One-time and Revocable Ring Signature with Logarithmic Size in Blockchain

Yang Li[1][0009-0008-4220-7567] and Wei Wang[1][0000-0002-5974-1589]

Dawei Zhang[1][0000-0001-5942-8245] and Xu Han[1][0000-0002-1030-2462]

[1] School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China

Abstract. Ring signature (RS) allows users to demonstrate to verifiers their membership within a specified group (ring) without disclosing their identities. Based on this, RS can be used as a privacy protection technology for users' identities in blockchain. However, there is currently a lack of RS schemes that are fully applicable to the blockchain applications: Firstly, users can only spend a UTXO once, and the current RS schemes are not yet perfect in a one-time manner. At the same time, the current RS schemes are not sufficiently developed in terms of regulation. Secondly, the size of the current RS is mostly linearly related to the number of ring members. When there are many members, the transaction processing speed is slow.

We propose a one-time and revocable ring signature with logarithmic size in blockchain based on the Sigma-Protocols. Our scheme compresses the RS size and enables users to sign in the blockchain transactions. The scheme allows two RS generated with the same private key for a same UTXO to be linked together. Additionally, it allows regulatory authority to recover the signer's identity at any time. A security model was presented, and its security properties, namely, unforgeability, anonymity, one-time, revocability, and non-slanderability were proven in the random oracle model.

Our scheme compresses the RS size to $O(logN)$ where N is the number of ring users, enabling blockchain transactions to have better processing speeds. And it can prevent double-spending attacks in blockchain and allows regulatory authority to recover the identity of the signer.

Keywords: Ring Signatures, Blockchain, Sigma-Protocols, Revocability, One-time, Logarithmic Size.

1 Introduction

Blockchain, also known as a distributed ledger, publicly records all transaction information. As many users do not want to expose their identities, it becomes particularly important to protect the privacy of users' identities in blockchain. The definition of identity privacy protection in blockchain is given in paper [1]. RS, a type of privacy protection technology that allows users to sign a transaction while keeping their identities hidden within a set (ring), can effectively obscure users' identities.

Therefore, RS can be employed as a privacy protection measure for users' identities and incorporated into blockchain transactions. The signature verification process allows for the authentication of the transaction's reliability, without revealing the identity of the signer within the ring.

Rivest et al.[2] proposed the concept of the RS. A ring is composed of a group of potential signers who voluntarily join, while the real signer uses his secret key to generate a RS. This RS can be verified by everyone, but the verifiers cannot determine which member of the ring specifically generated the signature. Therefore, the original motivation of RS is to facilitate anonymous reporting that permits whistleblowers to refrain from disclosing their identity, while ensuring the credibility of the report through the verification of the RS.

Linkable ring signature [3,4,5] adds a linkable property to the RS, such that if two RS are generated by the same signer, they will be linked. Linkable ring signature is suitable for electronic voting scenarios, where an individual possessing voting rights can cast votes on behalf of all voters without revealing his identity. However, if a voter votes twice, he will be linked. The paper [6] introduces a linkable ring signature scheme with strong anonymity and proves its security in the random oracle model. In numerous practical applications, the implementation of linkability requires a trade-off with anonymity, which has led to the emergence of traceable ring signature schemes.

Traceable ring signature, as described in [7], is an extension of RS that provides traceability. In the event that a legitimate signer generates two RS using the same tag, his identity becomes exposed. The ID-based RS scheme [8] only reveals the signer's identity when two RS are linked together.

Revocable ring signature [9] is an extension of RS that incorporates the property of revocability. The revocation authority has the ability to forcibly recover the signer's identity at any time.

It is important to apply both linkability and revocability of RS to blockchain transactions. In addition to extending the properties of RS, the reduction in the size of RS is very important.

1.1 Related Work

Ring Signature Schemes with Linear-Size. CryptoNote v 2.0 [10] introduced a one-time RS scheme, where the RS is $\sigma = (I, c_1, \ldots, c_n, r_1, \ldots, r_n)$, and I is called the "key image" and n represents the number of ring users. The size of the RS increases linearly with the number of ring users. To reduce the size of the RS, the concept of Ring Confidential Transaction (RingCT) [11] was proposed, where the RS is $\sigma = (I, c_1, s_1, \ldots, s_n)$. Although this scheme reduces the signature size by approximately 50% compared to the scheme presented in [10], it still maintains a linear relationship with the quantity of ring users. Another RS introduced in the paper [12] has both recoverability and linkability, and similarly, it also exhibits a linear correlation between the RS size and the number of ring users.

Ring Signature Scheme with Constant-Size. To further reduce the size of signatures, the concept of RingCT 2.0 was proposed in paper [13]. This scheme uses accumulators to ensure that the output size of the RS is constant and independent of the number of ring users. Unfortunately, the combination of accumulators and zero-knowledge proofs leads to a performance that is unfit for blockchain transactions. Paper [8] proposed a RS scheme, and the RS has a constant size, but the signer's identity is only revoked if the two RS are linked. A practical constant-size RS was proposed in paper [20], and the scheme was based on bilinear pairing and accumulator.

Ring Signature Scheme with Square-Root Size. In paper [14], a RS with size $O(\sqrt{N})$ was proposed, where N represented the number of users in the ring. Nonetheless, this RS lacks the attributes of linkability and revocability, consequently rendering it unsuitable for addressing the demands of blockchain transactions. The size of the traceable ring signature proposed in [22] is $\mathrm{O}(\sqrt{N})$, but the real signer's identity will be revealed only when the two RS are linked and the two signed messages are different.

Ring Signature Scheme with Logarithmic Size. In paper [15], a RS with logarithmic size was proposed based on a variant of multi-signature to compress the size of RS. The Σ-protocols in paper [16] were also modified to compress the size of the RS to logarithmic size. Nonetheless, the lack of consideration for the linkability and revocability in the RS scheme precludes direct application of the aforementioned scheme to our specific blockchain transaction scenario. In paper [17,21], the size of the RS was reduced to logarithmic size but it did not consider regulatory issues in blockchain.

1.2 Motivations and Contributions

Motivations. Privacy preservation of the user's identity is necessary in blockchain transactions, and attaining this privacy through mere anonymous authentication is inadequate. Actually, identity obfuscation technology is needed to actualize identity privacy protection. Additionally, the payment process is an imperative component in blockchain transactions, necessitating the prevention of double-spending by users. Moreover, in practical applications, the ability for regulatory oversight is typically needed to prevent excessive anonymity, which can result in transaction review issues and disputes.

Hence, it is of vital significance to investigate RS schemes that can be applied in the aforementioned scenario by combining the characteristics of RS. In blockchain transactions, users possess the autonomy to select a confusion set (a set of ring users) and hide their identities within this set. To ensure the credibility of the transaction, verifiers can verify the RS. Additionally, linkability should be integrated to preempt double-spending by users. When a user repeatedly spends the same UTXO, it will be linked, leading to suspension of the payment. Revocability needs to be added because

regulatory agencies may need to recover the identity of the real signer if necessary. In the instance of a transaction dispute, regulatory agencies can forcibly recover the identity of the signer. Since blockchain transactions require relatively fast processing speeds, it is equally vital to reduce the size of RS.

Currently, the existing RS schemes fail to entirely fulfill the above requirements. Motivated by this, we propose a one-time and revocable RS scheme with logarithmic size.

Contributions.

(1) We propose a new RS scheme with a one-time and revocable primitive applicable for privacy protection and reliable supervision in blockchain. It solves the problem that the current RS schemes are not fully applicable to blockchain. And we provide security proofs for the scheme in the random oracle model, including five security properties, namely unforgeability, anonymity, one-time, revocability, and non-slanderability.

(2) The size of the current RS schemes is mostly linearly related to the number of ring users. We propose a one-time and revocable ring signature with logarithmic size based on the new Σ-protocols with logarithmic communication complexity as espoused in paper [18]. When the number of the ring users is 1024, the RS size of our scheme is 89% less than the [12].

(3) We present the application process of applying the one-time and revocable ring signature to the blockchain transactions and provide an instantiation scheme.

1.3 Paper Organisation

In Sect.2, we present two hard problems and detail the prerequisite knowledge of ElGamal encryption, Σ-Protocols, and others. We outline the security model of a one-time and revocable ring signature scheme in Sect.3. The specification of our proposed scheme is provided in Sect.4, where we also prove the security of its five security attributes. In Sect.5, we instantiate and experimentally verify the scheme, as well as compare it with other existing RS schemes at the theoretical level.

2 Preliminaries

2.1 Hard Problem

Definition 1 (Discrete Logarithm (DL) Problem). *The Discrete Logarithm (DL) Problem in* $\mathbb{G}_q$ *is defined as follows: Given a tuple* $(y,g) \in \mathbb{G}_q{}^2$, *where* $\mathbb{G}q$ *is a cyclic group and* $|\mathbb{G}_q|=q$ *for a prime number q, finding a x such that* $y = g^x \pmod q$.

Definition 2 (Decisional Diffie-Hellman (DDH) Problem). *The Decisional Diffie-Hellman (DDH) Problem in the cyclic group* $\mathbb{G}_q$ *is defined as follows: Given a quadruple* $(g, g^a, g^b, Z) \in \mathbb{G}_q{}^4$ *where* $\mathbb{G}q$ *is a cyclic group and* $|\mathbb{G}_q|=q$ *for a prime number q, judging whether* $Z=g^{ab}$ *is established.*

2.2 ElGamal Public Key Encryption

We utilize the ElGamal encryption scheme to encrypt the signer's public key with a revocation authority's public key, and restore the real signer's public key with a revocation private key during revocation. The ElGamal encryption scheme is an algorithmic quadruple (**Setup**、**KeyGen**、**Encryption**、**Decryption**) :

— $param \leftarrow \mathbf{Setup}(\lambda)$: On input a security parameter λ, chooses a cyclic group $\mathbb{G}_q$, outputs public parameters $\mathrm{param} = \{\mathbb{G}_q, g, q\}$, where g is a generator in $\mathbb{G}_q$.

— $(sk, vk) \leftarrow \mathbf{KeyGen}(param)$: On input public parameters $param = \{\mathbb{G}_q, g, q\}$, generates a private/public key pair (sk, vk), where $vk = g^{sk} (mod\ q)$.

— $C \leftarrow \mathbf{Encryption}(M, vk_r)$: On input the message to be encrypted, denoted as M, and a receiver's public key $vk_r = y_r$, the sender randomly chooses a number $k \in Z_q$ and calculates $c_1 = g^k$ as the first part of the ciphertext C. Then the sender uses y_r to calculate the second part $c_2 = {y_r}^k M \pmod q$. Outputs the result $C = \{c_1, c_2\}$.

— $M \leftarrow \mathbf{Decryption}(C, sk_r)$: Takes the input $C = \{c_1, c_2\}$ and the receiver's private key $sk_r = x_r$, computes $M = c_2 \setminus {c_1}^{x_r} \pmod q$ to obtain the message M.

2.3 Pedersen Commitment Based on Elliptic Curve

Pedersen commitment based on elliptic curves consists of two phases: commitment generation phase and commitment opening phase, and it possesses additive homomorphism.

The commitment generation phase: Let G and H be two points in an elliptic curve. The sender randomly picks a blinding factor r, and v represents the information to be committed. The sender generates a commitment $C = r * G + v * H$ and sends it to the verifier.

The commitment opening phase: During the opening phase, the sender sends r and v to the verifier, then verifier calculates $C' = r * G + v * H$ and verifies if $C' = C$.

Additive homomorphism: If the commitment value of message v_1 is $C(v_1) = r_1 * G + v_1 * H$ and the commitment value of message v_2 is $C(v_2) = r_2 * G + v_2 * H$, then it satisfies $C(v_1 + v_2) = C(v_1) + C(v_2)$.

2.4 Σ-Protocols

In paper [16], the Σ-protocol is a specific category of 3-move interactive proof system, facilitating the prover's task of persuading the verifier about the truth of a given statement. Consider a polynomial-time decidable ternary relation, denoted as R. We define a witness w for a statement u if $(ck, u, w) \in R$. Define the CRS-dependent language

$$L_{ck} = \{u | \exists w: (ck, u, w) \in R\} \tag{1}$$

as the collection of statements u that possess a witness w within the relation R.

A Σ-protocol for R comprises a triplet of stateful interactive algorithms ($\mathcal{G}$, $\mathcal{P}$, $\mathcal{V}$), each operating within probabilistic polynomial time. The ensuing execution of a Σ-protocol describes the interaction of the algorithms:

$ck \leftarrow \mathcal{G}(1^{\lambda})$: Generates the commitment key.

$a \leftarrow \mathcal{P}(ck, u, w)$: Takes the input (ck, u, w) and the prover generates the initial message a.
$x \leftarrow \{0,1\}^{\lambda}$: The verifier randomly picks the challenge value x.
$z \leftarrow \mathcal{P}(x)$: The prover returns z as the answer to x.
$b \leftarrow \mathcal{V}(ck, u, a, x, z)$: The verifier algorithm, returns 1 if the proof is accepted, whereas it returns 0 otherwise.

According to the paper [16], we will give respectively a Σ-protocol for commitment to 0 or 1 and a Σ-protocol for one out of N commitments being a commitment which opening result is 0.

Σ-protocol for Commitment to 0 or 1. Let ck be a commitment key, which is a set of public parameters such as $\{\mathbb{G}_q, g, q, h\}$. c is a commitment value, m is the information to be committed, r is a random number and R is a relation that indicates the committed value of m is either 0 or 1:

$$R = \{(ck, c(m,r)) | c = Com_{ck}(m; r) \ and \ m \in \{0,1\} \ and \ r \in Z_q\} \tag{2}$$

$\mathrm{Com_{ck}(m; r)}$ is a homomorphic commitment that:

$$Com_{ck}(m_0; r_0) \cdot Com_{ck}(m_1; r_1) = Com_{ck}(m_0 + m_1; r_0 + r_1) \tag{3}$$

$\mathcal{P}(ck, c, (m,r))$		$\mathcal{V}(ck, c)$
$a, s, t \leftarrow Z_q$	$c_a, c_b \rightarrow$	
$c_a = Com_{ck_0}(a; s)$		
$c_b = Com_{ck_0}(am; t)$	$x \leftarrow \{0,1\}^{\lambda} \leftarrow$	Accept if and only if
$f = mx + a$		$c_a, c_b \in C_{ck}, \mathrm{f}, z_a, z_b \in Z_q$
$z_a = rx + s$	$f, z_a, z_b \rightarrow$	$c^x c_a = Com_{ck_0}(f; z_a)$
$z_b = r(x - f) + t$		$c^{x-f} c_b = Com_{ck_0}(0; z_b)$

Fig. 1. Σ-protocol for Commitment to m is 0 or 1

In Fig. 1, we give a Σ-protocol for R, where $\mathcal{P}$, $\mathcal{V}$ are running on $ck \leftarrow \mathcal{G}(1^{\lambda})$. And C_{ck} denotes the commitment space using ck as the commitment key, $m \in \{0,1\}$ and $r \in Z_q$. The prover has the capability to demonstrate to the verifier that he possess the witness m and can effectively persuade the verifier that the $m \in \{0,1\}$.

This protocol is perfectly complete, meaning that if commitment c represents a commitment to either 0 or 1, it can be successfully verified. It also possesses soundness, guaranteeing that the value of m is indeed 0 or 1. Additionally, this protocol offers perfect zero-knowledge with respect to an honest verifier, denoting that throughout the execution of this protocol, the verifier is unable to learn the actual value of m.

Σ-protocol for One Out of N Commitments Containing 0. This protocol is an extension of the previous protocol, which consists of N commitments, only one of which is a commitment to 0, and satisfies the following relationship R.

$$R = \{(ck, (c_0, \ldots, c_{N-1}), (l, r)) | c_0, \ldots, c_{N-1} \in C_{ck} \; and$$
$$l \in \{0, \ldots, N-1\} \; and \; r \quad \in Z_q \; and \; c_l = Com_{ck}(0; r)\} \tag{4}$$

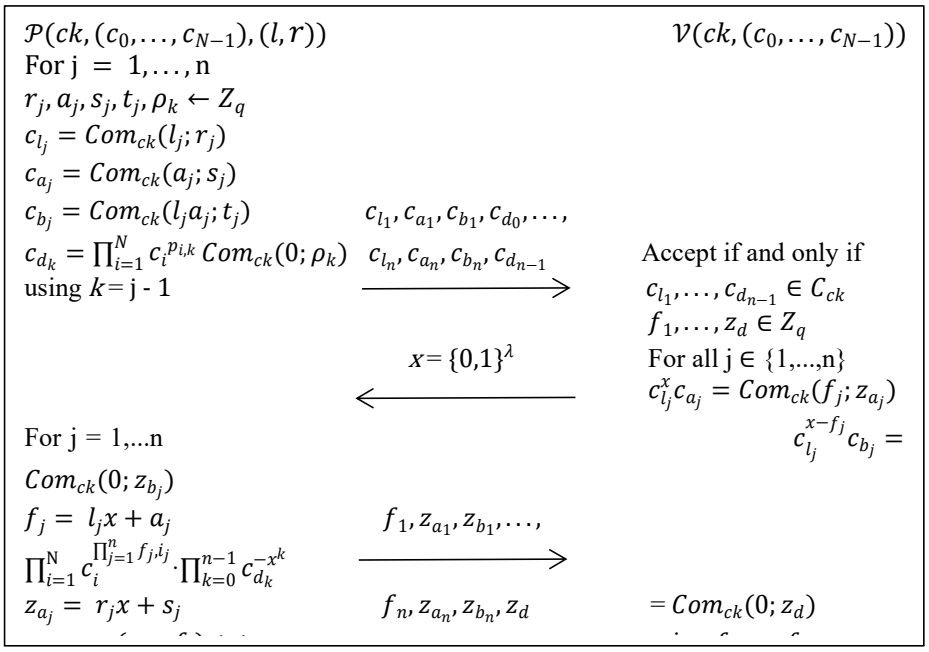

Fig. 2. Σ-protocol for Commitment to $m = 0$ in list $c_0, \ldots, c_{N-1}$

In Fig. 2, we can see that the protocol expresses indices in binary form $n = log_2 N$. The assumption in the protocol is that $f_{j,1} = f_j$ and $f_{j,0} = x - f_j$, so for each i that the product $\prod_{j=1}^{n} f_{j,i_j}$ is a polynomial for the form (5), where δ_{il} is Kronecker's delta, i.e., $\delta_{il} = 1$ when $i = l$ and $\delta_{il} = 0$ when $i \neq l$.

$$p_i(x) = \delta_{il} x^n + \sum_{k=0}^{n-1} p_{i,k} x^k \tag{5}$$

In the form $c_{d_k} = \prod_{i=1}^{N} c_i^{p_{i,k}} Com_{ck}(0; \rho_k)$, the $p_{i,k}$ is from (5).

The protocol is perfectly complete, meaning that if one of the commitments in $c_0, \ldots, c_{N-1}$ is a commitment to 0, it can be verified successfully. It also has soundness, meaning that the protocol guarantees that at least one commitment's opening result is 0. Additionally, it has zero-knowledge property, indicating that during running the protocol, the verifier cannot learn the value of the index for the commitment to 0.

3 Scheme Definition and Security Model

3.1 Definition of One-time and Revocable Ring Signature with Logarithmic Size

One-time and Revocable Ring Signature with Logarithmic Size. Within a RS, the ring is formed by a collection of public keys belonging to the ring users. When creating a RS, the signer is required to posses the secret key associated with one of the public keys within the ring. The term 'one-time' in this paper refers to a signer using his private key to sign an event only once, while revocability refers to the ability of a revocation authority to recover the signer's identity at any time.

A one-time and revocable ring signature scheme encompasses six algorithms, denoted as (**SysGen**, **KeyGen**, **Sign**, **Verify**, **OneTimeLink**, **Revoke**), which are detailed as follows.

— $pp \leftarrow \mathbf{SysGen}(1^{\lambda})$: This is a probabilistic polynomial-time (PPT) algorithm. It accepts a security parameter λ as its input and produces a collection of system public parameters pp.

— $(sk, vk) \leftarrow \mathbf{KeyGen}(pp)$: This is a PPT algorithm. It accepts pp as its input and generates a public and secret key pair (sk, vk).

— $\sigma \leftarrow \mathbf{Sign}(event, R, sk_l, vk_{rev}, M)$: Takes the input of an event description $event$, a set $R=\{vk_0,\ldots,vk_{N-1}\}$ containing N public keys, a secret key sk_l of a signer which corresponds to $vk_l \in R$, the public key vk_{rev} of a revocation authority and a message M, the algorithm produce a signature σ. And one-time tag $T \in \sigma$ and $C \in \sigma$.

— $0/1 \leftarrow \mathbf{Verify}(event, R, vk_{rev}, M, \sigma)$: Accepts the following inputs: an event description $event$, a set $R=\{vk_0,\ldots,vk_{N-1}\}$ containing N public keys, the revocation authority's public key vk_{rev}, a message M and the signature σ. If σ is valid, the algorithm outputs 1 and adds $T \in \sigma$ to the one-time tag list L. Otherwise, the algorithm outputs 0.

— $0/1 \leftarrow \mathbf{OneTimeLink}(\sigma, L)$: On input a signature σ and a one-time tag list L. If the one-time tag T exists in L, meaning the private key that produces the σ has been previously used, the algorithm will output 1. If not, it outputs a result of 0.

— $vk_l \leftarrow \mathbf{Revoke}(R, sk_{rev}, \sigma)$: Takes the input of a set $R=\{vk_0,\ldots,vk_{N-1}\}$ containing N public keys, a revocation private key sk_{rev} which corresponds to vk_{rev}, a valid signature σ, returns the real signer's public key vk_l.

A one-time and revocable ring signature scheme should satisfies the ensuing correctness properties:

· $Verification\ Correctness$. When a signature is generated by an honest signer, the verification process will confirm it as a valid signature, and as a result, the algorithm **Verify** will produce an output of 1.

$$1 \leftarrow \mathbf{Verify}(event, R, vk_{rev}, M, \mathbf{Sign}(event, R, sk_l, vk_{rev}, M))$$

· $One-time\ Correctness$. The algorithm outputs 0 when the signature private key has been employed for generating a RS for the same *event*.

$$0 \leftarrow \mathbf{OneTimeLink}(\sigma, L)$$

· $Revocation\ Correctness$. If a valid signature is produced by an honest signer, then the revocation authority can certainly recover the signer's identity.

$$vk_l \leftarrow \mathbf{Revoke}(R, sk_{rev}, \sigma)$$

Here σ is a RS produced from the **SysGen** by an honest signer. The ciphertext C used for revocation is obtained using the public key vk_{rev}, allowing the revocation authority to reinstate the signer's identity by means of the secret key sk_{rev}.

Below, we will describe the security definition of a one-time and revocable ring signature scheme. The scheme should satisfy five security properties, namely unforgeability, anonymity, one-time, revocability, and non-slanderability. Before providing the definitions of these security properties, we first introduce the following oracle to simulate the adversary's attack capabilities on the security of the scheme.

· $Registering\ Oracle$ (O_J). Adversary A is able to query O_J to add a new user to the system. O_J will return the public key of the new user. O_J simulates the ability of adversary A to obtain the public keys of honest users.

· $Capturing\ Member\ Oracle$ (O_C). Adversary A holds the capability to interact with O_C to acquire the secret key sk_i corresponding to a certain public key vk_i. O_C accepts a public key and returns the corresponding secret key while also incorporating the public key into the set S_C. O_C simulates A's ability to control a user's key pair (vk_i, sk_i) in the ring.

· $Signing\ Oracle$ (O_S). Adversary A can query O_S by using $(event, R, vk_{rev}, M)$ as input, where $event$ is an event description, R is a set of N public keys generated by **KeyGen**. vk_{rev} is a revocation public key. O_S can execute the Sign algorithm to return to A a valid RS $\sigma \leftarrow \text{Sign}(event, R, sk_l, vk_{rev}, M)$, where sk_l is the secret key corresponding to vk_l in R. O_S simulates that adversary A can obtain some valid RS corresponding to R by observation.

3.2 Security Definitions

Unforgeability. Unforgeability pertains to the attacker's inability to create a valid signature for a new message without possessing knowledge of any private key within the ring. We define the unforgeability of the scheme through an interactive game involving the challenger C and the adversary A:

— C runs the algorithm **SysGen** for generating public parameters for the execution system, and sends the generated public parameters pp to A.

— A queries O_J for a set of public keys $R=\{vk_0,\ldots,vk_{N-1}\}$ and adaptively makes query to O_S. A will obtain a valid RS corresponding to the set R.

— A selects a message M, an event description event, the set $R'\subseteq R$ of N' public keys that make up the ring, a revoke public key vk_{rev}. Then A constructs a RS $\sigma \leftarrow A(event, R', vk_{rev}, M)$ such that each public key in R' has not been previously queried as an input from O_C, and σ has not been obtained through querying to O_S, i.e., σ entirely generated by A. A sends σ to C.

— C runs the Verify algorithm, if $\mathbf{Verify}(event, R', vk_{rev}, M, \sigma)=1$, A wins the game

We represent the advantage of adversary A in breaking the unforgeability of the scheme as

$$\mathrm{Adv}_{\mathrm{A}}^{\mathrm{Unf}}(\lambda) = P_r[A \text{ wins the game}].$$

Definition 3 (Unforgeability). *If* $\boldsymbol{Adv_A^{Unf}(\lambda)}$ *is negligible for any probabilistic polynomial-time (PPT) adversary* $\boldsymbol{A}$*, then the one-time and revocable ring signature scheme is unforgeable*.

Anonymity. Anonymity pertains to the likelihood that an attacker can identify the actual signer of a valid RS with a probability not exceeding 1/N, where N represents the count of users within the ring. This condition assumes that the private keys of the ring users remain confidential and undisclosed to the public. We define the anonymity of the scheme through an interactive game involving the challenger C and the adversary A:

— C runs the algorithm **SysGen** for generating public parameters for the execution system, and sends the generated public parameters pp to A.

— A queries O_J for a set of public keys R=$\{vk_0,\ldots,vk_{N-1}\}$ and adaptively makes query to O_S. A will obtain some valid RS corresponding to the set R.

— A selects a message M, an event description $event$, two public keys $vk_0 \in R'$ and $vk_1 \in R'$ where $R' \subseteq R$, a revoke public key vk_{rev}. Then A sends $(event, vk_0, vk_1, vk_{rev}, M)$ to C. C randomly picks $b \in \{0,1\}$ and computes a RS σ using vk_b. C sends σ to A. It is required that the vk_0 and vk_1 have not been queried to O_S and O_c as users of the ring.

— A guesses $b' \in \{0,1\}$, if $b'=b$, A wins the game.

We represent the advantage of adversary A in breaking the anonymity of the scheme as

$$\mathrm{Adv}_{\mathrm{A}}^{\mathrm{Anon}}(\lambda) = \left| P_r[b' = b] - \frac{1}{2} \right|.$$

Definition 4 (Anonymity). *If* $Adv_A^{Anon}(\lambda)$ *is negligible for any probabilistic polynomial-time (PPT) adversary A, then the one-time and revocable ring signature scheme is anonymous to the signer.*

One-time. One-time means that if an attacker uses the same key to sign the same event twice, the output of the OneTimeLink algorithm will be 0. Applying one-time to UTXO payment transactions can prevent double-spending. When a verifier detects that the one-time tag of a payment transaction already exists in the one-time tag list, the transaction will be blocked from continuing because this indicates that the signer is attempting to make duplicate payments using the same UTXO. We define the one-time of the scheme by the interactive game between a challenger C and an adversary A as follows:

— C runs the algorithm **SysGen** for generating public parameters for the execution system, and sends the generated public parameters pp to A.

— A queries O_J for a set of public keys $R=\{vk_0,...,vk_{N-1}\}$ and adaptively makes query to O_S amd O_C. A will obtain some valid RS and some private keys associated with the public keys in R.
— A selects a public key $vk_i \in R$ and constructs two RS σ_0 and σ_1 using the same private key sk_i. A sends σ_0 and σ_1 to C. It is required that sk_i is the output of O_C and σ_0,σ_1 are not the output of O_S.
— C runs the algorithm **Verify** and **OneTimeLink**.

A wins the game if

$\mathbf{Verify}(event, R, vk_{rev}, M, \sigma_0)$=1, $\mathbf{Verify}(event, R, vk_{rev}, M, \sigma_1)$=1, and $\mathbf{OneTimeLink}(\sigma_0, L)$=1 , $\mathbf{OneTimeLink}(\sigma_1, L)$=1.

We represent the advantage of adversary A in breaking the one-time of the scheme as

$$\mathrm{Adv}_{\mathrm{A}}^{\mathrm{OneTime}}(\lambda) = P_r[A \text{ wins the game}].$$

Definition 5 (One-time). *If $Adv_A^{OneTime}(\lambda)$ is negligible for any probabilistic polynomial-time (PPT) adversary A, then the one-time and revocable ring signature scheme has one-time.*

Remark. Consider the following scenario: a signer possesses two signing keys which can generate different one-time tags for the same transaction. This is allowed as long as the signer is using different UTXO.

Revocability. Revocability refers to the ability of the revocation authority to restore the true identity of the signer in any circumstance, regardless of whether the signature is one-time or not. The attacker wants to achieve the purpose that the signer identity recovered by the revocation authority is different from that of the attacker. We define the revocability of the scheme by the interactive game between a challenger C and an adversary A as follows:
— C runs the algorithm **SysGen** for generating public parameters for the execution system, and sends the generated public parameters pp to A.
— A queries O_J for a set of public keys $R=\{vk_0,...,vk_{N-1}\}$ and adaptively makes query to O_S amd O_C. A will obtain some valid RS and some private keys associated with the public keys in R.
— A selects a message M, an event description $event$, the set $R' \subseteq R$ contains N' public keys that make up the ring, a revoke public key vk_{rev} and constructs a RS σ which generates by using a private key sk_l corresponding to the vk_l. A sends σ to C. It is required that σ is not the output of O_S.
— C runs the algorithm $vk' \leftarrow \mathbf{Revoke}(R', sk_{rev}, \sigma)$, if $vk' \neq vk_l$, A wins the game.

We represent the advantage of adversary A in breaking the revocability of the scheme as

$$\mathrm{Adv}_{\mathrm{A}}^{\mathrm{Rev}}(\lambda) = P_r[A \text{ wins the game}].$$

Definition 6 (Revocability). *If $Adv_A^{Rev}(\lambda)$ is negligible for any probabilistic polynomial-time (PPT) adversary A, then the one-time and revocable ring signature scheme is revocable.*

Non-Slanderability. Non-Slanderability refers to the impossibility for an attacker to create a one-time tag that is identical to the one created with another user's private key. In a transactional context, non-slanderability means that an attacker cannot use defamatory means to steal the assets of other users. We define the non-slanderability of the scheme by the interactive game between a challenger C and an adversary A as follows:

— C runs the algorithm **SysGen** for generating public parameters for the execution system, and sends the generated public parameters pp to A.

— A queries O_J for a set of public keys $R=\{vk_0,...,vk_{N-1}\}$ and adaptively makes query to O_S amd O_C, especially A can use $vk_i=\{vk_0,vk_1,...,vk_{j-1},vk_{j+1},...,vk_{N-1}\}$ as the input of O_C and obtain the output $sk_i=\{sk_0,sk_1,...,sk_{j-1},sk_{j+1},...,sk_{N-1}\}$.

— A sends C a message M, an event description $event$, a set $R'\subseteq R$ containing vk_j and a revoke public key vk_{rev}. C generates a RS σ using vk_j as the signer. C sends σ to A and the σ containing the one-time tag I_j. It is required that vk_j has not been queried to O_C as an input and it has not been queried to O_S as a ring member.

— A selects a message M, an event description $event$, the set $R'\subseteq R$ of N' public keys and $vk_j \subseteq R'$, a revoke public key vk_{rev} and constructs a RS $\sigma^* \leftarrow A(event,R^*,sk_i,vk_{rev},M)$. σ^* contains a one-time tag I. A sends σ^* to C.

— C runs the algorithm Verify, if the algorithm outputs 1 and $I=I_j$, A wins the game.

We represent the advantage of adversary A in breaking the non-slanderability of the scheme as

$$\mathrm{Adv}_{\mathrm{A}}^{\mathrm{Nons}}(\lambda)= P_r[A \text{ wins the game}].$$

Definition 7 (Non-Slanderability). *If $Adv_A^{Nons}(\lambda)$ is negligible for any probabilistic polynomial-time (PPT) adversary A, then the one-time and revocable ring signature scheme is non-slanderable.*

4 Scheme Description

4.1 One-time and revocable ring signature scheme with logarithmic size description

— $pp \leftarrow \mathbf{SysGen}(1^\lambda)$: On input a security parameter λ, chooses a cyclic group $\mathbb{G}q$ with prime order q, g and h are generators of $\mathbb{G}q$. Assuming the discrete logarithm problem in $\mathbb{G}_q$ is difficult. Define two cryptographic hash functions: $H_1:\{0,1\}^* \rightarrow Z_q$ and $H_2:\{0,1\}^* \rightarrow \mathbb{G}_q$. Outputs the public parameters $pp=\{\mathbb{G}_q,g,h,q,H_1,H_2\}$.

— $(sk,vk) \leftarrow \mathbf{KeyGen}(pp)$: On input the pp. According to the idea in paper [16], each public key in the set of ring users is a commitment to 0 that is $vk = c = Com_{ck}(m;r) = Com_{ck}(0;r)$. If we instantiate the scheme with Pedersen

commitment then vk=$Com_{ck}(0;r) = h^0 g^r = g^r$, the public key of a ring member is vk=c=g^r. We assume $N = 2^n$. User i, where $i \in$[0,N-1], randomly chooses $r_i \in Z_q$ as the private key and computes $vk_i = g^{r_i} (\mathrm{mod}\ q)$. So the user i has a private/public key pair (sk_i , vk_i) . Assuming the signer is the l -th user, he has key pair(sk_l,vk_l)=(r_l,g^{r_l}). Revocation authority has key pair (sk_{rev},vk_{rev})=(r_{rev}, $g^{r_{rev}}$). We denote SK as the set of possible private keys and VK as the set of possible public keys.

— $\sigma \leftarrow$ **Sign**($event, R, sk_l, vk_{rev}, M$): It accepts a message M, and a event description $event$, a set R contains N' public keys $\{vk_0,..., vk_{N'-1}\}$, a signer's private key $sk_l \in SK$, where $l \in [0, N-1]$ corresponding to a $vk_l \in R$, a revoke public key $vk_{rev} \in VK$ as inputs. In order to facilitate representation, we will assume $sk_l = x_l$, $vk_l = y_l$, $sk_{rev} = x_{rev}$, $vk_{rev} = y_{rev}$ and $y_i = \{vk_0,..., vk_{N'-1}\}$. The signer needs to prove its existence in the ring, but the verifier cannot know which public key secret index it corresponds to in the ring. Therefore, following the approach described in [16], the signer needs to prove he/she knows the opening value of one commitment among N commitments to 0, and this proof is combined with the Fiat-Shamir heuristic to construct a non-interactive zero-knowledge proof of a RS. The signer with the private key x_l generates a RS according to the following steps:

1. Compute the one-time tag by committing to x_l:
   (a) $E \leftarrow H_2(event)$;
   (b) $T \leftarrow E^{x_l}$.
2. Randomly pick $u \in Z_q$ and encrypt y_l using y_{rev} according to the ElGamal Encryption:
   (a) $C_1 \leftarrow g^u$;
   (b) $C_2 \leftarrow {y_{rev}}^u y_l$;
   (c) $C \leftarrow \{C_1, C_2\}$.
3. Signer generates a RS σ based on the interactive zero-knowledge proof PK1 :
   (a) The signer needs to prove he possesses the opening result x_l of one of the N commitments to 0:

$$\mathrm{PK1}\{(x_l): y_0 = Com_{ck}(0;x_0) \vee y_1 = Com_{ck}(0;x_1) \vee \ldots \vee y_l = Com_{ck}(0;x_l) \vee \ldots \vee y_{N-1} = Com_{ck}(0;x_{N-1})\}.$$

   This is a process of partial knowledge proof. Here, ck represents the public parameters pp. The PK1 is shown in Fig. 3.

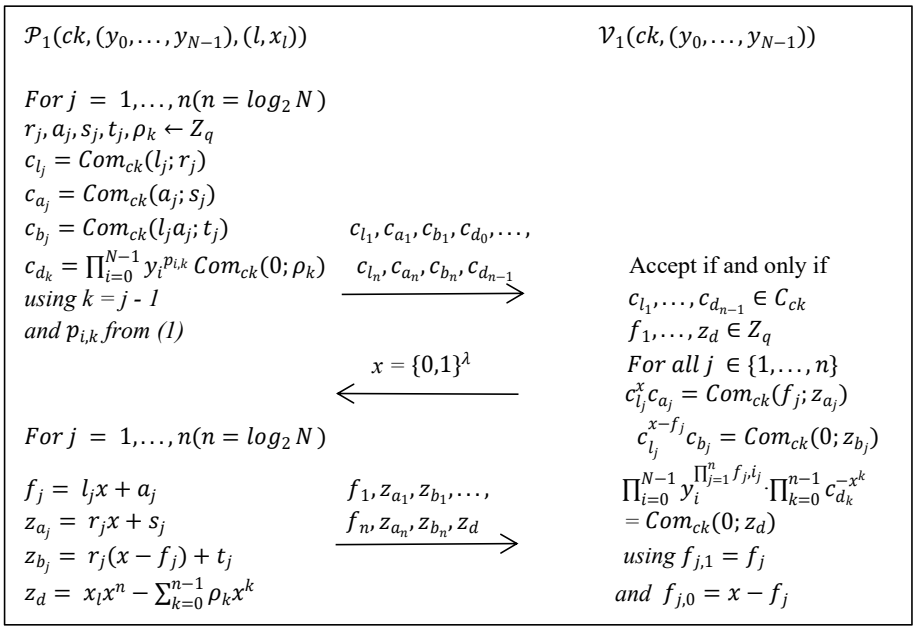

Fig. 3. Signer proves that he/she have a witness corresponding to a public key in y_l

Let a_1=$\{c_{l_1},c_{a_1},c_{b_1},c_{d_0},\ldots,c_{l_n},c_{a_n},c_{b_n},c_{d_{n-1}}\}$,
z_1=$\{f_1,z_{a_1},z_{b_1},\ldots,f_n,z_{a_n},z_{b_n},z_d\}$.

(b) The signer needs to prove he possesses the private key x_l used to generate the one-time tag T:

$$\mathrm{PK2}\{(x_l): T = Com_{ck'}(0;x_0) \vee T = Com_{ck'}(0;x_1) \vee \ldots \vee T = Com_{ck'}(0;x_l) \vee \ldots \vee T = Com_{ck'}(0;x_{N-1})\}.$$

This is a process of partial knowledge proof. Here, ck' represents the public parameters$\{\mathbb{G}_q, event, H_2\}$. The PK2 is shown in Fig. 4.

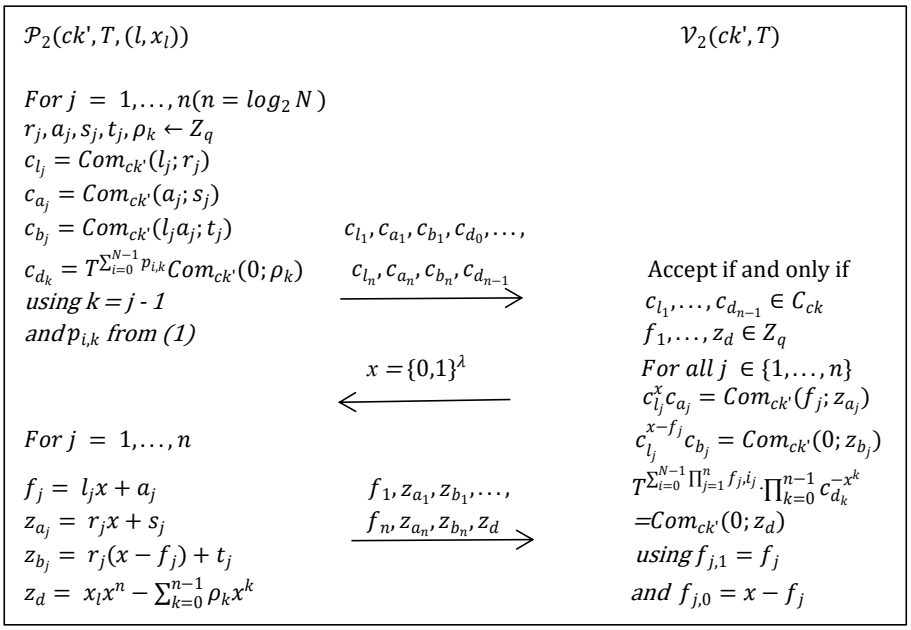

Fig. 4. Signer proves that he/she have the private key used to generate the one-time tag T

Let a_2=$\{c_{l_1}, c_{a_1}, c_{b_1}, c_{d_0}, \ldots, c_{l_n}, c_{a_n}, c_{b_n}, c_{d_{n-1}}\}$
and z_2=$\{f_1, z_{a_1}, z_{b_1}, \ldots, f_n, z_{a_n}, z_{b_n}, z_d\}$.

(c) The signer needs to prove they have the witness u:
PK3{(u): $([C_1 = Com_{ck_0}(0; u) \wedge (C_2 \setminus y_0) = Com_{ck_1}(0; u))] \vee [C_1 = Com_{ck_0}(0; u) \wedge (C_2 \setminus y_1) = Com_{ck_1}(0; u)] \vee \ldots \vee (C_1 = Com_{ck_0}(0; u) \wedge (C_2 \setminus y_l) = Com_{ck_1}(0; u)) \vee \ldots \vee [C_1 = Com_{ck_0}(0; u) \wedge (C_2 \setminus y_{N-1}) = Com_{ck_1}(0; u)]$}

This proof consists of two parts: the first part conducts a proof of knowledge, while the second part conducts a partial knowledge proof. Here, ck_0=$\{\mathbb{G}_q, g, h, q, H_1, H_2\}$, ck_1=$\{\mathbb{G}_q, y_{rev}, h, q, H_1, H_2\}$. The PK3 is shown in Fig. 5.

Let c represent C_1 and $\{c_0, \ldots, c_{N-1}\}$ represent $\{(C_2 \setminus y_0), \ldots, (C_2 \setminus y_{N-1})\}$.

Let a_3=$\{c_a, c_b, c_{l_1}, c_{l_1}, c_{a_1}, c_{b_1}, c_{d_0}, \ldots, c_{l_n}, c_{a_n}, c_{b_n}, c_{d_{n-1}}\}$ and z_3=$\{f, z_a, z_b, f_1, z_{a_1}, z_{b_1}, \ldots, f_n, z_{a_n}, z_{b_n}, z_d\}$, $\mathcal{P}$=$\{\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3\}$, $\mathcal{V}$=$\{\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3\}$.

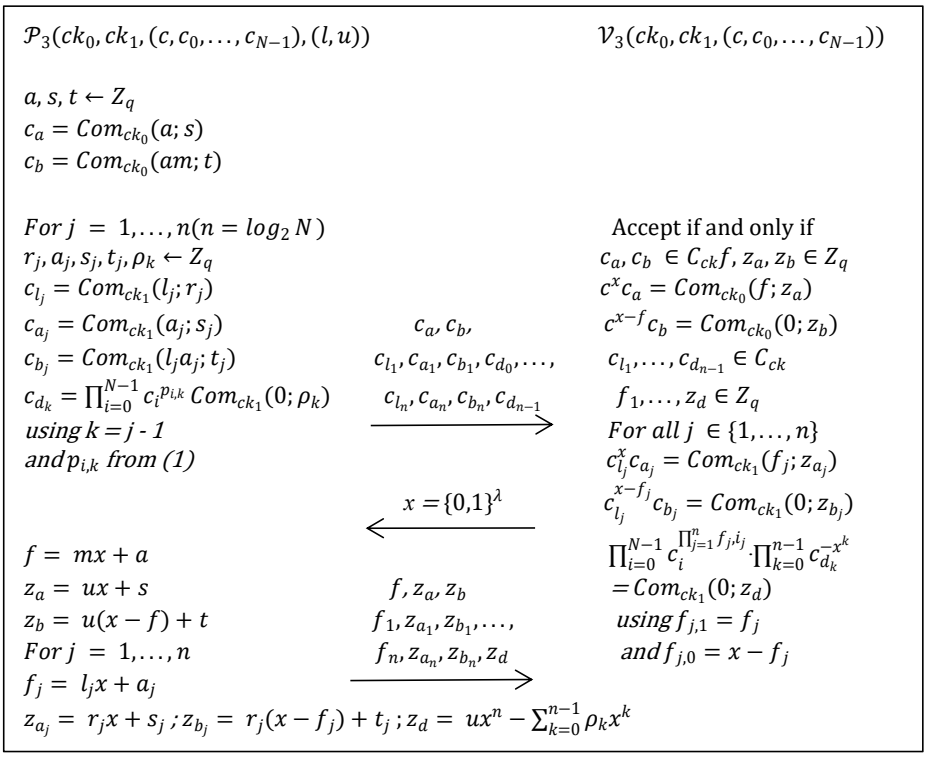

Fig. 5. Signer proves that he/she have the witness u

Combining the aforementioned interactive zero-knowledge proof with the Fiat-Shamir heuristic. Let a_1, a_2, a_3 in the Σ-protocol, the initial message M, the event description $event$, a set R, a one-time tag T and the ciphertext C as the inputs of hash function that is $H_1(event, R, M, T, C, g, h, y_{rev}, a_1, a_2, a_3)$. And let challenge value $x = H_1(event, R, M, T, C, g, h, y_{rev}, a_1, a_2, a_3)$. Combine the aforementioned (a), (b), and (c) together:

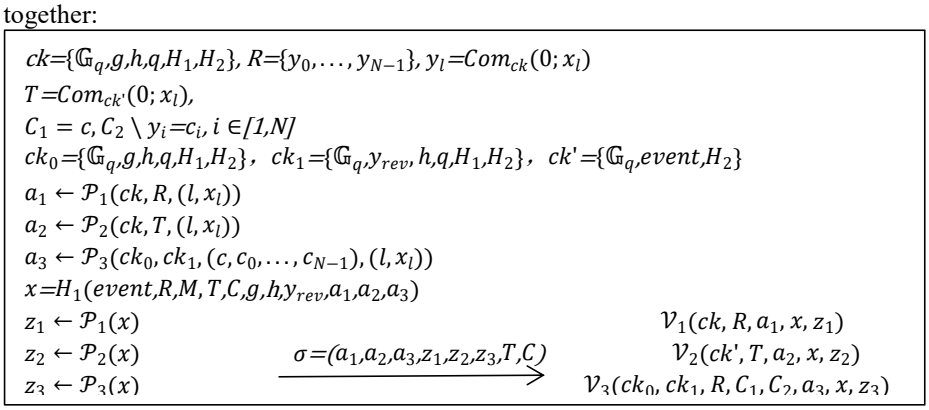

Fig. 6. The process of generating and validating a ring signature σ

So we can get the RS σ

$$\sigma=(a_1,a_2,a_3,z_1,z_2,z_3,T,C).$$

— 0/1← **Verify**$(event, R, vk_{rev}, M, \sigma)$: It accepts inputs $(event, R, vk_{rev}, M, \sigma)$ where M is a message, $event$ is an event's description, R is the set of N public keys, vk_{rev} is the revocation authority's public key and σ is a RS. The algorithm **Verify** consists of three sub-verification algorithms, namely $\mathcal{V}_1$, $\mathcal{V}_2$ and $\mathcal{V}_3$. These sub-verification processes are designed to individually ascertain whether the signer holds the secret key of a particular ring user, has the private key required for computing a one-time tag, and possesses the necessary witness for generating the ciphertext C. The verification of the RS proceeds in accordance with the subsequent steps:

1. $x=H_1(event, R, M, T, C, g, y_{rev}, a_1, a_2, a_3)$;
2. Obtain C_1, C_2 from $C=\{C_1, C_2\}$;
3. For $\mathcal{V}_1$, takes as input $\{ck, R, a_1, x, z_1\}$, returns the result $r_1=0/1$;
4. For $\mathcal{V}_2$, takes as input $\{ck', T, a_2, x, z_2\}$, returns the result $r_2=0/1$;
5. For $\mathcal{V}_3$, takes as input $\{ck_0, ck_1, R, C_1, C_2, a_3, x, z_3\}$, returns the result $r_3=0/1$;
6. If $r_1=1, r_2=1$ and $r_3=1$, Verify outputs 1. Otherwise, the algorithm outputs 0.

— 0/1← **OneTimeLink**(σ, L): On input a RS σ containing the one-time tag T and a one-time tag list L. If there is a tag T' in the list L that is the same as T, the algorithm outputs 0. Otherwise, the result is 1.

— $vk_l \leftarrow$ **Revoke**(R, sk_{rev}, σ): On input a set R of N public keys, a revocation secret key sk_{rev} and a valid RS σ. The process for the revocation authority to recover the genuine signer of a RS involves the following procedures:

1. Have C_1 and C_3 from C;
2. $\exists vk_l \in R(l \in [0, N-1])$ such that $vk_l = C_2 \setminus {C_1}^{x_{rev}}$.

4.2 Correctness Analysis

Verification Correctness. The verifier follows these steps when verifying the validity of the RS:

The verifier runs the algorithm $\mathcal{V}_1$. Due to $N=2^n$, for $j=1,\ldots,n$, if the commitment corresponding to the signer's secret index is a commitment to 0, then $c_{l_j}^x c_{a_j} = Com_{ck}(f_j; z_{a_j})$ and $c_{l_j}^{x-f_j} c_{b_j} = Com_{ck}(0; z_{b_j})$ can be verified to pass; Due to $c_{l_j}^x c_{a_j} = Com_{ck}(l_j; r_j)^x \cdot Com_{ck}(a_j; s_j) = Com_{ck}(l_j \cdot x + a_j; r_j \cdot x + s_j) = Com_{ck}(f_j; z_{a_j})$, $c_{l_j}^{x-f_j} c_{b_j} = (c_{l_j}^x \setminus c_{l_j}^{f_j}) c_{b_j} = Com_{ck}(l_j(x-f_j); r_j(x-f_j)) \cdot Com_{ck}(l_j a_j; t_j) = Com_{ck}(l_j(x-f_j+a_j); r_j(x-f_j)+t_j)$ and $f_j = l_j x + a_j$, the formula $l_j(x-f_j+a_j) = l_j x(1-l_j)$ established; *if* $l_j=0$, $c_{l_j}^x c_{a_j} = Com_{ck}(0; z_{b_j})$ can be verified to pass; If the signer has the private key and his public key is a commitment to 0, the formula $\prod_{i=1}^{N} c_i^{\prod_{j=1}^{n} f_{j,i_j}} \cdot \prod_{k=0}^{n-1} c_{d_k}^{-x^k} = Com_{ck}(0; z_d)$ can be verified to pass, because

$$\prod_{k=0}^{n-1} c_{d_k}^{-x^k} = c_{d_0}^{-x^0} \cdot c_{d_1}^{-x^1} \cdot \ldots \cdot c_{d_{n-1}}^{-x^{n-1}} = (c_1^{p_{1,0}} \cdot c_2^{p_{2,0}} \cdot \ldots \cdot c_N^{p_{N,0}} \cdot Com_{ck}(0; \rho_0)) \cdot (c_1^{p_{1,1}} \cdot c_2^{p_{2,1}} \cdot \ldots \cdot c_N^{p_{N,1}} \cdot Com_{ck}(0; \rho_1))^{-x^1} \cdot \ldots \cdot$$

$$(c_1{}^{p_{1,n-1}}\cdot c_2{}^{p_{2,n-1}}\cdot\ldots\cdot c_N{}^{p_{N,n-1}}\cdot Com_{ck}(0;\rho_{n-1}))^{-x^{n-1}}.$$

Here $p_{i,k}$ is from the polynomial of $\prod_{j=1}^{n} f_j, i_j$: $p_i(x) = \delta_{il}x^n + \sum_{k=0}^{n-1} p_{i,k}x^k$, and $\prod_{i=0}^{N-1} p_i\,(x)$=$(\delta_{0l}x^n + \sum_{k=0}^{n-1} p_{0,k}x^k)\cdot(\delta_{1l}x^n + \sum_{k=0}^{n-1} p_{1,k}x^k)\ldots(\delta_{N-1l}x^n + \sum_{k=0}^{n-1} p_{N-1,k}x^k$); The formula

$$\prod_{i=1}^{N} p_i\,(x) = (p_{1,0}x^0\cdot p_{1,1}x^1\cdot\ldots\cdot p_{1,n-1}x^{n-1})\cdot\ldots\cdot (x^n+p_{l,0}x^0\cdot p_{l,1}x^1\cdot\ldots\cdot p_{l,n-1}x^{n-1})\cdot\ldots\cdot (p_{N,0}x^0\cdot p_{N,1}x^1\cdot\ldots\cdot p_{N,n-1}x^{n-1})$$

established only if $i = l$; We need to compute $\prod_{j=1}^{n} f_j, i_j$:

if i_j =1, $f_{j,1} = \delta_{1l_j}x + a_j$,otherwise $f_{j,0} = x - f_j = \delta_{0l_j}x - a_j$. So each f_j, i_j is a polynomial composed of $x, a_1, a_2,\ldots,a_n$, and the polynomial coefficients calculated in this way are related to $p_i(x)$: $\prod_{i=1}^{N} c_i^{\prod_{j=1}^{n} f_{j,i_j}} = c_1^{p_1(x)}\cdot c_2^{p_2(x)}\cdot\ldots\cdot c_N^{p_N(x)}$. So only when $i = l$, the formula $\prod_{k=0}^{n-1} c_{d_k}^{-x^k}$ and $\prod_{i=1}^{N} c_i^{\prod_{j=1}^{n} f_{j,i_j}}$ can cancel each other out.

And then only have the formula:

$$c_l^{x^n}\cdot Com_{ck}(0;\rho_0)\cdot Com_{ck}(0;\rho_1)^{-x^1}\cdot\ldots\cdot Com_{ck}(0;\rho_{n-1})^{-x^{n-1}}.$$

Due to

$$c_l^{x^n} = Com_{ck}(0;r)^{x^n},$$

the formula $c_l^{x^n}\cdot Com_{ck}(0;\rho_0)\cdot Com_{ck}(0;\rho_1)^{-x^1}\cdot\ldots\cdot Com_{ck}(0;\rho_{n-1})^{-x^{n-1}}$ is equal to $Com_{ck}(0; rx^n - \rho_0 - \rho_1{}^{-x^1} - \ldots - \rho_{n-1}{}^{-x^{n-1}}) = Com_{ck}(0;z_d)$. Therefore, when the signer honestly generates a RS, the signature can be verified successfully.

The verifier runs the algorithm $\mathcal{V}_2$. The validation process of equations $c_{l_j}^{x}c_{a_j} = Com_{ck}(f_j;z_{a_j})$ *and* $c_{l_j}^{x-f_j}c_{b_j} = Com_{ck}(0;z_{b_j})$ is consistent with that in $\mathcal{V}_1$. The validation process of equation $T^{\Sigma_{i=1}^{N}\prod_{j=1}^{n} f_{j,i_j}}\cdot\prod_{k=0}^{n-1} c_{d_k}^{-x^k}$ is also consistent with that of equation $\prod_{i=1}^{N} c_i^{\prod_{j=1}^{n} f_{j,i_j}}\cdot\prod_{k=0}^{n-1} c_{d_k}^{-x^k} = Com_{ck}(0;z_d)$, but due to $c_1 = c_2 = \ldots = c_N = T$, $\prod_{i=1}^{N} c_i^{\prod_{j=1}^{n} f_{j,i_j}}$ is written in the form of $T^{\Sigma_{i=0}^{N-1}\prod_{j=1}^{n} f_{j,i_j}}$.

The verifier runs the algorithm $\mathcal{V}_3$. The validation process of equations $c_{l_j}^{x}c_{a_j} = Com_{ck_1}(f_j;z_{a_j})$, $c_{l_j}^{x-f_j}c_{b_j} = Com_{ck_1}(0;z_{b_j})$ and $\prod_{i=1}^{N} c_i^{\prod_{j=1}^{n} f_{j,i_j}}\cdot\prod_{k=0}^{n-1} c_{d_k}^{-x^k} = Com_{ck_1}(0;z_d)$ is consistent with that in $\mathcal{V}_1$; If the signer has the witness u, the equations $c^x c_a = Com_{ck_0}(f;z_a)$ and $c^{x-f}c_b = Com_{ck_0}(0;z_b)$ can be verified successfully. Only if $m \in \{0,1\}$, the equation

$c^x c_a = Com_{ck_0}(m;r)^x\cdot Com_{ck_0}(a;s) = Com_{ck_0}(mx+a; rx+s) = Com_{ck_0}(f;z_a)$,

$c^{x-f}c_b = (c^x \setminus c^f)c_b = Com_{ck_0}(mx(1-m); r(x-f)+t)$.

If (1), (2), and (3) can all be verified, then the RS generated by an honest signer must be verified.

One-time Correctness. If the signer can calculate the one-time tag according to the following formulas and add the one-time tag of the valid RS to the one-time tag list during the verification stage, it can ensure the correctness of the one-time:

$$E \leftarrow H_2(event);$$

$$T \leftarrow E^{x_l}. \tag{6}$$

So the signer can only use his private key to sign once. When the private key is used again, the one-time tag already exists in the one-time tag list, and the algorithm OneTimeLink will output 0.

Revocation Correctness. If the signer uses the revocation authority's public key of the revocation authority to encrypt the signer's public key according to the ElGamal Encryption, then the revocation private key must be used to revoke the signer's identity according to the following algorithm, that is, restore the signer's identity:

$$vk_l = C_2 \setminus {C_1}^{x_{rev}} = {y_{rev}}^{u} y_l \setminus g^{u x_{rev}} = y_l. \tag{7}$$

4.3 Size Analysis

Our scheme constructs a RS with N users as

$$\sigma = (a_1, a_2, a_3, z_1, z_2, z_3, T, C) \tag{8}$$

where a_1, a_2, a_3 are the elements generated during the commitment stage of NIZK proof, with a quantity of 4*logN,* 4*logN*, and 4*logN*+2, respectively. z_1, z_2 and z_3 are the elements generated during the response stage to the challenge, with a quantity of 3*logN*+1, 3*logN*+1, and 3*logN*+4, respectively. Hence, the size of the RS is (21*logN*+8)λ where λ is the security parameter.

5 Security Analysis

5.1 Unforgeability

Theorem 1 (Unforgeability). *In the random oracle model, if the discrete logarithm (DL) problem is hard, then the scheme satisfies unforgeability.*

Proof. If we consider the presence of an adversary A within the security model, capable of undermining the unforgeability of the scheme, it becomes feasible to devise a simulator B aimed at addressing the DL problem. Given an instance of the problem (g, g^a) on a cyclic group ($\mathbb{G}_q$,g,q) as input, and B provides oracles to run the game defined in Definition 3.

B can provide the oracles as follows:

— $Random\ Oracle\ H_1$: B establishes a list L_1 to keep record of all queries and responses. If a query already exists in L_1, B will return the corresponding response.

Otherwise, B randomly picks $d \in Z_q$ and returns d, while also recording the query and d in L_1.

— $Random\ Oracle\ H_2$: B establishes a list L_2 to keep record of all queries and responses. If a query already exists in L_2, B will return the corresponding response. Otherwise, B randomly selects $r \in Z_q$ and returns g^r, while also recording the query and g^r in L_2.

— $Registering\ Oracle\ O_J$: When the adversary makes the first N queries, the oracle returns the public key, forming the set of public keys R=$\{vk_0,...,vk_{N-1}\}$ (B does not know the private keys). If the adversary continues to query, B executes the algorithm **KeyGen** to generate public and private key pairs and returns the generated public key.

— $Capturing\ Member\ Oracle\ O_C$: A queries with the vk_i which was an output of O_J. If $vk_i \in R$, B halts. Otherwise, B returns the sk_i corresponding to vk_i.

— $Signing\ Oracle\ O_S$: On input a signing query with a message M, an event description $event$, a set R of N public keys, a signer's public key vk and a revocation authority's public key vk_{rev}, if $vk \notin R$, B can directly generates the ring signature using the private key. Otherwise, B computes as follows:

(a) If $event$ has not been made, B queries H_2 with $event$ and gets $E = H_2(event)$, where $E = g^r$ and B knows r. Otherwise, B returns the corresponding response in the L_2. B computes $T = vk^r$.

(b) B randomly picks $u \in Z_q$ and computes $C_1 = g^u$, $C_2 = vk_{rev}{}^u vk$, and generates $C = \{C_1, C_2\}$.

(c) If $event$ has been made, B uses the corresponding response as the challenge value x. Otherwise, B queries H_1 to obtain the challenge value x. Then B randomly picks $f_1,\dots,z_d \leftarrow Z_q$, $c_{l_1},\dots c_{l_n}, c_{d_0},\dots c_{d_{n-1}} \leftarrow Com_{ck}(0)$. And B computes $c_{a_j} = c_{l_j}{}^{-x} Com_{ck}(f_j; z_{a_j})$, $c_{b_j} = c_{l_j}{}^{x-f} Com_{ck}(0; z_{b_j})$ for j =1,..., n, and $c_{d_k} = \prod_{i=0}^{N-1} y_i^{\prod_{j=1}^{n} f_{j,i_j}} \cdot \prod_{k=1}^{n-1} c_{d_k}^{-x^k} \cdot Com_{ck}(0; -z_d)$ where $y_i \in R$ and l is the secret index of the signer's public key in R; B randomly picks $f_1',\dots,z_d' \leftarrow Z_q$, $c_{l_1}',\dots c_{l_n}', c_{d_0}',\dots c_{d_{n-1}}' \leftarrow Com_{ck}(0)$. Then B computes $c_{a_j}' = c_{l_j}'{}^{-x} Com_{ck'}(f_j'; z_{a_j}')$, $c_{b_j}' = c_{l_j}'{}^{x-f} Com_{ck'}(0; z_{b_j}')$, $c_{d_k}' = \prod_{i=0}^{N-1} T^{\prod_{j=1}^{n} f_{j,i_j}}{}' \cdot \prod_{k=1}^{n-1} c_{d_k}'^{-x^k} \cdot Com_{ck'}(0; -z_d')$; Finally, B randomly picks $f, z_a, z_b \leftarrow Z_q$, and B computes $c_a = c^{-x} Com_{ck_0}(f; z_a)$, $c_b = c^{f-x} Com_{ck_0}(0; z_b)$ where $c = C_1$. B randomly picks $f_1'',\dots,z_d'' \leftarrow Z_q$, $c_{l_1}'',\dots c_{l_n}'', c_{d_0}'',\dots c_{d_{n-1}}'' \leftarrow Com_{ck_1}(0)$ and B computes $c_{a_j}'' = c_{l_j}''{}^{-x} Com_{ck_1}(f_j''; z_{a_j}'')$, $c_{b_j}'' = c_{l_j}''{}^{x-f} Com_{ck_1}(0; z_{b_j}'')$, $c_{d_k}'' = \prod_{i=0}^{N-1} c_i''{}^{\prod_{j=1}^{n} f_{j,i_j}}{}'' \cdot \prod_{k=1}^{n-1} c_{d_k}''^{-x^k} \cdot Com_{ck_1}(0; -z_d'')$, where $c_i'' = C_2/y_i (i \in [0, N-1])$.

(d) B returns the RS σ=$(a_1,a_2,a_3,z_1,z_2,z_3,T,C)$ where

$$a_1=\{c_{l_1},\dots,c_{l_n},c_{a_1},\dots,c_{a_n},c_{b_1},\dots,c_{b_n},c_{d_0},\dots,c_{d_{n-1}}\},$$
$$z_1=\{f_1,\dots,f_n,z_{a_1},\dots,z_{a_n},z_{b_1},\dots,z_{b_n},z_d\},$$
$$a_2=\{c_{l_1}',\dots,c_{l_n}',c_{a_1}',\dots,c_{a_n}',c_{b_1}',\dots,c_{b_n}',c_{d_0}',\dots,c_{d_{n-1}}'\},$$
$$z_2=\{f_1',\dots,f_n',z_{a_1}',\dots,z_{a_n}',z_{b_1}',\dots,z_{b_n}',z_d'\},$$

$$a_3=\{c''_{l_1},\ldots,c''_{l_n},c''_{a_1},\ldots,c''_{a_n},c''_{b_1},\ldots,c''_{b_n},c''_{d_0},\ldots,c''_{d_{n-1}}\},$$
$$z_3=\{f''_1,\ldots,f''_n,z''_{a_1},\ldots,z''_{a_n},z''_{b_1},\ldots,z''_{b_n},z''_d\}.$$

Due to the σ is valid, it is indistinguishable for A whether this is a simulated algorithm or a real algorithm.

(1) B sends the public parameters pp to A, and set $y_l=g^a$, $x_l=a$.

(2) A queries O_J for a set of public keys $R=\{y_0,\ldots,y_{N-1}\}$ and adaptively makes query to O_S.

(3) A starts to forge a RS. Assume A forges a RS $\sigma^{(0)}=(a_1{}^{(0)},a_2{}^{(0)},a_3{}^{(0)},z_1{}^{(0)},z_2{}^{(0)},z_3{}^{(0)},T^{(0)},C^{(0)})$ using the set $R'\subseteq R$ and R' contains N' public keys. B again drives A to obtain N RS through the same query for N different challenges and N+1 RS contain $f_1^{(0)},\ldots,z_d^{(0)},\ldots,f_1^{(n)},\ldots,z_d^{(n)}$. Due to the same query, the answers to $c_{l_1},\ldots,c_{d_{n-1}}$ in a_1 are all the same. Following the idea in paper [4], B can calculate the private key a according to the following formula:

$$y_l=g^a=\prod_{e=0}^{n}(c_l{}^{(x^{(e)})^n}\cdot\prod_{k=0}^{n-1}c_{*k}^{(x^{(e)})^k})^{\alpha_e}=Com_{ck}(0;\sum_{e=0}^{n}a'_e\,z_d^{(e)})$$

(4) B can compute $a=\sum_{e=0}^{n}a'_e\,z_d^{(e)}$, which means B can obtain the solution to the DL problem instance.

Assume ε is the probability that A can successfully forge a RS, according to the forking lemma [19], the successful rewind simulation is $\varepsilon/2^n$. B can solve DL problem with a non-negligible probability, which contradicts the original hypothesis, so this scheme satisfies unforgeability.

The definition of the DDH assumption in the paper [12] is as follows:

Definition 8 (A Different Decisional Diffie-Hellman (DDH) Assumption). *Let* $\mathbb{G}q$ *be a cyclic group where the order of* $|\mathbb{G}q|=q$. *On input uniformly random* $(r_0,r_1,r_2,r'_0,r'_1,r'_2)\in\mathbb{G}q^6$. *Assume that* $\alpha_0=g^{r_0}$, $\beta_0=g^{r_1}$, $\gamma_0=g^{r_2}$, $\alpha_1=g^{r'_0}$, $\beta_0=g^{r'_1}$ *and* $\gamma_0=g^{r'_0r'_1}$. *The adversary A takes a guess of* $b\leftarrow\{0,1\}$; $(\alpha,\beta,\gamma)=(\alpha_b,\beta_b,\gamma_b)$. *Define the probability of A guess being acccurate as* $Pr[B(\alpha,\beta,\gamma)=b]=\frac{1}{2}+\frac{1}{Q(\lambda)}$, *where* $Q(\lambda)$ *is a polynomial related to security parameter* λ.

5.2 Anonymity

Theorem 2 (Anonymity). *In the random oracle model, if the DDHP is hard, then the scheme satisfies anonymity.*

Proof. If we consider the presence of an adversary A within the security model, capable of undermining the anonymity of the scheme, it becomes feasible to devise a simulator B aimed at addressing the DDHP. Given an instance of the problem (α,β,γ) as input, and B provides oracles to run the game defined in Definition 4. Let y_l=α, $H_2(R')$=β and T=γ.

— B sends the public parameters pp to A.

— A queries O_J for a set of public keys $R=\{y_0,\ldots,y_{N-1}\}$ and adaptively makes query to O_S.

— A sends B a message M, an event description $event$, two public keys $y_0 \in R'$ and $y_1 \in R'$ where $R' \subseteq R$ and a revoke public key y_{rev}. B randomly picks $b \in \{0,1\}$ and computes a RS σ corresponding y_b:

(1) B randomly picks $u \in Z_q$ and computes $C_1 = g^u$, $C_2 = vk_{rev}{}^u vk$, and generates $C = \{C_1, C_2\}$.

(2) If $event$ has been made, B uses the corresponding response as the challenge value x. Otherwise, B queries H_1 to obtain the challenge value x. B randomly picks $f_1, \ldots, z_d \leftarrow Z_q$, $c_{l_1}, \ldots c_{l_n}, c_{d_0}, \ldots c_{d_{n-1}} \leftarrow Com_{ck}(0)$. And B computes $c_{a_j} = c_{l_j}{}^{-x} Com_{ck}(f_j; z_{a_j})$, $c_{b_j} = c_{l_j}{}^{x-f} Com_{ck}(0; z_{b_j})$ for j =1,..., n , and $c_{d_k} = \prod_{i=0}^{N-1} y_i^{\prod_{j=1}^{n} f_{j,i_j}} \cdot \prod_{k=1}^{n-1} c_{d_k}^{-x^k} \cdot Com_{ck}(0; -z_d)$ where $y_i \in R'$ and l is the secret index of the signer's public key in R' ; B randomly picks $f_1', \ldots, z_d' \leftarrow Z_q$, $c_{l_1}', \ldots c_{l_n}', c_{d_0}', \ldots c_{d_{n-1}}' \leftarrow Com_{ck}(0)$. Then B computes $c_{a_j}' = c_{l_j}'{}^{-x} Com_{ck'}(f_j'; z_{a_j}')$, $c_{b_j}' = c_{l_j}'{}^{x-f} Com_{ck'}(0; z_{b_j}')$, $c_{d_k}' = \prod_{i=0}^{N-1} c_i'^{\prod_{j=1}^{n} f_{j,i_j}'} \cdot \prod_{k=1}^{n-1} c_{d_k}'^{-x^k} \cdot Com_{ck'}(0; -z_d')$, where $c_1' = c_2' = , \ldots, c_N' = T = \gamma$; Finally, B randomly picks $f, z_a, z_b \leftarrow Z_q$, and B computes $c_a = c^{-x} Com_{ck_0}(f; z_a)$, $c_b = c^{f-x} Com_{ck_0}(0; z_b)$ where $c = C_1$. B randomly picks $f_1'', \ldots, z_d'' \leftarrow Z_q$, $c_{l_1}'', \ldots c_{l_n}'', c_{d_0}'', \ldots c_{d_{n-1}}'' \leftarrow Com_{ck_1}(0)$ and B computes $c_{a_j}'' = c_{l_j}''{}^{-x} Com_{ck_1}(f_j''; z_{a_j}'')$, $c_{b_j}'' = c_{l_j}''{}^{x-f} Com_{ck_1}(0; z_{b_j}'')$, $c_{d_k}'' = \prod_{i=0}^{N-1} c_i''^{\prod_{j=1}^{n} f_{j,i_j}''} \cdot \prod_{k=1}^{n-1} c_{d_k}''^{-x^k} \cdot Com_{ck_1}(0; -z_d'')$, where $c_i'' = C_2 / y_i (i \in [0, N-1])$.

(3) $\boldsymbol{B}$ returns the RS $\boldsymbol{\sigma = (a_1, a_2, a_3, z_1, z_2, z_3, \gamma, C)}$ where

$$a_1 = \{c_{l_1}, \ldots, c_{l_n}, c_{a_1}, \ldots, c_{a_n}, c_{b_1}, \ldots, c_{b_n}, c_{d_0}, \ldots, c_{d_{n-1}}\},$$

$$z_1 = \{f_1, \ldots, f_n, z_{a_1}, \ldots, z_{a_n}, z_{b_1}, \ldots, z_{b_n}, z_d\},$$

$$a_2 = \{c_{l_1}', \ldots, c_{l_n}', c_{a_1}', \ldots, c_{a_n}', c_{b_1}', \ldots, c_{b_n}', c_{d_0}', \ldots, c_{d_{n-1}}'\},$$

$$z_2 = \{f_1', \ldots, f_n', z_{a_1}', \ldots, z_{a_n}', z_{b_1}', \ldots, z_{b_n}', z_d'\},$$

$$a_3 = \{c_{l_1}'', \ldots, c_{l_n}'', c_{a_1}'', \ldots, c_{a_n}'', c_{b_1}'', \ldots, c_{b_n}'', c_{d_0}'', \ldots, c_{d_{n-1}}''\},$$

$$z_3 = \{f_1'', \ldots, f_n'', z_{a_1}'', \ldots, z_{a_n}'', z_{b_1}'', \ldots, z_{b_n}'', z_d''\}.$$

(4) A guesses the secret index of the signer. A returns $A(\sigma) = l'(l' \in \{0.1\})$. If $l'=l$, B returns 1. Otherwise B returns 0/1 with equal probability. The probability of A solving the DDHP can be shown that:

$$\frac{1}{2} + Pr[1|inst \in DDHP] - Pr[1|inst \notin DDHP] = \frac{1}{2} + Pr[l' = l|inst \in DDHP] + Pr[l' \neq l|inst \in DDHP] \cdot Pr[r = 1] - Pr[l' = l|inst \notin DDHP] - Pr[l' \neq l|inst \notin DDHP] \cdot Pr[r = 0]$$

$$= \frac{1}{2} + \frac{1}{2} + \frac{1}{Q(\lambda)} + (1 - (\frac{1}{2} + \frac{1}{Q(\lambda)})) \cdot \frac{1}{2} - \frac{1}{2} - (1 - \frac{1}{2}) \cdot \frac{1}{2}$$

$$= \frac{1}{2} + \frac{1}{2Q(\lambda)}$$

If A can guess the secret index of the real signer with a probability $\frac{1}{2}+\frac{1}{\mathrm{Q}(\lambda)}$, B can solve the *DDHP* with the probability $\frac{1}{2}+\frac{1}{2\mathrm{Q}(\lambda)}$, which contradicts the original hypothesis, so this scheme satisfies anonymity.

5.3 One-time

Theorem 3 (One-time). *In the random oracle model, if the discrete logarithm (**DL**) problem is hard, then the scheme satisfies one-time.*

Proof. If we consider the presence of an adversary A within the security model, capable of undermining the one-time of the scheme, it becomes feasible to devise a simulator B aimed at addressing the DL problem. Given an instance of the problem $(\mathbb{G}\mathrm{q},g,q)$ as input, and B provides oracles to run the game defined in Definition 5. Let y_l=α and x_l=a.

(1) B sends the public parameters pp to A.

(2) A queries O_J for a set of public keys R=$\{y_0,\ldots,y_{N-1}\}$ and adaptively makes query to O_S and O_C. In Theorem 1, B does not possess the private keys corresponding to the first N public keys. In order to enable A to satisfy the condition of possessing a private key a in Definition 4, it is allowed for B to possess one private key and send it to A when A requests the O_C.

(3) A constructs two RS $\sigma_0^{(0)}$ and $\sigma_1^{(0)}$ using the same private key x_l and sends them to B. And the $\sigma_0^{(0)}$ and $\sigma_1^{(0)}$ are valid. Assume that the one-time tags corresponding to $\sigma_0^{(0)}$ and $\sigma_1^{(0)}$ are respectively denoted as $T_1=\mathrm{E}^{x_l}$ and $T_2=\mathrm{E}^{x_l^{'}}$. Here, E is used by A to query the result of H_2 for $event$.

(4) For $\sigma_0^{(0)}$, B again drives A to obtain N RS $\sigma_0^{(1)},\ldots,\sigma_0^{(n)}$ through the same query for N different challenges. And then B can get the signer's private key a=$\sum_{e=0}^{n}\alpha_e z_d^{(e)}$; Similarly, for $\sigma_1^{(0)}$, B can drives A to obtain N $\sigma_1^{(1)},\ldots,\sigma_1^{(n)}$ RS and get the signer's private key a'=$\sum_{e=0}^{n}\alpha_e^{'} z_d^{'(e)}$.

As A only has one private key, the two private keys extracted by B are equal. Furthermore, because the same $event$ corresponds to the same result from H_2, the two one-time tags $T_1$and T_2 are identical. Therefore, once T_1 is added to list L, T_2 will not pass verification and the scheme is one-time. Additionally, during the aforementioned process, B extracts the private key and obtains the solution to the problem instance, solving the DL problem and contradicting the original hypothesis. Thus, this scheme satisfies the one-time.

5.4 Revocability

Theorem 4 (Revocability). *In the random oracle model, if the the scheme satisfies unforgeability, then the scheme satisfies revocability.*

Proof. B sends the public parameters pp to A. A queries O_J for a set of public keys R=$\{y_0,\ldots,y_{N-1}\}$ and adaptively makes query to O_S and O_C. A constructs a ring signature σ using x_l which is corresponding to $y_l(y_l\in R')$. A sends σ to B.

As the scheme satisfies unforgeability, ring signature σ is valid. Assuming B obtains the signer's public key $y_l' \neq y_l$ during the execution of algorithm $Revoke$, meaning A wins the game, then σ includes ciphertext $C=\{C_1, C_2\}$, where $C_1 = g^u$ and $C_2 \leftarrow y_{rev}{}^u y_l'$. In this case, A possesses the private key corresponding to public key y_l', but the σ is generated by using x_l. On the other hand, if A could possess another private key to win the game, B could also drive A to forge a ring signature to obtain the private key (as in Theorem 1), i.e., B could solve the DL problem. Therefore, the scheme satisfies revocability.

5.5 Non-Slanderability

Theorem 5 (Non-Slanderability). *In the random oracle model, if the discrete logarithm (DL) problem is hard, then the scheme satisfies non-slanderability.*

Proof. Assuming that there is an adversary A in the security model that can break the Non-Slanderability of the scheme, we can construct a simulator B to solve the DL problem. In the game defined in Definition 7, A can query O_C to get $x_i=\{x_0, x_1, ..., x_{j-1}, x_{j+1}, ..., x_{N-1}\}$, but A cannot obtain the private key x_j of the user they desire to slander. A sends B an event description $event$, a message M, a set R of public keys, a signer's public key y and a revocation authority's public key y_{rev}. B simulates a ring signature $\sigma(T = E^{x_j})$ and sends it to A. A can continue to adaptively query the oracle to obtain ring signatures. Then A starts to construct a ring signature σ^* which contains T^*. A sends σ^* to B.

B runs the algorithm $Verify$. Assume A can win the game, i.e., σ^* is valid and $T=E^{x_j}=E^{x_j'}=T^*$. Because the same $event$ corresponds to the same result from H_2, we can get the conclusion $x_j=x_j'$, i.e., A has the private key of user j, and this contradicts the original hypothesis. If A could possess the private key of user j, B could also drive A to forge a ring signature to obtain the private key, i.e., B could solve the DL problem. Therefore, the scheme satisfies non-slanderability.

6 Instantiation and Experiments

6.1 Comparison with Other Schemes

In this section, we compare our scheme and state of the art schemes, with respect to parameters such as size, one-time, and revocability, on a theoretical level. The solutions we choose to compare satisfy at least one of linkability and revocability.

Table 1. Comparison with different schemes.

Scheme	Linkability	Revocability	Size
[10]	√	×	$(2N+1)\lambda$
[11]	√	×	$(N+1)\lambda$
[12]	√	√	$(2N+4)\lambda$

Our scheme	√	√	$(21logN+8)\lambda$

We conducted theoretical comparisons of the schemes [10], [11], [12], and our scheme in terms of linkability, revocability, and the size of RS, as shown in Table 1. N represents the number of users in the ring, and λ is the security parameter. For the scheme [10] and our scheme, when $N \geq 128$, the RS size of our scheme is smaller than [10], and our scheme possesses revocability that scheme [10] does not have. For the scheme [11] and our scheme, when $N \geq 256$, the size of our scheme is smaller than [11], and our scheme possesses revocability that scheme [11] does not have. For the scheme [12] and our scheme, when $N \geq 128$, the size of our scheme is smaller.

When $N = 1024$, the RS size of our scheme is 89% less than the RS size of [12].

It can be seen that our scheme has significant advantages in terms of RS size, which helps reduce the communication overhead of blockchain transactions.

6.2 Scheme Instantiation and Experiments

Applying our RS scheme to blockchain transactions, the transaction process is as follows:

(1) The user (signer) applies for a private/public key pair from the registration system. The registration system runs **KeyGen** to generate a public key vk and a secret key sk for the user and returns (sk, vk) to the user.

(2) The user's public key is registered on the blockchain. The blockchain initializes a public key list L_1 and a one-time tags list L_2. When a new user registers, his public key is added to L_1.

(3) Before initiating a transaction, the user needs to obtain a collection of public keys (ring) from the blockchain and includes the user's public key in this ring. The user then executes the algorithm **Sign**, signing the transaction with sk and sending the generated RS to the blockchain.

(4) The verifier validates the signature by running **Verify**. If the signature is valid, then verifier checks for double-spending attacks by executing **OneTimeLink**. If double-spending behavior is detected, the transaction is halted; otherwise, the one-time tag in the signature is added to L_2 and the RS is added on the blockchain.

(5) In the case of transaction disputes or other situations, regulatory authorities can retrieve the RS from the blockchain and use their private key to recover the signer's identity, by running **Revoke**.

The process of a user making a transaction on the blockchain is shown in Fig. 7, during which the user uses a RS to protect the privacy of his identity.

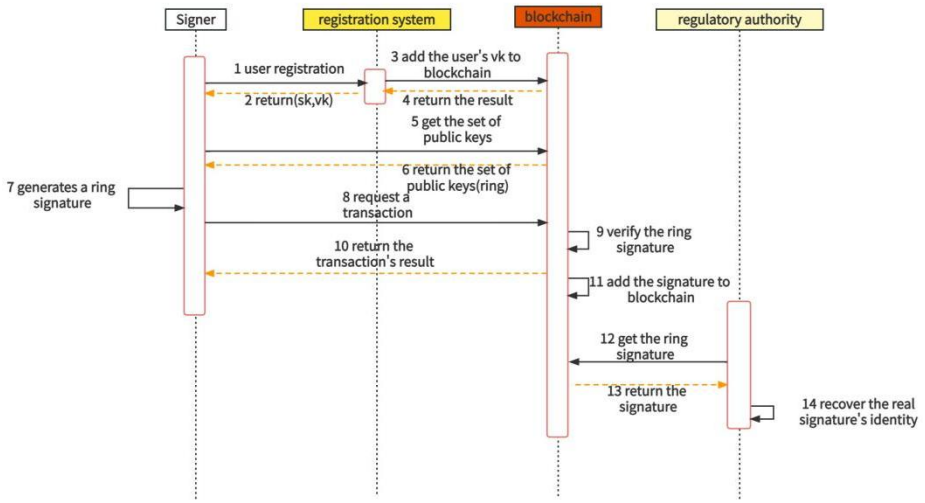

Fig. 7. The process of a blockchain transaction by using our scheme

An instantiation of our scheme and experiments. We use pedersen commitment based on Curve25519 in KUNLUN Library [23] to instantiate our RS scheme.
(1) System initialization, generating public parameters. Generates two generators of $\mathbb{G}_q$ is G = 046B17D1F2E12C4247F8BCE6E563A440F277037D812DEB33A0F4A13945D898C2964FE342E2FE1A7F9B8EE7EB4A7C0F9E162BCE33576B315ECECBB6406837BF51F5 and H = 042F9D894478F2DECAC4EC8799958733726541CA94CFBBE0CC6DAE59B89ECC08BB52609ED4A60EA6AD5DE66DB79FCB4F49059B1174ABC129BAE674279F77A1D953.
(2) Assume user's private key is x_l, we compute the public key $y_l = x_l * G$. And Generates the regulatory authority's public key and secret key pair (x_{rev}, y_{rev}).
(3) We randomly picks some users' public keys as the ring users of R.
(4) To generate a one-time tag, we take signer's private key and the $event$ as inputs to the function in KUNLUN Library to get T.
(5) To generate a $C=\{C_1, C_2\}$, we first pick a random number as u, and then computes C_1 = u*G and $C_2 = y_{rev}*u+y_l$.
(6) To generate a RS σ, we also need to compute a_1,a_2,a_3 and z_1,z_2,z_3 where a_1,a_2,a_3 are all commitments such as $c_{l_j} = Com_{ck'}(l_j; r_j) = l_j*G+r_j*H$. Then we use SM3 hash function to generate the challenge value x. And use x and x_l to compute z_1, z_2, z_3 where the calculations are all operations of BigInt. Then we can generate the RS σ.
(7) The validation algorithms are generated in a similar way. We define the functions $\mathcal{V}_1$, $\mathcal{V}_2$ and $\mathcal{V}_3$ using the parameters generated in (6).
(8) We define the **OneTimeLink** and **Revoke** algorithms where the **OneTimeLink** is to iterate through the one-time tags list and determine whether a once-time tag

exists in the list. We compare the real signer's public key use C and x_{rev}: $y = C_2 \setminus {C_1}^{x_{rev}}$.

By running experiments, it was verified that our RS scheme was correct and valid.

References

1. Song JW, Zhang DW, Han X, Du Y. Supervised Identity Privacy Protection Scheme in Bl ockchain. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/6517.htm.
2. Rivest R L, Shamir A, Tauman Y. How to leak a secret[C]//Advances in Cryptology—ASIACRYPT 2001: 7th International Conference on the Theory and Application of Cryptology and Information Security Gold Coast, Australia, December 9–13, 2001 Proceedings 7. Springer Berlin Heidelberg, 2001: 552-565.
3. Liu J K, Wei V K, Wong D S. Linkable spontaneous anonymous group signature for ad hoc groups[C]//Information Security and Privacy: 9th Australasian Conference, ACISP 2004, Sydney, Australia, July 13-15, 2004. Proceedings 9. Springer Berlin Heidelberg, 2004: 325-335.
4. Liu J K, Wong D S. Linkable ring signatures: Security models and new schemes[C]//Computational Science and Its Applications–ICCSA 2005: International Conference, Singapore, May 9-12, 2005, Proceedings, Part II 5. Springer Berlin Heidelberg, 2005: 614-623.
5. Tsang P P, Wei V K. Short linkable ring signatures for e-voting, e-cash and attestation[C]//International Conference on Information Security Practice and Experience. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005: 48-60.
6. Liu J K, Au M We selected two points on the curve H, Susilo W, et al. Linkable ring signature with unconditional anonymity[J]. IEEE Transactions on Knowledge and Data Engineering, 2013, 26(1): 157-165.
7. Fujisaki E, Suzuki K. Traceable ring signature[C]//International Workshop on Public Key Cryptography. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007: 181-200.
8. Au M H, Liu J K, Susilo W, et al. Secure ID-based linkable and revocable-iff-linked ring signature with constant-size construction[J]. Theoretical Computer Science, 2013, 469: 1-14.
9. Liu D Y W, Liu J K, Mu Y, et al. Revocable ring signature[J]. Journal of Computer Science and Technology, 2007, 22: 785-794.
10. Van Saberhagen N. CryptoNote v 2.0[J]. 2013.
11. Noether S, Mackenzie A. Ring confidential transactions[J]. Ledger, 2016, 1: 1-18.
12. Zhang X, Liu J K, Steinfeld R, et al. Revocable and linkable ring signature[C]//Information Security and Cryptology: 15th International Conference, Inscrypt 2019, Nanjing, China, December 6–8, 2019, Revised Selected Papers 15. Springer International Publishing, 2020: 3-27.
13. Sun S F, Au M H, Liu J K, et al. Ringct 2.0: A compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency monero[C]//Computer Security–ESORICS 2017: 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part II 22. Springer International Publishing, 2017: 456-474.
14. Chandran N, Groth J, Sahai A. Ring signatures of sub-linear size without random oracles[C]//International Colloquium on Automata, Languages, and Programming. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007: 423-434.

15. Jia Y, Sun S F, Zhang Y, et al. PBT: A New Privacy-Preserving Payment Protocol for Blockchain Transactions[J]. IEEE Transactions on Dependable and Secure Computing, 2020, 19(1): 647-662.
16. Groth J, Kohlweiss M. One-out-of-many proofs: Or how to leak a secret and spend a coin[C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015: 253-280.
17. Backes M, Döttling N, Hanzlik L, et al. Ring signatures: logarithmic-size, no setup—from standard assumptions[C]//Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part III 38. Springer International Publishing, 2019: 281-311.
18. Groth J, Kohlweiss M. One-out-of-many proofs: Or how to leak a secret and spend a coin[C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015: 253-280.
19. Herranz J, Sáez G. Forking lemmas for ring signature schemes[C]//International Conference on Cryptology in India. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003: 266-279.
20. Qin M J, Zhao Y L, Ma Z J. Practical constant-size ring signature[J]. Journal of Computer Science and Technology, 2018, 33: 533-541.
21. Yuen T H, Sun S, Liu J K, et al. Ringct 3.0 for blockchain confidential transaction: Shorter size and stronger security[C]//Financial Cryptography and Data Security: 24th International Conference, FC 2020, Kota Kinabalu, Malaysia, February 10–14, 2020 Revised Selected Papers 24. Springer International Publishing, 2020: 464-483.
22. Fujisaki E. Sub-linear size traceable ring signatures without random oracles[J]. IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences, 2012, 95(1): 151-166.
23. https://github.com/yuchen1024/Kunlun