal{P}}$, and $\mathcal{O}$ is a rewindable oracle that can rewind the transcript $\langle \tilde{\mathcal{P}}(\mathsf{pp}, x, \tilde{w}), \mathcal{V}(\mathsf{pp}, x) \rangle$ and control the randomness in the verifier.

Definition 13 (Public Coin). *An argument system $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ is called* public-coin*, if the verifier chooses her messages uniformly at random, independent from the messages sent by the prover. Let e be the public-coin challenge. The transcript of a public-coin argument system is defined as $\mathsf{tr} = \langle \mathcal{P}(\mathsf{pp}, x, \omega), \mathcal{V}(\mathsf{pp}, x; e) \rangle$.*

Definition 14 (Computationally Special Soundness). *An argument system $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ is* computationally γ-special sound*, if there exists an expected polynomial-time emulator $\mathcal{E}$, such that for any PPT adversary $\mathcal{A}$:*

$$\Pr\left[\begin{array}{c}\left(\bigwedge_{i=1}^{\gamma} \mathtt{Accept}[\mathsf{tr}_i] = 1\right) \Rightarrow \\ (\mathsf{pp}, x, \omega) \in \mathscr{R}\end{array}\middle| \begin{array}{c}\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), \\ (x, (\mathsf{tr}_i)_{i=1}^{\gamma}) \leftarrow \mathcal{A}[\mathsf{pp}], \\ \omega \leftarrow \mathcal{E}[\mathsf{pp}, x, (\mathsf{tr}_i)_{i=1}^{\gamma}]\end{array}\right] \geq 1 - \mathrm{negl}(\lambda)$$

A $(2\mu{+}1)$-move, public-coin interactive argument system is computationally $(\gamma_1, ..., \gamma_\mu)$-special sound*, if there exists an expected polynomial-time emulator capable of producing a witness ω with probability $1-\mathsf{negl}(\lambda)$ for a given statement x provided with a set of accepting transcripts arranged in a $(\gamma_1, ..., \gamma_\mu)$-tree structure, which represents $\prod_{i=1}^{\mu} \gamma_i$ accepting transcripts, such that the nodes representing the prover's messages, the edges representing the verifier's challenges, and the paths from the root to leaf nodes representing the accepting transcripts.*

Lemma 15 (Forking Lemma [BCC$^+$16]). *Let $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ be a $(2\mu{+}1)$-move, public-coin interactive argument system. Let $\mathcal{E}$ be an expected polynomial-time emulator that succeeds with probability $1 - \mathsf{negl}(\lambda)$ in extracting a witness from a $(\gamma_1, ..., \gamma_\mu)$-tree of accepting transcripts. If $\prod_{i=1}^{\mu} \gamma_i$ is bounded above by a polynomial in λ, then $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ satisfies CWE.*

By Forking Lemma (Lemma (15)), it suffices to show CWE by computationally $(\gamma_1, ..., \gamma_\mu)$-special soundness.

We are also interested in *zero-knowledge* arguments that do not leak the information about the witness beyond what can be inferred from the truth of the statement.

Definition 16 (Special Honest-Verifier Zero-Knowledge (SHVZK)). *A public-coin argument system $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ satisfies SHVZK, if there exists a PPT simulator $\mathcal{S}$, such that for any PPT adversary $\mathcal{A}$:*

$$\left|\Pr\left[\begin{array}{c}\mathtt{Accept}[\mathsf{tr}] \\ = 1 \\ \wedge\ (\mathsf{pp}, x, \omega) \in \mathscr{R}\end{array}\middle| \begin{array}{c}\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), \\ (x, \omega, e) \leftarrow \mathcal{A}[\mathsf{pp}], \\ \mathsf{tr} \leftarrow \langle \mathcal{P}(\mathsf{pp}, x, \omega), \mathcal{V}(\mathsf{pp}, x; e) \rangle\end{array}\right] - \Pr\left[\begin{array}{c}\mathtt{Accept}[\mathsf{tr}] \\ = 1 \\ \wedge\ (\mathsf{pp}, x, \omega) \in \mathscr{R}\end{array}\middle| \begin{array}{c}\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), \\ (x, \omega, e) \leftarrow \mathcal{A}[\mathsf{pp}], \\ \mathsf{tr} \leftarrow \mathcal{S}[\mathsf{pp}, x; e]\end{array}\right]\right| \leq \mathrm{negl}(\lambda)$$

Definition 17 (Fiat-Shamir Transformation). *A multi-move interactive public-coin argument of knowledge can be converted to a non-interactive argument of knowledge by replacing the public-coin challenges with the output of a cryptographic hash function. The hash function will produce seemingly random output and can be a suitable replacement for the verifier.*

The Fiat-Shamir transformation can be applied to make our interactive protocols non-interactive using the random oracle model in the security proofs [FS87]. This is especially useful for reducing a logarithmic number of moves to a single move.

3.1 Ring Signature Schemes

As a special class of zero-knowledge arguments of knowledge, *ring signature* schemes allow a member of an ad hoc group to sign a message anonymously within the group by a publicly verifiable signature. A *ring* is a set of known public keys, whose corresponding secret keys remain private with the individual members. A public key is constructed by $\mathsf{pk} = P^{\mathsf{sk}} \in \mathbb{G}$ (or $\mathbb{G}_1$) from a secret key $\mathsf{sk} \in \mathbb{Z}_p$. Given a ring $\vec{\mathbf{pk}} = (\mathsf{pk}_i)_{i \in [n]}$, the prover aims to prove that $j \in [n]$ and $\mathsf{sk} \in \mathbb{Z}_p$, such that $\mathsf{pk}_j = P^{\mathsf{sk}}$.

Note that this paper considers *interactive ring signature protocols*, which can be converted to non-interactive publicly verifiable ring signature schemes via Fiat-Shamir transformation by replacing the verifier-supplied challenges by hashes of the previous commitments and messages [FS87].

Moreover, we consider prefix-dependent ring signature schemes in this paper. Let $\mathcal{F}$ be a set of feasible prefixes, which may represent a set of potential topics or domains that a ring signature will be applied to.

A ring signature scheme commonly consists of the following methods:

- $\underline{\mathtt{Setup}[1^\lambda] \mapsto \mathsf{pp}}$: This method is given a security level parameter λ and produces a public parameter pp to set up the scheme.
- $\underline{\mathtt{KeyGen}[\mathsf{pp}] \mapsto (\vec{\mathbf{pk}}, \vec{\mathbf{sk}})}$: This method is given a public parameter pp and produces a set of n randomly selected public keys $\vec{\mathbf{pk}}$ and the corresponding secret keys $\vec{\mathbf{sk}}$.
- $\underline{\mathtt{Sign}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}, \mathsf{sk}] \mapsto \sigma}$: This method is given a public parameter pp, a ring $\vec{\mathbf{pk}}$, a prefix $\mathsf{f} \in \mathcal{F}$, a message m and a secret key sk corresponding to one of the public keys in $\vec{\mathbf{pk}}$. It produces a signature σ.
- $\underline{\mathtt{Verify}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}, \sigma] \mapsto \{0,1\}}$: This method checks if the message m and signature σ are consistent under prefix f, and σ is signed by a secret key corresponding to one of the public keys in $\vec{\mathbf{pk}}$. It returns 1 for a valid signature or 0 otherwise.

To model the adversary's behavior, we define two oracles that an adversary can access to. The *corruption oracle* captures the scenarios, where an adversary can obtain the secret key from a corrupt source. The *signing oracle* captures the scenarios, where the adversary can observe the signatures from an honest user given the public key, but not his secret key.

Definition 18 (Corruption Oracle $\mathcal{CO}$). *When $\mathcal{CO}$ is queried with a public key pk, it returns its corresponding secret key sk. After the query, $\mathcal{CO}$ adds $\{\sigma, (\mathsf{f}, \mathsf{pk})\}$ to the set $\vec{\mathbf{pk}}_{\mathcal{CO}}$.*

Definition 19 (Signing Oracle $\mathcal{SO}$). *When $\mathcal{SO}$ is queried with a public parameter pp, a prefix f, a message m and a public key pk, it returns a signature σ using the corresponding secret key sk and method $\mathtt{Sign}$. After the query, $\mathcal{SO}$ adds $\{\sigma, (\mathsf{f}, \mathsf{pk})\}$ to the set $\Sigma_{\mathcal{SO}}$.*

We define three major security properties for a ring signature scheme [BDH+19, BEHM22]. *Completeness* captures the notion of correctness, if executed faithfully. *Anonymity* captures the notion that no adversary can distinguish the signatures from two different secret keys in a way better than random guessing. *Unforgeability* captures the notion that no adversary can forge a valid signature without knowing the secret key, even though he can observe the signatures from an honest user or access the other secret keys from the ring.

Definition 20 (Completeness). *A ring signature satisfies* completeness*, if*

$$\Pr\left[\mathtt{Verify}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}, \sigma] = 1 \;\middle|\; \begin{array}{c} \mathsf{pp} \leftarrow \mathtt{Setup}(1^\lambda), \\ (\vec{\mathbf{pk}}, \vec{\mathbf{sk}}) \leftarrow \mathtt{KeyGen}[\mathsf{pp}], \\ \mathsf{f} \in \mathcal{F},\ \mathsf{sk} \in \vec{\mathbf{sk}}, \\ \sigma \leftarrow \mathtt{Sign}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}, \mathsf{sk}] \end{array}\right] \geq 1 - \mathrm{negl}(\lambda)$$

A ring signature scheme satisfies perfect completeness*, if* $\mathrm{negl}(\lambda) = 0$.

Definition 21 (Anonymity). *A ring signature satisfies* anonymity*, if for any PPT adversary $\mathcal{A}$ such that*

$$\left|\Pr\left[\begin{array}{c} b' \leftarrow \mathcal{A}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}, \sigma], \\ b' = b \ \wedge\ i_1 \neq i_2 \\ \wedge(\mathsf{pk}^*_{i_1}, \mathsf{pk}^*_{i_2}) \subseteq \vec{\mathbf{pk}} \backslash \vec{\mathbf{pk}}_{\mathcal{CO}} \\ \wedge(\mathsf{f}, \mathsf{pk}^*_{i_1}) \notin \Sigma_{\mathcal{SO}} \\ \wedge(\mathsf{f}, \mathsf{pk}^*_{i_2}) \notin \Sigma_{\mathcal{SO}} \end{array} \;\middle|\; \begin{array}{c} \mathsf{pp} \leftarrow \mathtt{Setup}(1^\lambda), \\ (\vec{\mathbf{pk}}, \vec{\mathbf{sk}}) \leftarrow \mathtt{KeyGen}[\mathsf{pp}], \\ (i_1, i_2, \mathsf{f}, \mathsf{m}, \vec{\mathbf{pk}}^*) \leftarrow \mathcal{A}^{\mathcal{CO}, \mathcal{SO}}[\mathsf{pp}, \vec{\mathbf{pk}}], \\ b \xleftarrow{\$} \{1,2\}, \\ \sigma \leftarrow \mathtt{Sign}[\mathsf{pp}, \vec{\mathbf{pk}}^*, \mathsf{f}, \mathsf{m}, \mathsf{sk}_{i_b}] \end{array}\right] - \tfrac{1}{2}\right| \leq \mathrm{negl}(\lambda)$$

Definition 22 (Unforgeability). *A ring signature satisfies* unforgeability*, if for any PPT adversary $\mathcal{A}$ such that*

$$\Pr\left[\begin{array}{c} \mathtt{Verify}[\mathsf{pp}, \vec{\mathbf{pk}}', \mathsf{f}, \mathsf{m}, \sigma] = 1 \\ \wedge\ \vec{\mathbf{pk}}' \subseteq \vec{\mathbf{pk}} \backslash \vec{\mathbf{pk}}_{\mathcal{CO}} \\ \wedge\ \sigma \notin \Sigma_{\mathcal{SO}} \end{array} \;\middle|\; \begin{array}{c} \mathsf{pp} \leftarrow \mathtt{Setup}(1^\lambda), \\ (\vec{\mathbf{pk}}, \vec{\mathbf{sk}}) \leftarrow \mathtt{KeyGen}[\mathsf{pp}], \\ (\vec{\mathbf{pk}}', \mathsf{f}, \mathsf{m}, \sigma) \leftarrow \mathcal{A}^{\mathcal{CO}, \mathcal{SO}}[\mathsf{pp}, \vec{\mathbf{pk}}] \end{array}\right] \leq \mathrm{negl}(\lambda)$$

3.2 Prefix Linkable Ring Signature Schemes

Prefix linkable ring signature schemes provide an additional method to link any pair of signatures if they are signed from the same secret key under a given prefix:

- $\underline{\texttt{Link}[\mathsf{pp}, \mathsf{f}, \sigma, \sigma'] \mapsto \{0,1\}}$: This method is given a public parameter pp and a pair of signatures (possibly signing on different messages). It returns 1 if (σ, σ') are signed from the same secret key under prefix f or 0 otherwise.

We next define two additional security properties for a linkable ring signature scheme [BDH+19, BEHM22]. *Prefix linkability* captures the notion that no adversary can generate $n+1$ valid but pairwise unlinked signatures from n secret keys under the same prefix. *Non-Slanderability* captures the notion that no adversary can forge a valid signature without knowing the secret key that is linked to an honest user, by observing the signatures from an honest user or any other secret keys from the ring.

Definition 23 (Prefix Linkability). *A linkable ring signature satisfies* prefix linkability*, if for any PPT adversary $\mathcal{A}$ such that*

$$\Pr\left[\begin{array}{c|c} \texttt{Verify}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}_i, \sigma_i] & \mathsf{pp} \leftarrow \texttt{Setup}(1^\lambda), \\ = 1, \forall i \in [n+1] & (\vec{\mathbf{pk}}, \vec{\mathbf{sk}}) \leftarrow \texttt{KeyGen}[\mathsf{pp}], \\ \wedge\ \texttt{Link}[\mathsf{pp}, \mathsf{f}, \sigma_i, \sigma_j] = 0, & \mathsf{f} \leftarrow \mathcal{A}^{\mathcal{CO}, \mathcal{SO}}[\mathsf{pp}, \vec{\mathbf{pk}}], \\ \forall i \neq j \in [n+1] & (\mathsf{m}_i, \sigma_i)_{i \in [n+1]} \leftarrow \mathcal{A}^{\mathcal{CO}, \mathcal{SO}}[\mathsf{pp}, \vec{\mathbf{pk}}] \end{array}\right] \leq \mathrm{negl}(\lambda)$$

Definition 24 (Non-Slanderability). *A linkable ring signature satisfies* non-slanderability*, if for any PPT adversary $\mathcal{A}$ such that*

$$\Pr\left[\begin{array}{c|c} \texttt{Verify}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}'', \sigma''] = 1 & \mathsf{pp} \leftarrow \texttt{Setup}(1^\lambda), \\ \wedge\ \texttt{Verify}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}', \sigma'] = 1 & (\vec{\mathbf{pk}}, \vec{\mathbf{sk}}) \leftarrow \texttt{KeyGen}[\mathsf{pp}], \\ \wedge\ \texttt{Link}[\mathsf{pp}, \mathsf{f}, \sigma', \sigma''] = 1 & (\mathsf{f}, \mathsf{m}', \sigma') \leftarrow \mathcal{A}^{\mathcal{SO}}[\mathsf{pp}, \vec{\mathbf{pk}}], \\ \wedge\ \sigma' \notin \Sigma_{\mathcal{SO}} & (\mathsf{m}'', \sigma'') \leftarrow \mathcal{A}^{\mathcal{CO}, \mathcal{SO}}[\mathsf{pp}, \vec{\mathbf{pk}}] \end{array}\right] \leq \mathrm{negl}(\lambda)$$

We follow a 2-staged definition [BDH+19] that allows the adversary to generate signature σ'' with full access to the secret keys after producing the "slandering" signature σ'.

3.3 Linkability Schema and Tags

To implement linkability in ring signatures, in this paper we follow a common schema from DualDory:

1. *Secret Key Commitment*: First, a signer commits the known secret keys by a Pedersen/AFGHO commitment. The signer commits sk by

$$\texttt{Cm}^P[\mathsf{sk}; \mathsf{r}_{\mathsf{cm}}] \triangleq \begin{cases} P^{\mathsf{sk}} \cdot Q^{\mathsf{r}_{\mathsf{cm}}}, & \text{for Pedersen commitment} \\ \mathsf{e}(P, L)^{\mathsf{sk}} \cdot \mathtt{Q}^{\mathsf{r}_{\mathsf{cm}}}, & \text{for AFGHO commitment} \end{cases}$$

   where $P \xleftarrow{\$} \mathbb{G}$ (or $\mathbb{G}_1$) and $L \xleftarrow{\$} \mathbb{G}_2$.
2. *Ring Signature Proof*: Then, the signer provides a ring signature σ with a proof-of-knowledge showing that the committed secret key (sk) correspond to a public keys in the ring $\vec{\mathbf{pk}}$. The details of ring signature proof construction will be explained in several linkable ring signature schemes in the next sections.
3. *Tags*: The signer also provides a *tag* ($\mathtt{tag}$), which is generated via a one-way tag function $\texttt{Tag}[\cdot]$ from a prefix f and a secret key sk, such that method $\texttt{Link}$ can track the list of used tags to determine if sk has been used before under a given f. A tag should uniquely identify a signature under the same prefix and secret key. Let $\texttt{Hash}$ be a hash mapping from a prefix f to a group element in a separate group $\mathbb{G}'$. We define a prefix-dependent tag function by

$$\texttt{Tag}[\mathsf{f}, \mathsf{sk}] \triangleq (\texttt{Hash}[\mathsf{f}])^{\mathsf{sk}} \in \mathbb{G}'$$

4. *Tag Proof*: Finally, the signer is required to provide a zero-knowledge proof-of-knowledge showing the same secret keys in both the commitment and the tag. Fig. 1 presents a Schnorr proof-of-knowledge to check if the same secret key present in $\texttt{Cm}[\mathsf{sk}]$ and $\mathtt{tag}$. It is based on the tag proof in [BEHM22].

Theorem 25 ([BEHM22]). *Π_{tag} satisfies perfect completeness, perfect SHVZK and CWE under the DLog assumption.*

Fig. 1: Zero-knowledge proof-of-knowledge tag proof

$$\Pi_{\mathrm{tag}}\left[\mathsf{f}\in\mathcal{F},\mathsf{cm};\ \mathsf{sk}\in\mathbb{Z}_p^k,\mathsf{r}_{\mathsf{cm}}\in\mathbb{Z}_p^*\right]$$

$$\mathcal{P}\text{'s Input}:\mathsf{cm}\triangleq\mathtt{Cm}[\mathsf{sk};\mathsf{r}_{\mathsf{cm}}] \tag{2}$$

$$\mathcal{P}:\mathsf{r}_B,\mathsf{a}\xleftarrow{\$}\mathbb{Z}_p^* \tag{3}$$

$$\mathcal{P}\Rightarrow\mathcal{V}:\mathtt{tag}\triangleq(\mathtt{Hash}[\mathsf{f}]^{\mathsf{sk}})\in\mathbb{G}',\quad \mathsf{A}\triangleq(\mathtt{Hash}[\mathsf{f}])^{\mathsf{a}}\in\mathbb{G}',\quad B\triangleq\mathtt{Cm}[\mathsf{a};\mathsf{r}_B] \tag{4}$$

$$\mathcal{P}\Leftarrow\mathcal{V}:\rho\xleftarrow{\$}\mathbb{Z}_p^* \tag{5}$$

$$\mathcal{P}\Rightarrow\mathcal{V}:\mathsf{a}'\triangleq\mathsf{a}+\rho\cdot\mathsf{sk}\in\mathbb{Z}_p,\quad \mathsf{r}'\triangleq\mathsf{r}_B+\rho\cdot\mathsf{r}_{\mathsf{cm}}\in\mathbb{Z}_p \tag{6}$$

$$\mathcal{V}:\text{Check}\begin{cases}(\mathtt{Hash}[\mathsf{f}])^{a'}\stackrel{?}{=}A\cdot\mathtt{tag}_i^{\rho}\\ \mathtt{Cm}[\mathsf{a}';\mathsf{r}']\stackrel{?}{=}B\cdot\mathsf{cm}^{\rho}\end{cases} \tag{7}$$

4 DualDory

In this section, we present DualDory and an attack to break its linkability. DualDory [BEHM22] is a linkable ring signature scheme, based on non-linkable ring signature scheme DualRing [YEL+21], which adds linkability to DualRing and applies Dory [Lee21] to compress the proof size and verification time for inner-product relations.

In the following, we focus on the *uncompressed* construction of DualDory, which is an interactive version of linkable DualRing, without using Dory. Uncompressed DualDory has a linear proof size and linear verification time. Note that Dory only enhances efficiency by compressing the proof size and verification time to be logarithmically bounded. But if uncompressed DualDory is not linkable, then compressed DualDory is also not linkable.

The basic idea of DualRing is to treat the ring $(\vec{\mathbf{pk}})$ as the generators in a commitment. The prover first commits $\vec{\mathbf{c}}$ as commitment $X=(\mathbf{pk})^{-\vec{\mathbf{c}}}$, such that $c_i=0$ if the secret key to the i-th pk_i is known, otherwise c_i is random. After the commitment, the verifier issues a challenge $\tilde{c}$ requiring $\sum_{i\in[n]}c_i=\tilde{c}$. In order to satisfy this summation constraint, the prover needs to set the previously zero-valued c_i to be the difference of the previous sum and $\tilde{c}$. Finally, the prover reveals $\vec{\mathbf{c}}$ and provides a proof-of-knowledge of a secret key in $\vec{\mathbf{pk}}$ by a Schnorr proof-of-knowledge on $(\mathbf{pk})^{\vec{\mathbf{c}}}\cdot X$.

Recall that a public key is constructed by $\mathsf{pk}_i=P^{\mathsf{sk}_i}$ from a secret key $\mathsf{sk}\in\mathbb{Z}_p$. Given $\vec{\mathbf{pk}}=(\mathsf{pk}_i)_{i\in[n]}$, the prover aims to prove $j\in[n]$ and $\mathsf{sk}\in\mathbb{Z}_p$, such that $\mathsf{pk}_j=P^{\mathsf{sk}}$.

DualDory extends DualRing by replacing the public keys by normalized ratios $(\widetilde{\mathsf{pk}}_i\triangleq\frac{\mathsf{cm}}{\mathsf{pk}_i})_{i\in[n]}$ (where $\mathsf{cm}=\mathtt{Cm}[\mathsf{sk}]$ is a commitment of the known secret key sk in the ring). The uncompressed DualDory protocol is presented in Fig. 2a.

4.1 Malleability Attack on DualDory

We report a possible attack to break the linkability of DualDory. The attack arises from the fact that an attacker may know more than one secret key in the ring, because any public keys can be provided by the users and the users may collude to share secret keys. Hence, an attacker can set multiple zero-valued c_i's in commitment X. As a result, the attacker has more than one degree of freedom of setting zero-valued c_i's to satisfy the summation constraint $(\sum_{i\in[n]}c_i=\tilde{c})$, which enables a malleability attack on the commitment (cm) to pass the verification of Schnorr proof-of-knowledge with arbitrary sk' in cm.

The attack on DualDory is presented in Fig. 2b. Let us explain how the attack works. We suppose that a malicious prover $(\mathcal{P}')$ knows two different secret keys $j_1,j_2\in[n]$, such that $\mathsf{pk}_{j_1}\triangleq P^{\mathsf{sk}_1}$ and $\mathsf{pk}_{j_2}\triangleq P^{\mathsf{sk}_2}$. Now the prover can pass the verification using arbitrary sk' in $\mathsf{cm}'\triangleq P^{\mathsf{sk}'}\cdot Q^{\mathsf{r}_{\mathsf{cm}}}$. We set (c_{j_1},c_{j_2}) according to Eqn. (20). Hence, we obtain $\mathsf{sk}'(c_{j_1}+c_{j_2})=\mathsf{sk}_1c_{j_1}+\mathsf{sk}_2c_{j_2}$, and

$$\widetilde{\mathsf{pk}}_{j_1}^{c_{j_1}}\cdot\widetilde{\mathsf{pk}}_{j_2}^{c_{j_2}}=\frac{(P^{\mathsf{sk}'}\cdot Q^{\mathsf{r}_{\mathsf{cm}}})^{c_{j_1}+c_{j_2}}}{P^{\mathsf{sk}_1c_{j_1}}\cdot P^{\mathsf{sk}_2c_{j_2}}}=Q^{\mathsf{r}_{\mathsf{cm}}(c_{j_1}+c_{j_2})}=Q^{\mathsf{r}_{\mathsf{cm}}c'} \tag{23}$$

Note that the falsified commitment (cm') can pass the verification in Eqn (22) as follows:

$$Q^{\mathsf{r}'}=Q^{\mathsf{r}_X+\mathsf{r}_{\mathsf{cm}}c'}=Q^{\mathsf{r}_X}\cdot\widetilde{\mathsf{pk}}_{j_1}^{c_{j_1}}\cdot\widetilde{\mathsf{pk}}_{j_2}^{c_{j_2}}=Q^{\mathsf{r}_X}\cdot\prod_{i\in[n]\setminus\{j_1,j_2\}}\widetilde{\mathsf{pk}}_i^{-c_i}\cdot\prod_{i\in[n]}\widetilde{\mathsf{pk}}_i^{c_i} \tag{24}$$

Fig. 2: DualDory and an attack on its linkability

$$\Pi_{\text{u.dd}}\left[\vec{\mathbf{pk}} \in \mathbb{G}^n, \mathsf{f} \in \mathcal{F};\ \mathsf{sk} \in \mathbb{Z}_p, j \in [n]\right]$$

$$\mathcal{P} : \mathsf{r_{cm}}, \mathsf{r}_X, c_1, ..., c_{j-1}, c_{j+1}, ..., c_n \xleftarrow{\$} \mathbb{Z}_p^* \tag{8}$$

$$\mathsf{cm} \triangleq P^{\mathsf{sk}} \cdot Q^{\mathsf{r_{cm}}} \in \mathbb{G}, \quad \widetilde{\mathsf{pk}}_i \triangleq \frac{\mathsf{cm}}{\mathsf{pk}_i}\ \forall i \in [n], \quad X \triangleq Q^{\mathsf{r}_X} \cdot \prod_{i \in [n]\setminus\{j\}} \widetilde{\mathsf{pk}}_i^{-c_i} \in \mathbb{G} \tag{9}$$

$$\mathcal{P} \Rightarrow \mathcal{V} : (\mathsf{cm}, X) \tag{10}$$

$$\mathcal{P} \Leftarrow \mathcal{V} : \widetilde{c} \xleftarrow{\$} \mathbb{Z}_p^* \tag{11}$$

$$\mathcal{P} : c_j \triangleq \widetilde{c} - \sum_{i \in [n]\setminus j} c_i, \quad \mathsf{r} \triangleq \mathsf{r}_X + \mathsf{r_{cm}} c_j \tag{12}$$

$$\mathcal{P} \Rightarrow \mathcal{V} : (c_1, ..., c_n, \mathsf{r}) \ \textit{// The proof can be compressed by Dory} \tag{13}$$

$$\mathcal{V} : \text{Check} \begin{cases} Q^{\mathsf{r}} \stackrel{?}{=} X \cdot \prod_{i \in [n]} \widetilde{\mathsf{pk}}_i^{c_i} \\ \widetilde{c} \stackrel{?}{=} \sum_{i \in [n]} c_i \end{cases} \tag{14}$$

// The verification can be compressed by Dory

$$\mathcal{V}\ \&\ \mathcal{P} : \text{Run } \Pi_{\text{tag}}[\mathsf{f}, \mathsf{cm}; \mathsf{sk}, \mathsf{r_{cm}}] \tag{15}$$

(a) Uncompressed DualDory protocol

$$\mathcal{P}' : \mathsf{r_{cm}}, \mathsf{r}_X, c_1, ..., c_{j_1-1}, c_{j_1+1}, ..., c_{j_2-1}, c_{j_2+1}, ...c_n \xleftarrow{\$} \mathbb{Z}_p^* \tag{16}$$

$$\mathsf{cm}' \triangleq P^{\mathsf{sk}'} \cdot Q^{\mathsf{r_{cm}}} \in \mathbb{G}, \quad \widetilde{\mathsf{pk}}_i \triangleq \frac{\mathsf{cm}'}{\mathsf{pk}_i}\ \forall i \in [n], \quad X' \triangleq Q^{\mathsf{r}_X} \cdot \prod_{i \in [n]\setminus\{j_1, j_2\}} \widetilde{\mathsf{pk}}_i^{-c_i} \in \mathbb{G} \tag{17}$$

$$\mathcal{P}' \Rightarrow \mathcal{V} : (\mathsf{cm}', X') \tag{18}$$

$$\mathcal{P}' \Leftarrow \mathcal{V} : \widetilde{c} \xleftarrow{\$} \mathbb{Z}_p^* \tag{19}$$

$$\mathcal{P}' : c' \triangleq \widetilde{c} - \sum_{i \in [n]\setminus\{j_1, j_2\}} c_i, \quad \mathsf{r}' \triangleq \mathsf{r}_X + \mathsf{r_{cm}} c', \quad c_{j_1} \triangleq \frac{c'(\mathsf{sk}' - \mathsf{sk}_2)}{\mathsf{sk}_1 - \mathsf{sk}_2}, \quad c_{j_2} \triangleq c' - c_{j_1} \tag{20}$$

$$\mathcal{P}' \Rightarrow \mathcal{V} : (c_1, ..., c_n, \mathsf{r}') \tag{21}$$

$$\mathcal{V} : \text{Check} \begin{cases} Q^{\mathsf{r}'} \stackrel{?}{=} X' \cdot \prod_{i \in [n]} \widetilde{\mathsf{pk}}_i^{c_i} \\ \widetilde{c} \stackrel{?}{=} \sum_{i \in [n]} c_i \end{cases} \tag{22}$$

(b) Malleability attack on DualDory with a malicious prover $\mathcal{P}'$

The ramification of our attack is that Eqn (14) is not sufficient to guarantee prefix-linkability in DualDory[1]. To mitigate such attacks, we will present a new linkable ring signature in the pairing setting in Sec 7.

5 Omniring

Before presenting our schemes, we explain the ideas of Omniring. Omniring [LRR+19] is a confidential transaction system, which contains a linkable ring signature scheme. This section refers to the Omniring linkable ring signature scheme.

We use a vector $\vec{\mathbf{c}}$ to represent the private knowledge of the known secret key in a ring. In Omniring, $\vec{\mathbf{c}}$ needs to be a unit basis vector, such that $c_i \in \{0, 1\}$ for all $i \in [n]$ and $\sum_{i \in [n]} c_i = 1$, i.e., $c_i = 1$ represents the i-th secret key in the ring is known, or 0 otherwise. Omniring adopts Bulletproofs [BBB+18] to prove $\vec{\mathbf{c}}$ as a unit basis vector (in a similar way to how Bulletproofs is applied to a range proof). Like DualDory, the prover first commits $\vec{\mathbf{c}}$ to generators $\vec{\mathbf{pk}}$, and then applies a proof-of-knowledge of a secret key in the ring. Unlike DualDory, Omniring does not suffer from the attack in Sec. 4.1, because $\vec{\mathbf{c}}$ is restricted to be a unit basis vector, despite that an attacker may know multiple secret keys in the ring. Fig. 3 presents the Omniring protocol.

[1] Although [BEHM22] contains a proof on linkability, it makes an implicit assumption that the prover commits only one zero-valued c_i in X (see [HC24] for a detailed discussion).

In the following, we outline the key ideas of Omniring protocol.

1. Checking Unit Basis Vector. First, the prover commits a secret $\vec{\mathbf{c}}$. The verifier checks if $\vec{\mathbf{c}}$ is a unit basis vector by Bulletproofs. Specifically, the prover commits $(\vec{\mathbf{c}}, \vec{\mathbf{c}}')$ as $A \triangleq \vec{\mathbf{G}}^{\vec{\mathbf{c}}} \cdot \vec{\mathbf{H}}^{\vec{\mathbf{c}}'} \cdot Q^{\mathsf{r}_A}$, where $\vec{\mathbf{c}}' \triangleq \vec{\mathbf{c}} - 1$, the bit-wise complement of $\vec{\mathbf{c}}$. The requirements that $\vec{\mathbf{c}}$ is a unit basis vector and $\vec{\mathbf{c}}'$ as its bit-wise complement can be checked by the following constraints for any given random challenge $y \xleftarrow{\$} \mathbb{Z}_p^*$ and vector $\vec{y}^n \triangleq (1, y, y^2, ..., y^{n-1})$:

$$\begin{cases} \langle \vec{\mathbf{c}},\ \vec{\mathbf{1}} \rangle \stackrel{?}{=} 1 \\ \langle \vec{\mathbf{c}} - \vec{\mathbf{1}} - \vec{\mathbf{c}}',\ \vec{y}^n \rangle \stackrel{?}{=} 0 \\ \langle \vec{\mathbf{c}},\ \vec{\mathbf{c}}' \circ \vec{y}^n \rangle \stackrel{?}{=} 0 \end{cases} \tag{25}$$

Note that Eqns (25) can be checked together by random linear combination with a given random challenge $z \xleftarrow{\$} \mathbb{Z}_p^*$:

$$z^2 \cdot \langle \vec{\mathbf{c}},\ \vec{\mathbf{1}} \rangle + z \cdot \langle \vec{\mathbf{c}} - \vec{\mathbf{1}} - \vec{\mathbf{c}}',\ \vec{y}^n \rangle + \langle \vec{\mathbf{c}},\ \vec{\mathbf{c}}' \circ \vec{y}^n \rangle \stackrel{?}{=} z^2 \tag{26}$$

where $\vec{\mathbf{1}}$ is a vector with all entries as 1. Eqn (26) can be re-expressed as an inner-product relation:

$$\left\langle \vec{\mathbf{c}} - z \cdot \vec{\mathbf{1}},\quad \vec{y}^n \circ (\vec{\mathbf{c}}' + z \cdot \vec{\mathbf{1}}) + z^2 \cdot \vec{\mathbf{1}} \right\rangle \stackrel{?}{=} \delta(y, z) \tag{27}$$

where $\delta(y, z) \triangleq z^2 + (z - z^2) \cdot \langle \vec{\mathbf{1}}, \vec{y}^n \rangle - z^3 \cdot \langle \vec{\mathbf{1}}, \vec{\mathbf{1}} \rangle$.

To add zero knowledge to Eqn (27), the prover can mask $\vec{\mathbf{c}}$ by $\vec{\mathbf{c}} + x \cdot \vec{\mathbf{s}}_1$ and $\vec{\mathbf{c}}'$ by $\vec{\mathbf{c}}' + x \cdot \vec{\mathbf{s}}_2$, where $\vec{\mathbf{s}}_1, \vec{\mathbf{s}}_2 \xleftarrow{\$} \mathbb{Z}_p^{*n}$ are random masks and $x \xleftarrow{\$} \mathbb{Z}_p^*$ is a random challenge. We define vectors $\vec{\mathbf{l}}, \vec{\mathbf{r}}$ as the masked left and right vectors on LHS of Eqn (27).

Next, the verifier can utilize Bulletproofs, a compressed argument system for checking the inner-product relation of Eqn (27). The Bulletproofs protocol $\Pi_{\text{bp.ip}}$ is presented in Fig. 3b, which is a recursive protocol for proving an inner-product relation: $Z = \vec{\mathbf{G}}^{\vec{\mathbf{l}}} \cdot \vec{\mathbf{H}}^{\vec{\mathbf{r}}} \cdot K^{\langle \vec{\mathbf{l}}, \vec{\mathbf{r}} \rangle}$, given $Z \in \mathbb{G}$ and known generators $\vec{\mathbf{G}}, \vec{\mathbf{H}} \in \mathbb{G}^n$ for some witness $\vec{\mathbf{l}}, \vec{\mathbf{r}} \in \mathbb{Z}_p^n$.

2. Knowledge of Secret Key in the Ring. Second, to prove the knowledge of the secret key of a public key in the ring, the prover reuses the commitment A (for proving $(\vec{\mathbf{c}}, \vec{\mathbf{c}}')$) and sets the generators[2] $\vec{\mathbf{G}} \triangleq \vec{\mathbf{pk}}^w \circ \vec{\mathbf{R}}$, where w is random challenge and $\vec{\mathbf{R}} \xleftarrow{\$} \mathbb{G}^n$ are selected at random. The prover must also commit $\hat{A} \triangleq \vec{\mathbf{R}}^{\vec{\mathbf{c}}} \cdot \vec{\mathbf{H}}^{\vec{\mathbf{c}}'} \cdot Q^{\hat{\mathsf{r}}_A} = R_j \cdot \vec{\mathbf{H}}^{\vec{\mathbf{c}}'} \cdot Q^{\hat{\mathsf{r}}_A}$, where $j \in [n]$ is the index of the known secret key sk in the ring such that $\mathsf{pk}_j = P^{\mathsf{sk}}$. Then, the verifier can verify the knowledge of a secret key in the ring by checking: $A \stackrel{?}{=} \mathsf{cm}^w \cdot \hat{A}$, provided that $\mathsf{r}_A = \mathsf{r}_{\mathsf{cm}} + \hat{\mathsf{r}}_A$ and $\vec{\mathbf{c}}$ is a committed unit basis vector in commitment A. The two steps of checking a unit basis vector and the knowledge of a secret key in the ring can be combined into Omniring protocol Π_{onrg} in Fig. 3a.

3. Single Multi-Exponentiation. Suppose $n = 2^m$ for some $m \in \mathbb{Z}^*$. Although Bulletproofs is a recursive protocol, we can collapse the recursions into a single multi-exponentiation at the final step [BBB+18]. In particular, we set $\vec{\mathbf{H}} \triangleq \vec{\mathbf{R}}$. As a result, in the single multi-exponentiation, there are $2n$ group exponentiations in Bulletproofs with generators $(\vec{\mathbf{pk}}, \vec{\mathbf{R}})$. The proof size of Bulletproofs includes $2 \log n$ $\mathbb{G}$ elements.

Remarks. Note that our description of Omniring protocol differs from the one in [LRR+19]. Here, we use a commitment of the secret key, rather than via a tag function combined in Bulletproofs, which is in line with the linkability schema of DualDory (see Sec. 3.3). As a result, the prover is required to use the tag proof in Fig. 1 to establish linkability.

[2] The setting of $\vec{\mathbf{G}} \triangleq \vec{\mathbf{pk}}^w \circ \vec{\mathbf{R}}$ is to prevent an attacker with the knowledge of multiple secret keys to obtain non-trivial discrete logarithmic relations, which will otherwise undermine the soundness of Bulletproofs.

Fig. 3: Omniring linkable ring signature protocol

$$\Pi_{\text{onrg}}\Big[\vec{\mathbf{pk}} \in \mathbb{G}^n, \mathsf{f} \in \mathcal{F};\ \mathsf{sk} \in \mathbb{Z}_p, j \in [n]\Big]$$

$$\mathcal{P} : \vec{\mathbf{c}} \triangleq (c_i)_{i\in[n]} \text{ where } c_i = \begin{cases} 0, \text{ if } i \neq j \\ 1, \text{ if } i = j \end{cases} \tag{28}$$

$$\vec{\mathbf{c}}' \triangleq \vec{\mathbf{c}} - \vec{\mathbf{1}}, \quad \mathsf{r}_{\mathsf{cm}}, \hat{\mathsf{r}}_A \xleftarrow{\$} \mathbb{Z}_p^* \tag{29}$$

$$\mathcal{P} \Rightarrow \mathcal{V} : \mathsf{cm} \triangleq P^{\mathsf{sk}} \cdot Q^{\mathsf{r}_{\mathsf{cm}}} \in \mathbb{G}, \quad \hat{A} \triangleq \vec{\mathbf{R}}^{\vec{\mathbf{c}}} \cdot \vec{\mathbf{H}}^{\vec{\mathbf{c}}'} \cdot Q^{\hat{\mathsf{r}}_A} \in \mathbb{G} \tag{30}$$

$$\mathcal{P} \Leftarrow \mathcal{V} : w \xleftarrow{\$} \mathbb{Z}_p^+ \tag{31}$$

$$\mathcal{V}\ \&\ \mathcal{P} : \vec{\mathbf{G}} \triangleq \vec{\mathbf{pk}}^w \circ \vec{\mathbf{R}} \in \mathbb{G}^n, \quad \vec{\mathbf{H}} \triangleq \vec{\mathbf{R}} \in \mathbb{G}^n \tag{32}$$

$$\mathcal{P} : \mathsf{r}_S \xleftarrow{\$} \mathbb{Z}_p^*, \quad \vec{\mathbf{s}}_1, \vec{\mathbf{s}}_2 \xleftarrow{\$} \mathbb{Z}_p^{*n}, \quad \mathsf{r}_A \triangleq w \cdot \mathsf{r}_{\mathsf{cm}} + \hat{\mathsf{r}}_A \in \mathbb{Z}_p \tag{33}$$

$$\mathcal{P} \Rightarrow \mathcal{V} : A \triangleq \vec{\mathbf{G}}^{\vec{\mathbf{c}}} \cdot \vec{\mathbf{H}}^{\vec{\mathbf{c}}'} \cdot Q^{\mathsf{r}_A} \in \mathbb{G}, \quad S \triangleq \vec{\mathbf{G}}^{\vec{\mathbf{s}}_1} \cdot \vec{\mathbf{H}}^{\vec{\mathbf{s}}_2} \cdot Q^{\mathsf{r}_S} \in \mathbb{G} \tag{34}$$

$$\mathcal{P} \Leftarrow \mathcal{V} : y, z \xleftarrow{\$} \mathbb{Z}_p^* \tag{35}$$

$$\mathcal{P} : t_1 \triangleq \langle \vec{\mathbf{s}}_1, \vec{y}^n \circ (\vec{\mathbf{c}}' + z \cdot \vec{\mathbf{1}}) \rangle + \langle \vec{\mathbf{s}}_1, z^2 \cdot \vec{\mathbf{1}} \rangle + \langle \vec{\mathbf{c}}, \vec{y}^n \circ \vec{\mathbf{s}}_2 \rangle - \langle z \cdot \vec{\mathbf{1}}, \vec{y}^n \circ \vec{\mathbf{s}}_2 \rangle \in \mathbb{Z}_p \tag{36}$$

$$t_2 \triangleq \langle \vec{\mathbf{s}}_1, \vec{y}^n \circ \vec{\mathbf{s}}_2 \rangle \in \mathbb{Z}_p, \quad \tau_1, \tau_2 \xleftarrow{\$} \mathbb{Z}_p^* \tag{37}$$

$$\mathcal{P} \Rightarrow \mathcal{V} : T_1 = F^{t_1} \cdot Q^{\tau_1} \in \mathbb{G}, \quad T_2 = F^{t_2} \cdot Q^{\tau_2} \in \mathbb{G} \tag{38}$$

$$\mathcal{P} \Leftarrow \mathcal{V} : x \xleftarrow{\$} \mathbb{Z}_p^* \tag{39}$$

$$\mathcal{P} : \vec{\mathbf{l}} \triangleq \vec{\mathbf{c}} + x \cdot \vec{\mathbf{s}}_1 - z \cdot \vec{\mathbf{1}} \in \mathbb{Z}_p^n, \quad \vec{y}^{-n} \triangleq \left(y^{-i+1}\right)_{i\in[n]} \in \mathbb{Z}_p^n \tag{40}$$

$$\vec{\mathbf{r}} \triangleq \vec{y}^n \circ \left(\vec{\mathbf{c}}' + x \cdot \vec{\mathbf{s}}_2 + z \cdot \vec{\mathbf{1}}\right) + z^2 \cdot \vec{\mathbf{1}} \in \mathbb{Z}_p^n \tag{41}$$

$$\mathcal{P} \Rightarrow \mathcal{V} : \hat{t} \triangleq \langle \vec{\mathbf{l}}, \vec{\mathbf{r}} \rangle \in \mathbb{Z}_p, \quad \tau_x \triangleq \tau_2 \cdot x^2 + \tau_1 \cdot x \in \mathbb{Z}_p, \quad \mathsf{r}_W \triangleq \mathsf{r}_A + \mathsf{r}_S \cdot x \in \mathbb{Z}_p, \quad W \triangleq \vec{\mathbf{G}}^{\vec{\mathbf{l}}} \cdot \left(\vec{\mathbf{H}}^{\vec{y}^{-n}}\right)^{\vec{\mathbf{r}}} \in \mathbb{G} \tag{42}$$

$$\mathcal{V} : \text{Check} \begin{cases} A \stackrel{?}{=} \mathsf{cm}^w \cdot \hat{A} \\ F^{\hat{t}} \cdot Q^{\tau_x} \stackrel{?}{=} F^{\delta(y,z)} \cdot T_1^x \cdot T_2^{x^2} \\ W \cdot Q^{\mathsf{r}_W} \stackrel{?}{=} A \cdot S^x \cdot \vec{\mathbf{G}}^{-z\cdot\vec{\mathbf{1}}} \cdot \left(\vec{\mathbf{H}}^{\vec{y}^{-n}}\right)^{z\cdot\vec{y}^n + z^2\cdot\vec{\mathbf{1}}} \end{cases} \tag{43}$$

$$\mathcal{V}\ \&\ \mathcal{P} : \text{Run } \Pi_{\text{bp.ip}}\Big[n, \vec{\mathbf{G}}, \vec{\mathbf{H}}^{\vec{y}^{-n}}, W \cdot K^{\hat{t}};\ \vec{\mathbf{l}}, \vec{\mathbf{r}}\Big] \tag{44}$$

$$\text{Run } \Pi_{\text{tag}}[\mathsf{f}, \mathsf{cm}; \mathsf{sk}, \mathsf{r}_{\mathsf{cm}}] \tag{45}$$

(a) Omniring linkable ring signature main protocol

$$\Pi_{\text{bp.ip}}\Big[n \in \mathbb{Z}^+, \vec{\mathbf{G}}, \vec{\mathbf{H}} \in \mathbb{G}^n, Z \in \mathbb{G};\ \vec{\mathbf{l}}, \vec{\mathbf{r}} \in \mathbb{Z}_p^n\Big]$$

IF $n = 1$

$$\mathcal{P} \Rightarrow \mathcal{V} : l(=\vec{\mathbf{l}}), r(=\vec{\mathbf{r}}) \in \mathbb{Z}_p \tag{46}$$

$$\mathcal{V} : \hat{t} \triangleq l \cdot r \in \mathbb{Z}_p, \quad G(=\vec{\mathbf{G}}) \in \mathbb{G}, \quad H(=\vec{\mathbf{H}}) \in \mathbb{G} \tag{47}$$

$$\text{Check } G^l \cdot H^r \cdot K^{\hat{t}} \stackrel{?}{=} Z \tag{48}$$

ELSE $n > 1$

$$\mathcal{P} : \hat{t}_1 \triangleq \langle \vec{\mathbf{l}}_R, \vec{\mathbf{r}}_L \rangle \in \mathbb{Z}_p, \quad \hat{t}_2 \triangleq \langle \vec{\mathbf{l}}_L, \vec{\mathbf{r}}_R \rangle \in \mathbb{Z}_p \tag{49}$$

$$\mathcal{P} \Rightarrow \mathcal{V} : L \triangleq \vec{\mathbf{G}}_L^{\vec{\mathbf{l}}_R} \cdot \vec{\mathbf{H}}_R^{\vec{\mathbf{r}}_L} \cdot K^{\hat{t}_1} \in \mathbb{G}, \quad R \triangleq \vec{\mathbf{G}}_R^{\vec{\mathbf{l}}_L} \cdot \vec{\mathbf{H}}_L^{\vec{\mathbf{r}}_R} \cdot K^{\hat{t}_2} \in \mathbb{G} \tag{50}$$

$$\mathcal{P} \Leftarrow \mathcal{V} : \alpha \xleftarrow{\$} \mathbb{Z}_p^* \tag{51}$$

$$\mathcal{P} : \vec{\mathbf{l}}' \triangleq \vec{\mathbf{l}}_L + \alpha \cdot \vec{\mathbf{l}}_R \in \mathbb{Z}_p^{\frac{n}{2}}, \quad \vec{\mathbf{r}}' \triangleq \alpha \cdot \vec{\mathbf{r}}_L + \vec{\mathbf{r}}_R \in \mathbb{Z}_p^{\frac{n}{2}} \tag{52}$$

$$\mathcal{V}\ \&\ \mathcal{P} : \vec{\mathbf{G}}' \triangleq \vec{\mathbf{G}}_L^{\alpha} \circ \vec{\mathbf{G}}_R \in \mathbb{G}^{\frac{n}{2}}, \quad \vec{\mathbf{H}}' \triangleq \vec{\mathbf{H}}_L \circ \vec{\mathbf{H}}_R^{\alpha} \in \mathbb{G}^{\frac{n}{2}} \tag{53}$$

$$\mathcal{V} : Z' \triangleq L^{\alpha^2} \cdot Z^{\alpha} \cdot R \in \mathbb{G} \tag{54}$$

$$\mathcal{V}\ \&\ \mathcal{P} : \text{Run } \Pi_{\text{bp.ip}}\Big[\tfrac{n}{2}, \vec{\mathbf{G}}', \vec{\mathbf{H}}', Z';\ \vec{\mathbf{l}}', \vec{\mathbf{r}}'\Big] \tag{55}$$

(b) Recursive Bulletproof protocol for checking inner-product relation: $\vec{\mathbf{G}}^{\vec{\mathbf{l}}} \cdot \vec{\mathbf{H}}^{\vec{\mathbf{r}}} \cdot K^{\hat{t}} \stackrel{?}{=} Z$, such that $\hat{t} = \langle \vec{\mathbf{l}}, \vec{\mathbf{r}} \rangle$.

6 LLRing-DL Linkable Ring Signature Scheme

In this section, we present LLRing-DL linkable ring signature scheme, which improves the verification efficiency of Omniring, reducing the number of group exponentiations to n, at the expense of doubling the proof size to $4 \log n$ $\mathbb{G}$ elements and precomputing n group exponentiations for each ring.

Particularly, we make the two following modifications to Omniring:

1. Segregated Bulletproofs. Recall that Omniring relies on Bulletproofs for proving inner-product relations, which normally takes two distinct n-vectors of generators $(\vec{\mathbf{G}}, \vec{\mathbf{H}})$, and hence, needs $2n$ group exponentiations in the single multi-exponentiation at the final step. We present a variant of Bulletproofs, which separates the inner-product relation $(\vec{\mathbf{G}}^{\vec{\mathbf{l}}} \cdot \vec{\mathbf{H}}^{\vec{\mathbf{r}}} \cdot K^{\hat{t}} \stackrel{?}{=} Z)$ into two parts and checks them separately:

$$\vec{\mathbf{G}}^{\vec{\mathbf{l}}} \stackrel{?}{=} Z_1, \quad \vec{\mathbf{H}}^{\vec{\mathbf{r}}} \cdot K^{\hat{t}} \stackrel{?}{=} Z_2 \tag{56}$$

Note that the above relations can be checked together by random liner combination:

$$(\vec{\mathbf{G}}^{\vec{\mathbf{l}}})^{\theta} \cdot \vec{\mathbf{H}}^{\vec{\mathbf{r}}} \cdot K^{\hat{t}} \stackrel{?}{=} Z_1^{\theta} \cdot Z_2 \tag{57}$$

where θ is a random number only known to the verifier.

As a result, we can set $\vec{\mathbf{G}} = \vec{\mathbf{H}}$ (because they are used to prove separate relations), and hence, n group exponentiations may be needed in the single multi-exponentiation for one n-vector generators.

We present the segregated Bulletproofs in Fig 4b, which can be applied to proving general inner-product relations with a reduced number of group exponentiations.

2. Re-defining Generators. Note that it is not sufficient to reduce the group exponentiations in the verification of Omniring by setting $\vec{\mathbf{G}} = \vec{\mathbf{H}}$ in the segregated Bulletproofs. Since $\vec{\mathbf{G}} \triangleq \vec{\mathbf{pk}}^{w} \circ \vec{\mathbf{R}}$ in Omniring, there are still $2n$ generators $(\vec{\mathbf{pk}}, \vec{\mathbf{R}})$ in the single multi-exponentiation.

Therefore, we set $\vec{\mathbf{G}} \triangleq \vec{\mathbf{pk}} \circ \vec{\mathbf{R}}$, where $\vec{\mathsf{r}} = (\mathsf{r}_i \triangleq \mathtt{Hash}[\mathsf{pk}_i])_{i\in[n]}$ and $\vec{\mathbf{R}} \triangleq \vec{R}^{\circ\vec{\mathsf{r}}} = (R^{\mathsf{r}_i})_{i\in[n]}$. This setting makes each R_i dependent on pk_i, and prevents an attacker from manipulating sk_i for a malleability attack by exploiting a known discrete logarithm relation between pk_i and R_i.

In `LLRing-DL`, the verifier first precomputes $\vec{\mathbf{R}} \triangleq (R^{\mathsf{r}_i})_{i\in[n]}$ and generators $\vec{\mathbf{G}} = \vec{\mathbf{H}} \triangleq \vec{\mathbf{pk}} \circ \vec{\mathbf{R}}$ for a given ring $\vec{\mathbf{pk}}$. Then, it applies segregated Bulletproof $\Pi_{\text{sbp.ip}}$ with only n group exponentiations in the single multi-exponentiation with generators $\vec{\mathbf{G}}$. Also, we need to separate the commitments of $\vec{\mathbf{l}}$ and $\vec{\mathbf{r}}$ in the linkable ring signature.

Note that setting $\vec{\mathbf{G}} \triangleq \vec{\mathbf{pk}}^{w} \circ \vec{\mathbf{R}}$ in Omniring and checking $A \stackrel{?}{=} \mathsf{cm}^{w} \cdot \hat{A}$ can prevent the prover from making invalid commitment $\hat{A}$. In `LLRing-DL`, the prover commits $\hat{A} \triangleq R_j \cdot Q^{\hat{\mathsf{r}}_A} = R^{\mathsf{r}_j} \cdot Q^{\hat{\mathsf{r}}_A}$. To prevent the prover from making invalid commitment $\hat{A}$, the verifier also requires an additional Schnorr proof-of-knowledge (Π_{sch} in Fig 5) for checking if some r_j is committed in $\hat{A}$ for a valid commitment.

Based on the above ideas, we develop the full protocol of `LLRing-DL` and present it in Fig 4.

`LLRing-DL` takes n group exponentiations for verification and n group exponentiations for ring-dependent but signature-independent precomputation. The proof size of segregated Bulletproofs includes $4 \log n$ $\mathbb{G}$ elements. Note that the precomputation is updatable, such that one only needs to precompute the R_i's for the new pk_i's. If a ring is shared among multiple linkable ring signatures, the precomputation can be amortized efficiently.

Theorem 26. *$\Pi_{\text{sbp.ip}}$ satisfies computationally witness-extended emulation (CWE) or it can obtain a non-trivial discrete logarithm relation.*

Theorem 27. `LLRing-DL` *satisfies perfect completeness, anonymity, unforgeability, linkability, non-slanderability.*

The proofs can be found in Appendix 11.

Remarks. Segregated Bulletproofs $\Pi_{\text{sbp.ip}}$ can be applied to other proofs based on inner-product relations (e.g., range proofs). For a range proof, it achieves n group exponentiations in verification (i.e., 50% reduction in Bulletproofs range proof [BBB+18]) with a double proof size of Bulletproofs range proof. Note that Swiftange [WCL24] recently proposes to use a quadratic compressed Σ-protocol for range proofs with 50% reduction in group exponentiations and doubling the proof size of Bulletproofs range proof. Segregated Bulletproofs achieves the same benefits of Swiftrange without using quadratic compressed Σ-protocol.

Fig. 4: LLRing-DL linkable ring signature protocol

$$\Pi_{\text{mlr.dl}}\Big[\vec{\mathbf{pk}} \in \mathbb{G}^n, \mathsf{f} \in \mathcal{F};\ \mathsf{sk} \in \mathbb{Z}_p, j \in [n]\Big]$$

$$\mathcal{V}\ \&\ \mathcal{P}: \text{Pre-Compute } (\mathsf{r}_i \triangleq \mathtt{Hash}[\mathsf{pk}_i])_{i\in[n]}, \vec{\mathbf{R}} \triangleq \vec{R}^{\circ\vec{\mathsf{r}}}, \vec{\mathbf{G}} = \vec{\mathbf{H}} \triangleq \vec{\mathbf{pk}} \circ \vec{\mathbf{R}}$$

$$\mathcal{P}: \vec{\mathbf{c}} \triangleq (c_i)_{i\in[n]} \text{ where } c_i = \begin{cases} 0, \text{ if } i \neq j \\ 1, \text{ if } i = j \end{cases} \tag{58}$$

$$\vec{\mathbf{c}}' \triangleq \vec{\mathbf{c}} - \vec{\mathbf{1}}, \quad \mathsf{r}_{\mathsf{cm}}, \hat{\mathsf{r}}_A \xleftarrow{\$} \mathbb{Z}_p^* \tag{59}$$

$$\mathcal{P} \Rightarrow \mathcal{V}: \mathsf{cm} \triangleq P^{\mathsf{sk}} \cdot Q^{\mathsf{r}_{\mathsf{cm}}} \in \mathbb{G}, \quad \hat{A} \triangleq R_j \cdot Q^{\hat{\mathsf{r}}_A} = R'^{j} \cdot Q^{\hat{\mathsf{r}}_A} \in \mathbb{G} \tag{60}$$

$$\mathcal{V}\ \&\ \mathcal{P}: \text{Run } \Pi_{\text{sch}}[\hat{A}, R; \mathbf{r}_j, \hat{\mathsf{r}}_A] \tag{61}$$

$$\mathcal{P}: \mathsf{r}_B, \mathsf{r}_{S_1}, \mathsf{r}_{S_2} \xleftarrow{\$} \mathbb{Z}_p^*, \quad \vec{\mathbf{s}}_1, \vec{\mathbf{s}}_2 \xleftarrow{\$} \mathbb{Z}_p^{*n}, \quad \mathsf{r}_A \triangleq \mathsf{r}_{\mathsf{cm}} + \hat{\mathsf{r}}_A \in \mathbb{Z}_p \tag{62}$$

$$\mathcal{P} \Rightarrow \mathcal{V}: A \triangleq \vec{\mathbf{G}}^{\vec{\mathbf{c}}} \cdot Q^{\mathsf{r}_A} \in \mathbb{G}, \quad S_1 \triangleq \vec{\mathbf{G}}^{\vec{\mathbf{s}}_1} \cdot Q^{\mathsf{r}_{S_1}} \in \mathbb{G}, \quad B \triangleq \vec{\mathbf{H}}^{\vec{\mathbf{c}}'} \cdot Q^{\mathsf{r}_B} \in \mathbb{G}, \quad S_2 \triangleq \vec{\mathbf{H}}^{\vec{\mathbf{s}}_2} \cdot Q^{\mathsf{r}_{S_2}} \in \mathbb{G} \tag{63}$$

$$\mathcal{P} \Leftarrow \mathcal{V}: y, z \xleftarrow{\$} \mathbb{Z}_p^* \tag{64}$$

$$\mathcal{P}: t_1 \triangleq \langle \vec{\mathbf{s}}_1, \vec{y}^n \circ (\vec{\mathbf{c}}' + z \cdot \vec{\mathbf{1}}) \rangle + \langle \vec{\mathbf{s}}_1, z^2 \cdot \vec{\mathbf{1}} \rangle + \langle \vec{\mathbf{c}}, \vec{y}^n \circ \vec{\mathbf{s}}_2 \rangle - \langle z \cdot \vec{\mathbf{1}}, \vec{y}^n \circ \vec{\mathbf{s}}_2 \rangle \in \mathbb{Z}_p \tag{65}$$

$$t_2 \triangleq \langle \vec{\mathbf{s}}_1, \vec{y}^n \circ \vec{\mathbf{s}}_2 \rangle \in \mathbb{Z}_p, \quad \tau_1, \tau_2 \xleftarrow{\$} \mathbb{Z}_p^* \tag{66}$$

$$\mathcal{P} \Rightarrow \mathcal{V}: T_1 = F^{t_1} \cdot Q^{\tau_1} \in \mathbb{G}, \quad T_2 = F^{t_2} \cdot Q^{\tau_2} \in \mathbb{G} \tag{67}$$

$$\mathcal{P} \Leftarrow \mathcal{V}: x \xleftarrow{\$} \mathbb{Z}_p^* \tag{68}$$

$$\mathcal{P}: \vec{\mathbf{l}} \triangleq \vec{\mathbf{c}} + x \cdot \vec{\mathbf{s}}_1 - z \cdot \vec{\mathbf{1}} \in \mathbb{Z}_p^n, \quad \vec{y}^{-n} \triangleq \left(y^{-i+1}\right)_{i\in[n]} \in \mathbb{Z}_p^n \tag{69}$$

$$\vec{\mathbf{r}} \triangleq \vec{y}^n \circ \left(\vec{\mathbf{c}}' + x \cdot \vec{\mathbf{s}}_2 + z \cdot \vec{\mathbf{1}}\right) + z^2 \cdot \vec{\mathbf{1}} \in \mathbb{Z}_p^n \tag{70}$$

$$\mathcal{P} \Rightarrow \mathcal{V}: \hat{t} \triangleq \langle \vec{\mathbf{l}}, \vec{\mathbf{r}} \rangle \in \mathbb{Z}_p, \quad \tau_x \triangleq \tau_2 \cdot x^2 + \tau_1 \cdot x \in \mathbb{Z}_p, \quad \mathsf{r}_{W_1} \triangleq \mathsf{r}_A + \mathsf{r}_{S_1} \cdot x \in \mathbb{Z}_p, \quad \mathsf{r}_{W_2} \triangleq \mathsf{r}_B + \mathsf{r}_{S_2} \cdot x \in \mathbb{Z}_p \tag{71}$$

$$W_1 \triangleq \vec{\mathbf{G}}^{\vec{\mathbf{l}}} \in \mathbb{G}, \quad W_2 \triangleq \left(\vec{\mathbf{H}}^{\vec{y}^{-n}}\right)^{\vec{\mathbf{r}}} \in \mathbb{G} \tag{72}$$

$$\mathcal{V}: \text{Check} \begin{cases} A \stackrel{?}{=} \mathsf{cm} \cdot \hat{A} \\ F^{\hat{t}} \cdot Q^{\tau_x} \stackrel{?}{=} F^{\delta(y,z)} \cdot T_1^x \cdot T_2^{x^2} \\ W_1 \cdot Q^{\mathsf{r}_{W_1}} \stackrel{?}{=} A \cdot S_1^x \cdot \vec{\mathbf{G}}^{-z\cdot\vec{\mathbf{1}}} \\ W_2 \cdot Q^{\mathsf{r}_{W_2}} \stackrel{?}{=} B \cdot S_2^x \cdot \left(\vec{\mathbf{H}}^{\vec{y}^{-n}}\right)^{z\cdot\vec{y}^n + z^2\cdot\vec{\mathbf{1}}} \end{cases} \tag{73}$$

$$\mathcal{V}\ \&\ \mathcal{P}: \text{Run } \Pi_{\text{sbp.ip}}\Big[n, \vec{\mathbf{G}}, \vec{\mathbf{H}}^{\vec{y}^{-n}},\ W_1,\ W_2 \cdot K^{\hat{t}};\ \vec{\mathbf{l}}, \vec{\mathbf{r}}\Big] \tag{74}$$

// Call segregated Bulletproofs protocol for inner-product relation proof

$$\text{Run } \Pi_{\text{tag}}[\mathsf{f}, \mathsf{cm}; \mathsf{sk}, \mathsf{r}_{\mathsf{cm}}] \tag{75}$$

(a) LLRing-DL linkable ring signature main protocol

$$\Pi_{\text{sbp.ip}}\Big[n \in \mathbb{Z}^+, \vec{\mathbf{G}}, \vec{\mathbf{H}} \in \mathbb{G}^n, Z_1 \in \mathbb{G}, Z_2 \in \mathbb{G};\ \vec{\mathbf{l}}, \vec{\mathbf{r}} \in \mathbb{Z}_p^n\Big]$$

IF $n = 1$

$$\mathcal{P} \Rightarrow \mathcal{V}: l(=\vec{\mathbf{l}}), r(=\vec{\mathbf{r}}) \in \mathbb{Z}_p \tag{76}$$

$$\mathcal{V}: \hat{t} \triangleq l \cdot r \in \mathbb{Z}_p, \quad G(=\vec{\mathbf{G}}) \in \mathbb{G}, \quad H(=\vec{\mathbf{H}}) \in \mathbb{G}, \quad \theta \xleftarrow{\$} \mathbb{Z}_p^* \tag{77}$$

$$\text{Check } G^{l\cdot\theta} \cdot H^r \cdot K^{\hat{t}} \stackrel{?}{=} Z_1^{\theta} \cdot Z_2 \tag{78}$$

ELSE $n > 1$

$$\mathcal{P}: \hat{t}_1 \triangleq \langle \vec{\mathbf{l}}_R, \vec{\mathbf{r}}_L \rangle \in \mathbb{Z}_p, \quad \hat{t}_2 \triangleq \langle \vec{\mathbf{l}}_L, \vec{\mathbf{r}}_R \rangle \in \mathbb{Z}_p \tag{79}$$

$$\mathcal{P} \Rightarrow \mathcal{V}: L_1 \triangleq \vec{\mathbf{G}}_L^{\vec{\mathbf{l}}_R} \in \mathbb{G}, \quad L_2 \triangleq \vec{\mathbf{H}}_R^{\vec{\mathbf{r}}_L} \cdot K^{\hat{t}_1} \in \mathbb{G}, \quad R_1 \triangleq \vec{\mathbf{G}}_R^{\vec{\mathbf{l}}_L} \in \mathbb{G}, \quad R_2 \triangleq \vec{\mathbf{H}}_L^{\vec{\mathbf{r}}_R} \cdot K^{\hat{t}_2} \in \mathbb{G} \tag{80}$$

$$\mathcal{P} \Leftarrow \mathcal{V}: \alpha \xleftarrow{\$} \mathbb{Z}_p^* \tag{81}$$

$$\mathcal{P}: \vec{\mathbf{l}}' \triangleq \vec{\mathbf{l}}_L + \alpha \cdot \vec{\mathbf{l}}_R \in \mathbb{Z}_p^{\frac{n}{2}}, \ \vec{\mathbf{r}}' \triangleq \alpha \cdot \vec{\mathbf{r}}_L + \vec{\mathbf{r}}_R \in \mathbb{Z}_p^{\frac{n}{2}} \tag{82}$$

$$\mathcal{V}\ \&\ \mathcal{P}: \vec{\mathbf{G}}' \triangleq \vec{\mathbf{G}}_L^{\alpha} \circ \vec{\mathbf{G}}_R \in \mathbb{G}^{\frac{n}{2}}, \ \vec{\mathbf{H}}' \triangleq \vec{\mathbf{H}}_L \circ \vec{\mathbf{H}}_R^{\alpha} \in \mathbb{G}^{\frac{n}{2}} \tag{83}$$

$$\mathcal{V}: Z_1' \triangleq L_1^{\alpha^2} \cdot Z_1^{\alpha} \cdot R_1 \in \mathbb{G}, \ Z_2' \triangleq L_2^{\alpha^2} \cdot Z_2^{\alpha} \cdot R_2 \in \mathbb{G} \tag{84}$$

$$\mathcal{V}\ \&\ \mathcal{P}: \text{Run } \Pi_{\text{sbp.ip}}\Big[\tfrac{n}{2}, \vec{\mathbf{G}}', \vec{\mathbf{H}}', Z_1', Z_2';\ \vec{\mathbf{l}}', \vec{\mathbf{r}}'\Big] \tag{85}$$

(b) Segregated Bulletproofs protocol for checking inner-product relation: $\vec{\mathbf{G}}^{\vec{\mathbf{l}}} \stackrel{?}{=} Z_1$ and $\vec{\mathbf{H}}^{\vec{\mathbf{r}}} \cdot K^{\hat{t}} \stackrel{?}{=} Z_2$, such that $\hat{t} = \langle \vec{\mathbf{l}}, \vec{\mathbf{r}} \rangle$.

Fig. 5: Schnorr proof-of-knowledge for checking if some m is committed in cm

$$\Pi_{\text{sch}}\left[\mathsf{cm} \in \mathbb{G}, R \in \mathbb{G};\ \mathsf{m} \in \mathbb{Z}_p, \mathsf{r}_{\mathsf{cm}} \in \mathbb{Z}_p\right]$$

$$\mathcal{P} : \mathsf{r}_C \xleftarrow{\$} \mathbb{Z}_p^*, \quad \mathsf{c} \xleftarrow{\$} \mathbb{Z}_p \tag{86}$$

$$\mathcal{P} \Rightarrow \mathcal{V} : C \triangleq \mathtt{Cm}^R[\mathsf{c}; \mathsf{r}_C] \tag{87}$$

$$\mathcal{P} \Leftarrow \mathcal{V} : \rho \xleftarrow{\$} \mathbb{Z}_p^* \tag{88}$$

$$\mathcal{P} \Rightarrow \mathcal{V} : \mathsf{c}' \triangleq \mathsf{c} + \rho \cdot \mathsf{m} \in \mathbb{Z}_p,\ \mathsf{r}' \triangleq \mathsf{r}_C + \rho \cdot \mathsf{r}_{\mathsf{cm}} \in \mathbb{Z}_p \tag{89}$$

$$\mathcal{V} : \text{Check } \mathtt{Cm}^R[\mathsf{c}'; \mathsf{r}'] \stackrel{?}{=} C \cdot \mathsf{cm}^\rho \tag{90}$$

7 LLRing-P Linkable Ring Signature Scheme

To eliminate the attacks on the linkability of DualDory, this section presents LLRing-P linkable ring signature scheme with logarithmic verifiability in the pairing setting. We base on the unit basis vector approach in Omniring by replacing Bulletproofs by Dory [Lee21], and Pedersen commitment by AFGHO commitment.

However, there are some obstacles in adapting the Bulletproofs approach to Dory. Bulletproofs checks the conformity of a unit basis vector by $\vec{\mathbf{G}}^{\vec{\mathbf{l}}} \cdot (\vec{\mathbf{H}}^{\vec{y}^{-n}})^{\vec{\mathbf{r}}} \cdot K^{\langle \vec{\mathbf{l}}, \vec{\mathbf{r}} \rangle} \stackrel{?}{=} Z$. But $\vec{\mathbf{H}}^{\vec{y}^{-n}}$ depend on verifier-supplied challenge y. But Dory relies on precomputation with only pre-defined signature-independent generators. This cannot be adopted in Dory.

Hence, we design LLRing-P to overcome these obstacles, without adopting the Bulletproofs approach. We outline the key ideas of LLRing-P as follows.

1. Checking Unit Basis Vector. First, the verifier can check if $\vec{\mathbf{c}}$ is a unit basis vector (which represents the knowledge of the known secret keys in the ring) by the following equations:

$$\begin{cases} \langle \vec{\mathbf{c}},\ \vec{\mathbf{1}} \rangle \stackrel{?}{=} 1 \\ \vec{\mathbf{c}} \circ (\vec{\mathbf{c}} - \vec{\mathbf{1}}) \stackrel{?}{=} \vec{\mathbf{0}} \end{cases} \tag{91}$$

Note that Eqns (91) are equivalent to the following equations via bilinear pairing:

$$\begin{cases} \mathsf{e}(\vec{L}', \vec{L})^{\vec{\mathbf{c}}} = \prod_{i \in [n]} \mathsf{e}(L_i', L_i)^{c_i} \stackrel{?}{=} \mathsf{e}(L', L), \\ \mathsf{e}(\vec{\mathbf{G}}, \vec{\mathbf{H}})^{\vec{\mathbf{c}} \circ \vec{\mathbf{c}}} = \prod_{i \in [n]} \mathsf{e}(G_i, H_i)^{c_i \cdot c_i} \stackrel{?}{=} \prod_{i \in [n]} \mathsf{e}(G_i, H_i)^{c_i} = \mathsf{e}(\vec{\mathbf{G}}, \vec{\mathbf{H}})^{\vec{\mathbf{c}}} \end{cases} \tag{92}$$

where $\vec{\mathbf{G}} \xleftarrow{\$} \mathbf{G}_1^n$, $\vec{\mathbf{H}} \xleftarrow{\$} \mathbf{G}_2^n$, $L' \xleftarrow{\$} \mathbf{G}_1$, $L \xleftarrow{\$} \mathbf{G}_2$, and $\vec{L}' \triangleq (L', ..., L') \in \mathbf{G}_1^n$, $\vec{L} \triangleq (L, ..., L) \in \mathbf{G}_2^n$.

2. Knowledge of Secret Key in the Ring. Suppose $\mathsf{r}_i \triangleq \mathtt{Hash}[\mathsf{pk}_i]$ and $R_i \triangleq R^{\mathsf{r}_i}$. To prove the knowledge of $\mathsf{pk}_j = P^{\mathsf{sk}}$, the prover first commits via AFGHO commitments: $\mathsf{cm} \triangleq \mathsf{e}(P, L)^{\mathsf{sk}} \cdot \mathtt{Q}^{\mathsf{r}_{\text{cm}}}$ and $\hat{A} \triangleq \mathsf{e}(R, L)^{\mathsf{r}_j} \cdot \mathtt{Q}^{\hat{\mathsf{r}}_A}$. If $\vec{\mathbf{c}}$ is a unit basis vector, then the verifier can check the knowledge of a secret key in the ring by checking the following:

$$\mathsf{cm} \cdot \hat{A} = \mathsf{e}(P^{\mathsf{sk}} \cdot R^{\mathsf{r}_j}, L) \cdot \mathtt{Q}^{\mathsf{r}_{\text{cm}} + \hat{\mathsf{r}}_A} \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{pk}} \circ \vec{\mathbf{R}}, \vec{L})^{\vec{\mathbf{c}}} \cdot \mathtt{Q}^{\mathsf{r}} \tag{93}$$

where $\mathsf{r} \triangleq \mathsf{r}_{\text{cm}} + \hat{\mathsf{r}}_A$ is provided by the prover.

3. Compression by Dory. Next, we combine the two steps of checking a unit basis vector and the knowledge of a secret key in the ring into one single protocol. Moreover, we will compress the protocol by Dory to achieve logarithmic verifiability.

Dory [Lee21] is a compressed zero-knowledge arguments of knowledge protocol with logarithmic verification efficiency and proof size for checking the satisfiability of the following set of inner-product relations:

$$\begin{cases} D_0 \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{\Omega}}, \vec{\mathbf{\Theta}}) \cdot \mathtt{Q}^{\mathsf{r}_0} \\ D_1 \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{\Omega}}, \vec{\mathbf{\Lambda}}) \cdot \mathtt{Q}^{\mathsf{r}_1} \\ D_2 \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{\Gamma}}, \vec{\mathbf{\Theta}}) \cdot \mathtt{Q}^{\mathsf{r}_2} \end{cases} \tag{94}$$

Fig. 6: LLRing-P linkable ring signature protocol

$$\Pi_{\text{mlr.p}}\Big[\vec{\mathbf{pk}} \in \mathbb{G}^n, \mathsf{f} \in \mathcal{F};\ \mathsf{sk} \in \mathbb{Z}_p, j \in [n]\Big]$$

$$\mathcal{V}\ \&\ \mathcal{P}: \text{Pre-Compute } \vec{\mathsf{r}} = (\mathsf{r}_i \triangleq \mathtt{Hash}[\mathsf{pk}_i])_{i\in[n]},\quad \vec{\mathbf{R}} \triangleq \vec{R}^{\circ\vec{\mathsf{r}}},\quad \vec{\mathbf{G}} \triangleq \vec{\mathbf{pk}} \circ \vec{\mathbf{R}},\quad \mathsf{e}(\vec{\mathbf{G}}, \vec{L}),\ \mathsf{e}(\vec{L'}, \vec{\mathbf{H}}),\ \mathsf{e}(L', L) \in \mathbb{G}_T \tag{98}$$

$$\mathcal{P}: \vec{\mathbf{c}} \triangleq (c_i)_{i\in[n]} \text{ where } c_i = \begin{cases} 0, \text{ if } i \neq j \\ 1, \text{ if } i = j \end{cases} \tag{99}$$

$$\mathsf{r}_{\mathsf{cm}}, \hat{\mathsf{r}}_A, \mathsf{r}_B \xleftarrow{\$} \mathbb{Z}_p^*,\quad \mathsf{r}_A \triangleq \mathsf{r}_{\mathsf{cm}} + \hat{\mathsf{r}}_A \tag{100}$$

$$\mathcal{P} \Rightarrow \mathcal{V}: \mathsf{cm} \triangleq \mathsf{e}(P, L)^{\mathsf{sk}} \cdot \mathtt{Q}^{\mathsf{r}_{\mathsf{cm}}} \in \mathbb{G}_T,\ \hat{A} \triangleq \mathsf{e}(R, L)^{\mathsf{r}_j} \cdot \mathtt{Q}^{\hat{\mathsf{r}}_A} \in \mathbb{G}_T,\quad B \triangleq \mathsf{e}(\vec{\mathbf{G}}, \vec{\mathbf{H}})^{\vec{\mathbf{c}}} \cdot \mathtt{Q}^{\mathsf{r}_B} \in \mathbb{G}_T \tag{101}$$

$$\mathcal{V}\ \&\ \mathcal{P}: \text{Run } \Pi_{\text{sch}}[\hat{A}, R; \mathbf{r}_j, \hat{\mathsf{r}}_A] \tag{102}$$

$$\mathcal{V}: A \triangleq \mathsf{cm} \cdot \hat{A} \tag{103}$$

$$\mathcal{V}\ \&\ \mathcal{P}: \text{Run } \Pi_{\text{do.ip}}\Big[n, \vec{\mathbf{G}}, \vec{\mathbf{H}}, \mathsf{e}(L', L), \mathsf{e}(\vec{L'}, \vec{\mathbf{H}}), A; \vec{L'}, \vec{L}^{\circ\vec{\mathbf{c}}}, 0, 0, \mathsf{r}_A\Big] \tag{104}$$

$$\text{Run } \Pi_{\text{do.ip}}\Big[n, \vec{\mathbf{G}}, \vec{\mathbf{H}}, B, B, B; \vec{\mathbf{G}}^{\circ\vec{\mathbf{c}}}, \vec{\mathbf{H}}^{\circ\vec{\mathbf{c}}}, \mathsf{r}_B, \mathsf{r}_B, \mathsf{r}_B\Big] \tag{105}$$

$$\text{Run } \Pi_{\text{do.ip}}\Big[n, \vec{\mathbf{G}}, \vec{\mathbf{H}}, A, B, \mathsf{e}(\vec{\mathbf{G}}, \vec{L}); \vec{\mathbf{G}}^{\circ\vec{\mathbf{c}}}, \vec{L}, \mathsf{r}_A, \mathsf{r}_B, 0\Big] \tag{106}$$

// The above 3 Dory checks can be batched into a single Dory check

$$\text{Run } \Pi_{\text{tag}}[\mathsf{f}, \mathsf{cm}; \mathsf{sk}, \mathsf{r}_{\mathsf{cm}}] \tag{107}$$

where $D_0, D_1, D_2 \in \mathbb{G}_T$ are given commitments and $\vec{\mathbf{\Gamma}} \xleftarrow{\$} \mathbb{G}_1^n, \vec{\mathbf{\Lambda}} \xleftarrow{\$} \mathbb{G}_2^n$ are known random generators, and ($\vec{\mathbf{\Omega}} \in \mathbb{G}_1^n, \vec{\mathbf{\Theta}} \in \mathbb{G}_2^n,\ \mathsf{r}_0, \mathsf{r}_1, \mathsf{r}_2 \in \mathbb{Z}_p^*$) are the witnesses. See Appendix 10 for a more detailed description of recursive Dory protocol.

As in LLRing-DL, we set $\vec{\mathbf{G}} \triangleq \vec{\mathbf{pk}} \circ \vec{\mathbf{R}}$ and $A \triangleq \mathsf{cm} \cdot \hat{A}$. To apply Dory, we incorporate the checking of Eqns (92) and (93) into the checking of the following three sets of inner-product relations:

1. The first equation in Eqns (92) can be incorporated into the checking of the following set of inner-product relations with given commitment A:

$$\begin{cases} D'_0 \triangleq \mathsf{e}(L', L) \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{\Omega}}', \vec{\mathbf{\Theta}}') \cdot \mathtt{Q}^{\mathsf{r}'_0} \\ D'_1 \triangleq \mathsf{e}(\vec{L'}, \vec{\mathbf{H}}) \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{\Omega}}', \vec{\mathbf{H}}) \cdot \mathtt{Q}^{\mathsf{r}'_1} \\ D'_2 \triangleq A \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{G}}, \vec{\mathbf{\Theta}}') \cdot \mathtt{Q}^{\mathsf{r}'_2} \end{cases} \tag{95}$$

   given commitments (D'_0, D'_1, D'_2) and generators $(\vec{\mathbf{G}}, \vec{\mathbf{H}})$, and the witnesses are $(\vec{\mathbf{\Omega}}' = \vec{L'}, \vec{\mathbf{\Theta}}' = \vec{L}^{\circ\vec{\mathbf{c}}}, \mathsf{r}'_0 = \mathsf{r}'_1 = 0, \mathsf{r}'_2 = \mathsf{r}_A)$.
2. The second equation in Eqns (92) can be incorporated into the checking of the following set of inner-product relations with given commitment $B \triangleq \mathsf{e}(\vec{\mathbf{G}}, \vec{\mathbf{H}})^{\vec{\mathbf{c}}} \cdot \mathtt{Q}^{\mathsf{r}_B}$ that is provided by the prover:

$$\begin{cases} D''_0 \triangleq B \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{\Omega}}'', \vec{\mathbf{\Theta}}'') \cdot \mathtt{Q}^{\mathsf{r}''_0} \\ D''_1 \triangleq B \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{\Omega}}'', \vec{\mathbf{H}}) \cdot \mathtt{Q}^{\mathsf{r}''_1} \\ D''_2 \triangleq B \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{G}}, \vec{\mathbf{\Theta}}'') \cdot \mathtt{Q}^{\mathsf{r}''_2} \end{cases} \tag{96}$$

   given commitments (D''_0, D''_1, D''_2) and generators $(\vec{\mathbf{G}}, \vec{\mathbf{H}})$, and the witnesses are $(\vec{\mathbf{\Omega}}'' = \vec{\mathbf{G}}^{\circ\vec{\mathbf{c}}}, \vec{\mathbf{\Theta}}'' = \vec{\mathbf{H}}^{\circ\vec{\mathbf{c}}}, \mathsf{r}''_0 = \mathsf{r}''_1 = \mathsf{r}''_2 = \mathsf{r}_B)$.
3. Finally, to ensure that the same $\vec{\mathbf{c}}$ is involved in both commitments A and B, the verifier checks the following set of inner-product relations:

$$\begin{cases} D'''_0 \triangleq A \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{\Omega}}''', \vec{\mathbf{\Theta}}''') \cdot \mathtt{Q}^{\mathsf{r}'''_0} \\ D'''_1 \triangleq B \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{\Omega}}''', \vec{\mathbf{H}}) \cdot \mathtt{Q}^{\mathsf{r}'''_1} \\ D'''_2 \triangleq \mathsf{e}(\vec{\mathbf{G}}, \vec{L}) \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{G}}, \vec{\mathbf{\Theta}}''') \cdot \mathtt{Q}^{\mathsf{r}'''_2} \end{cases} \tag{97}$$

   given commitments (D'''_0, D'''_1, D'''_2) and generators $(\vec{\mathbf{G}}, \vec{\mathbf{H}})$, and the witnesses are $(\vec{\mathbf{\Omega}}''' = \vec{\mathbf{G}}^{\circ\vec{\mathbf{c}}}, \vec{\mathbf{\Theta}}''' = \vec{L}, \mathsf{r}'''_0 = \mathsf{r}_A, \mathsf{r}'''_1 = \mathsf{r}_B, \mathsf{r}'''_2 = 0)$.

At last, we then can apply Dory to check Eqns. (95)-(97) in LLRing-P. Note that $\mathsf{e}(\vec{\mathbf{G}}, \vec{L}), \mathsf{e}(L', L), \mathsf{e}(\vec{L'}, \vec{\mathbf{H}})$ can be precomputed, while cm and $\hat{A}$ are provided by the prover. Based on the above ideas, we develop the protocol of LLRing-P and present it in Fig 6.

Note that Dory supports batching the proofs over multiple sets of inner-product relations, as long as they share the same common known generators. Therefore, we can batch the checking of the three sets of inner-product relations (Eqns. (95)-(97)) into one set of inner-product relations by a single Dory check. See batching of Dory in Appendix 10. As a result of batching, `LLRing-P` takes $10 \log n$ $\mathbb{G}_T$ exponentiations + 1 pairing for verification, and n $\mathbb{G}_1$ exponentiations + $3n$ pairings for ring-dependent precomputation. The proof size includes $6 \log n$ $\mathbb{G}_T$ elements. The precomputation is updatable, and can be amortized among multiple signatures with the same ring. We omit the ring-independent precomputation in Dory that does not involve $\vec{\mathbf{pk}}$, because it can be precomputed once at setup.

Theorem 28. `LLRing-P` *satisfies perfect completeness, anonymity, unforgeability, linkability, non-slanderability.*

The proof can be found in Appendix 12.

Remarks. Our technique of applying Dory to check Eqns (91) also applies to the checking of the bit decomposition of a number. Hence, this also applies to range proofs and enables logarithmically verifiable range proofs, which may be of independent interest.

8 Evaluation

In this section, we empirically evaluate our schemes and compare them with Omniring in concrete implementation. We implemented our schemes in Java. We evaluated its performance by conducting 100 independent trials on a Windows machine equipped with a 13th Gen Intel(R) Core(TM) i5-13400F, operating at 2500 MHz, with 10 cores, 16 logical processors, and 32GB of RAM. For elliptic curve-related operations, we employ the Java Pairing Based Cryptography (JPBC) library [DCI11] with Type F pairing curve. We explain the evaluation results as follows.

- **Verification Time**. To better illustrate the superior performance of `LLRing` schemes, we replicated Omniring in the same environment. The comparison of verification cost performance between Omniring, `LLRing-DL`, and `LLRing-P` is shown in Fig 7. It is demonstrated that while both Omniring and `LLRing-DL` exhibit linear verification times, whereas `LLRing-P` achieves logarithmic verification time. Notably, `LLRing-DL` improves the verification time considerably compared to Omniring. Moreover, as the ring size increases beyond 128, the advantage of `LLRing-P` becomes increasingly prominent.
- **Ring-Dependent Precomputation Time**. Fig 8 shows the precomputation time of `LLRing` for each ring. Since the proving time of `LLRing-P` is not on the same order of magnitude as `LLRing-DL`, we have adopted a log scale for both the x-axis and y-axis. A comparison reveals that the precomputation time required for `LLRing-P` is approximately 73 times that of `LLRing-DL`.
- **Proof Size**. In Fig 9, the proof sizes for Omniring, `LLRing-DL`, and `LLRing-P` are compared across ring sizes ranging from 8 to 512. All schemes show a logarithmic increase in proof size as expected. Omniring's proof size increases from 1056 to 1824 bytes, `LLRing-DL` ranges from 1600 to 3136 bytes, and `LLRing-P` exhibits a significant increase from 11104 to 24928 bytes, indicating varying degrees of scalability among the schemes.
- **Proving Time**. The proving times for different schemes, as shown in Fig 10, where we also applied a log scale to both the x-axis and y-axis for comparison. It is observed that the proving time of Omniring and `LLRing-DL` are quite similar, with `LLRing-DL` being about 12.9% less than Omniring. However, the proving time of `LLRing-P` is approximately 20 times that of `LLRing-DL`.

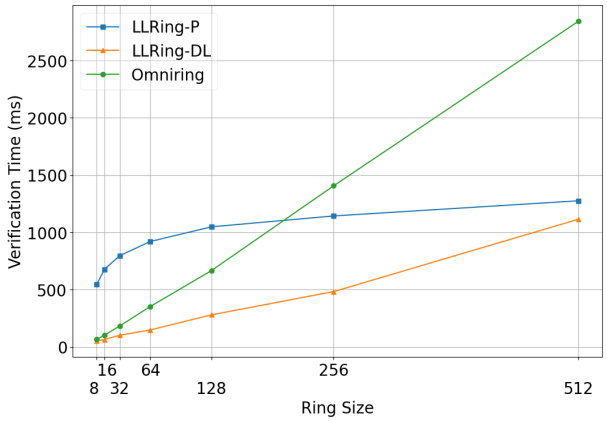

Fig. 7: Verification time comparison

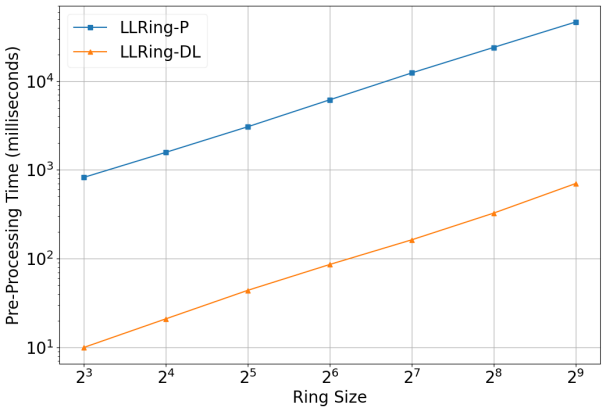

Fig. 8: Ring-dependent precomputation time comparison

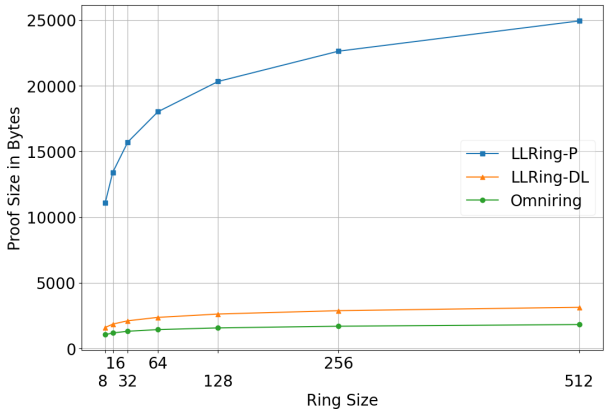

Fig. 9: Proof size comparison

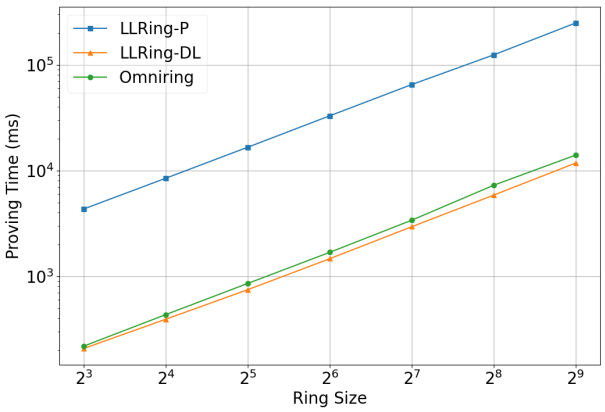

Fig. 10: Proving time comparison

9 Conclusion

To advance efficient linkable ring signatures, we make three novel contributions in this paper, including a discovery of an attack on the linkability of DualDory that breaks its linkability, a rectified linkable ring signature scheme in the pairing setting with logarithmic verifiability (`LLRing-P`), and an improved linkable ring signature scheme in the discrete logarithm setting (`LLRing-DL`) with 50% reduction in group exponentiations for verification, as compared with Omniring. We implemented our schemes, which provide competitive performance in concrete implementation to enable diverse anonymized applications, such as e-voting and confidential transactions. In future work, we will extend our work to other types of ring signature schemes, like traceable ring signature schemes. We will also implement our ring signature schemes on real-world blockchain systems for trustless decentralized confidential transaction applications.

References

ACF21. Thomas Attema, Ronald Cramer, and Serge Fehr. Compressing proofs of k-out-of-n partial knowledge. In *CRYPTO*, pages 65–91, 2021.

ACJT00. Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *CRYPTO*, pages 255–270, 2000.

AOS02. Masayuki Abe, Miyako Ohkubo, and Koutarou Suzuki. 1-out-of-n signatures from a variety of key. In *ASIACRYPT*, 2002.

BBB+18. Benedikt Bunz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *IEEE SP*, pages 315–334, 05 2018.

BBS04. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *CRYPTO*, 2004.

BCC+16. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *EUROCRYPT*, 2016.

BDH+19. Michael Backes, Nico Doettling, Lucjan Hanzlik, Kamil Kluczniak, and Jonas Schneider. Ring signatures: Logarithmic-size, no setup—from standard assumptions. In *EUROCRYPT*, 2019.

BEHM22. Jonathan Bootle, Kaoutar Elkhiyaoui, Julia Hesse, and Yacov Manevich. Dualdory: Logarithmic-verifier linkable ring signatures through preprocessing. In *ESORICS*, page 427–446, 2022.

BsCG+14. Eli Ben-sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *IEEE SP*, pages 459–474, 05 2014.

DCI11. Angelo De Caro and Vincenzo Iovino. jpbc: Java pairing based cryptography. In *ISCC*, pages 850–855, 2011.

DKNS04. Yevgeniy Dodis, Aggelos Kiayias, Antonio Nicolosi, and Victor Shoup. Anonymous identification in ad hoc groups. In *EUROCRYPT*, 2004.

eth17. Ethereum Foundation EIP-197: Precompiled contracts for optimal ate pairing check on the elliptic curve alt bn128. `https://eips.ethereum.org/EIPS/eip-197`, 2017.

FS87. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology — CRYPTO' 86*, pages 186–194, 1987.

GK15. Jens Groth and MArkulf Kohlweiss. One-out-of-many proofs: Or how to leak a secret and spend a coin. In *EUROCRYPT*, 2015.

HC24. Xiangyu Hui and Sid Chi-Kin Chau. Note on the Proof of Linkability of DualDory. Technical report, CSIRO Data61, 2024.

Lee21. Jonathan Lee. Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments. In *TCC*, page 1–34, 2021.

LLNW16. Benoît Libert, San Ling, Khoa Nguyen, and Huaxiong Wang. Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In *EUROCRYPT*, pages 1–31, 2016.

LRR+19. Russell W. F. Lai, Viktoria Ronge, Tim Ruffing, Dominique Schröder, Sri Aravinda Krishnan Thyagarajan, and Jiafan Wang. Omniring: Scaling private payments without trusted setup. In *ACM CCS*, page 31–48, 2019.

Noe15. Shen Noether. Ring signature confidential transactions for Monero. *Cryptology ePrint Archive - IACR*, 2015.

RST01. Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. ASIACRYPT, page 552–565, 2001.

WCL24. Nan Wang, Sid Chi-Kin Chau, and Dongxi Liu. SwiftRange: A Short and Efficient Zero-Knowledge Range Argument For Confidential Transactions and More. In *IEEE SP*, 2024.

YEL+21. Tsz Hon Yuen, Muhammed F. Esgin, Joseph K. Liu, Man Ho Au, and Zhimin Ding. Dualring: Generic construction of ring signatures with efficient instantiations. In *CRYPTO*, 2021.

YSL+20. Tsz Hon Yuen, Shi-Feng Sun, Joseph K. Liu, Man Ho Au, Muhammed F. Esgin, Qingzhao Zhang, and Dawu Gu. Ringct 3.0 for blockchain confidential transaction: Shorter size and stronger security. In *FC*, pages 464–483, 2020.

Appendix

10 Recursive Dory for Inner-product Relations

Dory [Lee21] provides a compressed protocol with logarithmic verification efficiency and proof size for $\langle D_0 \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{\Omega}}, \vec{\mathbf{\Theta}}) \cdot \mathtt{Q}^{\mathsf{r}_0},\ D_1 \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{\Omega}}, \vec{\mathbf{\Lambda}}) \cdot \mathtt{Q}^{\mathsf{r}_1},\ D_2 \stackrel{?}{=} \mathsf{e}(\vec{\mathbf{\Gamma}}, \vec{\mathbf{\Theta}}) \cdot \mathtt{Q}^{\mathsf{r}_2} \rangle$, given commitments $D_0, D_1, D_2 \in \mathbb{G}_T$ and known random generators $\vec{\mathbf{\Gamma}} \xleftarrow{\$} \mathbb{G}_1^n, \vec{\mathbf{\Lambda}} \xleftarrow{\$} \mathbb{G}_2^n$, with some private witness ($\vec{\mathbf{\Omega}} \in \mathbb{G}_1^n, \vec{\mathbf{\Theta}} \in \mathbb{G}_2^n$, $\mathsf{r}_0, \mathsf{r}_1, \mathsf{r}_2 \in \mathbb{Z}_p^*$). We describe the Dory protocol $\mathit{\Pi}_{\text{do.ip}}$ using a recursive argument in Fig. 11. Note that the choices of $\vec{\mathbf{\Gamma}}', \vec{\mathbf{\Lambda}}'$ do not matter. One possible setting is $\vec{\mathbf{\Gamma}}' \triangleq \vec{\mathbf{\Gamma}}_L, \vec{\mathbf{\Lambda}}' \triangleq \vec{\mathbf{\Lambda}}_L$.

The proof size of Dory includes $6 \log n$ $\mathbb{G}_T$ elements, 1 $\mathbb{G}_1$ element and 1 $\mathbb{G}_2$ element. The verification includes 1 pairing, $10 \log n + 2$ $\mathbb{G}_T$ exponentiations, 1 $\mathbb{G}_1$ exponentiation and 1 $\mathbb{G}_2$ exponentiation. The precomputation includes $3n$ pairings. The proving includes $3n$ pairings, $2 \log n$ $\mathbb{G}_1$ exponentiations and $2 \log n$ $\mathbb{G}_2$ exponentiations.

Batching. It is possible to batch multiple Dory proofs into a single proof [Lee21]. Given $\big(D_0 = \mathsf{e}(\vec{\mathbf{\Omega}}, \vec{\mathbf{\Theta}}) \cdot \mathtt{Q}^{\mathsf{r}_0}, D_1 = \mathsf{e}(\vec{\mathbf{\Omega}}, \vec{\mathbf{\Lambda}}) \cdot \mathtt{Q}^{\mathsf{r}_1}, D_2 = \mathsf{e}(\vec{\mathbf{\Gamma}}, \vec{\mathbf{\Theta}}) \cdot \mathtt{Q}^{\mathsf{r}_2}\big)$ and $\big(D_0' = \mathsf{e}(\vec{\mathbf{\Omega}}', \vec{\mathbf{\Theta}}') \cdot \mathtt{Q}^{\mathsf{r}'_0}, D_1' = \mathsf{e}(\vec{\mathbf{\Omega}}', \vec{\mathbf{\Lambda}}) \cdot \mathtt{Q}^{\mathsf{r}'_1}, D_2' = \mathsf{e}(\vec{\mathbf{\Gamma}}, \vec{\mathbf{\Theta}}') \cdot \mathtt{Q}^{\mathsf{r}'_2}\big)$ with shared generators $(\vec{\mathbf{\Gamma}}, \vec{\mathbf{\Lambda}})$, we define $\mathtt{X} \triangleq \mathsf{e}(\vec{\mathbf{\Omega}}, \vec{\mathbf{\Theta}}') \cdot \mathsf{e}(\vec{\mathbf{\Omega}}', \vec{\mathbf{\Theta}}) \cdot \mathtt{Q}^{\mathsf{r}_\mathtt{X}}$, $D_0'' \triangleq D_0^{\gamma^2} \cdot \mathtt{X}^{\gamma} \cdot D_0'$, $D_1'' \triangleq D_1^{\gamma} \cdot D_1'$ and $D_2'' \triangleq D_2^{\gamma} \cdot D_2'$, where $\gamma \xleftarrow{\$} \mathbb{Z}_p^*$. Then the new witnesses to (D_0'', D_1'', D_2'') are $(\vec{\mathbf{\Omega}}'' \triangleq \vec{\mathbf{\Omega}} \circ \vec{\mathbf{\Omega}}'^{\gamma}, \vec{\mathbf{\Theta}}'' \triangleq \vec{\mathbf{\Theta}} \circ \vec{\mathbf{\Theta}}'^{\gamma}, \mathsf{r}''_0 \triangleq \gamma^2 \mathsf{r}_0 + \gamma \mathsf{r}_\mathtt{X} + \mathsf{r}'_0, \mathsf{r}''_1 \triangleq \gamma \mathsf{r}_1 + \mathsf{r}'_1, \mathsf{r}''_2 \triangleq \gamma \mathsf{r}_2 + \mathsf{r}'_2)$.

Fig. 11: Recursive Dory protocol for checking inner-product relations

$$\mathit{\Pi}_{\text{do.ip}}\Big[n \in \mathbb{Z}^+, \vec{\mathbf{\Gamma}} \in \mathbb{G}_1^n, \vec{\mathbf{\Lambda}} \in \mathbb{G}_2^n, D_0 \in \mathbb{G}_T, D_1 \in \mathbb{G}_T, D_2 \in \mathbb{G}_T;\ \vec{\mathbf{\Omega}} \in \mathbb{G}_1^n, \vec{\mathbf{\Theta}} \in \mathbb{G}_2^n, \mathsf{r}_0 \in \mathbb{Z}_p^*, \mathsf{r}_1 \in \mathbb{Z}_p^*, \mathsf{r}_2 \in \mathbb{Z}_p^*\Big]$$

$\mathcal{V}$: Pre-Compute $\quad \chi \triangleq \mathsf{e}(\vec{\mathbf{\Gamma}}, \vec{\mathbf{\Lambda}}) \in \mathbb{G}_T, \quad \mathit{\Delta}_{1L} \triangleq \mathsf{e}(\vec{\mathbf{\Gamma}}_L, \vec{\mathbf{\Lambda}}') \in \mathbb{G}_T, \mathit{\Delta}_{1R} \triangleq \mathsf{e}(\vec{\mathbf{\Gamma}}_R, \vec{\mathbf{\Lambda}}') \in \mathbb{G}_T$ (108)

$\mathit{\Delta}_{2L} \triangleq \mathsf{e}(\vec{\mathbf{\Gamma}}', \vec{\mathbf{\Lambda}}_L) \in \mathbb{G}_T, \mathit{\Delta}_{2R} \triangleq \mathsf{e}(\vec{\mathbf{\Gamma}}', \vec{\mathbf{\Lambda}}_R) \in \mathbb{G}_T$ (109)

IF $n = 1$

$\mathcal{P}$: $\mathit{\Omega}' \xleftarrow{\$} \mathbb{G}_1, \quad \mathit{\Theta}' \xleftarrow{\$} \in \mathbb{G}_2, \quad \mathsf{r}_{P_1}, \mathsf{r}_{P_2}, \mathsf{r}_Q, \mathsf{r}_R \xleftarrow{\$} \mathbb{Z}_p^*$ (110)

$\mathcal{P} \Rightarrow \mathcal{V}$: $P_1 \triangleq \mathsf{e}(\mathit{\Omega}', \mathit{\Gamma}) \cdot \mathtt{Q}^{\mathsf{r}_{P_1}} \in \mathbb{G}_T, \quad P_2 \triangleq \mathsf{e}(\mathit{\Lambda}, \mathit{\Theta}') \cdot \mathtt{Q}^{\mathsf{r}_{P_2}} \in \mathbb{G}_T$ (111)

$Q \triangleq \mathsf{e}(\mathit{\Omega}', \mathit{\Theta}) \cdot \mathsf{e}(\mathit{\Omega}, \mathit{\Theta}') \cdot \mathtt{Q}^{\mathsf{r}_Q} \in \mathbb{G}, \quad R \triangleq \mathsf{e}(\mathit{\Omega}', \mathit{\Theta}') \cdot \mathtt{Q}^{\mathsf{r}_R} \in \mathbb{G}_T$ (112)

$\mathcal{P} \Leftarrow \mathcal{V}$: $\epsilon \xleftarrow{\$} \mathbb{Z}_p^*$ (113)

$\mathcal{P} \Rightarrow \mathcal{V}$: $E_1 \triangleq \mathit{\Omega}' \cdot \mathit{\Omega}^{\epsilon} \in \mathbb{G}_1, \quad E_2 \triangleq \mathit{\Theta}' \cdot \mathit{\Theta}^{\epsilon} \in \mathbb{G}_2$ (114)

$\mathsf{r}_1 \triangleq \mathsf{r}_{P_1} + \epsilon \cdot \mathsf{r}_1 \in \mathbb{Z}_p^*, \quad \mathsf{r}_2 \triangleq \mathsf{r}_{P_2} + \epsilon \cdot \mathsf{r}_2 \in \mathbb{Z}_p^*, \quad \mathsf{r}_0 \triangleq \mathsf{r}_R + \epsilon \cdot \mathsf{r}_Q + \epsilon^2 \cdot \mathsf{r}_0 \in \mathbb{Z}_p^*$ (115)

$\mathcal{V}$: $\theta \xleftarrow{\$} \mathbb{Z}_p^*, \quad \mathsf{r} \triangleq \mathsf{r}_0 + \theta \cdot \mathsf{r}_2 + \theta^{-1} \cdot \mathsf{r}_1$ (116)

Check $\mathsf{e}(E_1 \cdot \mathit{\Gamma}^{\theta}, E_2 \cdot \mathit{\Lambda}^{\theta^{-1}}) \stackrel{?}{=} \chi \cdot R \cdot Q^{\epsilon} \cdot D_0^{\epsilon^2} \cdot P_2^{\theta} \cdot D_2^{\theta \cdot \epsilon} \cdot P_1^{\theta^{-1}} \cdot D_1^{\theta^{-1} \cdot \epsilon} \cdot \mathtt{Q}^{\mathsf{r}}$ (117)

ELSE $n > 1$

$\mathcal{P}$: $\mathsf{r}_{1L}, \mathsf{r}_{1R}, \mathsf{r}_{2L}, \mathsf{r}_{2R}, \mathsf{r}_{W1}, \mathsf{r}_{W2} \xleftarrow{\$} \mathbb{Z}_p^*$ (118)

$\mathcal{P} \Rightarrow \mathcal{V}$: $D_{1L} \triangleq \mathsf{e}(\vec{\mathbf{\Omega}}_L, \vec{\mathbf{\Lambda}}') \cdot \mathtt{Q}^{\mathsf{r}_{1L}} \in \mathbb{G}_T, \quad D_{1R} \triangleq \mathsf{e}(\vec{\mathbf{\Omega}}_R, \vec{\mathbf{\Lambda}}') \cdot \mathtt{Q}^{\mathsf{r}_{1R}} \in \mathbb{G}_T$ (119)

$D_{2L} \triangleq \mathsf{e}(\vec{\mathbf{\Gamma}}', \vec{\mathbf{\Theta}}_L) \cdot \mathtt{Q}^{\mathsf{r}_{2L}} \in \mathbb{G}_T, \quad D_{2R} \triangleq \mathsf{e}(\vec{\mathbf{\Gamma}}', \vec{\mathbf{\Theta}}_R) \cdot \mathtt{Q}^{\mathsf{r}_{2R}} \in \mathbb{G}_T$ (120)

$\mathcal{P} \Leftarrow \mathcal{V}$: $\beta \xleftarrow{\$} \mathbb{Z}_p^*$ (121)

$\mathcal{P}$: $\vec{\mathbf{\Omega}}^{\circ} \triangleq \vec{\mathbf{\Omega}} \circ \vec{\mathbf{\Gamma}}^{\beta} \in \mathbb{G}_1^n, \quad \vec{\mathbf{\Theta}}^{\circ} \triangleq \vec{\mathbf{\Theta}} \circ \vec{\mathbf{\Lambda}}^{\beta^{-1}} \in \mathbb{G}_2^n$ (122)

$\mathcal{P} \Rightarrow \mathcal{V}$: $W_1 \triangleq \mathsf{e}(\vec{\mathbf{\Omega}}_L^{\circ}, \vec{\mathbf{\Theta}}_R^{\circ}) \cdot \mathtt{Q}^{\mathsf{r}_{W1}} \in \mathbb{G}_T, \quad W_2 \triangleq \mathsf{e}(\vec{\mathbf{\Omega}}_R^{\circ}, \vec{\mathbf{\Theta}}_L^{\circ}) \cdot \mathtt{Q}^{\mathsf{r}_{W2}} \in \mathbb{G}_T, \quad \tilde{\mathsf{r}}_0 \triangleq \mathsf{r}_0 + \beta \cdot \mathsf{r}_1 + \beta^{-1} \cdot \mathsf{r}_2$ (123)

$\mathcal{P} \Leftarrow \mathcal{V}$: $\alpha \xleftarrow{\$} \mathbb{Z}_p^*$ (124)

$\mathcal{P}$: $\vec{\mathbf{\Omega}}' \triangleq (\vec{\mathbf{\Omega}}_L^{\circ})^{\alpha} \circ \vec{\mathbf{\Omega}}_R^{\circ} \in \mathbb{G}_1^{\frac{n}{2}}, \quad \vec{\mathbf{\Theta}}' \triangleq (\vec{\mathbf{\Theta}}_L^{\circ})^{\alpha^{-1}} \circ \vec{\mathbf{\Theta}}_R^{\circ} \in \mathbb{G}_2^{\frac{n}{2}}$ (125)

$\mathcal{V}$: $D_0' \triangleq D_0 \cdot \chi \cdot D_1^{\beta^{-1}} \cdot D_2^{\beta} \cdot W_1^{\alpha} \cdot W_2^{\alpha^{-1}} \in \mathbb{G}_T, \quad D_1' \triangleq D_{1L}^{\alpha} \cdot D_{1R} \cdot \mathit{\Delta}_{1L}^{\alpha\beta} \cdot \mathit{\Delta}_{1R}^{\beta} \in \mathbb{G}_T$ (126)

$D_2' \triangleq D_{2L}^{\alpha^{-1}} \cdot D_{2R} \cdot \mathit{\Delta}_{2L}^{\alpha^{-1}\beta^{-1}} \cdot \mathit{\Delta}_{2R}^{\beta^{-1}} \in \mathbb{G}_T$ (127)

$\mathsf{r}'_0 \triangleq \tilde{\mathsf{r}}_0 + \alpha \cdot \mathsf{r}_{W1} + \alpha^{-1} \cdot \mathsf{r}_{W2}, \quad \mathsf{r}'_1 \triangleq \alpha \cdot \mathsf{r}_{1L} + \mathsf{r}_{1R}, \quad \mathsf{r}'_2 \triangleq \alpha^{-1} \cdot \mathsf{r}_{2L} + \mathsf{r}_{2R}$ (128)

$\mathcal{V}$ & $\mathcal{P}$: Run $\mathit{\Pi}_{\text{do.ip}}[\frac{n}{2}, \vec{\mathbf{\Gamma}}', \vec{\mathbf{\Lambda}}', D_0', D_1', D_2'; \vec{\mathbf{\Omega}}', \vec{\mathbf{\Theta}}', \mathsf{r}'_0, \mathsf{r}'_1, \mathsf{r}'_2]$

Theorem 29 ([Lee21]). *$\Pi_{\text{do.ip}}$ satisfies perfect completeness, SHVZK and CWE (or it breaks the SXDH assumption).*

11 Proofs for LLRing-DL

Theorem 30. *$\Pi_{\text{sbp.ip}}$ satisfies computationally witness-extended emulation (CWE) or it can obtain a non-trivial discrete logarithm relation.*

First, we consider checking separate relations:

$$\vec{\mathbf{G}}^{\vec{\mathbf{l}}} \stackrel{?}{=} Z_1, \quad \vec{\mathbf{H}}^{\vec{\mathbf{r}}} \cdot K^{\hat{t}} \stackrel{?}{=} Z_2 \tag{129}$$

We follow the approach in Bulletproofs [BBB+18]. Note that typical Bulletproofs satisfies 4-special soundness at each iteration. The segregated Bulletproofs behaves the same manner as typical Bulletproofs, and hence also satisfies 4-special soundness. Hence, CWE follows from applying Lemma 15.

Next, we consider checking a combined relation by random liner combination:

$$(\vec{\mathbf{G}}^{\vec{\mathbf{l}}})^{\theta} \cdot \vec{\mathbf{H}}^{\vec{\mathbf{r}}} \cdot K^{\hat{t}} \stackrel{?}{=} Z_1^{\theta} \cdot Z_2 \tag{130}$$

Since $\theta \xleftarrow{\$} \mathbb{Z}_p^*$ is known by the verifier only, the probability the above equation is true but $\hat{t} \neq \langle \vec{\mathbf{l}}, \vec{\mathbf{r}} \rangle$ is bounded by $\text{negl}(\lambda)$, based on Schwartz-Zippel Lemma.

It is evident that $\Pi_{\text{mlr.dl}}$ satisfies perfect completeness.

Theorem 31. *Linkable ring signature protocol $\Pi_{\text{mlr.dl}}$ satisfies anonymity.*

Assume there exists a PPT adversary $\mathcal{A}$ that can break the anonymity of $\Pi_{\text{mlr.dl}}$ with non-negligible probability, then there exists a distinguisher $\mathcal{D}$ that can break the DDH assumption in the following game that utilizes $\mathcal{A}$:

Game Setup. Let $\mathsf{pp} \leftarrow \mathtt{Setup}[1^\lambda]$ and $(\vec{\mathbf{pk}}, \vec{\mathbf{sk}}) \leftarrow \mathtt{KeyGen}[\mathsf{pp}]$. Consider $(X, Y, Z) \in \mathbb{G}^3$, where either (X, Y, Z) is a DDH tuple (i.e., $X = P^x, Y = P^y, Z = P^{xy}$) or $(X, Y, Z) \xleftarrow{\$} \mathbb{G}^3$. Pick $j \in [n]$ and set $\mathsf{pk}_j = X$. We define the following oracles for $\mathcal{A}$ when accessing $\Pi_{\text{mlr.dl}}$:

- *Corruption Oracle $\mathcal{CO}$*: When $\mathcal{CO}$ is queried with $(\vec{\mathbf{pk}}, i \in [n])$,
  - If $i \neq j$, then it returns sk, where $P^{\mathsf{sk}} = \mathsf{pk}_i$.
  - If $i = j$, then it aborts.
- *Random Oracle $\mathcal{RO}$*: This oracle simulates $\mathtt{Hash}'[\mathsf{f}]$. When $\mathcal{RO}$ is queried with f, if f has not been queried before, it returns $Y^{\mathsf{r_f}}$, where $\mathsf{r_f} \xleftarrow{\$} \mathbb{Z}_p^*$. Otherwise, it returns $Y^{\mathsf{r_f}}$, where $\mathsf{r_f}$ is previously chosen for f. We use $\mathtt{Hash}'[\mathsf{f}]$ in Π_{tag}
- *Signing Oracle $\mathcal{SO}$*: When $\mathcal{SO}$ is queried with $(\vec{\mathbf{pk}}, i \in [n], \mathsf{m}, \mathsf{f})$, it returns a valid signature σ as follows:
  - If $i \neq j$, then it returns $\sigma \leftarrow \mathtt{Sign}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}, \mathsf{sk}_i]$.
  - If $i = j$, then it returns a simulated signature as follows. Let $\mathsf{cm} \triangleq X \cdot Q^{\mathsf{r_{cm}}}$ and $\mathsf{tag} \triangleq Z^{\mathsf{r_f}}$. To generate a valid proof for tag, let $(\mathsf{a}', \mathsf{r}') \xleftarrow{\$} \mathbb{Z}_{\mathsf{p}}^{*2}$,

$$A \triangleq \mathtt{Hash}'[\mathsf{f}]^{\mathsf{a}'} \cdot \mathsf{tag}^{-\rho}, \tag{131}$$

$$B \triangleq P^{\mathsf{a}'} \cdot X^{-\rho} \cdot Q^{\mathsf{b}' - \mathsf{r_f}\rho} \tag{132}$$

Hence, $(A, B, \mathsf{a}', \mathsf{r}', \mathsf{tag})$ can pass the verification in Π_{tag}. Note that it is statistically indistinguishable from other valid proofs that pass the verification.

Next, let $\vec{\mathbf{c}}$ represent a unit basis vector, such that $c_i = 0$ when $i \neq j$ and $c_j = 1$. Finally, to generate the rest of signature proof by $\Pi_{\text{mlr.dl}}$.

Game. $\mathcal{D}$ tries to determine if (X, Y, Z) is a DDH tuple. By the definition of anonymity (Definition 21), we assume that $\mathcal{A}$ issues $(n-2)$ corruption queries for distinct indices. In the game, $\mathcal{A}$ will be required to provide a tuple $(i_1, i_2, \mathsf{m}, \mathsf{f}, \vec{\mathbf{pk}}^*)$. $\mathcal{D}$ aborts if $(\mathsf{f}, \mathsf{pk}^*_{i_1}) \in \Sigma_{\mathcal{SO}}$, or $(\mathsf{f}, \mathsf{pk}^*_{i_2}) \in \Sigma_{\mathcal{SO}}$, or $\mathsf{pk}_j \notin \{\mathsf{pk}^*_{i_1}, \mathsf{pk}^*_{i_2}\}$, or $(\mathsf{pk}^*_{i_1}, \mathsf{pk}^*_{i_2}) \not\subseteq \vec{\mathbf{pk}} \backslash \vec{\mathbf{pk}}_{\mathcal{CO}}$. The probability that $\mathcal{D}$ does not abort is $\frac{n-1}{\binom{n}{2}} = \frac{2}{n}$. $\mathcal{D}$ sets $b \leftarrow j$. Suppose $\mathcal{A}$ can break the anonymity with a non-negligible probability ϵ. If (X, Y, Z) is a DDH tuple, then $\mathcal{A}$ should correctly guess $b' = j$ with probability ϵ. As a result, $\mathcal{D}$ can break the DDH assumption with a non-negligible probability of $\epsilon \cdot \frac{2}{n}$.

Theorem 32. *Linkable ring signature protocol $\Pi_{\text{mlr.dl}}$ satisfies linkability.*

Assume there exist a PPT adversary $\mathcal{A}$, which can break the linkability of $\Pi_{\text{mlr.dl}}$ with non negligible property ϵ, then there exists an algorithm $\mathcal{B}$ that can break the DLOG assumption between $X, Y \xleftarrow{\$} \mathbb{G}$ in the following game that utilizes $\mathcal{A}$:

Game Setup. Given public parameters $\mathsf{pp} \leftarrow \mathtt{Setup}[1^\lambda]$. Let $P = X, R = Q = Y$ in pp. $(\vec{\mathbf{pk}}, \vec{\mathbf{sk}}) \leftarrow \mathtt{KeyGen}[\mathsf{pp}]$. We define the following oracles for $\mathcal{A}$ when accessing $\Pi_{\text{mlr.dl}}$:

- *Corruption Oracle $\mathcal{CO}$*: When $\mathcal{CO}$ is queried with index $i \in [n]$, return sk_i.
- *Signning Oracle $\mathcal{SO}$*: $\mathcal{SO}$: When $\mathcal{SO}$ is queried with $(\vec{\mathbf{pk}}, i \in [n], \mathsf{m}, \mathsf{f})$, return a valid signature $\sigma \leftarrow \mathtt{Sign}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}, \mathsf{sk}_i]$.

Game. $\mathcal{B}$ tries to derive the discrete logarithm relations among X and Y. By the definition of prefix-linkability (Definition 23), $\mathcal{A}$ generates $n+1$ signatures based on the same prefix f, where $\mathtt{Verify}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}, \sigma] = 1$ must holds for all $i \in [n+1]$, and $\mathtt{Link}[\mathsf{pp}, \mathsf{f}, \vec{\mathbf{pk}}, \sigma_i, \sigma_j] = 0$ holds for all $i \neq j \in [n+1]$. This implies that $\mathsf{tag}_i \neq \mathsf{tag}_j$ for all $i \neq j \in [n+1]$. By leveraging the CWE property of the tag proof, we can efficiently extract witnesses $\mathsf{sk}'_i, \mathsf{r}'_{\text{cm}_i} \in \mathbb{Z}_p$ such that $\forall i \in [n+1]$,

$$\mathsf{cm}_i = P^{\mathsf{sk}'_i} Q^{\mathsf{r}'_{\text{cm}_i}} \wedge \mathsf{tag}_i = \mathtt{Hash}[\mathsf{f}]^{\mathsf{sk}'_i}.$$

Since all of the tags are pairwise distinct, and each value sk'_i is uniquely determined, this implies the $n+1$ cm_i opens to $n+1$ distinct values $\mathsf{sk}'_i \in \mathbb{Z}_p$. Without loss of generality, we assume that $\mathsf{sk}'_{n+1} \notin \{\mathsf{sk}_1, \ldots, \mathsf{sk}_n\}$. Next, utilizing the CWE property of $\Pi_{\text{sbp.ip}}$ and Bulletproofs' range proof, we can extract $\mathsf{r}'_{\mathsf{A}_{n+1}}$ and a unit basis vector $\vec{\mathbf{c}}'_{n+1}$ from σ_{n+1}, in which the k-th bit is 1, $k \in [n]$, and all other bits are 0. Thanks to the CWE property of Π_{sch}, we can further extract $\mathsf{r}'_{j_{n+1}}, \hat{\mathsf{r}}'_{A_{n+1}}$ After the witness extraction the following equation holds:

$$(\vec{\mathbf{pk}} \circ \vec{\mathbf{R}})^{\vec{\mathbf{c}}'_{n+1}} \cdot Q^{\mathsf{r}'_{A_{n+1}}} = \mathsf{cm}_{n+1} \cdot \hat{A}_{n+1} \cdot Q^{\mathsf{r}'_{\text{cm}_{n+1}} + \hat{\mathsf{r}}'_{A_{n+1}}} \tag{133}$$

$$\Rightarrow \mathsf{pk}_k \cdot R_k = P^{\mathsf{sk}'_{n+1}} \cdot R^{\mathsf{r}'_{j_{n+1}}} \cdot Q^{\mathsf{r}'_{\text{cm}_{n+1}} + \hat{\mathsf{r}}'_{A_{n+1}} - \mathsf{r}'_{A_{n+1}}} \tag{134}$$

We know we can substitute R_k as $R^{\mathtt{Hash}[\mathsf{pk}_k]}$, then we obtain

$$\mathsf{pk}_k \cdot R^{\mathtt{Hash}[\mathsf{pk}_k]} = P^{\mathsf{sk}'_{n+1}} \cdot R^{\mathsf{r}'_{j_{n+1}}} \cdot Q^{(\mathsf{r}'_{\text{cm}_{n+1}} + \hat{\mathsf{r}}'_{A_{n+1}} - \mathsf{r}'_{A_{n+1}})} \tag{135}$$

$$\Rightarrow P^{(\mathsf{sk}_k - \mathsf{sk}'_{n+1})} = Q^{(\mathsf{r}'_{\text{cm}_{n+1}} - \hat{\mathsf{r}}'_{A_{n+1}} - \mathsf{r}'_{A_{n+1}})} \cdot R^{(\mathsf{r}'_{j_{n+1}} - \mathtt{Hash}[\mathsf{pk}_k])} \tag{136}$$

$$\Rightarrow X^{(\mathsf{sk}_k - \mathsf{sk}'_{n+1})} = Y^{(\mathsf{r}'_{\text{cm}_{n+1}} - \hat{\mathsf{r}}'_{A_{n+1}} - \mathsf{r}'_{A_{n+1}} + \mathsf{r}'_{j_{n+1}} - \mathtt{Hash}[\mathsf{pk}_k])} \tag{137}$$

$$\Rightarrow X^{\frac{(\mathsf{sk}_k - \mathsf{sk}'_{n+1})}{(\mathsf{r}'_{\text{cm}_{n+1}} - \hat{\mathsf{r}}'_{A_{n+1}} - \mathsf{r}'_{A_{n+1}} + \mathsf{r}'_{j_{n+1}} - \mathtt{Hash}[\mathsf{pk}_k])}} = Y \tag{138}$$

Then $\mathcal{B}$ can return $x = \frac{(\mathsf{sk}_k - \mathsf{sk}'_{n+1})}{(\mathsf{r}'_{\text{cm}_{n+1}} - \hat{\mathsf{r}}'_{A_{n+1}} - \mathsf{r}'_{A_{n+1}} + \mathsf{r}'_{j_{n+1}} - \mathtt{Hash}[\mathsf{pk}_k])}$ to break the Dlog assumption with non negligible probability of ϵ.

Theorem 33. *Linkable ring signature protocol $\Pi_{\text{mlr.dl}}$ satisfies non-sladerability.*

Assume there exist a PPT adversary $\mathcal{A}$, which can break the non-sladerability of $\Pi_{\text{mlr.dl}}$ with non negligible property ϵ, then there exists an algorithm $\mathcal{B}$ that can break the DLOG assumption between $X, Y \xleftarrow{\$} \mathbb{G}$ in the following game that utilizes $\mathcal{A}$:

Game Setup. Given public parameters $\mathsf{pp} \leftarrow \mathtt{Setup}[1^\lambda]$. Let $P = X$ in pp. $(\vec{\mathbf{pk}}, \vec{\mathbf{sk}}) \leftarrow \mathtt{KeyGen}[\mathsf{pp}]$. Pick $j \in [n]$ and set $\mathsf{pk}_j = Y$. We define the following oracles for $\mathcal{A}$ when accessing $\Pi_{\text{mlr.dl}}$:

- *Random Oracle $\mathcal{RO}$*: This oracle simulates $\mathtt{Hash}'[\mathsf{f}]$. When $\mathcal{RO}$ is queried with f, if f has not been queried before, it returns $Y^{\mathsf{r_f}}$, where $\mathsf{r_f} \xleftarrow{\$} \mathbb{Z}_p^*$. Otherwise, it returns $Y^{\mathsf{r_f}}$, where $\mathsf{r_f}$ is previously chosen for f. We use $\mathtt{Hash}'[\mathsf{f}]$ in $\mathit{\Pi}_{\mathrm{tag}}$
- *Corruption Oracle $\mathcal{CO}$*: When $\mathcal{CO}$ is queried with $i \in [n]$, if $i = j$, it aborts. Otherwise, it returns sk_i.
- *Signing Oracle $\mathcal{SO}$*: When $\mathcal{SO}$ is queried with $(\vec{\mathbf{pk}}, i \in [n], \mathsf{m}, \mathsf{f})$, it returns a valid signature σ as follows:
  - If $i \neq j$, then it returns $\sigma \leftarrow \mathtt{Sign}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}, \mathsf{sk}_i]$ and add σ to set $\mathit{\Sigma}_{\mathcal{SO}}$.
  - If $i = j$, then it returns a simulated signature as follows. Let $\mathsf{cm} \triangleq Y \cdot Q^{\mathsf{r_{cm}}}$ and $\mathsf{tag} \triangleq Y^{\mathsf{r_f}}$. To generate a valid proof for tag, let $(\mathsf{a}', \mathsf{r}') \xleftarrow{\$} \mathbb{Z}_{\mathsf{p}}^{*2}$,

$$A \triangleq \mathtt{Hash}'[\mathsf{f}]^{\mathsf{a}'} \cdot \mathsf{tag}^{-\rho}, \tag{139}$$

$$B \triangleq P^{\mathsf{a}'} \cdot X^{-\rho} \cdot Q^{\mathsf{b}' - \mathsf{r_f}\rho} \tag{140}$$

    Hence, $(A, B, \mathsf{a}', \mathsf{r}', \mathsf{tag})$ can pass the verification in $\mathit{\Pi}_{\mathrm{tag}}$. Note that it is statistically indistinguishable from other valid proofs that pass the verification.
    Next, let $\vec{\mathbf{c}}$ represent a unit basis vector, such that $c_i = 0$ when $i \neq j$ and $c_j = 1$. Finally, to generate the rest of signature proof by $\mathit{\Pi}_{\mathrm{mlr.dl}}$. Before return the simulated signature σ, add σ to set $\mathit{\Sigma}_{\mathcal{SO}}$.

Game. $\mathcal{B}$ tries to derive the discrete logarithm relations among X and Y. By the definition of prefix-linkability (Definition 24), $\mathcal{A}$ first produce a signature σ', a prefix f, and a message m', such that $\mathtt{Verify}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}', \sigma'] = 1$, without issuing any corruption queries. $\mathcal{B}$ aborts if $\sigma' \in \mathit{\Sigma}_{\mathcal{SO}}$. $\mathcal{B}$ retrieves tag' from $\pi_{\mathsf{tag}'}$ and it will abort if $\exists i \in [n]_{i \neq j}, \mathtt{Hash}[\mathsf{f}]^{\mathsf{sk}_i} = \mathsf{tag}'$. Note that all public keys and signatures from $\mathcal{SO}$ are statistically indistinguishable. Therefore, the probability of $\mathcal{B}$ does not abort is $\frac{1}{n}$. Next, make $\mathcal{CO}$ accessible, $\mathcal{A}$ should returns a tuple (σ'', m''), where $\mathtt{Verify}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}'', \sigma''] = 1 \wedge \mathtt{Link}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \sigma', \sigma''] = 1$. By utilizing the CWE property of $\mathit{\Pi}_{\mathrm{mlr.dl}}$, we can extract $\mathsf{cm}'' = pk_i \cdot Q^{r''_{cm}}$ from σ'', for some $i \in [n]$, and by the CWE property of tag proof we have $\mathsf{cm}'' = X^{\mathsf{sk}_i} \cdot Q^{r''_{cm}}$ for some $i \in [n]$. Since $\mathcal{B}$ does not abort, and consider $\mathsf{tag}' \neq \mathtt{Hash}[\mathsf{f}]^{\mathsf{sk}_i} \forall i \in [n] : i \neq j$. Hence, we can deduce that $\mathsf{tag}' = \mathtt{Hash}[\mathsf{f}]^{\mathsf{sk}_j}$. Thanks to the CWE property of the tag proof, $\mathcal{B}$ is able to extract sk_j and r'_{cm}, where $X^{\mathsf{sk}_j} = Y$. Then $\mathcal{B}$ return sk_j to break the Dlog assumption with non negligible probability of $\epsilon \cdot \frac{1}{n}$.

Theorem 34. *Linkable ring signature protocol $\mathit{\Pi}_{\mathrm{mlr.dl}}$ satisfies unforgeability.*

By Theorem 8 in [BEHM22], if a ring signature scheme is linkable and non-slanderable, then it is also unforgeable.

12 Proofs for LLRing-P

It is evident that $\mathit{\Pi}_{\mathrm{mlr.p}}$ satisfies perfect completeness.

Theorem 35. *Linkable ring signature protocol $\mathit{\Pi}_{\mathrm{mlr.p}}$ satisfies anonymity.*

Assume there exists a PPT adversary $\mathcal{A}$ that can break the anonymity of $\mathit{\Pi}_{\mathrm{mlr.p}}$ with non-negligible probability, then there exists a distinguisher $\mathcal{D}$ that can break the DDH assumption in $\mathbb{G}_T$ in the following game that utilizes $\mathcal{A}$:

Game Setup. Let $\mathsf{pp} \leftarrow \mathtt{Setup}[1^\lambda]$ and $(\vec{\mathbf{pk}}, \vec{\mathbf{sk}}) \leftarrow \mathtt{KeyGen}[\mathsf{pp}]$. Consider $(X, Y, Z) \in \mathbb{G}_T^3$, where either (X, Y, Z) is a DDH tuple (i.e., $X = \mathsf{e}(P, L)^x, Y = \mathsf{e}(P, L)^y, Z = \mathsf{e}(P, L)^{xy}$) or $(X, Y, Z) \xleftarrow{\$} \mathbb{G}_T^3$. Among these, we set $X = \mathsf{e}(X', L)$, where either $X' = P^x$ or $X' \xleftarrow{\$} \mathbb{G}_1$. Pick $j \in [n]$ and set $\mathsf{pk}_j = X'$. We define the following oracles for $\mathcal{A}$ when accessing $\mathit{\Pi}_{\mathrm{mlr.p}}$:

- *Corruption Oracle $\mathcal{CO}$*: When $\mathcal{CO}$ is queried with $(\vec{\mathbf{pk}}, i \in [n])$,
  - If $i \neq j$, then it returns sk, where $P^{\mathsf{sk}} = \mathsf{pk}_i$.
  - If $i = j$, then it aborts.

- *Random Oracle $\mathcal{RO}$*: This oracle simulates $\mathtt{Hash}'[\mathsf{f}]$. When $\mathcal{RO}$ is queried with f, if f has not been queried before, it returns $Y^{\mathsf{r_f}}$, where $\mathsf{r_f} \xleftarrow{\$} \mathbb{Z}_p^*$. Otherwise, it returns $Y^{\mathsf{r_f}}$, where $\mathsf{r_f}$ is previously chosen for f. We use $\mathtt{Hash}'[\mathsf{f}]$ in $\mathit{\Pi}_{\mathrm{tag}}$
- *Signing Oracle $\mathcal{SO}$*: When $\mathcal{SO}$ is queried with $(\vec{\mathbf{pk}}, i \in [n], \mathsf{m}, \mathsf{f})$, it returns a valid signature σ as follows:
  - If $i \neq j$, then it returns $\sigma \leftarrow \mathtt{Sign}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}, \mathsf{sk_i}]$.
  - If $i = j$, then it returns a simulated signature as follows. Let $\mathsf{cm} \triangleq X \cdot Q^{\mathsf{r_{cm}}}$ and $\mathsf{tag} \triangleq Z^{\mathsf{r_f}}$. To generate a valid proof for tag, let $(\mathsf{a}', \mathsf{r}') \xleftarrow{\$} \mathbb{Z}_p^{*2}$,

$$A \triangleq \mathtt{Hash}'[\mathsf{f}]^{\mathsf{a}'} \cdot \mathsf{tag}^{-\rho}, \tag{141}$$

$$B \triangleq P^{\mathsf{a}'} \cdot X^{-\rho} \cdot Q^{\mathsf{b}'-\mathsf{r_f}\rho} \tag{142}$$

    Hence, $(A, B, \mathsf{a}', \mathsf{r}', \mathsf{tag})$ can pass the verification in $\mathit{\Pi}_{\mathrm{tag}}$. Note that it is statistically indistinguishable from other valid proofs that pass the verification.
    Next, let $\vec{\mathbf{c}}$ represent a unit basis vector, such that $c_i = 0$ when $i \neq j$ and $c_j = 1$. Finally, to generate the rest of signature proof by $\mathit{\Pi}_{\mathrm{mlr.p}}$.

Game. $\mathcal{D}$ tries to determine if (X, Y, Z) is a DDH tuple. By the definition of anonymity (Definition 21), we assume that $\mathcal{A}$ issues $(n-2)$ corruption queries for distinct indices. In the game, $\mathcal{A}$ will be required to provide a tuple $(i_1, i_2, \mathsf{m}, \mathsf{f}, \vec{\mathbf{pk}}^*)$. $\mathcal{D}$ aborts if $(\mathsf{f}, \mathsf{pk}_{i_1}^*) \in \Sigma_{\mathcal{SO}}$, or $(\mathsf{f}, \mathsf{pk}_{i_2}^*) \in \Sigma_{\mathcal{SO}}$, or $\mathsf{pk}_j \notin \{\mathsf{pk}_{i_1}^*, \mathsf{pk}_{i_2}^*\}$, or $(\mathsf{pk}_{i_1}^*, \mathsf{pk}_{i_2}^*) \not\subseteq \vec{\mathbf{pk}} \backslash \vec{\mathbf{pk}}_{\mathcal{CO}}$. The probability that $\mathcal{D}$ does not abort is $\frac{n-1}{\binom{n}{2}} = \frac{2}{n}$. $\mathcal{D}$ sets $b \leftarrow j$. Suppose $\mathcal{A}$ can break the anonymity with a non-negligible probability ϵ. If (X, Y, Z) is a DDH tuple, then $\mathcal{A}$ should correctly guess $b' = j$ with probability ϵ. As a result, $\mathcal{D}$ can break the DDH assumption with a non-negligible probability of $\epsilon \cdot \frac{2}{n}$.

Theorem 36. *Linkable ring signature protocol $\mathit{\Pi}_{\mathrm{mlr.p}}$ satisfies linkability.*

Assume there exist a PPT adversary $\mathcal{A}$, which can break the linkability of $\mathit{\Pi}_{\mathrm{mlr.p}}$ with non negligible property ϵ, then there exists an algorithm $\mathcal{B}$ that can break DPair assumption with two generators $P_1, P_2 \in \mathbb{G}_1$.

Game Setup. Given public parameters $\mathsf{pp} \leftarrow \mathtt{Setup}[1^\lambda]$. Let $P = P_1, R = P_2$ in pp. $(\vec{\mathbf{pk}}, \vec{\mathbf{sk}}) \leftarrow \mathtt{KeyGen}[\mathsf{pp}]$. We define the following oracles for $\mathcal{A}$ when accessing $\mathit{\Pi}_{\mathrm{mlr.p}}$:

- *Corruption Oracle $\mathcal{CO}$*: When $\mathcal{CO}$ is queried with index $i \in [n]$, return sk_i.
- *Signning Oracle $\mathcal{SO}$*: $\mathcal{SO}$: When $\mathcal{SO}$ is queried with $(\vec{\mathbf{pk}}, i \in [n], \mathsf{m}, \mathsf{f})$, return a valid signature $\sigma \leftarrow \mathtt{Sign}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}, \mathsf{sk_i}]$.

Game. By the definition of prefix-linkability (Definition 23), $\mathcal{A}$ generates $n+1$ signatures based on the same prefix f, where $\mathtt{Verify}(\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}_i, \sigma_i) = 1$ must holds for all $i \in [n+1]$, and $\mathtt{Link}(\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \sigma_i, \sigma_j) = 0$ holds for all $i \neq j \in [n+1]$. This implies that $\mathsf{tag}_i \neq \mathsf{tag}_j$ for all $i \neq j \in [n+1]$. By leveraging the CWE property of the $\mathit{\Pi}_{\mathrm{tag}}$, we can efficiently extract witnesses $\mathsf{sk}_i', \mathsf{r}_{\mathrm{cm}_i}' \in \mathbb{Z}_p$ such that $\forall i \in [n+1]$,

$$\mathsf{cm}_i = \mathsf{e}(P^{\mathsf{sk}_i'}, L) \cdot Q^{\mathsf{r}_{\mathrm{cm}_i}'} \wedge \mathsf{tag}_i = \mathtt{Hash}[\mathsf{f}]^{\mathsf{sk}_i'}.$$

Since all of the tags are pairwise distinct, and each value sk_i' is uniquely determined, this implies the $n+1$ cm_i opens to $n+1$ distinct values $\mathsf{sk}''_i \in \mathbb{Z}_p$. Without loss of generality, we assume $\mathsf{sk}'_{n+1} \notin \{\mathsf{sk}_1, \ldots, \mathsf{sk}_n\}$. By the CWE property of $\mathit{\Pi}_{\mathrm{sch}}$ we can extract $A_{n+1} = \mathsf{e}(P^{\mathsf{sk}'_{n+1}} \cdot R^{\mathsf{r}'}, L) \cdot \mathtt{Q}^{\mathsf{r}_{A_{n+1}}}$. Based on the CWE property of $\mathit{\Pi}_{\mathrm{do.ip}}$, we can extract the following relations from the Dory proof for Eqn (95) of σ_{n+1}:

$$\begin{cases} \mathsf{e}(\vec{\mathbf{\Omega}}', \vec{\mathbf{H}}) = \mathsf{e}(\vec{L}', \vec{\mathbf{H}}) \\ \mathsf{e}(\vec{\mathbf{G}}, \vec{\mathbf{\Theta}}') = \mathsf{e}(P^{\mathsf{sk}'_{n+1}} \cdot R^{\mathsf{r}'}, L) \\ \mathsf{e}(\vec{\mathbf{\Omega}}', \vec{\mathbf{\Theta}}') = \mathsf{e}(L', L) \end{cases} \tag{143}$$

Suppose $H_i \neq 1$ and $G_i \neq 1$ for all $i \in [n]$. Let $\vec{\mathbf{\Theta}}' = \vec{L}^{\circ \vec{\mathbf{a}}}$ and $\vec{\mathbf{\Omega}}' = \vec{L}'^{\circ \vec{\mathbf{b}}}$ for some vectors $\vec{\mathbf{a}}$ and $\vec{\mathbf{b}}$. First, we obtain

$$\mathsf{e}(\vec{\mathbf{\Omega}}', \vec{\mathbf{H}}) = \mathsf{e}(\vec{L}'^{\circ \vec{\mathbf{b}}}, \vec{H}) = \mathsf{e}(\vec{L}', \vec{\mathbf{H}}) \tag{144}$$

That is, $\mathsf{e}(\vec{L'},\vec{H})^{\vec{\mathbf{b}}} = \mathsf{e}(\vec{L'},\vec{H})^{\vec{\mathbf{1}}}$. If $\vec{\mathbf{b}} \neq \vec{\mathbf{1}}$, we would obtain a non-trivial logarithm relation among $\big(\mathsf{e}(L',H_i)\big)_{i\in[n]}$. Hence, we obtain

$$\begin{cases} \mathsf{e}(\vec{\mathbf{G}},\vec{\mathbf{L}})^{\vec{\mathbf{a}}} = \mathsf{e}(P^{\mathsf{sk}'_{n+1}}\cdot R^{\mathsf{r}'},L) \\ \mathsf{e}(\vec{\mathbf{L}'},\vec{\mathbf{L}})^{\vec{\mathbf{a}}} = \mathsf{e}(L',L) \end{cases} \tag{145}$$

Next, we can extract the following relations from the Dory proof for Eqn (97) of σ_{n+1}:

$$\begin{cases} \mathsf{e}(\vec{\mathbf{\Omega}}''',\vec{\mathbf{H}}) = B_{n+1}\cdot \mathtt{Q}^{-\mathsf{r}_B} \\ \mathsf{e}(\vec{\mathbf{G}},\vec{\mathbf{\Theta}}''') = \mathsf{e}(\vec{\mathbf{G}},\vec{L}) \\ \mathsf{e}(\vec{\mathbf{\Omega}}''',\vec{\mathbf{\Theta}}''') = \mathsf{e}(P^{\mathsf{sk}'_{n+1}}\cdot R^{\mathsf{r}'},L) \end{cases} \tag{146}$$

Similarly, we let $\vec{\mathbf{\Theta}}''' = \vec{L}^{\circ\vec{\mathbf{g}}}$ and $\vec{\mathbf{\Omega}}''' = \vec{\mathbf{G}}^{\circ\vec{\mathbf{h}}}$ for some vectors $\vec{\mathbf{g}}$ and $\vec{\mathbf{h}}$. Next, We have

$$\mathsf{e}(\vec{\mathbf{G}},\vec{\mathbf{\Theta}}''') = \mathsf{e}(\vec{\mathbf{G}},\vec{L}^{\circ\vec{\mathbf{g}}}) = \mathsf{e}(\vec{\mathbf{G}},\vec{L}) \tag{147}$$

It is evident that $\vec{\mathbf{g}} = \vec{1}$. Then, we obtain

$$\mathsf{e}(\vec{\mathbf{G}},\vec{\mathbf{L}})^{\vec{\mathbf{h}}} = \mathsf{e}(P^{\mathsf{sk}'_{n+1}}\cdot R^{\mathsf{r}'},L) \tag{148}$$

$$\mathsf{e}(\vec{\mathbf{G}},\vec{\mathbf{H}})^{\vec{\mathbf{h}}} = B_{n+1}\cdot \mathtt{Q}^{-\mathsf{r}_B} \tag{149}$$

For the Dory proof for Eqn (96) of σ_{n+1}, we can extract

$$\begin{cases} \mathsf{e}(\vec{\mathbf{\Omega}}'',\vec{\mathbf{H}}) = B_{n+1}\cdot \mathtt{Q}^{-\mathsf{r}_B} \\ \mathsf{e}(\vec{\mathbf{G}},\vec{\mathbf{\Theta}}'') = B_{n+1}\cdot \mathtt{Q}^{-\mathsf{r}_B} \\ \mathsf{e}(\vec{\mathbf{\Omega}}'',\vec{\mathbf{\Theta}}'') = B_{n+1}\cdot \mathtt{Q}^{-\mathsf{r}_B} \end{cases} \tag{150}$$

Let $\vec{\mathbf{\Theta}}'' = \vec{H}^{\circ\vec{\mathbf{e}}}$ and $\vec{\mathbf{\Omega}}'' = \vec{\mathbf{G}}^{\circ\vec{\mathbf{f}}}$. We obtain

$$\begin{cases} \mathsf{e}(\vec{\mathbf{G}},\vec{\mathbf{H}})^{\vec{\mathbf{e}}} = B_{n+1}\cdot \mathtt{Q}^{-\mathsf{r}_B} \\ \mathsf{e}(\vec{\mathbf{G}},\vec{\mathbf{H}})^{\vec{\mathbf{f}}} = B_{n+1}\cdot \mathtt{Q}^{-\mathsf{r}_B} \\ \mathsf{e}(\vec{\mathbf{G}},\vec{\mathbf{H}})^{\vec{\mathbf{e}}\circ\vec{\mathbf{f}}} = B_{n+1}\cdot \mathtt{Q}^{-\mathsf{r}_B} \end{cases} \tag{151}$$

Without breaking the discrete logarithm relation among $\big(\mathsf{e}(G_i,H_i)\big)_{i\in[n]}$, Eqns (151) implies the following:

$$\vec{\mathbf{e}} = \vec{\mathbf{f}} = \vec{\mathbf{e}}\circ\vec{\mathbf{f}} \tag{152}$$

Combining Eqn (149), we obtain $\vec{\mathbf{e}} = \vec{\mathbf{h}}$. Based on Eqn (145) and Eqn (148), we obtain $\vec{\mathbf{a}} = \vec{\mathbf{h}}$. Above all, we obtain

$$\begin{cases} \mathsf{e}(\vec{\mathbf{G}},\vec{\mathbf{H}})^{\vec{\mathbf{a}}\circ\vec{\mathbf{a}}} = \mathsf{e}(\vec{\mathbf{G}},\vec{\mathbf{H}})^{\vec{\mathbf{a}}} \\ \mathsf{e}(\vec{L'},\vec{L})^{\vec{\mathbf{a}}} = \mathsf{e}(L',L) \end{cases} \tag{153}$$

Therefore, we can conclude that $\vec{\mathbf{a}}$ is a unit basis vector. Otherwise, we would obtain a non-trivial logarithm relation to make these equations hold.

Since $\vec{\mathbf{a}}$ is a unit basis vector, with the j^{th} bit as 1, other bits as 0. Then we have:

$$\mathsf{e}(\vec{\mathbf{G}},\vec{L})^{\vec{\mathbf{a}}} = \mathsf{e}(\mathbf{G}_j,\vec{L}) = \mathsf{cm}_{n+1}\cdot\hat{A}_{n+1}\cdot\mathtt{Q}^{-\mathsf{r}_{A_{n+1}}} \tag{154}$$

$\mathcal{B}$ can extract j by checking $\mathsf{e}(\mathbf{G}_j,\vec{L}) \stackrel{?}{=} \mathsf{cm}_{n+1}\cdot\hat{A}_{n+1}\cdot\mathtt{Q}^{-\mathsf{r}_{A_{n+1}}}$ for $j\in[n]$. Thanks to the CWE property of $\mathit{\Pi}_{\text{tag}}$ and $\mathit{\Pi}_{\text{sch}}$, we can obtain:

$$\mathsf{e}(P^{\mathsf{sk}'_j}\cdot R^{\mathsf{r}_j},L) = \mathsf{e}(P^{\mathsf{sk}'_{n+1}}\cdot R^{\mathsf{r}'},L) \tag{155}$$

We can rewrite it as:

$$\mathsf{e}(P^{\mathsf{sk}'_j},L)\cdot\mathsf{e}(R^{\mathsf{r}_j},L) = \mathsf{e}(P^{\mathsf{sk}'_{n+1}},L)\cdot\mathsf{e}(R^{\mathsf{r}'},L) \tag{156}$$

$$\Rightarrow \mathsf{e}(P,L^{\mathsf{sk}'_j})\cdot\mathsf{e}(R,L^{\mathsf{r}_j}) = \mathsf{e}(P,L^{\mathsf{sk}'_{n+1}})\cdot\mathsf{e}(R,L^{\mathsf{r}'}) \tag{157}$$

$$\Rightarrow \mathsf{e}(P,L^{\mathsf{sk}'_j})\cdot\mathsf{e}(R,L^{\mathsf{r}_j})\cdot\mathsf{e}(P,L^{-\mathsf{sk}'_{n+1}})\cdot\mathsf{e}(R,L^{-\mathsf{r}'}) = 1 \tag{158}$$

$$\Rightarrow \mathsf{e}(P,L^{\mathsf{sk}'_j-\mathsf{sk}'_{n+1}})\cdot\mathsf{e}(R,L^{\mathsf{r}_j-\mathsf{r}'}) = 1 \tag{159}$$

Then $\mathcal{B}$ can return $(P'_1,P'_2) = (L^{\mathsf{sk}'_j-\mathsf{sk}'_{n+1}},L^{\mathsf{r}_j-\mathsf{r}'})$ to break the DPair assumption with non negligible probability of ϵ.

Theorem 37. *Linkable ring signature protocol $\Pi_{\text{mlr.p}}$ satisfies non-sladerability.*

Assume there exist a PPT adversary $\mathcal{A}$, which can break the non-sladerability of $\Pi_{\text{mlr.p}}$ with non negligible property ϵ, then there exists an algorithm $\mathcal{B}$ that can break the DLOG assumption between $(X = \mathsf{e}(X', L), Y = \mathsf{e}(Y', L)) \in \mathbb{G}_T^2$, where $(X', Y') \xleftarrow{\$} \mathbb{G}_1^2$. in the following game that utilizes $\mathcal{A}$:

Game Setup. Given public parameters $\mathsf{pp} \leftarrow \mathtt{Setup}[1^\lambda]$. Let $P = X'$ in pp. $(\vec{\mathbf{pk}}, \vec{\mathbf{sk}}) \leftarrow \mathtt{KeyGen}[\mathsf{pp}]$. Pick $j \in [n]$ and set $\mathsf{pk}_j = Y'$. We define the following oracles for $\mathcal{A}$ when accessing $\Pi_{\text{mlr.p}}$:

- *Random Oracle $\mathcal{RO}$*: This oracle simulates $\mathtt{Hash}'[\mathsf{f}]$. When $\mathcal{RO}$ is queried with f, if f has not been queried before, it returns $Y^{\mathsf{r_f}}$, where $\mathsf{r_f} \xleftarrow{\$} \mathbb{Z}_p^*$. Otherwise, it returns $Y^{\mathsf{r_f}}$, where $\mathsf{r_f}$ is previously chosen for f. We use $\mathtt{Hash}'[\mathsf{f}]$ in Π_{tag}
- *Corruption Oracle $\mathcal{CO}$*: When $\mathcal{CO}$ is queried with $i \in [n]$, if $i = j$, it aborts. Otherwise, it returns sk_i.
- *Signing Oracle $\mathcal{SO}$*: When $\mathcal{SO}$ is queried with $(\vec{\mathbf{pk}}, i \in [n], \mathsf{m}, \mathsf{f})$, it returns a valid signature σ as follows:
  - If $i \neq j$, then it returns $\sigma \leftarrow \mathtt{Sign}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}, \mathsf{sk_i}]$ and adds σ to the set $\Sigma_{\mathcal{SO}}$.
  - If $i = j$, then it returns a simulated signature as follows. Let $\mathsf{cm} \triangleq Y \cdot Q^{\mathsf{r_{cm}}}$ and $\mathsf{tag} \triangleq Y^{\mathsf{r_f}}$. To generate a valid proof for tag, let $(\mathsf{a}', \mathsf{r}') \xleftarrow{\$} \mathbb{Z}_p^{*2}$,

$$A \triangleq \mathtt{Hash}'[\mathsf{f}]^{\mathsf{a}'} \cdot \mathsf{tag}^{-\rho}, \tag{160}$$

$$B \triangleq P^{\mathsf{a}'} \cdot X^{-\rho} \cdot Q^{\mathsf{b}' - \mathsf{r_f}\rho} \tag{161}$$

    Hence, $(A, B, \mathsf{a}', \mathsf{r}', \mathsf{tag})$ can pass the verification in Π_{tag}. Note that it is statistically indistinguishable from other valid proofs that pass the verification.
    Next, let $\vec{\mathbf{c}}$ represent a unit basis vector, such that $c_i = 0$ when $i \neq j$ and $c_j = 1$. Finally, to generate the rest of signature proof by $\Pi_{\text{mlr.p}}$. Before return the simulated signature σ, it adds σ to the set $\Sigma_{\mathcal{SO}}$.

Game. $\mathcal{B}$ tries to derive the discrete logarithm relations among X and Y. By the definition of prefix-linkability (Definition 24), $\mathcal{A}$ first produce a signature σ', a prefix f, and a message m', such that $\mathtt{Verify}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}', \sigma'] = 1$, without issuing any corruption queries. $\mathcal{B}$ aborts if $\sigma' \in \Sigma_{\mathcal{SO}}$. $\mathcal{B}$ retrieves tag' from the tag proof in σ' and it will abort if $\exists i \in [n]_{i \neq j}, \mathtt{Hash}[\mathsf{f}]^{\mathsf{sk}_i} = \mathsf{tag}'$. Note that all public keys and signatures from $\mathcal{SO}$ are statistically indistinguishable. Therefore, the probability of $\mathcal{B}$ does not abort is $\frac{1}{n}$. Next, make $\mathcal{CO}$ accessible, $\mathcal{A}$ should returns a tuple (σ'', m''), where $\mathtt{Verify}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \mathsf{m}'', \sigma''] = 1 \wedge \mathtt{Link}[\mathsf{pp}, \vec{\mathbf{pk}}, \mathsf{f}, \sigma', \sigma''] = 1$. We know we can write cm'' in σ'' as $\mathsf{e}(\mathsf{pk_i}, L) \cdot \mathsf{Q}^{r''_{cm}}$, for some $i \in [n]$, and by the CWE property of tag proof we have $\mathsf{cm}'' = e(X', L)^{\mathsf{sk}_i} \cdot \mathsf{Q}^{r''_{cm}}$ for some $i \in [n]$. Since $\mathcal{B}$ does not abort, and consider $\mathsf{tag}' \neq \mathtt{Hash}[\mathsf{f}]^{\mathsf{sk}_i} \forall i \in [n] : i \neq j$. Hence, we can deduce that $\mathsf{tag}' = \mathtt{Hash}[\mathsf{f}]^{\mathsf{sk}_j}$. Thanks to the CWE property of the tag proof, $\mathcal{B}$ is able to extract sk_j and r'_{cm}, where $X^{\mathsf{sk}_j} = Y$. Then $\mathcal{B}$ return sk_j to break the Dlog assumption with non negligible probability of $\epsilon \cdot \frac{1}{n}$.

Theorem 38. *Linkable ring signature protocol $\Pi_{\text{mlr.p}}$ satisfies unforgeability.*

By Theorem 8 in [BEHM22], if a ring signature scheme is linkable and non-slanderable, then it is also unforgeable.