



Article

A Lattice-Based Certificateless Traceable Ring Signature Scheme

Junbin Liang ¹, Jianye Huang ², Qiong Huang ^{1,*}, Liantao Lan ¹ and Man Ho Allen Au ³

¹ College of Mathematics and Informatics, South China Agricultural University, Guangzhou 510642, China
² School of Computing and Information Technology, University of Wollongong, Wollongong 2522, Australia
³ Department of Computing, Hong Kong Polytechnic University, Hong Kong, China
* Correspondence: qhuang@scau.edu.cn

Abstract: A ring signature (RS) scheme enables a group member to sign messages on behalf of its group without revealing the definite signer identify, but this also leads to the abuse of anonymity by malicious signers, which can be prevented by traceable ring signatures (TRS). Up until that point, traceable ring signatures have been secure based on the difficult problem of number-theoretic (discrete logarithms or RSA), but since the advent of quantum computers, traditional traceable ring signatures may no longer be secure. Thus Feng proposed a lattice based TRS, which are resistant to attacks by quantum computers. However, that works did not tackle the certificate management problem. To close this gap, a quantum-resistant certificateless TRS scheme was proposed in the study. To the best of our knowledge, this is the first lattice based certificateless TRS. In detail, a specific TRS scheme was combined with the lattice-based certificateless signature technology to solve the certificate management problem while avoid key escrow problem. Additionally, a better zero-knowledge protocol is used to improve the computational efficiency of the scheme, and by reducing the soundness error of the zero-knowledge protocol, the number of runs of the zero-knowledge protocol is reduced, so that the communication overhead of the scheme is reduced. Under random oracle model, the proposed scheme satisfies tag-linkability, anonymity, exculpability and is secure based on the SIS problem and the DLWE problem. In conclusion, the proposed scheme is more practical and promising in e-voting.

Keywords: post-quantum cryptography; traceable ring signature; certificateless; lattice; zero-knowledge



Citation: Liang, J.; Huang, J.; Huang, Q.; Lan, L.; Au, M.H.A. A Lattice-Based Certificateless Traceable Ring Signature Scheme. *Information* **2023**, *14*, 160. <https://doi.org/10.3390/info14030160>

Academic Editor: Danilo Avola

Received: 21 January 2023
Revised: 26 February 2023
Accepted: 27 February 2023
Published: 2 March 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

With the popularity of the Internet, a variety of intelligent applications have come into being, and the security of data and privacy is an important prerequisite in the information age. RS provides users with unconditional anonymity and does not require a ring administrator for their services. However, in e-voting [1,2], e-cash [3], blockchain [4], Vehicular Ad-Hoc Network (VANET) [5] and some application scenarios [6] that do not require full anonymity, unconditional anonymity allows some malicious users to sign the same event without restriction and without revealing their identity information, which poses certain security risks. For example, in e-voting and e-cash, users can sign multiple times for the same event without being detected, so both e-voting and e-cash require the protection of user anonymity while providing non-reusability, especially in e-cash, where non-reusability prevents double-spending attacks. This is where TRS comes in to meet these needs. The TRS is a balance between the strong traceability of group signatures and the strong anonymity of RS. When a malicious user irresponsibly signs multiple times in the same event, the traceability of the TRS can detect and find the malicious user, and when the user signs honestly, the anonymity remains. Thus, in applications where full anonymity is not required, TRS is more useful.

TRS operates primarily within the framework of PKI [7]. In this case, each user's public-private key pair is generated by themselves, and in order to let others know that the user has a correspondence with their published public key, a trusted third-party authority

needs to be brought in to issue a digital certificate proving the correspondence, which we generally refer to as a CA. As generating and verifying certificates require a certain amount of storage space and computational overhead, the management of certificates requires huge overhead as the number of users increases, which is often referred to as the CMP.

To solve CMP described above and to ease the administrative burden, Shamir et al. [8] designed an identity-based cryptography scheme in 1984. Its aim to optimize public key management by removing public key certificates. Instead of generating a public key, the user simply selects the identity information associated with it himself as the public key, for instance, name, email, etc. The user then simply sends his identity information to KGC and KGC generates private key to the user by identity information. As all of the private keys are produced by the KGC, there is a certain security risk if the KGC is malicious.

To solve KEP, Al-Riyami and Paterson et al. [9] designed a CLPKC scheme in 2003. Under the certificate-free framework, the user's private key is divided into two components, one component is the private key generated by KGC, the other component is a random value of the user's own choice, so KGC cannot get all the keys. Through such a design, CLPKC solves the KEP while removing the public key certificate.

But past CLTRS have been based on number-theoretic assumptions. Facing quantum computers, cryptographic schemes relied on traditional number theory may be able to be broken efficiently, and applications [10] designed around the security of traditional cryptographic schemes can become insecure. In 1994, the Shor algorithm proposed by Shor et al. [11] can break the widely used RSA encryption algorithm, so cryptographic schemes based on the RSA hard assumption may become less secure. Not only that, there are some quantum algorithms that can crack other difficult problems in number theory, the traditional cryptographic schemes may present security risks, so cryptographic schemes based on number theory will face great challenges. To address the security concerns posed by quantum computers, post-quantum cryptographic schemes and applications based on post-quantum cryptographic schemes have emerged successively, these include isogeny-based cryptographic scheme [12], the supersingular isogeny key encapsulation protocol [13], the variant of SABER and Falcon [14], the signature scheme CRYSTALS-Dilithium [15], lattice cryptography also being one of post-quantum cryptographic schemes. In 1996, Ajtai et al. [16] pioneered to prove that the difficulty of lattice-hard problems is equivalent in the average case to the worst case, which makes lattice-based cryptosystem have a theoretical basis for providing security proofs. In addition, lattice ciphers can build complex applications. Therefore, this paper aims to design a certificate-less traceable ring signatures on lattice, which is important for application background such as VAENT and e-voting.

1.1. Related Works

RS is first proposed by Rivest et al. [17] in 2001, it differs from group signatures in that it can be formed spontaneously, thus eliminating the need for group administrators. Subsequently, research on RS has grown and various RS schemes [18–23] have been proposed. Although RS provides anonymity to the users in the ring, they also facilitate malicious users, who can sign an event without restriction without revealing their identity. In some scenarios (e.g., electronic voting), we don't want that to happen. Figure 1 is the signature process of the RS.

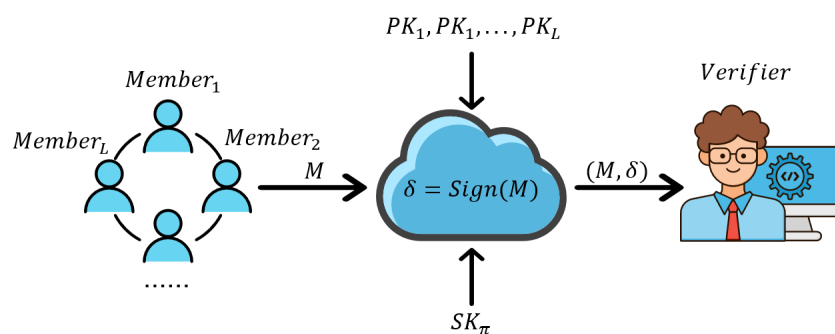


Figure 1. Ring Signature.

To limit the degree of anonymity of RS, Liu et al. [24] in 2004 constructed a LRS, it can link two signatures which signed by the same member. LRS limits the abuse of anonymity to a certain extent. Various improvements to LRS schemes [3,25–30] have subsequently been proposed. Although LRS can limit anonymity to some extent, honest users still have no way of knowing exactly which user is maliciously signing in multiple times.

In order to trace the identity of the malicious user, TRS emerged. This concept was first proposed by Fujisaki et al. [31] in 2007, where each user can only sign at most once under the same tag, and when a user signs two different messages, they are tracked by the tracking algorithm and their identity is revealed. Thus, TRS can make honest members anonymous while tracking malicious members, making TRS useful in many applications. Various improvements to TRS schemes [4,6,32–36] have also been proposed. Of these, ref. [32] improves the scheme of [31] by reducing the signature size and communication overhead. The relationship between the signature and ring sizes is linear in [31], while the signature size in [32] is square-root related to the ring size. Ref. [33] proposed an identity-based TRS, so that TRS does not have to bear huge certificate management overhead. Ref. [34] extends the number of times users can sign on the same label in TRS, allowing users to sign for K times. Only when the number of signatures exceeds K times, the user is traced. Ref. [6] constructs a TRS based on certificate-less, which eliminates the overhead of CMP of [31] and solves the KEP of [32], and it is proved security under the SM. Ref. [35] improves the security model on the basis of [6]. Compared with [6], ref. [35] does not use bilinear pairing, so the computation cost is reduced. Ref. [36] proposes an identity-based TRS whose signature size is constant. The TRS constructed by [4] supports multi-user authorization.

RS, LRS and TRS relied on number theory have made a lot of achievements. With the maturation of quantum computers, cryptography schemes that rely on number theory may have security risks. Therefore, it is of great significance to construct cryptographic schemes that can resist quantum computer attacks. Hard problems with lattices is able to achieve worst-case to average-case reductions, and many cryptographic primitives can be constructed using lattices. Therefore, lattice-based cryptographic schemes have become the focus of many researchers.

In 2010, Brakerski et al. [37] designed a generic construction for RS and instantiated the generic construction, proposing the first lattice-based RS. Two years later, Tian et al. [38] and Lyubashevsky et al. [39] each came up with a lattice-based RS, where Lyubashevsky's scheme does not use algorithm to get trapdoors, but draws directly from inside the normal distribution, so it will run more efficiently. Libert et al. [40] designed a RS with a Merkle tree based accumulator and a Stern-like protocol in 2016, its signature size has been reduced to a logarithmic relationship with the ring size from the original linear relationship. After two years, Wang et al. [41] improves on [39] by designing a RS scheme without trapdoors, and the scheme is more secure and efficient. In 2019, Lu et al. [42] proposes a practical RS called Raptor, which suitable for small-scale rings.

Melchor et al. [43] designed a LRS from lattice, which based on [39] by combining the Fiat-Shamir model. Based on the core idea of [40], Yang et al. [44] designed a LRS on lattice in 2017. Torres et al. [45] and Baum et al. [46] both constructed a one-time LRS from lattice in 2018, respectively, and their schemes can be regarded as a migration of discrete

logarithm based LRS [24] on lattice. Lu et al. [42] provided a new idea to design one-time LRS in 2019. In 2021, Le et al. [47] constructed two identity-based LRS, one constructed on standard lattice and the other on ideal lattice. Hu et al. [48] constructs a lattice-based LRS by designing a new lattice basis extending algorithm in 2022. Ye et al. [49] designed an LRS based on the NTRU lattice to optimize the signature size and improve the efficiency of the signature algorithm.

In 2020, Feng et al. [50] constructs a general framework for TRS and designs the first lattice-based TRS by instantiating this framework on standard lattices.

In summary, there are fewer lattice-based TRS schemes, and the existing lattice-based TRS have CMP, which may be hindered in application to practical scenarios.

1.2. Our Contributions

In this paper, in order to effectively remove the burden of certificate management in [50] and prevent KEP, we designed an efficient CLTRS on lattice.

1. The security of most TRS schemes relies on difficult assumptions in number theory, and the security of these schemes may become insecure for quantum computers, so we designed a lattice-based CLTRS. In our scheme, we treat the member's identity information as the member's public key, thus effectively solving the CMP in [50]. Meanwhile, we divide the member's private key into two parts, one produced by KGC and one chosen by member, and by doing so, our scheme also does not have the KEP like in [29,36,47].
2. Based on SIS, DLWE hard assumptions, uniqueness of function F and multi-input correlation intractability of hash family, our scheme is proved to be security and it satisfies the tag-linkability, type I anonymity, type II anonymity, type I exculpability and type II exculpability under the ROM. Furthermore, compared to [51,52], although the signature size of our scheme is larger, our member secret key is smaller and also achieves public traceability, which the identity will no longer be anonymous if the member maliciously signs twice under the same label.
3. Our scheme combines the efficient zero-knowledge protocol proposed by [53], and to argue the relation that our scheme wants to prove, we make modifications based on [53] to construct the privacy-preserving primitives suitable for our scheme. The Stern protocol used in [50] has a soundness error of $2/3$ for single run protocol, while the efficient zero-knowledge protocol used in our scheme has a soundness error of $1/\text{poly}$ for single run protocol. To obtain negligible soundness error, our scheme repeats the zero-knowledge protocol fewer times than [50]. Thus, compared to the scheme in the [50], the size of the zero-knowledge proof generated by our scheme will be smaller for the same parameters. When computing the root node of the Merkle tree, [50] retains only one auxiliary node at each level, therefore, for the verification of the path from leaf node to root node, the parent node of leaf node needs to be calculated by the leaf node and its sibling nodes first, and then the parent node of the upper layer can be calculated by the sibling nodes of the layer where the parent node and the parent node reside, and so on, and finally the root node can be obtained. This verification process is a serial calculation process. In this paper, two auxiliary nodes are reserved in each layer of the tree. Therefore, when calculating the root node, the process of calculating the left and right child nodes of each layer to obtain the parent node can be calculated in parallel, and there is no need to calculate step by step from the lowest leaf node. Therefore, the verification process of our scheme is more efficient.

2. Certificate-Less Traceable Ring Signature

2.1. Definition

An CLTRS consists of eight efficient algorithms as follows:

$(pp, msk) \leftarrow \mathbf{CLTRS.Setup}(1^\lambda)$: Input the security parameter 1^λ , and outputs a public system parameter pp and a master secret key msk .

- $s_i \leftarrow \text{ExtractPartialPrivateKey}(pp, id_i, msk)$: Input pp , an identity $id_i \in \{0, 1\}^*$ of user U_i for $i \in [L]$, and msk , then generate a partial private key s_i with respect to id_i .
- $v_i \leftarrow \text{SetSecretValue}(pp, id_i)$: Input $pp, id_i \in \{0, 1\}^*$ of user U_i for $i \in [L]$, then returns a secret value v_i to the user U_i .
- $sk_i \leftarrow \text{SetPrivateKey}(pp, s_i, v_i)$: Input pp, s_i and v_i , then returns a secret key sk_i corresponding to the user U_i .
- $y_i \leftarrow \text{SetPublicKey}(pp, sk_i)$: Input pp and $sk_i = (s_i, v_i)$, then returns a public key y_i corresponding to the user U_i .
- $\delta \leftarrow \text{CLTRS.Sign}(pp, I, R, M, id, sk_{id})$: Input pp , an issue I , a ring R , a message M , an identity id and sk_{id} , outputs a ring signature δ .
- $b \leftarrow \text{CLTRS.Verify}(pp, I, R, M, \delta)$: Input pp , an issue I , a ring R , a message M and a ring signature δ , return $b = 1$ if accepting the signature or $b = 0$ for rejecting it.
- $\eta \leftarrow \text{CLTRS.Trace}(pp, I, R, M, \delta, M', \delta')$: Input pp , an issue I , a ring R , and two valid tuples (M, δ) and (M', δ') , outputs $\eta \in \{accept, linked, id\}$.

Definition 1 (Completeness). *An CLTRS scheme satisfies complete, if for $(pp, msk) \leftarrow \text{CLTRS.Setup}(1^\lambda), s_i \leftarrow \text{ExtractPartialPrivateKey}(pp, id_i, msk), v_i \leftarrow \text{SetSecretValue}(pp, id_i), sk_i \leftarrow \text{SetPrivateKey}(pp, s_i, v_i), y_i \leftarrow \text{SetPublicKey}(pp, sk_i), \delta \leftarrow \text{CLTRS.Sign}(pp, I, R, M, id, sk_{id})$, there is $1 \leftarrow \text{CLTRS.Verify}(pp, I, R, M, \delta)$ holds. That is,*

$$\Pr \left[\begin{array}{l} 0 \leftarrow \text{CLTRS.Verify}(pp, I, R, M, \delta) : \\ (pp, msk) \leftarrow \text{CLTRS.Setup}(1^\lambda), \\ s_i \leftarrow \text{ExtractPartialPrivateKey}(pp, id_i, msk), \\ v_i \leftarrow \text{SetSecretValue}(pp, id_i), \\ sk_i \leftarrow \text{SetPrivateKey}(pp, s_i, v_i), \\ y_i \leftarrow \text{SetPublicKey}(pp, sk_i), \\ \delta \leftarrow \text{CLTRS.Sign}(pp, I, R, M, id, sk_{id}). \end{array} \right] \leq \text{negl}(n).$$

Definition 2 (Public Traceability). *An CLTRS scheme satisfies publicly traceable, if for $(pp, msk) \leftarrow \text{CLTRS.Setup}(1^\lambda), s_i \leftarrow \text{ExtractPartialPrivateKey}(pp, id_i, msk), v_i \leftarrow \text{SetSecretValue}(pp, id_i), sk_i \leftarrow \text{SetPrivateKey}(pp, s_i, v_i), y_i \leftarrow \text{SetPublicKey}(pp, sk_i), \delta \leftarrow \text{CLTRS.Sign}(pp, I, R, M, id, sk_{id})$ and $\delta' \leftarrow \text{CLTRS.Sign}(pp, I, R, M', id', sk_{id'})$, there is overwhelming probability that*

$$\text{CLTRS.Trace}(pp, I, R, M, \delta, M', \delta') = \begin{cases} \text{accept} & , \text{if} & id \neq id', \\ \text{linked} & , \text{else if} & M = M', \\ id & , \text{otherwise} & M \neq M', \end{cases}$$

holds.

2.2. Security Models

In the framework of the certificateless, two categories of adversaries exist. The first type of adversary \mathcal{A}_I is an external adversary, which can replace the user’s public key but cannot access the system master secret key; the second type of adversary \mathcal{A}_{II} is an internal adversary, which can control the KGC and direct the generation of the system secret key but cannot replace the target user’s public key.

A secure CLTRS scheme should satisfy *tag-linkability*, type I *anonymity*, type II *anonymity*, type I *exculpability* and type II *exculpability*. The following are definitions of these security properties.

Definition 3 (Tag-Linkability). *Tag-Linkability means that under the same label, for any PPT adversary \mathcal{A} , even if it has L ring member secret keys, it cannot generate $L + 1$ valid signatures*

satisfying that any two signatures are generated by different secret keys. Since an adversary \mathcal{A} in tag linkability can know all of the user's secret keys, there is no need to consider distinguishing the type of adversary in this security model. We define the probability of \mathcal{A} winning the following games as the \mathcal{A} 's advantage.

1. **Setup.** Execute $\text{CLTRS.Setup}(1^\lambda)$ algorithm, the challenger \mathcal{C} obtains pp and msk , and then sends pp to \mathcal{A} but msk is saved itself.
2. **Query.** \mathcal{A} conducts the following queries:
 - $\text{CreateUserQuery}(pp, id_i)$. When the adversary \mathcal{A} submits a user id_i , \mathcal{C} checks it in $\text{list}_{\text{CU}}(id_i, s_i, v_i, sk_i, y_i)$, if id_i exists, then returns (pk_i, sk_i) ; otherwise, \mathcal{C} runs algorithms $\text{ExtractPartialPrivateKey}$, SetSecretValue , SetPrivateKey , SetPublicKey to generate s_i, v_i, sk_i and y_i , then returns y_i to adversary \mathcal{A} and adds tuple $(id_i, s_i, v_i, sk_i, y_i)$ to list_{CU} .
 - $\text{PartialPrivateKeyQuery}(pp, id_i)$. When the adversary \mathcal{A} submits a user id_i , \mathcal{C} checks id_i in the list_{CU} and returns s_i to \mathcal{A} .
 - $\text{ReplacePublicKeyQuery}(pp, id_i, y'_i)$. When the adversary \mathcal{A} submits the user id_i and y_i , \mathcal{C} substitutes y_i for y'_i .
 - $\text{SignQuery}(pp, id_i, I, R, M)$. When the adversary \mathcal{A} submits the user id_i , issue I , ring R and message M , \mathcal{C} return $\delta \leftarrow \text{CLTRS.Sign}(pp, sk_i, I, R, M)$ to \mathcal{A} .
3. **Forgery.** \mathcal{A} forgeries $L + 1$ tuples (I, R, M_h, δ_h) with $h \in \{1, \dots, L + 1\}$. \mathcal{A} wins if
 - (a) $1 \leftarrow \text{CLTRS.Verify}(pp, I, R, M_h, \delta_h)$ for $\forall h \in \{1, \dots, L + 1\}$;
 - (b) $\text{accept} \leftarrow \text{CLTRS.Trace}(pp, I, R, M_k, \delta_k, M_h, \delta_h)$ for all $k, h \in \{1, \dots, L + 1\}$ with $k \neq h$.
 - (c) \mathcal{A} has at most L ring member sk , and the pk of these ring members are all in tag Γ .

The probability of \mathcal{A} winning the game holds in relation to \mathcal{A} 's advantage as follows

$$\text{Adv}_{\mathcal{A}}^{\text{TaL}} = \Pr[\mathcal{A} \text{ wins}].$$

Anonymity means that given any valid signature, it is hard for anyone to realize the identity of the signer. There is a need to distinguish between \mathcal{A}_I and \mathcal{A}_{II} in the security model of anonymity, so a secure CLTRS, needs to satisfy both type I Anonymity and type II Anonymity.

Definition 4 (Type I Anonymity). *if a CLTRS is type I Anonymity, the advantage of \mathcal{A}_I to win the following games is negligible.*

1. **Setup.** Execute $\text{CLTRS.Setup}(1^\lambda)$ algorithm, the challenger \mathcal{C} obtains pp and msk , and saves msk secretly, \mathcal{A}_I obtains pp from \mathcal{C} .
2. **Query1.** \mathcal{A}_I conducts the following four kinds of queries:
 - $\text{CreateUserQuery}(pp, id_i)$. When the adversary \mathcal{A}_I submits a user id_i , \mathcal{C} checks it in $\text{list}_{\text{CU}}(id_i, s_i, v_i, sk_i, y_i)$, if id_i exists, then returns y_i ; otherwise, \mathcal{C} runs algorithms $\text{ExtractPartialPrivateKey}$, SetSecretValue , SetPrivateKey , SetPublicKey to generate s_i, v_i, sk_i and y_i , then returns y_i to adversary \mathcal{A}_I and adds tuple $(id_i, s_i, v_i, sk_i, y_i)$ to list_{CU} .
 - $\text{PartialPrivateKeyQuery}(pp, id_i)$. When the adversary \mathcal{A}_I submits a user id_i , \mathcal{C} checks id_i in the list_{CU} and returns s_i to \mathcal{A}_I .
 - $\text{ReplacePublicKeyQuery}(pp, id_i, y'_i)$. When the adversary \mathcal{A}_I submits the user id_i and y_i , \mathcal{C} substitutes y_i for y'_i .
 - $\text{SignQuery}(pp, id_i, I, R, M)$. When the adversary \mathcal{A}_I submits the user id_i , issue I , ring R and message M , \mathcal{C} return $\delta \leftarrow \text{CLTRS.Sign}(pp, sk_i, I, R, M)$ to \mathcal{A}_I .
3. **Challenge.** \mathcal{A}_I refers two tuples (M^*, I^*, R^*, id_0) and (M^*, I^*, R^*, id_1) to \mathcal{C} , where $id_0, id_1 \in R^*$ such that $\text{PPKQ}(pp, id_0), \text{PPKQ}(pp, id_1), \text{SQ}(pp, id_0, I^*, R^*, \cdot)$ and $\text{SQ}(pp, id_1, I^*, R^*, \cdot)$ have not been referred to **Query 1**. \mathcal{C} chooses randomly $b \xleftarrow{\$} \{0, 1\}$, returns $\delta \leftarrow \text{CLTRS.Sign}(pp, sk_b, I^*, R^*, M)$ to \mathcal{A}_I .

4. **Query2.** Similarly to **Query 1**, in addition to \mathcal{A}_I does not have access to $PPKQ(pp, id_0)$, $PPKQ(pp, id_1)$, $SQ(pp, id_0, I^*, R^*, \cdot)$ and $SQ(pp, id_1, I^*, R^*, \cdot)$.
5. **Guess.** \mathcal{A}_I outputs a guess b' for b . \mathcal{A} wins if $b'=b$.

The probability of \mathcal{A}_I winning the game holds in relation to \mathcal{A}_I 's advantage as follows

$$Adv_{\mathcal{A}_I}^{anon} = |\Pr[b' = b] - 1/2|.$$

Definition 5 (Type II Anonymity). if a CLTRS is type II Anonymity, the advantage of \mathcal{A}_{II} winning the following game is negligible.

1. **Setup.** Execute $CLTRS.Setup(1^\lambda)$ algorithm, the challenger \mathcal{C} obtains pp and msk , and then sends them to \mathcal{A}_{II} .
2. **Query1.** \mathcal{A}_{II} conducts the following four kinds of queries:
 - $CreateUserQuery(pp, id_i)$. The same as CUQ of type I Anonymity.
 - $PartialPrivateKeyQuery(pp, id_i)$. The same as RPKQ of type I Anonymity.
 - $ReplacePublicKeyQuery(pp, id_i, y'_i)$. The same as SQ of type I Anonymity..
 - $SignQuery(pp, id_i, I, R, M)$. When the adversary \mathcal{A}_{II} submits the user id_i , issue I , ring R and message M , \mathcal{C} return $\delta \leftarrow CLTRS.Sign(pp, sk_i, I, R, M)$ to \mathcal{A}_{II} .
3. **Challenge.** \mathcal{A}_{II} refers two tuples (M^*, I^*, R^*, id_0) and (M^*, I^*, R^*, id_1) to \mathcal{C} , where $id_0, id_1 \in R^*$ such that $RPKQ(pp, id_0)$, $RPKQ(pp, id_1)$, $SQ(pp, id_0, I^*, R^*, \cdot)$ and $SQ(pp, id_1, I^*, R^*, \cdot)$ have not been referred to **Query 1**. \mathcal{C} chooses randomly $b \xleftarrow{\$} \{0, 1\}$, returns $\delta \leftarrow CLTRS.Sign(pp, sk_b, I^*, R^*, M)$ to \mathcal{A}_{II} .
4. **Query2.** Similarly to **Query 1**, in addition to \mathcal{A}_{II} does not have access to $RPKQ(pp, id_0)$, $RPKQ(pp, id_1)$, $SQ(pp, id_0, I^*, R^*, \cdot)$ and $SQ(pp, id_1, I^*, R^*, \cdot)$.
5. **Guess.** \mathcal{A}_{II} outputs a guess b' for b . \mathcal{A} wins if $b'=b$.

The probability of \mathcal{A}_{II} winning the game holds in relation to \mathcal{A}_{II} 's advantage as follows

$$Adv_{\mathcal{A}_{II}}^{anon} = |\Pr[b' = b] - 1/2|.$$

Exculpability implies that anonymity is lost only if the identical ring member signs different messages under the identical label. There is a need to distinguish between \mathcal{A}_I and \mathcal{A}_{II} in the security model of exculpability, so a secure CLTRS, needs to satisfy both type I exculpability and type II exculpability.

Definition 6 (Type I Exculpability). if a CLTRS is type I exculpability, the advantage of \mathcal{A}_I winning the following game is negligible.

1. **Setup.** Execute $CLTRS.Setup(1^\lambda)$ algorithm, the challenger \mathcal{C} obtains pp and msk , and saves msk secretly, \mathcal{A}_I obtains pp from \mathcal{C} .
2. **Query.** \mathcal{A}_I conducts the following four kinds of queries:
 - $CreateUserQuery(pp, id_i)$. When the adversary \mathcal{A}_I submits a user id_i , \mathcal{C} generates s_i , v_i , sk_i and y_i , returns (pk_i, sk_i) to adversary \mathcal{A}_I , and then adds tuple $(id_i, s_i, v_i, sk_i, y_i)$ to $list_{CU}$.
 - $PartialPrivateKeyQuery(pp, id_i)$. When the adversary \mathcal{A}_I submits a user id_i , \mathcal{C} checks id_i in the $list_{CU}$ and returns s_i to \mathcal{A}_I .
 - $ReplacePublicKeyQuery(pp, id_i, y'_i)$. When the adversary \mathcal{A}_I submits the user id_i and the public key y_i , \mathcal{C} substitutes y_i for y'_i .
 - $SignQuery(pp, id_i, I, R, M)$. When the adversary \mathcal{A}_I submits the user id_i , issue I , ring R and message M , \mathcal{C} return $\delta \leftarrow CLTRS.Sign(pp, sk_i, I, R, M)$ to \mathcal{A}_I .
3. **Forgery.** \mathcal{A}_I forgeries two tuples $(I^*, R^*, \overline{M}, \overline{\delta})$ and (I^*, R^*, M', δ') . It wins if
 - (a) $1 \leftarrow CLTRS.Verify(pp, I^*, R^*, \overline{M}, \overline{\delta})$;
 - (b) $1 \leftarrow CLTRS.Verify(pp, I^*, R^*, M', \delta')$;
 - (c) \mathcal{A}_I has not been queried about $PPKQ(pp, id^*)$ and $CUQ(pp, id^*)$ where $id^* \in R^*$;
 - (d) \mathcal{A}_I has made at most one of $SQ(pp, id^*, I^*, R^*, \overline{M})$ and $SQ(pp, id^*, I^*, R^*, M')$;

$$(e) \quad pk^* \leftarrow \text{CLTRS.Trace}(pp, I^*, R^*, \overline{M}, \overline{\delta}, M', \delta').$$

The probability of \mathcal{A}_I winning the game holds in relation to \mathcal{A}_I 's advantage as follows

$$Adv_{\mathcal{A}_I}^{Excul} = \Pr[\mathcal{A}_I \text{ wins}].$$

Definition 7 (Type II Exculpability). *if a CLTRS is type II exculpability, the advantage of \mathcal{A}_{II} winning the following game is negligible.*

1. **Setup.** Execute $\text{CLTRS.Setup}(1^\lambda)$ algorithm, the challenger \mathcal{C} obtains pp and msk , and sends them to \mathcal{A}_{II} .
2. **Query.** \mathcal{A}_{II} conducts the following four kinds of queries:
 - $\text{CreateUserQuery}(pp, id_i)$. The same as CUQ of type I exculpability.
 - $\text{PartialPrivateKeyQuery}(pp, id_i)$. The same as PPKQ of type I exculpability.
 - $\text{ReplacePublicKeyQuery}(pp, id_i, y'_i)$. The same as RPKQ of type I exculpability.
 - $\text{SignQuery}(pp, id_i, I, R, M)$. The same as SQ of type I exculpability.
3. **Forgery.** \mathcal{A}_{II} forgeries two tuples $(I^*, R^*, \overline{M}, \overline{\delta})$ and (I^*, R^*, M', δ') . It wins if
 - (a) $1 \leftarrow \text{CLTRS.Verify}(pp, I^*, R^*, \overline{M}, \overline{\delta})$;
 - (b) $1 \leftarrow \text{CLTRS.Verify}(pp, I^*, R^*, M', \delta')$;
 - (c) \mathcal{A}_{II} has not been queried about $\text{RPKQ}(pp, id^*)$ and $\text{CUQ}(pp, id^*)$ where $id^* \in R^*$;
 - (d) \mathcal{A}_{II} has made at most one of $\text{SQ}(pp, id^*, I^*, R^*, \overline{M})$ and $\text{SQ}(pp, id^*, I^*, R^*, M')$;
 - (e) $pk^* \leftarrow \text{CLTRS.Trace}(pp, I^*, R^*, \overline{M}, \overline{\delta}, M', \delta')$.

The probability of \mathcal{A}_{II} winning the game holds in relation to \mathcal{A}_{II} 's advantage as follows

$$Adv_{\mathcal{A}_{II}}^{Excul} = \Pr[\mathcal{A}_{II} \text{ wins}].$$

As proved in [31], if a TRS implements both tag-linkability and exculpability, then it also implements unforgeability. This also holds for CLTRS.

3. Preliminaries

3.1. Hardness Assumptions

Definition 8 (Short Integer Solution Problem [16]). *Let $n, m, q \in \mathbb{N}$, and a real number $\beta \in \mathbb{R}$, the SIS problem is: For an arbitrary random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, searching a non-zero vector $\mathbf{s} \in \mathbb{Z}^m$, such that $\mathbf{A} \cdot \mathbf{s} = 0 \pmod q$ and $\|\mathbf{s}\|_\infty \leq \beta$.*

Definition 9 (Decisional Learning With Errors Problem [54]). *Let $n, q \geq 2, \mathbf{u} \in \mathbb{Z}_q^n$. Given k instances, DLWE problem is to identify whether the k instances are drawn from a random distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ or from distribution $\mathbf{A}_{\mathbf{u}, \beta}$, where $\mathbf{A}_{\mathbf{u}, \beta}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ is achieved by picking a vector in $\mathbf{x} \xleftarrow{\$} \mathbb{Z}_q^n$ and a positive real number in $t \xleftarrow{\$} \odot$, and then returning $(\mathbf{x}, y = \mathbf{x} \cdot \mathbf{u} + t)$.*

Definition 10 (Decisional Learning With Rounding Problem [55]). *Given k instances, DLWR problem is to identify whether the k instances are drawn from a random distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_p$ or from distribution \mathbf{A}_β , where \mathbf{A}_β over $\mathbb{Z}_q^n \times \mathbb{Z}_p$ is achieved by picking a vector in $\mathbf{x} \xleftarrow{\$} \mathbb{Z}_q^n$ and picking a integer in $\beta \xleftarrow{\$} \mathbb{Z}_p$, and then returning $(\mathbf{x}, \mathbf{b} = \lfloor \mathbf{x}^T \cdot \mathbf{fi} \rfloor_p)$. For any $\mathbf{x}' \in \mathbb{Z}_q^m$, we denote $\lfloor \mathbf{x}' \rfloor_p$ as $\lfloor (p/q)\mathbf{x}' \rfloor \pmod p$.*

3.2. Trapdoor Mechanism

Definition 11 (G-trapdoor [56]). *Let $\hat{m}, q, n, k \in \mathbb{N}$ and $\mathbf{A} \in \mathbb{Z}_q^{\hat{m} \times n}, \mathbf{G} \in \mathbb{Z}_q^{\hat{m} \times \hat{m}k}$ be matrices with $n \geq \hat{m}k$. Define $\mathbf{H} \in \mathbb{Z}_q^{\hat{m} \times \hat{m}}$ as an invertible matrix. The matrix \mathbf{A} and its corresponding*

G-trapdoor \mathbf{R} satisfy the following constraints: $\mathbf{A} \begin{bmatrix} \mathbf{R} \\ \mathbf{I}_{\hat{m}k} \end{bmatrix} = \mathbf{HG} \pmod q$.

We define $\mathbf{G} = \mathbf{I}_n \otimes \mathbf{g}^t \in \mathbb{Z}_q^{\hat{m} \times \hat{m}k}$, where $k = \lceil \log_2 q \rceil$, $\mathbf{g}^t = (1, 2, \dots, 2^{k-1}) \in \mathbb{Z}^{\hat{m} \times \hat{m}}$, $\mathbf{I}_{\hat{m}} \in \mathbb{Z}^{\hat{m} \times \hat{m}}$ and \otimes denotes the tensor product. In the following, let $q \geq 2$, $\bar{m} \geq 1$, and $n = \bar{m} + \hat{m}k$.

- (GenTrap Algorithm [56]): Given a uniformly random matrix $\bar{\mathbf{A}} \in \mathbb{Z}_q^{\hat{m} \times \bar{m}}$ and an invertible matrix $\mathbf{H} \in \mathbb{Z}_q^{\hat{m} \times \hat{m}}$, the PPT algorithm outputs a random matrix $\mathbf{A} = \lceil \bar{\mathbf{A}} \rceil \mathbf{H} \mathbf{G} - \bar{\mathbf{A}} \mathbf{R}$ and a G-trapdoor $\mathbf{R} \sim D_{\sigma}^{\bar{m} \times \hat{m}k}$ (the \sim indicates that the distribution of G-trapdoor \mathbf{R} obeys the Gaussian distribution $D_{\sigma}^{\bar{m} \times \hat{m}k}$). Also, $s_1(\mathbf{R}) \leq \sigma \cdot \frac{1}{\sqrt{2\pi}} \cdot (\sqrt{\bar{m}} + \sqrt{\hat{m}k})$.
- (SampleD Algorithm [56]): Given a G-trapdoor $\mathbf{R} \in \mathbb{Z}_q^{\bar{m} \times \hat{m}k}$ for $\mathbf{A} \in \mathbb{Z}_q^{\hat{m} \times (\bar{m} + \hat{m}k)}$, an invertible matrix $\mathbf{H} \in \mathbb{Z}_q^{\hat{m} \times \hat{m}}$, a uniform vector $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^{\hat{m}}$ and Gaussian parameter $\sigma \geq \sqrt{7(s_1(\mathbf{R}))^2 + 1} \cdot \omega(\sqrt{\log \hat{m}})$, the PPT algorithm outputs a vector $\mathbf{e} \in \mathbb{Z}^{\bar{m} + \hat{m}k}$ sampled from a distribution that is statistically close to $D_{\Lambda_{\mathbf{u}}^{\perp}(\mathbf{A}), \sigma}$.

For simplicity we set \mathbf{H} to the unit matrix \mathbf{I}_n and omit it.

3.3. Pseudorandom Function Family

Let $n, p, q, m \in \mathbb{N}$ which are polynomial in security parameter λ and $H : \{0, 1\}^* \rightarrow \mathbb{Z}_q^{m \times n}$. The specific description of the PRF is as follows:

- KeyGen. The key generation algorithm randomly chooses a vector $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^n$ and outputs the vector \mathbf{u} .
- Eval. Given a string Γ of any length, the evaluation algorithm returns $F^H(\mathbf{u}, \Gamma) = \lfloor H(\Gamma) \cdot \mathbf{u} \rfloor_p$.

Definition 12 (PRF in the QROM [50]). For any quantum PPT adversary \mathcal{A} , if $F^H : \mathcal{S} \times \mathcal{U} \rightarrow \mathcal{T}$ satisfies pseudorandomness under QROM, then the following conclusion is satisfied.

$$\left| \Pr \left[\mathcal{A}^{F^H(s, \cdot), H(\cdot)}(1^n) = 1 : s \leftarrow \mathcal{S} \right] - \Pr \left[\mathcal{A}^{\Theta(\cdot), H(\cdot)}(1^n) = 1 : \Theta \leftarrow \mathcal{F}[\mathcal{U} : \mathcal{T}] \right] \right| \leq \text{negl}(n),$$

where H is a QROM and $\mathcal{F}[\mathcal{U} : \mathcal{T}]$ indicates any functions from \mathcal{S} to \mathcal{T} .

Lemma 1 ([50]). Let $n, m, q, p \in \mathbb{N}^+$, \mathcal{B}_{χ} be a U -bounded error distribution s.t. $m > (n + 1) \log q$, $q \geq p \cdot \sqrt{m} \cdot U \cdot n^{\omega(1)}$. Under the LWE difficulty assumption, F^H has pseudorandomness in the QROM.

In our scheme, we need F to have the following two properties.

1. **Uniqueness** For a string of arbitrary length $x \leftarrow \chi$, the following probabilities conditions are satisfied:

$$\Pr[\exists \mathbf{s}_1, \mathbf{s}_2 \in \mathcal{S}, \mathbf{s}_1 \neq \mathbf{s}_2 \wedge F_{\mathbf{s}_1}(x) = F_{\mathbf{s}_2}(x)] \leq \text{negl}(\lambda).$$

2. **Intersection – Free Range** For any two distinct vectors $\mathbf{y}_1, \mathbf{y}_2 \in Y$ and any polynomial $N(\cdot)$, the following probabilities conditions are satisfied:

$$\Pr[\exists k_1, k_2, h_1, h_2 \leq N(\lambda), k_1 \mathbf{y}_1 + h_1 \mathbf{z}_1 = k_2 \mathbf{y}_2 + h_2 \mathbf{z}_2 : \mathbf{z}_1, \mathbf{z}_2 \leftarrow Y] \leq \text{negl}(\lambda),$$

where Y is composed of vectors with rational numbers as elements.

Lemma 2 (Uniqueness [44]). If $m \geq n \cdot (\log q + 1) / (\log p - 1)$, then F^H is a secure function with uniqueness.

Definition 13 (Sparse Relation [57]). A relation ensemble $Q = \{Q_{\mu} \subset (\{0, 1\}^{y(\mu)})^{k(\mu)} \times (\{0, 1\}^{x(\mu)})^{k(\mu)}\}$ is sparse, if for any (x_1, \dots, x_k) the following conclusion is satisfied:

$$\Pr \left[(y_1, \dots, y_k) \leftarrow (\{0, 1\}^{y(\mu)})^{k(\mu)} : (x_1, \dots, x_k, y_1, \dots, y_k) \in Q_\mu \right] \leq \text{negl}(\mu).$$

Definition 14 (Multi-input correlation intractability [50]). *Given a relation ensemble $Q = \{Q_\mu \subset (\{0, 1\}^{y(\mu)})^{k(\mu)} \times (\{0, 1\}^{x(\mu)})^{k(\mu)}\}$, we say that hash family $\mathcal{H}=(HK,H)$ satisfies correlation intractable if any efficient adversary has the following relation holds:*

$$\Pr[hk \leftarrow HK(1^\mu); (x_1, \dots, x_k) \leftarrow \mathcal{A}(hk) : (x_1, \dots, x_k, H_{hk}(x_1), \dots, H_{hk}(x_k)) \in Q_\mu] \in \text{negl}(\mu)$$

Lemma 3 ([50]). *If $F : S \times B \rightarrow T$ is a unique PRF with an intersection free range, it holds that*

$$\Pr[a \leftarrow \mathcal{X} : R_a \text{ is not sparse}] \leq \text{negl}(\lambda).$$

3.4. Lattice-Based Accumulator

The accumulator presented in [53] will be used to construct our scheme. Let λ represent the security parameter and $n = \text{poly}(\lambda)$, $q = \text{poly}(\lambda)$. Let $L = 2^\ell, k = \lceil \log q \rceil, m = nk$ where $\ell \in \mathbb{N}^+$. The specific algorithm is shown below:

- **A.Setup.** Sample a random matrix $\mathbf{B} = (\mathbf{B}_1|\mathbf{B}_2) \xleftarrow{\$} \mathbb{Z}_q^{n \times 2m}$ and returns the value of the public parameter $para = \mathbf{B}$.
- **A.Acc.** The algorithm sets $\mathbf{t}_{\ell,i} = \mathbf{c}_i$ when given $\mathcal{R} = \{\mathbf{c}_i\}_{i \in [0, L-1]} \in (\{0, 1\}^m)^L$. Next for $j \in [0, \ell - 1]$ and $i \in [0, 2^j - 1]$, it defines $\mathbf{t}_{j,i} = \text{bin}(\mathbf{B}_1 \cdot \mathbf{t}_{j+1,2i} + \mathbf{B}_2 \cdot \mathbf{t}_{j+1,2i+1})$. Lastly, the accumulated value $\mathbf{t}_{0,0}$ is returned.
- **A.Witness.** The algorithm generates $\mathbf{t}_{j,i}$ just like the accumulate algorithm when given $\mathcal{R} = \{\mathbf{c}_i\}_{i \in [0, L-1]} \in (\{0, 1\}^m)^L$ and an element $\mathbf{c} = \mathbf{c}_i^*$. Finally, $(\text{bin}(i^*), \{\mathbf{t}_{j,f(j)}\}_{j \in [1, \ell]}, \{\mathbf{t}_{j,g(j)}\}_{j \in [1, \ell]})$ is returned, where $f(j) = \lfloor i^* / (2^{\ell-j}) \rfloor$ and $g(j) = 4 \lfloor \frac{i^*}{2^{\ell-j-1}} \rfloor - \lfloor \frac{i^*}{2^{\ell-j}} \rfloor + 1$.
- **A.Verify.** Given an accumulated value \mathbf{t} , an element \mathbf{c} and a witness $(i, \{\mathbf{h}_j\}_{j \in [1, \ell]}, \{\mathbf{k}_j\}_{j \in [1, \ell]})$, the algorithm returns 1 if

$$\begin{cases} \text{bin}(\mathbf{B}_{1+i[1]} \cdot \mathbf{h}_1 + \mathbf{B}_{2-i[1]} \cdot \mathbf{k}_1) = \mathbf{t}, \\ \forall j \in [2, \ell], \text{bin}(\mathbf{B}_{1+i[j]} \cdot \mathbf{h}_j + \mathbf{B}_{2-i[j]} \cdot \mathbf{k}_j) = \mathbf{h}_{j-1}. \end{cases}$$

4. Our Certificate-Less Traceable Ring Signature Scheme

4.1. Construction

In our scheme, the ring R denotes the set of user’s identity information and user’s public key, $L = 2^\ell$ is the capacity of ring R , and a label Γ is the contains both ring R and issue I . \mathcal{R} denotes the set of leaf node that participates in the accumulator operation.

- **Setup(1^λ)**
 1. Choose lattice parameter $n = \mathcal{O}(\lambda)$, Gaussian parameter σ_1, σ_2 , a U-bound distribution $B_{\mathcal{X}}$, integer $q \geq 2$ and prime p satisfies $q \geq p \cdot U \cdot n^{w(1)}$, $\hat{N} = w(\log \lambda)$, $\bar{m} > 1, k = \lceil \log q \rceil, n := \bar{m} + \hat{m}k \geq \mathcal{O}(\hat{m} \log q), m > (n + 1) \cdot (\log q), m \geq n(\log q + 1) / (\log p - 1), m' = m \cdot \lceil \log p \rceil, m'' = n \cdot \lceil \log q \rceil$.
 2. Set $params = (\hat{m}, n, m, m', m'', p, q, t)$, choose hash functions $H_1 : \{0, 1\}^* \rightarrow \mathbb{Z}_q^{m \times 2n}, H_2 : \{0, 1\}^* \rightarrow \mathbb{Z}_p^m$ and $H_3 : \{0, 1\}^* \rightarrow \mathbb{Z}_q^{\hat{m}}$.
 3. Choose random matrices $\mathbf{A}_0 \leftarrow \mathbb{Z}_q^{m \times n}, \mathbf{D}_1, \mathbf{D}_2 \leftarrow \mathbb{Z}_q^{n \times m'}, \mathbf{B} = [\mathbf{B}_1|\mathbf{B}_2] \leftarrow \mathbb{Z}_q^{n \times m''}$.
 4. Randomly select a matrix $\mathbf{A}' \in \mathbb{Z}_q^{\hat{m} \times \bar{m}}$, then running *Gentrap* to obtain $(\mathbf{A}, \mathbf{T}_A)$, and set $mpk = \mathbf{A} \in \mathbb{Z}_q^{\hat{m} \times n}$ and $m sk = \mathbf{T}_A \sim D_{\sigma_1}^n$.
Output $pp = (params, H_1, H_2, H_3, \mathbf{A}, \mathbf{A}_0, \mathbf{B}, \mathbf{D}_1, \mathbf{D}_2)$.
- **ExtractPartialPrivateKey(pp, msk, id_i)**
For an arbitrary identity $id_i \in \{0, 1\}^*$, define the associated vector \mathbf{a}_{id_i} as

$$\mathbf{a}_{id_i} = H_3(id_i) \in \mathbb{Z}_q^{\hat{m}},$$

run $SampleD(\mathbf{A}, \mathbf{T}_A, \mathbf{a}_{id_i}, \sigma_2)$ to get $\mathbf{s}_i \in \mathbb{Z}_q^n$, where $\sigma_2 \geq \sqrt{7(s_1(\tilde{T}_A))^2 + 1}$, $\mathbf{A} \cdot \mathbf{s}_i = \mathbf{a}_{id_i} \pmod q$, and \mathbf{s}_i is statistically close to $D_{\Lambda^u(\mathbf{A}), \sigma_2}$. Output \mathbf{s}_i .

- **SetSecretValue**(pp, id_i)
The signer selects a random vector $\mathbf{k} \in D_{\mathbb{Z}^n, \sigma_2}$, satisfying $\|\mathbf{k}\| \leq \sigma_2 \cdot \sqrt{n}$ and set his secret value $\mathbf{v}_i = \mathbf{k}$.
- **SetPrivateKey**($pp, \mathbf{s}_i, \mathbf{v}_i$)
On input the public parameters pp , signer's partial private key $\mathbf{s}_i = \mathbf{s}_{id_i}$ and secret value $\mathbf{v}_i = \mathbf{k}$, this algorithm sets full private key $sk_i = (\mathbf{s}_i, \mathbf{v}_i)$.
- **SetPublicKey**(pp, sk_i)
On input the public parameters pp and user's full private key sk_i , the user computes $\mathbf{y}_i = \lfloor \mathbf{A}_0 \cdot \mathbf{v}_i \rfloor_p \in \mathbb{Z}_p^m$ and returns his public key \mathbf{y}_i .
- **Sign**(pp, sk_π, I, R, M)
Prase $pp = (params, H_1, H_2, H_3, \mathbf{A}, \mathbf{A}_0, \mathbf{B}, \mathbf{D}_1, \mathbf{D}_2)$, $\Gamma = \{R = (r_i = (id_i, \mathbf{y}_i))_{[L]}, I\}$, $sk_\pi = (\mathbf{s}_\pi, \mathbf{v}_\pi)$.

1. Compute $\mathbf{A}_\Gamma = (\mathbf{A}_{\Gamma_1} | \mathbf{A}_{\Gamma_2}) = H_1(\Gamma) \in \mathbb{Z}_q^{m \times 2n}$.
2. Compute $\mathbf{b}_\pi = \lfloor \mathbf{A}_{\Gamma_1} \cdot \mathbf{s}_\pi + \mathbf{A}_{\Gamma_2} \cdot \mathbf{v}_\pi \rfloor_p$, and $\mathbf{b}_0 = H_2(\Gamma, M) \in \mathbb{Z}_p^m$.
3. Compute $\alpha = \frac{\mathbf{b}_\pi - \mathbf{b}_0}{\pi} \pmod p$, and $\mathbf{b}_i = \mathbf{b}_0 + \alpha \cdot i$ for all $i \neq \pi$.
4. Compute $\mathbf{c}_j = bin(\mathbf{D}_1 \cdot bin(H_2(r_j)) + \mathbf{D}_2 \cdot bin(\mathbf{b}_j))$ for $j \in [L]$, and define $\mathcal{R} = \{(\mathbf{c}_j)_{[L]}\}$.
5. Compute $\mathbf{t} = A.Acc(\mathbf{B}, \mathcal{R})$ and $w_A = A.Witness(\mathbf{B}, \mathcal{R}, \mathbf{t}, \mathbf{c}_\pi)$ where $w_A = (\mathbf{wit}, \{\mathbf{h}_j\}_{j \in [1, \ell]}, \{\mathbf{k}_j\}_{j \in [1, \ell]})$.
6. Define $\chi = (\mathbf{A}, \mathbf{A}_0, \mathbf{A}_\Gamma, \mathbf{D}_1, \mathbf{D}_2, \mathbf{B}, \mathbf{t})$ as the statement, and define $W = (\mathbf{sk}_\pi = (\mathbf{s}_\pi, \mathbf{v}_\pi), r_\pi, \mathbf{b}_\pi, \mathbf{c}_\pi, w_A)$ as the witness. Run $Proof(\chi, W)$ of NIZKAoK to obtain a proof:

$$\begin{aligned} \Pi = SPK\{(\mathbf{A}, \mathbf{A}_0, \mathbf{A}_\Gamma, \mathbf{D}_1, \mathbf{D}_2, \mathbf{B}, \mathbf{t}), (\mathbf{sk}_\pi, r_\pi, \mathbf{b}_\pi, \mathbf{c}_\pi, w_A) : \\ \mathbf{A} \cdot \mathbf{s}_\pi = \mathbf{a}_{id_\pi} \wedge \mathbf{y}_\pi = \lfloor \mathbf{A}_0 \cdot \mathbf{v}_\pi \rfloor_p \wedge \mathbf{b}_\pi = \lfloor \mathbf{A}_{\Gamma_1} \cdot \mathbf{s}_\pi + \mathbf{A}_{\Gamma_2} \cdot \mathbf{v}_\pi \rfloor_p \\ \wedge \mathbf{c}_\pi = bin(\mathbf{D}_1 \cdot bin(H_2(r_\pi)) + \mathbf{D}_2 \cdot bin(\mathbf{b}_\pi)) \\ \wedge A.Verify_{\mathbf{B}}(\mathbf{t}, \mathbf{c}_\pi, (\mathbf{wit}, \{\mathbf{h}_j\}_{j \in [1, \ell]}, \{\mathbf{k}_j\}_{j \in [1, \ell]}))\}[M]. \end{aligned}$$

7. Output the signature $\delta = (\alpha, \Pi)$.
- **Verify**(pp, I, R, δ, M)
Prase $pp = (params, H_1, H_2, H_3, \mathbf{A}, \mathbf{A}_0, \mathbf{B}, \mathbf{D}_1, \mathbf{D}_2)$, $\Gamma = \{R = (r_i = (id_i, \mathbf{y}_i))_{[L]}, I\}$, $\delta = (\alpha, \Pi)$.
 1. Let $\mathbf{A}_\Gamma = (\mathbf{A}_{\Gamma_1} | \mathbf{A}_{\Gamma_2}) = H_1(\Gamma)$ and $\mathbf{b}_0 = H_2(\Gamma, M)$.
 2. For all $i \in [L]$, we calculate $\mathbf{b}_i = \mathbf{b}_0 + \alpha \cdot i$.
 3. Calculate $\mathbf{c}_j = bin(\mathbf{D}_1 \cdot bin(H_2(r_j)) + \mathbf{D}_2 \cdot bin(\mathbf{b}_j))$ for $j \in [L]$, let $\mathcal{R} = \{(\mathbf{c}_j)_{[L]}\}$.
 4. Calculate $\mathbf{t} = A.Acc(\mathbf{B}, \mathcal{R})$.
 5. Define $\chi = (\mathbf{A}, \mathbf{A}_0, \mathbf{A}_\Gamma, \mathbf{D}_1, \mathbf{D}_2, \mathbf{B}, \mathbf{t})$ as statement, and run $v \leftarrow Verify(\chi, \Pi)$ of NIZKAoK.
 6. Output 1 if $v = 1$. Otherwise, output 0.
 - **Trace**($pp, \Gamma, M, \delta, M', \delta'$)
Prase $pp = (params, H_1, H_2, H_3, \mathbf{A}, \mathbf{A}_0, \mathbf{B}, \mathbf{D}_1, \mathbf{D}_2)$, $\Gamma = \{R = (r_i = (id_i, \mathbf{y}_i))_{[L]}, I\}$, $\delta = (\alpha, \Pi)$, $\delta' = (\alpha', \Pi')$.
 1. Let $\mathbf{b}_0 = H_2(\Gamma, M)$, then calculate $\mathbf{b}_i = \mathbf{b}_0 + \alpha \cdot i$ for all $i \in [L]$.
 2. Let $\mathbf{b}'_0 = H_2(\Gamma, M')$, then calculate $\mathbf{b}'_i = \mathbf{b}'_0 + \alpha' \cdot i$ for all $i \in [L]$.
 3. If for all $i \in [L]$ there is $b_i = b'_i$, return linked.
 4. If only one index $i \in [L]$ satisfies $b_i = b'_i$, return id_i .
 5. otherwise, return accept.

4.2. Correctness

- **Completeness.** In our scheme, α is generated by subtracting two integer vectors and then dividing by π , where $\pi \in [1, L]$ is an integer since p is a large prime number

and is much larger than L , so π is relatively prime to the prime p , then α is always an integer vector. The verifier can always restore the sequence t_i based on α , label Γ , and message M . Because of the *completeness* of the NIZKAoK protocol, for honestly generated signatures, the verifier always outputs $1 \leftarrow \text{CLTRS.Verify}(pp, \Gamma, \delta, M)$.

- **Public Traceability.** According to the definition of public traceability, there are three cases.
 1. When $M = M'$ and $\pi = \pi'$, it means that the signer signs the same message twice, so we can get $\mathbf{b}_0 = \mathbf{b}'_0$ and $\mathbf{b}_\pi = \mathbf{b}'_{\pi'}$, then $\alpha = \frac{\mathbf{b}_\pi - \mathbf{b}_0}{\pi} = \frac{\mathbf{b}_{\pi'} - \mathbf{b}'_0}{\pi'} = \alpha'$ can be easily computed. According to the equation $\mathbf{b}_i = \mathbf{b}_0 + \alpha \cdot i$, we know that $\mathbf{b}_i = \mathbf{b}'_i$ for all $i \in [1, L]$. Thereby, **CLTRS.Trace** algorithm will output “linked”.
 2. When $M \neq M'$ and $\pi = \pi'$, it means that the signer signs different messages. In this case we can get $\mathbf{b}_0 \neq \mathbf{b}'_0$ but $\mathbf{b}_\pi = \mathbf{b}'_{\pi'}$. According to the equation $\mathbf{b}_i = \mathbf{b}_0 + \alpha \cdot i$, we observe that $\mathbf{b}_i = \mathbf{b}'_i$ when i is the position of the signer in the ring. So **CLTRS.Trace** algorithm will output pk_i .
 3. When $\pi \neq \pi'$, it means that different signers sign messages. As shown in [50], the **CLTRS.Trace** algorithm will output accept with overwhelming probability.

5. The Underlying Zero-Knowledge Argument System

In this section, we will introduce efficient zero-knowledge protocols presented in [53], which will be used to construct our scheme. Similar to the Stern protocol, the efficient zero-knowledge protocol can represent most lattice-based relations. Different from the Stern protocol, the efficient zero-knowledge protocol adds an additional set \mathcal{M} to express the linear equation, which is used to express the quadratic constraint of the witness; and it has the same standard soundness as the Stern protocol, but the soundness error is only the inversion of a polynomial.

The efficient ZKAoK protocol proves relation \mathfrak{R} as follow:

$$\mathfrak{R} = \{(\mathbf{A}, \mathbf{e}, \mathcal{M}), (\mathbf{s}) \in (\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m \times ([1, n]^3)^l) \times (\mathbb{Z}_q^n) : \mathbf{A} \cdot \mathbf{s} = \mathbf{e} \wedge \forall (m, l, r) \in \mathcal{M}, \mathbf{s}[m] = \mathbf{s}[l] \cdot \mathbf{s}[r]\}$$

where the set \mathcal{M} is used to express the quadratic constraint of the witness x .

The specific construction corresponding to the relation \mathfrak{R} is given in the [53] and interested readers can go to learn more.

We will use the above efficient ZKAoK to build our CLTRS scheme. In our protocol, the prover \mathcal{P} sends the public statement $X = (\mathbf{A}, \mathbf{A}_0, \mathbf{A}_\Gamma, \mathbf{D}_1, \mathbf{D}_2, \mathbf{B}, \mathbf{t})$ and the proof Π to the verifier \mathcal{V} , and \mathcal{V} believes that secret witness $W = (\mathbf{sk}_\pi, r_\pi, \mathbf{b}_\pi, \mathbf{c}_\pi, \mathbf{w}_A = (\mathbf{wit}, \{\mathbf{h}_j\}_{j \in [1, l]}, \{\mathbf{k}_j\}_{j \in [1, l]}))$ possessed by \mathcal{P} satisfies the following relationship after verifying Π .

$$\begin{aligned} \mathfrak{R}_{\text{CLTRS}} = & \{(\mathbf{A}, \mathbf{A}_0, \mathbf{A}_\Gamma, \mathbf{D}_1, \mathbf{D}_2, \mathbf{B}, \mathbf{t}) \in \\ & \mathbb{Z}_q^{\hat{m} \times n} \times \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{n \times m'} \times \mathbb{Z}_q^{n \times m'} \times \mathbb{Z}_q^{n \times 2m''} \times \mathbb{Z}_q^n, \\ & (\mathbf{sk}_\pi = (\mathbf{s}_\pi, \mathbf{v}_\pi), r_\pi = (id_\pi, \mathbf{y}_\pi), \mathbf{b}_\pi, \mathbf{c}_\pi, \mathbf{w}_A = (\mathbf{wit}, \{\mathbf{h}_j\}_{j \in [1, l]}, \{\mathbf{k}_j\}_{j \in [1, l]})) \\ & \in \mathbb{Z}_q^{2n} \times (0, 1)^\ell \times \mathbb{Z}_p^m \times \mathbb{Z}_p^m \times \{0, 1\}^{m''} \times ((0, 1)^l \times \{(0, 1)^{m''}\}^l \times \{(0, 1)^{m''}\}^l) : \\ & H_3(id_\pi) = \mathbf{a}_{id_\pi} = \mathbf{A} \cdot \mathbf{s}_\pi \pmod q \tag{1} \\ & \wedge \mathbf{b}_\pi = \lfloor \mathbf{A}_\Gamma \cdot \mathbf{sk}_\pi \rfloor_p = \lfloor \mathbf{A}_{\Gamma_1} \cdot \mathbf{s}_\pi + \mathbf{A}_{\Gamma_2} \cdot \mathbf{v}_\pi \rfloor_p \tag{2} \\ & \wedge \mathbf{y}_\pi = \lfloor \mathbf{A}_0 \cdot \mathbf{v}_\pi \rfloor_p \tag{3} \\ & \wedge \mathbf{c}_\pi = \text{bin}(\mathbf{D}_1 \cdot \text{bin}(H_2(r_\pi)) + \mathbf{D}_2 \cdot \text{bin}(\mathbf{b}_\pi)) \tag{4} \\ & \wedge A.\text{Verify}_B(\mathbf{t}, \mathbf{c}_\pi, (\mathbf{wit}, \{\mathbf{h}_j\}_{j \in [1, l]}, \{\mathbf{k}_j\}_{j \in [1, l]})) \tag{5} \end{aligned}$$

Next, we show how the five equations in relation $\mathfrak{R}_{\text{CLTRS}}$ can be reduced to some instance in relation \mathfrak{R} . Among them, we are able to directly apply the transformation of

Section 4.4 in [53] to reduce the Equation (5) to the relation \mathfrak{R} , so the following focuses on the reduction of the other four equations.

5.1. ZKAoK of Linear Equation with Short Solution

For the first equation $\mathbf{a}_{id_\pi} = \mathbf{A} \cdot \mathbf{s}_\pi \pmod q$, we need to hide the \mathbf{s}_π and \mathbf{a}_{id_π} , so the Equation (1) can be transformed to the following form:

$$\mathbf{A} \cdot \mathbf{s}_\pi - \mathbf{a}_{id_\pi} = \mathbf{0} \pmod q. \tag{6}$$

By observing the Equation (6) we let $\overline{\mathbf{A}}_1 = (\mathbf{A} | -\mathbf{I}_{\hat{m}})$, $\overline{\mathbf{x}}_1 = (\mathbf{s}_\pi^T \ \mathbf{a}_{id_\pi}^T)^T$ and $\overline{\mathbf{y}}_1 = \mathbf{0}$. Therefore, to prove that Equation (1) holds, it is equivalent convert to prove that following equation holds:

$$\overline{\mathbf{A}}_1 \cdot \overline{\mathbf{x}}_1 = \overline{\mathbf{y}}_1 \pmod q. \tag{7}$$

For the fourth equation $\mathbf{c}_\pi = \text{bin}(\mathbf{D}_1 \cdot \text{bin}(H_2(r_{id_\pi})) + \mathbf{D}_2 \cdot \text{bin}(\mathbf{b}_\pi))$, we need to hide the \mathbf{c}_π , r_{id_π} and \mathbf{b}_π . Let $\mathbf{d}_1 = \text{bin}(H_2(r_{id_\pi}))$ and $\mathbf{d}_2 = \text{bin}(\mathbf{b}_\pi)$. The Equation (4) can be transformed to the following form:

$$\mathbf{D}_1 \cdot \mathbf{d}_1 + \mathbf{D}_2 \cdot \mathbf{d}_2 - \mathbf{H}_n \cdot \mathbf{c}_\pi = \mathbf{0} \pmod q. \tag{8}$$

By observing the Equation (8) we set the new witness $\overline{\mathbf{x}}_2 = (\mathbf{d}_1^T \ \mathbf{d}_2^T \ \mathbf{c}_\pi^T)^T$, set $\overline{\mathbf{A}}_2 = (\mathbf{D}_1 | \mathbf{D}_2 | -\mathbf{H}_n)$ and $\overline{\mathbf{y}}_2 = \mathbf{0}$ where $\mathbf{H}_n = \mathbf{I}_n \otimes (1 \ 2 \ \dots \ 2^{k_q-1})$, $k_q = \lceil \log q \rceil$. To prove that Equation (8) holds, it is equivalent to prove the following equation holds:

$$\overline{\mathbf{A}}_2 \cdot \overline{\mathbf{x}}_2 = \overline{\mathbf{y}}_2 \pmod q. \tag{9}$$

Then we need to do a binary decomposition of the witness $\overline{\mathbf{x}}_1$. We let $\overline{\mathbf{x}}_b = \text{bin}(\overline{\mathbf{x}}_1)$ and $\overline{\mathbf{A}}_b = \overline{\mathbf{A}}_1 \cdot \mathbf{H}_{n+\hat{m}}$, where bin is a binary decomposition function and $\mathbf{H}_{n+\hat{m}} = \mathbf{I}_{n+\hat{m}} \otimes (1 \ 2 \ \dots \ 2^{k_q-1})$. Then we combine Equation (7) and Equation (9) as below:

$$\mathbf{A}_{com} = \begin{pmatrix} \overline{\mathbf{A}}_b & \mathbf{0} \\ \mathbf{0} & \overline{\mathbf{A}}_2 \end{pmatrix}, \mathbf{x}_{com} = (\overline{\mathbf{x}}_b^T \ \overline{\mathbf{x}}_2^T)^T, \mathbf{y}_{com} = (\overline{\mathbf{y}}_1^T \ \overline{\mathbf{y}}_2^T)^T.$$

At last we set $\mathcal{M} = \{(i, i, i)\}_{i \in [1, (2\hat{m}+2n)k_q+2mk_p]}, k_p = \lceil \log p \rceil$.

In doing so, we reduce the relation $\mathfrak{R}_{(1)} = \{(\mathbf{A}), (\mathbf{s}_\pi, \mathbf{a}_{id_\pi}) \in (\mathbb{Z}_q^{\hat{m} \times n} \times (\mathbb{Z}_q^n \times \mathbb{Z}_q^{\hat{m}}))\}$ and $\mathfrak{R}_{(3)} = \{(\mathbf{D}_1, \mathbf{D}_2), (\text{bin}(H_2(r_{id_\pi})), \text{bin}(\mathbf{b}_\pi), \mathbf{c}_\pi) \in (\mathbb{Z}_q^{n \times m'} \times \mathbb{Z}_q^{n \times m'} \times (\{0, 1\}^{mk_p} \times \{0, 1\}^{mk_p} \times \{0, 1\}^{nk_q}))\}$ to \mathfrak{R} , both the witness and \mathcal{M} are size of $(\hat{m} + 2n)k_q + 2mk_p$.

5.2. ZKAoK of PRF Preimage

For the second Equation (2) $\mathbf{b}_\pi = \lfloor \mathbf{A}_\Gamma \cdot \mathbf{s} \mathbf{k}_\pi \rfloor_p$, we also need to hide the $\mathbf{s} \mathbf{k}_\pi$ and \mathbf{b}_π . Specifically, let m, n be positive integers, p be a prime number and q is an integer with $q > p \geq 2$. We can express Equation (2) as the following relation:

$$\mathfrak{R}_{(2)} = \{(\mathbf{A}_\Gamma), (\mathbf{s} \mathbf{k}_\pi, \mathbf{b}_\pi) \in (\mathbb{Z}_q^{m \times n}) \times (\mathbb{Z}_q^{2n} \times \mathbb{Z}_p^m) : \mathbf{b}_\pi = \lfloor \mathbf{A}_\Gamma \cdot \mathbf{s} \mathbf{k}_\pi \rfloor_p\}$$

We demonstrate the above relation $\mathfrak{R}_{(2)}$ by reducing it to an instance of the relation \mathfrak{R} through suitable transformations. We convert the $\mathbf{b}_\pi = \lfloor \mathbf{A}_\Gamma \cdot \mathbf{s} \mathbf{k}_\pi \rfloor_p \pmod q$ to the following form:

$$\begin{cases} \mathbf{A}_\Gamma \cdot \mathbf{s} \mathbf{k}_\pi = \mathbf{b}'_\pi \pmod q, \\ \lfloor \frac{p}{q} \cdot \mathbf{b}'_\pi \rfloor = \mathbf{b}_\pi \pmod p. \end{cases}$$

As shown in [44,58], the second equation $\lfloor \frac{p}{q} \cdot \mathbf{b}'_\pi \rfloor = \mathbf{b}_\pi \pmod p$ holds iff each element of the vector $\frac{q}{p} \mathbf{b}_\pi - \mathbf{b}'_\pi$ is in $[0, \frac{q}{p})$, and thus above equation can be transformed in the following:

$$\begin{cases} \mathbf{A}_\Gamma \cdot \mathbf{sk}_\pi = \mathbf{b}'_\pi \pmod q, \\ \mathbf{e}' = \frac{q}{p} \mathbf{b}_\pi - \mathbf{b}'_\pi \pmod q. \end{cases}$$

Then we can transform it into a linear equation with short solution. Next, we will describe how the process of reduction is carried out. In the following, we will omit all mod q operations for simplicity.

First, we will draw on the separation technique mentioned in [59] to convert \mathbf{e}' to a binary vector \mathbf{e}'_b . Setting $\zeta = \frac{q}{p} - 1, k = \lceil \log \zeta \rceil$. The size of \mathbf{e}'_b is $k \cdot m$.

Let $\mathbf{g} = (\lfloor \frac{\zeta+1}{2} \rfloor \parallel \dots \parallel \lfloor \frac{\zeta+2^{i-1}}{2^i} \rfloor \parallel \dots \parallel \lfloor \frac{\zeta+2^{k-1}}{2^k} \rfloor \parallel)$, $\mathbf{G} = \mathbf{I}_m \otimes \mathbf{g}$, which satisfies $\mathbf{G} \cdot \mathbf{e}'_b = \mathbf{e}'$.

Then, let $\mathbf{sk}_\pi^* = \text{bin}(\mathbf{sk}_\pi), \mathbf{b}'_{\pi^*} = \text{bin}(\mathbf{b}'_\pi), \mathbf{b}^*_{\pi} = \text{bin}(\frac{q}{p} \mathbf{b}_\pi), \mathbf{H}_m = \mathbf{I}_m \otimes (1 \ 2 \ 4 \ \dots \ 2^{k_q-1}), \mathbf{H}_n = \mathbf{I}_n \otimes (1 \ 2 \ 4 \ \dots \ 2^{k_q-1})$, where $k_q = \lceil \log q \rceil$. We set

$$\mathbf{A}'_1 = \begin{pmatrix} \mathbf{A}_\Gamma \cdot \mathbf{H}_n & -\mathbf{H}_m & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_m & \mathbf{G} & -\mathbf{H}_m \end{pmatrix},$$

$$\mathbf{x}'_1 = (\mathbf{sk}_\pi^{*T} \ \mathbf{b}'_{\pi^*T} \ \mathbf{e}'_bT \ \mathbf{b}^*_{\pi T})^T, \mathbf{u}'_1 = (\mathbf{0} \ \mathbf{0})^T.$$

For the third Equation (3) $\mathbf{y}_\pi = \lfloor \mathbf{A}_0 \cdot \mathbf{v}_\pi \rfloor_p$, its structural form is similar to that of Equation (2). Referring to the transformation of Equation (2) above, we can obtain

$$\begin{cases} \mathbf{A}_0 \cdot \mathbf{v}_\pi = \mathbf{y}'_\pi \pmod q, \\ \mathbf{e}'' = \frac{q}{p} \mathbf{y}_\pi - \mathbf{y}'_\pi \pmod q. \end{cases}$$

Then, let $\mathbf{v}^*_\pi = \text{bin}(\mathbf{v}_\pi), \mathbf{y}'_{\pi^*} = \text{bin}(\mathbf{y}'_\pi), \mathbf{y}^*_{\pi} = \text{bin}(\frac{q}{p} \mathbf{y}_\pi), \mathbf{H}_m = \mathbf{I}_m \otimes (1 \ 2 \ 4 \ \dots \ 2^{k_q-1}), \mathbf{H}_n = \mathbf{I}_n \otimes (1 \ 2 \ 4 \ \dots \ 2^{k_q-1})$, where $k_q = \lceil \log q \rceil$. We set

$$\mathbf{A}'_2 = \begin{pmatrix} \mathbf{A}_0 \cdot \mathbf{H}_n & -\mathbf{H}_m & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_m & \mathbf{G} & -\mathbf{H}_m \end{pmatrix},$$

$$\mathbf{x}'_2 = (\mathbf{v}^*_{\pi T} \ \mathbf{y}'_{\pi^*T} \ \mathbf{e}''_bT \ \mathbf{y}^*_{\pi T})^T, \mathbf{u}'_2 = (\mathbf{0} \ \mathbf{0})^T.$$

Then we combine Equation (2) and Equation (3) as below:

$$\mathbf{A}'_{com} = \begin{pmatrix} \mathbf{A}'_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{A}'_2 \end{pmatrix}, \mathbf{x}'_{com} = (\mathbf{x}'_1T \ \mathbf{x}'_2T)^T, \mathbf{u}'_{com} = (\mathbf{u}'_1T \ \mathbf{u}'_2T)^T.$$

At last we set $\mathcal{M} = \{(i, i, i)\}_{i \in [1, (3n+2m)k_q + 2mk + 2mk_p]}$.

By doing so, we reduce the relation $\mathfrak{R}_{(2)} = \{(\mathbf{A}_\Gamma), (\mathbf{sk}_\pi, \mathbf{b}_\pi) \in (\mathbb{Z}_q^{m \times 2n}) \times (\mathbb{Z}_q^{2n}) \times (\mathbb{Z}_p^m) : \mathbf{b}_\pi = \lfloor \mathbf{A}_\Gamma \cdot \mathbf{sk}_\pi \rfloor_p\}$ and $\mathfrak{R}_{(3)} = \{(\mathbf{A}_0), (\mathbf{v}_\pi, \mathbf{y}_\pi) \in (\mathbb{Z}_q^{m \times n}) \times (\mathbb{Z}_q^n) \times (\mathbb{Z}_p^m) : \mathbf{y}_\pi = \lfloor \mathbf{A}_0 \cdot \mathbf{v}_\pi \rfloor_p\}$ to \mathfrak{R} , both the witness and \mathcal{M} are size of $(2n + 2m)k_q + 2mk + 2mk_p$.

6. Security Analysis

First of all, the output result is accepted when the two tuples (I, R, M, δ) and (I, R, M', δ') are used as input to the **CLTRS.Trace** algorithm, which indicates that there is no index number s.t. $\mathbf{b}_i = \mathbf{b}'_i$, where $(\mathbf{b}_i)_{[L]}$ and $(\mathbf{b}'_i)_{[L]}$ are computed from (I, R, M, δ) and (I, R, M', δ') respectively. Along this line of thinking, if we can show that the probability that other cases lead to an output result that is not accepted is negligible or impossible, then the proof of tag-linkability is complete. Other cases include the following two possibilities: (1) when there exist two or more indexes i_k satisfying $\mathbf{b}_{i_k} = \mathbf{b}'_{i_k}$ for $k = 0, 1$, then it is shown that $\mathbf{b}_j = \mathbf{b}'_j$ for all $j \in [L]$, because two different lines have two points of intersection, it follows that the two lines must coincide, as a result, **CLTRS.Trace** algorithm will output *linked*. (2) when only one index exists that satisfies $\mathbf{b}_i = \mathbf{b}'_i$, this means that the same signer has signed different messages, so **CLTRS.Trace** algorithm will output pk_i , which indicates that the user's identity will be exposed. Therefore, we have the following conclusions.

Theorem 1 (Tag-linkability). *Under the ROM, if the SIS assumption holds, the underlying NIZKAoK is proof of knowledge and F has uniqueness, then the CLTRS scheme is tag-linkable.*

Proof. According to the definition of the traceable algorithm, we can easily know that when the return result obtained from two tuples (I, R, M, δ) and (I, R, M', δ') as input is accepted, it means that there is no exist $\mathbf{b}_j = \mathbf{b}'_j$ where $j \in [L]$. And for $L + 1$ valid signatures, the result obtained for any two of them as input to the tracing algorithm is accepted, it indicates that there exist $L + 1$ different $\mathbf{b}_{j,\pi}$, where $j \in [L + 1]$. Since the generation of $\mathbf{b}_{j,\pi}$ corresponds uniquely to sk_j and sk_j corresponds uniquely to pk_j , so $\mathbf{b}_{j,\pi}$ corresponds uniquely to the public key pk_j , which indicates the existence of $L + 1$ different public keys, while there are at most L different public keys in the ring, hence the contradiction, so the adversary cannot break the tag-linkability. If \mathcal{A} can break the tag-linkability, then \mathcal{C} can construct the following algorithm to break the SIS hard problem. The simulation works as follows:

- **SIS instance.** The challenger \mathcal{C} receives an SIS instance $\mathbf{F} \leftarrow \mathbb{Z}_q^{n \times n}$, \mathcal{C} needs to look for a non-zero vector $\mathbf{s} \in \mathbb{Z}_q^n$ satisfying $\mathbf{F} \cdot \mathbf{s} = \mathbf{0} \pmod q$, which $\|\mathbf{s}\| \leq \beta$ and $\beta = \sigma_2 \cdot \sqrt{n}$.
1. **Setup:** Given the system parameter 1^λ , instead of running the real zero-knowledge, \mathcal{C} invokes simulator \mathcal{S}_{sim} to generate the public parameters, the remaining parameters are generated unchanged except that mpk is replaced with \mathbf{F} , then sends pp to \mathcal{A} and msk is kept itself.
 2. **Query.** \mathcal{C} initializes initially empty lists $list_{H_1}, list_{H_2}, list_{H_3}, list_{CU}, list_{SQ}$ and keeps the consistency of answers to adversaries by maintaining these tables,
 - (a) **H₁ Query.** When \mathcal{A} submits a label Γ_i to this oracle, \mathcal{C} first checks if Γ_i in the $list_{H_1}$, if it exists, returns the corresponding \mathbf{A}_{Γ_i} . Otherwise, \mathcal{C} picks a random matrix $\mathbf{A}_{\Gamma_i} \leftarrow \mathbb{Z}_q^{m \times 2n}$, and returns it to \mathcal{A} .
 - (b) **H₂ Query.** When \mathcal{A} submits a tuple (Γ_i, M_i) , \mathcal{C} first checks if the tuple in the $list_{H_2}$, if it exists, returns the corresponding $\mathbf{b}_{i,0}$. Otherwise, \mathcal{C} picks a random vector $\mathbf{b}_{i,0} \leftarrow \mathbb{Z}_p^m$, and returns it to \mathcal{A} .
 - (c) **H₃ Query.** When \mathcal{A} submits an id_i to this oracle, \mathcal{C} first checks if id_i in the $list_{H_3}$, if it exists, returns the corresponding \mathbf{a}_{id_i} . Otherwise, \mathcal{C} randomly selects a $\mathbf{s}_i \leftarrow \mathbb{Z}_p^n$ where \mathbf{s}_i is statistically close to $D_{\Lambda^u(\mathbf{A}), \sigma_2}$, computes $\mathbf{a}_{id_i} = \mathbf{F} \cdot \mathbf{s}_i \pmod q$, and returns it to \mathcal{A} .
 - (d) **CreateUserQuery.** When \mathcal{A} submits an id_i , \mathcal{C} first checks if id_i in the $list_{CU}$, if it exists, returns the corresponding key pairs (pk_i, sk_i) . Otherwise, \mathcal{C} calls the **PartialPrivateKeyQuery** with id_i to obtain \mathbf{s}_i , and then runs *SetSecretValue*, *SetPrivateKey*, *SetPublicKey* algorithm to generate \mathbf{v}_i, sk_i and \mathbf{y}_i , returns (pk_i, sk_i) to adversary \mathcal{A} , then adds tuple $(id_i, \mathbf{s}_i, \mathbf{v}_i, sk_i, \mathbf{y}_i)$ to $list_{CU}$.
 - (e) **PartialPrivateKeyQuery.** When \mathcal{A} submits an id_i , \mathcal{C} first checks if \mathbf{a}_{id_i} in the $list_{H_3}$, if it exists, returns the corresponding partial private key \mathbf{s}_i . Otherwise, \mathcal{C} runs H_3 to generate \mathbf{s}_i and returns it to \mathcal{A} .
 - (f) **ReplacePublicKeyQuery.** When \mathcal{A} submits an id_i and a \mathbf{y}'_i to this oracle, \mathcal{C} substitutes \mathbf{y}_i for \mathbf{y}'_i .
 - (g) **SignQuery.** When the adversary \mathcal{A} submits the user id_i , issue I , ring R and message M , \mathcal{C} return $\delta \leftarrow CLTRS.Sign(pp, sk_i, I, R, M)$ to \mathcal{A} .
 3. **Forgery.** We assume that the signature δ_i contains the sequence $(\mathbf{b}_{i,1}, \dots, \mathbf{b}_{i,L+1})$. This assumption is well-founded because we can simply recover the sequence from the tuple (I, R, M_i, δ_i) . The probability that \mathcal{A} forges $L + 1$ valid signatures $\delta_i = (\alpha_i, \Pi_i)$ on $L + 1$ tuples (I, R, M_i) while the trace algorithm takes any two signatures as input and obtains an output that is both accept is non-negligible, where $i \in [L + 1]$. That means
 - (a) $CLTRS.Verify(pp, I, R, \delta_i, M_i) = 1, \forall i \in [L + 1]$,
 - (b) $CLTRS.Trace(pp, I, R, M_h, \delta_h, M_k, \delta_k) = accept, \forall h, k \in [L + 1], s.t., h \neq k$.

4. **Analysis.** During the initialization phase, although we do not run the real ZKAoK protocol but invoke a simulator \mathcal{S}_{sim} for the ZKAoK protocol, the adversary \mathcal{A} cannot detect our substitution due to the zero-knowledge property. According to the proof of knowledge property, the extractor \mathcal{E} capable of extracting the witness from each $(M_i, \delta_i = (\alpha_i, \Pi_i))$, where witness $w = (\pi_i, sk_{\pi_i})$, since condition (ii) holds and the unique of the function F , so there is no $sk_{\pi_k} = sk_{\pi_h}$ where $k, h \in [L + 1]$. For each secret key $sk_{\pi_i} = (\mathbf{s}_{\pi_i}, \mathbf{v}_{\pi_i})$ we have the following relationship holding:

$$H_3(id_i) = \mathbf{F} \cdot \mathbf{s}_{\pi_i} \wedge \mathbf{y}_i = \lfloor \mathbf{A}_0 \cdot \mathbf{v}_{\pi_i} \rfloor_p.$$

There are only L public keys in R . Then it means that there exists one pk matching two different sk. For the equation $H_3(id_i) = \mathbf{F} \cdot \mathbf{s}_{\pi_i}$, if there exists another \mathbf{s}^* satisfying $H_3(id_i) = \mathbf{F} \cdot \mathbf{s}^*$, then it can get $\mathbf{F} \cdot (\mathbf{s}_{\pi_i} - \mathbf{s}^*) = \mathbf{0} \pmod q$, and $(\mathbf{s}_{\pi_i} - \mathbf{s}^*)$ is a resolution to the SIS problem, therefore, the probability of this happening is negligible. For the equation $\mathbf{y}_i = \lfloor \mathbf{A}_0 \cdot \mathbf{v}_{\pi_i} \rfloor_p$, if there exists another \mathbf{v}^* satisfying $\mathbf{y}_i = \lfloor \mathbf{A}_0 \cdot \mathbf{v}^* \rfloor_p$, then it breaks Uniqueness of F , the probability of this happening is also negligible. In summary, the probability of the adversary \mathcal{A} being able to forge a successful forgery is negligible. Therefore, our scheme is satisfies tag-linkability.

□

Theorem 2 (Type I Anonymity). *The CLTRS scheme is type I anonymous security under the ROM, if the DLWE assumption holds and the underlying NIZKAoK is zero-knowledge.*

Proof. In the proof of anonymity, we are proving it by means of game hopping, which achieves indistinguishability between games by changing some negligible setting between every two games in order to be undetectable to the adversary. That is, the probability that arbitrary PPT adversary \mathcal{A}_I can distinguish the difference between every two games is negligible. We define the advantage of \mathcal{A}_I in Game i by $Adv_{\mathcal{A}_I, Game\ i}^{Anon_1}(1^\lambda)$.

Game 0: It is type I anonymous game which $b = 0$. \mathcal{C} operates algorithm *CLTRS.Setup* in the Setup phase, saves *msk* secretly, \mathcal{A}_I obtains *pp* from \mathcal{C} .

Game 1: The game and Game 0 are identical, besides when \mathcal{A}_I submits a *PPKQ*(id_i) to H_3 , the challenger chooses randomly $\mathbf{a}_{id_i} \leftarrow \mathbb{Z}_p^m$ and set $H_3(id_i) = \mathbf{a}_{id_i}$, then \mathcal{C} keeps a $list_{H_3}(id_i, \mathbf{a}_{id_i})$ to respond consistently. In ROM, \mathcal{A}_I is unable to discern the difference between Game 1 and Game 0. Therefore, we obtain

$$Adv_{\mathcal{A}_I, Game\ 1}^{Anon_1}(1^\lambda) \approx Adv_{\mathcal{A}_I, Game\ 0}^{Anon_1}(1^\lambda).$$

Game 2: The game and Game 1 are identical, besides \mathcal{C} uses simulator \mathcal{S} replace the real NIZK proof system. \mathcal{C} uses \mathcal{S} to obtain *pp* instead of using the real NIZK. By doing so, the challenger is able to generate valid proofs properly without the use of sk_b . When \mathcal{A}_I submits the challenge message M^* during the challenge phase, \mathcal{C} invokes \mathcal{S} to generate the valid simulation proof Π^* instead of running the real NIZK to generate Π_{real} . Based on the zero-knowledge property, we have

$$Adv_{\mathcal{A}_I, Game\ 2}^{Anon_1}(1^\lambda) \approx Adv_{\mathcal{A}_I, Game\ 1}^{Anon_1}(1^\lambda).$$

Game 3: The game and Game 2 are identical, besides when it is necessary to run the F_{sk_b} , \mathcal{C} first finds whether a corresponding tuple $(\Gamma_i, M_i, \mathbf{b}_{i,\pi})$ exists for the $list_{F_{sk_b}}$ and, if so, returns the corresponding $\mathbf{b}_{i,\pi}$. Otherwise, \mathcal{C} takes a stochastic vector $\mathbf{b}_{i,r}$, sends it to \mathcal{A}_I , and adds the vector $\mathbf{b}_{i,r}$ to the $list_F$. So \mathcal{C} can simulate F_{sk_b} by keeping $list_F$. For any vector $\mathbf{b} = \lfloor \mathbf{A}_{\Gamma_1} \cdot \mathbf{s} + \mathbf{A}_{\Gamma_2} \cdot \mathbf{v} \rfloor_p$, We can convert it to $\frac{q}{p} \cdot \mathbf{b} - \mathbf{A}_{\Gamma_2} \cdot \mathbf{v} = \mathbf{A}_{\Gamma_1} \cdot \mathbf{s} + \mathbf{e} \pmod q$ where $\mathbf{e} \in [0, \frac{q}{p}]$, if adversary \mathcal{A}_I can distinguish between $\frac{q}{p} \cdot \mathbf{b} - \mathbf{A}_{\Gamma_2} \cdot \mathbf{v}$ and $\frac{q}{p} \cdot \mathbf{b}_{i,r} - \mathbf{A}_{\Gamma_2} \cdot \mathbf{v}$ without knowing secret key \mathbf{s} and error \mathbf{e} , \mathcal{C} may utilize the adversary to crack DLWE problem. Due to the difficulty of the DLWE problem, the probability of \mathcal{A}_I to distinguish between \mathbf{b} and $\mathbf{b}_{i,r}$ is negligible. Thus, we have

$$(\mathbf{A}_\Gamma, \Gamma, \mathbf{v}, \mathbf{b} : \mathbf{b} \leftarrow F_{sk_b}(\Gamma)) \approx (\mathbf{A}_\Gamma, \Gamma, \mathbf{v}, \mathbf{b} : \mathbf{b} \leftarrow \mathbb{Z}_p^m).$$

And thus

$$Adv_{\mathcal{A}_I, Game\ 3}^{Anon_I}(1^\lambda) \approx Adv_{\mathcal{A}_I, Game\ 2}^{Anon_I}(1^\lambda).$$

Obviously, in Game 3, the generation of signatures α is essentially replaced by random numbers, meaning that the generation of α is already independent of sk_b . By the same token, when Game 0 is selected with b of 1, the change between each two games is the same as the change from Game 0 to Game 3 above, so when $b = 1$, Game 0 is also indistinguishable from Game 3, and since whether $b = 0$ or $b = 1$, the final Game 3 obtained is identical, it can be easily obtained that when $b = 0$ is selected and when $b = 1$ is selected, it is indistinguishable for the adversary is indistinguishable for the adversary \mathcal{A}_I . Thus we have

$$Adv_{\mathcal{A}_I, Game\ 0, b=0}^{Anon_I}(1^\lambda) \approx Adv_{\mathcal{A}_I, Game\ 0, b=1}^{Anon_I}(1^\lambda).$$

So our scheme satisfies type I anonymity.

□

Theorem 3 (Type II Anonymity). *The CLTRS scheme is type II anonymous security under the ROM, if the DLWE assumption holds and the underlying NIZKAoK is zero-knowledge.*

Proof. The idea of the proving of Theorem 3 is similar to that of Theorem 2, except that in the proving of Theorem 3 it is the second type of adversary \mathcal{A}_{II} that has to be faced. However, the goal is still to make it difficult for \mathcal{A}_{II} to identify the difference between each of the two games. That is, the probability that arbitrary PPT \mathcal{A}_{II} can distinguish the difference between every two games is negligible. We define the advantage of \mathcal{A}_{II} in Game i by $Adv_{\mathcal{A}_{II}, Game\ i}^{Anon_{II}}(\lambda)$.

Game 0: It is type II anonymous game which $b = 0$. \mathcal{C} operates algorithm $CLTRS.Setup$ in the Setup phase, then sends pp and msk to \mathcal{A}_{II} .

Game 1: The game and Game 0 are identical, besides \mathcal{C} use simulator \mathcal{S} replace the real NIZK proof system. \mathcal{C} calls simulator \mathcal{S} to produce the public parameters instead of using the real NIZK. By doing so, the challenger is able to generate valid proofs properly without the use of sk_b . When \mathcal{A}_{II} submits the challenge message M^* during the challenge phase, \mathcal{C} invokes simulator \mathcal{S} to generate a valid simulation proof Π^* instead of running the real NIZK to generate Π_{real} . Based on the zero-knowledge property, we have

$$Adv_{\mathcal{A}_{II}, Game\ 1}^{Anon_{II}}(1^\lambda) \approx Adv_{\mathcal{A}_{II}, Game\ 0}^{Anon_{II}}(1^\lambda).$$

Game 2: The game and Game 1 are identical, besides when it is necessary to run the F_{sk_b} , \mathcal{C} first finds whether a corresponding tuple $(\Gamma_i, M_i, \mathbf{b}_{i,\pi})$ exists for the $list_{F_{sk_b}}$ and, if so, returns the corresponding $\mathbf{b}_{i,\pi}$. Otherwise, \mathcal{C} takes a stochastic vector $\mathbf{b}_{i,r}$, sends it to \mathcal{A}_{II} , and adds the vector $\mathbf{b}_{i,r}$ to the $list_F$. So \mathcal{C} can simulate F_{sk_b} by keeping $list_F$. For any vector $\mathbf{b} = \lfloor \mathbf{A}_{\Gamma_1} \cdot \mathbf{s} + \mathbf{A}_{\Gamma_2} \cdot \mathbf{v} \rfloor_p$, We can convert it to $\frac{q}{p} \cdot \mathbf{b} - \mathbf{A}_{\Gamma_1} \cdot \mathbf{s} = \mathbf{A}_{\Gamma_2} \cdot \mathbf{v} + \mathbf{e} \pmod q$ where $\mathbf{e} \in [0, \frac{q}{p})$, if adversary \mathcal{A}_{II} can distinguish between $\frac{q}{p} \cdot \mathbf{b} - \mathbf{A}_{\Gamma_1} \cdot \mathbf{s}$ and $\frac{q}{p} \cdot \mathbf{b}_{i,r} - \mathbf{A}_{\Gamma_1} \cdot \mathbf{s}$ without knowing secret key \mathbf{s} and error \mathbf{e} , \mathcal{C} may utilize the adversary to crack DLWE problem. Due to the difficulty of the DLWE problem, the probability of an adversary \mathcal{A}_{II} being able to distinguish between \mathbf{b} and $\mathbf{b}_{i,r}$ is negligible. Thus, we have

$$(\mathbf{A}_\Gamma, \Gamma, \mathbf{v}, \mathbf{b} : \mathbf{b} \leftarrow F_{sk_b}(\Gamma)) \approx (\mathbf{A}_\Gamma, \Gamma, \mathbf{v}, \mathbf{b} : \mathbf{b} \leftarrow \mathbb{Z}_p^m).$$

And thus

$$Adv_{\mathcal{A}_{II}, Game 2}^{Anon_{II}}(1^\lambda) \approx Adv_{\mathcal{A}_{II}, Game 1}^{Anon_{II}}(1^\lambda).$$

The same principle as the type I anonymity, we change $b = 0$ of Game 0 to $b = 1$ and then to Game 2, the adversary still cannot identify the distinction between Game 0 and Game 2 at $b = 1$, so we can get a real anonymous game where the adversary cannot distinguish whether the value of b is selected as 0 or 1. Thus, we have

$$Adv_{\mathcal{A}_{II}, Game 0, b=0}^{Anon}(1^\lambda) \approx Adv_{\mathcal{A}_{II}, Game 0, b=1}^{Anon}(1^\lambda).$$

So our scheme satisfies type II anonymity.

□

Theorem 4 (Type I Exculpability). *If hash family H is multi-input correlation intractable, NIZKAoK is proof of knowledge, and the function F^H is unique with an intersection-free range, the CLTRS is type I exculpable under the ROM.*

Proof. Assume that there has a PPT adversary \mathcal{A}_I with access to ROM, it inquires the random oracles H_1, H_2, H_3 up to $q_{h_1}, q_{h_2}, q_{h_3}$ times and makes a maximum of q_s signing queries, if it can break the exculpability with non-negligible probability ϵ , then we may construct adversaries \mathcal{C}_I and \mathcal{C}_{II} to break uniqueness of F and multi-input correlation intractable of H with probability $\frac{\epsilon}{2q_{h_1} \cdot q_s}$ and $\frac{\epsilon}{2}$, respectively.

- **Adversary \mathcal{C}_I .** Given a challenge matrix $\tilde{\mathbf{A}}_{\Gamma_1} \in \mathbb{Z}_q^{m \times n}$ which is random, it is an instance of function uniqueness. The adversary \mathcal{C}_I need to look for a non-zero vector $\tilde{\mathbf{s}} \in \mathbb{Z}_q^n$ and a vector $\tilde{\mathbf{e}}_s \in (-\frac{q}{p}, \frac{q}{p})^m$ satisfying $\tilde{\mathbf{A}}_{\Gamma_1} \cdot \tilde{\mathbf{s}} = \tilde{\mathbf{e}}_s \pmod q$, adversary \mathcal{C}_I constructs the following game to attack the uniqueness of the function.
 - 1.1 **Setup.** Given the system parameter 1^λ , instead of running the real zero-knowledge protocol, \mathcal{C} calls simulator \mathcal{S}_{sim} to produce the public parameters, and remaining parameters are generated unchanged, then sends pp to \mathcal{A}_I and msk is kept itself.
 - 1.2 **Queries.** \mathcal{C}_I initializes initially empty lists $list_{H_1}, list_{H_2}, list_{H_3}, list_{CU}, list_{SQ}$ and keeps the consistency of answers to adversaries by maintaining these tables. We assume that adversary \mathcal{A}_I must have done the following queries before forging: \mathcal{A}_I will submit a query with label $\Gamma^* = (I^*, R^*)$ to H_1 for the i^* -th time where $i^* \in [1, q_{h_1}]$ and submit a query with tuple $(id^*, \Gamma^* = (I^*, R^*), M^*)$ to SQ for the j^* -th time where $j^* \in [1, q_s]$ and $id^* \in R^*$.
 - (a) **H₁ Query.** For the $i \in [1, q_h]$ times of asking, if $i = i^*$, \mathcal{A}_I will submit the label Γ^* , \mathcal{C}_I picks a random matrix $\mathbf{A}_{\Gamma^*} \leftarrow \mathbb{Z}_q^{m \times n}$, and then sets $\mathbf{A}_{\Gamma^*} = (\tilde{\mathbf{A}}_{\Gamma_1} | \mathbf{A}_{\Gamma^*})$, at last returns it to \mathcal{A}_I ; otherwise if $i \neq i^*$, \mathcal{A}_I will submit the label Γ_i , \mathcal{C}_I picks a random matrix $\mathbf{A}_{\Gamma_i} \leftarrow \mathbb{Z}_q^{m \times 2n}$ and outputs it to \mathcal{A}_I .
 - (b) **H₂ Query.** When \mathcal{A}_I submits a tuple (Γ_i, M_i) , \mathcal{C}_I first checks if the tuple in the $list_{H_2}$, if it exists, returns the corresponding $\mathbf{b}_{i,0}$. Otherwise, \mathcal{C}_I picks a random vector $\mathbf{b}_{i,0} \leftarrow \mathbb{Z}_p^m$, and returns it to \mathcal{A}_I .
 - (c) **H₃ Query.** When \mathcal{A}_I submits an id_i , \mathcal{C}_I first checks if id_i in the $list_{H_3}$, if it exists, returns the corresponding \mathbf{a}_{id_i} . Otherwise, \mathcal{C}_I randomly selects a vector $\mathbf{s}_i \leftarrow \mathbb{Z}_p^n$ where $\mathbf{s}_i \in D_{\Lambda^u(\mathbf{A}), \sigma_2}$ satisfying $\|\mathbf{s}_i\| \leq \sigma_2 \sqrt{n}$, computes $\mathbf{a}_{id_i} = \mathbf{A} \cdot \mathbf{s}_i$, and returns it to \mathcal{A}_I .
 - (d) **CreateUserQuery.** When \mathcal{A}_I submits an id_i , \mathcal{C}_I first checks if id_i in the $list_{CU}$, if it exists, returns the corresponding key pairs (pk_i, sk_i) . Otherwise, \mathcal{C}_I calls the **PartialPrivateKeyQuery** with id_i to obtain \mathbf{s}_i , and then runs *SetSecretValue*, *SetPrivateKey*, *SetPublicKey* algorithm to generate \mathbf{v}_i, sk_i and \mathbf{y}_i , returns (pk_i, sk_i) to adversary \mathcal{A}_I , then adds tuple $(id_i, \mathbf{s}_i, \mathbf{v}_i, sk_i, \mathbf{y}_i)$ to $list_{CU}$.

- (e) **PartialPrivateKeyQuery.** When \mathcal{A}_I submits an id_i , \mathcal{C}_I first checks if \mathbf{a}_{id_i} in the $list_{H_3}$, if it exists, returns the corresponding \mathbf{s}_i . Otherwise, \mathcal{C}_I runs H_3 to generate \mathbf{s}_i and outputs it to \mathcal{A}_I .
- (f) **ReplacePublicKeyQuery.** When \mathcal{A}_I submits an id_i and a y'_i , \mathcal{C} substitutes y_i for y'_i .
- (g) **SignQuery.** For the $j \in [1, q_s]$ times of asking, if $j = j^*$, \mathcal{A}_I will submit a tuple (id^*, Γ^*, M^*) , instead of running $F_{sk_{id^*}}$, \mathcal{C}_I computes $\mathbf{b}_{j^*, \pi^*} = \lfloor \tilde{\mathbf{A}}_{\Gamma_1} \cdot \mathbf{s}_{\pi^*} + \mathbf{A}_{\Gamma_2^*} \cdot \mathbf{v}_{\pi^*} \rfloor_p$, and invokes simulator \mathcal{S}_{sim} to generate the proof, the remaining steps remain unchanged, returns the generated signature δ^* to \mathcal{A}_I . Otherwise if $j \neq j^*$, \mathcal{A}_I will submit a tuple (id_j, Γ_j, M_j) , \mathcal{C}_I runs F_{sk_j} to compute $\mathbf{b}_{j, \pi}$, the remaining steps remain unchanged, returns the generated signature δ_j to \mathcal{A}_I .

1.3 **Forgery.** Assume \mathcal{A}_I outputs two tuples $(I^*, R^*, \bar{M}, \bar{\delta})$ and (I^*, R^*, M', δ') s.t.

- (a) $\text{CLTRS.Verify}(pp, I^*, R^*, \bar{M}, \bar{\delta}) = 1$;
- (b) $\text{CLTRS.Verify}(pp, I^*, R^*, M', \delta') = 1$;
- (c) \mathcal{A}_I has not been queried about $\text{PPKQ}(pp, id^*)$ and $\text{CUQ}(pp, id^*)$ where $id^* \in R^*$;
- (d) \mathcal{A}_I has made at most one of $\text{SQ}(pp, id^*, I^*, R^*, \bar{M})$ and $\text{SQ}(pp, id^*, I^*, R^*, M')$;
- (e) $\text{CLTRS.Trace}(pp, I^*, R^*, \bar{M}, \bar{\delta}, M', \delta') = pk^*$.

Suppose π^* is the location of pk^* in R^* .

1.4 **Analysis.** Here, we assume that one of the two signatures output by the adversary is the challenge signature of the previous query, which indicates a situation where the user honestly generates a signature and the adversary \mathcal{A}_I forges another valid signature to trap the honest user. We suppose that $\bar{\delta}$ is the challenge signature δ^* . According to proof of knowledge, the extractor \mathcal{E} capable of extracting witnesses $w = (\mathbf{s}_{\pi'}, \mathbf{v}_{\pi'})$ of δ' , since condition (iii) holds, according to the CLTRS.Trace algorithm, there is one and only one vector at the identical position in the sequence of \mathbf{b}_i^* and \mathbf{b}'_i is equal, which means $\lfloor \tilde{\mathbf{A}}_{\Gamma_1} \cdot \mathbf{s}_{\pi^*} + \mathbf{A}_{\Gamma_2^*} \cdot \mathbf{v}_{\pi^*} \rfloor_p = \lfloor \tilde{\mathbf{A}}_{\Gamma_1} \cdot \mathbf{s}_{\pi'} + \mathbf{A}_{\Gamma_2^*} \cdot \mathbf{v}_{\pi'} \rfloor_p$, then we can obtain $\tilde{\mathbf{A}}_{\Gamma_1} \cdot (\mathbf{s}_{\pi^*} - \mathbf{s}_{\pi'}) = \tilde{\mathbf{e}}_s \pmod q$. This means that for a random matrix $\tilde{\mathbf{A}}_{\Gamma_1}$, we find the nonzero vector $\tilde{\mathbf{s}} = (\mathbf{s}_{\pi^*} - \mathbf{s}_{\pi'})$ and the vector $\tilde{\mathbf{e}}_s \in (-\frac{q}{p}, \frac{q}{p})^m$ satisfying $\tilde{\mathbf{A}}_{\Gamma_1} \cdot \tilde{\mathbf{s}} = \tilde{\mathbf{e}}_s$. So we have broken through the unique of function F with non-negligible probability $\frac{\epsilon}{2q_{h_1} \cdot q_s}$. Therefore, it is infeasible that the adversary \mathcal{A}_I will succeed in attacking in this case.

- **Adversary \mathcal{C}_{II} .** Adversary \mathcal{C}_{II} will construct the following game. If \mathcal{A}_I is capable of winning type I exculpability game by a probability of ϵ , then \mathcal{C}_{II} uses adversary \mathcal{A}_I to break multi-input correlation intractability of H with probability $\frac{\epsilon}{2}$.

2.1 **Setup.** Same as 1.1 Setup.

2.2 **Queries.** \mathcal{C}_{II} initializes initially empty lists $list_{H_1}$, $list_{H_2}$, $list_{H_3}$, $list_{CU}$, $list_{SQ}$ and keeps consistency of answers to adversaries by maintaining these tables.

- (a) **H₁ Query.** When \mathcal{A}_I submits a label Γ_i to this oracle, \mathcal{C}_{II} first checks if Γ_i in the $list_{H_1}$, if it exists, returns the corresponding \mathbf{A}_{Γ_i} . Otherwise, \mathcal{C}_{II} picks a random matrix $\mathbf{A}_{\Gamma_i} \leftarrow \mathbb{Z}_q^{m \times 2n}$, and returns it to \mathcal{A}_I .
- (b) **H₂ Query.** Same as 1.2 H₂ Query.
- (c) **H₃ Query.** Same as 1.2 H₃ Query.
- (d) **CreateUserQuery.** Same as 1.2 CreateUserQuery.
- (e) **PartialPrivateKeyQuery.** Same as 1.2 PartialPrivateKeyQuery.
- (f) **ReplacePublicKeyQuery.** Same as 1.2 ReplacePublicKeyQuery.
- (g) **SignQuery.** When \mathcal{A}_I submits a tuple (id_i, I, R, M) to this oracle, \mathcal{C}_{II} first checks if tuple (id_i, I, R, M) in the $list_{SQ}$, if it exists, returns the corresponding signature $\delta = (\alpha, \Pi)$. Otherwise, \mathcal{C}_{II} runs CLTRS.Sign to generate signature $\delta = (\alpha, \Pi)$ and returns it to \mathcal{A}_I , then adds the tuple $(id_i, I, R, M, \alpha, \Pi)$ to $list_{SQ}$.

2.3 **Forgery.** Same as 1.3 Forgery.

2.4 **Analysis.** In this case, neither tuple (I^*, R^*, \overline{M}) nor tuple (I^*, R^*, M') has been made $SQ(pp, \cdot, \cdot, id^*)$. That is, the user did not perform a signature, but the adversary \mathcal{A}_I forged two valid signatures to trap the user. Since condition (v) holds, according to the definition of *CLTRS.Trace* algorithm, the sequence \mathbf{b}_i and value of index π^* in the sequence \mathbf{b}'_i are equal which means

$$H_2(\Gamma^*, \overline{M}) + \pi^* \overline{\alpha} = H_2(\Gamma^*, M') + \pi^* \alpha',$$

Same as case 1, according to proof of knowledge, there exists an extractor \mathcal{E} capable of extracting witnesses $w = (\overline{\pi}, sk_{\overline{\pi}})$ and $w' = (\pi', sk_{\pi'})$ from $\overline{\Pi}$ and Π' respectively,

$$F_{sk_{\overline{\pi}}}(\Gamma^*) = H_2(\Gamma^*, \overline{M}) + \pi^* \overline{\alpha}, \text{ and } F_{sk_{\pi'}}(\Gamma^*) = H_2(\Gamma^*, M') + \pi^* \alpha',$$

Then, it holds that

$$H_{hk}(\Gamma^*, \overline{M}) + \pi^* \frac{F_{sk_{\overline{\pi}}}(\Gamma^*) - H_{hk}(\Gamma^*, \overline{M})}{\overline{\pi}} = H_{hk}(\Gamma^*, M') + \pi^* \frac{F_{sk_{\pi'}}(\Gamma^*) - H_{hk}(\Gamma^*, M')}{\pi'}$$

Here we also consider two situations, the first situation is when $\overline{\pi} = \pi'$, but in this case, $\overline{\pi} = \pi'$ means that there is a dishonest user who signs two different messages, so the output is targeted to that dishonest user. Another situation is when $\overline{\pi} \neq \pi'$, we can get

$$\overline{\pi} \pi' \cdot \overline{\mathbf{b}}_0 + \pi^* \pi' (\overline{\mathbf{b}}_{\overline{\pi}} - \overline{\mathbf{b}}_0) = \overline{\pi} \pi' \cdot \mathbf{b}'_0 + \pi^* \overline{\pi} (\mathbf{b}'_{\pi'} - \mathbf{b}'_0),$$

it means that $((\Gamma^*, \overline{M}), (\Gamma^*, M'), H_2(\Gamma^*, \overline{M}), H_2(\Gamma^*, M')) \in R_a$. As shown in [50], R_a is a sparse relation. Thus, if \mathcal{A}_I wins the type I exculpability game with non-negligible probability ϵ , then \mathcal{C}_{II} can break the multi-input correlation intractability of H_2 with non-negligible probability $\frac{\epsilon}{2}$. It contradicts the multi-input correlation intractability of H_2 .

To sum up, the adversary \mathcal{A}_I cannot make a successful attack to exculpability, so our scheme satisfies the type I exculpability. \square

Theorem 5 (Type II Exculpability). *If hash family H is multi-input correlation intractable, NIZKAoK is proof of knowledge, and the function F is unique with an intersection-free range, the CLTRS is type II exculpable under the ROM.*

Proof. The proving of this theorem is analogous to the way of proving Theorem 4. \mathcal{A}_{II} inquiries the random oracles H_1, H_2, H_3 up to $q_{h_1}, q_{h_2}, q_{h_3}$ times and makes a maximum of q_s signing queries, if it can break the exculpability with non-negligible probability ϵ , we may construct adversaries \mathcal{D}_I and \mathcal{D}_{II} to break uniqueness of F and multi-input correlation intractable of H with probability $\frac{\epsilon}{2q_{h_1} \cdot q_s}$ and $\frac{\epsilon}{2}$, respectively.

- **Adversary \mathcal{D}_I .** Given a challenge matrix $\tilde{\mathbf{A}}_{\Gamma_2} \in \mathbb{Z}_q^{m \times n}$ which is random, it is an instance of function uniqueness. The adversary \mathcal{D}_I need to look for a non-zero vector $\tilde{\mathbf{v}} \in \mathbb{Z}_q^n$ and a vector $\tilde{\mathbf{e}}_v \in (-\frac{q}{p}, \frac{q}{p})^m$ satisfying $\tilde{\mathbf{A}}_{\Gamma_2} \cdot \tilde{\mathbf{v}} = \tilde{\mathbf{e}}_v \pmod q$, adversary \mathcal{D}_I constructs the following game to attack the uniqueness of the function.

3.1 **Setup.** Similar to 1.1 **Setup**, except that msk does not require confidentiality but is sent to the adversary \mathcal{A}_{II} along with the public parameter pp .

3.2 **Queries.** \mathcal{D}_I initializes initially empty lists $list_{H_1}, list_{H_2}, list_{H_3}, list_{CU}, list_{PPK}, list_{SQ}$ and keeps consistency of answers to adversaries by maintaining these tables. We assume that adversary \mathcal{A}_{II} must have done the following queries before forging: \mathcal{A}_{II} will submit a query with label $\Gamma^* = (I^*, R^*)$ to H_1 for the i^* -th time where $i^* \in [1, q_{h_1}]$ and submit a query with tuple $(id^*, \Gamma^* = (I^*, R^*), M^*)$ to SQ for the j^* -th time where $j^* \in [1, q_s]$ and $id^* \in R^*$.

- (a) **H_1 Query.** For the $i \in [1, q_h]$ times of asking, if $i = i^*$, \mathcal{A}_{II} will submit the label Γ^* , \mathcal{D}_I picks a random matrix $\mathbf{A}_{\Gamma^*} \leftarrow \mathbb{Z}_q^{m \times n}$ and sets $\mathbf{A}_{\Gamma^*} = (\mathbf{A}_{\Gamma^*} | \tilde{\mathbf{A}}_{\Gamma_2})$, at last returns it to \mathcal{A}_{II} ; otherwise if $i \neq i^*$, \mathcal{A}_{II} will submit the label Γ_i , \mathcal{D}_I picks a random matrix $\mathbf{A}_{\Gamma_i} \leftarrow \mathbb{Z}_q^{m \times 2n}$ and returns it to \mathcal{A}_{II} .

- (b) **H₂ Query.** Same as 1.2 H₂ Query.
- (c) **H₃ Query.** When \mathcal{A}_{II} submits an id_i , \mathcal{D}_I randomly selects a vector $\mathbf{a}_{id_i} \leftarrow \mathbb{Z}_q^m$ and sets $H_3(id_i) = \mathbf{a}_{id_i}$, then returns \mathbf{a}_{id_i} to \mathcal{A}_{II} .
- (d) **CreateUserQuery.** When \mathcal{A}_{II} submits an id_i , \mathcal{D}_I first checks if id_i in the $list_{CU}$, if it exists, returns the corresponding key pairs (pk_i, sk_i) . Otherwise, \mathcal{D}_I calls $SampleD(A, T_A, \mathbf{a}_{id_i}, \sigma_2)$ to obtain \mathbf{s}_i when id_i exists in $list_{H_3}$, if id_i exists in $list_{H_3}$, \mathcal{D}_I calls $ExtractPartialPrivateKey$ algorithm to obtain \mathbf{s}_i , and then runs $SetSecretValue, SetPrivateKey, SetPublicKey$ algorithm to generate \mathbf{v}_i, sk_i and \mathbf{y}_i , sends (pk_i, sk_i) to adversary \mathcal{A}_{II} , then adds tuple $(id_i, \mathbf{s}_i, \mathbf{v}_i, sk_i, \mathbf{y}_i)$ to $list_{CU}$.
- (e) **ParticalPrivateKeyQuery.** When \mathcal{A}_{II} submits an id_i , \mathcal{D}_I first checks if the id_i exists in $list_{CU}$, if it exists, returns the corresponding \mathbf{s}_i ; otherwise \mathcal{D}_I checks if the id_i exists in $list_{PPK}$, if it exists, returns the corresponding \mathbf{s}_i ; otherwise \mathcal{D}_I checks if the id_i exists in $list_{H_3}$, if it exists, runs $SampleD(A, T_A, \mathbf{a}_{id_i}, \sigma_2)$ to get \mathbf{s}_i and returns it to \mathcal{A}_{II} ; otherwise, \mathcal{D}_I runs $H_3(id_i)$ to get \mathbf{a}_{id_i} , and returns $\mathbf{s}_i \leftarrow SampleD(A, T_A, \mathbf{a}_{id_i}, \sigma_2)$ to \mathcal{A}_{II} .
- (f) **ReplacePublicKeyQuery.** When \mathcal{A}_{II} submits an id_i and a \mathbf{y}'_i , \mathcal{D}_I replaces the user's public key \mathbf{y}_i with \mathbf{y}'_i .
- (g) **SignQuery.** For the $j \in [1, q_s]$ times of asking, if $j = j^*$, \mathcal{A}_{II} will submit a tuple (id^*, Γ^*, M^*) , instead of running $F_{sk_{id^*}}$, \mathcal{D}_I computes $\mathbf{b}_{j^*, \pi^*} = \lfloor \mathbf{A}_{\Gamma_1^*} \cdot \mathbf{s}_{\pi^*} + \tilde{\mathbf{A}}_{\Gamma_2^*} \cdot \mathbf{v}_{\pi^*} \rfloor_p$, and invokes simulator \mathcal{S}_{sim} to generate the proof, the remaining steps remain unchanged, returns the generated signature δ^* to \mathcal{A}_{II} . Otherwise if $j \neq j^*$, \mathcal{A}_{II} will submit a tuple (id_j, Γ_j, M_j) , \mathcal{D}_I runs F_{sk_j} to compute $\mathbf{b}_{j, \pi}$, the remaining steps remain unchanged, returns the generated signature δ_j to \mathcal{A}_{II} .

3.3 **Forgery.** Assume \mathcal{A}_{II} outputs two tuples $(I^*, R^*, \overline{M}, \overline{\delta})$ and (I^*, R^*, M', δ') s.t.

- (a) **CLTRS.Verify** $(pp, I^*, R^*, \overline{M}, \overline{\delta}) = 1$;
- (b) **CLTRS.Verify** $(pp, I^*, R^*, M', \delta') = 1$;
- (c) \mathcal{A}_{II} has not been queried about $RPKQ(pp, id^*)$ and $CUQ(pp, id^*)$ where $id^* \in R^*$;
- (d) \mathcal{A}_{II} has made at most one of $SQ(pp, id^*, I^*, R^*, \overline{M})$ and $SQ(pp, id^*, I^*, R^*, M')$;
- (e) **CLTRS.Trace** $(pp, I^*, R^*, \overline{M}, \overline{\delta}, M', \delta') = pk^*$.

Suppose π^* is the location of pk^* in R^* .

3.4 **Analysis.** Here, we assume that one of the two signatures output by the adversary is the challenge signature of the previous query, which indicates a situation where the user honestly generates a signature and the adversary \mathcal{A}_{II} forges another valid signature to trap the honest user. We supposes that δ is the challenge signature δ^* . According to proof of knowledge, the extractor \mathcal{E} capable of extracting witnesses $w = (\mathbf{s}_{\pi^*}, \mathbf{v}_{\pi'})$ of δ' , since condition (iii) holds, according to the $CLTRS.Trace$ algorithm, there is one and only one vector at the identical position in the sequence of \mathbf{b}'_i and \mathbf{b}'_i is equal, which means $\lfloor \mathbf{A}_{\Gamma_1^*} \cdot \mathbf{s}_{\pi^*} + \tilde{\mathbf{A}}_{\Gamma_2} \cdot \mathbf{v}_{\pi^*} \rfloor_p = \lfloor \mathbf{A}_{\Gamma_1} \cdot \mathbf{s}_{\pi^*} + \tilde{\mathbf{A}}_{\Gamma_2} \cdot \mathbf{v}_{\pi'} \rfloor_p$, then we can obtain $\tilde{\mathbf{A}}_{\Gamma_2} \cdot (\mathbf{v}_{\pi^*} - \mathbf{v}_{\pi'}) = \tilde{\mathbf{e}}_v \pmod q$. This means that for a random matrix $\tilde{\mathbf{A}}_{\Gamma_2}$, we find the nonzero vector $\tilde{\mathbf{v}} = (\mathbf{v}_{\pi^*} - \mathbf{v}_{\pi'})$ and the vector $\tilde{\mathbf{e}}_v \in (-\frac{q}{p}, \frac{q}{p})^m$ satisfying $\tilde{\mathbf{A}}_{\Gamma_2} \cdot \tilde{\mathbf{v}} = \tilde{\mathbf{e}}_v$. So we have broken through the unique of function F with non-negligible probability $\frac{\epsilon}{2q_{h_1} \cdot q_s}$. Therefore, it is infeasible that the adversary \mathcal{A}_{II} will succeed in attacking in this case.

- **Adversary \mathcal{D}_{II} .** Adversary \mathcal{D}_{II} will construct the following game. If adversary \mathcal{A}_{II} can break the type II exculpability game with probability ϵ , then \mathcal{D}_{II} uses adversary \mathcal{A}_{II} to break the multi-input correlation intractability of the hash function H with probability $\frac{\epsilon}{2}$.

4.1 **Setup.** Same as 3.1 Setup.

4.2 **Queries.** \mathcal{D}_{II} initializes initially empty lists $list_{H_1}, list_{H_2}, list_{H_3}, list_{CU}, list_{PPK}, list_{SQ}$ and keeps consistency of answers to adversaries by maintaining these tables.

- (a) **H₁ Query.** Same as 2.2 **H₁ Query.**
- (b) **H₂ Query.** Same as 1.2 **H₂ Query.**
- (c) **H₃ Query.** Same as 3.2 **H₃ Query.**
- (d) **CreateUserQuery.** Same as 3.2 **CreateUserQuery.**
- (e) **PartialPrivateKeyQuery.** Same as 3.2 **PartialPrivateKeyQuery.**
- (f) **ReplacePublicKeyQuery.** Same as 3.2 **ReplacePublicKeyQuery.**
- (g) **SignQuery.** Same as 2.2 **SignQuery.**

4.3 **Forgery.** Same as 3.3 **Forgery.**

4.4 **Analysis.** In this case, neither tuple (I^*, R^*, \bar{M}) nor tuple (I^*, R^*, M') has been made $SQ(pp, \cdot, \cdot, id^*)$. Similar to the **Analysis** in 4.4, we can obtain the following equation as well

$$\bar{\pi}\pi' \cdot \bar{\mathbf{b}}_0 + \pi^* \pi' (\bar{\mathbf{b}}_{\pi} - \bar{\mathbf{b}}_0) = \bar{\pi}\pi' \cdot \mathbf{b}'_0 + \pi^* \bar{\pi} (\mathbf{b}'_{\pi'} - \mathbf{b}'_0),$$

it means that $((\Gamma^*, \bar{M}), (\Gamma^*, M'), H_2(\Gamma^*, \bar{M}), H_2(\Gamma^*, M')) \in R_a$. As shown in [50], R_a is a sparse relation. Thus, if \mathcal{A}_{II} break the type II exculpability game with non-negligible probability ϵ , then \mathcal{D}_{II} may break the multi-input correlation intractability of H_2 with non-negligible probability $\frac{\epsilon}{2}$. It contradicts the multi-input correlation intractability of H_2 .

To sum up, the adversary \mathcal{A}_{II} cannot make a successful attack to exculpability, so our scheme satisfies the type II exculpability.

□

In summary, our scheme satisfies type I exculpability for \mathcal{A}_I and type II exculpability for \mathcal{A}_{II} . Therefore, our scheme satisfies exculpability.

7. Efficiency

In the aspect of efficiency, we concentrate on the signature length of the scheme. Looking at the form of the signature it is easy to see that it is made up of two components α and Π , one being the parameters used by the tracing algorithm and the other being the proof Π generated by the NIZK protocol, as the NIZK proof protocol is obtained by the interactive ZK protocol employing the Fiat-Shamir transformation. That's why it is also known as SPK. The scale of the signature δ of our scheme depends mainly on the scale of ZK proof Π , so the scale of the signature δ can be estimated by analysing the size of ZK proof Π . As shown in [53], the Π produced by efficient ZKAoK in Section 4 consists of two parts, a commitment and N tuples. So, we can obtain

$$\|\Pi\| = (\log(2p + 1) + \kappa + (3l_1 + 2l_2 + 2n + 2\ell) \cdot \log q) \cdot N + (l_1 + n) \cdot \log q,$$

where n is the witness size and ℓ is the size of \mathcal{M} .

The proved statement contains five equations. Every relational equation can be transformed into an example of the relation \mathfrak{R} . The primitive to be argued by Equations (1) and (4) is linear equation with short solution; therefore, Equations (1) and (4) can be integrated together and then reduced to the relation \mathfrak{R} . According to the conclusion in the section above, we can know that the size of witness and M are both $(\hat{m} + 2n)k_q + 2mk_p$; while the primitive to be argued by Equations (2) and (3) is PRF preimage, we can know that the size of witness and M both are $(2n + 2m)k_q + 2mk + 2mk_p$. The last relation $\mathfrak{R}_{(5)}$ argues knowledge of a member in the accumulator. Just like analysis in [53], the length of witness is $2\ell + 4l_1\ell + 2l_2\ell$ and the size of \mathcal{M} is $\ell + 2l_1\ell + 2l_2\ell$. We combining above five relations to get the size of the witness is

$$\mathfrak{W} = (3n + \hat{m} + 2m)k_q + 2mk + 2mk_p + 2\ell + 4n\ell + 2nk_q\ell,$$

and the size of \mathcal{M} is

$$\mathfrak{M} = (3n + \hat{m} + 2m)k_q + 2mk + 2mk_p + \ell + 4n\ell + 2nk_q\ell,$$

(the repeated part such as $\mathbf{a}_{id_i}, \mathbf{s}_i, \mathbf{b}_i, \mathbf{c}_i$, we only need to counted only once).

We set $\hat{p}, \hat{\kappa}, \hat{l}_1, \hat{l}_2, \hat{N}$ be parameters used in NIZK protocol. So we have

$$\|\Pi\| = (\log(2\hat{p} + 1) + \hat{\kappa} + (3\hat{l}_1 + 2\hat{l}_2) \cdot k + 2\mathfrak{W} + 2\mathfrak{M}) \cdot \hat{N} + (\hat{l}_1 + \mathfrak{W}) \cdot k$$

bits.

Compared with [51], msk and sk of our scheme is shorter, and although the signature scale is larger compared to [51], our scheme implements linkability and public traceability. Compared with [52], our scheme has smaller sk and the scheme in [52] does not implement linkability and traceability;

Compared with [50], in the case of the same public key and secret key, the communication cost of our scheme is relatively small, the size of signature of [50] is $t \cdot \mathcal{O}(n \log^3 q + \log \mathcal{L} \cdot n \log q)$, and the signature of our scheme is $\hat{N} \cdot \mathcal{O}(n \log^2 q + \log \mathcal{L} \cdot n \log q)$ under the same parameter setting. \hat{N} and t , respectively, are the number of protocols that need to be performed to achieve negligible soundness error. Since the zero-knowledge single-execution protocol used by our scheme has a smaller soundness error, less time is required to reach a negligible soundness error, and thus the resulting signature size is smaller. The detailed comparison is shown in Tables 1 and 2 below.

Table 1. Theoretical estimation of key sizes and signature sizes of lattice-based ring signature.

Scheme	PK Size	MSK Size	SK Size	Signature Size
CLR [51]	$\mathcal{O}(n^2 \log^2 q)$	$\mathcal{O}(n^2 \log^3 q)$	$\mathcal{O}(n^2 \log^3 q)$	$\mathcal{O}(\mathcal{L} \cdot n \log^2 q)$
CLR [52]	$\mathcal{O}(n^2 \log^2 q + \mathcal{L} \cdot n^2 \log q)$	$\mathcal{O}(n \log q)$	$\mathcal{O}(n^2 \log^3 q + \mathcal{L} \cdot n^2 \log^2 q)$	$\mathcal{O}(n^2 \log^2 q)$
TRS [50]	$\mathcal{O}(n \log^3 q)$	-	$\mathcal{O}(n \log^2 q)$	$t \cdot \mathcal{O}(n \log^4 q + \log \mathcal{L} \cdot n \log^2 q)$
Ours	$\mathcal{O}(n \log^3 q)$	$\mathcal{O}(n \log q)$	$\mathcal{O}(n \log^2 q)$	$\hat{N} \cdot \mathcal{O}(n \log^3 q + \log \mathcal{L} \cdot n \log^2 q)$

Table 2. Functional comparison of lattice-based ring signature.

Scheme	Quantum Resistant	Anonymity	Unforgeability	Linkability	Traceability	Certificateless
CLR [51]	✓	✓	✓	×	×	✓
CLR [52]	✓	✓	✓	×	×	✓
TRS [50]	✓	✓	✓	✓	✓	×
Ours	✓	✓	✓	✓	✓	✓

$-|ID|$ denotes the length of the ID, \mathcal{L} represents the size of the ring, PK, MSK, SK denote public key, master secret key, secret key respectively.

8. Conclusions

In this paper, we constructed a CLTRS scheme on lattice, and it satisfies tag-linkability, anonymity, and exculpability under the ROM. We used a more efficient ZK protocol to replace the Stern-like protocol, the soundness error of the single-execution protocol is reduced from 2/3 to 1/poly, thus reducing the communication cost. And our scheme not only eliminates the burden of certificate management, but also eliminates the problem of key escrow. In the future work, we can consider using ideal lattices or modular lattices to replace standard lattices, and construct the scheme under standard model. It is also worth studying to replace zero-knowledge proof by attribute-based signature scheme so as to avoid the use of zero-knowledge proof.

Author Contributions: Conceptualization, J.L., J.H. and Q.H.; methodology, J.L., J.H. and Q.H.; writing—original draft preparation, J.L.; writing—review and editing, Q.H., L.L. and M.H.A.A.; supervision, Q.H., L.L. and M.H.A.A.; project administration, Q.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Major Program of Guangdong Basic and Applied Research (2019B030302008), National Natural Science Foundation of China (62272174, 61872152), and Science and Technology Program of Guang-zhou (201902010081).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare that they have no competing interests.

Abbreviations

The following abbreviations are used in this manuscript:

Abbreviation	Full Name
RS	Ring Signature
LRS	Linkable Ring Signature
TRS	Traceable Ring Signature
SIS	Short Integer Solution
DLWE	Decisional Learning With Error
ROM	Random Oracle Model
QROM	Quantum Random Oracle Model
ZK	Zero-Knowledge
ZKAoK	Zero-Knowledge Argument of Knowledge
NIZKAoK	Non-interactive Zero-Knowledge Argument of Knowledge
PRF	Pseudorandomness Function
VANET	Vehicular Ad-Hoc Network
PKI	Public Key Infrastructure
CA	Certificate Authority
CMP	Certificate Management Problem
KGC	Key Generation Center
KEP	Key Escrow Problem
SPK	Signature Proof Knowledge
CLPKC	Certificate-less Public Key Encryption
CLTRS	Certificate-less Traceable Ring Signature
CUQ	Create User Query
PPKQ	Partial Private Key Query
RPKQ	Replace Public Key Query
SQ	Sign Query

References

1. Chow, S.S.M.; Liu, J.K.; Wong, D.S. Robust Receipt-Free Election System with Ballot Secrecy and Verifiability. In Proceedings of the Network and Distributed System Security Symposium, NDSS 2008, The Internet Society, San Diego, California, USA, 10–13 February 2008.
2. Zhou, Y.; Dong, S.; Yang, Y. Ring signature scheme based on lattice and its application on anonymous electronic voting. *Ksii Trans. Internet Inf. Syst. (Tiis)* **2022**, *16*, 287–304.
3. Tsang, P.P.; Wei, V.K. Short Linkable Ring Signatures for E-Voting, E-Cash and Attestation. In Proceedings of the Information Security Practice and Experience, First International Conference, ISPEC 2005, Singapore, 11–14 April 2005; Volume 3439, pp. 48–60. [[CrossRef](#)]
4. Tang, F.; Pang, J.; Cheng, K.; Gong, Q. Multiauthority Traceable Ring Signature Scheme for Smart Grid Based on Blockchain. *Wirel. Commun. Mob. Comput.* **2021**, *2021*, 5566430:1–5566430:9. [[CrossRef](#)]
5. Han, L.; Cao, S.; Yang, X.; Zhang, Z. Privacy Protection of VANET Based on Traceable Ring Signature on Ideal Lattice. *IEEE Access* **2020**, *8*, 206581–206591. [[CrossRef](#)]
6. Gu, K.; Wang, L.; Wu, N.; Liao, N. Traceable Certificateless Ring Signature Scheme for no Full Anonymous Applications. *Int. J. Netw. Secur.* **2018**, *20*, 762–773.
7. Diffie, W.; Hellman, M.E. New directions in cryptography. *IEEE Trans. Inf. Theory* **1976**, *22*, 644–654. [[CrossRef](#)]
8. Shamir, A. Identity-Based Cryptosystems and Signature Schemes. In Proceedings of the Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, CA, USA, 19–22 August 1984; Volume 196, pp. 47–53. [[CrossRef](#)]
9. Al-Riyami, S.S.; Paterson, K.G. Certificateless Public Key Cryptography. In Proceedings of the Advances in Cryptology—ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, 30 November–4 December 2003; Volume 2894, pp. 452–473. [[CrossRef](#)]

10. Bisheh-Niasar, M.; Azarderakhsh, R.; Kermani, M.M. Cryptographic Accelerators for Digital Signature Based on Ed25519. *IEEE Trans. Very Large Scale Integr. Syst.* **2021**, *29*, 1297–1305. [[CrossRef](#)]
11. Shor, P.W. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In Proceedings of the 35th Annual Symposium on Foundations of Computer Science, IEEE Computer Society, Santa Fe, NM, USA, 20–22 November 1994; pp. 124–134. [[CrossRef](#)]
12. Ni, Z.; Kundi, D.; O'Neill, M.; Liu, W. A High-Performance SIKE Hardware Accelerator. *IEEE Trans. Very Large Scale Integr. Syst.* **2022**, *30*, 803–815. [[CrossRef](#)]
13. Tian, J.; Wu, B.; Wang, Z. High-Speed FPGA Implementation of SIKE Based on an Ultra-Low-Latency Modular Multiplier. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2021**, *68*, 3719–3731. [[CrossRef](#)]
14. Sarker, A.; Kermani, M.M.; Azarderakhsh, R. Efficient Error Detection Architectures for Postquantum Signature Falcon's Sampler and KEM SABER. *IEEE Trans. Very Large Scale Integr. Syst.* **2022**, *30*, 794–802. [[CrossRef](#)]
15. Berzati, A.; Viera, A.C.; Chartouni, M.; Madec, S.; Vergnaud, D.; Vigilant, D. A Practical Template Attack on CRYSTALS-Dilithium. Cryptology ePrint Archive, Paper 2023/050. Available online: <https://eprint.iacr.org/2023/050> (accessed on 20 February 2023).
16. Ajtai, M. Generating Hard Instances of Lattice Problems (Extended Abstract). In Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; pp. 99–108. [[CrossRef](#)]
17. Rivest, R.L.; Shamir, A.; Tauman, Y. How to Leak a Secret. In Proceedings of the Advances in Cryptology—ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, 9–13 December 2001; Volume 2248, pp. 552–565. [[CrossRef](#)]
18. Zhang, F.; Kim, K. ID-Based Blind Signature and Ring Signature from Pairings. In Proceedings of the Advances in Cryptology—ASIACRYPT 2002, 8th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, New Zealand, 1–5 December 2002; Volume 2501, pp. 533–547. [[CrossRef](#)]
19. Dodis, Y.; Kiayias, A.; Nicolosi, A.; Shoup, V. Anonymous Identification in Ad Hoc Groups. In Proceedings of the Advances in Cryptology—EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, 2–6 May 2004; Volume 3027, pp. 609–626. [[CrossRef](#)]
20. Herranz, J.; Sáez, G. New Identity-Based Ring Signature Schemes. In Proceedings of the Information and Communications Security, 6th International Conference, ICICS 2004, Malaga, Spain, 27–29 October 2004; Volume 3269, pp. 27–39. [[CrossRef](#)]
21. Chan, T.K.; Fung, K.; Liu, J.K.; Wei, V.K. Blind Spontaneous Anonymous Group Signatures for Ad Hoc Groups. In Proceedings of the Security in Ad-Hoc and Sensor Networks, First European Workshop, ESAS 2004, Berlin/Heidelberg, Germany, 6 August 2004; Volume 3313, pp. 82–94. [[CrossRef](#)]
22. Chow, S.S.M.; Yiu, S.; Hui, L.C.K. Efficient Identity Based Ring Signature. In Proceedings of the Applied Cryptography and Network Security, Third International Conference, ACNS 2005, New York, NY, USA, 7–10 June 2005; Volume 3531, pp. 499–512. [[CrossRef](#)]
23. Chen, Y.; Susilo, W.; Mu, Y. Convertible identity-based anonymous designated ring signatures. *Int. J. Secur. Netw.* **2006**, *1*, 218–225. [[CrossRef](#)]
24. Liu, J.K.; Wei, V.K.; Wong, D.S. Linkable Spontaneous Anonymous Group Signature for Ad Hoc Groups (Extended Abstract). In Proceedings of the Information Security and Privacy: 9th Australasian Conference, ACISP 2004, Sydney, Australia, 13–15 July 2004; Volume 3108, pp. 325–335. [[CrossRef](#)]
25. Liu, J.K.; Wong, D.S. Enhanced Security Models and a Generic Construction Approach for Linkable Ring Signature. *Int. J. Found. Comput. Sci.* **2006**, *17*, 1403–1422. [[CrossRef](#)]
26. Bender, A.; Katz, J.; Morselli, R. Ring Signatures: Stronger Definitions, and Constructions without Random Oracles. *J. Cryptol.* **2009**, *22*, 114–138. [[CrossRef](#)]
27. Au, M.H.; Liu, J.K.; Susilo, W.; Yuen, T.H. Certificate Based (Linkable) Ring Signature. In Proceedings of the Information Security Practice and Experience, Third International Conference, ISPEC 2007, Hong Kong, China, 7–9 May 2007, Volume 4464, pp. 79–92. [[CrossRef](#)]
28. 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.* **2013**, *56*, 407–421. [[CrossRef](#)]
29. Deng, L.; Jiang, Y.; Ning, B. Identity-Based Linkable Ring Signature Scheme. *IEEE Access* **2019**, *7*, 153969–153976. [[CrossRef](#)]
30. Deng, L.; Shi, H.; Gao, Y. Certificateless Linkable Ring Signature Scheme. *IEEE Access* **2020**, *8*, 54641–54651. [[CrossRef](#)]
31. Fujisaki, E.; Suzuki, K. Traceable Ring Signature. In Proceedings of the Public Key Cryptography—PKC 2007, 10th International Conference on Practice and Theory in Public-Key Cryptography, Beijing, China, 16–20 April 2007, Volume 4450, pp. 181–200. [[CrossRef](#)]
32. Fujisaki, E. Sub-Linear Size Traceable Ring Signatures without Random Oracles. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **2012**, *95-A*, 151–166. [[CrossRef](#)]
33. 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.* **2013**, *469*, 1–14. [[CrossRef](#)]
34. Bultel, X.; Lafourcade, P. k-Times Full Traceable Ring Signature. In Proceedings of the 11th International Conference on Availability, Reliability and Security, ARES 2016 IEEE Computer Society, Salzburg, Austria, 31 August–2 September 2016; pp. 39–48. [[CrossRef](#)]

35. Gu, K.; Dong, X.; Wang, L. Efficient traceable ring signature scheme without pairings. *Adv. Math. Commun.* **2020**, *14*, 207–232. [[CrossRef](#)]
36. Peng, X.; Gu, K.; Liu, Z.; Zhang, W. Traceable Identity-Based Ring Signature for Protecting Mobile IoT Devices. In Proceedings of the Data Mining and Big Data—6th International Conference, DMBD 2021, Guangzhou, China, 20–22 October 2021; Volume 1454, pp. 158–166. [[CrossRef](#)]
37. Brakerski, Z.; Kalai, Y.T. A Framework for Efficient Signatures, Ring Signatures and Identity Based Encryption in the Standard Model. *IACR Cryptol. ePrint Arch.* **2010**, 2010, 86.
38. Tian, M.M.; Huang, L.S.; Yang, W. Efficient lattice-based ring signature scheme. *Jisuanji Xuebao (Chin. J. Comput.)* **2012**, *35*, 712–718. [[CrossRef](#)]
39. Lyubashevsky, V. Lattice Signatures without Trapdoors. In Proceedings of the Advances in Cryptology—EUROCRYPT 2012—31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, 15–19 April 2012; Volume 7237, pp. 738–755. [[CrossRef](#)]
40. 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 Proceedings of the Advances in Cryptology—EUROCRYPT 2016—35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, 8–12 May 2016; Volume 9666, pp. 1–31. [[CrossRef](#)]
41. Wang, S.; Zhao, R.; Zhang, Y. Lattice-based ring signature scheme under the random oracle model. *Int. J. High Perform. Comput. Netw.* **2018**, *11*, 332–341. [[CrossRef](#)]
42. Lu, X.; Au, M.H.; Zhang, Z. Raptor: A Practical Lattice-Based (Linkable) Ring Signature. In Proceedings of the Applied Cryptography and Network Security—17th International Conference, ACNS 2019, Bogota, Colombia, 5–7 June 2019; Volume 11464, pp. 110–130. [[CrossRef](#)]
43. Melchor, C.A.; Bettaieb, S.; Boyen, X.; Fousse, L.; Gaborit, P. Adapting Lyubashevsky’s Signature Schemes to the Ring Signature Setting. In Proceedings of the Progress in Cryptology—AFRICACRYPT 2013, 6th International Conference on Cryptology in Africa, Cairo, Egypt, 22–24 June 2013; Volume 7918, pp. 1–25. [[CrossRef](#)]
44. Yang, R.; Au, M.H.; Lai, J.; Xu, Q.; Yu, Z. Lattice-Based Techniques for Accountable Anonymity: Composition of Abstract Stern’s Protocols and Weak PRF with Efficient Protocols from LWR. *IACR Cryptol. ePrint Arch.* **2017**, 2017, 781.
45. Torres, W.A.A.; Steinfeld, R.; Sakzad, A.; Liu, J.K.; Kuchta, V.; Bhattacharjee, N.; Au, M.H.; Cheng, J. Post-Quantum One-Time Linkable Ring Signature and Application to Ring Confidential Transactions in Blockchain (Lattice RingCT v1.0). In Proceedings of the Information Security and Privacy—23rd Australasian Conference, ACISP 2018, Wollongong, NSW, Australia, 11–13 July 2018; Volume 10946, pp. 558–576. [[CrossRef](#)]
46. Baum, C.; Lin, H.; Oechsner, S. Towards Practical Lattice-Based One-Time Linkable Ring Signatures. In Proceedings of the Information and Communications Security—20th International Conference, ICICS 2018, Lille, France, 29–31 October 2018; Volume 11149, pp. 303–322. [[CrossRef](#)]
47. Le, H.Q.; Vo, B.; Duong, D.H.; Susilo, W.; Le, N.T.; Fukushima, K.; Kiyomoto, S. Identity-Based Linkable Ring Signatures From Lattices. *IEEE Access* **2021**, *9*, 84739–84755. [[CrossRef](#)]
48. Hu, M.; Liu, Z. Lattice-Based Linkable Ring Signature in the Standard Model. *IACR Cryptol. ePrint Arch.* **2022**, 9, 101.
49. Ye, Q.; Wang, M.; Meng, H.; Xia, F.; Yan, X. Efficient Linkable Ring Signature Scheme over NTRU Lattice with Unconditional Anonymity. *Comput. Intell. Neurosci.* **2022**, 2022, 8431874. [[CrossRef](#)]
50. Feng, H.; Liu, J.; Li, D.; Li, Y.; Wu, Q. Traceable ring signatures: General framework and post-quantum security. *Des. Codes Cryptogr.* **2021**, *89*, 1111–1145. [[CrossRef](#)]
51. Zhang, M.; Chen, X. A Post-quantum Certificateless Ring Signature Scheme for Privacy-Preserving of Blockchain Sharing Economy. In Proceedings of the Artificial Intelligence and Security—7th International Conference, ICAIS 2021, Dublin, Ireland, 19–23 July 2021; Volume 12737, pp. 265–278. [[CrossRef](#)]
52. Dong, S.; Zhou, Y.; Yang, Y.; Yao, Y. A certificateless ring signature scheme based on lattice. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e7385. [[CrossRef](#)]
53. Yang, R.; Au, M.H.; Zhang, Z.; Xu, Q.; Yu, Z.; Whyte, W. Efficient Lattice-Based Zero-Knowledge Arguments with Standard Soundness: Construction and Applications. In Proceedings of the Advances in Cryptology—CRYPTO 2019—39th Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2019; Volume 11692, pp. 147–175. [[CrossRef](#)]
54. Regev, O. On lattices, learning with errors, random linear codes, and cryptography. In Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, 22–24 May 2005; pp. 84–93. [[CrossRef](#)]
55. Banerjee, A.; Peikert, C.; Rosen, A. Pseudorandom Functions and Lattices. In Proceedings of the Advances in Cryptology—EUROCRYPT 2012—31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, 15–19 April 2012; Volume 7237, pp. 719–737. [[CrossRef](#)]
56. Micciancio, D.; Peikert, C. Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. In Proceedings of the Advances in Cryptology—EUROCRYPT 2012—31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, 15–19 April 2012; Volume 7237, pp. 700–718. [[CrossRef](#)]
57. Holmgren, J.; Lombardi, A. Cryptographic Hashing from Strong One-Way Functions (Or: One-Way Product Functions and Their Applications). In Proceedings of the 59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, 7–9 October 2018; pp. 850–858. [[CrossRef](#)]

58. Libert, B.; Ling, S.; Nguyen, K.; Wang, H. Zero-Knowledge Arguments for Lattice-Based PRFs and Applications to E-Cash. In Proceedings of the Advances in Cryptology—ASIACRYPT 2017—23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, 3–7 December 2017; Volume 10626, pp. 304–335. [[CrossRef](#)]
59. Ling, S.; Nguyen, K.; Stehlé, D.; Wang, H. Improved Zero-Knowledge Proofs of Knowledge for the ISIS Problem, and Applications. In Proceedings of the Public-Key Cryptography—PKC 2013—16th International Conference on Practice and Theory in Public-Key Cryptography, Nara, Japan, 26 February–1 March 2013; Volume 7778, pp. 107–124. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.