Hindawi
Security and Communication Networks
Volume 2022, Article ID 3938321, 10 pages
https://doi.org/10.1155/2022/3938321

Research Article

A New Code-Based Traceable Ring Signature Scheme

Yanhong Qi [1,2] **and Li-Ping Wang** [1,2]

[1]*State Key Laboratory of Information Security, Institute of Information Engineering, CAS, Beijing, China*
[2]*School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China*

Correspondence should be addressed to Li-Ping Wang; wangliping@iie.ac.cn

Received 19 November 2021; Revised 29 January 2022; Accepted 10 March 2022; Published 29 April 2022

Academic Editor: Shichang Xuan

Traceable ring signatures (TRS) can reveal the identity of the signer if he signs two different messages on the same tag in the group of users. They are widely used in e-voting and cryptocurrencies such as Monero. However, there is still no secure code-based TRS scheme in the random oracle model (ROM). In this paper, we propose a code-based TRS scheme whose security is based on the hardness of the syndrome decoding problem and 2-regular null syndrome decoding problem. We show that our scheme is secure in the ROM in terms of tag-linkability, anonymity, and culpability. The signature size of our scheme is logarithmic in terms of the ring size.

1. Introduction

Ring signatures can be regarded as special group signatures, but they are differing from group signatures in that there is no group administrator in ring signatures. So, we cannot trace the real identity of the signer like group signatures. Ring signatures allow users from a group to sign messages on behalf of the group. The verifier of the ring signature can check the correctness of the signature but cannot know which person in the group is the real signer. If the same signer generates two signatures, the verifier cannot identity the signer. However, in many application scenarios [1–4], the signature represents the use by the signer of his rights, so it is important to be able to trace the signer who signs twice. For example, in a voting system for an event, users in the group can anonymously vote for the event. Dishonest users can use anonymity to vote multiple times for their own benefit. Therefore, in this case, a verifier wants to track the identities of dishonest users while protecting the privacy of honest users.

In order to solve this problem, the first traceable ring signature (TRS) scheme was proposed by Fujisaki and Suzuki [5]. TRS schemes have many applications in e-voting and cryptocurrencies such as Monero. A TRS scheme can track the dishonest's information while protecting the privacy of honest users. A TRS scheme has a tag that contains public keys of the group members and an issue. For example, an issue may be an election or a social problem. Group members can post any signed and anonymous opinions on the issue only once per tag. If a member wants to support his first opinion and submits another signed opinion, his identity will be immediately revealed. More specifically, if the signer signs the same message twice with the same tag, one will see that the two signatures are linked. If the signer signs different messages with the same tag, a TRS scheme can not only prove that the two signatures are related but also expose the identity of the signer.

There are many TRS schemes [1–3, 6–8] based on factoring and discrete logarithm problems. With the emergence of large-scale quantum computers, most classic asymmetric cryptography is threatened because Shor's algorithm can solve factoring and discrete logarithm problems in polynomial times [9]. Therefore, the postquantum secure TRS schemes have attracted much attention. Branco and Mateus proposed the first post-quantum TRS scheme which is based on coding theory [10]. However, the TRS was pointed out to be unsafe due to the use of the Cramer-Damgård-Schoenmakers (CDS) framework [11] to construct OR relation in [12]. The authors in [12] also proposed a general framework and instantiated the framework with lattice-

based building blocks. A hash-based one-time traceable ring signature was proposed in [13]. [14] was an extension of the work in [12]. In [14], not only lattice-based instantiation of the framework was given but also the instantiation based on symmetric-key primitives. Unlike the framework proposed in [14], we use a different way to construct the signing process and the detailed information is described in III.

Code-based cryptography is a hot topic because it is thought to be secure against attacks by quantum computers. The first code-based signature scheme appeared in 2001 [15], And then, code-based signature schemes have developed greatly in the last years [16–18]. The first code-based ring signature was proposed in 2007 [19]. After that, many variations related to ring signatures have been proposed, such as linkable ring signature schemes [20], threshold ring signature schemes [21–23], and group signature schemes [24–28]. However, there is still no secure code-based TRS scheme in the random oracle model (ROM).

Our Contributions. In this paper, we propose a new code-based TRS scheme in the ROM. Our scheme is an improvement on [10]. Instead of using the CDS framework in [10], we employ an accumulator [29] to construct the OR relation. Our scheme is based on the syndrome decoding (SD) problem and 2-regular null syndrome decoding (2-RNSD) problem, and we give the security analysis of the scheme in the ROM. More precisely, we construct a new protocol called Acc-GStern's protocol by adding new relationships to the GStern's protocol [10]. The GStern's protocol is for a prover to prove that he has the knowledge of an error vector $\mathbf{e}$ for two instances of the SD problem. The Acc-GStern's protocol is for one prover to prove that not only does he have the knowledge of a witness $\mathbf{e}$ for one of the several instances of the general syndrome decoding (GSD) problem but also he has values that can correctly be accumulated into the root of the code-based Merkle-tree.

Consider members in the ring. For $1 \le i \le L$, let $(\mathbf{H}, \mathbf{s}_i)$ be the public keys of each user $\mathcal{P}_i$, and the corresponding private key is $\mathbf{e}_i$. To sign a message, $\mathcal{P}_i$ collects the public keys of remaining $L-1$ members in the ring and uses them to get a hash value $\widetilde{H}$. Then, $\mathcal{P}_i$ uses his secret key to get a vector $\mathbf{r}_i = \widetilde{H}\mathbf{e}_i^T$ and adds a hash function on $\mathbf{r}_i$ to get. Next, the user uses a special hash function [29] to get the leaves of the Merkle-tree and applies the Fiat-Shamir transform [30] on the Acc-GStern's protocol to get the signature. If the signer signs two different messages in the same ring, the identity of the signer is revealed.

The remainder of this paper is as follows: Section 2 introduces the necessary preliminary knowledge needed in this paper. In Section 3, we present our TRS scheme. The security proof and analysis of the scheme are given in Section 4. Efficiency is shown in Section 5. Finally, the conclusion is drawn in Section 6.

2. Preliminaries

2.1. Notations. Let us start with some notations. We write $\mathbb{Z}_2$ as the set $\{0, 1\}$ and use $[n]$ to denote the set. We use $(\mathbf{r}_i)_{i\in[L]}$ to represent the sequence $(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_L)$. Vectors and matrices will be represented in boldface lowercase letters and bold capital letters, respectively. If S is a finite set, it means that y is chosen uniformly at random from S. If $\mathcal{A}$ is an algorithm, we use $y \leftarrow \mathcal{A}(x)$ to show that when running with input x, the output of this algorithm is y. Let $w(\mathbf{y})$ be the Hamming weight of the vector $\mathbf{y}$. We represent the transpose of $\mathbf{b}$ in terms of $\mathbf{b}^T$. The bit-wise addition operation modulo 2 is denoted by $\oplus$. The function A which is negligible under the parameter n is denoted by $\mathrm{negl}(n)$, i.e., $A \le 1/\mathrm{poly}(n)$, where $\mathrm{poly}(n)$ represents any polynomial in n.

2.2. Hard Problems. In this section, we are going to cover some of the difficult problems used later.

Problem 1 (syndrome decoding (SD) problem). *Let $\mathbf{H} \in \{0,1\}^{(n-k)\times n}$ be a parity-check matrix of an $[n, k]$ random linear code, $\mathbf{s} \in \{0,1\}^{n-k}$ be a binary vector, $t \ge 0$ be an integer, find $\mathbf{e} \in \mathbb{Z}_2^n$ such that $w(\mathbf{e}) \le t$, and $\mathbf{H}\mathbf{e}^T = \mathbf{s}^T$.*

This problem is proven to be NP-complete in the worst case [31]. The distance between the uniform distribution over $\mathbb{Z}_2^{(n-k)\times n} \times \mathbb{Z}_2^{n-k}$ and $(\mathbf{H}, \mathbf{H}\mathbf{e}^T)$ is negligible [25].

Lemma 1 (see [10]). *Let n, $k' \in \mathbb{Z}$, and $k' \le n/2$. Let $\mathbf{H}$ be a random matrix in $\mathbb{Z}_2^{k'\times n}$ and $\mathbf{s}$ be a random vector in $\mathbb{Z}_2^{k'}$. The probability that one can find $\mathbf{e}$ that satisfies $\mathbf{H}\mathbf{e}^T = \mathbf{s}^T$ is negligible.*

Problem 2 (general syndrome decoding (GSD) problem). *Let $\mathbf{H}$, $\mathbf{G} \in \{0,1\}^{(n-k)\times n}$ be binary matrices, $\mathbf{s}, \mathbf{r}$ be binary vectors, and $t \ge 0$ be an integer. The problem is to find $\mathbf{e} \in \mathbb{Z}_2^n$ such that $w(\mathbf{e}) \le t$ and $\mathbf{H}\mathbf{e}^T = \mathbf{s}^T$, $\mathbf{G}\mathbf{e}^T = \mathbf{r}^T$.*

The RSD problem is also proved to be NP-complete since the SD problem can be trivially reduced to the GSD problem [10].

Lemma 2. *Let n, $k' \in \mathbb{Z}$, and $k' \le n/4$. Let $\mathbf{H}$, $\mathbf{G} \in \{0,1\}^{(n-k)\times n}$ be two random matrices in $\mathbb{Z}_2^{k'\times n}$ and $\mathbf{s}, \mathbf{r}$ be two random vectors in $\mathbb{Z}_2^{k'}$. The probability that one can find $\mathbf{e}$ that satisfies (HTML translation failed) and $\mathbf{G}\mathbf{e}^T = \mathbf{r}^T$ is negligible.*

Definition 1 (see [32]). *A regular word is a vector of length n and weight w, and it has exactly one nonzero position in each w intervals $[(i-1)n/w; in/w]_{i=1,\ldots,w}$. Furthermore, if the weight of each interval is two or zero, the word is called 2-regular. A 2-regular word is the sum of two regular words.*

Problem 3 (2-regular null syndrome decoding (2-RNSD) problem). *Let n, k, c, $m \in \mathbb{Z}$, and $m = 2^c \cdot k/c$. Let $\mathbf{B} \in \mathbb{Z}_2^{n\times m}$ be a randomly matrix. The problem is to find a nonzero 2-regular word $\mathbf{z}$ such that $\mathbf{B} \cdot \mathbf{z} = 0$.*

This problem turns out to be NP hard in the worst case [32].

Definition 2 (see [33]). *For any probabilistic polynomial time adversary $\mathcal{A}$, a collision-resistant hash function h satisfies*

$$\Pr[(\mathbf{x}, \mathbf{x}') \leftarrow \mathcal{A}(1^\lambda, h): \mathbf{x} \ne \mathbf{x}', \quad h(\mathbf{x}) = h(\mathbf{x}')] \le \rho(\nu),$$

where $\rho(\nu)$ is a negligible function.

Definition 3 (see [14]). *A noninteractive protocol $\prod = (\mathbf{Setup}, \mathcal{P}, \mathcal{V})$ for a relation R is zero-knowledge, if there*

exists a pair of PPT algorithms called simulator (S_O, S_P) *s.t. for every PPT adversary* $\mathcal{A}$*, we have that*

$$\begin{aligned}&|\Pr\left[b = 1:\ pp \leftarrow \mathbf{Setup}\left(1^\lambda\right),\quad b \leftarrow \mathcal{A}^{\mathcal{O}_1(pp,\cdot,\cdot)}(pp)\right],\\&\quad - \Pr\left[b = 1:\ (pp,\zeta) \leftarrow S_O\left(1^\lambda\right), b \leftarrow \mathcal{A}^{\mathcal{O}_2(pp,\zeta,\cdot)}(pp)\right]| \quad (1)\\&\leq negl(\lambda),\end{aligned}$$

where $\mathcal{O}_1$ and $\mathcal{O}_2$ first validate that the input $(x, w) \in R$, else return $\perp$; otherwise, $\mathcal{O}_1$ outputs $\pi \leftarrow \mathcal{P}_i$, and $\mathcal{O}_2$ outputs $\pi \leftarrow S_P$.

2.3. Merkle-Tree-Based Accumulator. In this section, we first introduce a Merkle-tree-based accumulator scheme [29] which is a building block of our traceable ring scheme. The accumulator is a one-way membership function that takes a set R as input and outputs a constant size value $\mathbf{u}$. At the same time, a value $\mathbf{d} \in R$ has a short witness $\mathbf{w}$, which makes the verifier believe that $\mathbf{d}$ was accumulated to $\mathbf{u}$. The accumulator based on Merkle-tree structure is efficient and is also based on the following code-based hash function $\mathcal{H}$.

Definition 4 (see [29]). *Let* $m = 2 \cdot 2^c \cdot n/c$, RE: $\{0,1\}^n \longrightarrow \{0,1\}^{2^c \cdot n/c}$ *be an encoding function that maps* $\mathbf{x}$ *to* (HTML translation failed). *Consider a random matrix* $\mathbf{B} = [\mathbf{B}_0|\mathbf{B}_1]$*, where* $\mathbf{B}_0, \quad \mathbf{B}_1 \in \mathbb{Z}_2^{n\times m/2}$*. The hash function* $\mathcal{H} = \{h_{\mathbf{B}}|\mathbf{B} \in \mathbb{Z}_2^{n\times m}\}$ *mapping* $\{0,1\}^n \times \{0,1\}^n$ to $\{0,1\}^n$ *is defined as*

$$h_{\mathbf{B}}(\mathbf{u}_0, \mathbf{u}_1) = \mathbf{B}_0 \cdot RE(\mathbf{u}_0) \oplus \mathbf{B}_1 \cdot RE(\mathbf{u}_1). \quad (2)$$

Lemma 3 (see [29]). *The above function family* $\mathcal{H}$ *is collision-resistant if the 2-RNSD problem is hard.*

The accumulator scheme consists of four algorithms:

(1) $\mathbf{Setup}(\lambda)$*: the input is public parameters* (pp)*. The output is a key* $\mathbf{B}$ *for the hash function.*

(2) $\mathbf{Accu}_{\mathbf{B}}(R = \{\mathbf{d}_0, \ldots, \mathbf{d}_{N-1}\} \subseteq (\{0,1\}^n)^N)$*: the input is all the elements in R that treats each element as a leaf node of the Merkle-tree. The output is the root note* $\mathbf{u}$*, which is also called the accumulated value, accumulating by R.*

(3) $\mathbf{WitGen}_{\mathbf{B}}(R, \mathbf{d})$*: the input is* $\mathbf{d}$*. If* $\mathbf{d} \in R$*, the output is the witness* w *for* $\mathbf{d}$*. Otherwise, the output is* $\perp$*.*

(4) $\mathbf{Verify}_{\mathbf{B}}(\mathbf{u}, \mathbf{d}, w)$*: the inputs are* $\mathbf{u}, \mathbf{d}$*, and* w*. In order to obtain the accumulate value* $\mathbf{u}$*, the verifier checks whether* w *is the valid hash path of* $\mathbf{u}$*.*

2.4. Traceable Ring Signatures. In this section, we give the definition and security model of traceable ring signatures. For simplicity of discussion, we denote $\overline{pk} = (\mathrm{pk}_1, \ldots, \mathrm{pk}_L)$, where pk_i is the public key of each user in the ring. Let issue be a string that represents the target of the signature (for example, a transaction or an election).

2.4.1. Syntax. A TRS scheme contains four polynomial time algorithms defined as follows.

(i) $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathbf{KeyGen}(1^\lambda)$: the input is the security parameter λ. **KeyGen** generates public and secret parameters and outputs the pair of public and secret key $(\mathrm{pk}, \mathrm{sk})$.

(ii) $\sigma \leftarrow \mathbf{Sign}(\mathrm{sk}_i, T, M)$: the inputs are the secret sk_i of the user $\mathcal{P}_i$, a tag $T := (\overline{pk}, \text{issue})$, and a message $M \in \{0,1\}^*$. The output is a signature σ on message M with the tag T. The $\overline{pk}$ contains all the members in the ring and pk_i should be in $\overline{pk}$.

(iii) $b \leftarrow \mathbf{Ver}(T, M, \sigma)$: the inputs are the tag $T = (\overline{pk}, \text{issue})$, the signature σ, and the message M. The output is $b = 1$ if accepting the signature or $b = 0$ if not accepting it.

(iv) $s \leftarrow \mathbf{Trace}(T, M_1, M_2, \sigma_1, \sigma_2)$: the inputs are the tag T and two message/signature pairs (M_1, σ_1), (M_2, σ_2) that correspond to private keys sk_i and sk_j, respectively. If $\mathbf{Ver}(T, \sigma_1, M_1) = 1$ and $\mathbf{Ver}(T, \sigma_2, M_2) = 1$, the output is s, that is either equal to linked, accept, or pk_i:

$$\mathrm{Trace}(T, M_1, M_2, \sigma_1, \sigma_2) := \begin{cases}\text{accept}, & \text{if } i \neq j,\\ \text{linked}, & \text{else if } M_1 = M_2,\\ \mathrm{pk}_i, & \text{otherwise}\,(M_1 \neq M_2).\end{cases} \quad (3)$$

The correctness conditions of TRS are completeness and public traceability. The definitions are given as follows:

Definition 5 (completeness). *Let* $i \in [L]$ *and* $T := (\overline{pk}, \text{issue})$ *for some* issues*. If for all* $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathbf{KeyGen}(1^\lambda)$, $\sigma \leftarrow \mathbf{Sign}(\mathrm{sk}_i, T, M)$ *and all* M*, it holds that* $\mathbf{Ver}(T, M, \sigma) = 1$.

Definition 6 (public traceability). *A TRS satisfies public traceable if the following conditions are satisfied: for all* M_1, M_2, issue*, for* $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathbf{KeyGen}(1^\lambda)$, $\sigma_1 \leftarrow \mathbf{Sign}(\mathrm{sk}_i, T, M_1)$*, and* $\sigma_2 \leftarrow \mathbf{Sign}(\mathrm{sk}_j, T, M_2)$*, it holds with an overwhelming probability of* equation (1).

2.4.2. Security Definitions. We use the security model in [5]. The security requirements of traceable ring signatures include the following three properties: tag-linkability, anonymity, and exculpability. The requirement of unforgeability (as defined in ordinary ring signatures) is not essential because the signature is unforgeable if a TRS satisfies both tag-linkability and exculpability [5].

Suppose $\mathcal{A}$ is a probabilistic polynomial time (PPT) adversary and the security parameter is λ. Let N be the number of members in the ring; $T = (\overline{pk}, \text{issue})$ be the tag. By $\mathrm{negl}(n, R)$, we denote a function which is negligible on the parameters n and R.

(1) Tag-Linkability. Take the security parameter λ as input, output T, and $N + 1$ valid pairs of message/signature. The adversary can get N pairs of message/signature by accessing N pairs of public and secret keys. The adversary's advantage over the scheme is $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{tagLink}}$:

$$\mathrm{Adv}_{\mathscr{A}}^{\mathrm{tagLink}}(\lambda, N) \coloneqq \Pr\left[\mathrm{Expt}^{\mathscr{A}}\right], \tag{4}$$

where $\mathrm{Expt}^{\mathscr{A}}$ is

(1) $(T, (M_1, \sigma_1), \ldots, (M_N, \sigma_N)) \leftarrow \mathscr{A}(1^\lambda)$

(2) If all $i \in 1, \ldots, N+1$, $\mathbf{Ver}(T, M_i, \sigma_i) = 1$ and $i, j \in \{1, \ldots, N+1\}$, $i \neq j$, $\mathbf{Trace}(T, M_i, M_j, \sigma_i, \sigma_j) = \mathrm{accept}$.

If for all the PPT adversaries $\mathscr{A}$, $\mathrm{Adv}_{\mathscr{A}}^{\mathrm{tagLink}}(\lambda, N) \leq \mathrm{negl}(\lambda, N)$, the scheme satisfies tag-linkability.

(2) Anonymity. Let $\mathscr{A}$ be a PPT adversary, $(\mathrm{pk}_0, \mathrm{sk}_0)$, $(\mathrm{pk}_1, \mathrm{sk}_1)$ are two public/secret key pairs generated by $\mathbf{KeyGen}(1^\lambda)$. Consider the following game:

(1) $(\mathrm{pk}_i, \mathrm{sk}_i) \leftarrow \mathbf{KeyGen}(1^\lambda)$, $i \in \{0, 1\}$

(2) $b \xleftarrow{\$}$

(3) $b' \leftarrow \mathscr{A}^{\mathbf{Sign}(\mathrm{sk}_0, \cdot), \mathbf{Sign}(\mathrm{sk}_1, \cdot), \mathbf{Sign}(\mathrm{sk}_b, \cdot)}(\mathrm{pk}_0, \mathrm{pk}_1)$

(4) If $b = b'$, output 1; otherwise, output 0.

Let $\mathbf{Sign}(\mathrm{sk}_b, \cdot)$ be a signing oracle. $\mathscr{A}$ cannot ask queries to $\mathbf{Sign}(\mathrm{sk}_b, \cdot)$ with different tags nor can $\mathscr{A}$ ask queries with the same tag to both $\mathbf{Sign}(\mathrm{sk}_b, \cdot)$ and $\mathbf{Sign}(\mathrm{sk}_0, \cdot)$ or $\mathbf{Sign}(\mathrm{sk}_b, \cdot)$ and $\mathbf{Sign}(\mathrm{sk}_1, \cdot)$. If the output of this game is 1, the adversary wins the game. The advantage that $\mathscr{A}$ wins the game is

$$\mathrm{Adv}_{\mathscr{A}}^{\mathrm{anon}}(\lambda, N) \coloneqq \Pr[b = b'] - \frac{1}{2}. \tag{5}$$

If $\mathrm{Adv}_{\mathscr{A}}^{\mathrm{anon}}(\lambda, N) \leq \mathrm{negl}(\lambda, N)$, the scheme is anonymous.

(3) Exculpability. This requirement is presented to ensure the adversary $\mathscr{A}$ cannot construct two valid pairs of message/signature without knowing the secret key sk_i of the user $\mathscr{P}_i$. The game is described as follows:

(1) $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathbf{KeyGen}(1^\lambda)$

(2) $(T, M_1, \sigma_1), (T, M_2, \sigma_2) \leftarrow \mathscr{A}^{\mathbf{Sign}(\mathrm{sk}, \cdot)}(\mathrm{pk})$

(3) $a \leftarrow \mathrm{Trace}(T, M_1, \sigma_1, M_2, \sigma_2)$

(4) output a.

In this game, $\mathbf{Ver}(T, M_1, \sigma_1) = 1$ and $\mathbf{Ver}(T, M_2, \sigma_2) = 1$. For the message/signature pairs that $\mathscr{A}$ accesses the signing oracle $\mathbf{Sign}(\mathrm{sk}, \cdot)$ cannot link to at least one of the σ_1 or σ_2. This means that there is at least one message in M_1 and M_2 that has not been queried in the $\mathbf{Sign}(\mathrm{sk}, \cdot)$. The advantage that $\mathscr{A}$ wins the following game is

$$\mathrm{Adv}_{\mathscr{A}}^{\mathrm{excul}}(\lambda, N) = \Pr[a = \mathrm{pk}] \tag{6}$$

If $\mathrm{Adv}_{\mathscr{A}}^{\mathrm{excul}}(\lambda, N) \leq \mathrm{negl}(\lambda, N)$, the TRS satisfies exculpability.

3. A Code-Based TRS Scheme

3.1. A Proof of Knowledge Protocol. We use a so-called GStern's protocol in [10] to construct our TRS scheme. Given $\mathbf{H}$, $\mathbf{G}$ and $\mathbf{s}$, $\mathbf{r}$ in the GSD problem, the prover $\mathscr{P}_i$ wants the verifier $\mathscr{V}$ to confirm that he has a small weight vector $\mathbf{e}$ such that $\mathbf{He}^T = \mathbf{s}^T$ and $\mathbf{Ge}^T = \mathbf{r}^T$. In other words, the protocol is a proof of knowledge protocol for the GSD problem. To be self-contained, we give the detailed description in **Algorithm 1**, in which h denotes a cryptographic hash function.

The GStern's protocol satisfies the following three natures: completeness, special soundness, and honest-verifier zero-knowledge (HVZK) [10]. To construct our TRS scheme, we apply the code-based Merkle-tree accumulator [29] to GStern's protocol. The statistical zero-knowledge argument of the accumulator allows the prover $\mathscr{P}_i$ to convince the verifier $\mathscr{V}$, under zero-knowledge conditions, that $\mathscr{P}$ knows a value correctly accumulated into the code-based Merkle-tree root described above. Let the uniformly random matrix $\mathbf{B} \in \mathbb{Z}_2^{n \times m}$ and the accumulated value $\mathbf{u}$ be the input. The goal of $\mathscr{P}_i$ is to convince $\mathscr{V}$ that he has a value $\mathbf{d}$ and a valid witness w. The relationship with the accumulator is $R_{acc} = \{((\mathbf{B}, \mathbf{u}) \in \mathbb{Z}_2^{n \times m} \times \{0,1\}^n, \quad d \in \{0,1\}^n, w \in \{0,1\}^l \times (\{0,1\}^n)^l)$: $\mathrm{Verify}_{\mathbf{B}}(\mathbf{u}, \mathbf{d}, w)\} = 1\}$. The authors of [29] gave specific techniques how to reduce the relationship R_{acc} to the abstract relationship $R_{\mathrm{abstract}} = \{(\mathbf{M}, \mathbf{v}), \quad \mathbf{w} \in \mathbb{Z}_2^{K \times L} \times \mathbb{Z}_2^K \times \mathrm{VALID}: \mathbf{M} \cdot \mathbf{w} = \mathbf{v}\}$, where $\mathbf{M}, \mathbf{w}, \mathbf{v}$ are obtained by doing some transforms to $\mathbf{B}, \mathbf{u}, \mathbf{d}$. In other words, if we want to construct relation R_{acc}, we only need to construct relation R_{abstract}. We summarize the above method into the new protocol Acc-GStern's protocol described in Algorithm 2. The COM denotes a commitment scheme and $\mathcal{S}$ is a finite set, where each $\phi \in \mathcal{S}$ is associated with a permutation Γ_ϕ of L elements. In addition, VALID is a subset of $\{0,1\}^L$.

Lemma 4. *The protocol presented in **Algorithm 2** is complete, special sound, and HVZK.*

Proof. Our new protocol is a combination of GStern's protocol and the accumulator protocol. If an honest prover follows the protocol, then he always gets accepted by the verifier. Thus, the protocol has perfect completeness. If there is a simulator who extracts a valid witness from two valid transcripts $(\mathrm{com}, \mathrm{ch}, \mathrm{resp})$ and $(\mathrm{com}, \mathrm{ch}', \mathrm{resp}')$ of Acc-GStern's protocol with $\mathrm{ch} \neq \mathrm{ch}'$, where com, $\mathrm{ch}(\mathrm{ch}')$ and $\mathrm{resp}(\mathrm{resp}')$ are commitments, challenges, and responses, respectively, he can extract a valid witness. We consider the following cases:

(1) When $\mathrm{ch} = 0$ and $\mathrm{ch}' = 1$, the simulator can extract $\mathbf{e}$ from $\mathbf{y}$ and $\mathbf{y} + \mathbf{e}$. For $\phi_2 = \phi$, the simulator can extract $\mathbf{w}$ from $\Gamma_\phi(\mathbf{w})$.

(2) When $\mathrm{ch} = 0$ and $\mathrm{ch}' = 2$, the simulator can extract $\mathbf{e}$ from δ and $\delta(\mathbf{e})$. For $\phi_3 = \phi$, the simulator can extract $\mathbf{w}$ from $\Gamma_\phi(\mathbf{w})$ and ϕ.

(3) When $\mathrm{ch} = 1$ and $\mathrm{ch}' = 2$, the simulator can extract $\mathbf{e}$ from δ and $\delta(\mathbf{e})$ and the simulator can extract $\mathbf{w}$ from $\mathbf{r}_w$ and $\mathbf{w} \oplus \mathbf{r}_w$.

Finally, we prove HVZK of the protocol. When $b = 0$, the simulator easily reveals $\mathbf{y}$, $\mathbf{y} + \mathbf{e}$, $\Gamma_\phi(\mathbf{w})$, and $\Gamma_\phi(\mathbf{r}_w)$. When $b = 1$, the simulator gets $\mathbf{x}$, where $\mathbf{Hx}^T = \mathbf{s}^T$ and reveals a vector $\mathbf{y}$, where $\mathbf{M} \cdot (\mathbf{y} \oplus \mathbf{r}_w) \oplus \mathbf{v} = \mathbf{M} \cdot \mathbf{r}_w$. When $b = 2$, the

(1) **Parameters:** n, k, t
(2) **Private information:** $\mathbf{e} \in \mathbb{Z}_2^n$ and $w(\mathbf{e}) = t$.
(3) **Public information:** $\mathbf{H}, \mathbf{G}, \mathbf{s}, \mathbf{r}$, where $\mathbf{He}^T = \mathbf{s}^T$ and $\mathbf{Ge}^T = \mathbf{r}^T$.
(4) **The prover $\mathscr{P}_i$:**
(i) chooses $y^{\$} \longleftarrow$ and a permutation δ.
(ii) sets $\mathbf{c}_1 = h(\delta, \mathbf{Hy}^T, \mathbf{Gy}^T)$, $\mathbf{c}_2 = h(\delta(\mathbf{y}))$,
(iii) and $\mathbf{c}_3 = h(\delta(\mathbf{y} + \mathbf{e}))$.
(iv) sends $\mathbf{c}_1$, $\mathbf{c}_2$, and $\mathbf{c}_3$.
(5) **The verifier $\mathscr{V}$:** - sends $b^{\$} \longleftarrow$.
(6) **The prover $\mathscr{P}_i$:**
(i) if $b = 0$, sets $f \coloneqq \{\mathbf{y}, \delta\}$.
(ii) if $b = 1$, sets $f \coloneqq \{\mathbf{y} + \mathbf{e}, \delta\}$.
(iii) if $b = 2$, sets $f \coloneqq \{\delta(\mathbf{y}), \delta(\mathbf{e})\}$.
(iv) sends f.
(7) **The verifier $\mathscr{V}$:**
(i) if $b = 0$, accepts if $h(\delta, \mathbf{Hy}^T, \mathbf{Gy}^T) = c_1$ and $h(\delta(\mathbf{y})) = c_2$.
(ii) if $b = 1$, accepts if $h(\delta, \mathbf{H}(\mathbf{y} + \mathbf{e})^T + \mathbf{s}^T, \mathbf{G}(\mathbf{y} + \mathbf{e})^T + \mathbf{r}^T) = c_1$ and $h(\delta(\mathbf{y} + \mathbf{e})) = c_3$.
(iv) if $b = 2$, accepts if $h(\delta(\mathbf{y})) = c_2$, $h(\delta(\mathbf{y}) + \delta(\mathbf{e})) = c_3$, and $w(\delta(\mathbf{e})) = t$.

Algorithm 1: GStern's protocol.

(1) **Parameters:** n, k, t, K, $L \in \mathbb{N}$, $L \geq K$.
(2) **Private information:** $\mathbf{e}, \mathbf{w}$.
(3) **Public information:**
(i) $\mathbf{H}, \mathbf{s}, \mathbf{G}, \mathbf{r}$, where $\mathbf{He}^T = \mathbf{s}^T$ and $\mathbf{Ge}^T = \mathbf{r}^T$.
(ii) $\mathbf{M}, \mathbf{v}$, where $\mathbf{M} \cdot \mathbf{w} = \mathbf{v}$.
(4) **The prover $\mathscr{P}$:**- chooses $\mathbf{y}^{\$} \longleftarrow$, $\mathbf{r}_w{}^{\$} \longleftarrow$, a permutation δ, a permutation $\phi^{\$} \longleftarrow$, randomness ρ_1, ρ_2, ρ_3 for COM.
(i) sets $\mathbf{c}_1 = h(\delta, \mathbf{Hy}^T, \mathbf{Gy}^T)$, $\mathbf{c}_2 = h(\delta(\mathbf{y}))$, and $\mathbf{c}_3 = h(\delta(\mathbf{y} + \mathbf{e}))$.
(ii) sets $C_1 = \mathrm{COM}(\phi, \mathbf{M} \cdot \mathbf{r}_w; \rho_1)$, $C_2 = \mathrm{COM}(\Gamma_\phi(\mathbf{r}_w); \rho_2)$. and $C_3 = \mathrm{COM}(\Gamma_\phi(\mathbf{w} \oplus \mathbf{r}_w); \rho_3)$.
(iii) sends $\mathbf{c}_1$, $\mathbf{c}_2$, $\mathbf{c}_3$, $\mathbf{C}_1$, $\mathbf{C}_2$, and $\mathbf{C}_3$.
(5) **The verifier $\mathscr{V}$:**
(i) sends $b \overset{\$}{\leftarrow}$.
(6) **The prover $\mathscr{P}$:**
(i) if $b = 0$, computes $\mathbf{t}_w = \Gamma_\phi(\mathbf{w})$, $\mathbf{t}_r = \Gamma_\phi(\mathbf{r}_w)$, sets $f \coloneqq \{\mathbf{y}, \delta, \mathbf{t}_w, \mathbf{t}_r, \rho_2, \rho_3\}$.
(ii) if $b = 1$, computes $\mathbf{w}_2 = \mathbf{w} \oplus \mathbf{r}_w$, $\phi_2 = \phi$, sets $f \coloneqq \{\mathbf{y} + \mathbf{e}, \delta, \phi_2, \mathbf{w}_2, \rho_1, \rho_3\}$.
(iii) if $b = 2$, computes $\phi_3 = \phi$, $\mathbf{w}_3 = \mathbf{r}_w$, sets $f \coloneqq \{\delta(\mathbf{y}), \delta(\mathbf{e}), \phi_3, \mathbf{w}_3, \rho_1, \rho_2\}$.
(iv) sends f.
(7) **The verifier $\mathscr{V}$:**
(i) if $b = 0$, accepts if $h(\delta, \mathbf{Hy}^T, \mathbf{Gy}^T) = c_1, h(\delta(\mathbf{y})) = c_2, \mathbf{t}_w \in \mathrm{VALID}, C_2 = \mathrm{COM}(\mathbf{t}_r; \rho_2), C_3 = \mathrm{COM}(\mathbf{t}_w \oplus \mathbf{t}_r; \rho_3)$.
(ii) if $b = 1$, accepts if $h(\delta, \mathbf{H}(\mathbf{y} + \mathbf{e})^T + \mathbf{s}^T, \mathbf{G}(\mathbf{y} + \mathbf{e})^T + \mathbf{r}^T) = c_1, h(\delta(\mathbf{y} + \mathbf{e})) = c_3, C_1 = \mathrm{COM}(\phi_2, \mathbf{M} \cdot \mathbf{w}_2 \oplus \mathbf{v}; \rho_1)andC_3 = \mathrm{COM}(\Gamma_{\phi_2}(\mathbf{w}_2); \rho_3)$.
(iii) if $b = 2$, accepts if $h(\delta(\mathbf{y})) = c_2, h(\delta(\mathbf{y}) + \delta(\mathbf{e})) = c_3$, $w(\delta(\mathbf{e})) = t, C_1 = \mathrm{COM}(\phi_3, \mathbf{M} \cdot \mathbf{w}_3; \rho_1)andC_2 = \mathrm{COM}(\Gamma_{\phi_3}(\mathbf{w}_3); \rho_2)$.

Algorithm 2: Acc-GStern's protocol.

simulator obtains a vector with weight t and a vector of length L. □

3.2. Description of the Scheme. In this section, we give the description of our new TRS scheme in **Algorithm 3**. Generally speaking, the scheme is constructed by using the noninteractive protocol $\prod \coloneqq (\mathbf{Setup}, \mathscr{P}, \mathscr{V})$ which is obtained by combining Acc-GStern's protocol with Fiat-Shamir transform [30].

First, we use the public information T and a collision-resistant hash function h_1 to construct the matrix $\tilde{H}$. Then, we construct a set of random syndromes $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_L$, one of which, i.e., some $\mathbf{r}_i$, is a vector associated with the secret key of the actual signer. When the same signer signs two different messages with the same tag, the vector $\mathbf{r}_i$ will be the same and so we can identity the signer. We also use a collision-resistant hash function g to generate the other $\mathbf{r}_j$, where $j \neq i$, to prevent the signer from cheating.

Let $\mathscr{P}_1, \mathscr{P}_2, \ldots, \mathscr{P}_L$ be the members of the ring, and $\overline{pk} = (\mathrm{pk}_1, \mathrm{pk}_2, \ldots, \mathrm{pk}_L)$ be the public keys of the members. The tag is $T = (\mathrm{issue}, \overline{pk})$. Let issue be a string of signed targets (for example, an election or a transaction). Let $h_1 \colon \mathbb{Z}_2^* \longrightarrow \mathbb{Z}_2^{(n-k)\times n}$, $g \colon \mathbb{Z}_2^* \longrightarrow \mathbb{Z}_n^{(n-k)}$ and h_2 be three

(1) **Parameters:** $n, t, c, r, k' \in \mathbb{N}$, $m = 2 \cdot 2^c \cdot k'/c$, $k = 3n/4$, $\mathbf{H} \xleftarrow{\$}$.
(2) **KeyGen:** For each user $\mathcal{P}_i$, $i \in [L]$
(i) randomly chooses $\mathbf{e}_i \in \{0,1\}^n$ such that $w(\mathbf{e}_i) = t$.
(ii) computes $\mathbf{s}_i^T = \mathbf{H}\mathbf{e}_i^T$.
(iii) theprivatekey: $\mathbf{e}_i$.
(iv) thepublickey: $\mathbf{H}, \mathbf{B}, \mathbf{s}_i$.
(3) **Sign:** To generate a signature on message (HTML translation failed), the user $\mathcal{P}_i$
(i) computes $\tilde{H} = h_1(T)$ and $\tilde{H}\mathbf{e}_i^T = \mathbf{r}_i^T$.
(ii) sets $A_0 = \mathbf{r}_i + g(M) + \cdots + g^i(M)$.
(iii) computes $\mathbf{r}_j = A_0 + g(M) + \cdots + g^j(M)$, for $j \in [L]$, $j \neq i$.
(iv) computes $\mathbf{d}_i = h_2(\mathbf{s}_i, \mathbf{r}_i)$, for all $i \in [L]$, and defines $R = (\mathbf{d}_i)_{i \in [L]}$.
(v) computes $\mathbf{u} = \mathrm{Accu}_{\mathbf{B}}(R)$ and $\mathbf{w}_i = \mathrm{WitGen}_{\mathbf{B}}(R, \mathbf{d}_i)$.
(vi) let $X := (\mathbf{H}, \tilde{H}, \mathbf{s}_i, \mathbf{r}_i, \mathbf{B}, \mathbf{d}_i, \mathbf{u})$ be the public input of the protocol $\prod$, and let $W := (\mathbf{w}_i, \mathbf{e}_i)$ be the secret inputs. Then, the user runs $\mathcal{P}$ to generate a noninteractive proof ν.
(vii) outputs the signature $\sigma = (\nu, A_0)$.
(4) **Verify:** To verify the signature σ on message M, the verifier
(i) computes $\mathbf{r}_i = A_0 + g(M) + g^2(M) \ldots + g^i(M)$, for all $i \in [L]$.
(ii) computes $\mathbf{d}_i = h_2(\mathbf{s}_i, \mathbf{r}_i)$, for all $i \in [L]$, and defines $R = (\mathbf{d}_i)_{[L]}$.
(iii) computes $\mathbf{u} = \mathrm{Accu}_{\mathbf{B}}(R)$.
(iv) let $X := (\mathbf{H}, \tilde{H}, \mathbf{s}_i, \mathbf{r}_i, \mathbf{B}, \mathbf{d}_i, \mathbf{u})$ be the public input of the protocol $\prod$. Then, the verifier runs $\mathcal{V}$ to get μ.
(v) if $\mu = 1$, outputs 1. Otherwise, outputs 0.
(5) **Trace:** When the verifier is given two signatures (T, M, σ) and (T, M', σ'), where $\sigma = (\nu, A_0)$ and $\sigma' = (\nu', A_0')$, the verifier
(i) if $\mathbf{Verify}(T, M, \sigma) = 1$ and $\mathbf{Verify}(T, M', \sigma') = 1$, continue.
(ii) computes $\mathbf{r}_j = A_0 + g(M) + \cdots + g^j(M)$ and $\mathbf{r}_j' = A_0' + g(M') + \cdots + g^j(M')$ for all $j \in [L]$.
(iii) if $\mathbf{r}_j = \mathbf{r}_j'$ for all $j \in [L]$, outputs linked.
(iv) else if only one index $j \in [L]$ makes $\mathbf{r}_j = \mathbf{r}_j'$, outputs pk_i.
(v) otherwise, outputs accept.

Algorithm 3: Our TRS scheme.

different collision-resistant hash functions. The hash function h_2 is the special function used in the accumulator as we defined in Definition 4. In addition, $g^i(x)$ is the function g applied i times on input x.

4. Security and Analysis

In this section, we present the analysis of our TRS scheme from two aspects: correctness analysis and security analysis.

4.1. Correctness Analysis. If a TRS scheme satisfies completeness and public traceability which are given in Definition 5 and Definition 6, respectively, we say that the TRS scheme is correct.

4.1.1. Completeness. The completeness of our TRS is easily verified. Since h and g are collision-resistant hash functions, and A_0 is generated by the tag and the secret key $\mathbf{e}_i$, the signer can always generate all $\mathbf{r}_i$ and $\mathbf{d}_i$ for all $i \in [L]$, just like in the signing algorithm. The verifier can recover all $\mathbf{r}_i$ from A_0 and also recover all $\mathbf{d}_i$. Due to the completeness of the underlying protocol, the output of the verification algorithm is always 1 when the input of the verification algorithm is the honest signature ν.

4.1.2. Public Traceability. Next, we give the proof of public traceability from three cases:

(i) Case 1. Suppose that $M = M'$, $i = i'$. Therefore, we have $\mathbf{r}_i^T = h_1(T)\mathbf{e}_i^T = h_1(T)\mathbf{e}_i'^T = \mathbf{r}_i'^T$, and we get $\mathbf{r}_j = \mathbf{r}_j'$ for all $j \in [L]$. In this case, the output of **Trace** is always linked.

(ii) Case 2. Suppose that $M \neq M'$, $i = i'$. So, we have $\mathbf{r}_i^T = h_1(T)\mathbf{e}_i^T = h_1(T)\mathbf{e}_i'^T = \mathbf{r}_i'^T$. However, due to the collision resistance of the hash function g, we have $\mathbf{r}_j \neq \mathbf{r}_j'$, for $j \in [L]$, $j \neq i$ with overwhelming probability. Therefore, only $\mathbf{r}_i = \mathbf{r}_i'$ in the two sequences $(\mathbf{r}_j)_{j \in [L]}$ and $(\mathbf{r}_j')_{j \in [L]}$. In this case, the output of **Trace** is always pk_i.

(iii) Case 3. Suppose that $i \neq i'$ and $M = M'$. Thus, we have $\mathbf{r}_i^T = h_1(T)\mathbf{e}_i^T$, $h_1(T)\mathbf{e}_i'^T = \mathbf{r}_i'^T$, and $\mathbf{r}_i \neq \mathbf{r}_i'$. Then, due to the hash function, we have $\mathbf{r}_j \neq \mathbf{r}_j'$ for all $j \in [L]$ with overwhelming probability. If $i \neq i'$ and $M \neq M'$, we obtain $\mathbf{r}_i \neq \mathbf{r}_i'$. If the output of the algorithm **Trace** is not *accept*, there must be some $j \neq i$, $j \in [L]$, satisfying $\mathbf{r}_j = \mathbf{r}_j'$. However, we have $\mathbf{r}_j = A_0 + g(M) + \cdots + g^j(M)$ and $\mathbf{r}_j' = A_0' + g(M') + \cdots + g^j(M')$. Due to the collision resistance of hash functions and the difficulty of the GSD problem, the probability of the existence of $\mathbf{r}_j$ and $\mathbf{r}_j'$ is negligible. Therefore, the output of **Trace** is accept with overwhelming probability.

4.2. Security Analysis. We use the security definition of TRS in [5], which formalized security requirements called anonymity, tag-linkability, and exculpability.

Theorem 1. *Our scheme is secure in the random oracle model, i.e., satisfying anonymity, tag-linkability, and exculpability.*

We prove Theorem 1 from the following three aspects: proof for anonymity, proof for tag-linkability, and proof for exculpability.

4.2.1. Proof for Anonymity. To show that our scheme satisfies anonymity, we define a PPT adversary $\mathscr{A}$, a challenger $\mathscr{C}$, and a series of games G_i, $i = 0, 1, 2, 3, 4$. There are three signing oracles: $\mathbf{Sign}_{sk_0}, \mathbf{Sign}_{sk_1}, \mathbf{Sign}_b$. The advantage of the adversary $\mathscr{A}$ in G_i is denoted by $\mathrm{Adv}^{\mathrm{anon}}_{\mathscr{A},G_i}$.

G_0: this game is just like the game defined in (2), in which $b = 0$. The adversary $\mathscr{A}$ can access three oracles. The challenger $\mathscr{C}$ honestly runs the $\mathbf{Sign}_{\mathrm{sk}_j}$ oracle with corresponding secret keys to reply to $\mathscr{A}$'s queries to these there oracles.

G_1: let (S_P, S_O) be the simulators of the protocol $\prod$. The challenger $\mathscr{C}$ runs S_O to simulate the public parameters, instead of running **Setup** of the protocol $\prod$. When the adversary $\mathscr{A}$ makes queries to $\mathbf{Sign}_{\mathrm{sk}_b}$, the challenger $\mathscr{C}$ runs S_P to answer the query.

Due to Definition 3, we have

$$\mathrm{Adv}^{\mathrm{anon}}_{\mathscr{A},G_0}(\lambda) \approx \mathrm{Adv}^{\mathrm{anon}}_{\mathscr{A},G_1}(\lambda). \tag{7}$$

G_2: the difference between G_2 and G_1 is that the challenger $\mathscr{C}$ creates an empty table δ. We assume that the sequence $(\mathbf{r}_1^{(i)}, \mathbf{r}_2^{(i)}, \ldots, \mathbf{r}_L^{(i)})$ is computed by the user $\mathscr{P}_i$, and i is the position of pk_i in T. The adversary $\mathscr{A}$ gets access to $\mathbf{Sign}_{\mathrm{sk}_b}$ and the query is (T, M).The challenger $\mathscr{C}$ does not use $h_1(T)\mathbf{e}_b^T$ to get $\mathbf{r}_b^{(b)}$. The challenger $\mathscr{C}$ checks whether there is a tuple $(T, M, \mathbf{r}_*^{(b)})$ in δ, where $\mathbf{r}_*^{(b)}$ is the vector in δ together with (T, M). If the tuple exists, then $\mathscr{C}$ uses $\mathbf{r}_*^{(b)}$ and runs the simulator S_P to generate the signature. Otherwise, the challenger $\mathscr{C}$ randomly chooses a vector $\mathbf{r}_b^{(b)} \xleftarrow{\$}$, sets $\mathbf{r}_*^b = \mathbf{r}_b^{(b)}$, and adds $(T, M, \mathbf{r}_b^{(b)})$ to δ. Then, the challenger $\mathscr{C}$ generates the signature using $\mathbf{r}_b^{(b)}$.

Due to the GSD problem, the adversary cannot calculate $\mathbf{e}$ from $\mathbf{r}_b^{(b)}$ and the public parameters. So, $\mathscr{A}$ cannot distinguish G_1 and G_2:

$$\mathrm{Adv}^{\mathrm{anon}}_{\mathscr{A},G_1}(\lambda) \approx \mathrm{Adv}^{\mathrm{anon}}_{\mathscr{A},G_2}(\lambda). \tag{8}$$

G_3: the difference between G_3 and G_2 is that the challenger $\mathscr{C}$ uses $\mathbf{sk}_1$ to generate $\mathbf{r}_j$ and answers the query to $\mathbf{Sign}_{\mathrm{sk}_b}$. G_4: the difference between G_0 and G_4 is the value of b. G_4 defines $b = 1$.

Due to the zero-knowledge property of the underlying protocol and the hardness of the GSD problem, the outputs of all the above games contain nothing about b. Therefore, the adversary can guess the right b with probability 1/2 and the advantage that the adversary $\mathscr{A}$ wins the game is negligible.

4.2.2. Proof for Tag-Linkability. We assume that the PPT adversary is $\mathscr{A}$ and the sequence $(\mathbf{r}_i)_{i\in[L]}$ is contained in the signature which can be reconstructed from (T, M_i, ν_i, A_0). The $(L+1)$ message/signature pairs $(M_1, \sigma_1), \ldots, (M_{L+1}, \sigma_{L+1})$ with tag T are generated by the adversary $\mathscr{A}$. Suppose that $\mathscr{A}$ wins the tag-linkability game. Thus, we have $(a)\mathbf{Verify}(\sigma_i, M_i, T) = 1, \quad \forall i \in [L+1]$; and $(b)\mathbf{Trace}(T, M_i, \sigma_i, M_j, \sigma_j) = \mathrm{accept}, \quad \forall i, j \in [L+1], i \neq j$.

Let I_{real} and I_{S_O} be the sets of all the parameters honestly generated in our construction and all the parameters generated by the simulator S_O, respectively. Due to the zero-knowledge property of the protocol, $\mathscr{A}$ cannot distinguish I_{real} and I_{S_o}. Therefore, there is an extractor that can extract a witness $w = (i, \mathbf{e}_i)$ for each (M_i, σ_i). There is at most only one witness with overwhelming probability because of the hardness of the GSD problem, and correspondingly, $\mathbf{s}_i^T = \mathbf{H}\mathbf{e}_i^T$ and $\mathbf{r}_i^T = h_1(T)\mathbf{e}_i^T, \quad i \in [L+1]$. Since T only contains L public keys, there must exist $\mathbf{s}_i = \mathbf{s}_j, \quad i \neq j$ in the sequence $(\mathrm{pk}_i)_{i\in[L+1]}$. Since all public keys in T are distinct, we have $\mathbf{e}_i = \mathbf{e}_j$. Finally, we get $\mathbf{r}_i^T = h_1(T)\mathbf{e}_i^T = h_1(T)\mathbf{e}_j^T = \mathbf{r}_j^T$, which means that two signatures σ_i and σ_j cannot be accepted by **Trace**. However, this contradicts our previous assumptions.

4.2.3. Proof for Exculpability. Suppose that $\mathscr{A}$ is a PPT adversary. We define the following games $G_i, \quad i = 0, 1$.

G_0: this is the real exculpability game. In this game, the challenger $\mathscr{C}$ runs **KeyGen** to generate $(\mathrm{pk}_j, \mathrm{sk}_j)$, where j is the position of pk_j in T. $\mathscr{C}$ runs $\mathbf{Sign}_{sk_j}$ to reply to $\mathscr{A}$'s queries to $\mathbf{Sign}_{sk_j}$.

G_1: the challenger $\mathscr{C}$ runs the simulator of the protocol $\prod$ to generate the public parameters and randomly chooses $\mathbf{s}_i \xleftarrow{\$}$. $\mathscr{C}$ creates an empty table δ to record each query. The challenger $\mathscr{C}$ receives a query (T, M) to $\mathbf{Sign}_{\mathrm{sk}_j}$ and the secret key sk_j corresponds to the public key pk_j in T, where (HTML translation failed) is the position of pk_j in T. However, $\mathscr{C}$ does not compute $\mathbf{r}_j^{(j)}$ in terms of $h_1(T)\mathbf{e}_j^T$. If $(T, M, \mathbf{r}_*^{(j)})$ is not in the table δ, where $\mathbf{r}_*^{(j)}$ is the vector in δ together with (T, M), he randomly chooses a vector $\mathbf{r}_*^{(j)} \xleftarrow{\$}$ and adds $(T, M, \mathbf{r}_*^{(j)})$ to the table δ and generates the signature with $\mathbf{r}_*^{(j)}$. Otherwise, $\mathscr{C}$ sets $\mathbf{r}_j^{(j)}$ as $\mathbf{r}_*^{(j)}$ and uses the simulator S_P to generate the signature.

Due to the zero-knowledge property of the underlying protocol and the hardness of the GSD problem, the advantage that $\mathscr{A}$ distinguishes G_0 and G_1 is

$$\mathrm{Adv}^{\mathrm{excul}}_{\mathscr{A},G_0}(\lambda) \approx \mathrm{Adv}^{\mathrm{excul}}_{\mathscr{A},G_1}(\lambda). \tag{9}$$

Then, we show that a successful attack is impossible in G_1. If $\mathscr{A}$ can output two valid pairs (T, M, σ) and (T, M', σ'), the two valid pairs can satisfy:

(1) $\mathbf{Verify}(T, M, \sigma) = 1$ and $\mathbf{Verify}(T, M', \sigma') = 1$,

(2) $\mathbf{Trace}(T, M, \sigma, M', \sigma') = pk_i$, where i is the position of pk_i in T.

We consider the following two cases:

(i) Case 1. Suppose that one of those two pairs can be linked with the table δ and the pair is $(T, M', \quad \sigma' = (\nu', A_0'))$. Therefore, there is a pair $(T, M^*, \sigma^* = (\nu^*, A_0))$ in δ that can be linked with $(T, M', \sigma' = (\nu', A_0'))$ where M^* and σ^* are the

Table 1: Our TRS scheme.

Scheme	n	k	t	c	k'	r	L	p	Pk size (bit)	Signature size (bit)	Security
1	2400	2006	58	8	896	509	3	220	3.0×10^7	7.9×10^7	128
2	4150	3307	132	14	1792	1024	3	440	4.2×10^9	1.1×10^{10}	256

vectors in δ in the target pair and (ν^*, A_0) are the vectors corresponding to σ^*. We can get a witness $w = (i', \mathbf{e}_i')$ from the extractor of the protocol $\prod$. In this way, the elements in the pair should satisfy $\mathbf{s}_i'^T = H\mathbf{e}_i'^T$, $\mathbf{r}_i'^T = h_1(T)\mathbf{e}_i'^T$ and $\mathbf{r}_j' = A_0' + g(M') + \cdots + g^j(M')$ for all $j \in [L]$, $j \neq i$, and due to that the two pairs are linked, $\mathbf{s}^* = \mathbf{s}_i'$ and $\mathbf{r}_j^* = \mathbf{r}_j'$, for all $j \in [L]$. However, $\mathbf{s}^*$ is generated by the simulator of the protocol $\prod$ and the vectors $\mathbf{r}_j^*$, $j \in [L]$ are randomly chosen in $\mathbb{Z}_2^{n-k}$ in G_1. Considering the hardness of the GSD problem, the adversary $\mathcal{A}$ cannot generate the valid $\mathbf{e}'$ and $\mathbf{r}'$.

(ii) Case 2. Suppose that neither of the two pairs can be associated with the table δ. So, we have $\mathbf{r}_j = \mathbf{r}_j'$, which means that $A_0 + g(M) + \cdots + g^j(M) = A_0' + g(M') + \cdots + g^j(M')$, for all $j \in [L]$. We can also extract the witness $(i, \mathbf{e}_i)$ and $(i', \mathbf{e}_{i'})$ from ν and ν', respectively. We have $h_1(T)\mathbf{e}_i^T = \mathbf{r}_i^T$ and $h_1(T)\mathbf{e}_{i'}^T = \mathbf{r}_{i'}^T$, which holds that $h_1(T)\mathbf{e}_i^T = h_1(T)\mathbf{e}_{i'}^T$. If $i = i'$, the **Trace** will output $\mathbf{s}_i$. However, $\mathbf{r}_i$ is chosen randomly in $\mathbb{Z}_2^{n-k}$ in G_1, and so $\mathcal{A}$ cannot construct a vector $\mathbf{e}_i$ which satisfies $H\mathbf{e}_i^T = \mathbf{s}_i^T$ and $h_1(T)\mathbf{e}_i^T = \mathbf{r}_i^T$. In this condition, we have $i \neq i'$. Due to the hardness of the GSD problem, the probability that the adversary $\mathcal{A}$ constructs the valid $\mathbf{e}_i, \mathbf{e}_i', \mathbf{r}_i, \mathbf{r}_i'$ is negligible. Therefore, the adversary $\mathcal{A}$ cannot have a successful attack.

5. Efficiency

In this section, we consider the efficiency of our scheme in three aspects: public key sizes, secret key sizes, and signature sizes.

(1) Public key size: the public key in our scheme consists of $(\mathbf{H}, \mathbf{B}) \in \in \mathbb{Z}_2^{(n-k)\times n} \times \mathbb{Z}_2^{r\times m}$ and vectors $\mathbf{s}_i \in \mathbb{Z}_2^{n-k}$, for $i \in [L]$. The public key size of our scheme is $n^2 - k(n+L) + rm + Ln$ bits.

(2) Secret key size: the secret key of each signer $\mathcal{P}_i$ is a vector $\mathbf{e}_i \in \mathbb{Z}_2^n$, and the bit length of $\mathbf{e}_i$ is n.

(3) Signature size: the signature size in our scheme is determined by the proof ν, and the bit length of the signature is $O(\lambda \cdot \log L)$. Specifically, the signature size of our scheme includes the following three aspects: (i) the size of three hash values and three commitments is 6λ bits, where λ is the security parameter. (ii) For each case, $b = 0$, $b = 1$, or $b = 2$, the size of response is $4\,\mathrm{lm} + 4\,\mathrm{nl} - 2n + 2\lambda$ bits, where $l = \log L$. (iii) The bit size of the vector A_0 is $n - k$. The repetition number of Acc-GStern's protocol is defined as p (for example, if the cheating probability of Acc-GStern's protocol is approximately 2^{-128}, $p = 220$). To sum up, the signature size of our scheme is $6\lambda + 4\,\mathrm{lmp} + 4\,\mathrm{nlp} - 2\,\mathrm{np} + 2p\lambda + n - k$ bits. Since $l = \log L$ and L is the ring size, our signature size is logarithmic in the ring size.

According to decoding attacks in [34–36], we set the parameters of our scheme under 128 bit and 256 bit security in Table 1.

We give the implementation of our scheme on an Intel(R) Core(TM) i5-1035G1 CPU @ 2.20 GHz, and we implemented our scheme based on Python. We use the first set of parameters in Table 1 to implement our scheme. The running time of **KeyGen** to generate a pair of (pk, sk) is about 3 ms. However, due to the use of the hash function defined in Definition 4, the total running time for our signature scheme to generate a signature is about 2 minutes.

Since our scheme is an improvement over [10], the signature size in the original scheme is linear in the number of ring size, while the signature size of our scheme is logarithmic in the ring size. To the best of our knowledge, there are no other code-based TRS schemes. However, for some applications, the key and signature sizes of our scheme are still large. Finding new techniques to reduce the sizes of code-based traceable ring signature schemes and improve the efficiency of code-based traceable ring signature schemes are our future research direction.

6. Conclusion

In this paper, we construct a new code-based TRS scheme. The signature size of our scheme is logarithmic in the size of the ring. We then provide the tag-linkability, anonymity, and exculpability of our scheme and so our scheme is secure in the random oracle model under the assumption of the hardness of the SD problem and the 2-RNSD problem.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work of L.P. Wang was supported in part by the National Natural Science Foundation of China (Grant no. 61872355), Mathematical Tianyuan Foundation of National Natural Science Foundation of China (Grant No.12026427), and National Key Research and Development Program of China (No. 2018YFA0704703).

References

[1] E. Fujisaki, "Sub-linear size traceable ring signatures without random oracles," in *Proceedings of the Topics in Cryptology - CT-RSA 2011 - The Cryptographers' Track at the RSA Conference 2011*, pp. 393–415, Springer, San Francisco, CA, USA, 14 February 2011.

[2] C. Hu and D. Li, "Forward-secure traceable ring signature," in *Proceedings of the 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, pp. 200–204, IEEE Computer Society, Qingdao, China, 30 July 2007.

[3] X. Bultel and P. Lafourcade, "k-times full traceable ring signature," in *Proceedings of the 2016 11th International Conference on Availability, Reliability and Security, ARES 2016*, pp. 39–48, IEEE, Salzburg, Austria, 31 August 2016.

[4] S. S. M. Chow, J. K. Liu, and D. S. Wong, "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, 24 February 2008.

[5] E. Fujisaki and K. Suzuki, "Traceable ring signature," in *Proceedings of the Public Key Cryptography - PKC 2007, 10th International Conference on Practice and Theory in Public-Key Cryptography*, pp. 181–200, Springer, Beijing, China, 16 April 2007.

[6] M. H. Au, J. K. Liu, W. Susilo, and T. H. Yuen, "Secure id-based linkable and revocable-iff-linked ring signature with constant-size construction," *Theoretical Computer Science*, vol. 469, no. 1–14, pp. 1–14, 2013.

[7] F. Tang, J. Pang, K. Cheng, and Q. Gong, "Multiauthority traceable ring signature scheme for smart grid based on blockchain," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 5566430, 9 pages, 2021.

[8] X. Peng, K. Gu, Z. Liu, and W. Zhang, "Traceable identity-based ring signature for protecting mobile iot devices," in *Proceedings of the Data Mining and Big Data - 6th International Conference, DMBD 2021*, pp. 158–166, Springer, Guangzhou, China, 20 October 2021.

[9] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pp. 124–134, IEEE Computer Society, Santa Fe, New Mexico, USA, 20 November 1994.

[10] P. Branco and P. Mateus, "A traceable ring signature scheme based on coding theory," in *Proceedings of the International Conference on Post-Quantum Cryptography*, pp. 387–403, Springer, Chongqing, China, 29 May 2019.

[11] R. Cramer, I. Damgård, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in *Proceedings of the Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference*, pp. 174–187, Springer, Santa Barbara, California, USA, 21 August 1994.

[12] H. Feng, J. Liu, Q. Wu, and Y.-N. Li, "Traceable ring signatures with post-quantum security," in *Proceedings of the Topics in Cryptology - CT-RSA 2020 - The Cryptographers' Track at the RSA Conference 2020*, pp. 442–468, Springer, San Francisco, CA, USA, 24 February 2020.

[13] A. Scafuro and B. Zhang, "One-time traceable ring signatures," in *Proceedings of the Computer Security - ESORICS 2021 - 26th European Symposium on Research in Computer Security*, pp. 481–500, Springer, Darmstadt, Germany, 4 October 2021.

[14] H. Feng, J. Liu, D. Li, Y.-N. Li, and Q. Wu, "Traceable ring signatures: general framework and post-quantum security," *Designs, Codes and Cryptography*, vol. 89, no. 6, pp. 1111–1145, 2021.

[15] N. T. Courtois, M. Finiasz, and N. Sendrier, "How to achieve a mceliece-based digital signature scheme," in *Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security*, pp. 157–174, Springer, 2001.

[16] N. Aragon, O. Blazy, P. Gaborit, A. Hauteville, and G. Zémor, "Durandal: a rank metric based signature scheme," in *Proceedings of the Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 728–758, Springer, Darmstadt, Germany, 19 May 2019.

[17] T. Debris-Alazard, N. Sendrier, and J.-P. Tillich, "Wave: a new family of trapdoor one-way preimage sampleable functions based on codes," in *Proceedings of the Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security*, pp. 21–51, Springer, Kobe, Japan, 8 December 2019.

[18] E. Persichetti, "Efficient one-time signatures from quasi-cyclic codes: a full treatment," *Cryptography*, vol. 2, no. 4, p. 30, 2018.

[19] D. Zheng, X. Li, and K. Chen, "Code-based ring signature scheme," *International Journal on Network Security*, vol. 5, no. 2, pp. 154–157, 2007.

[20] P. Branco and P. Mateus, "A code-based linkable ring signature scheme," in *Proceedings of the Provable Security - 12th International Conference, ProvSec 2018*, pp. 203–219, Springer, Jeju, South Korea, 25 October 2018.

[21] C. Aguilar Melchor, P.-L. Cayrel, P. Gaborit, and F. Laguillaumie, "A new efficient threshold ring signature scheme based on coding theory," *IEEE Transactions on Information Theory*, vol. 57, no. 7, pp. 4833–4842, 2011.

[22] L. Dallot and D. Vergnaud, "Provably secure code-based threshold ring signatures," in *Proceedings of the Cryptography and Coding, 12th IMA International Conference, Cryptography and Coding 2009*, pp. 222–235, Springer, Cirencester, UK, 15 December 2009.

[23] H. Assidi, E. B. Ayebie, and E. M. Souidi, "An efficient code-based threshold ring signature scheme," *Journal of Information Security and Applications*, vol. 45, pp. 52–60, 2019.

[24] Q. Alamélou, O. Blazy, S. Cauchie, and P. Gaborit, "A practical group signature scheme based on rank metric," in *Proceedings of the Arithmetic of Finite Fields - 6th International Workshop, WAIFI 2016*, pp. 258–275, Springer, Ghent, Belgium, 13 July 2016.

[25] M. F. Ezerman, H. T. Lee, S. Ling, K. Nguyen, and H. Wang, "A provably secure group signature scheme from code-based assumptions," in *Proceedings of the Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security*, pp. 260–285, Springer, Auckland, New Zealand, 29 December 2015.

[26] B. E. Ayebie, H. Assidi, and E. M. Souidi, "A new dynamic code-based group signature scheme," in *Proceedings of the Codes, Cryptology and Information Security - Second International Conference, C2SI 2017*, pp. 346–364, Springer, Rabat, Morocco, 10 April 2017.

[27] Q. Alamélou, O. Blazy, S. Cauchie, and P. Gaborit, "A code-based group signature scheme," *Designs, Codes and Cryptography*, vol. 82, no. 1-2, pp. 469–493, 2017.

[28] M. F. Ezerman, H. T. Lee, S. Ling, K. Nguyen, and H. Wang, "Provably secure group signature schemes from code-based assumptions," *IEEE Transactions on Information Theory*, vol. 66, no. 9, pp. 5754–5773, 2020.

[29] K. Nguyen, H. Tang, H. Wang, and N. Zeng, "New code-based privacy-preserving cryptographic constructions," *Lecture Notes in Computer Science*, in *Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security*, pp. 25–55, Springer, Kobe, Japan, 4 December 2019.

[30] A. Fiat and A. Shamir, "How to prove yourself: practical solutions to identification and signature problems," in *Proceedings of the Advances in Cryptology - CRYPTO '86*, pp. 186–194, Springer, Santa Barbara, California, USA, 11 August 1986.

[31] E. Berlekamp, R. McEliece, and H. Van Tilborg, "On the inherent intractability of certain coding problems (corresp.)," *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 384–386, 1978.

[32] D. Augot, M. Finiasz, and N. Sendrier, "A family of fast syndrome based cryptographic hash functions," in *Proceedings of the Progress in Cryptology - Mycrypt 2005, First International Conference on Cryptology*, pp. 64–83, Springer, Kuala Lumpur, Malaysia, 28 September 2005.

[33] Y. Zhang, D. He, F. Zhang, X. Huang, and D. Li, "An efficient blind signature scheme based on SM2 signature algorithm," in *Proceedings of the Information Security and Cryptology - 16th International Conference, Inscrypt 2020*, pp. 368–384, Springer, Guangzhou, China, 11 December 2020.

[34] R. C. Torres and N. Sendrier, "Analysis of information set decoding for a sub-linear error weight," in *Proceedings of the Post-Quantum Cryptography - 7th International Workshop, PQCrypto 2016*, pp. 144–161, Springer, Fukuoka, Japan, 24 February 2016.

[35] A. Becker, A. Joux, A. May, and A. Meurer, "Decoding random binary linear codes in 2 n/20: how 1 + 1 = 0 improves information set decoding," in *Proceedings of the Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 520–536, Springer, Cambridge, UK, 15 April 2012.

[36] D. J. Bernstein, T. Lange, C. Peters, and P. Schwabe, "Faster 2-regular information-set decoding," in *Proceedings of the Coding and Cryptology - Third International Workshop, IWCC 2011*, pp. 81–98, Springer, Qingdao, China, 30 May 2011.