Traceable and linkable ring signatures, traceable range proofs and applications on regulatable privacy-preserving blockchains

Wulu Li[1], Lei Chen[1], Xin Lai[1], Xiao Zhang[1], and Jiajun Xin[1]

Shenzhen Onething Technologies Co., Ltd., Shenzhen, China
liwulu@onething.net

Abstract. Privacy protection has been extensively studied in the current blockchain research field. As representations, Monero and Zerocash have realized completely anonymous and amount-hiding transactions. However, nonregulation can lead to abuse of privacy, which brings about serious risks of breaking laws and committing crimes. Therefore, it is crucial to study the privacy-preserving blockchain systems with regulatory functions. In this paper, we discuss the regulatory model (regulator behavior, user behavior) on the privacy-preserving blockchains from application scenarios and finally select unconditional regulation, static regulation, and self-participation of users as the core principles, which is currently the closest approach to the "decentralization" of regulatable privacy-preserving blockchains. On the basis of the established regulatory model, we first propose a traceable and linkable ring signature scheme (TLRS) by use of classic ring signatures, one-time signatures and zero-knowledge proofs, which realizes the unforgeability, anonymity, linkability, nonslanderability and traceability of users' identities. Moreover, we first propose traceable Borromean range proof (TBoRP) and traceable Bulletproofs range proof (TBuRP) by use of Fiat-Shamir transform and DH assumptions, which realize the completeness, soundness, zero-knowledge and traceability of the transaction amounts. We also apply the newly designed schemes and other zero-knowledge proof protocols to achieve richer regulatory functions.

Keywords: Regulatable blockchains · Privacy preserving · Decentralization · Traceable and linkable ring signatures · Traceable range proofs · Zero-knowledge proofs.

1 Introduction

Blockchain technology was first proposed by Nakamoto[21] in 2008. It is an application system that combines multiple underlying techniques including P2P networks, distributed data storage, network consensus protocols and cryptographic algorithms. It has features of open, transparentness, non-tamperability, traceability, and has various applications such as digital currency (including Bitcoin[21], Ethereum [9], Monero[28], Zerocash[25], etc.), credit deposit, anti-counterfeiting and medical health, etc. Blockchain technology has been widely concerned by government, financial departments, scientific academia, and has potential to lead the next industrial technology revolution. In June 2019, Facebook announced "Libra"[13]- an international blockchain-based digital currency, together with Visa, Mastercard, ebay, PayPal, etc. This indicates that the world is gradually entering a new era of digital currency based on blockchain technology.

Traditional blockchain-based digital currency represented by Bitcoin and Ethereum, realized decentralized transaction and public accounting. However, the data (transaction account and amount) on the chain is stored in plaintext and can be accessed by any user, making traditional blockchains restricted in various scenarios (such as salary, donation, bidding, taxation, etc.) as they provides no privacy protection. For the sake of privacy protection, scientific academia have proposed solutions such as Confidential Transaction[19], Mimblewimble[16], Monero[28] and Zerocash[25], etc. Among them, Monero successively followed the direction of Cryptonote[28], Ring-CT[22], Bulletproofs[8], in fact, Monero uses linkable ring signature scheme to hide identity of initiator, uses Diffie-Hellman key exchange scheme to hide identity of receiver and uses range proof (Borromean, Bulletproofs) to hide the amount of transaction. By contrast, Zerocash is deeply related with the zero-knowledge succinct non-interactive argument of knowledge (zk-SNARKs), which provides preimage proof of hash commitment, and therefor achieves fully privacy of identity and amount. Nevertheless, zk-SNARKs technique uses *common reference string* (CRS) of Gigabyte size and it is based on non-falsifiable assumption, that weakens its potential competitiveness against other cryptocurrencies. Recent projects, such as DERO[11] and ZETHER[7], all used Monero-type techniques (ring signatures, range proofs, Bulletproofs) to achieve privacy, in this paper, we also focus on modification and optimization for Monero system.

It should be noted that strong privacy-preserving blockchains have no regulatory functions, this feature brings about potential risks of illegal purposes such as illegal transactions, asset transfers, money laundering, fraud, etc. Violators can go unpunished with impunity, which seriously restricts the application prospects of current privacy-preserving blockchain systems, and it cannot be recognized by national regulatory agencies. At the same time, in the application scenario of privacy-preserving digital currency, regulators and policy-making institutions need to have a comprehensive understanding of the economic operation and development, and also need the regulatory function of the privacy currency system. Therefore, for the sake of legitimate, healthy and sustainable development, the privacy-preserving blockchain system must add the regulatory functions in the application scenario.

A possible solution for regulatable privacy-preserving blockchain is consortium blockchain system with trusted center, where CA can distribute or certify keys for users to achieve anonymous and regulatable transactions, by making use of cryptographic protocol such as group signature, zero-knowledge proof, etc. But in the scenario of group signature, malicious center may have the ability to forge signatures of users, or to authorize malicious users into the group, which is harmful to the entire system, so it is not a decentralized solution to achieve both anonymity and regulatability. To summarize, it is necessary to study blockchain systems that meet the characteristics of decentralization, have regulatory functions, protect users' privacy, and support future digital currency applications.

1.1 Related Works

In this section, we introduce the cryptographic concepts used by Monero, and some other works deeply related with Monero.

Ring Signatures Ring signature is a special type of signature scheme, in which signer can sign on behalf of a group chosen by himself, while retaining anonymous within the group. In ring signature, signer selects a list of public key $L_{PK} = \{PK_1, \cdots, PK_n\}$ as the ring elements, and uses his secret key SK_π to sign, verifier cannot determine signer's identity. Ring signature was first proposed by Rivest, Shamir and Tauman[24] in 2001, they constructed ring signature schemes based on RSA trapdoor permutation and Robin trapdoor function, in the random oracle model. In 2002, Abe *et al.*[1] proposed AOS ring signature, which simultaneously supported discrete logarithm (via Sigma protocol) and RSA trapdoor functions (via hash and sign), also in the random oracle model. In 2006, Bender *et al.*[6] introduced the first ring signature scheme in the standard model, by making use of pairing technique. In 2015, Maxwell *et al.*[20] gave Borromean signature scheme, which is a multi-ring signature based on AOS, reduce the signature size from $N + n$ to $N + 1$. It's worth emphasizing that the sizes of ring signatures in these schemes are linear with the number of ring elements.

In 2004, building from RSA accumulator, Dodis *et al.*[12] proposed a ring signature scheme with constant signature size in the random oracle model. In 2007, Chandran *et al.*[10] gave a standard model ring signature scheme with $O(\sqrt{n})$ signature size, from pairing technique and require CRS. In 2015, under the discrete logarithm assumption, Groth *et al.*[15] introduced a ring signature scheme with $O(\log n)$ signature size, in the random oracle model. In reality, the schemes mentioned above have shorter signature sizes than Borromean scheme asymptotically when n is sufficient large, but when n is small ($n \leq 64$), these schemes are less efficient as Borromean, and are not used in Monero system.

Linkable Ring Signatures Linkable ring signature is a variant of ring signature, in which the identity of the signer in a ring signature remains anonymous, but two ring signatures can be linked if they are signed by the same signer. Linkable ring signatures are suitable in many different practical applications such as privacy-preserving digital currency (Monero), e-Voting, cloud data storage security, etc. In Monero, linkability is used to check whether double spending happens. The first linkable ring signature scheme is proposed by Liu *et al.*[18]

in 2004, under discrete logarithm assumption, in the random oracle model. Later, Tsang *et al.*[27] and Au *et al.*[2] proposed accumulator-based linkable ring signatures with constant signature size. In 2013, Yuen *et al.*[29] gave a standard model linkable ring signature scheme with $O(\sqrt{n})$ signature size, from pairing technique. In 2014, Liu *et al.*[17] gave a linkable ring signature with unconditional anonymity, he also gave the formalized security model of linkable ring signature, which we will follow in this paper. In 2015, Back *et al.*[5] proposed a efficient linkable ring signature scheme LSAG, which shorten the signature size of [18]. In 2016, based on work of Fujisaki *et al.*[14], Noether *et al.*[22] gave a linkable multi-ring signature scheme MLSAG, which support transactions with multiple inputs, and was used by Monero. In 2017, Sun *et al.*[26] proposed Ring-CT 2.0, which is an accumulator-based linkable ring signature with constant signature size, but is less competitive when n is small.

Besides, there are other attempts such as identity-based linkable ring signature[4], certificate-based linkable ring signature[3], these schemes both achieve shorter signature sizes, but do not follow the principle of "decentralization", and will be omitted for brevity.

Traceable Ring Signatures Traceable ring signature is another variant of linkable ring signature, the identity of the signer in a ring signature remains anonymous to everyone, but when a signer signs two ring signatures with one secret key (illegal ring signatures), the signatures will be linked and the signer's identity will be opened. The first traceable ring signature is proposed by Fujisaki *et al.*[14] in 2007, based on discrete logarithm assumption, their scheme provide conditional traceability. To the best of our knowledge, there are no linkable ring signatures that provide unconditional traceability.

Range Proofs Range proof is a zero-knowledge proof to prove a committed hidden value lies within a certain range without revealing the value. The Pedersen-commitment-based range proofs are used in Monero system. In 2015, Neother *et al.*[22] gave the Borromean range proof, building from the Borromean ring signature[20], with proof size linear with the binary length of range. In 2018, Bünz *et al.*[8] introduced Bulletproofs, an efficient non-interactive zero-knowledge proof protocol with short proofs and without a trusted setup, the proof size is only logarithmic in the witness size and it is used in projects such as Monero, DERO, ZETHER. To the best of our knowledge, there are no traceable range proofs that provide regulatory function.

1.2 Our Contributions

In this paper, we first discuss and classify the regulatory models on the privacy-preserving blockchains to determine the regulatory model used in our constructions. Then we give the design method of the privacy-preserving blockchain system under the framework of the Monero. Specifically, we give the construction and security proof of the traceable and linkable ring signature (TLRS), give the construction and security proof of the traceable range proof, including traceable Borromean range proof (TBoRP) and traceable Bulletproofs range proof (TBuRP).

Regulatory Model Regulations on the privacy-preserving blockchain can be classified according to various classification methods. We will introduce them separately.

- In regard to regulation mode, it can be divided into *conditional regulation* and *unconditional regulation*: conditional regulation means that the regulator can only trace the identity of a malicious user (illegal ring signature, double spending) and cannot trace the identity of a normal user. Unconditional regulation means that the regulator can trace the identity of any ring signature user, as well as the transaction amount.
- In regard to participation mode of regulator, it can be divided into *dynamic regulation* and *static regulation*: dynamic regulation means that for every transaction in the chain, regulator is required to confirm the legality of the transaction, the computing power and bandwidth requirement of the regulator is high. Static regulation means that the regulator is not responsible for the legality verification of the transaction, and is not responsible for the work of accounting, packing blocks, etc. It only performs calculations when regulation is required, and the computing power and bandwidth requirements of the regulator are not high.
- In regard to participation mode of user, it can be divided into *self-participation* and *passive participation*: self-participation means that the users choose to join the chain and generate the public and private keys and addresses independently, other users (including the regulator) cannot obtain the users' private keys and cannot forge the users' signatures. Passive participation means that the users' public and private keys are distributed or certificated by the center, and users must trust the center unconditionally.

According to the design requirements of decentralization and strong regulation, we choose *unconditional regulation*, *static regulation*, and *self-participation* of users as the main principles of the regulatable privacy-preserving blockchains.

Traceable and Linkable Ring Signatures In this paper, we slightly modify the definition of traceable ring signature by introducing the concept of regulator who can trace users' identities for all transactions (including legal and illegal ring signatures). It's worth emphasizing that the new definition of traceable ring signature is deeply related with group signature as they share the property of unconditional traceability, but there are also differences between them:

- In group signatures, users' keys are distributed or certificated by a trusted center (group manager), which is a centralized setting. In our modified definition of traceable ring signature, users' keys are generated by themselves,

no one in the blockchain can forge their signature, slander users or double spending, even for malicious regulator, which is the closest approach to decentralization.
- In group signatures, management of group members (join or delete) is done by group manager, users cannot join the group independently, which also brings heavy computing load for group manager. In our modified definition of traceable ring signature, users can participate independently, without regulator's permission, which is also a decentralized setting.

Moreover, we obtain the traceable and linkable ring signature by combining our traceable ring signature with linkable ring signature, where the traceability follows the discussion above, and the linkability remains the same that two ring signatures can be linked if they are signed by the same signer. Informally, we give the first construction of traceable and linkable ring signature scheme (TLRS) by using classic ring signature, one-time signature and zero-knowledge proofs as components. We give a brief introduction of TLRS as follows:

1. The public parameter is $(\mathbb{G}, q, g_1, g_2, h = g_2^y)$, where g_1, g_2 are generators of elliptic curve, which are uniformly generated by system, y is the regulation trapdoor, generated by regulator.
2. User generates his (PK, SK) by use of public parameter, and add a tracing key TK together with the proof of TK's validity.
3. When signing, the user publishes a one-time public key OPK, uses ring private key RSK for ring signature σ_1, and uses one-time private key OSK for one-time signature σ_2.
4. The verifier first verifies the validity of TK's validity proof, then checks whether OPK appears in previous signatures to determine whether illegal signature (double spending) occurs. Then verifies the validity of classic ring signature σ_1 and one-time signature σ_2. If all pass, output 1 (*accepted*); otherwise, output 0 (*rejected*).
5. The regulator can trace the identity of signer by calculation with OPK and TK.

In the construction of TLRS, although public parameter with trapdoor is used for regulation, it has no influences to users' private keys. Under the discrete logarithm assumption, nobody else can steal the secret keys, nor forge TLRS signatures of users. Moreover, for malicious regulator (or malicious user who steals the trapdoor y), he can only break the anonymity of TLRS, but cannot generate illegal TLRS signatures (double spending), cannot slander other users (make other legal TLRS signatures illegal) nor break traceability (escaping from regulation), which is the closest approach to the "decentralization" requirement, while achieving higher security level.

Traceable Borromean Range Proofs Traceable range proof is a special variant of range proof, in which there is a regulator can trace the amount with trapdoors. The zero-knowledge property of traceable range proof only holds for users without possession of trapdoors. We give the construction of traceable range proofs for the first time, including traceable Borromean range proofs (TBoRP) and traceable Bulletproofs range proofs (TBuRP), we give a brief introduction of TBoRP below, and introduce TBuRP in the next subsection:

In Borromean range proofs used in Monero, the pedersen commitment of transaction amount a is $c = g^x h^a$, for a's binary expansion $a = a_0 + 2a_1 + \cdots + 2^{n-1}a_{n-1}$, the prover generates a ring of two elements for every bit (from 0 to $n-1$) and finally generate a ring signature for n rings, using Borromean ring signatures. We modify the Borromean range proofs by adding public parameter $(g, h = g^y)$ with trapdoor y generated by regulator, then for every bit $a_i, i = 0, \cdots, n-1$, the prover adds a tracing key TK_i with the proofs of all TK_i's validity, whose validity can be checked by arbitrary verifier. The verifier also checks the validity of Borromean ring signatures and the correctness of binary expansion. The regulator traces $a_i = 0$ *or* 1 for every $i = 0, \cdots, n-1$ by making use of trapdoor and TK_i, then compute the amount $a = a_0+2a_1+\cdots+2^{n-1}a_{n-1}$.

In our construction of TBoRP, regulator cannot compute x form $c = g^x h^a$, which partially protects privacy of users, and it is a balance between regulation and privacy protection.

Traceable Bulletproofs Range Proofs We modify the Bulletproofs to achieve regulatory function by adding trapdoors into the public parameters. Informally, assume n (bit length of range) is even ($n = 32$ in Monero), for different generators $\mathbf{g} = (g_0, \cdots, g_{n-1})$ generated independently by system, the regulator generates $y_0, \cdots, y_{n/2-1}$ as trapdoors, then computes $h_{2i} = g_{2i}^{y_i}, h_{2i+1} = g_{2i+1}^{y_i}$ for $i = 0, \cdots, n/2-1$, the regulator outputs $\mathbf{g} = (g_0, \cdots, g_{n-1}), \mathbf{h} = (h_0, \cdots, h_{n-1})$ as public parameters.

For a's commitment $c = h^x g^a$ and binary expansion $a = a_0 + 2a_1 + \cdots + 2^{n-1}a_{n-1}$, the prover computes $TK_0, \cdots, TK_{n-1}$ together with the proofs of all TK_i's validity, then generates the rest part of Bulletproofs. The verifier checks the validity of Bulletproofs as well as the validity of proofs of all TK_i, when all pass, output *accepted*, otherwise output *rejected*. The regulator can trace the amount a by calculating with trapdoors and TK_i.

In our construction of TBuRP, the soundness holds for malicious regulator, and the number of trapdoors and tracing keys can be modified for different requirement of size and time, which gives potential replacement for regulatable Bulletproofs-based blockchains.

More Applications The regulatable privacy-preserving blockchain system constructed in this paper can be applied in more application scenarios:

Multiple Regulator For the case of multiple regulators in the blockchain, such as several different kinds of digital currencies, each of which is regulated by each bank, how to calculate the total amount, how to transfer the money from bank A to bank B, this is a simple multiple regulator scenario. Our regulatable privacy-preserving blockchain system supports: each regulator uses his own trapdoor to generate public parameters, users can use zero-knowledge proofs such as switch proof to give the proof of privacy commitments migration among different regulators, and proof of the calculations $(+, -, \times, \div)$ of privacy data belonging to different regulators.

Auxiliary Computing The scheme of this paper supports the auxiliary computing function. The user's private data is stored in the chain (the regulator is A), the auxiliary computing party holds another private data (the regulator is B), by using zero-knowledge proofs, the auxiliary computing party can complete the addition, subtraction, multiplication and division calculations between the two private data, without the participation of other users, and the auxiliary computing party does not know the result of the calculations (which the regulator A knows). This auxiliary computing can be applied to the actual scenarios of international trades, calculation of interests, calculation of taxes, etc.

1.3 Paper Organization

The classification and discussion of the regulatory model has been completed in 1.2; in section 2 we give some preliminaries; in section 3 we give the construction and security proof of the traceable and linkable signature (TLRS); in section 4 we give the construction and safety proofs of the traceable Borromean range proof (TBoRP) and the traceable Bulletproofs range proof (TBuRP); The applications of the scheme in more scenarios are given in section 5; in section 6 we give the summary and future works.

2 Preliminaries

2.1 Notations

In this paper, in order to be consistent with Bulletproofs, we use multiplicative cyclic group $\mathbb{G}$ to represent elliptic group with group order $|\mathbb{G}| = q$, g is the generator of $\mathbb{G}$, group multiplication is $g_1 \cdot g_2$ and exponentiation is g^a. We use $H(\cdot)$ to represent hash function and *negl* to represent negligible functions.

2.2 Ring Signatures

Classic Ring Signatures Classic ring signature scheme usually consists of four algorithms: Setup, KeyGen, Sign, and Verify:

- $\mathsf{Par} \leftarrow \mathsf{Setup}(\lambda)$ is a probabilistic polynomial time (PPT) algorithm which, on input a security parameter λ, outputs the set of security parameters par which includes λ.
- $(PK_i, SK_i) \leftarrow \mathsf{KeyGen}(\mathsf{Par})$ is PPT algorithm which, on input security parameters par, outputs a private/public key pair (PK_i, SK_i).
- $\sigma \leftarrow \mathsf{Sign}(SK_\pi, \mu, L_{PK})$ is a ring signature algorithm which, on input user's secret key SK_π, a list of users' public keys $L_{PK} = \{PK_1, \cdots, PK_n\}$, where $PK_\pi \in L_{PK}$, and message μ, output a ring signature σ.

- $1/0 \leftarrow \mathsf{Verify}(\mu, \sigma, L_{PK})$ is a verify algorithm which, on input message μ, a list of users' public keys L_{PK} and ring signature σ, output 1 for accept or 0 for reject.

The security definition of ring signature contains *unforgeability* and *anonymity*. Before giving their definitions, we consider the following oracles which together model the ability of the adversaries in breaking the security of the schemes, in fact, the adversaries are allowed to query the four oracles below:

- $c \leftarrow \mathcal{RO}(a)$. *Random oracle*, on input a, random oracle returns a random value.
- $PK_i \leftarrow \mathcal{JO}(\perp)$. *Joining oracle*, on request, adds a new user to the system. It returns the public key PK_i of the new user.
- $SK_i \leftarrow \mathcal{CO}(PK_i)$. *Corruption oracle*, on input a public key PK_i that is a query output of $\mathcal{JO}$, returns the corresponding private key SK_i.
- $\sigma \leftarrow \mathcal{SO}(PK_\pi, \mu, L_{PK})$. *Signing oracle*, on input a list of users' public keys L_{PK}, the public key of the signer PK_π, and a message μ, returns a valid ring signature σ.

Definition 1 (*Unforgeability*) *Unforgeability for ring signature schemes is defined in the following game between the simulator $\mathcal{S}$ and the adversary $\mathcal{A}$, simulator $\mathcal{S}$ runs Setup to provide public parameters for $\mathcal{A}$, $\mathcal{A}$ is given access to oracles $\mathcal{RO}$, $\mathcal{JO}$, $\mathcal{CO}$ and $\mathcal{SO}$. $\mathcal{A}$ wins the game if he successfully forges a ring signature $(\sigma^*, L^*_{PK}, \mu^*)$ satisfying the following:*

1. $\mathsf{Verify}(\sigma^*, L^*_{PK}, \mu^*) = 1$.
2. *Every $PK_i \in L^*_{PK}$ is returned by $\mathcal{A}$ to $\mathcal{JO}$.*
3. *No $PK_i \in L^*_{PK}$ is queried by $\mathcal{A}$ to $\mathcal{CO}$.*
4. *(μ^*, L^*_{PK}) is not queried by $\mathcal{A}$ to $\mathcal{SO}$.*

We give the advantage of $\mathcal{A}$ in forge attack as follows:

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

A ring signature scheme is unforgeable if for any PPT adversary $\mathcal{A}$, $\mathrm{Adv}_{\mathcal{A}}^{forge} = negl$.

Definition 2 (*Anonymity*) *Anonymity for ring signature schemes is defined in the following game between the simulator $\mathcal{S}$ and the adversary $\mathcal{A}$, simulator $\mathcal{S}$ runs Setup to provide public parameters for $\mathcal{A}$, $\mathcal{A}$ is given access to oracles $\mathcal{RO}$, $\mathcal{JO}$ and $\mathcal{CO}$. $\mathcal{A}$ gives a set of public keys $L_{PK} = \{PK_1, \cdots, PK_n\}$, $\mathcal{S}$ randomly picks $\pi \in \{1, \cdots, n\}$ and computes $\sigma = \mathsf{Sign}(SK_\pi, \mu, L_{PK})$, where SK_π is a corresponding private key of PK_π and send σ to $\mathcal{A}$, then $\mathcal{A}$ output a guess $\pi^* \in \{1, \cdots, n\}$. $\mathcal{A}$ wins the game if he successfully guesses $\pi^* = \pi$.*

We give the advantage of $\mathcal{A}$ in anonymity attack as follows:

$$\mathrm{Adv}_{\mathcal{A}}^{anon} = |\Pr[\pi^* = \pi] - 1/n|.$$

A ring signature scheme is anonymous if for any PPT adversary $\mathcal{A}$, $\mathrm{Adv}_{\mathcal{A}}^{anon} = negl$.

We give the introduction of AOS ring signature[1] as an example:

- $\mathsf{Par} \leftarrow \mathsf{Setup}(\lambda)$: system chooses elliptic curve $\mathbb{G}$ and a generator g as the public parameter.
- $(PK_\pi, SK_\pi) \leftarrow \mathsf{KeyGen}(\mathsf{Par})$: according to the public parameter, user P_π chooses $x \in \mathbb{Z}_q^*$ uniformly at random, compute g^x and sets $(PK_\pi, SK_\pi) = (g^x, x)$.
- $\sigma \leftarrow \mathsf{Rsign}(SK_\pi, \mu, L_{PK})$: when user P_π generate a ring signature for message μ, he chooses other $n-1$ users' public keys, together with his own PK_π to obtain a set of public keys $L_{PK} = \{PK_1, \cdots, PK_n\}$, where $PK_\pi \in L_{PK}$, then he does as follows:
  1. P_π chooses $r_\pi \in \mathbb{Z}_q^*$ uniformly at random, then computes $c_{\pi+1} = H(g^{r_\pi}, L_{PK}, \mu)$;
  2. For $i = \pi+1, \cdots, n, 1, \cdots, \pi-1$, P_π chooses $z_i \in \mathbb{Z}_q^*$ uniformly at random and compute $c_{i+1} = H(g^{z_i}/(PK_i)^{c_i}, L_{PK}, \mu)$;
  3. P_π computes $z_\pi = r_\pi + xc_\pi$;
  4. Output the ring signature $\sigma = (c_1; z_1, \cdots, z_n)$.
- $1/0 \leftarrow \mathsf{Verify}(\mu, \sigma, L_{PK})$: for a ring signature (μ, L_{PK}, σ), the verifier computes and checks $c_{i+1} \stackrel{?}{=} H(g^{z_i}/(PK_i)^{c_i}, L_{PK}, \mu)$ for $i = 1, \cdots, n$, if all pass then outputs 1, otherwise outputs 0.

The AOS ring signature scheme is unforgeable and anonymous in the random oracle model. In 2015, Maxwell *et al.*[20] gave the Borromean signature scheme, which is a multi-ring signature based on AOS, reduce the signature size from $N + n$ to $N + 1$, where $N = mn$ with ring number n and m elements in each ring. We omit the detailed description of Borromean ring signature for brevity.

Linkable Ring Signatures Based on classic ring signatures, linkable ring signature has the function of linkability, that is, when two ring signatures are signed by the same signer, they are linked by the algorithm Link. We give the definition of Link below:

- $linked/unlinked \leftarrow \mathsf{Link}((\sigma, \mu, L_{PK}), (\sigma', \mu', L_{PK'}))$: verifier checks the two ring signatures are linked or not, output the result.

The security definition of linkable ring signature contains *unforgeability*, *anonymity*, *linkability* and *nonslanderability*. The *unforgeability* is the same as Definition 1, and the *anonymity* is slightly different from Definition 2 with additional requirements that all public keys in L_{PK} are returned by $\mathcal{A}$ to $\mathcal{JO}$ and all public keys in L_{PK} are not queried by $\mathcal{A}$ to $\mathcal{CO}$. In the rest of this paper, we use this modified definition of *anonymity* in TLRS and its security proof.

We give the definition of *linkability* and *nonslanderability* as follows:

Definition 3 (*Linkability*) *Linkability for linkable ring signature schemes is defined in the following game between the simulator $\mathcal{S}$ and the adversary $\mathcal{A}$, simulator $\mathcal{S}$ runs Setup to provide public parameters for $\mathcal{A}$, $\mathcal{A}$ is given access to oracles $\mathcal{RO}$, $\mathcal{JO}$, $\mathcal{CO}$ and $\mathcal{SO}$. $\mathcal{A}$ wins the game if he successfully forges k ring signatures $(\sigma_i, L^i_{PK}, \mu_i), i = 1, \cdots, k$, satisfying the following:*

1. *All σ_is are not returned by $\mathcal{A}$ to $\mathcal{SO}$.*
2. *All L^i_{PK} are returned by $\mathcal{A}$ to $\mathcal{JO}$.*
3. $\mathsf{Verify}(\sigma_i, L^i_{PK}, \mu_i) = 1, i = 1, \cdots, k.$
4. *$\mathcal{A}$ queried $\mathcal{CO}$ less than k times.*
5. $\mathsf{Link}((\sigma_i, L^i_{PK}, \mu_i), (\sigma_j, L^j_{PK}, \mu_j)) = unlinked$ *for* $i, j \in \{1, \cdots, k\}$ *and* $i \neq j$.

We give the advantage of $\mathcal{A}$ in linkability attack as follows:

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

A linkable ring signature scheme is linkable if for any PPT adversary $\mathcal{A}$, $\mathrm{Adv}^{link}_{\mathcal{A}} = negl$.

The *nonslanderability* of a linkable ring signature scheme is that $\mathcal{A}$ cannot slander other honest users by generating a signature linked with signatures of honest users. In the following we give the definition:

Definition 4 (*Nonslanderability*) *Nonslanderability for linkable ring signature schemes is defined in the following game between the simulator $\mathcal{S}$ and the adversary $\mathcal{A}$, simulator $\mathcal{S}$ runs* Setup *to provide public parameters for $\mathcal{A}$, $\mathcal{A}$ is given access to oracles $\mathcal{RO}$, $\mathcal{JO}$, $\mathcal{CO}$ and $\mathcal{SO}$. $\mathcal{A}$ gives a list of public keys L_{PK}, a message μ and a public key $PK_\pi \in L_{PK}$ to $\mathcal{S}$, $\mathcal{S}$ returns the corresponding signature $\sigma \leftarrow \mathsf{Sign}(SK_\pi, L_{PK}, \mu)$ to $\mathcal{A}$. $\mathcal{A}$ wins the game if he successfully outputs a ring signature $(\sigma^*, L^*_{PK}, \mu^*)$, satisfying the following:*

1. $\mathsf{Verify}(\sigma^*, L^*_{PK}, \mu^*) = 1.$
2. *PK_π is not queried by $\mathcal{A}$ to $\mathcal{CO}$.*
3. *PK_π is not queried by $\mathcal{A}$ as input to $\mathcal{SO}$.*
4. $\mathsf{Link}((\sigma, L_{PK}, \mu), (\sigma^*, L^*_{PK}, \mu^*)) = linked.$

We give the advantage of $\mathcal{A}$ in slandering attack as follows:

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

A linkable ring signature scheme is nonslanderable if for any PPT adversary $\mathcal{A}$, $\mathrm{Adv}^{slander}_{\mathcal{A}} = negl$.

According to [17], linkability and nonslanderability imply unforgeability:

Lemma 5 ([17]) *If a linkable ring signature scheme is linkable and nonslanderable, then it is unforgeable.*

2.3 Zero-knowledge proofs

Zero-knowledge proof system is a proof system (P, V) in which a prover proves to the verifier that he has a certain knowledge but does not reveal the knowledge itself. The formal definition is that given language L and relation R, for $\forall x \in L$, there exists a witness w such that $(x, w) \in R$, to prove $x \in L$ without disclosing w. The transcript between prover and verifier is $\langle P(x, w), V(x)\rangle$, the proof is correct (or wrong) if $\langle P(x, w), V(x)\rangle = 1(or\ 0)$. The security notions of zero-proof system contains *completeness*, *soundness*, *zero-knowledge*:

Definition 6 (*Completeness*) *(P, V) has* ***completeness*** *for any non-uniform polynomial time adversary $\mathcal{A}$,*

$$\mathsf{Pr}[(x, w) \leftarrow \mathcal{A}(1^\lambda) : (x, w) \notin R \text{ or } \langle P(x, w), V(x) \rangle = 1] = 1 - negl.$$

When the probability equals 1, then (P, V) has perfect completeness.

Definition 7 (*Soundness*) *(P, V) has* ***soundness*** *for any non-uniform polynomial time adversary $\mathcal{A}$ and $x \notin L$,*

$$\mathsf{Pr}[(x, s) \leftarrow \mathcal{A}(1^\lambda) : \langle P(x, w), V(x) \rangle = 1] = negl.$$

In Σ protocols with Fiat-Shamir transformation in the random oracle model, we use the notion of ***special soundness****, that is, for a 3-round interactive proof protocol, if a non-uniform polynomial time adversary $\mathcal{A}$ can generate 2 valid proofs $(x, c, e_1, s_1), (x, c, e_2, s_2)$, then there exists a extraction algorithm Ext which can extract a witness $(x, w) \in R$, where c represents the commitment, $e_i s$ are challenges and $s_i s$ are responses.*

Definition 8 (*Zero-knowledge*) *(P, V) has* ***perfect*** *(or* ***computational****)* ***zero-knowledge****, for any non-uniform polynomial time (or PPT) adversary $\mathcal{A}$,*

$$\mathsf{Pr}[(x, w) \leftarrow \mathcal{A}(1^\lambda); tr \leftarrow \langle P(x, w), V(x, \rho) \rangle : (x, w) \in R \text{ and } \mathcal{A}(tr) = 1] = (\text{or} \approx_c)$$

$$\mathsf{Pr}[(x, w) \leftarrow \mathcal{A}(1^\lambda); tr \leftarrow S(x, \rho) : (x, w) \in R \text{ and } \mathcal{A}(tr) = 1].$$

In Fiat-Shamir-based protocol, the randomness of ρ is from the output of hash function, it is said to be ***public coin*** *and the protocol is* ***honest-verifier zero-knowledge****.*

Pedersen Commitment Pedersen commitment[23] was proposed in 1991, for elliptic curve $(\mathbb{G}, q = |\mathbb{G}|, g, h)$, where g is a generator of $\mathbb{G}$, h is a random element with discrete logarithm unknown to anyone.

Definition 9 (*Pedersen commitment*) *The pedersen commitment for a is $c = g^x h^a$, where $x \in \mathbb{Z}_q^*$ is a blinding element. Under the hardness of discrete logarithm, Pedersen commitment has the following properties:*

- *(Hiding) Any (computational unbounded) adversary $\mathcal{A}$ cannot distinguish $c = g^x h^a$ from $c' = g^{x'} h^{a'}$.*
- *(Binding) Any PPT adversary $\mathcal{A}$ cannot generate another secret a' binding with $c = g^x h^a = g^{x'} h^{a'}$.*
- *(Homomorphic) Given $c_1 = g^x h^a, c_2 = g^y h^b$, then $c_1 \cdot c_2 = g^{x+y} h^{a+b}$ is a new commitment for $a + b$.*

We use pedersen commitment to construct commitment proof and switch proof.

Commitment Proof For pedersen commitment $c = g^x h^a$, we can prove the knowledge of x, a without revealing them by commitment proof:

1. Prover generates $r_1, r_2 \in \mathbb{Z}_q^*$ uniformly at random, computes $e = H(g^{r_1} h^{r_2})$.
2. Prover computes $z_1 = r_1 + ex, z_2 = r_2 + ea$, output proof $\pi(c) = (e, z_1, z_2)$.
3. Verifier checks $e \stackrel{?}{=} H(g^{z_1} h^{z_2} / c^e)$.

Commitment proof is an extension of Schnorr signature with perfect completeness, special soundness and honest verifier zero-knowledge.

Switch Proof For two pedersen commitments $c_1 = g^x h_1^a$ and $c_2 = g^y h_2^a$, we can prove the equality of hidden value ($a = a$) by switch proof (h_1 switch to h_2):

1. Prover generates $r_1, r_2, r \in \mathbb{Z}_q^*$ uniformly at random, computes $e = H(g^{r_1} h_1^r, g^{r_2} h_2^r)$.
2. Prover computes $z_1 = r_1 + ex, z_2 = r_2 + ey, z_3 = r + ea$, output proof $\pi(c_1, c_2) = (e, z_1, z_2, z_3)$.
3. Verifier checks $e \stackrel{?}{=} H(g^{z_1} h_1^{z_3} / c_1^e, g^{z_2} h_2^{z_3} / c_2^e)$.

Specially, when $x = y$, then prover samples $r_1 = r_2$, then $z_1 = z_2$, which shorten the proof size. The switch proof also has perfect completeness, special soundness and honest verifier zero-knowledge.

2.4 Range proofs

Borromean Range Proof Borromean range proof is used in Monero to provide the validity proof of transaction amount ($a \in [0, 2^n - 1]$) by making use of Borromean ring signature and pedersen commitment:

- Setup: System chooses public parameters $(\mathbb{G}, q, g, h)$.
- Gen:
  1. According to the public parameters, amount $a \in [0, 2^n - 1]$, prover computes the commitment $c = g^x h^a$;
  2. Prover computes the binary expansion $a = a_0 + \cdots + 2^{n-1} a_{n-1}, a_i = 0, 1$ for $i = 0, \cdots, n-1$;
  3. Prover samples $x_0, \cdots, x_{n-1}$ uniformly, satisfying $x_0 + \cdots + x_{n-1} = x$;
  4. For every $i = 0, \cdots, n-1$, prover computes $c_i = g^{x_i} h^{2^i a_i}, c_i' = g^{x_i} h^{2^i a_i - 2^i}$, prover outputs $L_{PK}^i = (c_i, c_i')$.
- Prove:
  1. Prover generates n sets of PKs by $L = \{L_{PK}^0, \cdots, L_{PK}^{n-1}\}$;
  2. Prover runs Borromean ring signature, outputs $\sigma = \mathsf{Sign}(L, SK, c)$, where $SK = (x_0, \cdots, x_{n-1})$.
- Verify: For $i = 0, \cdots, n-1$, verifier checks as follows:
  1. Check $\prod c_i \stackrel{?}{=} c$;
  2. Check $c_i / c_i' \stackrel{?}{=} h^{2^i}$;
  3. Check the validity of Borromean ring signature σ, if all pass, then outputs 1, otherwise 0.

Bulletproofs Range Proof Bulletproofs, proposed by Bünz *et al.*[8] in 2018, is an efficient zero-knowledge with $O(\log n)$ proof size, and is widely used in inner-product argument, range proof and proof for arithmetic circuits. The Bulletproofs range proof also uses pedersen commitment:

– Setup: System chooses public parameters $(\mathbb{G}, q, g, h, \mathbf{g}, \mathbf{h})$, where $\mathbf{g} = (g_0, \cdots, g_{n-1}), \mathbf{h} = (h_0, \cdots, h_{n-1}) \in \mathbb{G}^n$.
– Gen:
  1. According to the public parameters, amount $a \in [0, 2^n - 1]$, prover computes the commitment $c = h^x g^a$;
  2. Prover computes the binary expansion $a = a_0 + \cdots + 2^{n-1} a_{n-1}, a_i = 0, 1$, sets $\mathbf{a}_L = (a_0, \cdots, a_{n-1})$;
  3. Prover computes $\mathbf{a}_R = \mathbf{a}_L - \mathbf{1}^n = (a_0 - 1, \cdots, a_{n-1} - 1)$;
  4. Prover samples $\alpha \in \mathbb{Z}_q$ uniformly at random, computes
$$A = h^\alpha \mathbf{g}^{\mathbf{a}_L} \mathbf{h}^{\mathbf{a}_R} = h^\alpha g_1^{a_1} \cdots g_{n-1}^{a_{n-1}} h_1^{a_1 - 1} \cdots h_{n-1}^{a_{n-1}-1};$$
  5. Prover samples $\mathbf{s}_L, \mathbf{s}_R \in \mathbb{Z}_q^n, \rho \in \mathbb{Z}_q$ uniformly at random, computes $S = h^\rho \mathbf{g}^{\mathbf{s}_L} \mathbf{h}^{\mathbf{s}_R}$.
– Prove:
  1. Prover sends c, A, S to verifier;
  2. Verifier samples $y, z \in \mathbb{Z}_q$ uniformly at random, and sends them to prover;
  3. Prover computes T_1, T_2 and sends them to verifier;
  4. Verifier samples $x \in \mathbb{Z}_q$ uniformly at random, and sends it to prover;
  5. Prover computes $\tau_x, \mu, t, \mathbf{l}, \mathbf{r}$ and sends them to verifier.

The computation of $T_1, T_2, \tau_x, \mu, t, \mathbf{l}, \mathbf{r}$ as well as the verification algorithm is omitted for brevity, please refer to [8] for detailed description.

3 Traceable and Linkable Ring Signatures

In this section, we give the construction and security proof of traceable and linkable ring signature scheme (TLRS), the TLRS achieves unforgeability, anonymity, linkability, nonslanderability and traceability. In the scenario of blockchain, unforgeability works for security of uses' accounts, anonymity works for anonymity of uses' identities, linkability and nonslanderability works for prevention of double-spending (actively or passively), traceability works for unconditional regulation of identity.

3.1 Construction

In our construction of TLRS, we use classic ring signature (for single ring, AOS as example) or multi-ring signature (Borromean as example) as the ring signature component of TLRS, use ECDSA or Schnorr signature as the one-time signature component of TLRS. Actually, these schemes are anonymous and unforgeable, which will be used in the security proof. Moveover, Trace algorithm is added into TLRS to realize the regulation of identity, which makes TLRS more suitable in the regulatable privacy-preserving blockchains.

We give the introduction of TLRS in the following (single ring as example):

- $\mathsf{Par} \leftarrow \mathsf{Setup}(\lambda)$: system chooses elliptic curve $\mathbb{G}$ and generators $g_1, g_2 \in \mathbb{G}$ independently, the regulator generates $y \in \mathbb{Z}_q^*$ as the trapdoor, computes $h = g_2^y$ with h's order as large as possible, system outputs $(\mathbb{G}, q, g_1, g_2, h)$ as the public parameters, in which the regulator dose not know the relation between g_1 and h.
- $(PK, SK) \leftarrow \mathsf{KeyGen}(\mathsf{Par})$:
  1. According to the public parameters $(\mathbb{G}, q, g_1, g_2, h)$, user Alice samples $x, a \in \mathbb{Z}_q^*$, computes $RPK = g_1^x h^a, TK = g_2^a, OPK = h^a$;
  2. Alice gives the validity proof $\pi(RPK, TK)$, that is, she gives the switch proof that $RPK = g_1^x h^a$ and $RPK \cdot TK = g_1^x (g_2 h)^a$ share the same exponents $(x = x, a = a)$ with respect to (g_1, h) and $(g_1, g_2 h)$;
  3. Alice outputs $PK = (RPK, TK, \pi(RPK, TK))$, and retains $SK = (RSK = x, OSK = a)$.
- $\sigma \leftarrow \mathsf{Sign}(SK_\pi, \mu, L_{PK})$:
  1. For a message μ, Alice chooses another $n-1$ users, together with her own public key, to generate a list of public keys $L_{PK} = \{PK_1, \cdots, PK_n\}$, where Alice's $PK = PK_\pi \in L_{PK}$;
  2. Alice outputs $OPK = h^{a_\pi}$, computes
  $$L_{RPK} = \{RPK_1 \cdot OPK^{-1}, \cdots, RPK_n \cdot OPK^{-1}\}$$
  $$= \{g_1^{x_1} h^{a_1 - a_\pi}, \cdots, g_1^{x_n} h^{a_n - a_\pi}\};$$
  3. Alice runs ring signature $\sigma_1 \leftarrow \mathsf{Rsign}(RSK, \mu, L_{RPK}, OPK)$ using L_{RPK} and $RSK = x_\pi$, outputs σ_1;
  4. Alice runs one-time signature $\sigma_2 \leftarrow \mathsf{Osign}(OSK, \sigma_1, OPK)$ using $OPK = h^{a_\pi}$ and $OSK = a_\pi$;
  5. Alice outputs $\sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, OPK)$.
- $1/0 \leftarrow \mathsf{Verify}(\sigma_1, \sigma_2, \mu, L_{PK}, OPK)$:
  1. Verifier checks the validity of $\pi(RPK_i, TK_i)$ for every $1, \cdots, n$;
  2. Verifier checks $L_{RPK} \stackrel{?}{=} \{RPK_1 \cdot OPK^{-1}, \cdots, RPK_n \cdot OPK^{-1}\}$;
  3. Verifier checks the validity of ring signature σ_1 and signature σ_2;
  4. If all pass then outputs 1, otherwise outputs 0.
- *linked*/*unlinked* $\leftarrow \mathsf{Link}(\sigma, \sigma')$: For two TLRS signatures $\sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, OPK)$ and $\sigma' = (\sigma_1', \sigma_2', \mu', L_{PK}', OPK')$, if $OPK = OPK'$ then verifier outputs *linked*, otherwise outputs *unlinked*.
- $\pi^* \leftarrow \mathsf{Trace}(\sigma, y)$: For $\sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, OPK)$, the regulator extracts $TK_1, \cdots, TK_n$ from L_{PK}, computes TK_i^y for $i = 1, \cdots, n$, outputs the smallest π^* such that $OPK = TK_{\pi^*}^y$ as the trace result, otherwise outputs $\perp$.

In the scenario of privacy-preserving blockchains, using UTXO model, the $TK = g_2^a$ can be regarded as the UTXO public key generated in the last transaction, and user will publish the UTXO public key TK, $RPK, \pi(RPK, TK)$ after receiving the money. The validity of $\pi(RPK, TK)$ can be checked by designated nodes, which not happens during the transaction and can reduce the cost of transaction verification. OPK can be regarded as the Key-image of UTXO, and Link is used for detection of double-spending. Trace is used for regulation of identity by regulator, which brings the regulatory function to the blockchains.

Correctness

Theorem 10 (Correctness of TLRS) *For an honest user Alice in TLRS, she can complete the ring signature and one-time signature, and regulator can trace her identity.*

Proof. In TLRS, for Alice's public key $PK = PK_\pi = (g_1^x h^a, g_2^a, \pi(g_1^x h^a, g_2^a))$, then Alice will output $OPK = h^a$ with $L_{RPK} = \{g_1^{x_1} h^{a_1 - a_\pi}, \cdots, g_1^{x_n} h^{a_n - a_\pi}\}$. Since $g_1^{x_\pi} h^{a_\pi - a_\pi} = g_1^{x_\pi}$, then Alice can use $RSK = x$ to generate ring signature. For $OPK = h^a$, then Alice can use $OSK = a$ to generate one-time signature (take h as basis element).

For regulator, he can compute $TK^y = g_2^{ay} = h^a = OPK$ and then outputs $Trace(\sigma, y) = \pi$. □

3.2 Security proofs

Security Model On the basis of security definitions for linkable ring signature, a PPT adversary $\mathcal{A}$ is given access to oracles $\mathcal{RO}$, $\mathcal{JO}$, $\mathcal{CO}$ and $\mathcal{SO}$, and security of TLRS contains unforgeability, anonymity, linkability, nonslanderability and traceability. Considering the existence of regulator, who can trace the identities of users, so the anonymity only holds for someone not possesses the trapdoor. Moreover, the unforgeability, linkability, nonslanderability remain the same as linkable ring signature, even for malicious regulator (or adversary who steals the trapdoor), he cannot forge signature of other users, break the linkability and nonslanderability of TLRS, which means that malicious regulator cannot spend money of other users, double spend or slander other users.

Traceability enables regulator with ability to trace users' identities, for any PPT adversary $\mathcal{A}$ with possession of trapdoor, he cannot escape from regulation. We give the formal definition of traceability as follows:

Definition 11 (*Traceability*) *Traceability for traceable and linkable ring signature schemes (TLRS) is defined in the following game between the simulator $\mathcal{S}$ and the adversary $\mathcal{A}$, simulator $\mathcal{S}$ runs* Setup *to provide public parameters for $\mathcal{A}$, $\mathcal{A}$ is given access to oracles $\mathcal{RO}$, $\mathcal{JO}$, $\mathcal{CO}$. $\mathcal{A}$ generates a list of public keys $L_{PK} = \{PK_1, \cdots, PK_n\}$, $\mathcal{A}$ wins the game if he successfully generates a TLRS signature (σ, L_{PK}, μ) using $PK_\pi \in L_{PK}$, satisfying the following:*

1. *Verify*$(\sigma, L_{PK}, \mu) = 1$.

2. $TK_i \neq TK_j$ *for* $1 \leq i < j \leq n$.
3. $\mathsf{Trace}(\sigma, y) \neq \pi$ *or* $\mathsf{Trace}(\sigma, y) = \perp$.

We give the advantage of $\mathcal{A}$ in traceability attack as follows:

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

A linkable ring signature scheme is traceable if for any PPT adversary $\mathcal{A}$, $\mathrm{Adv}_{\mathcal{A}}^{trace} = negl$.

Anonymity

Theorem 12 (Anonymity) *TLRS is anonymous for any PPT adversary $\mathcal{A}$ (without possession of trapdoor).*

Proof. Assume $\mathcal{A}$ is playing the game with $\mathcal{S}$ in Definition 2, $\mathcal{A}$ he generates a message μ and a list of public keys $L_{PK} = \{PK_1, \cdots, PK_n\}$, where $PK_i = (RPK_i = g_1^{x_i} h^{a_i}, TK_i = g_2^{a_i}, \pi(RPK_i, TK_i)$, all PK_is are returned by $\mathcal{JO}$, and $\mathcal{S}$ knows all $SK = (x_i, a_i)$.

Consider the following games between $\mathcal{S}$ and $\mathcal{A}$:

- **Game 0**. $\mathcal{S}$ samples $\pi \in \{1, \cdots, n\}$ uniformly, publishes $OPK = h^{a_\pi}$ and $L_{RPK} = \{g_1^{x_1} h^{a_1 - a_\pi}, \cdots, g_1^{x_n} h^{a_n - a_\pi}\}$, generates ring signature $\sigma_1 = \mathsf{Rsign}(RSK, \mu, L_{RPK}, OPK)$ and one-time signature $\sigma_2 = \mathsf{Osign}$ (OSK, σ_1, OPK), outputs $\sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, OPK)$ to $\mathcal{A}$. When $\mathcal{A}$ receives σ, he gives a guess $\pi^* \in \{1, \cdots, n\}$.
- **Game 1**. $\mathcal{S}$ uniformly samples $\pi \in \{1, \cdots, n\}, r \in \mathbb{Z}_q^*$, publishes $OPK = h^r$ and $L_{RPK} = \{g_1^{x_1} h^{a_1 - r}, \cdots, g_1^{x_n} h^{a_n - r}\}$, generates ring signature $\sigma_1 = \mathsf{Rsign}(\mu, L_{RPK}, OPK)$ by programming the random oracle, then generates one-time signature $\sigma_2 = \mathsf{Osign}(OSK, \sigma_1, OPK)$, outputs $\sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, OPK)$ to $\mathcal{A}$. When $\mathcal{A}$ receives σ, he gives a guess $\pi^* \in \{1, \cdots, n\}$.

In the two games above, Game 0 is the actual game between $\mathcal{S}$ and $\mathcal{A}$, and Game 1 is the simulated game in the random oracle model. In game 1, r is uniformly sampled by $\mathcal{S}$, which is statistical independent from the L_{PK}, then $\mathsf{Pr}_{\mathcal{A}}[\pi^* = \pi] = 1/n$.

Then we only need to prove that game 0 and game 1 are computational indistinguishable. If fact, the differences between the two games are generation of OPK and L_{RPK}. According to DH assumption, $(g_2, h, g_2^{a_\pi}, h^{a_\pi})$ and $(g_2, h, g_2^{a_\pi}, h^r)$ are computational indistinguishable, then $\mathcal{A}$ cannot distinguish h^{a_π} (in game 0) from h^r (in game 1). Meanwhile, due to the hiding items $x_1, \cdots, x_n$, $\mathcal{A}$ cannot distinguish $\{g_1^{x_1} h^{a_1 - a_\pi}, \cdots, g_1^{x_n} h^{a_n - a_\pi}\}$ from $\{g_1^{x_1} h^{a_1 - r}, \cdots, g_1^{x_n} h^{a_n - r}\}$, then we know game 0 and game 1 are computational indistinguishable, which finishes the anonymity proof of TLRS. □

Linkability

Theorem 13 (Linkability) *TLRS is linkable for any PPT adversary $\mathcal{A}$, including malicious regulator.*

Proof. For a PPT adversary $\mathcal{A}$ with possession of the trapdoor y, but does not know the relation between g_1 and $(g_2, h = g_2^y)$, when $\mathcal{A}$ finished the link game with $\mathcal{S}$ in Definition 3, we assume that $\mathcal{A}$ wins the link game with nonnegligible advantage δ, that is, $\mathcal{A}$ returned k TLRS signatures $\sigma_i = (\sigma_1^i, \sigma_2^i, \mu_i, L_{PK}^i, OPK^i)$, $i = 1, \cdots, k$ (σ_1^is are ring signatures, σ_2^is are one-time signatures), which satisfy the following:

1. All $\sigma_i, i = 1, \cdots, k$ are not returned by $\mathcal{SO}$.
2. All public keys from $L_{PK}^i, i = 1, \cdots, k$ are returned by $\mathcal{JO}$.
3. $\mathsf{Verify}(\sigma_i, L_{PK}^i, \mu_i) = 1$ for $i = 1, \cdots, k$.
4. $\mathcal{A}$ queried $\mathcal{CO}$ less than k times.
5. $\mathsf{Link}((\sigma_i, L_{PK}^i, \mu_i), (\sigma_j, L_{PK}^j, \mu_j)) = \mathit{unlinked}$ for $i \neq j \in \{1, \cdots, k\}$.

We first prove a statement that, for a list of users' public keys $L_{PK} = \{PK_1, \cdots, PK_n\}$ returned by $\mathcal{JO}$, where $PK_i = (g_1^{x_i} h^{a_i}, g_2^{a_i}, \pi(g_1^{x_i} h^{a_i}, g_2^{a_i}))$, any PPT adversary $\mathcal{A}$ generates a valid TLRS signature $\sigma \not\leftarrow \mathcal{SO}$ if and only if he quires the $\mathcal{CO}$ at least once, except for negligible probability $\epsilon_0 = negl(n)$.

- $\Rightarrow$. If $\mathcal{A}$ gets $SK = (x_i, a_i)$ from $\mathcal{CO}$, and then $\mathcal{A}$ can run the TLRS signature scheme to generate a valid signature $\sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, OPK)$.
- $\Leftarrow$. Assume $\mathcal{A}$ did not query the $\mathcal{CO}$ and $\mathcal{SO}$ for $L_{PK} = \{PK_1, \cdots, PK_n\}$ and finished the TLRS signature over $L_{PK} = \{PK_1, \cdots, PK_n\}$ with non-negligible probability δ_1. We first prove that $\mathcal{A}$ does not know any of the secret keys in L_{PK}. Actually, under the hardness of discrete logarithm, $\mathcal{A}$ cannot compute a_i from $TK = g_2^{a_i}, i = 1, \cdots, n$, then the probability of $\mathcal{A}$ obtaining any of (x_i, a_i) is $\epsilon_1 = negl(n)$.
  Next, according to the assumption that $\mathcal{A}$ generates a valid signature $\sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, OPK)$, then he must have finished the one-time signature σ_2. Since the one-time signature scheme achieves unforgeability, then $\mathcal{A}$ knows $OSK = b$ except for negligible probability $\epsilon_2 = negl(n)$, we get that $L_{RPK} = \{g_1^{x_1} h^{a_1 - b}, \cdots, g_1^{x_n} h^{a_n - b}\}$ and $\mathcal{A}$ finished the ring signature σ_1 with L_{RPK}. According to the unforgeability of ring signature, we get that $\mathcal{A}$ knows at least one of the correspond z satisfying $g_1^{x_j} h^{a_j - b} = g_1^z$ for $j \in \{1, \cdots, n\}$, except for negligible probability $\epsilon_3 = negl(n)$, which means $\mathcal{A}$ got a solution for $g_1^{x_j} h^{a_j - b} = g_1^z$ with nonnegligible probability $\delta_1 - \epsilon_1 - \epsilon_2 - \epsilon_3$, this contradicts with the hardness of discrete logarithm problems. Then we get that $\mathcal{A}$ generates a valid TLRS signature $\sigma \not\leftarrow \mathcal{SO}$ if and only if he quires the $\mathcal{CO}$ at least once, except for negligible probability.

According to the fourth requirement that the number of times of $\mathcal{A}$ querying $\mathcal{CO}$ is $\leq k-1$, and $\mathcal{A}$ returned k valid TLRS signatures $\sigma_i = (\sigma_1^i, \sigma_2^i, \mu_i, L_{PK}^i, OPK^i)$, $i = 1, \cdots, k$, then we know there are two TLRS signatures from the same query of $\mathcal{CO}$, saying $SK = (z, b)$ from $PK = (g_1^z h^b, g_2^b, \pi(g_1^z h^b, g_2^b))$, and $\mathcal{A}$ finished two unlinked valid TLRS signature, then there is at least one $OPK = h^{b'} \neq h^b$ from the two TLRS signatures (otherwise they will be linked). We have $L_{RPK} = \{g_1^{x_1} h^{a_1 - b'}, \cdots, g_1^{x_n} h^{a_n - b'}\}$, since $\exists j \in \{1, \cdots, n\}$ $s.t.$ $(x_j, a_j) = (z, b)$, then we have $g_1^{x_j} h^{a_j - b'} = g_1^z h^{b-b'}$ with $b \neq b'$ and $\mathcal{A}$ cannot compute x $s.t.$ $g_1^x = g_1^z h^{b-b'}$ under the hardness assumption of discrete logarithm problem, except for negligible probability $\epsilon = negl(n)$, then we have that $\mathcal{A}$ successfully forge a ring signature for $L_{RPK} = \{g_1^{x_1} h^{a_1 - b'}, \cdots, g_1^{x_n} h^{a_n - b'}\}$ with nonnegligible probability $\delta - \epsilon - k\epsilon_0$, which contradicts to the unforgeability of ring signature, then we finish the linkability proof of TLRS. □

Nonslanderability

Theorem 14 (Nonslanderability) *TLRS is nonslanderable for any PPT adversary $\mathcal{A}$, including malicious regulator.*

Proof. For a PPT adversary $\mathcal{A}$ with possession of the trapdoor y, but does not know the relation between g_1 and $(g_2, h = g_2^y)$, when $\mathcal{A}$ finished the slandering game with $\mathcal{S}$ in Definition 4, $\mathcal{A}$ gave a list of public keys L_{PK}, a message μ and a public key $PK_\pi \in L_{PK}$ to $\mathcal{S}$, $\mathcal{S}$ returns the corresponding signature $\sigma \leftarrow \mathsf{Sign}(SK_\pi, L_{PK}, \mu)$ to $\mathcal{A}$. We assume that $\mathcal{A}$ wins the slandering game with nonnegligible advantage δ, that is, $\mathcal{A}$ successfully outputs a ring signature $\sigma^* = (\sigma_1^*, \sigma_2^*, \mu^*, L_{PK}^*, OPK^*)$, satisfying the following:

1. $\mathsf{Verify}(\sigma^*, L_{PK}^*, \mu^*) = 1$.
2. PK_π is not queried by $\mathcal{A}$ to $\mathcal{CO}$.
3. PK_π is not queried by $\mathcal{A}$ as input to $\mathcal{SO}$.
4. $\mathsf{Link}((\sigma, L_{PK}, \mu), (\sigma^*, L_{PK}^*, \mu^*)) = linked$.

From the definition of Link, we know that $OPK^* = OPK = h^{a_\pi}$, since PK_π was not queried by $\mathcal{A}$ to $\mathcal{CO}$ and $\mathcal{SO}$, then $\mathcal{A}$ does not know $OSK = a_\pi$ except for negligible probability $\epsilon = negl(n)$ under the hardness of discrete logarithm problems. Then we know $\mathcal{A}$ forged one-time signature σ_2^* with nonnegligible advantage $\delta - \epsilon$, which contradicts to the unforgeability of one-time signature, then we finish the nonslanderability proof of TLRS. □

According to lemma 5, we get the unforgeability of TLRS:

Corollary 15 (Unforgeability) *TLRS is unforgeable for any PPT adversary $\mathcal{A}$, including malicious regulator.*

Traceability

Theorem 16 (Traceability) *TLRS is traceable for any PPT adversary $\mathcal{A}$, including malicious regulator.*

Proof. For a PPT adversary $\mathcal{A}$ with possession of the trapdoor y, but does not know the relation between g_1 and $(g_2, h = g_2^y)$, when $\mathcal{A}$ finished the tracing game with $\mathcal{S}$ in Definition 11, $\mathcal{A}$ generates a list of public keys $L_{PK} = \{PK_1, \cdots, PK_n\}$, we assume that $\mathcal{A}$ wins the tracing game with nonnegligible advantage δ, that is, $\mathcal{A}$ generates a TLRS signature $\sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, OPK)$ using $PK_\pi \in L_{PK}$, satisfying the following:

1. $\mathsf{Verify}(\sigma, L_{PK}, \mu) = 1$.
2. $TK_i \neq TK_j$ for $1 \leq i < j \leq n$.
3. $\mathsf{Trace}(\sigma, y) \neq \pi$ or $\mathsf{Trace}(\sigma, y) = \perp$.

Since σ_2 is a valid one-time signature, then $OPK = h^b$ and $\mathcal{A}$ knows $OSK = b$ except for negligible probability ϵ_1 under the unforgeability of one-time signature, we have $L_{RPK} = \{g_1^{x_1} h^{a_1 - b}, \cdots, g_1^{x_n} h^{a_n - b}\}$. According to $\mathcal{A}$ signed σ_1 with PK_π, if $b \neq a_\pi$, then the ring signing public key is $g_1^{x_\pi} h^{a_\pi - b} = g_1^z$, and $\mathcal{A}$ knows $RSK = z$ except for negligible probability ϵ_2 under the unforgeability of ring signature, then $\mathcal{A}$ successfully generates a relation $g_1^{x_\pi - z} h^{a_\pi - b} = 1$ with nonnegligible advantage $\delta - \epsilon_1 - \epsilon_2$, which contradicted to the hardness of discrete logarithm problem, then we have $b = a_\pi$.

From the requirement that $\mathsf{Trace}(\sigma, y) \neq \pi$, we know $TK_\pi^y \neq OPK = h^{a_\pi}$, we can set $TK_\pi = g_1^s g_2^t$ without loss of generality, then we get the validity proof $\pi(RPK_\pi, TK_\pi) = \pi(g_1^{x_\pi} h^{a_\pi}, g_1^s g_2^t)$, which is the switch proof between $g_1^{x_\pi} h^{a_\pi}$ and $g_1^{x_\pi} h^{a_\pi} \cdot g_1^s g_2^t = g_1^{x_\pi + s} g_2^{y a_\pi + t}$ with respect to (g_1, h) and $(g_1, g_2 h)$. Assume $\pi(RPK_\pi, TK_\pi) = (e, z_1, z_2)$, then we have

$$e = H(g_1^{z_1} h^{z_2} / (g_1^{x_\pi} h^{a_\pi})^e, g_1^{z_1} (g_2 h)^{z_2} / (g_1^{x_\pi + s} g_2^{y a_\pi + t})^e).$$

According to the Fiat-Shamir-based switch proof in section 2.3, we know that e is computed by $e \leftarrow H(g_1^{r_1} g_2^{r_2}, g_1^{r_3} g_2^{r_4})$, then we have

$$r_1 = z_1 - e x_\pi,\ r_2 = y(z_2 - e a_\pi),\ r_3 = z_1 - e x_\pi - es,\ r_4 = (y+1) z_2 - e(y a_\pi + t).$$

Then $es = r_1 - r_3$, $e(t - a_\pi) = (1 + y^{-1}) r_2 - r_4$, if $s \neq 0$ then $e = (r_1 - r_3) s^{-1}$, which means $\mathcal{A}$ computes e before he runs the hash function (query the random oracle), this happens with negligible probability, meanwhile, if $t \neq a_\pi$ then $e = ((1 + y^{-1}) r_2 - r_4)/(t - a_\pi)$, also happens with negligible probability. Finally we get $s = 0, t = a_\pi$ and $TK_\pi = g_2^{a_\pi}$, which means $TK_\pi^y = OPK$ and $\mathsf{Trace}(\sigma, y) = \pi$, this contradicts to the assumptions before, then we finish the traceability proof of TLRS. $\square$

4 Traceable Range Proofs

In this section we give the constructions and security proofs of two traceable range proof schemes: traceable Borromean range proof (TBoRP) and traceable Bulletproofs range proof (TBuRP). Similar to TLRS, provers generate their proofs by using parameters with trapdoors generated by regulator, which helps regulator recover the hidden amounts bitwise. Meanwhile, the validity of traceable range proof is publicly verified, which is suitable for the applications on blockchains.

4.1 Security Model

On the basis of security definitions for zero-knowledge proofs, the security of traceable range proof contains completeness, soundness, zero-knowledge and traceability. Considering the existence of regulator, who can trace the amounts of transactions, zero-knowledge only holds for someone not possesses the trapdoor. Moreover, the completeness remain the same as range proof, for any adversary $\mathcal{A}$. Soundness of TBoRP and TBuRP are different, for TBoRP, soundness holds for any PPT adversary $\mathcal{A}$ without possession of trapdoors; for TBuRP, soundness holds for any PPT adversary $\mathcal{A}$.

Traceability enables regulator with ability to trace amounts of transactions, for any PPT adversary $\mathcal{A}$ without possession of trapdoors, he cannot escape from regulation. We give the formal definition of traceability as follows:

Definition 17 (*Traceability*) *Traceability for traceable range proof is defined in the following game between the simulator $\mathcal{S}$ and the adversary $\mathcal{A}$, simulator $\mathcal{S}$ runs Setup to provide public parameters for $\mathcal{A}$, $\mathcal{A}$ is given access to oracles $\mathcal{RO}$. $\mathcal{A}$ generates a commitment c for a hidden value a and the range proof $\pi(c)$, $\mathcal{A}$ wins the game if:*

1. $\mathsf{Verify}(c, \pi(c)) = 1$.
2. $\mathsf{Trace}(\pi(c), trapdoors) \neq a$.

We give the advantage of $\mathcal{A}$ in traceability attack as follows:

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

A traceable range proof is traceable if for any PPT adversary $\mathcal{A}$ without possession of trapdoors, $\mathrm{Adv}_{\mathcal{A}}^{trace} = negl$.

4.2 Traceable Borromean Range proof

Construction In the construction of TBoRP, similar to Borromean range proof, we use pedersen commitment and bit expansion of amount, then add tracing keys bitwise into the proof, together with the validity proofs of tracing keys. The regulator can use the trapdoor and tracing keys to recover the hidden amount bitwise.

We give the introduction of TBoRP in the following:

- $\mathsf{Par} \leftarrow \mathsf{Setup}(\lambda)$: system chooses elliptic curve $\mathbb{G}$ and generators $g \in \mathbb{G}$, the regulator generates $y \in \mathbb{Z}_q^*$ as the trapdoor, computes $h = g^y$ with h's order as large as possible, system outputs $(\mathbb{G}, q, g, h)$ as the public parameters.
- $(L_{PK}, SK, c, \{TK_i, \pi(c_i, TK_i)\}_{i=0,\cdots,n-1}) \leftarrow \mathsf{Gen}(\mathsf{Par}, a)$:
  1. According to the public parameters $(\mathbb{G}, q, g, h)$ and amount $a \in [0, 2^n - 1]$, prover Alice samples $x \in \mathbb{Z}_q^*$ uniformly, computes $c = g^x h^a$ as the commitment;
  2. Alice computes the binary expansion $a = a_0 + \cdots + 2^{n-1}a_{n-1}, a_i = 0, 1$ for $i = 0, \cdots, n-1$, samples $x_0, \cdots, x_{n-1}$ uniformly, satisfying $x_0 + \cdots + x_{n-1} = x$;

3. For every $i = 0, \cdots, n-1$, Alice computes $c_i = g^{x_i} h^{2^i a_i}, c_i' = g^{x_i} h^{2^i a_i - 2^i}$, outputs $L_{PK}^i = (c_i, c_i')$;
4. For every $i = 0, \cdots, n-1$, Alice computes $TK_i = h^{x_i - 2^i a_i}$ and gives the TK_i's validity proof $\pi(c_i, TK_i)$ that TK_i is a power of h and $TK_i \cdot c_i = (gh)^{x_i}$ is a power of gh;
5. Alice outputs $L_{PK} = \{L_{PK}^0, \cdots, L_{PK}^{n-1}\}$, c, $\{TK_i, \pi(c_i, TK_i)\}_{i=0,\cdots,n-1}$ and retains $SK = (x_0, \cdots, x_{n-1})$, a.

– $\sigma \leftarrow \mathsf{Prove}(SK, c, L_{PK})$: Alice runs the Borromean ring signature for $L_{PK} = \{(c_0, c_0'), \cdots, (c_{n-1}, c_{n-1}')\}$, outputs $\sigma \leftarrow \mathsf{Rsign}(SK, c, L_{PK})$.
– $1/0 \leftarrow \mathsf{Verify}(\sigma, c, L_{PK}, \{TK_i, \pi(c_i, TK_i)\}_{i=0,\cdots,n-1})$:
  1. For every $i = 0, \cdots, n-1$, verifier checks the validity of $\pi(c_i, TK_i)$;
  2. Verifier checks $\prod c_i \stackrel{?}{=} c$;
  3. For every $i = 0, \cdots, n-1$, verifier checks $c_i / c_i' \stackrel{?}{=} h^{2^i}$;
  4. Verifier checks the validity of Borromean ring signature σ, if all pass then outputs 1, otherwise outputs 0.
– $a^* \leftarrow \mathsf{Trace}(\sigma, L_{PK}, y, \{TK_i\}_{i=0,\cdots,n-1})$:
  1. For every $i = 0, \cdots, n-1$, regulator computes c_i^y or $(c_i')^y$;
  2. For every $i = 0, \cdots, n-1$, if $c_i^y = TK_i$ then outputs $a_i^* = 1$, if $(c_i')^y = TK_i \cdot h^{2^i}$ then outputs $a_i^* = 0$;
  3. Regulator outputs $a^* = a_0^* + \cdots + 2^{n-1} a_{n-1}^*$.

Correctness

Theorem 18 (Correctness of TBoRP) *For an honest user Alice in TBoRP, she can complete the Borromean ring signature, and the regulator can trace her amount.*

Proof. According to the binary expansion $a = a_0 + \cdots + 2^{n-1} a_{n-1}$ of a, we know there is only one element in $L_{PK}^i = (c_i = g^{x_i} h^{2^i a_i}, c_i' = g^{x_i} h^{2^i a_i - 2^i})$, which is a power of g known by Alice, then Alice can use the secret keys for $i = 0, \cdots, n-1$ to finish the Borromean ring signature for $L_{PK} = \{L_{PK}^0, \cdots, L_{PK}^{n-1}\}$. Besides, we know that $\prod c_i = c$ and $c_i / c_i' = h^{2^i}$ from the generation algorithms. When $a_i = 0$, we know $c_i = g^{x_i} h^{2^i a_i} = g^{x_i}$, $TK_i = h^{x_i} = c_i^y$, when $a_i = 1$, we know $c_i' = g^{x_i}$, $TK_i \cdot h^{2^i} = h^{x_i} = (c_i')^y$, then we get the correctness of TBoRP. □

Proof of Zero-knowledge

Theorem 19 (Zero-knowledge of TBoRP) *TBoRP is computational zero-knowledge for any PPT adversary $\mathcal{A}$ (without possession of trapdoor).*

Proof. For every $i = 0, \cdots, n-1$, we consider the impact that TK_i being added into the system, and prove that (c_i, TK_i) does not reveal any knowledge of $a_i = 0$ or 1. Formally, we prove for $c_i = g^{x_i} h^{2^i a_i}, c_i' = g^{x_i} h^{2^i a_i - 2^i}$ with $c_i / c_i' = h^{2^i}$ is a constant, any PPT adversary $\mathcal{A}$ cannot distinguish $(c_i, TK_i) = (g^{x_i}, h^{x_i})$ (when $a_i = 0$) from $(c_i, TK_i) = (g^{x_i} h^{2^i}, h^{x_i - 2^i})$ (when $a_i = 1$).

Actually, we know that (g, h, g^{x_i}, h^{x_i}) and (g, h, g^{x_i}, r) are computational indistinguishable for uniformly generated $x_i \in \mathbb{Z}_q^*$, under the DH assumption. For g is a generator of $\mathbb{G}$, the distribution of (g, h, g^{x_i}, r) and (g, h, r', r) are identical. Let constant $u = h^{2^i}$, we know that the distribution of (g, h, r', r) and $(g, h, r'u, ru^{-1})$ are identical. Again from the DH assumption, we know $(g, h, r'u, ru^{-1})$ and $(g, h, g^{x_i}u, h^{x_i}u^{-1})$ are computational indistinguishable. Then we have the relations between the following distributions:

$$(g, h, g^{x_i}, h^{x_i}) \approx_c (g, h, r', r) = (g, h, r'u, ru^{-1}) \approx_c (g, h, g^{x_i}u, h^{x_i}u^{-1}).$$

Where g, h, u are constants, r, r', x_i are uniform random variables.

Since $(g^{x_i}, h^{x_i}) = (c_i, TK_i)_{a_i=0}$ and $(g^{x_i}u, h^{x_i}u^{-1}) = (g^{x_i}h^{2^i}, h^{x_i - 2^i}) = (c_i, TK_i)_{a_i=1}$, we know they are computational indistinguishable for any PP-T adversary $\mathcal{A}$, for every $i = 0, \cdots, n-1$, then we finish the zero-knowledge proof of TBoRP. □

Traceability

Theorem 20 (Traceability of TBoRP) *TBoRP is traceable for any PPT adversary $\mathcal{A}$ (without possession of trapdoor).*

Proof. For a PPT adversary $\mathcal{A}$ without possession of the trapdoor y, when $\mathcal{A}$ finished the tracing game with $\mathcal{S}$ in Definition 17, $\mathcal{A}$ generates a commitment c for a hidden value a and range proof $\pi(c) = (\sigma, c, L_{PK}, \{TK_i, \pi(c_i, TK_i)\}_{i=0,\cdots,n-1})$, We assume that $\mathcal{A}$ wins the tracing game with nonnegligible advantage δ, that is, $\pi(c)$ satisfying the following:

1. $\mathsf{Verify}(c, \pi(c)) = 1$.
2. $\mathsf{Trace}(\pi(c), trapdoors) \neq a$.

According to the soundness of Borromean range proof, we know $c = g^x h^a$ with $a \in [0, 2^n - 1]$ and $c_i = g^{x_i} h^{2^i a_i}$ for every $i = 0, \cdots, n-1$ except for negligible probability ϵ_1. If $\mathsf{Trace}(\pi(c), trapdoors) \neq a$, then there exists j *s.t.* $TK_j \neq h^{x_j - 2^j a_j}$, we set $TK_j = g^s h^t$ without loss of generality. From the validity proof $\pi(c_j, TK_j) = (z_1, z_2, e_1, e_2)$ which proves TK_j is a power of h and $TK_j \cdot c_j = g^{s+x_i} h^{t+2^i a_i}$ is a power of gh, then we have $e_1 = H(h^{z_1}/(TK_j)^{e_1})$ and $e_2 = H((gh)^{z_2}/(g^{s+x_j} h^{t+2^j a_j})^{e_2})$, similar to Theorem 16, we know that $e_1 = H(g^{r_1} h^{r_2})$, $e_2 = H(g^{r_3} h^{r_4})$, then we have:

$$r_1 = -se_1,\ r_2 = z_1 - te_1,\ r_3 = z_2 - e_2(s + x_j),\ r_4 = z_2 - e_2(t + 2^j a_j).$$

If $s \neq 0$, then we have $e_1 = -r_1 s^{-1}$, which means $\mathcal{A}$ computes e_1 before he runs the hash function (query the random oracle), this happens with negligible probability ϵ_2. So we get $s = 0$, then $r_3 - r_4 = e_2^{-1}(t + 2^j a_j - x_j)$, if $t + 2^j a_j - x_j \neq 0$, then $e_2 = (t + 2^j a_j - x_j)^{-1}(r_3 - r_4)$, which means $\mathcal{A}$ computes e_2 before he runs the hash function with nonnegligible probability $\delta - \epsilon_1 - \epsilon_2$. This is a contradiction, we get $TK_j = g^s h^t = h^{x_j - 2^j a_j}$ and then $TK_j = h^{x_j - 2^j a_j}$, which contradicts to the assumptions before, so we finish the traceability of TBoRP. □

Modification In the construction of TBoRP, there are totally n validity proofs $\pi(c_i, TK_i)$ and $2n$ verifications of them (n for TK_i and n for $TK_i \cdot c_i$), the size and time are large compared to original Borromean range proof. In this subsection we modify the validity proofs and verifications of all TK_is to improve their efficiency (with size reduced by n times, and time reduced by twice at least). We also modify the Trace algorithm to reduce the tracing time by twice. We only describe the differences in the following:

- $(L_{PK}, SK, c, \{TK_i\}_{i=0,\cdots,n-1}, \pi) \leftarrow \mathsf{Gen'}(\mathsf{Par}, a)$:
  4'. For every $i = 0, \cdots, n-1$, Alice computes $TK_i = h^{x_i - 2^i a_i}$ and $e_i = H(c_0, \cdots, c_{n-1}, TK_0, \cdots, TK_{n-1}, i)$, gives all TK_i's validity proof $\pi(\{c_i\}, \{TK_i\}, \{e_i\})$ that $\prod_{i=0}^{n-1} TK_i^{e_i}$ is a power of h and $\prod_{i=0}^{n-1} (TK_i \cdot c_i)^{e_i}$ is a power of gh;
  5'. Alice outputs $L_{PK} = \{L_{PK}^0, \cdots, L_{PK}^{n-1}\}$, c, $\{TK_i\}_{i=0,\cdots,n-1}, \pi$ and retains $SK = (x_0, \cdots, x_{n-1})$, a.
- $1/0 \leftarrow \mathsf{Verify'}(\sigma, c, L_{PK}, \{TK_i\}_{i=0,\cdots,n-1}, \pi)$:
  1'. Verifier computes $e_0, \cdots, e_{n-1}$ and checks the validity of $\pi(\{c_i\}, \{TK_i\}, \{e_i\})$.
- $a^* \leftarrow \mathsf{Trace'}(\sigma, L_{PK}, y, \{TK_i\}_{i=0,\cdots,n-1})$:
  1'. For every $i = 0, \cdots, n-1$, regulator computes c_i^y;
  2'. For every $i = 0, \cdots, n-1$, if $c_i^y = TK_i$ then outputs $a_i^* = 1$, otherwise outputs $a_i^* = 0$;

The TK_is validity proof $\pi(\{c_i\}, \{TK_i\}, \{e_i\})$ works as follows:

1. Let $P_1 = \prod_{i=0}^{n-1} TK_i^{e_i}$ and $P_2 = \prod_{i=0}^{n-1} (TK_i \cdot c_i)^{e_i}$, prover generates $r_1, r_2 \in \mathbb{Z}_q^*$, computes $f_1 = H(h^{r_1}), f_2 = H((gh)^{r_2})$, then computes $z_1 = r_1 + f_1 \sum_{i=0}^{n-1} e_i(x_i - 2^i a_i)$, $z_2 = r_2 + f_2 \sum_{i=0}^{n-1} e_i x_i$, outputs the proof is $\pi = (z_1, z_2, f_1, f_2)$.
2. Verifier checks $f_1 \stackrel{?}{=} H(h^{z_1}/(P_1)^{f_1})$ and $f_2 \stackrel{?}{=} H((gh)^{z_2}/(P_2)^{f_2})$.

The security of modified scheme is easy to prove and we omit them for brevity, here we only need to discuss the improvement of efficiency:

For size: in the original TBoRP, every $\pi(c_i, TK_i) = (z_1, z_2, e_1, e_2)$, then there are totally $2n$ elements from $\mathbb{G}$ and $2n$ elements from $\mathbb{Z}_q^*$. In the modified scheme, $\pi = (z_1, z_2, f_1, f_2)$, then there are only 2 elements from $\mathbb{G}$ and 2 elements from $\mathbb{Z}_q^*$. It is easy to see that we reduce the size by n times.

For proving time: in the original TBoRP, prover need $2n$ hash computations, $2n$ exponentiations. In the modified scheme, prover need $n{+}2$ hash computations, 2 exponentiations. It is easy to see that we reduce the proving time by about n times.

For verification time: in the original TBoRP, verifier need $2n$ hash computations, $4n$ exponentiations, $2n$ multiplications and $2n$ comparisons. In the modified scheme, verifier need $n + 2$ hash computations, $2n + 4$ exponentiations, $2n + 1$ multiplications and 2 comparisons. It is easy to see that we reduce the verification time by about twice.

For tracing time: in the modified scheme, regulator no longer computes $(c_i')^y$, which reduce the tracing time by about twice.

4.3 Traceable Bulletproofs Range proof

Construction In the construction of TBuRP, similar to TBoRP, we use pedersen commitment and bit expansion of amount, then add tracing keys bitwise into the proof, together with the validity proof of tracing keys. The regulator can use the trapdoors and tracing keys to recover the hidden amount.

We give the introduction of TBuRP in the following:

- $\mathsf{Par} \leftarrow \mathsf{Setup}(\lambda)$: system chooses elliptic curve $\mathbb{G}$ and generators $g, h, g_0, \cdots, g_{n-1} \in \mathbb{G}$, the regulator generates $y_0, \cdots, y_{n/2-1} \in \mathbb{Z}_q^*$ as the trapdoors, computes $h_{2i} = g_{2i}^{y_i}, h_{2i+1} = g_{2i+1}^{y_i}, i = 0, \cdots, n/2-1$, system outputs $(\mathbb{G}, q, g, h, \mathbf{g}, \mathbf{h})$ as the public parameters, where $\mathbf{g} = (g_0, \cdots, g_{n-1}) \in \mathbb{G}^n, \mathbf{h} = (h_0, \cdots, h_{n-1}) \in \mathbb{G}^n$.
- $(A, S, c, \{TK_i\}_{i=0,\cdots,n-1}, \pi(TK_0, \cdots, TK_{n-1}, A)) \leftarrow \mathsf{Gen}(\mathsf{Par}, a)$:
  1. According to the public parameters $(\mathbb{G}, q, g, h, \mathbf{g}, \mathbf{h})$ and amount $a \in [0, 2^n - 1]$, prover Alice samples $x \in \mathbb{Z}_q^*$ uniformly, computes $c = h^x g^a$ as the commitment (consistent with Bulletproofs);
  2. Alice computes the binary expansion $a = a_0 + \cdots + 2^{n-1} a_{n-1}, a_i = 0, 1$ for $i = 0, \cdots, n-1$, sets $\mathbf{a}_L = (a_0, \cdots, a_{n-1})$;
  3. Alice computes $\mathbf{a}_R = \mathbf{a}_L - \mathbf{1}^n = (a_0 - 1, \cdots, a_{n-1} - 1)$;
  4. Alice samples $\alpha \in \mathbb{Z}_q$ uniformly at random, computes
  $$A = h^\alpha \mathbf{g}^{\mathbf{a}_L} \mathbf{h}^{\mathbf{a}_R} = h^\alpha g_1^{a_1} \cdots g_{n-1}^{a_{n-1}} h_1^{a_1 - 1} \cdots h_{n-1}^{a_{n-1}-1};$$
  5. Alice samples $\mathbf{s}_L, \mathbf{s}_R \in \mathbb{Z}_q^n, \rho \in \mathbb{Z}_q$ uniformly at random, computes $S = h^\rho \mathbf{g}^{\mathbf{s}_L} \mathbf{h}^{\mathbf{s}_R}$;
  6. For every $j = 0, \cdots, n/2-1$, Alice computes $TK_{2j} = g_{2j}^{\alpha - a_{2j}} g_{2j+1}^{\alpha - a_{2j+1}}$, $TK_{2j+1} = h_{2j}^{-\alpha - a_{2j}+1} h_{2j+1}^{-\alpha - a_{2j+1}+1}$, the number of TK_is is n;
  7. Alice gives the validity proof $\pi(TK_0, \cdots, TK_{n-1}, A)$ of all TK_is that TK_{2j} is a production of g_{2j}'s power and g_{2j+1}'s power, TK_{2j+1} is a production of h_{2j}'s power and h_{2j+1}'s power, and $A \cdot \prod_{i=0}^{n-1} TK_i = (h \prod g_i / \prod h_i)^\alpha$ is a power of $h \prod g_i / \prod h_i$;
  8. Alice outputs $(A, S, c, \{TK_i\}_{i=0,\cdots,n-1}, \pi(TK_0, \cdots, TK_{n-1}, A))$.
- $(T_1, T_2, \tau_x, \mu, t, \mathbf{l}, \mathbf{r}) \leftarrow \mathsf{Prove}(A, S, c, \{TK_i\}_{i=0,\cdots,n-1}, \pi(TK_0, \cdots, TK_{n-1}, A))$:
  1. Prover sends $(A, S, c, \{TK_i\}_{i=0,\cdots,n-1}, \pi(TK_0, \cdots, TK_{n-1}, A))$ to verifier;
  2. Verifier samples $y, z \in \mathbb{Z}_q$ uniformly at random, and sends them to prover;
  3. Prover computes T_1, T_2 and sends them to verifier;
  4. Verifier samples $x \in \mathbb{Z}_q$ uniformly at random, and sends it to prover;
  5. Prover computes $\tau_x, \mu, t, \mathbf{l}, \mathbf{r}$ and sends them to verifier.
- $1/0 \leftarrow \mathsf{Verify}$: we only introduce the verification of $\pi(TK_0, \cdots, TK_{n-1}, A)$:
  1. For every $i = 0, \cdots, n-1$, verifier checks the validity of TK_i;
  2. Verifier computes $A \cdot \prod_{i=0}^{n-1} TK_i$ and checks the validity of $A \cdot \prod_{i=0}^{n-1} TK_i$;
  3. Verifier continues the rest verification of Bulletproofs;
  4. If all pass then outputs 1, otherwise outputs 0.
- $a^* \leftarrow \mathsf{Trace}(\{TK_i\}_{i=0,\cdots,n-1}, y_0, \cdots, y_{n/2-1})$:

1. For every $j = 0, \cdots, n/2-1$, regulator computes $TK_{2j+1} \cdot TK_{2j}^{y_j}$;
2. If $TK_{2j+1} \cdot TK_{2j}^{y_j} = h_{2j}h_{2j+1}$, then outputs $(a_{2j}^*, a_{2j+1}^*) = (0,0)$;
3. If $TK_{2j+1} \cdot TK_{2j}^{y_j} = h_{2j}^{-1}h_{2j+1}$, then outputs $(a_{2j}^*, a_{2j+1}^*) = (1,0)$;
4. If $TK_{2j+1} \cdot TK_{2j}^{y_j} = h_{2j}h_{2j+1}^{-1}$, then outputs $(a_{2j}^*, a_{2j+1}^*) = (0,1)$;
5. If $TK_{2j+1} \cdot TK_{2j}^{y_j} = h_{2j}^{-1}h_{2j+1}^{-1}$, then outputs $(a_{2j}^*, a_{2j+1}^*) = (1,1)$;
6. Regulator outputs $a^* = a_0^* + \cdots + 2^{n-1}a_{n-1}^*$.

Correctness

Theorem 21 (Correctness of TBuRP) *For an honest user Alice in TBuRP, she can complete the Bulletproofs, and the regulator can trace her amount.*

Proof. We already know the correctness (completeness) of Bulletproofs, it remains to prove correctness of Trace. Since $h_{2j} = g_{2j}^{y_j}, h_{2j+1} = g_{2j+1}^{y_j}$ and $TK_{2j} = g_{2j}^{\alpha-a_{2j}} g_{2j+1}^{\alpha-a_{2j+1}}$, $TK_{2j+1} = h_{2j}^{-\alpha-a_{2j}+1} h_{2j+1}^{-\alpha-a_{2j+1}+1}$ for every $j = 0, \cdots, n/2-1$, then we have:

- When $(a_{2j}, a_{2j+1}) = (0,0)$,
  $TK_{2j+1} \cdot TK_{2j}^{y_j} = (g_{2j}^{\alpha-a_{2j}} g_{2j+1}^{\alpha-a_{2j+1}})^{y_j} \cdot h_{2j}^{-\alpha-a_{2j}+1} h_{2j+1}^{-\alpha-a_{2j+1}+1} = h_{2j}h_{2j+1}$;
- When $(a_{2j}, a_{2j+1}) = (1,0)$,
  $TK_{2j+1} \cdot TK_{2j}^{y_j} = (g_{2j}^{\alpha-a_{2j}} g_{2j+1}^{\alpha-a_{2j+1}})^{y_j} \cdot h_{2j}^{-\alpha-a_{2j}+1} h_{2j+1}^{-\alpha-a_{2j+1}+1} = h_{2j}^{-1}h_{2j+1}$;
- When $(a_{2j}, a_{2j+1}) = (0,1)$,
  $TK_{2j+1} \cdot TK_{2j}^{y_j} = (g_{2j}^{\alpha-a_{2j}} g_{2j+1}^{\alpha-a_{2j+1}})^{y_j} \cdot h_{2j}^{-\alpha-a_{2j}+1} h_{2j+1}^{-\alpha-a_{2j+1}+1} = h_{2j}h_{2j+1}^{-1}$;
- When $(a_{2j}, a_{2j+1}) = (1,1)$,
  $TK_{2j+1} \cdot TK_{2j}^{y_j} = (g_{2j}^{\alpha-a_{2j}} g_{2j+1}^{\alpha-a_{2j+1}})^{y_j} \cdot h_{2j}^{-\alpha-a_{2j}+1} h_{2j+1}^{-\alpha-a_{2j+1}+1} = h_{2j}^{-1}h_{2j+1}^{-1}$.

Then we get the correctness of TBuRP. □

Proof of Zero-knowledge

Theorem 22 (Zero-knowledge of TBuRP) *TBuRP is computational zero-knowledge for any PPT adversary $\mathcal{A}$ (without possession of trapdoors).*

Proof. For the structure of $TK_{2j} = g_{2j}^{\alpha-a_{2j}} g_{2j+1}^{\alpha-a_{2j+1}}$, $TK_{2j+1} = h_{2j}^{-\alpha-a_{2j}+1} \cdot h_{2j+1}^{-\alpha-a_{2j+1}+1}$ in TBuRP, we prove that, TBuRP is computational zero-knowledge for all TK_is are substituted for $S_i = g_i^{\alpha-a_i}, T_i = h_i^{\alpha+a_i}$. In fact, $TK_{2j} = g_{2j}^{\alpha-a_{2j}} g_{2j+1}^{\alpha-a_{2j+1}} = S_{2j}S_{2j+1}$, $TK_{2j+1} = h_{2j}^{-\alpha-a_{2j}+1} h_{2j+1}^{-\alpha-a_{2j+1}+1} = (T_{2j}T_{2j+1})^{-1}h_{2j}h_{2j+1}$.

For every $i = 0, \cdots, n-1$, we only need to prove that (S_i, T_i) does not reveal any knowledge of $a_i = 0$ or 1. Formally, we prove for any PPT adversary $\mathcal{A}$, cannot distinguish $(S_i, T_i) = (g_i^{\alpha}, h_i^{\alpha})$ (when $a_i = 0$) from $(S_i, T_i) = (g_i^{\alpha-1}, h_i^{\alpha+1})$ (when $a_i = 1$). Using the same argument from Theorem 19, we know that $(g_i, h_i, g_i^{\alpha}, h_i^{\alpha})$ and $(g_i, h_i, g_i^{\alpha}, r)$ are computational indistinguishable for uniformly generated $\alpha \in \mathbb{Z}_q^*$, under the DH assumption. Meanwhile, $(g_i, h_i, g_i^{\alpha}, r)$ and (g_i, h_i, r', r) are identical distributions as g is a generator of $\mathbb{G}$. We also know that the distribution of (g, h, r', r) and $(g, h, r'g_i^{-1}, rh_i)$ are identical. Moreover, again from the DH assumption, we know $(g, h, r'g_i^{-1}, rh_i)$ and $(g_i, h_i, g_i^{\alpha-1}, h_i^{\alpha+1})$ are computational indistinguishable. So we have:

$$(g_i, h_i, g_i^{\alpha}, h_i^{\alpha}) \approx_c (g_i, h_i, r', r) = (g, h, r'g_i^{-1}, rh_i) \approx_c (g_i, h_i, g_i^{\alpha-1}, h_i^{\alpha+1}).$$

Where g_i, h_i are constants, r, r', α are uniform random variables.

Since $(g_i^{\alpha}, h_i^{\alpha}) = (S_i, T_i)_{a_i=0}$ and $(g_i, h_i, g_i^{\alpha-1}, h_i^{\alpha+1}) = (S_i, T_i)_{a_i=1}$, we know they are computational indistinguishable for any PPT adversary $\mathcal{A}$, for every $i = 0, \cdots, n-1$, then we finish the zero-knowledge proof of TBuRP. □

Traceability

Theorem 23 (Traceability of TBuRP) *TBoRP is traceable for any PPT adversary $\mathcal{A}$ (without possession of trapdoor).*

Proof. The proof is quite similar as Theorem 16 and Theorem 20, which will be omitted for brevity. □

From the traceability of TBuRP, we can also modify the Trace to Trace', which reduces the time of tracing:

– $a^* \leftarrow \mathsf{Trace}'(\{TK_i\}_{i=0,\cdots,n-1}, y_0, \cdots, y_{n/2-1})$:
  5'. Otherwise outputs $(a^*_{2j}, a^*_{2j+1}) = (1, 1)$.

We denote the modified scheme as TBuRP'.

4.4 Discussion and Comparison

In the construction of TBuRP', there are $n/2$ trapdoors with 2 bits per round in the Trace' algorithm, moreover, for n bits amount, we set the number of trapdoors is n_0 and the number of bits in each round of Trace algorithm is n_1, we can get a conclusion that $n_0 \cdot n_1 = n$. Meanwhile, for Trace algorithm, the computation time in each round requires is 2^{n_1}. To sum up, the total tracing time is $T = (2^{n_1} - 1) \cdot \frac{n}{n_1}$, which meets the minimum n when $n_1 = 1$. When $n_1 = 1$, there will be n trapdoors, together with $2n$ TK_is, which is twice as much as TBuRP', so we choose $n_1 = 2$ as our parameter ($T = 1.5n$), with $n/2$ trapdoors and n TK_is. In the future work, the n_1 can be modified (such as $1, 4, 8, \cdots, n$) to adapt to different application requirements with tracing time increased and tracing keys number (and size) decreased.

Scheme	n_0	TK_is	Tracing Time	Proof Size $(\mathbb{G}, \mathbb{Z}_q^*)$	Soundness for Regulator
TBoRP'	1	n	n	$(2, 2)$	honest
TBuRP'$(n_1 = 1)$	n	$2n$	n	$(2n+1, 2n+1)$	malicious
TBuRP'$(n_1 = 2)$	$n/2$	n	$1.5n$	$(2n+1, n+1)$	malicious

In the table above we compare TBoRP' (the modified TBoRP) with TBuRP' in various aspects (where the proof size is the size of all TK_is validity proof).

5 Applications

Applications of TLRS, TBoRP and TBuRP for multiple regulators and auxiliary computing will be introduced in the full version (coming soon).

6 Conclusion

In the current research field of blockchain technology, the most important requirements are decentralization, privacy protection and regulation. On the one hand, users need to have sufficient security mechanisms to ensure the privacy of their personal data; on the other hand, it is necessary for the regulatory agencies to maintain legitimacy in the blockchain and effectively combat crimes; meanwhile, the essential feature of blockchain technology is "decentralization". The privacy-preserving blockchain systems represented by Monero and Zerocash realize the functions of decentralization and privacy protection, but they cannot provide regulation. The consortium blockchain system with trusted center has privacy protection and regulation, but can not meet the decentralization requirements. How to balance and fully realize these three characteristics is an open problem in the academia and business community.

In this paper, we study and classify the regulatability of privacy-preserving blockchains, and determine the regulatory model with *unconditional regulation*, *static regulation*, and *self-participation* of users as the core principals. Then, based on cryptographic components such as classic ring signature, one-time signature and switch proofs, we propose the traceable and linkable ring signature TLRS for the first time, and give the security proofs. TLRS realizes the regulatory function for users' identities, and can prevent the malicious regulator from double spending, escaping from regulation, slandering users and forging signatures, which is a regulatable scheme that minimizes the regulator's power and meets the characteristic of "decentralization" to the greatest extent. Moreover, based on cryptographic techniques such as commitment proofs, binary expansion, and randomized combination, we propose the traceable Borromean range proof TBoRP and traceable Bulletproofs range proof TBuRP for the first time, together with their security proofs and modifications for efficiency. Both TBoRP and TBuRP achieve the regulatory function for amounts of transactions, and are well suited for replacement of Monero system. Finally, we give the applications including multiple regulators and auxiliary computing on the basis of TLRS, TBoRP and TBuRP, which have potential in future scenarios such as multi-currency transfer, international trades and taxing.

Future Works In the future, we need to study and improve in the following aspects:

1. For TLRS, TBoRP and TBuRP, improve their efficiency to reach the level of Monero system;

2. Considering the weakness of Pedersen commitment (in TBoRP), where regulator can alter the hidden amount by use of trapdoor, we need to design new schemes as well as new commitments to prevent the attack of malicious regulator;
3. Study new range proof systems without using of binary expansion to reduce the number of tracing keys;
4. Study post-quantum ring signatures and range proofs, such as lattice-based, code-based, multi-variant-based and isogen-based schemes to prepare for the future applications and replacement in the era of quantum computing.

References

1. Abe, M., Ohkubo, M., Suzuki, K.: 1-out-of-n signatures from a variety of keys. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 415–432. Springer (2002)
2. Au, M.H., Chow, S.S., Susilo, W., Tsang, P.P.: Short linkable ring signatures revisited. In: European Public Key Infrastructure Workshop. pp. 101–115. Springer (2006)
3. Au, M.H., Liu, J.K., Susilo, W., Yuen, T.H.: Certificate based (linkable) ring signature. In: International Conference on Information Security Practice and Experience. pp. 79–92. Springer (2007)
4. Au, M.H., Liu, J.K., Susilo, W., Yuen, T.H.: Secure id-based linkable and revocable-iff-linked ring signature with constant-size construction. Theoretical Computer Science **469**, 1–14 (2013)
5. Back, A.: Ring signature efficiency. Bitcointalk (accessed 1 May 2015) https://bitcointalk. org/index. php (2015)
6. Bender, A., Katz, J., Morselli, R.: Ring signatures: Stronger definitions, and constructions without random oracles. In: Theory of Cryptography Conference. pp. 60–79. Springer (2006)
7. Bünz, B., Agrawal, S., Zamani, M., Boneh, D.: Zether: Towards privacy in a smart contract world. IACR Cryptology ePrint Archive **2019**, 191 (2019)
8. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: 2018 IEEE Symposium on Security and Privacy (SP). pp. 315–334. IEEE (2018)
9. Buterin, V., et al.: A next-generation smart contract and decentralized application platform. white paper **3**, 37 (2014)
10. Chandran, N., Groth, J., Sahai, A.: Ring signatures of sub-linear size without random oracles. In: International Colloquium on Automata, Languages, and Programming. pp. 423–434. Springer (2007)
11. Community, D.: Dero white paper. URl: https://dero.io/attachment/Whitepaper.pdf (2019)
12. Dodis, Y., Kiayias, A., Nicolosi, A., Shoup, V.: Anonymous identification in ad hoc groups. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 609–626. Springer (2004)
13. Facebook: Libra white paper. URl: https://libra.org/en-US/white-paper/ (2019)
14. Fujisaki, E., Suzuki, K.: Traceable ring signature. In: International Workshop on Public Key Cryptography. pp. 181–200. Springer (2007)

15. Groth, J., Kohlweiss, M.: One-out-of-many proofs: Or how to leak a secret and spend a coin. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 253–280. Springer (2015)
16. Jedusor, T.E.: Mimblewimble (2016)
17. Liu, J.K., Au, M.H., Susilo, W., Zhou, J.: Linkable ring signature with unconditional anonymity. IEEE Transactions on Knowledge and Data Engineering **26**(1), 157–165 (2013)
18. Liu, J.K., Wei, V.K., Wong, D.S.: Linkable spontaneous anonymous group signature for ad hoc groups. In: Australasian Conference on Information Security and Privacy. pp. 325–335. Springer (2004)
19. Maxwell, G.: Confidential transactions. URL: https://people. xiph. org/˜greg/confidential_values. txt (Accessed 09/05/2016) (2015)
20. Maxwell, G., Poelstra, A.: Borromean ring signatures (2015)
21. Nakamoto, S., et al.: Bitcoin: A peer-to-peer electronic cash system (2008)
22. Noether, S., Mackenzie, A., et al.: Ring confidential transactions. Ledger **1**, 1–18 (2016)
23. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Annual International Cryptology Conference. pp. 129–140. Springer (1991)
24. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 552–565. Springer (2001)
25. Sasson, E.B., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M.: Zerocash: Decentralized anonymous payments from bitcoin. In: 2014 IEEE Symposium on Security and Privacy. pp. 459–474. IEEE (2014)
26. Sun, S.F., Au, M.H., Liu, J.K., Yuen, T.H.: Ringct 2.0: A compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency monero. In: European Symposium on Research in Computer Security. pp. 456–474. Springer (2017)
27. Tsang, P.P., Wei, V.K.: Short linkable ring signatures for e-voting, e-cash and attestation. In: International Conference on Information Security Practice and Experience. pp. 48–60. Springer (2005)
28. Van Saberhagen, N.: Cryptonote v 2.0 (2013)
29. Yuen, T.H., Liu, J.K., Au, M.H., Susilo, W., Zhou, J.: Efficient linkable and/or threshold ring signature without random oracles. The Computer Journal **56**(4), 407–421 (2013)