Hindawi
Security and Communication Networks
Volume 2022, Article ID 1417607, 9 pages
https://doi.org/10.1155/2022/1417607

Research Article

A Hybrid Design of Linkable Ring Signature Scheme with Stealth Addresses

Weizhou Li,[1] Zhiqiang Lin ,[2] and Qi Chen[3]

[1]*School of Mathematical Sciences, South China Normal University, Guangzhou 510631, China*
[2]*School of Mathematics and Information Science, Guangzhou University, Guangzhou 510006, China*
[3]*Institute of Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou 510006, China*

Correspondence should be addressed to Zhiqiang Lin; linzhiqiang0824@163.com

Received 13 November 2021; Revised 14 December 2021; Accepted 10 January 2022; Published 7 February 2022

Academic Editor: Yuling Chen

Blockchain is a transformational technology which affects finance, Internet, and politics. However, many privacy protection problems for blockchain are waiting to be solved. In this study, we propose a novel linkable ring signature scheme with stealth addresses, which enables the payer and payee of the transaction to be anonymous and unlinkable in the cryptocurrency. The scheme is combined with an elliptic curve discrete logarithm (ECD logarithm)-based key encapsulation mechanism (KEM) stage and a lattice-based signature stage. The master public key and master secret key are much smaller compared with the previous scheme. Complete secure proof of the scheme is also presented in this study.

1. Introduction

1.1. Background. As a novel technology, blockchain technology has been widely used in many fields since its introduction in [1–6]. The development of blockchain is also inseparable from digital signatures. Digital signature provides security and authentication for information during the process of information dissemination, such as protecting user's privacy and preventing double spending with the support of the anonymity and linkability of the ring signature scheme [7, 8].

In [9], the authors proposed a linkable ring signature scheme with stealth addresses denoted by SALRS. This scheme enables the payer and payee of the transaction to be anonymous and unlinkable in the cryptocurrency. Specifically, the linkable ring signature and stealth address [10–12] are employed in CryptoNote [10]. When a payer A wants to pay a payee B through a transaction, the payer B uses a stealth address to generate a derived public key. Then, the payer A uses the derived public key as the address of the payee B. Also, transactions cannot be identified because of the absence of the master public key. When the payee B, as a payer in transaction, wants to spend his coins on the derived public key, he generates a linkable ring signature with the support of a set of derived public keys. In order to verify the linkable ring signature, it is not necessary for anyone to find out that the actual signer is corresponding to the derived public key. When it comes to the linkability which can prevent double spending in a transaction, if two signatures are generated by the payee B corresponding to a derived public key, they will be detected as linked because the coin corresponding to the derived public key can be used only once. In this study, we focus on concrete construction of the SALRS scheme in order to enable both payer and payee of a transaction to be hidden in the cryptocurrency.

1.2. Our Contribution. We propose a novel concrete linkable ring signature scheme with stealth addresses based on the elliptic curve discrete logarithm (ECD logarithm) for the key encapsulation mechanism (KEM) stage and lattice for signature stage. The ECD-based KEM provides smaller keys. In particular, the size of the master public key is 510 bits, and the size of the master secret key is 512 bits, which is much smaller than the ones in the previous scheme in [9]. Moreover, all the secure properties which a SALRS should

have, including unforgeability, linkability, nonslanderability, anonymity, master-public-key-unlinkability, and derived-public-key-unlinkability, still keep.

1.3. Organization of This Paper. The rest of the study is organized as follows. Preliminaries are given in Section 2. In Section 3, we formally propose the SALRS scheme. Afterwards, the security models of the SALRS scheme are presented in Section 4. As a vital content of this study, in Section 5, our concrete SALRS scheme is showed. In Section 6, we analyze and prove the security of our SALRS scheme. Moreover, efficiency analysis, especially less storage cost of our SALRS scheme, is introduced in Section 7. Finally, we summarize this study and come out conclusions in Section 8.

1.4. Related Work. There are a lot of classic linkable ring signature schemes relied on the hardness number-theoretic problems, such as [13, 14]. Many of them have specific application scenarios, for example, [15, 16] are based on certificates and identity-based, respectively. However, a lot of cryptographic schemes based on classical number theories are suffered from future quantum computer's threats [17]. All the same, some advantages of cryptographic schemes relying on classical number theory, for example, the elliptic curve discrete logarithm [18], are faster calculation and less storage cost.

Lattice-based ring signatures were first introduced by Brakerski and Tauman-Kalai in 2010. They proposed a construction of ring signature scheme based on SIS assumption. Then, in 2013, Melchor et al. proposed a ring signature scheme based on LWE assumption. Until now, many lattice-based ring signature schemes have been proposed, such as [19–21].

The existing works on the linkable ring signature and stealth address have been proposed, e.g., [8, 22]. However, most of the existing works above either merely consider linkable ring signature or stealth address rather than both of them. Fortunately, literature [9] has successfully proposed a new cryptographic primitive denoted by SALRS. The new cryptographic primitive has not only combined the linkable ring signature with the stealth address but also captured adversarially chosen key attacks in the linkability model. Additionally, it is also potentially quantum resistant.

2. Preliminaries

In this section, before showing our concrete SALRS construction, we give some preliminary results about the mathematical background concerning bilinear maps and lattice and complexity assumptions. For more details, please refer to [23–26].

2.1. Mathematical Background. Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be the groups of prime order p, and g_i be a generator of $\mathbb{G}_i$ $(i = 1,2)$. Set $\mathbb{G}_1 \neq \mathbb{G}_2$, and there exists no efficient homomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$. We say that e: $\mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ is a bilinear, efficient, and computable map if it satisfies the following two properties.

(1) Bilinear: for $\forall\, a,b \in\in \mathbb{Z}_p$, where the integers modulo p is denoted by $\mathbb{Z}_p$, we have $e(g_1^a, g_1^b) = e\ (g_1, g_2)^{ab}$.

(2) Efficient: $e(g_1, g_2) \neq 1$.

Let q and n be two positive integers, and denote the integers modulo q by $\mathbb{Z}_q$, which will be represented in the range $(-(q/2), (q/2)]$ or $[-(q-1/2), (q-1/2)]$, where q is even or odd, respectively. Let R and R_q be the rings $\mathbb{Z}[X]/(X^n+1)$ and $\mathbb{Z}_q[X]/(X^n+1)$, respectively. We set $r = a_0 + a_1 X + \ldots + a_{n-1} X^{n-1} \in R$ and $\mathbf{r} = (r_1, \ldots r_k) \in R^k$ to define the l_1, l_2, and l_∞ norms of r and $\mathbf{r}$ as follows:

(1) $\|r\|_\infty \triangleq \max_i |a_i|$

(2) $\|r\|_1 \triangleq \sum_i |a_i|$

(3) $\|r\|_2 \triangleq \sqrt{|a_0|^2 + \cdots + |a_{n-1}|^2}$

(4) $\|\mathbf{r}\|_\infty \triangleq \max_i \|r_i\|_\infty$

(5) $\|r\|_1 \triangleq \sum_i \|r\|_1$

(6) $\|\mathbf{r}\|_2 \triangleq \sqrt{\|r_1\|_2^2 + \cdots + \|r_k\|_2^2}$

We also denote two sets:

(1) $SS_\eta \triangleq \backslash\{r \in R | \|r\|_\infty \leq \backslash\} \backslash \{r \in R |\, \|r\|_\infty \leq \eta \backslash\}$

(2) $\mathbf{B}_\theta \triangleq \backslash\{r \in\in, R_q |\, r$ has θ coefficients that are ± 1 and the rest are $0\backslash\}$

2.2. Complexity Assumptions. The security of our novel SALRS scheme is based on bilinear Diffie-Hellman 1 assumption, module-SIS assumption, and module-LWE assumption.

2.2.1. Bilinear Diffie-Hellman 1 (BDH-1) Assumption. Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be groups of prime order p, and g_i be a generator of $\mathbb{G}_i$ $(i = 1,2)$. Set $\mathbb{G}_1 \neq \mathbb{G}_2$, and there exists no efficient homomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$. e: $\mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ is a bilinear, efficient, and computable map. The Bilinear Diffie-Hellman 1 (BDH-1) problem is that given g_1, g_2, g_1^a, g_1^b, compute $e(g_1, g_2)$, where $a, b \in\in \mathbb{Z}_p$.

2.2.2. Module-SIS Assumption. The module-SIS problem with parameters (n, q, k, l, β) is that for uniformly random $\mathbf{A} \in\in R_q^{k\times l}$, $\mathbf{t} \in$, R_q^k, and $k \times k$ identity matrix $\mathbf{I}$, find $\mathbf{x} \in$, R^{k+l}, such that $\|\mathbf{x}\|_2 \leq \beta$ and $[\mathbf{A}|\mathbf{I}]\cdot\mathbf{x} = \mathbf{t}$. The problem can be adapted into the infinity-norm version, where $\|\mathbf{x}\|_\infty \leq \beta$. Additionally, the homogeneous version of the module-SIS problem is defined with $\mathbf{t} = \mathbf{0}$ and $\mathbf{x} \neq \mathbf{0}$.

2.2.3. Module-LWE Assumption. The module-LWE problem with parameters (n, q, k, l, η) is that for uniformly random $\mathbf{A} \in\in R_q^{k\times l}$, let $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \in R_q^k$, where $\mathbf{s} \in S_\eta^l$, $\mathbf{e} \in S_\eta^k$ have their entries selected concerning some distributions (uniform distribution, Gaussian distribution) over S_η. There are two versions about module-LWE. The search variant of module-LWE is to find $\mathbf{s}$ given $(\mathbf{A}, \mathbf{b})$. The decision variant is to distinguish $(\mathbf{A}, \mathbf{b})$ from a uniformly random pair over

$R_q^{k\times l}\times$, R_q^k. In this study, we use a transformed version of the decision variant of module-LWE, which is to distinguish (**A**, **As**) from (**A**, **r**), where **A**$\leftarrow$, $R_q^{k\times l}$, **s**$\leftarrow$, S_η^l, and **r**$\leftarrow$, R_q^k.

3. SALRS Scheme

The syntax of linkable ring signature scheme with stealth addresses (SALRS) was first purposed by [9], which realizes the cryptographic functions that a cryptocurrency wants to hide payers and payees of transactions. There are eight algorithms in a SALRS scheme.

Setup (λ) $\longrightarrow$ PP: the input to this algorithm is a security parameter λ and outputs the public parameters PP.

MasterKeyGen (PP) $\longrightarrow$ (MPK, MSK): the input to this algorithm is the public parameters (PP) and outputs the user's master key pair (MPK, MSK) (master public key, master secret key).

DerivedPublicKeyGen (MPK) $\longrightarrow$ DPK: the input to this algorithm is a master public key (MPK) and outputs the derived public key (DPK).

DerivedPublicKeyOwnerCheck (DPK, MPK, MSK) $\longrightarrow$ 1/0: the input to this algorithm is a derived public key (DPK) and a master key pair (MPK, MSK) and outputs $b \in \{0,1\}$. 1 and 0 indicate that the derived public key (DPK) is valid or invalid, respectively.

DerivedPublicKeyPublicCheck (DPK) $\longrightarrow$ 1/0: the input to this algorithm is a derived public key (DPK) and outputs $b \in \{0,1\}$. 1 and 0 indicate that the derived public key (DPK) is well-formed or not well-formed, respectively.

Sign (M, R, DPK, MPK, MSK) $\longrightarrow \sigma$: the input to this algorithm is a message M, a ring of well-formed derived public keys $R = (\mathrm{DPK}_1, \ldots, \mathrm{DPK}_r)$ (where we regard the public key ring R as an order set, namely, it consists of the public keys which are ordered and have indexes), a derived public key DPK $\in R$, and a master key pair (MPK, MSK) for the derived public key (DPK) and outputs a signature σ.

Verify(M, R, σ) $\longrightarrow$ 1/0: the input to this algorithm is a message M, a ring of well-formed derived public keys R, and a signature σ and outputs $b \in \{0,1\}$. 1 and 0 indicate that the signature σ is valid or invalid, respectively.

Link (M_0, R_0, σ_0, M_1, R_1, σ_1) $\longrightarrow$ 1/0: the input to this algorithm is two valid (message M, derived public key ring R, signature σ) tuples (M_0, R_0, σ_0) and (M_1, R_1, σ_1) and outputs 0 or 1. 1 and 0 indicate that the two signatures are linked or unlinked, respectively.

4. Security Model of SALRS

A SALRS scheme should be correctness, unforgeable, linkable, nonslanderable, anonymous, master-public-key-unlinkable, and derived-public-key-unlinkable, which ensure the scheme satisfying the security and privacy protection requirements of cryptocurrencies in most practical settings.

In the following games, we use $\mathscr{A}$ or $n()$ to denote any probabilistic polynomial time (PPT) adversary or polynomial, respectively.

4.1. Correctness. Correctness means that one can derive a "right" feedback while honestly performing the protocols.

Let PP $\leftarrow$ setup (λ),

(1) For any (MPK, MSK) $\leftarrow$ `MasterKeyGen` (PP) and any DPK $\leftarrow$ `DerivedPublicKeyGen` (MPK), we have `DerivedPublicKeyOwnerCheck` (DPK, MPK, MSK) = 1 and `DerivedPublicKeyPublicCheck` (DPK) = 1.

(2) For any message M, any ring of well-formed derived public keys R, and any derived public key DPK $\in R$, such that `DerivedPublicKeyOwnerCheck` (DPK, MPK, MSK) = 1 for some master key pair (MPK, MSK), we have verify (M, R, sign (M, R, DPK, MPK, MSK)) = 1.

(3) For any message M_i, any ring of well-formed derived public keys R_i, and any derived public key $\mathrm{DPK}_i \in \mathrm{R}_i$, such that `DerivedPublicKeyOwnerCheck` ($\mathrm{DPK}_i$, MPK_i, MSK_i) = 1 for some master key pair (MPK_i, MSK_i), let $\sigma_i \leftarrow$ sign (M_i, R_i, DPK_i, MPK_i, MSK_i) ($i = 0,1$). We have link (M_0, R_0, σ_0, M_1, R_1, σ_1) = 1 if $\mathrm{DPK}_0 = \mathrm{DPK}_1$, and Pr[link (M_0, R_0, σ_0, M_1, R_1, σ_1) = 0] $\geq 1 - negl$ (λ) if $\mathrm{DPK}_0 \neq \mathrm{DPK}_1$, where *negl* is a negligible function.

4.2. Unforgeability. Unforgeability means that only the user who knows the secret key for some public key in a ring can generate a valid signature.

4.2.1. Setup. PP $\leftarrow$ setup (λ) is run. PP is given to $\mathscr{A}$. $\{\{(\mathrm{MPK}_i, \mathrm{MSK}_i) \leftarrow \mathrm{MasterKeyGen}(\mathrm{PP})\}\}_{i=1}^{n(\lambda)}$ are run and $\{\mathrm{MPK}_i\}$ are given to $\mathscr{A}$.

4.2.2. Probing Phase. $\mathscr{A}$ can query the following oracles:

(1) Derived Public Key Adding Oracle, ODPKAdd(): it means that ODPKAdd (DPK, MPK) returns $b \leftarrow$ DerivedPublicKeyOwnerCheck (DPK, MPK, MSK) to $\mathscr{A}$. If $b = 1$, set $L_{dp\,k} = L_{dp\,k} \cup \{\mathrm{DPK}\}$, where $L_{dp\,k} = \varnothing$ is initialized.

(2) Signing Oracle, OSign(): it means that OSign (M, R, DPK), where DPK $\in R \cap$, $L_{dp\,k}$, returns $\sigma \leftarrow$ sign (M, R, DPK, MPK, MSK) to $\mathscr{A}$, where (MPK, MSK) is the master key pair for DPK.

4.2.3. Output Phase. $\mathscr{A}$ outputs a message M^*, a ring of well-formed derived public keys R^*, and a signature σ^*.

Let $S_{so} = \{(M, R, \mathrm{DPK}, \sigma)\}$ be the query-answer tuples for OSign. $\mathscr{A}$ succeeds if

(1) Verify (M^*, R^*, σ^*) = 1 and

(2) $R^* \subseteq L_{dp\,k}$ and

(3) (M^*, R^*, ?, σ^*) $\notin S_{so}$, where ? means that (M^*, R^*, σ^*) is not a tuple obtained by querying OSign.

Definition 1. The SALRS is unforgeable if for all $\mathcal{A}$, $A\ dv_{\mathcal{A}}^{uf}$ is negligible, where $A\ dv_{\mathcal{A}}^{uf} = \Pr[\mathcal{A}\ \text{succeeds}]$. We name the game for unforgeability Game_{uf}.

4.3. Linkability. Linkability means that if the key owner generates two or multiple valid signatures with respect to one derived public key, the signatures will be found to be linked.

4.3.1. Setup. PP $\leftarrow$ setup (λ) is run. PP is given to $\mathcal{A}$.

4.3.2. Output Phase. $\mathcal{A}$ outputs k $(k \geq (2)$ tuples $(M_i^*, R_i^*, \sigma_i^*)$ $(i = 1,\ldots,k)$.

$\mathcal{A}$ succeeds if

(1) Verify $(M_i^*, R_i^*, \sigma_i^*) = 1$ and

(2) Link $(M_i^*, \mathrm{R}_i^*, \sigma_i^*, M_j^*, \mathrm{R}_j^*, \sigma_j^*) = 0(\forall\ i, j \in [1,k],\ s.t.\ i \neq j)$ and

(3) $|\cup_{i=1}^{k} R_1^*| < k$.

Definition 2. The SALRS is linkable if for all $\mathcal{A}$, $A\ dv_{\mathcal{A}}^{\text{link}}$ is negligible, where $A\ dv_{\mathcal{A}}^{\text{link}} = \Pr[\mathcal{A}\ \text{succeeds}]$. We name the game for linkability Game link.

4.4. Nonslanderability. Nonslanderability means that no one can frame other users by creating a signature which is linked to a signature of the target user.

4.4.1. Setup. Same as that of Game_{uf}.

4.4.2. Probing Phase. Same as that of Game_{uf}.

4.4.3. Output Phase. $\mathcal{A}$ outputs two tuples (M', R', σ') and $(M*, R*, \sigma^*)$.

Let $S_{so} = \backslash\{(M, R, \text{DPK}, \sigma)\backslash\}$ be the query-answer tuples for OSign. $\mathcal{A}$ succeeds if

(1) Verify $(M*, R*, \sigma^*) = 1$ and

(2) $(M', R', \sigma') \in S_{so}$ for some derived public keys $\text{DPK}' \in R',\ \cap\ L_{dp\,k}$ and

(3) $(M*, R*, \text{DPK}', \sigma^*) \notin S_{so}$ and

(4) Link $(M*, R^*, \sigma^*, M', \mathrm{R}', \sigma') = 1$

Definition 3. The SALRS is nonslanderable if for all $\mathcal{A}$, $A\ dv_{\mathcal{A}}^{ns}$ is negligible, where $A\ dv_{\mathcal{A}}^{ns} = \Pr[\mathcal{A}\ \text{succeeds}]$. We name the game for nonslanderability Game_{ns}.

4.5. Anonymity. Anonymity means that no one can identify the signer's derived public key out of the ring, with a valid signature with respect to a ring of derived public keys.

4.5.1. Setup. Same as that of Game_{uf}.

4.5.2. Probing Phase 1. Same as the probing phase of Game_{uf}.

4.5.3. 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) $\text{DPK}_{i_0}, \text{DPK}_{i_1} \in R^* \cap L_{dp\,k}$

(2) None of OSign with DPK_{i_0} and DPK_{i_1} was queried.

A random bit $b \in \backslash\{0,1\backslash\}$ is chosen, and $\mathcal{A}$ is given the $\sigma \leftarrow$ sign $(M*, R^*, \text{DPK}_{i_b}, \text{MPK}, \text{MSK})$, where (MPK, MSK) is the master key pair for DPK_{i_b}.

4.5.4. Probing Phase 2. Same as the probing phase 1, but with the restriction that OSign with DPK_{i_0} and DPK_{i_1} cannot be queried.

4.5.5. Output Phase. $\mathcal{A}$ outputs a bit b' as its guess to b.

Definition 4. The SALRS is anonymous if for all $\mathcal{A}$, $A\ dv_{\mathcal{A}}^{\text{ano}}$ is negligible, where $A\ dv_{\mathcal{A}}^{\text{ano}} = |\Pr[b = b] - 1/2|$. We name the game for anonymity Game_{ano}.

4.6. Master-Public-Key-Unlinkability. Master-public-key-unlinkability means that with the support of a derived public key and the corresponding signatures, no one can distinguish which master public key is the one which it was derived from.

4.6.1. Setup. Same as that of Game_{uf}.

4.6.2. Probing Phase 1. Same as the probing phase of Game_{uf}.

4.6.3. Challenge. $\mathcal{A}$ outputs two distinct indices $1 \leq i_0, i_1 \leq n\ (\lambda)$. A random bit $b \in \backslash\{0,1\backslash\}$ is chosen, and $\text{DPK}^* \leftarrow$ `DerivedPublicKeyGen` (MPK_{i_b}) is given to $\mathcal{A}$. Set $L_{dp\,k} = L_{dp\,k} \cup, \text{DPK}^*\backslash\}$.

4.6.4. Probing Phase 2. Same as the probing phase 1, with the restriction that ODPKAdd $(\text{DPK}^*, \text{MPK}_{i_j})$ $(j \in \backslash\{0,1\backslash\})$ cannot be queried.

4.6.5. Output Phase. $\mathcal{A}$ outputs a bit $b' \in \backslash\{0,1\backslash\}$ as its guess to b.

Definition 5. The SALRS is master-public-key-unlinkable if for all $\mathcal{A}$, $A\ dv_{\mathcal{A}}^{\text{mpkunl}}$ is negligible, where $A\ dv_{\mathcal{A}}^{\text{mpkunl}} = |\Pr[b = b] - 1/2|$. We name the game for master-public-key-unlinkability $\text{Game}_{\text{mpkunl}}$.

4.7. Derived-Public-Key-Unlinkability. Derived-public-key-unlinkability means that with the support of two derived public keys and the corresponding signatures, no one can figure out whether they are derived from the same master public key.

4.7.1. Setup. Same as that of Game_{uf}.

4.7.2. Probing Phase 1. Same as the probing phase of Game_{uf}.

4.7.3. Challenge. $\mathcal{A}$ outputs two distinct indices $1 \le i_0, i_1 \le n$ (λ). A random bit $c \in \{0,1\}$ is chosen. Compute $\text{DPK}_0^* \leftarrow$ `DerivedPublicKeyGen` (MPK_{i_c}).

A random bit $b \in \{0,1\}$ is chosen. If $b = 0$, compute $\text{DPK}_1^* \leftarrow$ `DerivedPublicKeyGen` (MPK_{i_c}); otherwise, compute $\text{DPK}_1^* \leftarrow$ `DerivedPublicKeyGen` $(\text{MPK}_{i_{1-c}})$. $(\text{DPK}_0^*, \text{DPK}_1^*)$ are given to $\mathcal{A}$. Set $L_{dp\,k} = L_{dp\,k}, \cup \{\text{DPK}_0^*, \text{DPK}_1^*\}$.

4.7.4. Probing Phase 2. Same as the probing phase 1, with the restriction that ODPKAdd $(\text{DPK}_j^*, \text{MPK}_{i_k})$ $(j, k \in \{0,1\})$ can only be queried on at most one $j \in \{0,1\}$.

4.7.5. Output Phase. $\mathcal{A}$ outputs a bit b', $\in \{0,1\}$ as its guess to b.

Definition 6. The SALRS is derived-public-key-unlinkable, if for all $\mathcal{A}$, $A\,dv_{\mathcal{A}}^{\text{dpkunl}}$ is negligible, where $A\,dv_{\mathcal{A}}^{\text{dpkunl}} = |\Pr[b = b] - 1/2|$. We name the game for derived-public-key-unlinkability $\text{Game}_{\text{dpkunl}}$.

5. Our Concrete Scheme of SALRS

In this section, as a building block for our SALRS construction, we first introduce our novel concrete key encapsulation mechanism (KEM) based on the elliptic curve discrete logarithm. Then, we propose our concrete SALRS construction.

5.1. KEM Based on Elliptic Curve Discrete Logarithm. Formally, our novel concrete KEM based on the elliptic curve discrete logarithm consists of algorithms as follows.

5.1.1. Setup $(1^\lambda) \longrightarrow$ *Params.* The input to this algorithm is a security parameter 1^λ and outputs system global parameters params.

The params are generated as follows. Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be groups of prime order p, $\mathbb{Z}_p$ be an integer group of order p, g_i be a generator of $(i = 1,2)$, and e: $\mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ be a bilinear, efficient, and computable map. Set $\mathbb{G}_1 \neq \mathbb{G}_2$, and there exists no efficient homomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$. H: $\mathbb{G}_T \longrightarrow \mathbb{Z}_p$ is a collision-resistant hash function. Then, we set params=$(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \mathbb{Z}_p, g_1, g_2, H)$.

5.1.2. KeyGen (params) $\longrightarrow$ *(pk, sk).* The input to this algorithm is the params and output a (public key, secret key) pair (pk, sk).

The pair (pk, sk) is generated as follows. First, choose a random $\alpha \in \mathbb{Z}_p$ and then compute g_1^α. Finally, the pair (pk, sk) is set as (pk, sk)=(g_1^α, α).

5.1.3. Encaps (pk, params) $\longrightarrow$ *(AD, K).* The input to this algorithm is the pk and params, and output a ciphertext AD and a key K. We let AD and $\mathcal{K}$ denote the ciphertext space and key space, respectively.

The pair (AD, K) is generated as follows. First, choose a random $r \in \in \mathbb{Z}_p$, compute the key g_1^r, compute $HV \triangleq H(e(pk, g_2^r))$, and then compute g_1^{HV}. Finally, set $(AD, K) = (g_1^{HV}, g_1^r)$.

5.1.4. Decaps (params, AD, pk, sk) $\longrightarrow$ *K/* $\perp$. The input to this algorithm is the params, ciphertext AD, public key pk, and secret key sk and outputs a key K or a special symbol $\perp$ to indicate rejection.

The $K/\perp$ is generated as follows. If $\exists\ r \in \in \mathbb{Z}_p$, the equation $AD = g_1^{SHV}$ holds, where $SHV \triangleq H(e(g_1^r, g_2^{sk}))$, output a key $K = g_1^r$; otherwise, output a special symbol $\perp$.

5.2. Concrete SALRS Construction

5.2.1. `Setup` $(\lambda) \longrightarrow$ *PP.* The input to this algorithm is a security parameter λ, the algorithm sets the parameters n, q, k, l, m, η, γ, and θ, let H_A: $\{0,1\}^* \mapsto RR_q^{k \times l}$, expandV: $\mathcal{K} \mapsto S_\eta^l$, H_θ: $\{0,1\}^* \mapsto \mathbf{B}_\theta$, and H_m: $R_q^k \mapsto R_q^{m \times l}$ be functions which are random oracles. The algorithm runs:

(1) Set $\mathbf{A} \triangleq H_A$ (cstr), where cstr is a random string belonging to $\{0,1\}^*$

(2) Run params $\leftarrow$ KEM·setup (1^λ).

Output the public parameters, PP = $(n, q, k, l, m, \eta, \gamma, \theta, H_A$, cstr, $\mathbf{A}$, KEM, params, expandV, H_θ, $H_m)$. PP are implicit input parameters to every algorithm as follows.

5.2.2. MasterKeyGen (PP) $\longrightarrow$ *(MPK, MSK).* The input to this algorithm is the PP; the algorithm runs:

(1) (pk, sk)$\leftarrow$KEM·KeyGen (params)

(2) Set $\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}$, where $\mathbf{s} \overset{R}{\leftarrow} S_\eta^l$

Output MPK $\triangleq$ $(pk, \mathbf{t})$ and MSK $\triangleq$ $(sk, \mathbf{s})$.

5.2.3. DerivedPublicKeyGen (MPK) $\longrightarrow$ *DPK.* The input to this algorithm is the MPK = $(pk, \mathbf{t})$; the algorithm runs:

(1) Run $(AD, K) \leftarrow$ KEM·Encaps (pk, params).

(2) Set $\mathbf{s}' \triangleq$ expandV $(K) \in S_\eta^l$, $\mathbf{t}' \leftarrow \mathbf{A}\mathbf{s}'$, and $\hat{\mathbf{t}} \leftarrow \mathbf{t} + \mathbf{t}'$.

Output DPK $\triangleq$ $(AD, \hat{\mathbf{t}})$.

5.2.4. DerivedPublicKeyOwnerCheck (DPK, MPK, MSK) $\longrightarrow$ *1/0.* The input to this algorithm is a DPK, and pair

(MPK, MSK) with MPK = (pk, $\mathbf{t}$) and MSK = (sk, $\mathbf{s}$); the algorithm runs:

(1) If DPK $\in\in$AD $\times$, R_q^k, set DPK $\triangleq$ $(AD, \hat{\mathbf{t}}) \in\in$AD $\times$, R_q^k; otherwise, return 0.

(2) Run $K \leftarrow$ KEM·Decaps (params, AD, pk, sk).

(3) Set $\mathbf{s}' \triangleq$ expandV (K) and $\mathbf{t}' \leftarrow \mathbf{A}\mathbf{s}'$.

If $\hat{\mathbf{t}} = \mathbf{t} + \mathbf{t}'$, return 1; otherwise, return 0.

5.2.5. DerivedPublicKeyPublicCheck (DPK) ⟶ 1/0. The input to this algorithm is DPK; the algorithm runs: if DPK $\in\in$ AD $\times$, R_q^k, return 1; otherwise, return 0.

5.2.6. Sign (M, R, DPK, MPK, MSK) ⟶ σ. The input to this algorithm is a message M, a ring of well-formed derived public keys $R = (\mathrm{DPK}_1, \ldots, \mathrm{DPK}_r)$, a derived public key DPK $\in$ R, and the master key pair for DPK, where MPK = (pk, $\mathbf{t}$) and MSK = (sk, $\mathbf{s}$); the algorithm runs:

(1) Set $\mathrm{DPK}_i \triangleq (AD_i, \hat{\mathbf{t}}_i) \in\in$AD $\times$, R_q^k and $\mathbf{H}_i \triangleq$, H_m $(\hat{\mathbf{t}}_i)$ $(i = 1, \ldots, r)$.

(2) Let $\bar{i}$ be DPK = $\mathrm{DPK}_{\bar{i}} = (AD_{\bar{i}}, \hat{\mathbf{t}}_{\bar{i}})$, run $K \leftarrow$ KEM·**Decaps** (params, $AD_{\bar{i}}$, pk, sk). Set $\mathbf{s}'_{\bar{i}} \triangleq$ expand (K) and $\hat{\mathbf{s}}_{\bar{i}} \leftarrow \mathbf{s} + \mathbf{s}'_{\bar{i}}$.

(3) Use $\mathbf{H}_{\bar{i}}$ and $\hat{\mathbf{s}}_{\bar{i}}$ above, and set $\mathbf{I} \leftarrow \mathbf{H}_{\bar{i}}\hat{\mathbf{s}}_{\bar{i}}$.

(4) Set $\mathbf{w}_{\bar{i}} \leftarrow$ Ay and $\mathbf{v}_{\bar{i}} \leftarrow \mathbf{H}_{\bar{i}}\ \mathbf{y}$, where $\mathbf{y} \overset{R}{\leftarrow} S_\gamma^l$.

(5) Set $c_i \leftarrow H_\theta$ (M, R, $\mathbf{w}_{i-1}$, $\mathbf{v}_{i-1}$, $\mathbf{I}$), where we set $c_1 \leftarrow H_\theta$ (M, R, $\mathbf{w}_r$, $\mathbf{v}_r$, $\mathbf{I}$), and set $\mathbf{w}_i \leftarrow \mathbf{A}\mathbf{z}_i -$, $c_i\hat{\mathbf{t}}_i$ and $\mathbf{v}_i \leftarrow \mathbf{H}_i\mathbf{z}_i - c_i\,\mathbf{I}$, where $\mathbf{z}_i \overset{R}{\leftarrow} S_{\gamma-2\theta\eta}^l$, $i = \bar{i}+1, \cdots, r, 1, \cdots, \bar{i}-1$.

(6) Set $c_{\bar{i}} \leftarrow H_\theta$ (M, R, $\mathbf{w}_{\bar{i}-1}$, $\mathbf{v}_{\bar{i}-1}$, $\mathbf{I}$).

Set $\mathbf{z}_{\bar{i}} \leftarrow$ y $+ c_{\bar{i}}\hat{\mathbf{s}}_{\bar{i}}$. If $\mathbf{z}_{\bar{i}} \in$, $S_{\gamma-2\theta\eta}^l$, output $\sigma \triangleq (cc_1, \{\mathbf{z}_i\}_{i=1}^r, \mathbf{I}) \in$, $\mathbf{B}_\theta \times (SS_{\gamma-2\theta\eta}^l)^r \times R_q^m$; otherwise, return to (4).

5.2.7. Verify (M, R, σ) ⟶ 1/0. The input to this algorithm is a message M, a ring of well-formed derived public keys $R = (\mathrm{DPK}_1, \ldots, \mathrm{DPK}_r)$, and a signature $\sigma = (c_1, \{\mathbf{z}_i\}_{i=1}^r, \mathbf{I})$; the algorithm runs:

(1) If $c_1 \notin \mathbf{B}_\theta$ or $\mathbf{z}_i \notin S_{\gamma-2\theta\eta}^l$, $\exists\ i \in \{1, \ldots, r\}$, return 0.

(2) Set $\mathrm{DPK}_i \triangleq (AD_i, \hat{\mathbf{t}}_i) \in$ (AD, R_q^k) and $\mathbf{H}_i \triangleq \mathbf{H}_m$ $(\hat{\mathbf{t}}_i)$ $(i = 1, \ldots, r)$. Then, set $\mathbf{w}_i \leftarrow$ $\mathbf{A}\mathbf{z}_i -$ $c_i\hat{\mathbf{t}}_i$, $\mathbf{v}_i \leftarrow \mathbf{H}_i$ $\mathbf{z}_i - c_i\,\mathbf{I}$, and $c_{i+1} \leftarrow H_\theta$ (M, R, $\mathbf{w}_i$, $\mathbf{v}_i$, $\mathbf{I}$).

If $c_{r+1} = c_1$, return 1; otherwise, return 0.

5.2.8. Link (M_0, R_0, σ_0, M_1, R_1, σ_1) ⟶ 1/0. The input to this algorithm is two valid (message M, derived public key ring R, and signature σ) tuples (M_0, R_0, σ_0) and (M_1, R_1, σ_1), where $\sigma_0 = (c_1^{(0)}, \{\mathbf{z}_i^{(0)}\}_{i=1}^{r_0}, \mathbf{I}^{(0)})$ and $\sigma_1 = (c_1^{(1)}, \{\mathbf{z}_i^{(1)}\}_{i=1}^{r_1}, \mathbf{I}^{(1)})$; the algorithm runs: if $\mathbf{I}^{(0)} = \mathbf{I}^{(1)}$, return 1; otherwise, return 0.

6. Security Analysis of Our SALRS Construction

Now, we prove that our construction has the usual properties for a SALRS such as correctness, unforgeability, anonymity, linkability, nonslanderability, master-public-key-unlinkability, and derived-public-key-unlinkability.

6.1. Correctness Analysis. It is obvious that from our SALRS construction, (1) and (2) of correctness are satisfied. Therefore, we next prove (3) of correctness. Let $\sigma_j = (c_1^{(j)}, \{\mathbf{z}_i^{(j)}\}_{i=1}^{r_j}, \mathbf{I}^{(j)})$ be generated by sign (M_j, R_j, DPK_j, MPK_j, MSK_j) $(j = 0,1)$ and let $\mathrm{DPK}_i = (AD_i, \hat{t}_i)$.

(1) If $\mathrm{DPK}_0 = \mathrm{DPK}_1$, we have $\hat{\mathbf{s}}_0 = \hat{\mathbf{s}}_1$, and then $\mathbf{I}^{(0)} = \mathbf{I}^{(1)}$. In this case, we have link outputs 1.

(2) If $\mathrm{DPK}_0 \neq \mathrm{DPK}_1$, we now prove that link outputs 0 with overwhelming probability. If $\hat{\mathbf{t}}_0 \neq \hat{\mathbf{t}}_1$, we can see that $\mathbf{H}_m$ $(\hat{\mathbf{t}}_0)$, $\mathbf{H}_m$ $(\hat{\mathbf{t}}_1)$ are distinct. $\hat{\mathbf{s}}_0$ and $\hat{\mathbf{s}}_1$ are distinct. Then we have the result that the probability of $\mathbf{I}^{(0)} = \mathbf{H}_m$ $(\hat{\mathbf{t}}_0)$ $\hat{\mathbf{s}}_0 =$

$\mathbf{H}_m$ $(\hat{\mathbf{t}}_1)$ $\hat{\mathbf{s}}_1 = \mathbf{I}^{(1)}$ is negligible. If $\hat{\mathbf{t}}_0 = \hat{\mathbf{t}}_1$ but $AD_0 \neq AD_1$, we want to prove $\hat{\mathbf{s}}_0 = \hat{\mathbf{s}}_1$. We consider $\hat{\mathbf{s}}_0 \neq \hat{\mathbf{s}}_1$. If $\hat{\mathbf{s}}_0 \neq \hat{\mathbf{s}}_1$ and $\hat{\mathbf{t}}_0 = \mathbf{A}$ $\hat{\mathbf{s}}_0 = \mathbf{A}$ $\hat{\mathbf{s}}_1 = \hat{\mathbf{t}}_1$, its probability is negligible, so we must have $\hat{\mathbf{s}}_0 = \hat{\mathbf{s}}_1$. $\hat{\mathbf{s}}_0 = \hat{\mathbf{s}}_1$ have two cases.

(1) $\mathbf{s}_0 \neq \mathbf{s}_1$ and $\mathbf{s}_0' \neq \mathbf{s}_1'$:

The probability of this scenario is negligible because of the randomness of $\mathbf{s}_0$, $\mathbf{s}_1$, $\mathbf{s}_0'$, and $\mathbf{s}_1'$.

(2) $\mathbf{s}_0 = \mathbf{s}_1$ and $\mathbf{s}_0' = \mathbf{s}_1'$:

The probability of $\mathbf{s}$ $\mathbf{s}_0 = \mathbf{s}_1$ is negligible because two different executions of algorithm `DerivedPublicKeyGen` with $AD_0 \neq AD_1$ will produce distinct $\mathbf{s}_0' = \mathbf{s}_1'$ with overwhelming probability. This completes the correctness analysis.

6.2. Security Analysis. We use $\mathscr{A}$ to denote any probabilistic polynomial time (PPT) adversary in security games.

Theorem 1. *The SALRS construction is linkable.*

Proof. We now prove that our SALRS construction is linkable under module-SIS assumption. If $\mathscr{A}$ succeeds because (3) of linkability holds, it means that $\exists\ i,j \in [1,k]$ and $i \neq j$, $\mathrm{DPK}_{\bar{i}} = \mathrm{DPK}_{\bar{j}}$ where $\mathrm{DPK}_{\bar{i}} \in R_i^*$ and $\mathrm{DPK}_{\bar{j}} \in R_j^*$. Then, we set $\mathrm{DPK}_{\bar{i}} = (AD_{\bar{i}}, \hat{\mathbf{t}}_{\bar{i}})$ and $\mathrm{DPK}_{\bar{j}} = (AD_{\bar{j}}, \hat{\mathbf{t}}_{\bar{j}})$, and we have $\hat{\mathbf{t}}_{\bar{i}} = \hat{\mathbf{t}}_{\bar{j}} = \mathbf{A}$ $\hat{\mathbf{s}}_{\bar{j}} = \mathbf{A}\hat{\mathbf{s}}_{\bar{i}}$. With the support of module-SIS assumption, we have $\hat{\mathbf{s}}_{\bar{i}} = \hat{\mathbf{s}}_{\bar{j}}$ with overwhelming probability, which also means $\mathrm{DPK}_{\bar{i}} = \mathrm{DPK}_{\bar{j}}$ with overwhelming probability. From (1) of linkability and (1) of correctness, we have $\sigma_i^* = \mathrm{sign}\ (M_i^*, R_i^*, \mathrm{DPK}_{\bar{i}}, \mathrm{MPK}_{\bar{i}}, \mathrm{MSK}_{\bar{i}})$ and $\sigma_j^* = \mathrm{sign}\ (M_j^*, R_j^*, \mathrm{DPK}_{\bar{j}}, \mathrm{MPK}_{\bar{j}}, \mathrm{MSK}_{\bar{j}})$. Finally, from (3) of correctness, we can find that (2) of linkability is not satisfied. This completes the proof. □

Theorem 2. *The SALRS construction is nonslanderable.*

Proof. We now prove that our SALRS construction is nonslanderable under the correctness of the SALRS scheme. If $\mathcal{A}$ succeeds, from (1) and (2) of nonslanderability and (1) of correctness, we have $\sigma^* = \text{sign}(M^*, R^*, \text{DPK}^*, \text{MPK}^*, \text{MSK}^*)$ and $\sigma' = \text{sign}(M', R', \text{DPK}', \text{MPK}', \text{MSK}')$. Because of (4) of nonslanderability and (3) of correctness, we have $\text{DPK}^* = \text{DPK}'$ with overwhelming probability. So, we can find that (3) of nonslanderability is not satisfied. This completes the proof. □

Lemma 1. *(See [9]). If a SALRS scheme is linkable and nonslanderable, then it is unforgeable.*

Theorem 3. *The SALRS construction is unforgeable.*

Proof. According to Theorem 1, Theorem 2, and Lemma 1, our SALRS scheme is unforgeable. This completes the proof. □

Theorem 4. *The SALRS construction is anonymous.*

Proof. We now prove that our SALRS construction is anonymous under the decision module-LWE assumption. If $\mathcal{A}$ succeeds, we set $\sigma = (c_1, \{\mathbf{z}_i\}_{i=1}^{r}, \mathbf{I})$ and $\text{DPK}_{i_b} = (AD_{i_b}, \hat{\mathbf{t}}_{i_b})$ ($b \in \{0,1\}$). From algorithm sign, we have $\mathbf{I} = \mathbf{H}_{i_b}\hat{\mathbf{s}}_{i_b}$. It means that $\mathcal{A}$ can distinguish $\mathbf{H}_{i_0}\hat{\mathbf{s}}_{i_0}$ and $\mathbf{H}_{i_1}\hat{\mathbf{s}}_{i_1}$, which contradicts the decision module-LWE assumption. This completes the proof. □

Theorem 5. *The SALRS construction is master-public-key-unlinkable.*

Proof. We now prove that our SALRS construction is master-public-key-unlinkable under the BDH-1 assumption. If $\mathcal{A}$ succeeds, we set $\text{DPK}^* = (AD, \hat{\mathbf{t}})$, $AD = AD_i$, and $\hat{\mathbf{t}} = \hat{\mathbf{t}}_i$ ($i \in \{0,1\}$), that means that $\mathcal{A}$ can distinguish $(AD_i, \hat{\mathbf{t}}_i)$ with a nonnegligible probability. From the algorithm DerivedPublicKeyGen, we have $\hat{\mathbf{t}}_i = \mathbf{t}_i + \mathbf{t}_i'$, $\mathbf{t}_i' = \mathbf{A}\mathbf{s}_i'$, $\mathbf{s}_i' = \text{expandV}(K)$, and $(AD, K) \leftarrow \text{KEM}\cdot\textbf{Encaps}(\text{pk}, \text{params})$, where $K = K_i$. It is obvious that because of the randomness of $r \in \mathbb{G}_1$ and $K = g_1^r$ in the algorithm **Decaps**, $\mathcal{A}$ cannot distinguish K with an overwhelming probability. Therefore, $\mathcal{A}$ cannot distinguish $\mathbf{s}_i'$, so $\mathcal{A}$ cannot distinguish $\hat{\mathbf{t}}_i$ with an overwhelm probability.

We now prove that $\mathcal{A}$ cannot distinguish AD_i with an overwhelming probability too. If $\mathcal{A}$ can distinguish AD_i with a nonnegligible probability ϵ, we can construct a PPT algorithm $\mathcal{B}$ that solves the BDH-1 problem. To be specific, we assume that $\mathcal{A}$ and $\mathcal{B}$ play the game Game_{sec}, and $\mathcal{B}$ simulates the challenger and tries to solve the BDH-1 problem. Suppose the BDH-1 instance (g_1, g_2, g_1^a, g_1^b) is given to $\mathcal{B}$. $\mathcal{B}$ initializes system parameters and gets params=$(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \mathbb{Z}_p, g_1, g_2, H)$ from KEM. Then, $\mathcal{B}$ interacts with $\mathcal{A}$ as follows. □

6.2.1. Setup. $\mathcal{B}$ sends params to $\mathcal{A}$.

6.2.2. Query 1

(1) Key pair query: $\mathcal{A}$ asks $\mathcal{B}$ to use the algorithm KeyGen of KEM to compute a key pair (pk, sk)=(g_1^α, α) and return it to $\mathcal{A}$. $\mathcal{A}$ can only query at most q_1 times for key pairs.

(2) Public key query: $\mathcal{A}$ ask $\mathcal{B}$ to use the algorithm KeyGen of KEM to compute a key pair (pk, sk) and return the public key pk to $\mathcal{A}$. $\mathcal{A}$ can only query at most q_2 times for key pairs.

(3) Hash query: $\mathcal{B}$ sets a hash list Hlist. Hlist is initialized as an empty set. When $\mathcal{A}$ submits a random element $g_T \in \mathbb{G}_T$ to $\mathcal{B}$, $\mathcal{B}$ answers as follows. If it has not appeared in Hlist, $\mathcal{B}$ choose a random element $z \in \in \mathbb{Z}_p$ and returns it to $\mathcal{A}$. Then, $\mathcal{B}$ stores the tuple (g_T, z) in Hlist. Otherwise, $\mathcal{B}$ finds out the tuple (g_T, z) and returns z to $\mathcal{A}$. $\mathcal{A}$ can only query at most q_3 times for hash queries.

6.2.3. Challenge. $\mathcal{A}$ chooses a public key pk^* from (1) of query 1 and sends it to $\mathcal{B}$. $\mathcal{B}$ chooses a random bit δ, $\in \{0,1\}$; if $\delta = 0$, $\mathcal{B}$ sets $K^* = g_1^b$ and computes$AD^* \leftarrow$ Encaps (pk^*, params) and then returns them to $\mathcal{A}$. Otherwise, $\mathcal{B}$ chooses a random element AD^*, $\in \in \mathbb{G}_1$ and sets $K^* = g_1^b$ and returns them to $\mathcal{A}$.

6.2.4. Query 2. $\mathcal{A}$ can make queries as he does in query 1 except secret keys for pk^*.

6.2.5. Guess. Finally, $\mathcal{A}$ outputs a bit δ' as the guess of δ.

If $\mathcal{A}$ can distinguish AD_i, then $\mathcal{A}$ can guess the answer. Then, $\mathcal{B}$ chooses tuple (g_T, z) in Hlist, which satisfies $g_T = e(g_1, g_2)^{ab}$. Then, $\mathcal{B}$ outputs z as the solution to BDH-1 problem. The probability that $\mathcal{B}$ solves the BDH-1 problem is that $Pr[\mathcal{B}\text{succeeds}] = \epsilon \cdot Pr[\mathcal{A}(pk^* = g_1^a)] \cdot Pr[\mathcal{B}(z = e(g_1^a, g_1^b))]$.

If $\mathcal{B}$ succeeds in obtaining a solution of BDH-1 problem, the following conditions must be satisfied:

(1) $Pr[\mathcal{A}(pk^* = g_1^a)] \geq 1/q_2$ ($\mathcal{A}$ correctly chooses pk $*$).

(2) $Pr[\mathcal{B}(z = e(g_1^a, g_2^b))] \geq 1/q_3$ ($\mathcal{B}$ correctly chooses z).

Therefore, we have $Pr[\mathcal{B}\text{succeeds}] \geq \epsilon/(q_2 q_3))$. It means that the probability of the fact that $\mathcal{B}$ solves BDH-1 problem is nonnegligible, which contradicts BDH-1 assumption. This completes the proof.

Lemma 2. *(See [9]). If a SALRS scheme is master-public-key-unlinkable, then it is derived-public-key-unlinkable.*

Theorem 6. *The SALRS construction is derived-public-key-unlinkable.*

Proof. According to Theorem 5 and Lemma 2, our SALRS scheme is derived-public-key-unlinkable. This completes the proof. □

7. Efficiency Analysis

In this section, we make a comparison with the efficiency of the SALRS scheme in [9]. The parameters n, l, k, q, m, θ, η, and γ are set to be same as which in [9], i.e., $n = 256$, $l = 5$, $k = 3$, $q \approx 2^{35}$ and $q = 17 \bmod 32$, $m = 1$, $\theta = 60$, $\eta = 3$, and $\gamma = 699453$. Additionally, the functions H_A, H_m, H_θ, and expandV are set to be same as which in [9], that is to say, we use SHAKE-256 to implement the functions H_A, H_m, and expandV and use the algorithm SampleInBall to implement H_θ. Moreover, with the parameter selection above, we have the fact that in order to obtain the signature in our SALRS scheme, the signer has to run Step 4–Step 6 of algorithm sign at most twice. Because, the probability of restarting of Step 4–Step 6, which can be easily worked out, is $(2\theta\eta/\gamma + 0.5) \approx 1-, e^{-(2nl\theta\eta/\gamma)}$.

From [10, 18, 27], we can obtain an instantiation of KEM where the system global parameters params of KEM are set to be the public parameters of the stealth address scheme in [27]. Especially, the group $\mathbb{G}_1$ in our novel concrete KEM is instantiated to be the special elliptic curve called Ed25519 in [18].

The Ed25519 curve in [18] obviously tells us that in our SALRS scheme, the size of public key pk, secret key sk, or ciphertext AD is $(256–1) \times 2 = 510$ bits, $256 \times 2 = 512$ bits, or $(256–1) \times 2 = 510$ bits, respectively. On the other hand, the efficiency analysis of the SALRS scheme in [9] also tells us that its size of public key, secret key, or ciphertext is 1088 bytes, 2400 bytes, or $(1184–32) = 1152$ bytes, respectively. With the datum above, we can find that the size of public key in our SALRS scheme is smaller than that in [9], which means that from the construction of master public key (MPK), the size of MPK in our SALRS scheme is also smaller than that in [9]. The same applies to the master secret key (MSK) and derived public key (DPK). It comes out a conclusion that with regard to the size of MPK, MSK, and DPK, our SALRS scheme has less storage cost.

8. Conclusion

In this study, the linkable ring signature scheme with stealth addresses were addressed. Then, we proved the security of proposed schemes under the assumptions of BDH-1 problem, module-SIS problem, and module-LWE problem. The results showed that our schemes have all the properties that a linkable ring signature scheme with stealth addresses should have, i.e., unforgeability, anonymity, linkability, nonslanderability, master-public-key-unlinkability, and derived-public-key-unlinkability. Efficiency analysis showed that our SALRS scheme has less storage cost than the SALRS scheme in [9] under the same security conditions.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported in part by the National Key Research and Development Program of China (2021YFA1000600), the City School Joint Funding Project of Guangzhou City (202102010377), and the National Natural Science Foundation of China (61702124).

References

[1] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008, https://bitcoin.org/bitcoin.pdf.

[2] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," *Ieee Access*, vol. 4, pp. 2292–2303, 2016.

[3] T. Li, Y. Chen, Y. Wang et al., "Rational protocols and attacks in blockchain system," *Security and Communication Networks*, vol. 2020, Article ID 8839047, 11 pages, 2020.

[4] T. Li, Z. Wang, G. Yang, Y. Cui, Y. Chen, and X. Yu, "Semi-selfish mining based on hidden Markov decision process," *International Journal of Intelligent Systems*, vol. 36, no. 7, pp. 3596–3612, 2021.

[5] T. Li, Z. Wang, Y. Chen, C. Li, Y. Jia, and Y. Yang, "Is semi-selfish mining available without being detected?" *International Journal of Intelligent Systems*, 2021.

[6] Y. Chen, J. Sun, Y. Yang, T. Li, X. Niu, and H. Zhou, "Psspr: a source location privacy protection scheme based on sector phantom routing in wsns," *International Journal of Intelligent Systems*, 2021.

[7] S. Noether, "Ring signature confidential transactions for monero," *IACR Cryptol. ePrint Arch.*vol. 1098, p. 2015, 2015.

[8] S.-F. Sun, M. H. Au, J. K. Liu, and T. H. Yuen, Ringct 2.0: a compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency monero," in *Proceedings of the Computer Security - ESORICS 2017. In European Symposium on Research in Computer Security*, pp. 456–474, Springer, Oslo, Norway, September 2017.

[9] Z. Liu, K. Nguyen, G. Yang, H. Wang, and D. S. Wong, "A lattice-based linkable ring signature supporting stealth addresses," in *Lecture Notes in Computer Science*, pp. 726–746, Springer, Berlin, Germany, September 2019.

[10] N. Van Saberhagen, *Cryptonote,v. 2.0*, 2013, https://goo.gl/kfojVZ.

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

[12] P. Todd, "Stealth addresses," *Post on Bitcoin development mailing list*, 2014, https://www.mail-archive.com/bitcoin-development@lists.sourceforge.net/msg03613.html.

[13] J. K. Liu, M. H. Au, W. Susilo, and J. Zhou, "Linkable ring signature with unconditional anonymity," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 1, pp. 157–165, 2013.

[14] M. H. Au, S. S. M. Chow, W. Susilo, and P. P. Tsang, "Short linkable ring signatures revisited," in *Proceedings of the Public Key Infrastructure. In European Public Key Infrastructure Workshop*, pp. 101–115, Springer, Turin, Italy, June 2006.

[15] M. H. Au, J. K. Liu, W. Susilo, and T. H. Yuen, "Certificate based (linkable) ring signature," in *Proceedings of the International Conference on Information Security Practice and Experience*, pp. 79–92, Springer, Hong Kong, China, 2007.

[16] M. H. Au, J. K. Liu, W. Susilo, and T. H. Yuen, "Secure id-based linkable and revocable-iff-linked ring signature with constant-size construction," *Theoretical Computer Science*, vol. 469, pp. 1–14, 2013.

[17] L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky, "Lattice Signatures and Bimodal Gaussians," in *Proceedings of the Annual Cryptology Conference Advances in Cryptology - CRYPTO 2013*, pp. 40–56, Springer, Santa Barbara, CA, USA, August 2013.

[18] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, "High-speed high-security signatures," *Journal of cryptographic engineering*, vol. 2, no. 2, pp. 77–89, 2012.

[19] W. A. A. Torres, R. steinfeld, A. sakzad et al., "Post-quantum one-time linkable ring signature and application to ring confidential transactions in blockchain (lattice ringct v1. 0)," in *Proceedings of the Australasian Conference on Information Security and Privacy*, pp. 558–576, Springer, Wollongong, Australia, July 2018.

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

[21] Y. Ren, H. Guan, and Q. Zhao, "An efficient lattice-based linkable ring signature scheme with scalability to multiple layer," *Journal of Ambient Intelligence and Humanized Computing*, vol. 2021, pp. 1–10, 2021.

[22] N. Courtois and R. Mercer, "Stealth address and key management techniques in blockchain systems," in *Proceedings of the 3rd International Conference on Information Systems Security and Privacy ICISSP*, pp. 559–566, Porto, Portugal, January 2017.

[23] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Proceedings of the Advances in Cryptology - CRYPTO 2001*, pp. 213–229, Springer, California, CA, USA, August 2001.

[24] A. Langlois and D. Stehlé, "Worst-case to average-case reductions for module lattices," *Designs, Codes and Cryptography*, vol. 75, no. 3, pp. 565–599, 2015.

[25] S. D. Galbraith, K. G. Paterson, and N. P. Smart, "Pairings for cryptographers," *Discrete Applied Mathematics*, vol. 156, no. 16, pp. 3113–3121, 2008.

[26] M. S. Kiraz and O. Uzunkol, "Still wrong use of pairings in cryptography," 2016, https://www.researchgate.net/publication/301855144_Still_Wrong_Use_of_Pairings_in_Cryptography.

[27] J. Fan, Z. Wang, Y. Luo, J. Bai, Y. Li, and Y. Hao, "A new stealth address scheme for blockchain," in *Proceedings of the ACM Turing Celebration Conference-China*, Chendu, China, pp. 1–7, 2019.