9-2019

A lattice-based linkable ring signature supporting stealth addresses

Zhen LIU
Shanghai Jiaotong University

Khoa NGUYEN
Nanyang Technological University

Guomin YANG
Singapore Management University, gmyang@smu.edu.sg

Huaxiong WANG

Duncan S. WONG

Citation

LIU, Zhen; NGUYEN, Khoa; YANG, Guomin; WANG, Huaxiong; and WONG, Duncan S.. A lattice-based linkable ring signature supporting stealth addresses. (2019). *Computer Security: ESORICS 2019: 24th European Symposium on Research in Computer Security, Luxembourg, September 23-27: Proceedings*. 11735, 726-746.

Available at: https://ink.library.smu.edu.sg/sis_research/7413

A Lattice-Based Linkable Ring Signature Supporting Stealth Addresses

Zhen Liu[1(✉)], Khoa Nguyen[2], Guomin Yang[3], Huaxiong Wang[2], and Duncan S. Wong[4]

[1] Shanghai Jiao Tong University, Shanghai, China
liuzhen@sjtu.edu.cn
[2] School of Physical and Mathematical Sciences, Nanyang Technological University, Jurong East, Singapore
{khoantt,HXWang}@ntu.edu.sg
[3] University of Wollongong, Wollongong, Australia
gyang@uow.edu.au
[4] CryptoBLK and Abelian Foundation, Kowloon, China
duncanwong@cryptoblk.io

Abstract. First proposed in CryptoNote, a collection of popular privacy-centric cryptocurrencies have employed Linkable Ring Signature and a corresponding Key Derivation Mechanism (KeyDerM) for keeping the payer and payee of a transaction anonymous and unlinkable. The KeyDerM is used for generating a fresh signing key and the corresponding public key, referred to as a stealth address, for the transaction payee. The stealth address will then be used in the linkable ring signature next time when the payee spends the coin. However, in all existing works, including Monero, the privacy model only considers the two cryptographic primitives separately. In addition, to be applied to cryptocurrencies, the security and privacy models for Linkable Ring Signature should capture the situation that the public key ring of a signature may contain keys created by an adversary (referred to as adversarially-chosen-key attack), since in cryptocurrencies, it is normal for a user (adversary) to create self-paying transactions so that some maliciously created public keys can get into the system without being detected .

In this paper, we propose a new cryptographic primitive, referred to as Linkable Ring Signature Scheme with Stealth Addresses (SALRS), which comprehensively and strictly captures the security and privacy requirements of hiding the payer and payee of a transaction in cryptocurrencies, especially the adversarially-chosen-key attacks. We also propose a lattice-based SALRS construction and prove its security and privacy in the random oracle model. In other words, our construction provides

The work was supported by the National Natural Science Foundation of China (No. 61672339), the National Cryptography Development Fund (No. MMJJ20170111), the Gopalakrishnan - NTU Presidential Postdoctoral Fellowship 2018, the National Research Foundation, Prime Minister's Office, Singapore under its Strategic Capability Research Centres Funding Initiative, the Singapore Ministry of Education under Research Grant MOE2016-T2-2-014(S), and the Abelian Foundation.

K. Sako et al. (Eds.): ESORICS 2019, LNCS 11735, pp. 726–746, 2019.
https://doi.org/10.1007/978-3-030-29959-0_35

strong confidence on security and privacy in twofolds, i.e., being proved under strong models which capture the practical scenarios of cryptocurrencies, and being potentially quantum-resistant. The efficiency analysis also shows that our lattice-based SALRS scheme is practical for real implementations.

Keywords: Lattice-Based · Linkable ring signature · Stealth Address · Cryptocurrency · Privacy

1 Introduction

Conventional cryptocurrencies such as Bitcoin or Ethereum support the pseudonym level of anonymity, namely, the wallet addresses and the real identities are delinked while transactions are linked. For privacy coins, such as Monero or Zcash, one of the objectives in terms of anonymity is to keep both the payer and payee of a transaction anonymous and unlinkable.

For example, in CryptoNote [25], Linkable Ring Signature (LRS) [20] and Key Derivation Mechanism [25] (KeyDerM) are employed. When a payer, say Alice, wants to pay Bob (the payee) through a transaction, Alice uses KeyDerM to generate a derived public key DPK from Bob's master public key MPK, and uses DPK as Bob's address in the transaction. As MPK never appears, transactions involving Bob as the receiver cannot be identified. KeyDerM is also referred to as the Stealth Address (SA) [27] mechanism. When Bob wants to spend his coins on the derived public key DPK, i.e. acting as the payer of a transaction TX, he generates a linkable ring signature σ on the transaction TX (as the message) under a set (referred to as a 'ring') of derived public keys R such that $\mathsf{DPK} \in R$. Anyone can verify σ without being able to find out the actual signer is corresponding to DPK. The linkability is used for detecting any double-spending attempt, namely if two signatures are generated by Bob corresponding to DPK, they will be detected as linked as the coin corresponding to DPK is supposed to be used only once.

LRS and SA have attracted much attention recently in the community, for example, [5,8,9,11,21,22,26,28], and in cryptocurrencies, for example, Monero [24], which uses LRS and CryptoNote's KeyDerM as its underlying building blocks, and has a market capitalization valued at more than 1 billion USD [10]. However, as shown in Table 1, all the existing works [1–3,14–16,19,20,29–31] either only consider LRS or SA in the setting of standard signature schemes [11,21] rather than both of these primitives. Even in CrypotNote [25] and Monero [24], LRS and SA are both considered, but still separately rather than being analyzed under a unified security model, despite that LRS and SA are used in a tightly-coupled fashion in both CryptoNote and Monero. In particular, the signing keys and public keys used in LRS are generated by the SA mechanism. It is *not known* whether the security and privacy properties still hold when keys used by LRS are generated by the SA mechanism, while the SA mechanism does not generate keys independently.

The linkability of LRS requires that if two signatures are generated under the same key pair, these signatures can be linked publicly. Another feature of LRS, referred to as non-slanderability, requires that an adversary cannot frame a user by creating a signature that is linked to a signature of the user. Anonymity requires that for a signature with respect to ring R, no one can identify the real signer's public key out of R. When considering these security and privacy requirements of LRS, we investigate under the assumption that each key pair is generated independently. However, this is no longer the fact when LRS is used in CryptoNote or Monero as keys are generated using the SA mechanism. For SA, the master-public-key-unlinkability [21] property requires that given a derived public key and the corresponding (standard) signatures, an adversary cannot tell the master public key, from which the derived public key is generated, out of a set of known master public keys. Another requirement called derived-public-key-unlinkability [21] captures that given two derived public keys and corresponding (standard) signatures, an adversary cannot tell whether the two derived public keys are from the same master public key.

As Linkable Ring Signature and Stealth Address are used in practical scenarios, i.e., cryptocurrencies, *another concern is whether the security and privacy models, under which they are analyzed, capture the scenarios well.* In particular, in cryptocurrencies, an attacker may create some public keys maliciously and issue transactions using these public keys as payee's addresses. As long as these malicious created keys are well-formed, they will get into the blockchain as the normal ones and a user may include these malicious created keys in their rings to sign their transactions. As a result, to be practical, the security and privacy models must consider the attacks in such a scenario, which referred to as *adversarially-chosen-key attacks.* However, as shown in Table 1, the existing linkability models either do not consider the adversarially-chosen-key attacks or consider them but do not capture the application scenarios of cryptocurrencies.

1.1 Our Results

To address the above concerns, in this paper, we propose a new cryptographic primitive, named Linkable Ring Signature Scheme with Stealth Addresses (SALRS), which *comprehensively and strictly* captures the security and privacy requirements of hiding the payer and payee of a transaction in cryptocurrencies. Particularly, all the security models (namely strong unforgeability, signer-linkability, and signer-non-slanderability) and privacy models (namely signer-anonymity, master-public-key-unlinkability, and derived-public-key-unlinkability) are defined under SALRS, rather than under Linkable Ring Signature or Stealth Address separately. Also, all the models strictly capture the practical requirements of cryptocurrencies, especially the adversarially-chosen-key attacks.

We also propose a lattice-based SALRS construction and prove its security and privacy in the random oracle model. In other words, our construction provides strong confidence on security and privacy in twofolds: being proved under strong models which capture the practical scenarios of cryptocurrencies,

Table 1. Comparison with existing LRS and SA schemes

	Consider LRS and SA together	Capture adversarially chosen-key attacks in linkability model	Potentially quantum resistant
[1,29,30]	×, only LRS	×, have flaws[a]	×
[2,3,8,19,20,26,31]	×, only LRS	×	×
[14–16]	×, only LRS[b]	√[b]	×
[24,25]	×, LRS and SA separately	×	×
[32]	×, only LRS[c]	×	√
[5,9,28]	×, only LRS	×	√
[22]	×, only LRS	×, have flaws [a]	√
[11,21]	×, only SA	NA	×
this work	√	√	√

[a]The linkability models of [1,22,29,30] have flaws, as an adversary can trivially succeed, by outputting two signatures which are obtained by querying the signing oracle on two different public keys.

[b][14–16] proposed Traceable Ring Signature, which is similar to Linkable Ring Signature. The linkability model in [14–16] captures the adversarially-chosen-key attacks, but requires that all the signatures use the same ring.

[c]In [32], a key derivation mechanism for generating one-time public keys for the payees is proposed, but the derived public keys' anonymity (unlinkability to the payee's long-term key) is not considered.

and being potentially quantum-resistant. The efficiency analysis also shows that our lattice-based SALRS scheme is practical for real implementations. Table 1 shows a comparison between our results in this work and the existing works on Linkable Ring Signature and Stealth Address. It is worth noting that although lattice-based Linkable Ring Signature schemes [5,22,28,32] have been proposed recently, to the best of our knowledge, no lattice-based Stealth Address scheme has been introduced so far. Also, although some lattice-based ring signature schemes [13,18] can achieve logarithmic signature size in terms of the number of signers in the ring, these schemes are mainly of theoretical interest since they will produce much larger signatures for a normal ring size in real scenarios. In other words, our construction is the first practical and potentially quantum-resistant solution that hides the payers and payees of transactions in cryptocurrencies.

1.2 Outline

In Sect. 2 we propose and formalize the primitive Linkable Ring Signature Scheme with Stealth Addresses (SALRS), including the algorithm definitions and the security and privacy models. In Sect. 3 we propose a lattice-based SALRS construction, and prove its security and privacy in Sect. 4. The paper is concluded in Sect. 5.

2 Definitions of SALRS

In this section, we first define the SALRS system, which captures the cryptographic functionalities that a cryptocurrency needs to hide the payers and payees of the transactions. Then we formalize the security and privacy models that strictly capture the practical scenarios in cryptocurrencies.

2.1 Algorithm Definition

A Linkable Ring Signature Scheme with Stealth Addresses (SALRS) consists of the following algorithms:

- $\mathsf{Setup}(\lambda) \rightarrow \mathsf{PP}$. This is a probabilistic algorithm. On input a security parameter λ, the algorithm outputs system public parameters PP.
  The system public parameters PP *are common parameters used by all participants in the system, for example, the message space* $\mathcal{M}$*, the hash functions, etc.* In the following, λ and PP are implicit input parameters to every algorithm.
- $\mathsf{MasterKeyGen}() \rightarrow (\mathsf{MPK}, \mathsf{MSK})$. This is a probabilistic algorithm. The algorithm outputs a (master public key, master secret key) pair $(\mathsf{MPK}, \mathsf{MSK})$.
  Each user runs $\mathsf{MasterKeyGen}$ *algorithm to generate his (master public key, master secret key) pair.*
- $\mathsf{DerivedPublicKeyGen}(\mathsf{MPK}) \rightarrow \mathsf{DPK}$. This is a probabilistic algorithm. On input a master public key MPK, the algorithm outputs a derived public key DPK.
  Anyone can run this algorithm to generate a fresh derived public key from a master public key.
- $\mathsf{DerivedPublicKeyOwnerCheck}(\mathsf{DPK}, \mathsf{MPK}, \mathsf{MSK}) \rightarrow 1/0$. This is a deterministic algorithm. On input a derived public key DPK and a (master public key, master secret key) pair $(\mathsf{MPK}, \mathsf{MSK})$, the algorithm outputs a bit $b \in \{0, 1\}$, with $b = 1$ meaning that DPK is a valid derived public key generated from MPK and $b = 0$ otherwise.
  The owner of a master public key can use this algorithm to check whether a public key is derived from his master public key. In a cryptocurrency, a payee can use this algorithm to check whether he is the intended receiver of a coin on the public key.
- $\mathsf{DerivedPublicKeyPublicCheck}(\mathsf{DPK}) \rightarrow 1/0$. This is a deterministic algorithm. On input a derived public key DPK, the algorithm outputs a bit $b \in \{0, 1\}$, with $b = 1$ meaning that DPK is a well-formed derived public key and $b = 0$ otherwise.
  Anyone can use this algorithm to check whether a derived public key is well-formed. In a cryptocurrency, a payer can use this algorithm to check whether the derived public keys owned by others are well-formed so that he can use them as ring numbers for his ring signature generation.

- $\mathsf{Sign}(M, R, \mathsf{DPK}, (\mathsf{MPK}, \mathsf{MSK})) \rightarrow \sigma$. On input a message M, a ring of well-formed derived public keys $R = (\mathsf{DPK}_1, \ldots, \mathsf{DPK}_r)$[1], a derived public key $\mathsf{DPK} \in R$, and the master key pair $(\mathsf{MPK}, \mathsf{MSK})$ for DPK, the algorithm outputs a signature σ on the message M with respect to the ring R.
  The derived public keys $\mathsf{DPK}_1, \ldots, \mathsf{DPK}_r$ *may be generated from different master public keys.*
- $\mathsf{Verify}(M, R, \sigma) \rightarrow 1/0$. This is a deterministic algorithm. On input a message M, a ring of well-formed derived public keys R, and a purported signature σ on the message M with respect to the ring R, the algorithm outputs a bit $b \in \{0, 1\}$, with $b = 1$ meaning valid and $b = 0$ otherwise.
- $\mathsf{Link}(M_0, R_0, \sigma_0, M_1, R_1, \sigma_1) \rightarrow 1/0$. This is a deterministic algorithm. On input two valid signatures (M_0, R_0, σ_0), (M_1, R_1, σ_1), the algorithm outputs a bit $b \in \{0, 1\}$, with $b = 1$ meaning linked and $b = 0$ meaning unlinked.

Correctness. The scheme must satisfy the following correctness property: Let $\mathsf{PP} \leftarrow \mathsf{Setup}(\lambda)$,

- for any $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{MasterKeyGen}()$, $\mathsf{DPK} \leftarrow \mathsf{DerivedPublicKeyGen}(\mathsf{MPK})$, it holds that $\mathsf{DerivedPublicKeyOwnerCheck}(\mathsf{DPK}, \mathsf{MPK}, \mathsf{MSK}) = 1$ and $\mathsf{DerivedPublicKeyPublicCheck}(\mathsf{DPK}) = 1$.
- for any message $M \in \mathcal{M}$, any ring of well-formed derived public keys R, and any $\mathsf{DPK}_s \in R$ such that $\mathsf{DerivedPublicKeyOwnerCheck}(\mathsf{DPK}_s, \mathsf{MPK}, \mathsf{MSK}) = 1$ for some master key $(\mathsf{MPK}, \mathsf{MSK})$, it holds that $\mathsf{Verify}(M, R, \mathsf{Sign}(M, R, \mathsf{DPK}_s, \mathsf{MPK}, \mathsf{MSK})) = 1$.
- for any messages $M_0, M_1 \in \mathcal{M}$, any well-formed derived public key rings R_0, R_1, and any $\mathsf{DPK}_{s_0} \in R_0, \mathsf{DPK}_{s_1} \in R_1$ such that $\mathsf{DerivedPublicKeyOwnerCheck}(\mathsf{DPK}_{s_i}, \mathsf{MPK}_i, \mathsf{MSK}_i) = 1$ for some master key $(\mathsf{MPK}_i, \mathsf{MSK}_i)$ $(i = 0, 1)$, let $\sigma_i \leftarrow \mathsf{Sign}(M_i, R_i, \mathsf{DPK}_{s_i}, \mathsf{MPK}_i, \mathsf{MSK}_i)$ $(i = 0, 1)$. It holds that $\mathsf{Link}(M_0, R_0, \sigma_0, M_1, R_1, \sigma_1) = 1$ if $\mathsf{DPK}_{s_0} = \mathsf{DPK}_{s_1}$, and $\Pr[\mathsf{Link}(M_0, R_0, \sigma_0, M_1, R_1, \sigma_1) = 0] \geq 1 - negl(\lambda)$ if $\mathsf{DPK}_{s_0} \neq \mathsf{DPK}_{s_1}$, where $negl$ is a negligible function.

Remark: Note that it is open on whether the Sign algorithm is probabilistic or deterministic, which may depend on the concrete constructions.

2.2 Security and Privacy Models of SALRS

Below we define the security and privacy for SALRS. The security includes unforgeability, signer-linkability, and signer-non-slanderability, while the privacy includes signer-anonymity, master-public-key-unlinkability and derived-public-key-unlinkability. Unforgeability captures that only the user knowing the secret key for some public key in a ring can generate a valid signature with respect

[1] Below, we regard the public key ring as an ordered set, namely, it consists of a set of public keys, and when it is used in Sign and Verify algorithms, the public keys are ordered and each one has an index.

to the ring. Signer-linkability captures that with respect to *one* derived public key, if the key owner generates two or multiple valid signatures, these signatures will be detected to be linked, and this captures the security requirement of preventing double-spending in cryptocurrencies. Signer-non-slanderability captures that no one can frame other users by creating a signature that is linked to a signature of the target user. Signer-anonymity captures that given a valid signature with respect to a ring of derived public keys, no one can identify the signer's derived public key out of the ring. Master-public-key-unlinkability captures that given a derived public key and the corresponding signatures, no one can tell which master public key, out of a set of known master public keys, is the one from which it was derived. Derived-public-key-unlinkability captures that given two derived public keys and the corresponding signatures, no one can tell whether they are derived from the same master public key. Signer-anonymity captures the privacy-protection requirement in cryptocurrency of hiding the payer, while master-public-key-unlinkability and derived-public-key-unlinkability captures the privacy-protection requirements of hiding the payee and cutting the link between the payees of different transactions, respectively.

With these security and privacy models, SALRS captures the security and privacy-protection requirements of cryptocurrencies in the most practical setting. Especially, the rings are allowed to contain the derived public keys that an adversary generated from his own master public keys. This reflects the situations in practice that, an attacker may generate some derived public keys from his own master public keys, and issue transactions among these keys, attempting to launch some attacks, such as double-spending, or to compromise other users' security and/or privacy. On the other side, we show that signer-linkability and signer-non-slanderability together implies unforgeability, and master-public-key-unlinkability implies derived-public-key-unlinkability. Thus, for a SALRS construction, we only needs to focus on its signer-linkability, signer-non-slanderability, signer-anonymity, and master-public-key-unlinkability.

Definition 1 (Strong Unforgeability). *A SALRS scheme is strongly unforgeable if for any probabilistic polynomial time (PPT) adversary $\mathcal{A}$ and for any polynomial $n(\cdot)$, the advantage of $\mathcal{A}$ in the following game $\mathsf{Game}_{\mathsf{euf}}$, denoted by $Adv_{\mathcal{A}}^{euf}$, is negligible.*

1. **Setup.** $\mathsf{PP} \leftarrow \mathsf{Setup}(\lambda;\omega)$ *is run, where ω is the randomness used in* $\mathsf{Setup}()$. PP *and ω are given to $\mathcal{A}$.*
   $\{(\mathsf{MPK}_i, \mathsf{MSK}_i) \leftarrow \mathsf{MasterKeyGen}()\}_{i=1}^{n(\lambda)}$ *are run and* $\{\mathsf{MPK}_i\}_{i=1}^{n(\lambda)}$ *are given to $\mathcal{A}$.*
   An empty set $L_{dpk} = \emptyset$ is initialized, which will be used to store the valid derived public keys derived from the target master public keys. Note that L_{dpk} captures the scenarios that the valid derived public keys are stored on the blockchain and are publicly accessible.
   Note that giving to $\mathcal{A}$ the randomness ω, which is used by the Setup algorithm, implies the setup is public. This is to capture that the security does not rely on a trusted setup which may incur concerns on the existing of trapdoors.

2. **Probing Phase.** $\mathcal{A}$ *can adaptively query the following oracles:*
   - *Derived Public Key Adding Oracle* $\mathsf{ODPKAdd}(\cdot,\cdot)$*:*
     On input a derived public key DPK *and a master public key* MPK_i*, this oracle returns* $b \leftarrow \mathsf{DerivedPublicKeyOwnerCheck}(\mathsf{DPK}, \mathsf{MPK}_i, \mathsf{MSK}_i)$ *to* $\mathcal{A}$*. If* $b = 1$*, set* $L_{dpk} = L_{dpk} \cup \{\mathsf{DPK}\}$*.*
     This captures that $\mathcal{A}$ can try and test whether the derived public keys generated by him are accepted by the owner of the corresponding master public key.
   - *Signing Oracle* $\mathsf{OSign}(\cdot,\cdot,\cdot)$*:*
     On input a message $M \in \mathcal{M}$*, a ring of well-formed derived public keys* R*, and a derived public key* $\mathsf{DPK} \in R \cap L_{dpk}$*, this oracle returns* $\sigma \leftarrow \mathsf{Sign}(M, R, \mathsf{DPK}, \mathsf{MPK}_i, \mathsf{MSK}_i)$ *to* $\mathcal{A}$*, where* $(\mathsf{MPK}_i, \mathsf{MSK}_i)$ *is the master key pair for* DPK*.*
     Note that it only requires that the derived public key DPK is in L_{dpk}, i.e., the attacking targets for which the master secret keys are unknown to the adversary, without requiring $R \subseteq L_{dpk}$. This captures that $\mathcal{A}$ can obtain the signatures for messages, derived public key ring, and derived public key of its choice, where **the ring may contain deprived public keys which are created by the adversary even from the master public keys which are also created by the adversary (referred to as** adversarially-chosen-key attack**)**.
3. **Output Phase**. $\mathcal{A}$ *outputs a message* $M^* \in \mathcal{M}$*, a ring of well-formed derived public keys* R^**, and a signature* σ^**.*

Let $S_{so} = \{(M, R, \mathsf{DPK}, \sigma)\}$ *be the query-answer tuples for* $\mathsf{OSign}(\cdot,\cdot,\cdot)$*.* $\mathcal{A}$ *succeeds if (1)* $\mathsf{Verify}(M^*, R^*, \sigma^*) = 1$*, and (2)* $R^* \subseteq L_{dpk}$*, and (3)* $(M^*, R^*, ?, \sigma^*) \notin S_{so}$*, where '?' means wildcard, i.e.* (M^*, R^*, σ^*) *is not a (message, derived public key ring, signature) tuple obtained by querying* $\mathsf{OSign}(\cdot,\cdot,\cdot)$*. The advantage of* $\mathcal{A}$ *is* $Adv_{\mathcal{A}}^{euf} = \Pr[\mathcal{A}\ succeeds]$*.*

Remark: In the above model, as the adversarially-chosen-key attacks are considered, i.e., the adversary is allowed to specify the derived public key ring to contain well-formed derived public keys generated from the master public keys created by himself, it is not necessary to provide an oracle of corrupting the master secret keys in $\{\mathsf{MSK}\}_{\mathsf{i}=1}^{\mathsf{n}(\lambda)}$. The situations for the following models are similar.

Definition 2 (Signer-linkability). *A SALRS scheme is signer-linkable if for any PPT adversary* $\mathcal{A}$*, the advantage of* $\mathcal{A}$ *in the following game* $\mathsf{Game}_{\mathsf{snlink}}$*, denoted by* $Adv_{\mathcal{A}}^{snlink}$*, is negligible.*

1. **Setup.** $\mathsf{PP} \leftarrow \mathsf{Setup}(\lambda; \omega)$ *is run, where* ω *is the randomness used in* $\mathsf{Setup}()$*.* PP *and* ω *are given to* $\mathcal{A}$*.*
2. **Output Phase**. $\mathcal{A}$ *outputs* $k(\geq 2)$ *(message, ring of well-formed derived public keys, signature) tuples* $(M_i^*, R_i^*, \sigma_i^*)$ $(i = 1, \ldots, k)$*.*

$\mathcal{A}$ succeeds if (1) $\mathsf{Verify}(M_i^*, R_i^*, \sigma_i^*) = 1$ $(i = 1, 2, \ldots, k)$, *and (2)* $\mathsf{Link}(M_i^*, R_i^*, \sigma_i^*, M_j^*, R_j^*, \sigma_j^*) = 0$ $\forall i, j \in [1, k]$ *s.t.* $i \neq j$, *and (3)* $|\cup_{i=1}^{k} R_i^*| < k$. *The advantage of $\mathcal{A}$ is* $Adv_{\mathcal{A}}^{snlink} = \Pr[\mathcal{A}\ succeeds]$.

Remark: Note that the adversary's target is to attack the linkability property of the system, rather than attacking other users, thus we do not need to consider the target master public keys or derived public keys. Also, as the adversary is allowed to create the master public keys and derived public keys of its choice, we do not need to consider the signing oracles, corruption oracles, etc.

Definition 3 (Signer-non-slanderability). *A SALRS scheme is signer-non-slanderable if for any PPT adversary $\mathcal{A}$ and for any polynomial $n(\cdot)$, the advantage of $\mathcal{A}$ in the following game* $\mathsf{Game}_{\mathsf{snnsl}}$, *denoted by* $Adv_{\mathcal{A}}^{snnsl}$, *is negligible.*

1. **Setup.** *Same as that of* $\mathsf{Game}_{\mathsf{euf}}$ *in Def. 1.*
2. **Probing Phase.** *Same as that of* $\mathsf{Game}_{\mathsf{euf}}$ *in Def. 1.*
3. **Output Phase**. *$\mathcal{A}$ outputs two (message, ring of well-formed derived public keys, signature) tuples* $(\hat{M}, \hat{R}, \hat{\sigma})$ *and* (M^*, R^*, σ^*).

Let $S_{so} = \{(M, R, \mathsf{DPK}, \sigma)\}$ *be the query-answer tuples for* $\mathsf{OSign}(\cdot, \cdot, \cdot)$. *$\mathcal{A}$ succeeds if (1)* $\mathsf{Verify}(M^*, R^*, \sigma^*) = 1$, *and (2)* $(\hat{M}, \hat{R}, \hat{\mathsf{DPK}}, \hat{\sigma}) \in S_{so}$ *for some* $\hat{\mathsf{DPK}} \in \hat{R} \cap L_{dpk}$, *and (3)* $(M^*, R^*, \hat{\mathsf{DPK}}, \sigma^*) \notin S_{so}$, *and (4)* $\mathsf{Link}(M^*, R^*, \sigma*, \hat{M}, \hat{R}, \hat{\sigma}) = 1$. *The advantage of $\mathcal{A}$ is* $Adv_{\mathcal{A}}^{snnsl} = \Pr[\mathcal{A}\ succeeds]$.

Definition 4 (Signer-Anonymity). *A SALRS scheme is signer-anonymous if for any PPT adversary $\mathcal{A}$ and for any polynomial $n(\cdot)$, the advantage of $\mathcal{A}$ in the following game* $\mathsf{Game}_{\mathsf{snano}}$, *denoted by* $Adv_{\mathcal{A}}^{snano}$, *is negligible.*

1. **Setup.** *Same as that of* $\mathsf{Game}_{\mathsf{euf}}$ *in Def. 1.*
2. **Probing Phase 1.** *Same as the* Probing Phase *of* $\mathsf{Game}_{\mathsf{euf}}$ *in Def. 1.*
3. **Challenge Phase**. *$\mathcal{A}$ outputs a message M^*, a ring of well-formed derived public keys R^*, and two distinct indices $1 \leq i_0, i_1 \leq n(\lambda)$, such that*
   (1) $\mathsf{DPK}_{i_0}, \mathsf{DPK}_{i_1} \in R^* \cap L_{dpk}$, *and*
   (2) none of $\mathsf{OSign}(\cdot, \cdot, \mathsf{DPK}_{i_0})$, $\mathsf{OSign}(\cdot, \cdot, \mathsf{DPK}_{i_1})$ *was queried. A random bit $b \in \{0, 1\}$ is chosen, and $\mathcal{A}$ is given the signature* $\sigma \leftarrow \mathsf{Sign}(M^*, R^*, \mathsf{DPK}_{i_b}, \mathsf{MPK}, \mathsf{MSK})$, *where* $(\mathsf{MPK}, \mathsf{MSK})$ *is the master key pair for* DPK_{i_b}.
4. **Probing Phase 2**. *Same as the* **Probing Phase 1**, *but with the restriction that none of* $\mathsf{OSign}(\cdot, \cdot, \mathsf{DPK}_{i_0})$, $\mathsf{OSign}(\cdot, \cdot, \mathsf{DPK}_{i_1})$ *is queried.*
5. **Output Phase**. *$\mathcal{A}$ outputs a bit b' as its guess to b.*

The advantage of $\mathcal{A}$ is $Adv_{\mathcal{A}}^{snano} = |\Pr[b' = b] - \frac{1}{2}|$.

Definition 5 (Master-Public-Key-Unlinkability). *A SALRS scheme is Master Public-Key-Unlinkable if for any PPT adversary $\mathcal{A}$ and for any polynomial $n(\cdot)$, the advantage of $\mathcal{A}$ in the following game* $\mathsf{Game}_{\mathsf{mpkunl}}$, *denoted by* $Adv_{\mathcal{A}}^{mpkunl}$, *is negligible.*

1. **Setup.** *Same as that of* $\mathsf{Game}_{\mathsf{euf}}$ *in Def. 1.*
2. **Probing Phase 1.** *Same as the* Probing Phase *of* $\mathsf{Game}_{\mathsf{euf}}$ *in Def. 1.*
3. **Challenge.** $\mathcal{A}$ *outputs two distinct indices* $1 \leq i_0, i_1 \leq n(\lambda)$. *A random bit* $b \in \{0,1\}$ *is chosen, and* $\mathsf{DPK}^* \leftarrow \mathsf{DerivedPublicKeyGen}(\mathsf{MPK}_{i_b})$ *is given to* $\mathcal{A}$. *Set* $L_{dpk} = L_{dpk} \cup \{\mathsf{DPK}^*\}$.
4. **Probing Phase 2**. *Same as* **Phase 1**, *except that* $\mathsf{ODPKAdd}(\mathsf{DPK}^*, \mathsf{MPK}_{i_j})$ *(for* $j \in \{0,1\}$*) cannot be queried.*
5. **Guess**. $\mathcal{A}$ *outputs a bit* $b' \in \{0,1\}$ *as its guess to* b.

The advantage of $\mathcal{A}$ *is* $Adv_{\mathcal{A}}^{mpkunl} = |\Pr[b' = b] - \frac{1}{2}|$.

Remark: Note that $\mathsf{OSign}(\cdot,\cdot,\mathsf{DPK}^*)$ can be queried. This captures that neither the derived public key or the signatures leak the corresponding master public key.

Definition 6 (Derived-Public-Key-Unlinkability). *A SALRS scheme is Derived Public-Key-Unlinkable if for any PPT adversary* $\mathcal{A}$ *and for any polynomial* $n(\cdot)$*, the advantage of* $\mathcal{A}$ *in the following game* $\mathsf{Game}_{\mathsf{dpkunl}}$*, denoted by* $Adv_{\mathcal{A}}^{dpkunl}$*, is negligible.*

1. **Setup.** *Same as that of* $\mathsf{Game}_{\mathsf{euf}}$ *in Def. 1.*
2. **Probing Phase 1.** *Same as the* Probing Phase *of* $\mathsf{Game}_{\mathsf{euf}}$ *in Def. 1.*
3. **Challenge.** $\mathcal{A}$ *outputs two distinct indices* $1 \leq i_0, i_1 \leq n(\lambda)$.
   A random bit $c \in \{0,1\}$ *is chosen.*
   Compute $\mathsf{DPK}_0^* \leftarrow \mathsf{DerivedPublicKeyGen}(\mathsf{MPK}_{i_c})$.
   A random bit $b \in \{0,1\}$ *is chosen.*
   If $b = 0$*, compute* $\mathsf{DPK}_1^* \leftarrow \mathsf{DerivedPublicKeyGen}(\mathsf{MPK}_{i_c})$,
   otherwise, compute $\mathsf{DPK}_1^* \leftarrow \mathsf{DerivedPublicKeyGen}(\mathsf{MPK}_{i_{1-c}})$.
   $(\mathsf{DPK}_0^*, \mathsf{DPK}_1^*)$ *are given to* $\mathcal{A}$. *Set* $L_{dpk} = L_{dpk} \cup \{\mathsf{DPK}_0^*, \mathsf{DPK}_1^*\}$.
4. **Probing Phase 2**. *Same as* **Probing Phase 1**, *except that* $\mathsf{ODPKAdd}(\mathsf{DPK}_j^*, \mathsf{MPK}_{i_k})$ *(for* $j,k \in \{0,1\}$*) can be queried on at most one* $j \in \{0,1\}$.
5. **Guess**. $\mathcal{A}$ *outputs a bit* $b' \in \{0,1\}$ *as its guess to* b*, i.e., guess whether* DPK_0^* *and* DPK_1^* *are from the same master public key.*

The advantage of $\mathcal{A}$ *is* $Adv_{\mathcal{A}}^{dpkunl} = |\Pr[b' = b] - \frac{1}{2}|$.

Remark: Note that $\mathsf{OSign}(\cdot,\cdot,\mathsf{DPK}_j^*)$ (for $j = 0,1$) can be queried, and this captures that neither the derived public keys or the corresponding signatures leak whether they are from the same master public key.

As the above models captures the security and privacy requirements that the practice imposes on SALRS, the following two theorems show that *for a SALRS scheme, we only need to consider its signer-linkability, signer-non-slanderability, signer-anonymity, and master-public-key-unlinkability.*

Theorem 1. *If a SALRS scheme is signer-linkable and siner-non-slanderable, then it is strongly unforgeable.*

Proof. The proof resembles that for a similar conclusion in the setting of Traceable Ring Signature in [16]. We give the proof in Appendix A.

Theorem 2. *If a SALRS scheme is master-public-key-unlinkable, then it is derived-public-key-unlinkable.*

Proof. Observe $\mathsf{Game}_{\mathsf{mpkunl}}$ and $\mathsf{Game}_{\mathsf{dpkunl}}$, it is easy to see that, if there exists an adversary $\mathcal{A}$ that wins $\mathsf{Game}_{\mathsf{dpknul}}$ with non-negligible advantage, we can construct an algorithm $\mathcal{B}$ that interacts with $\mathcal{A}$ for game $\mathsf{Game}_{\mathsf{dpknul}}$, and makes use of $\mathcal{A}$'s output to win $\mathsf{Game}_{\mathsf{mpkunl}}$ with non-negligible advantage. We defer the proof details to the full version.

3 Our Construction

In this section, we first present some preliminaries in Sect. 3.1, including the concept of key-privacy in Key-Encapsulation Mechanism (KEM), which we will use as a building block for our SALRS construction, and some background of lattice. Then we propose a lattice-based SALRS construction in Sect. 3.2 and give the concrete parameters and building blocks in Sect. 3.3.

3.1 Preliminaries

3.1.1 Key-Privacy in KEM

Our construction will use KEM as a building block, but requires the underlying KEM to have an additional property, referred to as key-privacy, which asks that an adversary in possession of a ciphertext not be able to tell which specific public key, out of a set of known public keys, is the one under which the ciphertext was created, meaning the receiver is anonymous from the point of view of the adversary. It is worth mentioning that Bellare et al. [6] considered a similar concept on the setting of Public Key Encryption (PKE). Below we extend the usual KEM and formalize the concept of KEM with key-privacy.

Syntax. To capture the practice better, we augment the usual formalization of KEM to cover the cases that users may share some fixed "global" information.

A *key-encapsulation mechanism* (KEM) scheme is a tuple of probabilistic polynomial-time algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encaps}, \mathsf{Decaps})$ such that:

- $\mathsf{Setup}(\lambda) \to \mathsf{GP}$. On input a security parameter λ, the algorithm outputs system global parameters GP.
  The system global parameters GP *are common parameters used by all participants in the system, which may be just the security parameter* λ*, or include some additional information, for example, the key space, the ciphertext space, the hash functions, etc. As we will consider the key-privacy, here we require that* GP *include the key space* $\mathcal{K}$ *and ciphertext space* $\mathcal{C}$*.*
- $\mathsf{KeyGen}(\mathsf{GP}) \to (\mathsf{PK}, \mathsf{SK})$. This is a probabilistic algorithm. On input GP, the algorithm outputs a (public key, secret key) pair $(\mathsf{PK}, \mathsf{SK})$.

- $\mathsf{Encaps}(\mathsf{GP}, \mathsf{PK}) \rightarrow (C, \kappa)$. This is a probabilistic algorithm. On input GP and a public key PK, the algorithm outputs a ciphertext $C \in \mathcal{C}$ and a key $\kappa \in \mathcal{K}$.
- $\mathsf{Decaps}(\mathsf{GP}, C, \mathsf{PK}, \mathsf{SK}) \rightarrow \kappa/\perp$. This is a deterministic algorithm. On input GP, a ciphertext $C \in \mathcal{C}$, and a (public key, secret key) pair $(\mathsf{PK}, \mathsf{SK})$, the algorithm outputs a key $\kappa \in \mathcal{K}$ or a special symbol $\perp$ to indicate rejection.

Correctness. It is required that with all but negligible probability over $\mathsf{GP} \leftarrow \mathsf{Setup}(1^\lambda)$, $(\mathsf{PK}, \mathsf{SK}) \leftarrow \mathsf{KeyGen}(\mathsf{GP})$, and the random coins of Encaps, if $\mathsf{Encaps}(\mathsf{GP}, \mathsf{PK})$ outputs (C, κ), then $\mathsf{Decaps}(\mathsf{GP}, C, \mathsf{PK}, \mathsf{SK})$ outputs κ.

Security and Key-Privacy. Below we formalize the security and key-privacy models.

Definition 7 (CCA-Security of KEM). *A KEM scheme is CCA-secure if for any PPT adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following game $\mathsf{Game}_{\mathsf{ccasec}}$, denoted by $Adv_{\mathcal{A}}^{ccasec}$, is negligible.*

1. **Setup.** $\mathsf{GP} \leftarrow \mathsf{Setup}(\lambda; \omega)$ *is run, where ω is the randomness used in* $\mathsf{Setup}()$. GP *and ω are given to $\mathcal{A}$.* $(\mathsf{PK}, \mathsf{SK}) \leftarrow \mathsf{KeyGen}(\mathsf{GP})$ *is run and* PK *is given to $\mathcal{A}$.*
   Note that giving to $\mathcal{A}$ the randomness ω, which is used by the Setup algorithm, implies the setup is public. This is to capture that the security does not rely on a trusted setup which may incur the concerns on the existing of trapdoors.
2. **Challenge Phase.** $(C^*, \kappa) \leftarrow \mathsf{Encaps}(\mathsf{GP}, \mathsf{PK})$ *is run. A random bit b is chosen. If $b = 0$, set $\kappa^* := \kappa$, otherwise choose a uniformly random $\kappa^* \xleftarrow{R} \mathcal{K}$. $\mathcal{A}$ is given (C^*, κ^*).*
3. **Probing Phase.** *$\mathcal{A}$ can adaptively query an oracle* $\mathsf{ODecaps}(\cdot)$*, which takes a ciphertext $C \in \mathcal{C}$ and returns $\kappa \leftarrow \mathsf{Decaps}(\mathsf{GP}, C, \mathsf{PK}, \mathsf{SK})$ to $\mathcal{A}$, with the restriction that $\mathcal{A}$ cannot query* $\mathsf{ODecaps}(\cdot)$ *on the challenge C^*.*
4. **Output Phase**. *$\mathcal{A}$ outputs a bit b'.*

The advantage of $\mathcal{A}$ is $Adv_{\mathcal{A}}^{ccasec} = |\Pr[b' = b] - \frac{1}{2}|$.

Definition 8 (CCA-Key-Indistinguishability of KEM). *A KEM scheme is CCA-key-indistinguishable if for any PPT adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following game $\mathsf{Game}_{\mathsf{ccaki}}$, denoted by $Adv_{\mathcal{A}}^{ccaki}$, is negligible.*

1. **Setup.** *Same as that of* $\mathsf{Game}_{\mathsf{ccasec}}$.
2. **Challenge Phase.** $(C, \kappa^*) \leftarrow \mathsf{Encaps}(\mathsf{GP}, \mathsf{PK})$ *is run. A random bit b is chosen. If $b = 0$, set $C^* := C$, otherwise choose a uniformly random $C^* \xleftarrow{R} \mathcal{C}$. $\mathcal{A}$ is given (C^*, κ^*).*
3. **Probing Phase.** *Same as that of* $\mathsf{Game}_{\mathsf{ccasec}}$.
4. **Output Phase**. *$\mathcal{A}$ outputs a bit b'.*

The advantage of $\mathcal{A}$ is $Adv_{\mathcal{A}}^{ccaki} = |\Pr[b' = b] - \frac{1}{2}|$.

3.1.2 Lattice Background

Rings, Norms and Invertible Ring Elements. Let q be an even (resp. odd) positive integer, and denote by $\mathbb{Z}_q$ the integers modulo q, which will be represented in the range $(-\frac{q}{2}, \frac{q}{2}]$ (resp. $[-\frac{q-1}{2}, \frac{q-1}{2}]$). Let n be an positive integer, and let R and R_q be the rings $\mathbb{Z}[X]/(X^n+1)$ and $\mathbb{Z}_q[X]/(X^n+1)$, respectively. For $w = a_0 + a_1 X + \ldots + a_{n-1} X^{n-1} \in R$, define the l_∞, l_1 and l_2 norms of w as follows:

$$\|w\|_\infty = \max_i |a_i|, \qquad \|w\|_1 = \sum_i |a_i|, \qquad \|w\|_2 = \sqrt{|a_0|^2 + \ldots + |a_{n-1}|^2}.$$

Similarly, for $\mathbf{w} = (w_1, \ldots, w_k) \in R^k$, define:

$$\|\mathbf{w}\|_\infty = \max_i \|w_i\|_\infty, \qquad \|\mathbf{w}\|_1 = \sum_i \|w_i\|_1, \qquad \|\mathbf{w}\|_2 = \sqrt{\|w_1\|_2^2 + \ldots + \|w_k\|_2^2}.$$

Let S_η denote the set of all elements $w \in R$ such that $\|w\|_\infty \leq \eta$. As shown in [23], for prime $q > 2^{20}$ such that $q = 17 \bmod 32$, and for $\eta < \frac{1}{\sqrt{8}} \cdot q^{1/8}$, all non-zero elements of S_η are invertible in R_q.

Let $\mathbf{B}_\theta$ denote the set of all elements in R_q such that have θ coefficients that are either -1 or 1 and the rest are 0. Again, for prime $q > 2^{20}$ such that $q = 17 \bmod 32$, all elements of $\mathbf{B}_\theta$ are invertible and the difference of any two distinct elements from $\mathbf{B}_\theta$ is also invertible in R_q.

(Inhomogeneous) Module-SIS. The Inhomogeneous Module-SIS problem with parameters (n, q, k, ℓ, β) consists in finding $\mathbf{x} \in R^{k+\ell}$ such that $\|\mathbf{x}\|_2 \leq \beta$ and $[\mathbf{A} \mid \mathbf{I}] \cdot \mathbf{x} = \mathbf{t}$, for uniformly random $\mathbf{A} \in R_q^{k \times \ell}$, $\mathbf{t} \in R_q^k$ and $k \times k$ identity matrix $\mathbf{I}$. The problem can be adapted straightforwardly into its infinity-norm version, where $\mathbf{x}$ must satisfy $\|\mathbf{x}\|_\infty \leq \beta$. The homogeneous version is defined with $\mathbf{t} = \mathbf{0}$ and $\mathbf{x} \neq \mathbf{0}$.

Module-LWE. The Module-LWE problem with parameters (n, q, k, ℓ, η) is as follows. Let $\mathbf{A} \in R_q^{k \times \ell}$ be a uniformly random matrix. Let $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \in R_q^k$, where $\mathbf{s} \in S_\eta^\ell$, $\mathbf{e} \in S_\eta^k$ have entries chosen according to some distribution over S_η (e.g., the uniform distribution or a Gaussian distribution). The search variant of Module-LWE asks to recover $\mathbf{s}$ given $(\mathbf{A}, \mathbf{b})$. The decision variant (decision-Module-LWE) asks to distinguish $(\mathbf{A}, \mathbf{b})$ from a uniformly random pair over $R_q^{k \times \ell} \times R_q^k$. In this paper, similar to [5], we use a transformed version of the decision-Module-LWE problem, which is to distinguish $(\mathbf{A}, \mathbf{A}\mathbf{s})$ from $(\mathbf{A}, \mathbf{r})$ where $\mathbf{A} \leftarrow R_q^{k \times l}$, $\mathbf{s} \leftarrow S_\eta^l$ and $\mathbf{r} \leftarrow R_q^k$.

As shown in [17], the Module-SIS and Module-LWE problems enjoy worst-case to average-case reductions from hard problems in module lattices. Concrete parameters of these problems that provide high post-quantum security against the best known attacks are given in Dilithium [12] and Kyber [7].

3.2 Construction

– $\mathsf{Setup}(1^\lambda) \to \mathsf{PP}$. On input a security parameter λ, the algorithm sets the parameters $n, q, k, l, m,\ \eta, \gamma, \theta$ as specified in Sect. 3.3 below. Let Π_{kem} be a lattice-based KEM scheme which is CCA-secure and CCA-key indistinguishable, and let $\mathcal{C}_{kem}$ and $\mathcal{K}_{kem}$ denote Π_{kem}'s ciphertext space and key space, respectively. Let $H_{\mathsf{A}} : \{0,1\}^* \mapsto R_q^{k\times l}$, $\mathsf{ExpandV} : \mathcal{K}_{kem} \mapsto S_\eta^l$, $H_\theta : \{0,1\}^* \mapsto \mathbf{B}_\theta$, and $H_m : R_q^k \mapsto R_q^{m\times l}$ be functions that will be viewed as random oracles in the analyses. The algorithm does:
  1. Choose a random string $cstr \in \{0,1\}^*$, and set $\mathbf{A} := H_{\mathsf{A}}(cstr)$.
  2. Run $\mathsf{GP}_{kem} \leftarrow \Pi_{kem}.\mathsf{Setup}(1^\lambda; \omega)$, where ω is the randomness used in $\Pi_{kem}.\mathsf{Setup}()$.
  3. Output the public parameters

$$\mathsf{PP} = \big(n, q, k, l, m, \eta, \gamma, \theta, (H_{\mathsf{A}}, cstr, \mathbf{A}), (\Pi_{kem}, \omega, \mathsf{GP}_{kem}), \mathsf{ExpandV}, H_\theta, H_m\big).$$

  Note that including $(H_{\mathsf{A}}, cstr)$ and ω in PP is to ensure that no one knows any trapdoor for matrix $\mathbf{A}$ and GP_{kem} respectively.
  In the following, PP are implicit input parameters to every algorithm.

– $\mathsf{MasterKeyGen}() \to (\mathsf{MPK}, \mathsf{MSK})$. On input the implicit inputs, namely, the public parameters PP, the algorithm does:
  1. Run $(\mathsf{PK}_{kem}, \mathsf{SK}_{kem}) \leftarrow \Pi_{kem}.\mathsf{KeyGen}(\mathsf{GP}_{kem})$.
  2. Choose a uniformly random $\mathbf{s} \xleftarrow{R} S_\eta^l$, and set $\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}$.
  3. Output master public key MPK and master secret key MSK

$$\mathsf{MPK} := \big(\mathsf{PK}_{kem}, \mathbf{t}\big), \quad \mathsf{MSK} := \big(\mathsf{SK}_{kem}, \mathbf{s}\big).$$

– $\mathsf{DerivedPublicKeyGen}(\mathsf{MPK}) \to \mathsf{DPK}$. On input a master public key $\mathsf{MPK} = \big(\mathsf{PK}_{kem}, \mathbf{t}\big)$, the algorithm does:
  1. Run $(C, \kappa) \leftarrow \Pi_{kem}.\mathsf{Encaps}(\mathsf{PK}_{kem})$.
  2. Set $\mathbf{s}' := \mathsf{ExpandV}(\kappa) \in S_\eta^l$, $\mathbf{t}' \leftarrow \mathbf{A}\mathbf{s}'$, and set $\hat{\mathbf{t}} \leftarrow \mathbf{t} + \mathbf{t}'$.
  3. Output a derived public key $\mathsf{DPK} := (C, \hat{\mathbf{t}})$.

– $\mathsf{DerivedPublicKeyOwnerCheck}(\mathsf{DPK}, \mathsf{MPK}, \mathsf{MSK}) \to 1/0$. On input a derived public key DPK and a (master public key, master secret key) pair $(\mathsf{MPK}, \mathsf{MSK})$ with $\mathsf{MPK} = (\mathsf{PK}_{kem}, \mathbf{t})$, and $\mathsf{MSK} = (\mathsf{SK}_{kem}, \mathbf{s})$, the algorithm does:
  1. Check whether $\mathsf{DPK} \in \mathcal{C}_{kem} \times R_q^k$ holds. If it does not hold, return 0, otherwise, parse DPK to $\mathsf{DPK} := (C, \hat{\mathbf{t}}) \in C_{kem} \times R_q^k$.
  2. Run $\kappa \leftarrow \Pi_{kem}.\mathsf{Decaps}(C, \mathsf{PK}_{kem}, \mathsf{SK}_{kem})$.
  3. Set $\mathbf{s}' := \mathsf{ExpandV}(\kappa)$ and $\mathbf{t}' \leftarrow \mathbf{A}\mathbf{s}'$.
  4. If $\hat{\mathbf{t}} \stackrel{?}{=} \mathbf{t} + \mathbf{t}'$ holds, return 1, otherwise return 0.

– $\mathsf{DerivedPublicKeyPublicCheck}(\mathsf{DPK}) \to 1/0$. On input a derived public key DPK, the algorithm checks whether $\mathsf{DPK} \in \mathcal{C}_{kem} \times R_q^k$ holds. If it holds, return 1, otherwise return 0.

– $\mathsf{Sign}(M, R, \mathsf{DPK}, (\mathsf{MPK}, \mathsf{MSK})) \rightarrow \sigma$. On input a message M, a ring of well-formed derived public keys $R = (\mathsf{DPK}_1, \ldots, \mathsf{DPK}_r)$, a derived public key $\mathsf{DPK} \in R$, and the master key pair $(\mathsf{MPK}, \mathsf{MSK})$ for DPK where $\mathsf{MPK} = (\mathsf{PK}_{kem}, \mathbf{t})$ and $\mathsf{MSK} = (\mathsf{SK}_{kem}, \mathbf{s})$, the algorithm does:
  1. For $i = 1$ to r, parse $\mathsf{DPK}_i := (C_i, \hat{\mathbf{t}}_i) \in \mathcal{C}_{kem} \times R_q^k$ and set $\mathbf{H}_i := H_m(\hat{\mathbf{t}}_i)$.
  2. Let $\bar{i}$ be the index of DPK in R, i.e. $\mathsf{DPK} = \mathsf{DPK}_{\bar{i}} = (\mathcal{C}_{\bar{i}}, \hat{\mathbf{t}}_{\bar{i}})$. Run $\kappa \leftarrow \Pi_{kem}.\mathsf{Decaps}(C_{\bar{i}}, \mathsf{PK}_{kem}, \mathsf{SK}_{kem})$. Set $\mathbf{s}'_{\bar{i}} := \mathsf{ExpandV}(\kappa)$ and $\hat{\mathbf{s}}_{\bar{i}} \leftarrow \mathbf{s} + \mathbf{s}'_{\bar{i}}$. Note that it holds that $\hat{\mathbf{t}}_{\bar{i}} = \mathbf{A}\hat{\mathbf{s}}_{\bar{i}}$.
  3. Set $\mathbf{I} \leftarrow \mathbf{H}_{\bar{i}}\hat{\mathbf{s}}_{\bar{i}}$.
  4. Choose a uniformly random $\mathbf{y} \xleftarrow{R} S_\gamma^l$.
  5. Set $\mathbf{w}_{\bar{i}} \leftarrow \mathbf{A}\mathbf{y}$, $\mathbf{v}_{\bar{i}} \leftarrow \mathbf{H}_{\bar{i}}\mathbf{y}$.
  6. For $i = \bar{i}+1, \ldots, r, 1, \ldots, \bar{i}-1$, do
     (a) Set $c_i \leftarrow H_\theta(M, R, \mathbf{w}_{i-1}, \mathbf{v}_{i-1}, \mathbf{I})$.[2]
     (b) Choose a uniformly random $\mathbf{z}_i \leftarrow S_{\gamma-2\theta\eta}^l$.
     (c) Set $\mathbf{w}_i \leftarrow \mathbf{A}\mathbf{z}_i - c_i\hat{\mathbf{t}}_i$, $\mathbf{v}_i \leftarrow \mathbf{H}_i\mathbf{z}_i - c_i\mathbf{I}$.
  7. Set $c_{\bar{i}} \leftarrow H_\theta(M, R, \mathbf{w}_{\bar{i}-1}, \mathbf{v}_{\bar{i}-1}, \mathbf{I})$.
  8. Set $\mathbf{z}_{\bar{i}} \leftarrow \mathbf{y} + c_{\bar{i}}\hat{\mathbf{s}}_{\bar{i}}$.
  9. If $\mathbf{z}_{\bar{i}} \in S_{\gamma-2\theta\eta}^l$, output $\sigma := (c_1, \{\mathbf{z}_i\}_{i=1}^r, \mathbf{I}) \in \mathbf{B}_\theta \times (S_{\gamma-2\eta\theta}^l)^r \times R_q^m$, otherwise go to Step 4.

– $\mathsf{Verify}(M, R, \sigma) \rightarrow 1/0$. On input a message M, a ring of well-formed derived public keys $R = (\mathsf{DPK}_1, \ldots, \mathsf{DPK}_r)$, and a signature $\sigma = (c_1, \{\mathbf{z}_i\}_{i=1}^r, \mathbf{I})$, the algorithm does:
  1. If $(c_1 \notin \mathbf{B}_\theta) \vee (\exists i \in \{1, \ldots, r\}\ s.t.\ \mathbf{z}_i \notin S_{\gamma-2\theta\eta}^l)$, then return 0.
  2. For $i = 1, 2, \ldots, r$, do
     (a) Parse DPK_i to $\mathsf{DPK}_i := (C_i, \hat{\mathbf{t}}_i) \in \mathcal{C}_{kem} \times R_q^k$ and set $\mathbf{H}_i := H_m(\hat{\mathbf{t}}_i)$.
     (b) Set $\mathbf{w}_i \leftarrow \mathbf{A}\mathbf{z}_i - c_i\hat{\mathbf{t}}_i, \mathbf{v}_i \leftarrow \mathbf{H}_i\mathbf{z}_i - c_i\mathbf{I}$.
     (c) Set $c_{i+1} \leftarrow H_\theta(M, R, \mathbf{w}_i, \mathbf{v}_i, \mathbf{I})$.
  3. If $c_{r+1} \stackrel{?}{=} c_1$ holds, return 1, otherwise return 0.

– $\mathsf{Link}(M_0, R_0, \sigma_0, M_1, R_1, \sigma_1) \rightarrow 1/0$. On input two valid (message, derived public key ring, signature) tuples (M_0, R_0, σ_0), (M_1, R_1, σ_1) where $\sigma_0 = (c_1^{(0)}, \{\mathbf{z}_i^{(0)}\}_{i=1}^{r_0}, \mathbf{I}^{(0)})$, $\sigma_1 = (c_1^{(1)}, \{\mathbf{z}_i^{(1)}\}_{i=1}^{r_1}, \mathbf{I}^{(1)})$, if $\mathbf{I}^{(0)} \stackrel{?}{=} \mathbf{I}^{(1)}$ holds, the algorithm returns 1, otherwise returns 0.

3.3 Correctness and Concrete Parameters

This section analyzes the correctness of the proposed lattice-based SALRS scheme, specifies the parameters achieving 128 bits of security and evaluates the efficiency of the scheme.

Correctness. We first note that, the validity and well-formedness of a derived public key DPK, as verified by algorithms $\mathsf{DerivedPublicKeyOwnerCheck}$ and $\mathsf{DerivedPublicKeyPublicCheck}$ respectively, follows directly from the construction of DPK, the correctness of the underlying KEM scheme Π_{kem} and the fact that $\hat{\mathbf{t}} = \mathbf{t} + \mathbf{A}\mathbf{s}' = \mathbf{t} + \mathbf{t}' \in R_q^k$. Next, for an honestly generated signature

[2] Note that 1 is regarded as $r+1$, i.e., $c_1 \leftarrow H_\theta(M, R, \mathbf{w}_r, \mathbf{v}_r, \mathbf{I})$.

$\sigma = (c_1, \{\mathbf{z}_i\}_{i=1}^r, \mathbf{I})$, it holds that $c_1 \in \mathbf{B}_\theta$ and $\mathbf{z}_i \in S^l_{\gamma-2\theta\eta}$ for all $i \in \{1, \ldots, r\}$. Furthermore, by construction, the value c_{r+1} computed at Step 2 of algorithm Verify satisfies $c_{r+1} = c_1$. Therefore, σ is accepted by Verify.

We next analyze the correctness of algorithm Link. Let $\sigma_0 = (c_1^{(0)}, \{\mathbf{z}_i^{(0)}\}_{i=1}^{r_0}, \mathbf{I}^{(0)})$ and $\sigma_1 = (c_1^{(1)}, \{\mathbf{z}_i^{(1)}\}_{i=1}^{r_1}, \mathbf{I}^{(1)})$ be generated by $\mathsf{Sign}(M_0, R_0, \mathsf{DPK}_0, (\mathsf{MPK}_0, \mathsf{MSK}_0))$ and $\mathsf{Sign}(M_1, R_1, \mathsf{DPK}_1, (\mathsf{MPK}_1, \mathsf{MSK}_1))$, respectively. For $i = 0, 1$, let $\mathsf{DPK}_i = (C_i, \hat{\mathbf{t}}_i)$ and note that $\mathbf{I}^{(i)} = H_m(\hat{\mathbf{t}}_i)\hat{\mathbf{s}}_i$, where $\hat{\mathbf{s}}_i = \mathbf{s}_i + \mathbf{s}'_i$ and $\mathbf{s}_i, \mathbf{s}'_i$ are generated as specified by the scheme. Note that, if $\mathsf{DPK}_0 = \mathsf{DPK}_1$, then we have $\hat{\mathbf{s}}_0 = \hat{\mathbf{s}}_1$ and thus, $\mathbf{I}^{(0)} = \mathbf{I}^{(1)}$. In this case, algorithm Link outputs 1.

In the case $\mathsf{DPK}_0 \neq \mathsf{DPK}_1$, we will demonstrate that, with overwhelming probability, algorithm Link outputs 0. Indeed, if $\hat{\mathbf{t}}_0 \neq \hat{\mathbf{t}}_1$, then $H_m(\hat{\mathbf{t}}_0), H_m(\hat{\mathbf{t}}_1)$ are uniformly random and distinct, $\hat{\mathbf{s}}_0$ and $\hat{\mathbf{s}}_1$ are also distinct. Hence, the probability that $\mathbf{I}^{(0)} = H_m(\hat{\mathbf{t}}_0)\hat{\mathbf{s}}_0 = H_m(\hat{\mathbf{t}}_1)\hat{\mathbf{s}}_1 = \mathbf{I}^{(1)}$ is negligible (this is true if small elements of R_q are invertible). Now, suppose that $\hat{\mathbf{t}}_0 = \hat{\mathbf{t}}_1$ and $C_0 \neq C_1$. Then, unless one accidentally finds a collision where $\hat{\mathbf{s}}_0 \neq \hat{\mathbf{s}}_1$ and $\mathbf{A}\hat{\mathbf{s}}_0 = \mathbf{A}\hat{\mathbf{s}}_1$ (which happens only with negligible probability), we must have $\hat{\mathbf{s}}_0 = \hat{\mathbf{s}}_1$. The latter may occur in two scenarios:

- $\mathbf{s}_0 \neq \mathbf{s}_1$ and $\mathbf{s}'_0 \neq \mathbf{s}'_1$, but $\mathbf{s}_0 + \mathbf{s}'_0 = \mathbf{s}_1 + \mathbf{s}'_1$. Due to the randomness of the generations of $\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}'_0, \mathbf{s}'_1$, this scenario only happens with negligible probability.
- $\mathbf{s}_0 = \mathbf{s}_1$ and $\mathbf{s}'_0 = \mathbf{s}'_1$. Note that, if $\mathbf{s}_0, \mathbf{s}_1$ are obtained by two different executions of algorithm $\mathsf{MasterKeyGen}$, then $\mathbf{s}_0 = \mathbf{s}_1$ only happens with negligible probability. Furthermore, two different executions of algorithm $\mathsf{DerivePublicKeyGen}$ with $C_0 \neq C_1$ should produce distinct $\mathsf{s}'_0, \mathbf{s}'_1$ with overwhelming probability.

The above analysis shows that the given SALRS scheme is correct with overwhelming probability.

Lattice-Based Instantiation of the KEM Scheme Π_{kem}. We employ Kyber [7] to instantiate Π_{kem}, by setting GP_{kem} contains only the parameters $(n, k, q, \eta, d_u, d_v, d_t)$ and the hash function. Note that, the ciphertext in the CPA version of Kyber is pseudorandom based on the Decision Module-LWE (D-MLWE) assumption, and it hides not only the plaintext but also the public key. The CCA version of Kyber thus can be easily shown to satisfy not only CCA-security but also CCA-key-indistinguishability. For concreteness, we will use the Kyber variant Kyber768, which features public key size 1184 bytes and ciphertext size 1088 bytes.

Signing Trials. At Step 9 of the signing algorithm, if $\mathbf{z}_{\bar{i}} = \mathbf{y} + c_{\bar{i}}\hat{\mathbf{s}}_{\bar{i}} \notin S^l_{\gamma-2\theta\eta}$, then the signer has to go back to Step 4. Let us compute the probability of such restarting for uniformly random $\mathbf{y} \xleftarrow{R} S^l_\gamma$, $c_{\bar{i}} \in \mathbf{B}_\theta$ and $\hat{\mathbf{s}}_{\bar{i}} = \mathbf{s} + \mathbf{s}'_{\bar{i}} \in S^l_{2\eta}$. First, we have $\mathbf{x} := c_{\bar{i}}\hat{\mathbf{s}}_{\bar{i}} \in S^l_{2\theta\eta}$. For each entry $y_j \xleftarrow{R} [-\gamma, \gamma]$ of $\mathbf{y}$, and each entry $x_j \in [-2\theta\eta, 2\theta\eta]$ of $\mathbf{x}$, the probability that $y_j + x_j$ falls into the "safe zone"

$[-(\gamma - 2\theta\eta), \gamma - 2\theta\eta]$ is exactly the ratio between the cardinalities of the range $[-(\gamma - 2\theta\eta), \gamma - 2\theta\eta]$ and the range $[-\gamma, \gamma]$. Therefore, we have:

$$\Pr\left[\mathbf{z}_{\tilde{i}} \in S^l_{\gamma-2\theta\eta}\right] = \frac{|S^l_{\gamma-2\theta\eta}|}{|S^l_{\gamma}|} = \left(1 - \frac{2\theta\eta}{\gamma + 1/2}\right) \approx e^{-2nl\theta\eta/\gamma},$$

where we use the fact that parameter γ is set to be large compared to $1/2$. As a result, the probability of restarting is approximately close to $1 - e^{-2nl\theta\eta/\gamma}$. In particular, if we set parameters $n, l, \theta, \eta, \gamma$ so that $2nl\theta\eta/\gamma < \log_e(3)$ (see below), then, on average, the signer has to run Step 4-Step 9 of the signing algorithm less than 3 times.

Concrete Parameters and Efficiency. To set parameters that yield a scheme with at least 128 bits of security, we rely on the parameters and analyses of Dilithium [12], Kyber [7,23] and [4]. In particular, modulus q is set so that every element of R_q with infinity norm less than $\frac{1}{\sqrt{8}} \cdot 2^{35/8}$ is invertible, and parameters $n, l, \theta, \eta, \gamma$ are set so that the number of signing trials is less than 3 on average. Similar to [12], we can use SHAKE-256 to implement the functions H_{A}, ExpandV, and H_m, and use the SampleInBall algorithm in [12, Fig. 2] to implement H_θ. Table 2 shows the concrete parameters and efficiency of the proposed lattice-based SALRS.

Table 2. Concrete parameters and efficiency of the proposed lattice-based SALRS.

Parameter	Value
Dimension n	256
Modulus q	Prime $q \approx 2^{35}$ and $q = 17 \bmod 32$
Module SIS/LWE parameters (k, l, m)	$(3, 5, 1)$
Bounds $(\theta, \eta, \gamma, \gamma - 2\theta\eta)$	$(60, 3, 699453, 699093)$
Master public key (MPK) size	4.44 KB
Master secret key (MSK) size	2.97 KB
Derived public key (DPK) size	4.34 KB
Signature size $(r = 8)$	27.4 KB
Signature size $(r = 16)$	53.6 KB
Signature size $(r = 32)$	106.1 KB
Signature size $(r = 64)$	211.1 KB

4 Proofs of Security and Privacy

Theorem 3. *The SALRS scheme is signer-linkable in the random oracle model.*

Proof. We prove that the SALRS scheme is signer-linkable under the Module-SIS (MSIS) assumption. Due to space limitation, we defer the proof details to the full version.

Theorem 4. *The SALRS scheme is signer-anonymous in the random oracle model.*

Proof. We prove that the SALRS scheme has signer-anonymity under the Decision Module-LWE (D-MLWE) assumption. Due to space limitation, we defer the proof details to the full version.

Theorem 5. *The SALRS scheme is signer-non-slanderable in the random oracle model.*

Proof. We prove that the SALRS scheme is signer-non-slanderable under the Module-SIS (MSIS) and Decision Module-LWE (D-MLWE) assumptions. Due to space limitation, we defer the proof details to the full version.

Theorem 6. *The SALRS scheme is master-public-key-unlinkable in the random oracle model.*

Proof. Suppose the underlying KEM scheme is CCA secure and CCA Key Indistinguishable, we prove that the SALRS scheme is master-public-key-unlinkable under the Decision Module-LWE (D-MLWE) assumption. Due to space limitation, we defer the proof details to the full version.

5 Conclusion

In this paper, we proposed a new cryptographic primitive, referred to as Linkable Ring Signature Scheme with Stealth Addresses (SALRS), which comprehensively and strictly captures the security and privacy requirements of hiding the payer and payee of the transactions in cryptocurrencies. We also proposed a lattice-based SALRS construction and proved its security and privacy in the random oracle model. As a result, our construction provides strong confidence on security and privacy in twofolds, being proved under strong models which capture the practical scenarios of cryptocurrencies, and being potentially quantum-resistant. The efficiency analysis also shows that our lattice-based SALRS scheme is practical for real implementations.

A A Proof of Theorem 1

Proof (Sketch). Due to page limitation, below we give the proof sketch and defer the proof details to the full version.

Suppose there exists an adversary $\mathcal{A}$ that breaks the strong unforgeability, i.e. succeeds in $\mathsf{Game}_{\mathsf{euf}}$ with non-negligible advantage. We can construct an algorithm $\mathcal{B}$ that either succeeds $\mathsf{Game}_{\mathsf{snlink}}$ with non-negligible advantage or succeeds $\mathsf{Game}_{\mathsf{snnsl}}$ with non-negligible advantage.

$\mathcal{B}$ is offered two challengers $\mathcal{C}_0$ and $\mathcal{C}_1$, which will interact with $\mathcal{B}$ for $\mathsf{Game}_{\mathsf{snlink}}$ and $\mathsf{Game}_{\mathsf{snnsl}}$ respectively. On the other side, $\mathcal{B}$ interacts with $\mathcal{A}$ for $\mathsf{Game}_{\mathsf{euf}}$, making use of $\mathcal{C}_0$ or $\mathcal{C}_1$ behind, while it is indistinguishable from the view of $\mathcal{A}$.

At the **Output Phase** of $\mathsf{Game}_{\mathsf{euf}}$, $\mathcal{A}$ outputs a (message, derived public key ring, signature) tuple (M^*, R^*, σ^*), such that (1) $\mathsf{Verify}(M^*, R^*, \sigma^*) = 1$, and (2) $R^* \subseteq L_{dpk}$, and (3) (M^*, R^*, σ^*) is not returned by $\mathsf{OSign}(\cdot, \cdot, \cdot)$.

Wlog., let $R^* = (\mathsf{DPK}_1^*, \ldots, \mathsf{DPK}_k^*)$, $\mathcal{B}$ can obtain k (message, derived public key ring, signature) tuples $\{(M_i, R^*, \sigma_i)\}_{i=1}^k$ by making use of $\mathcal{C}_0$ or $\mathcal{C}_1$, such that (1) $\mathsf{Verify}(M_i, R^*, \sigma_i) = 1$ $(i = 1, 2, \ldots, k)$, and (2) $\mathsf{Link}(M_i, R^*, \sigma_i, M_j, R_j^*, \sigma_j) = 0$ $\forall i, j \in [1, k]$ *s.t.* $i \neq j$, where σ_i corresponds to DPK_i^*. Consider these $k+1$ signatures, we have that either the following **Case I** or the **Case II** happens:

- **Case I**: $\mathsf{Link}(M^*, R^*, \sigma^*, M_j, R^*, \sigma_j) = 0$ $\forall j \in \{1, \ldots, k\}$,
- **Case II**: $\exists \hat{i} \in \{1, \ldots, k\}$ *s.t.* $\mathsf{Link}(M^*, R^*, \sigma^*, M_{\hat{i}}, R^*, \sigma_{\hat{i}}) = 1$.

If **Case I** happens, these $k+1$ signatures can be used to win $\mathsf{Game}_{\mathsf{snlink}}$, otherwise, the two signatures (M^*, R^*, σ^*), $(M_{\hat{i}}, R^*, \sigma_{\hat{i}})$ can be used to win $\mathsf{Game}_{\mathsf{snnsl}}$.

References

1. Au, M.H., Chow, S.S.M., Susilo, W., Tsang, P.P.: Short linkable ring signatures revisited. EuroPKI **2006**, 101–115 (2006). https://doi.org/10.1007/11774716_9
2. Au, M.H., Liu, J.K., Susilo, W., Yuen, T.H.: Constant-size id-based linkable and revocable-iff-linked ring signature. INDOCRYPT **2006**, 364–378 (2006). https://doi.org/10.1007/11941378_26
3. Au, M.H., Liu, J.K., Susilo, W., Yuen, T.H.: Secure id-based linkable and revocable-iff-linked ring signature with constant-size construction. Theor. Comput. Sci. **469**, 1–14 (2013). https://doi.org/10.1016/j.tcs.2012.10.031
4. Baum, C., Damgård, I., Lyubashevsky, V., Oechsner, S., Peikert, C.: More efficient commitments from structured lattice assumptions. SCN **2018**, 368–385 (2018). https://doi.org/10.1007/978-3-319-98113-0_20
5. Baum, C., Lin, H., Oechsner, S.: Towards practical lattice-based one-time linkable ring signatures. ICICS **2018**, 303–322 (2018). https://doi.org/10.1007/978-3-030-01950-1_18
6. Bellare, M., Boldyreva, A., Desai, A., Pointcheval, D.: Key-privacy in public-key encryption. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 566–582. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_33
7. Bos, J.W., et al.: CRYSTALS - kyber: a CCA-secure module-lattice-based KEM. In: EuroS&P 2018. pp. 353–367 (2018). DOI: https://doi.org/10.1109/EuroSP.2018.00032
8. Boyen, X., Haines, T.: Forward-secure linkable ring signatures from bilinear maps. Cryptography **2**(4), 35 (2018). https://doi.org/10.3390/cryptography2040035
9. Branco, P., Mateus, P.: A code-based linkable ring signature scheme. ProvSec **2018**, 203–219 (2018). https://doi.org/10.1007/978-3-030-01446-9_12
10. CoinMarketCap: Top 100 cryptocurrencies by market capitalization. https://coinmarketcap.com. Accessed 27 Apr 2019
11. Courtois, N.T., Mercer, R.: Stealth address and key management techniques in blockchain systems. ICISSP **2017**, 559–566 (2017). https://doi.org/10.5220/0006270005590566
12. Ducas, L., et al.: Crystals-dilithium: a lattice-based digital signature scheme. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2018**(1), 238–268 (2018). https://doi.org/10.13154/tches.v2018.i1.238-268

13. Esgin, M.F., Steinfeld, R., Sakzad, A., Liu, J.K., Liu, D.: Short lattice-based one-out-of-many proofs and applications to ring signatures. IACR Cryptol. ePrint Arch. **2018**, 773 (2018)
14. Fujisaki, E.: Sub-linear size traceable ring signatures without random oracles. CT-RSA **2011**, 393–415 (2011). https://doi.org/10.1007/978-3-642-19074-2_25
15. Fujisaki, E.: Sub-linear size traceable ring signatures without random oracles. IEICE Trans. **95–A**(1), 151–166 (2012). https://doi.org/10.1587/transfun.E95.A.151
16. Fujisaki, E., Suzuki, K.: Traceable ring signature. PKC **2007**, 181–200 (2007). https://doi.org/10.1007/978-3-540-71677-8_13
17. Langlois, A., Stehle, D.: Worst-case to average-case reductions for module lattices. Des. Codes Crypt. **75**(3), 565–599 (2015). https://doi.org/10.1007/s10623-014-9938-4
18. Libert, B., Ling, S., Nguyen, K., Wang, H.: Zero-knowledge arguments for lattice-based accumulators: logarithmic-size ring signatures and group signatures without trapdoors. In: EUROCRYPT 2016 Part II. pp. 1–31 (2016). DOI: https://doi.org/10.1007/978-3-662-49896-5_1
19. Liu, J.K., Au, M.H., Susilo, W., Zhou, J.: Linkable ring signature with unconditional anonymity. IEEE Trans. Knowl. Data Eng. **26**(1), 157–165 (2014). https://doi.org/10.1109/TKDE.2013.17
20. Liu, J.K., Wei, V.K., Wong, D.S.: Linkable spontaneous anonymous group signature for ad hoc groups (extended abstract). ACISP **2004**, 325–335 (2004). https://doi.org/10.1007/978-3-540-27800-9_28
21. Liu, Z., Yang, G., Wong, D.S., Nguyen, K., Wang, H.: Key-insulated and privacy-preserving signature scheme with publicly derived public key. EuroS&P 2019, to appear https://eprint.iacr.org/2018/956
22. Lu, X., Au, M.H., Zhang, Z.: Raptor: a practical lattice-based (linkable) ring signature. IACR Cryptol. ePrint Archive **2018**, 857 (2018). https://eprint.iacr.org/2018/857
23. Lyubashevsky, V., Seiler, G.: Short, invertible elements in partially splitting cyclotomic rings and applications to lattice-based zero-knowledge proofs. In: EUROCRYPT 2018 Part I. pp. 204–224 (2018). DOI: 10.1007/978-3-319-78381-9_8
24. Noether, S., Mackenzie, A.: Ring confidential transactions. Ledger **1**, 1–18 (2016)
25. van Saberhagen, N.: Cryptonote v 2.0 (2013). https://cryptonote.org/whitepaper.pdf
26. Sun, S., Au, M.H., Liu, J.K., Yuen, T.H.: Ringct 2.0: A compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency monero. In: ESORICS 2017 Part II. pp. 456–474 (2017). DOI: https://doi.org/10.1007/978-3-319-66399-9_25
27. Todd, P.: Stealth addresses. https://lists.linuxfoundation.org/pipermail/bitcoin-dev/2014-January/004020.html
28. Torres, W.A.A., et al.: Post-quantum one-time linkable ring signature and application to ring confidential transactions in blockchain (lattice ringct v1.0). In: ACISP 2018. pp. 558–576 (2018). DOI: https://doi.org/10.1007/978-3-319-93638-3_32
29. Tsang, P.P., Wei, V.K.: Short linkable ring signatures for e-voting, e-cash and attestation. ISPEC **2005**, 48–60 (2005). https://doi.org/10.1007/978-3-540-31979-5_5
30. Tsang, P.P., Wei, V.K., Chan, T.K., Au, M.H., Liu, J.K., Wong, D.S.: Separable linkable threshold ring signatures. INDOCRYPT **2004**, 384–398 (2004). https://doi.org/10.1007/978-3-540-30556-9_30

31. Yuen, T.H., Liu, J.K., Au, M.H., Susilo, W., Zhou, J.: Efficient linkable and/or threshold ring signature without random oracles. Comput. J. **56**(4), 407–421 (2013). https://doi.org/10.1093/comjnl/bxs115
32. Zhang, H., Zhang, F., Tian, H., Au, M.H.: Anonymous post-quantum cryptocash. IACR Cryptol. ePrint Archive **2017**, 716 (2017). http://eprint.iacr.org/2017/716