Fully regulatable privacy-preserving blockchains against malicious regulators

Wulu Li, Lei Chen, Xin Lai, Xiao Zhang, and Jiajun Xin

Shenzhen Onething Technologies Co., Ltd., Shenzhen, China

Abstract. Privacy protection has been extensively studied in the current blockchain research field. As representations, Monero and Zerocash have realized fully 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 "decentralization" in regulatable privacy-preserving blockchains. On the basis of the established regulatory model, we propose a traceable and linkable ring signature scheme (TLRS) by usage of classic ring signatures, one-time signatures and zero-knowledge proofs. TLRS achieves unforgeability, anonymity, linkability, nonslanderability and traceability against malicious regulators under standard assumptions. Moreover, we give the first construction of traceable range proofs, including traceable Borromean range proof (TBoRP) and traceable Bulletproofs range proof (TBuRP) by usage of zero-knowledge proofs and D-H assumptions, with completeness, soundness, zero-knowledge and traceability of the transaction amounts. We modify the TBoRP to achieve security against malicious regulators by adding the mirror commitments. In addition, we discuss the traceability for long-term addresses, which finishes the first construction of the fully regulatable privacy-preserving blockchains against malicious regulators.

Keywords: Regulatable blockchains · Privacy preserving · Decentralization · Traceable and linkable ring signatures · Traceable range proofs · Malicious regulators.

1 Introduction

Blockchain technology was first proposed by Nakamoto 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, transparency, non-tamperability, traceability, and has various applications such as cryptocurrency (including Bitcoin, Ethereum, Monero, Zerocash, etc.), anti-counterfeiting, credit
deposit and medical health, etc. Blockchain technology has been widely con-
cerned by government, financial departments, scientific academia, and has po-
tential to lead the next industrial technology revolution. In June 2019, Facebook
announced “Libra”[12], an international blockchain-based cryptocurrency, jointly
with Visa, Mastercard, eBay, PayPal, etc. This indicates that the world is
gradually entering a new era of blockchain-based cryptocurrency.

Traditional blockchain-based cryptocurrency represented by Bitcoin and Eth-
ereum, 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, there are solutions pro-
posed during these years such as Confidential Transaction[19], Mimblewimble,
Dash[11], Monero[28] and Zerocash[25], etc. Among them, Monero successively
followed the direction of Cryptonote[28], Ring-CT[22], Bulletproofs[6]. In reality,
Monero uses linkable ring signature scheme to hide the identity of initiator, uses
Diffie-Hellman key exchange scheme to hide the identity of receiver and uses
range proof (Borromean, Bulletproofs) to hide the 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 amoun-
t. Nevertheless, zk-SNARKs technique uses common reference string (CRS) of
Gigabyte size and it is based on non-falsifiable assumptions, that weakens its
potential competitiveness against other cryptocurrencies. Recent projects, such
as DERO[9] and ZETHER[5], all used Monero-type techniques (ring signatures,
Bulletproofs) to protect privacy, in this paper, we also focus on modification for
Monero system.

It should be noted that strong privacy-preserving blockchains have no regu-
latory functions, this feature brings about potential risks of illegal purposes such
as illegal transactions, illegal asset transfers, money laundering, fraud, etc. Viola-
tors can go unpunished with impunity, which seriously restricts the application
prospects of privacy-preserving blockchains, and cannot be recognized by the
regulatory agencies. Meanwhile, in the system of privacy-preserving cryptocurrency,
regulators and policy-making institutions need to have a comprehensive
understanding of the economic operation and development, they need to trace
the identities and amounts of transactions. Therefore, for the sake of legitimate
and sustainable development, the privacy-preserving blockchain system must
have the functionality of full regulation (for signer identity, user address and
amount). Moreover, the regulatable privacy-preserving blockchain system need-
s to keep secure when regulators are corrupted or malicious, which means the
regulators can only trace identities and amounts in transactions, while cannot
double spend, corrupt users, slander users or escape from regulation.

A possible solution for regulatable privacy-preserving blockchain is consor-
tium blockchain system with trusted center, where CA can distribute or certify
keys for users to achieve anonymous and traceable transactions, by making use
of cryptographic protocols such as linkable group signatures\cite{32, 31}. But in the application of group signatures, malicious authority may have the ability to forge signatures of users, or to authorize malicious users into the group, which is harmful to the entire system, and it is not a decentralized solution to achieve both anonymity and traceability. To summarize, it is necessary to study blockchain systems that meet the characteristics of decentralization, have regulatory functions, protect users’ privacy, and support future applications of cryptocurrency.

1.1 Related Works

**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 $SK_{\pi}$ to sign, verifier cannot determine signer’s identity. Ring signature was first proposed by Rivest, Shamir and Tauman\cite{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.\cite{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.\cite{4} introduced the first ring signature scheme in the standard model, by making use of pairing technique. In 2015, Maxwell et al.\cite{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 signature sizes in these schemes are linear to the number of ring elements.

In 2004, building from RSA accumulator, Dodis et al.\cite{10} proposed a ring signature scheme with constant signature size in the random oracle model. In 2007, Chandran et al.\cite{8} 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.\cite{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, 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 cryptocurrency (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.\cite{18} in 2004, under discrete logarithm assumption, in the random oracle model. Later, Tsang et al.\cite{27} and Au et al.\cite{2} proposed accumulator-based linkable ring signatures with constant signature size. In 2013, Yuen et al.\cite{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.[3] proposed a efficient linkable ring signature scheme LSAG, which shorten the signature size of [18]. In 2016, based on work of Fujisaki et al.[13], 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 asymptotic smaller signature size than Ring CT, but is less competitive when n is small, besides, the anonymity of Ring-CT 2.0 is lower than Ring-CT for multiple inputs. In 2019, Yuen et al.[30] proposed Ring-CT 3.0, a modified Bulletproof-based 1-out-of-n proof protocol with logarithmic size, which has functionality of (linkable) ring signature and is being tested by the Monero group. In 2019, Goodell et al.[14] proposed CLSAG, which improved the efficiency of MLSAG.

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.[13] in 2007, based on discrete logarithm assumption, their scheme provide conditional traceability. In 2019, Li et al.[16] gives a construction of traceable Monero to achieve anonymity and traceability of identities by usage of paring-based accumulators, signature of knowledge and verifiable encryption from Ring-CT 2.0, their construction provide the functionality of traceable and linkable ring signature, but relies on extra steps of verifiable encryption and decryption. Besides, in [16], traceability of long-term address depends on zk-SNARKs with CRS, which is inefficient for computation and storage.

In this paper, we construct a traceable and linkable ring signature with unconditional traceability (signer identities and long-term addresses) and more efficient tracing algorithm, while under standard assumptions.

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 linear proof size to the binary length of range. In 2018, Bünz et al.[6] introduced Bulletproofs, an efficient non-interactive zero-knowledge proof protocol with short proofs and without a trusted setup, the proof size is only logarithmic to 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 the regulatory models on the privacy-preserving blockchains to determine the regulatory model used in our constructions. Then we give the construction of the fully regulatable privacy-preserving blockchains against malicious regulators. 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). In addition, we also give the construction to realize traceability for long-term addresses.

Regulatory Model Regulations on the privacy-preserving blockchain can be classified according to various classification methods:

- As for 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 signer, as well as the transaction amount and address.

- As for participation mode of regulators, 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 validity 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 validity verification of the transactions, 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.

- As for participation mode of users, 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 regulation, we choose unconditional regulation, static regulation, and self-participation of users as the key principles of the fully regulatable privacy-preserving blockchains.

Traceable and Linkable Ring Signatures In this paper, we slightly modify the definition of traceable ring signature in [13] by introducing the role of regulators who can trace signers’ identities in all transactions (including legal and illegal ring signatures). It’s worth emphasizing that the new definition of traceable ring signature is deeply related with (linkable) 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 authority, 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 spend, even for malicious regulator, which is the closest approach to “decentralization”.
- In group signatures, management of group (join or delete members) 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.

From the discussion above, we obtain the definition of traceability in linkable ring signatures, with the linkability remaining the same that two ring signatures can be linked if they are signed by the same signer. Informally, we give a construction of traceable and linkable ring signature scheme (TLRS) by usage of 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 \((G, q, g_1, g_2, h = g_2^y)\), 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 usage 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 \(\sigma_1\), and uses one-time private key \(OSK\) for one-time signature \(\sigma_2\).
4. The verifier checks the validity of \(TK\)’s validity proof, then checks whether \(OPK\) appears in previous signatures to determine whether illegal signature (double spending) occurs. Then checks the validity of classic ring signature \(\sigma_1\) and one-time signature \(\sigma_2\) and outputs the verification results.
5. The regulator can trace the identity of signer by using trapdoor \(y\).

In the construction of TLRS, under the discrete logarithm assumption, nobody else can steal the secret keys, nor forge TLRS signatures of users. Moreover, for malicious regulator (or adversary with possession of trapdoor), he can only break the anonymity of TLRS, but cannot generate illegal TLRS signatures (double spend), cannot slander other users (make other legal TLRS signatures illegal) or break traceability (escape from regulation), which is the closest approach to the “decentralization” requirement. Compared to [16], our TLRS has two main advantages:

1. Construction of TLRS is flexible, we can use arbitrary elliptic-based ring signature as component (such as AOS, Ring-CT 3.0) to achieve smaller signature sizes by choosing the most efficient ones in different applications.
2. Anonymity of TLRS (similar to Ring-CT) is stronger than [16] (similar to Ring-CT 2.0) for multiple inputs.

**Traceability of Long-term Addresses** We modify the key generation and spending algorithm in Cryptonote[28] by adding additional view key into user’s long-term address to realize anonymity except for regulators, who can use trapdoor to recover the address of receiver and then he can trace the addresses of all UTXOs in the blockchain.

**Traceable Borromean Range Proof** Traceable range proof is a special variant of range proof, in which there is a regulator can use trapdoors to trace the amount. The zero-knowledge property of traceable range proof only holds for users without possession of trapdoors. We give the first construction of traceable range proofs, 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 generates 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 all $TK_i$'s validity proofs, which 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 usage of trapdoor $y$ 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 $x$ from $c = g^x h^a$, which partially protects privacy of users, and it is a balance between regulation and privacy protection. We also give the modification of TBoRP (named TBoRP') by adding the mirror commitments to achieve soundness and traceability against malicious regulators, which means with possession of trapdoor, the adversary cannot generate fake proofs or escape from regulation.

**Traceable Bulletproofs Range Proof** We modify the Bulletproofs to achieve regulatory function by adding trapdoors into the public parameters. Informally, assume $n$ (bit length of range) is even (similar to Monero), for different generators $g = (g_0, \cdots, g_{n-1})$ generated independently by system, the regulator generates $y_0, \cdots, y_{n/2-1}$ as trapdoors, then computes $h_{2i} = g_{2i}^y, h_{2i+1} = g_{2i+1}^y$ for $i = 0, \cdots, n/2 - 1$, the regulator outputs $g = (g_0, \cdots, g_{n-1}), h = (h_0, \cdots, h_{n-1})$ as the 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 all $TK_i$'s validity
proofs, then generates the rest of Bulletproofs. The verifier checks the validity of Bulletproofs and the validity of proofs of all $TK_i$s. The regulator can trace the amount $a$ by calculating with trapdoors and $TK_i$s.

In our construction of TBuRP, the soundness holds for malicious regulator, and the number of trapdoors and tracing keys can be adjusted for different regulatory requirements in application, which gives potential replacement for regulatable Bulletproofs-based blockchains.

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 introduce the method to achieve traceability of long-time address; in section 5 we give the construction and security proofs of the traceable Borromean range proof (TBoRP) and the traceable Bulletproofs range proof (TBuRP); in section 6 we give the conclusion.

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 prime 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 $\text{negl}$ to represent negligible functions. For verifiers, 1 is for accept and 0 is for reject.

2.2 Ring Signatures

Classic Ring Signatures

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

- $\text{Par} \leftarrow \text{Setup}(\lambda)$ is a probabilistic polynomial time (PPT) algorithm which, on input a security parameter $\lambda$, outputs the set of security parameters $\text{par}$ which includes $\lambda$.
- $(PK_i, SK_i) \leftarrow \text{KeyGen}(\text{par})$ is PPT algorithm which, on input security parameters $\text{par}$, outputs a private/public key pair $(PK_i, SK_i)$.
- $\sigma \leftarrow \text{Sign}(SK_\pi, \mu, L_{PK})$ is a ring signature algorithm which, on input user’s secret key $SK_\pi$, a list of users’ public keys $L_{PK} = \{PK_1, \cdots, PK_n\}$, where $PK_\pi \in L_{PK}$, and message $\mu$, outputs a ring signature $\sigma$.
- $1/0 \leftarrow \text{Verify}(\mu, \sigma, L_{PK})$ is a verify algorithm which, on input message $\mu$, a list of users’ public keys $L_{PK}$ and ring signature $\sigma$, outputs 1 or 0.

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.
− $PK_i \leftarrow \mathcal{JO}(\bot)$. 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 $PK_\pi$, and a message $\mu$, returns a valid ring signature $\sigma$.

**Definition 1 (Unforgeability)** Unforgeability for ring signature schemes is defined in the following game between the simulator $S$ and the adversary $A$:

1. Simulator $S$ runs $\text{Setup}$ to provide public parameters for $A$, $A$ is given access to oracles $\mathcal{RO}$, $\mathcal{JO}$, $\mathcal{CO}$ and $\mathcal{SO}$. $A$ wins the game if he successfully forges a ring signature $(\sigma^*, L^*_{PK}, \mu^*)$ satisfying the following:
   1. $\text{Verify}(\sigma^*, L^*_{PK}, \mu^*) = 1$.
   2. Every $PK_i \in L^*_{PK}$ is returned by $A$ to $\mathcal{JO}$.
   3. No $PK_i \in L^*_{PK}$ is queried by $A$ to $\mathcal{CO}$.
   4. $(\mu^*, L^*_{PK})$ is not queried by $A$ to $\mathcal{SO}$.

   We give the advantage of $A$ in forge attack as follows:
   $$\text{Adv}_A^{\text{forge}} = \Pr[A \text{ wins}].$$

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

**Definition 2 (Anonymity)** Anonymity for ring signature schemes is defined in the following game between the simulator $S$ and the adversary $A$:

Simulator $S$ runs $\text{Setup}$ to provide public parameters for $A$, $A$ is given access to oracles $\mathcal{RO}$, $\mathcal{JO}$ and $\mathcal{CO}$. $A$ gives a set of public keys $L_{PK} = \{PK_1, \cdots, PK_n\}$, $S$ randomly picks $\pi \in \{1, \cdots, n\}$ and computes $\sigma = \text{Sign}(SK_\pi, \mu, L_{PK})$, where $SK_\pi$ is a corresponding private key of $PK_\pi$, and sends $\sigma$ to $A$, then $A$ output a guess $\pi^* \in \{1, \cdots, n\}$. $A$ wins the game if he successfully guesses $\pi^* = \pi$.

We give the advantage of $A$ in anonymity attack as follows:
$$\text{Adv}_A^{\text{anon}} = |\Pr[\pi^* = \pi] - 1/n|.$$ 

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

In the construction of TLRS, we use classic ring signature (unforgeable and anonymous in the random oracle model) as component, we may select AOS scheme (linear size) or Ring-CT 3.0 (logarithmic size) in our construction. The choice of ring signature is not restricted, we can choose the most suited ones (most efficient ones) for different ring sizes in different applications, we omit the detailed description of these ring signatures 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:

\[ \text{linked/unlinked} \leftarrow \text{Link}\left((\sigma, \mu, L_{PK}), (\sigma', \mu', L'_{PK})\right): \text{verifier checks the two ring signatures are linked or not, output the result.} \]

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 \( A \) to \( JO \) and all public keys in \( L_{PK} \) are not queried by \( A \) to \( CO \). 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 \( S \) and the adversary \( A \), simulator \( S \) runs \( \text{Setup} \) to provide public parameters for \( A \), \( A \) is given access to oracles \( RO, JO, CO \) and \( SO \). \( A \) 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 \( \sigma_i \)s are not returned by \( A \) to \( SO \).
2. All \( L^i_{PK} \) are returned by \( A \) to \( JO \).
3. \( \text{Verify}(\sigma_i, L^i_{PK}, \mu_i) = 1, i = 1, \cdots, k \).
4. \( A \) queried \( CO \) less than \( k \) times.
5. \( \text{Link}(\sigma_i, L^i_{PK}, \mu_i), (\sigma_j, L^i_{PK}, \mu_j)) = \text{unlinked} \) for \( i, j \in \{1, \cdots, k\} \) and \( i \neq j \).

We give the advantage of \( A \) in linkability attack as follows:

\[ \text{Adv}_{\text{link}}^A = \Pr[A \text{ wins}] \]

A linkable ring signature scheme is linkable if for any PPT adversary \( A \), \( \text{Adv}_{\text{link}}^A = \text{negl} \).

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

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

We give the advantage of $A$ in slandering attack as follows:

$$\text{Adv}_A^{\text{slander}} = \Pr[A \text{ wins}].$$

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

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 $A$,

$$\Pr[(x, w) \leftarrow A(1^\lambda) : (x, w) \not\in R \text{ or } \langle P(x, w), V(x) \rangle = 1] = 1 - \text{negl}.$$  

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 $A$ and $x \notin L$,

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

In $\Sigma$ 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 $A$ can generate 2 valid proofs $(x, c, e_1, s_1), (x, c, e_2, s_2)$, then there exists an extraction algorithm $\text{Ext}$ 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 $A$,

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

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

In Fiat-Shamir-based protocol, the randomness of $\rho$ 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 \((G, q = |G|, g, h)\), where \(g\) is a generator of \(G\), \(h\) 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 \(A\) cannot distinguish \(c = g^x h^a\) from \(c' = g^{x'} h^{a'}\).
- (Binding) Any PPT adversary \(A\) 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 \(a + b\).

**Proof of Committed Values** For commitment \(c = \prod_{i=1}^{n} g_i^{x_i}\), we can prove the knowledge of \(x_1, \cdots, x_n\) without revealing them by proof of committed values:

1. Prover generates \(r_1, \cdots, r_n \in \mathbb{Z}_q^*\) uniformly, computes \(e = H(\prod_{i=1}^{n} g_i^{r_i})\).
2. Prover computes \(z_i = r_i + ex_i\) for \(i = 1, \cdots, n\), output proof \(\pi(c) = (z_1, \cdots, z_n, e)\).
3. Verifier checks \(e \equiv H(\prod_{i=1}^{n} g_i^{z_i}/c^e)\).

Proof of committed values is an extension of Schnorr signature \((n = 1)\) 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, 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 \(\pi_{\text{Swit}}(c_1, c_2) = (z_1, z_2, z_3, e)\).
3. Verifier checks \(e \equiv H(g^{z_1} h_1^{z_1}/c_1^e, g^{z_2} h_2^{z_2}/c_2^e)\).

Specially, when \(x = y\), then prover samples \(r_1 = r_2\), then \(z_1 = z_2\), which shortens 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 \((G, q, g, h)\).
- **Gen**:...
inner-product argument, range proof and proof for arithmetic circuits. The Bul-

Bulletproofs Range Proof Bulletproofs, proposed by Bünz et al. 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 $(G, q, g, h, h)$, where $g = (g_0, \cdots, g_{n-1}), h = (h_0, \cdots, h_{n-1}) \in G^n$.
- **Gen**:
  1. According to the public parameters, amount $a \in [0, 2^n - 1]$, prover computes the commitment $c = g^a$.
  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 $c_i = g^{x_i} h^{2^{a_i}}, c'_i = g^{x_i} h^{2^{a_i-1}}$, prover outputs $L_{PK} = (c, c'_i)$.
- **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 $\sigma = \text{Sign}(L, SK, c)$, where $SK = (x_0, \cdots, x_{n-1})$.
- **Verify**: For $i = 0, \cdots, n-1$, verifier checks as follows:
  1. Checks $\prod c_i^0 = c$ and $c_i/c'_i \equiv h^2$;
  2. Checks the validity of Borromean ring signature $\sigma$, if all pass then outputs 1, otherwise outputs 0.

The computation of $T_1, T_2, \tau_x, \mu, t, l, r$ as well as the verification algorithm is omitted for brevity, please refer to [6] for detailed description.
3 Traceable and Linkable Ring Signature

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 against malicious regulators. In the scenario of blockchain, unforgeability works for security of users’ accounts, anonymity works for anonymity of signers’ identities, linkability and nonslanderability works for prevention of double-spending (actively or passively), traceability works for unconditional regulation of signers’ identities.

3.1 Construction

In our construction of TLRS, we use classic ring signature (AOS, Borromean or Ring-CT 3.0) as the ring signature component of TLRS, we use ECDSA or Schnorr signature as the one-time signature component of TLRS. Actually, these schemes are anonymous and unforgeable, which makes TLRS secure against malicious regulators. Moreover, Trace algorithm is added into TLRS to realize the regulation of signer’s identity, which makes TLRS suitable for application of fully regulatable privacy-preserving blockchains.

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

- \( \text{Par} \leftarrow \text{Setup}(\lambda) \): system chooses elliptic curve \( G \) and generators \( g_1, g_2 \in G \) independently, the regulator generates \( y \in \mathbb{Z}_q^* \) as the trapdoor, computes \( h = g_2^y \), system outputs \((G, q, g_1, g_2, h)\) as the public parameters, in which the regulator dose not know the relation between \( g_1 \) and \( h \).

- \((PK, SK) \leftarrow \text{KeyGen}(\text{Par})\):
  1. According to the public parameters \((G, q, g_1, g_2, h)\), user Alice samples \( x, a \in \mathbb{Z}_q^* \), computes \( RPK = g_1^x h^a, TK = g_2^y, OPK = h^a \);
  2. Alice gives the validity proof \( \pi(RPK, TK) = \pi_{\text{Swit}}(g_1^x h^a, g_1^x (g_2 h)^a) \), that is, she gives the switch proof between \( RPK = g_1^x h^a \) and \( RPK \cdot TK = g_1^x (g_2 h)^a \) that they share the same exponents \( (x = x, a = a) \) with basis \( (g_1, h) \) and \( (g_1, g_2 h) \);
  3. Alice outputs \( PK = (RPK, TK, \pi(RPK, TK)) \), and retains \( SK = (RSK = x, OSK = a) \).

- \( \sigma \leftarrow \text{Sign}(SK_x, \mu, L_{PK})\):
  1. For a message \( \mu \), 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 \( OPK = h^{a_{\pi}} \), then computes \( L_{RKP} = \{RPK_1 \cdot OPK^{-1}, \cdots, RPK_n \cdot OPK^{-1}\} \)
    \begin{align*}
    &= \{g_1^{x_1 h^{a_{\pi} - a_\pi}}, \cdots, g_1^{x_n h^{a_{\pi} - a_\pi}}\};
    \end{align*}
  3. Alice runs ring signature \( \sigma_1 \leftarrow \text{Rsign}(RSK, \mu, L_{RKP}, OPK) \) using \( L_{RKP} \) and \( RSK = x_\pi \), outputs \( \sigma_1 \);
4. Alice runs one-time signature $\sigma_2 \leftarrow \text{Osign}(OSK, \sigma_1, OPK)$ using $OPK = h^{a_2}$ and $OSK = a_2 (h$ as the generator);
5. Alice outputs $\sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, OPK)$.

- $1/0 \leftarrow \text{Verify}(\sigma_1, \sigma_2, \mu, L_{PK}, OPK)$:
  1. Verifier checks the validity of $\pi(RPK_i, TK_i)$ for every $1, \cdots, n$;
  2. Verifier checks $L_{RPK} \vdash \{RPK_1 \cdot OPK^{-1}, \cdots, RPK_n \cdot OPK^{-1}\}$;
  3. Verifier checks the validity of ring signature $\sigma_1$ and signature $\sigma_2$;
  4. If all passed then outputs 1, otherwise outputs 0.

- $\text{linked/unlinked} \leftarrow \text{Link}(\sigma, \sigma')$: 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 $\text{linked}$, otherwise outputs $\text{unlinked}$.

- $\pi^* \leftarrow \text{Trace}(\sigma, y)$: For $\sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, OPK)$, the regulator extracts $TK_1, \cdots, TK_n$ from $L_{PK}$, computes $TK^y_i$ for $i = 1, \cdots, n$, outputs the smallest $\pi^*$ such that $OPK = TK^y_\pi$ as the trace result, otherwise outputs $\perp$.

In the application of privacy-preserving blockchains, using UTXO model, the $TK = g^y$ can be regarded as the UTXO public key generated in the last transaction, and user will publish the UTXO public key $PK = (RPK, TK, \pi(RPK, TK))$ after receiving the money. The validity of $\pi(RPK, TK)$ can be checked by designated nodes before the transaction happens, which can reduce the cost of transaction verification. $OPK$ can be regarded as the Key-image of UTXO, and $\text{Link}$ is used for detection of double-spending. $\text{Trace}$ is used for tracing signers’ identities 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 correctly.

**Proof.** In TLRS, for Alice’s public key $PK = PK_\pi = (g_1^y h^a, g_2^a, \pi(g_1^y h^a, g_2^a))$, then Alice will output $OPK = h^a$ with $L_{RPK} = \{g_1^x h^{-a}, \cdots, g_n^x h^{-a}\}$. Since $g_1^{x^y} h^{-a} = g_1^\pi$, 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 ($h$ as the generator).

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

**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 $RO$, $JO$, $CO$ and $SO$, and security of TLRS contains unforgeability, anonymity, linkability, nonslanderability and traceability. Considering the existence of regulator, who can trace the identities of signers, 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 corrupts the regulator), he cannot forge signatures 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 signers’ identities, for any PPT adversary 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 $S$ and the adversary $A$:

- $S$ runs $\text{Setup}$ to provide public parameters for $A$, $A$ is given access to oracles $\text{RO}, \text{FO}, \text{CO}$. $A$ generates a list of public keys $L_{PK} = \{PK_1, \ldots, PK_n\}$, $A$ wins the game if he successfully generates a TLRS signature $(\sigma, L_{PK}, \mu)$ using $PK_{\pi} \in L_{PK}$, satisfying the following:

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

We give the advantage of $A$ in traceability attack as follows:

$$\text{Adv}^\text{trace}_A = \Pr[A \text{ wins}].$$

TLRS scheme is traceable if for any PPT adversary $A$, $\text{Adv}^\text{trace}_A = \text{negl}$.

**Proof of Anonymity**

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

**Proof.** Assume $A$ is playing the game with $S$ in Definition 2, $A$ he generates a message $\mu$ and a list of public keys $L_{PK} = \{PK_1, \ldots, PK_n\}$, where $PK_i = (\text{RKP}_i = g_i^z h^{a_i}, TK_i = g_i^{a_i}, \pi(\text{RKP}_i, TK_i))$, all $PK_i$s are returned by $\text{FO}$, and $S$ knows all $SK_i = (x_i, a_i)$.

Consider the following games between $S$ and $A$:

- **Game 0.** $S$ samples $\pi \in \{1, \ldots, n\}$ uniformly, publishes $\text{OPK} = h^{a_\pi}$ and $L_{\text{RKP}} = \{g_1^{x_1} h^{a_1-a_{\pi}}, \ldots, g_n^{x_n} h^{a_n-a_{\pi}}\}$, generates ring signature $\sigma_1 = \text{Rsign}(\text{RSK}, \mu, L_{\text{RKP}}, \text{OPK})$ and one-time signature $\sigma_2 = \text{Osign}(\text{OSK}, \sigma_1, \text{OPK})$, outputs $\sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, \text{OPK})$ to $A$. When $A$ receives $\sigma$, he gives a guess $\pi^* \in \{1, \ldots, n\}$.

- **Game 1.** $S$ uniformly samples $\pi \in \{1, \ldots, n\}, r \in \mathbb{Z}_q^*$, publishes $\text{OPK} = h^r$ and $L_{\text{RKP}} = \{g_1^{x_1} h^{a_1-r}, \ldots, g_n^{x_n} h^{a_n-r}\}$, generates ring signature $\sigma_1 = \text{Rsign}(\mu, L_{\text{RKP}}, \text{OPK})$ by programming the random oracle, then generates one-time signature $\sigma_2 = \text{Osign}(\text{OSK}, \sigma_1, \text{OPK})$, outputs $\sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, \text{OPK})$ to $A$. When $A$ receives $\sigma$, he gives a guess $\pi^* \in \{1, \ldots, n\}$.
In the two games above, Game 0 is the actual game between \( S \) and \( A \), and Game 1 is the simulated game in the random oracle model. In game 1, \( r \) is uniformly sampled by \( S \), which is statistical independent from the \( L_{PK} \), then

\[
Pr_{A}[\pi^* = \pi] = 1/n.
\]

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, \((g_2, h, g_2^{a_2}, h^{a_2})\) and \((g_2, h, g_2^{a_2}, h^r)\) are computational indistinguishable, then \( A \) cannot distinguish \( h^{a_2} \) (in game 0) from \( h^r \) (in game 1). Meanwhile, due to the hiding items \( x_1, \ldots, x_n \), \( A \) cannot distinguish \( \{g_1^x h^{a_1-a_x}, \ldots, g_n^x h^{a_n-a_x}\} \) from \( \{g_1^x h^{a_1-r}, \ldots, g_n^x h^{a_n-r}\} \), then we know game 0 and game 1 are computational indistinguishable, which finishes the anonymity proof of TLRS. \( \square \)

**Proof of Linkability**

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

**Proof.** For a PPT adversary \( A \) with possession of the trapdoor \( y \), but does not know the relation between \( g_1 \) and \((g_2, h = g_2^y)\), when \( A \) finished the link game with \( S \) in Definition 3, we assume that \( A \) wins the link game with nonnegligible advantage \( \delta \), that is, \( A \) returned \( k \) TLRS signatures \( \sigma_i = (\sigma_i^1, \sigma_i^2, \mu_i, L_{PK}^i, OPK^i) \), \( i = 1, \ldots, k \) (\( \sigma_i^1 \)s are ring signatures, \( \sigma_i^2 \)s are one-time signatures), satisfying the following requirements:

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

We first prove a statement that, for a list of users’ public keys \( L_{PK} = \{PK_1, \ldots, PK_n\} \) returned by \( \mathcal{J} \mathcal{O} \) with \( PK_i = (g_1^{x_i} h^{a_i}, g_2^{a_i}, \pi(g_1^{x_i} h^{a_i}, g_2^{a_i})) \), any PPT adversary \( A \) generates a valid TLRS signature \( \sigma = SO \) if and only if he queries the \( \mathcal{CO} \) at least once, except for negligible probability \( \varepsilon_0 = negl(n) \).

\[\Rightarrow \quad \text{If } A \text{ gets } SK = (x_i, a_i) \text{ from } \mathcal{CO}, \text{ then } A \text{ can run the TLRS signature scheme to generate a valid signature } \sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, OPK).\]

\[\Leftarrow \quad \text{Assume } A \text{ did not query the } \mathcal{CO} \text{ and } SO \text{ for } L_{PK} = \{PK_1, \ldots, PK_n\} \text{ and finished the TLRS signature over } L_{PK} = \{PK_1, \ldots, PK_n\} \text{ with non-negligible probability } \delta_1. \text{ We first prove that } A \text{ does not know any of the secret keys in } L_{PK}. \text{ Actually, under the hardness of discrete logarithm, } A \text{ cannot compute } a_i \text{ from } TK = g_2^{a_i}, i = 1, \ldots, n, \text{ then the probability of } A \text{ obtaining any of } (x_i, a_i) \text{ is } \varepsilon_1 = negl(n).\]

Next, according to the assumption that \( A \) generates a valid signature \( \sigma = (\sigma_1, \sigma_2, \mu, L_{PK}, OPK) \), then he must have finished the one-time signature \( \sigma_2 \). Since the one-time signature scheme achieves unforgeability, then \( A \)
knows OSK = b except for negligible probability $\epsilon_2 = negl(n)$, we get that $L_{RPK} = \{g_1^{z_1}h_{a_1-b}, \ldots, g_1^{z_n}h_{a_n-b}\}$ and A finished the ring signature $\sigma_1$ with $L_{RPK}$. According to the unforgeability of ring signature, we get that A knows at least one of the correspond $z$ satisfying $g_1^{z_j}h_{a_j-b} = g_1^j$ for $j \in \{1, \cdots, n\}$, except for negligible probability $\epsilon_3 = negl(n)$, which means A gets a solution for $g_1^{z_j}h_{a_j-b} = g_1^j$ with nonnegligible probability $\delta_1 - \epsilon_1 - \epsilon_2 - \epsilon_3$, this contradicts with the hardness of discrete logarithm problems. Then we get that A generates a valid TLRS signature $\sigma \leftarrow SO$ if and only if he queries the CO at least once, except for negligible probability.

According to the fourth requirement that the number of times of $A$ querying $CO$ is $\leq k-1$, and $A$ 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 CO, saying $SK = (z, b)$ from $PK = (g_1^{z_1}h, g_2^{z_2}, \pi(g_1^{z_1}h, g_2^b))$, and 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 $L_{RPK} = \{g_1^{z_1}h_{a_1-b'}, \ldots, g_1^{z_n}h_{a_n-b'}\}$, since $\exists j \in \{1, \cdots, n\}$ s.t. $(x_j, a_j) = (z, b)$, then we have $g_1^{z_j}h_{a_j-b'} = g_1^{z_j}h_{b-b'}$ with $b \neq b'$ and A cannot compute $x$ s.t. $g_1^{z_j} = g_1^{z_j}h_{b-b'}$ under the hardness assumption of discrete logarithm problem, except for negligible probability $\epsilon = negl(n)$, then we have that A successfully forge a ring signature for $L_{RPK} = \{g_1^{z_1}h_{a_1-b'}, \ldots, g_1^{z_n}h_{a_n-b'}\}$ with nonnegligible probability $\delta - \epsilon - k\epsilon_0$, which contradicts to the unforgeability of ring signature, then we finish the linkability proof of TLRS. □

**Proof of Nonslanderability**

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

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

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

From the definition of Link, we know that $OPK^* = OPK = h^{a_0}$, since $PK_\pi$ was not queried by $A$ to $CO$ and $SO$, then $A$ does not know OSK = $a_0$ except for negligible probability $\epsilon = negl(n)$ under the hardness of discrete logarithm problems. Then we know $A$ forged one-time signature $\sigma_2^*$ 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 \( A \), including malicious regulator.

**Proof of Traceability**

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

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

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

Since \( \sigma_2 \) is a valid one-time signature, then \( OPK = h^b \) and \( A \) knows \( OSK = b \) except for negligible probability \( \epsilon_1 \) under the unforgeability of one-time signature, we have \( L_{RPK} = \{g_1^{x_1} h^{a_{1}} \cdot \cdot \cdot g_1^{x_n} h^{a_{n}} - b \} \). According to \( A \) signed \( \sigma_1 \) with \( PK_q \), if \( b \neq a_{\pi} \), then the ring signing public key is \( g_1^{x_{\pi}} h^{a_{\pi} - b} = g_1^{z_a} \), and \( A \) knows \( RSK = z \) except for negligible probability \( \epsilon_2 \) under the unforgeability of ring signature, then \( A \) successfully generates a relation \( g_1^{x_{\pi} - z_a} h^{a_{\pi} - b} = 1 \) 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 \( \text{Trace}(\sigma, y) \neq \pi \), we know \( TK_q \neq OPK = h^{a_{\pi}} \), we can set \( TK_{\pi} = g_1^{z_1} g_2^{z_2} \) without loss of generality, then we get the validity proof \( \pi(RPK_{\pi}, TK_{\pi}) = \pi(g_1^{x_{\pi}} h^{a_{\pi}}, g_1^{z_1} g_2^{z_2}) \), which is a switch proof between \( g_1^{x_{\pi}} h^{a_{\pi}} \) and \( g_1^{z_1} g_2^{z_2} \) with respect to basis \((g_1, h)\) and \((g_1, g_2 h)\). Assume \( \pi(RPK_{\pi}, TK_{\pi}) = (z_1, z_2, e) \), then we have

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

According to the switch proof in section 2.3, we know that \( e \) is computed by

\[
e = H(g_1^{b_1} g_2^{b_2}, g_1^{b_1} g_2^{b_2}), \text{ then we have}
\]

\[
r_1 = z_1 - ex_{\pi}, \quad r_2 = y(z_2 - ea_{\pi}), \quad r_3 = z_1 - ex_{\pi} - es, \quad r_4 = (y + 1) z_2 - e(ya_{\pi} + t).
\]

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

In this section we introduce modification of key generation and spending algorithm in Monero system to achieve traceability of long-term addresses, by usage of switch proof and Diffie-Hellman key exchange.

In the Monero system, user’s long-term public key (address) \( PK = (A, B) \) contains view key \( A = g^a \) and spend key \( B = g^b \), where \( (a, b) \) is the secret key of users. When spending money to user Alice \( (PK = (A, B)) \), spender generates \( r \leftarrow \mathbb{Z}_q^* \), computes and outputs \( (R = g^r, UPK = g^{H(A^i)} , B) \), where \( UPK \) is the new UTXO’s public key. When receiving money, Alice checks \( r_A \) contains view key of switch proof and Diffie-Hellman key exchange. For every \( (PK, R, \pi) \in \text{Receive}(UPK, R_1, R_2, \pi(R_1, R_2)) \): When receiving money:

1. For every \( (UPK, R_1, R_2, \pi(R_1, R_2)) \) in the blockchain, Alice checks \( UPK \cdot H(R_1^i) \cdot B \);
2. If passed then computes \( USK = H(R_1^i) + b \) and receives the money to her wallet, otherwise outputs \( \perp \).
3. Verifier checks the validity of \( \pi(R_1, R_2) \) in every transaction, if passed then outputs 1, otherwise outputs 0.

Construction

- \( Par \leftarrow \text{Setup}(\lambda) \): system chooses elliptic curve \( G \) and a generator \( g \in G \), the regulator generates \( y \in \mathbb{Z}_q^* \) as the trapdoor, computes \( h = g^y \), system outputs \( (G, q, g, h) \) as the public parameters.
- \( (PK, SK, \pi(PK)) \leftarrow \text{Gen}(Par) \):
  1. According to the public parameters \( (G, q, g, h) \), user Alice samples \( a, b \in \mathbb{Z}_q^* \) uniformly, computes \( PK = (A_1 = h^a, A_2 = g^a, B = g^b) \);
  2. Alice proves that \( (g^a, h^a) \) share the same exponent, that is, Alice gives the switch proofs \( \pi(PK) = \pi_{Swit}(A_1, A_2) \) with basis \( (g, h) \) (with soundness for malicious regulator);
  3. Alice outputs her address \( PK = (A_1, A_2, B), \pi(PK) \), and retains \( SK = (a, b) \).
- \( (UPK, R_1, R_2, \pi(R_1, R_2)) \leftarrow \text{Spend}(PK) \): Assume a spender \( S \) wants to spend his money to Alice, he does as follows:
  1. \( S \) generates \( r \leftarrow \mathbb{Z}_q^* \) uniformly, computes \( R_1 = h^r, R_2 = A_2^i \) and \( \pi_{Swit}(R_1, R_2) \) for basis \( (h, A_2) \);
  2. \( S \) computes \( UPK = g^{H(A^i)} \cdot B \), then outputs \( (UPK, R_1, R_2, \pi(R_1, R_2)) \).
- \( USK \leftarrow \text{Receive}(UPK, R_1, R_2, \pi(R_1, R_2)) \): When receiving money:
  1. For every \( (UPK, R_1, R_2, \pi(R_1, R_2)) \) in the blockchain, Alice checks \( UPK \cdot g^{H(R_1^i)} \cdot B \);
  2. If passed then computes \( USK = H(R_1^i) + b \) and receives the money to her wallet, otherwise outputs \( \perp \).
  3. Verifier checks the validity of \( \pi(R_1, R_2) \) in every address, if passed then adds it to the valid address list, otherwise reject;
- \( (PK, SK, \pi(PK)) \leftarrow \text{Trace}(UPK, R_1, R_2, \pi(R_1, R_2), y) \):
  1. For every \( (UPK, R_1, R_2, \pi(R_1, R_2)) \) in the blockchain, regulator computes \( B^* = UPK \cdot g_1^{H(R_1^i)'} \).
2. Regulator searches all the addresses in the blockchain satisfying \( B = B^* \), outputs the corresponding address \((PK, \pi(PK))\), otherwise outputs \(\perp\).

It is easy to know that \( A_1' = h^{ar} = g^{ary} = R_1^r = R_2^r \), then \( UPK \cdot g^{-H(R_2')} = g^{H(R_1')-H(g^{ary})} \cdot B = B \), so we get the correctness of the modified scheme, where regulator can trace the receiver’s address correctly.

The security of the modified scheme contains anonymity and traceability. The anonymity holds for adversary without possession of trapdoor, relies on DH assumption (similar to Monero) and zero-knowledge of switch proof. The traceability holds for any PPT adversary, relies on soundness of switch proof, we omit the detailed proof due to lack of spaces.

**Theorem 17** The modified scheme is correct and secure for any PPT adversary.

## 5 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. Moreover, we give the modification of TBoRP (named TBoRP’) to achieve security against malicious regulators.

### 5.1 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 remains the same as in range proof, for any adversary \(\mathcal{A}\). Soundness of our schemes are different, for TBoRP, soundness holds for any PPT adversary \(\mathcal{A}\) without possession of trapdoors; for TBoRP’ and 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 for TBoRP and TBuRP), he cannot escape from regulation in TBoRP’. We give the formal definition of traceability as follows:

**Definition 18 (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 \(\text{Setup}\) 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. \(\text{Verify}(c, \pi(c)) = 1\).
2. \(\text{Trace}(\pi(c), \text{trapdoors}) \neq a\).
We give the advantage of $A$ in traceability attack as follows:

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

A traceable range proof is traceable if for any PPT adversary $A$, $\text{Adv}_{A}^{\text{trace}} = \text{negl}.$

5.2 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.

We give the introduction of TBoRP in the following:

- $\text{Par} \leftarrow \text{Setup}(\lambda)$: system chooses elliptic curve $G$ and a generator $g \in G$, the regulator generates $y \in \mathbb{Z}_p^*$ as the trapdoor, computes $h = g^y$, system outputs $(G, g, q, h)$ as the public parameters.

- $(L_{PK}, SK, c, \{TK_i\}, \pi(\{c_i\}, \{TK_i\}, \{e_i\})) \leftarrow \text{Gen(Par, a)}$:
  1. According to the public parameters and amount $a \in \{0, 2^n - 1\}$, prover Alice samples $x \in \mathbb{Z}_q$ uniformly, computes $c = g^x h^y$ 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 $c_i = g^{x_i} h^{2^i a_i}, c'_i = g^{x_i} h^{2^{i-1} a_i - 2^i}$, 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 $e_i = H(c_0, \cdots, c_{n-1}, TK_0, \cdots, TK_{n-1}, i)$, 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 $(a, SK = (x_0, \cdots, x_{n-1}))$.

- $\sigma \leftarrow \text{Prove}(SK, c, L_{PK})$: Alice runs the Borromean ring signature for $L_{PK} = \{(c_0, c'_0), \cdots, (c_{n-1}, c'_{n-1})\}$, outputs $\sigma \leftarrow \text{Rsign}(SK, c, L_{PK})$.

- $1/0 \leftarrow \text{Verify}(\sigma, c, L_{PK}, \{TK_i\}, \pi(\{c_i\}, \{TK_i\}, \{e_i\}))$:
  1. Verifier computes $e_0, \cdots, e_{n-1}$, checks the validity of $\pi(\{c_i\}, \{TK_i\}, \{e_i\})$;
  2. Verifier checks $\prod c_i = c$;
  3. For every $i = 0, \cdots, n-1$, verifier checks $c_i / c'_i \approx h^{2^i}$;
  4. Verifier checks the validity of Borromean ring signature $\sigma$, if all passed then outputs 1, otherwise outputs 0.

- $a^* \leftarrow \text{Trace}(\sigma, L_{PK}, y, \{TK_i\}_{i=0, \cdots, n-1})$:
  1. For every $i = 0, \cdots, n-1$, regulator computes $c''_i$
  2. For every $i = 0, \cdots, n-1$, if $c''_i = TK_i$ then outputs $a^*_i = 0$, otherwise outputs $a^*_i = 1$;
  3. Regulator outputs $a^* = a^*_0 + \cdots + 2^{n-1} a^*_{n-1}$ as the tracing result.

The $TK_i$’s validity proof $\pi(\{c_i\}, \{TK_i\}, \{e_i\})$ works as follows:
1. Let $P_1 = \prod_{i=0}^{n-1} TK_i^{-1}$ and $P_2 = \prod_{i=0}^{n-1} (TK_i \cdot c_i)^{-1}$, prover generates $r_1, r_2 \in \mathbb{Z}_q^*$, computes $f = H(h^r_1, (gh)^{r_2})$, then computes $z_1 = r_1 + f \sum_{i=0}^{n-1} e_i(x_i - 2^i a_i)$, $z_2 = r_2 + f \sum_{i=0}^{n-1} e_i x_i$, outputs the proof is $\pi = (z_1, z_2, f)$.

2. Verifier checks $f = H(h^{z_1}/(P_1)^f, (gh)^{z_2}/(P_2)^f)$.

**Proof of Correctness**

**Theorem 19 (Correctness of TBoRP)** For an honest user Alice, assume she completes TBoRP, the prover can trace the hidden amount correctly.

**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 $L_{PK} = \{c_i = g^{x_i} h^{2^i a_i}, e_i = g^{x_i} h^{2^i a_i - 2^i}\}$, 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 $c_i = g^{x_i} h^{2^i a_i} = g^{x_i}, TK_i = h^{x_i} = e_i^y$. When $a_i = 1$, we know $TK_i = h^{x_i-2^i}, c_i' = (g^{x_i} h^{2^i})^{y} = h^{x_i+2^i y}$, then $c_i' = TK_i$ iff $y = -1$, which happens with negligible probability, then we get the correctness of TBoRP. □

Soundness of TBoRP (for adversary $A$ without possession of trapdoor) easily follows the soundness of Borromean range proof, we omit it for brevity.

**Proof of Zero-knowledge**

**Theorem 20 (Zero-knowledge of TBoRP)** TBoRP is computational zero-knowledge for any PPT adversary $A$ (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 $(c_i, TK_i)$ is computational indistinguishable from uniform distribution when $a_i = 0$ or 1. Formally, we prove for $c_i = g^{x_i} h^{2^i a_i}, e_i' = g^{x_i} h^{2^i a_i - 2^i}$ with $c_i/c_i' = h^{2^i}$ is a constant, any PPT adversary $A$ cannot distinguish uniform distribution $U = (r', r)$ from $(c_i, TK_i) = (g^{x_i}, h^{x_i})$ (when $a_i = 0$) or $(c_i, TK_i) = (g^{x_i} h^{2^i}, h^{x_i-2^i})$ (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 $(g^{x_i} u, h^{x_i} u^{-1}) = (g^{x_i+1} h^{2^i}, h^{x_i-2^i}) = (c_i, TK_i)_{a_i=1}$, we know they are computational indistinguishable from $U = (r', r)$ for any PPT adversary $A$ without possession of trapdoor, for every $i = 0, \cdots, n-1$, then we finish the zero-knowledge proof of TBoRP. □
Proof of Traceability

Theorem 21 (Traceability of TBoRP) TBoRP is traceable for any PPT adversary $A$ (without possession of trapdoor).

Proof. For a PPT adversary $A$ without possession of the trapdoor $y$, when $A$ finished the tracing game with $S$ in Definition 18, $A$ generates a commitment $c$ for a hidden value $a$ and range proof $\pi_{TBo}(c) = (\sigma, c, L_{PK}, \pi(\{c_i\}, \{TK_i\}, \{e_i\}))$. We assume that $A$ wins the tracing game with nonnegligible advantage $\delta$, that is, $\pi_{TBo}(c)$ satisfying the following:

$$\text{Verify}(c, \pi_{TBo}(c)) = 1 \text{ and Trace}(\pi_{TBo}(c), y) \neq a.$$ 

According to the soundness of Borromean range proof, we know $c = g^x h^a$ with $a \in [0, 2^n - 1]$ and $c_i = g^x \cdot h^{2^i a_i}$ for every $i = 0, \ldots, n - 1$ except for negligible probability $\epsilon_1$, we set $TK_i = g^{x_i} h^{t_i}$. From the validity proof $\pi(\{c_i\}, \{TK_i\}, \{e_i\}) = (z_1, z_2, f)$ which proves $\prod_{i=0}^{n-1} TK_i^{e_i}$ is a power of $h$ and $\prod_{i=0}^{n-1} (TK_i \cdot c_i)^{e_i}$ is a power of $gh$, then we have $f = H(h^{z_1}/(\prod_{i=0}^{n-1} TK_i^{e_i})), (gh)^{z_2}/(\prod_{i=0}^{n-1} (TK_i \cdot c_i)^{e_i})$, similar to Theorem 16, we know that $f = H(g^{x_i} h^{t_i}, g^{x_i} h^{t_i}) = H(h^{z_1}/(\prod_{i=0}^{n-1} g^{x_i} h^{t_i})), (gh)^{z_2}/(\prod_{i=0}^{n-1} g^{x_i} h^{t_i})$, then we have:

$$r_1 = -f \sum s_i e_i, r_2 = z_1 - f \sum t_i e_i,$$

if $\sum s_i e_i \neq 0$, then we have $f = r_1 (\sum s_i e_i)^{-1}$, which means $A$ computes $f$ before he runs the hash function (query the random oracle), this happens with negligible probability $\epsilon_2$. So we get $\sum s_i e_i = 0$, if there exists $j \neq 0$ then $e_j = -s_j \sum_{i \neq j} e_i s_i$, which means $A$ computes $e_j$ before he runs the hash function (query the random oracle), this happens with negligible probability $\epsilon_3$, then $s_i = 0$ for $i = 0, \ldots, n - 1$. We have $r_3 - r_4 = f \sum e_i (t_i + 2^i a_i - x_i)$, if $\sum e_i (t_i + 2^i a_i - x_i) \neq 0$, then we have $f = (r_3 - r_4) (\sum e_i (t_i + 2^i a_i - x_i))^{-1}$, which means $A$ computes $f$ before he runs the hash function (query the random oracle), this happens with negligible probability $\epsilon_4$, then $\sum e_i (t_i + 2^i a_i - x_i) = 0$.

If $\text{Trace}(\pi_{TBo}(c), y) \neq a$, then there exists $j$ s.t. $TK_j = h^{t_j} \neq h^{x_j-2^a a_j}$, then $t_j + 2^j a_j - x_j \neq 0$, we have $e_j = (x_j - t_j - 2^i a_j)^{-1} (\sum_{i \neq j} e_i (t_i + 2^i a_i - x_i))$, which means $A$ computes $e_j$ before he runs the hash function with nonnegligible probability $\delta - \epsilon_1 - \epsilon_2 - \epsilon_3 - \epsilon_4$. This is a contradiction, so we get $TK_i = g^{x_i} h^{t_i}$ for $i = 0, \ldots, n - 1$ and then $\text{Trace}(\pi_{TBo}(c), y) = a$, which contradicts to the assumptions before, so we finish the traceability of TBoRP. $\square$

Modification In TBoRP, regulator can change the hidden value ($a \rightarrow b$) $c = g^x h^a = g^{x+y(a-b)} h^b$ with trapdoor $y$, this brings potential risk when regulator is malicious or corrupted, we need to modify TBoRP to achieves soundness and traceability against malicious regulators. In this subsection we introduce the construction of TBoRP, which realizes soundness and traceability for adversary with possession of trapdoor, by adding the mirror commitment $d = g^x h^a$.
• \( \text{Par} \leftarrow \text{Setup}'(\lambda) \): system chooses elliptic curve \( \mathbb{G} \) and generators \( g_1, g_2 \in \mathbb{G} \), the regulator generates \( y \in \mathbb{Z}_q^* \) as the trapdoor, computes \( h = g_2^y \), system outputs \((G, q, g_1, g_2, h)\) as the public parameters.

• \((L_{PK}, SK, c, d, \{d_i\}, \{TK_i\}, \pi_1, \pi_2) \leftarrow \text{Gen}'(\text{Par}, a)\):
  1. According to the public parameters \((G, q, g_1, g_2, h)\) and amount \( a \in [0, 2^n - 1] \), prover Alice samples \( x \in \mathbb{Z}_q^* \) uniformly, computes commitment \( c = g_1^x h^a \) and the mirror commitment \( d = g_2^x h^a \);
  2. Alice computes the binary expansion \( a = a_0 + \cdots + 2^{n-1}a_{n-1}, a_i = 0, 1 \) for \( i = 0, \ldots, n - 1 \), samples \( x_0, \ldots, x_{n-1} \) uniformly, satisfying \( x_0 + \cdots + x_{n-1} = x \);
  3. For every \( i = 0, \ldots, n-1 \), Alice computes \((c_i = g_1^{x_i} h^{2^{ai}}, c'_i = g_1^{x_i} h^{2^{ai}-2}) \) and \( d_i = g_2^{x_i} h^{2^{ai}} \), outputs \((L'_{PK} = (c_i, c'_i), d_i)\);
  4. For every \( i = 0, \ldots, n-1 \), Alice computes \( TK_i = h^{x_i-2^{ai}} \) and \( e_i = H(c_0, \ldots, c_{n-1}, d_0, \ldots, d_{n-1}, TK_0, \ldots, TK_{n-1}, i) \), gives validity proof of all \( TK_i \) and \( d_i \): \( (\pi_1, \pi_2) = \pi'(\{c_i\}, \{d_i\}, \{TK_i\}, \{e_i\}) \) that \( \pi_1 \) proves the validity of \( \{d_i\} \) and \( \pi_2 \) proves the validity of \( \{TK_i\} \);
  5. Alice outputs \((L_{PK} = \{L_{PK}', \cdots, L_{PK}^{n-1}\}, c, d, \{d_i\}, \{TK_i\}, \pi_1, \pi_2\) and retains \((a, SK = (x_0, \ldots, x_{n-1}))\).

• \( \sigma \leftarrow \text{Prove}'(SK, c, L_{PK}, d) \): Alice runs the Borromean ring signature for \( L_{PK} = \{(c_0, c'_0), \cdots, (c_{n-1}, c'_{n-1})\} \), outputs \( \sigma \leftarrow \text{Rsign}(SK, c, L_{PK}, d) \).

• \( 1/0 \leftarrow \text{Verify}'(\sigma, c, d, \{d_i\}, L_{PK}, \{TK_i\}, \pi_1, \pi_2)\):
  1. Verifier computes \( c_0, \ldots, c_{n-1} \) and checks the validity of \( \pi_1(\{c_i\}, \{d_i\}, \{e_i\}) \);
  2. Verifier checks the validity of \( \pi_2(\{c_i\}, \{TK_i\}, \{e_i\}) \);
  3. Verifier checks \( \prod c_i \equiv c, \prod d_i \equiv d \);
  4. For every \( i = 0, \ldots, n-1 \), verifier checks \( c_i/c'_i \equiv h^{2^i} \);
  5. Verifier checks the validity of Borromean ring signature \( \sigma \), if all passed then outputs 1, otherwise outputs 0.

• \( a^* \leftarrow \text{Trace}'(\sigma, \{d_i\}, y, \{TK_i\}_{i=0,\ldots,n-1})\):
  1. For every \( i = 0, \ldots, n - 1 \), regulator computes \( d_i^y \);
  2. For every \( i = 0, \ldots, n - 1 \), if \( d_i^y = TK_i \) then outputs \( a_i^* = 0 \), otherwise outputs \( a_i^* = 1 \);
  3. Regulator outputs \( a^* = a_0^* + \cdots + 2^{n-1}a_{n-1}^* \).

Noted that \( \pi_1(\{c_i\}, \{d_i\}, \{e_i\}) \) proves the validity of \( \{d_i\} \) (introduced below), and \( \pi_2(\{c_i\}, \{TK_i\}, \{e_i\}) \) proves the validity of \( \{TK_i\} \), same as the original T-BoRP in the appendix 5.2. Both of \( \pi_1, \pi_2 \) have soundness for malicious regulators (for regulators don’t know the relationship between \( g_1 \) and \( g_2 \)).

1. Prover computes \( R = \frac{\prod c_i^x}{\prod d_i^x} = (\frac{g_1}{g_2})^{\sum c_i x_i} \), generates \( r \in \mathbb{Z}_q^* \) and computes \( f = H((\frac{g_1}{g_2})^r); \)
2. Prover computes \( z = r + f \sum e_i x_i \), outputs the proof \( \pi_1 = (z, f); \)
3. Verifier checks \( f = H((\frac{g_1}{g_2})^z/R^f) \), if passed output 1, otherwise outputs 0.
The TBoRP′ can be divided into two parts, standard commitments $c, \{c_i\}$ and mirror commitments $d, \{d_i\}$, the algorithms for standard are same as Borromean range proof, which has soundness for any PPT adversary $A$, then TBoRP′ achieves the same soundness property. For traceability of TBoRP′, according to the soundness of validity proofs $\pi_1, \pi_2$, we know that even for malicious regulator, he have to generate $d_i, TK_i$ in the right form, then he cannot escape from regulation, we omit the detailed proof due to lack of spaces.

**Theorem 22** TBoRP′ has correctness, soundness, traceability for any PPT adversary, and zero-knowledge for any PPT adversary without trapdoor.

### 5.3 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:

1. $\text{Par} \leftarrow \text{Setup}(\lambda)$: system chooses elliptic curve $G$ and generators $g, h, g_0, \ldots$, $g_{n-1} \in G$, the regulator generates $y_0, \ldots, y_{n/2-1} \in \mathbb{Z}_q^*$ as the trapdoors, computes $h_{2i} = g_{2i}^{y_i}, h_{2i+1} = g_{2i+1}^{y_i}, i = 0, \ldots, n/2 - 1$, system outputs $(G, q, g, h, g, h)$ as the public parameters, where $g = (g_0, \ldots, g_{n-1}) \in G^n, h = (h_0, \ldots, h_{n-1}) \in G^n$.

2. $(A, S, c, \{TK_i\}_{i=0,\ldots,n-1}, \pi(TK_0, \ldots, TK_{n-1}, A)) \leftarrow \text{Gen}(\text{Par}, a)$:
   - According to the public parameters $(G, q, g, h, g, h)$ 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);
   - Alice computes the binary expansion $a = a_0 + \cdots + 2^{n-1}a_{n-1}, a_i = 0, 1$ for $i = 0, \ldots, n - 1$, sets $a_L = (a_0, \ldots, a_{n-1})$;
   - Alice computes $a_R = a_L - 1^n = (a_0 - 1, \ldots, a_{n-1} - 1)$;
   - Alice samples $\alpha \in \mathbb{Z}_q$ uniformly at random, computes
     \[
     A = h^\alpha g^{a_L} h^{a_R} = h^{a_0} g_0^{a_L} h^{a_1} \cdots g_{n-1}^{a_0} h_{n-1}^{a_1} - 1;
     \]
   - Alice samples $s_L, s_R \in \mathbb{Z}_q^n, \rho \in \mathbb{Z}_q$ uniformly at random, computes $S = h^n g^{\rho} h^\alpha$;
   - For every $j = 0, \ldots, n/2 - 1$, Alice computes $TK_{2j} = g_{2j}^{a_{2j}}, TK_{2j+1} = h_{2j+1}^{a_{2j+1}}$, the number of $TK$ is $n$;
   - Alice gives the validity proof $\pi(TK_0, \ldots, TK_{n-1}, A)$ of all $TK$s 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 $A \cdot \prod_{i=0}^{n-1} TK_i$ is a power of $h \prod g_i / \prod h_i$;
   - Alice outputs $(A, S, c, \{TK_i\}_{i=0,\ldots,n-1}, \pi(TK_0, \ldots, TK_{n-1}, A))$.

3. $(T_1, T_2, t, \mu, t, l, r) \leftarrow \text{Prove}(A, S, c, \{TK_i\}_{i=0,\ldots,n-1}, \pi(TK_0, \ldots, TK_{n-1}, A))$:
   - Alice sends $(A, S, c, \{TK_i\}_{i=0,\ldots,n-1}, \pi(TK_0, \ldots, TK_{n-1}, A))$ to verifier;
2. Verifier samples \( y, z \in \mathbb{Z}_q \) uniformly at random, sends them to Alice;
3. Alice 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 Alice;
5. Alice computes \( x, \mu, t, l, r \) and sends them to verifier.

\[ \frac{1}{\mathbb{Z}_q} \rightarrow \text{Verify: we only introduce the verification of } \pi(TK_0, \cdots, TK_{n-1}, A): \]
1. For every \( i = 0, \cdots, n - 1 \), verifier computes \( A \cdot \prod_{i=0}^{n-1} TK_i \) and checks
   the validity of \( A \cdot \prod_{i=0}^{n-1} TK_i \) and \( TK_i \);
2. Verifier continues the rest verification of Bulletproofs;
3. If all passed then outputs 1, otherwise outputs 0.

\[ a^* \leftarrow \text{Trace}\{TK_i\}_{i=0, \cdots, n-1}, y_0, \cdots, y_{n/2-1} : \]
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 \((a_{2j}^{y_j}, a_{2j+1}^{y_j}) = (0, 0)\);
3. If \( TK_{2j+1} \cdot TK_{2j}^{y_j} = h_{2j}^{-1} h_{2j+1} \), then outputs \((a_{2j}^{y_j}, a_{2j+1}^{y_j}) = (1, 0)\);
4. If \( TK_{2j+1} \cdot TK_{2j}^{y_j} = h_{2j} h_{2j+1}^{-1} \), then outputs \((a_{2j}^{y_j}, a_{2j+1}^{y_j}) = (0, 1)\);
5. If \( TK_{2j+1} \cdot TK_{2j}^{y_j} = h_{2j}^{-1} h_{2j+1}^{-1} \), then outputs \((a_{2j}^{y_j}, a_{2j+1}^{y_j}) = (1, 1)\);
6. Regulator outputs \( a^* = a_0^* + \cdots + 2^{n-1}a_{n-1}^* \).

**Proof of Correctness**

**Theorem 23 (Correctness of TBuRP)** For an honest user Alice, she can complete TBuRP, and the regulator can trace her amount correctly.

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

\[ \begin{align*}
  &- \text{When } (a_{2j}, a_{2j+1}) = (0, 0), \\
  &\quad TK_{2j+1} \cdot TK_{2j}^{y_j} = (g_{2j}^{-a_{2j}} g_{2j+1}^{-a_{2j+1}})^{y_j} \cdot h_{2j}^{-a_{2j}+1} h_{2j+1}^{-a_{2j+1}+1} = h_{2j} h_{2j+1}; \\
  &- \text{When } (a_{2j}, a_{2j+1}) = (1, 0), \\
  &\quad TK_{2j+1} \cdot TK_{2j}^{y_j} = (g_{2j}^{-a_{2j}} g_{2j+1}^{-a_{2j+1}})^{y_j} \cdot h_{2j}^{-a_{2j}+1} h_{2j+1}^{-a_{2j+1}+1} = h_{2j}^{-1} h_{2j+1}; \\
  &- \text{When } (a_{2j}, a_{2j+1}) = (0, 1), \\
  &\quad TK_{2j+1} \cdot TK_{2j}^{y_j} = (g_{2j}^{-a_{2j}} g_{2j+1}^{-a_{2j+1}})^{y_j} \cdot h_{2j}^{-a_{2j}+1} h_{2j+1}^{-a_{2j+1}+1} = h_{2j} h_{2j+1}; \\
  &- \text{When } (a_{2j}, a_{2j+1}) = (1, 1), \\
  &\quad TK_{2j+1} \cdot TK_{2j}^{y_j} = (g_{2j}^{-a_{2j}} g_{2j+1}^{-a_{2j+1}})^{y_j} \cdot h_{2j}^{-a_{2j}+1} h_{2j+1}^{-a_{2j+1}+1} = h_{2j}^{-1} h_{2j+1}.
\end{align*} \]

Then we get the correctness of TBuRP. \( \square \)

Soundness of TBuRP easily follows the soundness of Bulletproofs, we omit it for brevity.

**Proof of Zero-knowledge**

**Theorem 24 (Zero-knowledge of TBuRP)** TBuRP is computational zero-knowledge for any PPT adversary \( A \) (without possession of trapdoors).
Proof. For the structure of $TK_{2j} = g_{2j}^{\alpha \cdot a_{2j}} g_{2j+1}^{\alpha \cdot a_{2j+1}}$, $TK_{2j+1} = h_{2j}^{\alpha \cdot a_{2j+1}} h_{2j+1}^{\alpha \cdot a_{2j+1}+1}$ in TBuRP, we prove that, TBuRP is computational zero-knowledge for all $TK_{i}s$ are substituted by $S_i = g_i^{\alpha \cdot a_i}, T_i = h_i^{\alpha \cdot a_i}$. In fact, $TK_{2j} = g_{2j}^{\alpha \cdot a_{2j}} g_{2j+1}^{\alpha \cdot a_{2j+1}} = S_{2j} S_{2j+1}, TK_{2j+1} = h_{2j}^{\alpha \cdot a_{2j+1}} h_{2j+1}^{\alpha \cdot a_{2j+1}+1} = (T_{2j} T_{2j+1})^{-1} h_{2j} h_{2j+1}$.

For every $i = 0, \ldots, n - 1$, we need to prove that $(S_i, T_i)$ is computational indistinguishable from uniform distribution. Formally, we prove for any PPT adversary $A$, cannot distinguish uniform distribution $U = (r', r)$ from $(S_i, T_i)_{a_i = 0} = (g_i^\alpha, h_i^\alpha)$ or $(S_i, T_i)_{a_i = 1} = (g_i^{\alpha - 1}, h_i^{\alpha + 1})$. As in Theorem 20, 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 $g_i$ is a generator of $\mathbb{G}$. We also know that the distribution of $(g_i, h_i, r', r)$ and $(g_i, h_i, r' h_i^{-1}, r h_i)$ are identical. Moreover, again from the DH assumption, we know $(g_i, h_i, r' g_i^{-1}, r h_i)$ and $(g_i, h_i, g_i^{\alpha - 1}, h_i^{\alpha + 1})$ are computational indistinguishable. So we have:

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

Where $g_i, h_i$ are constants, $r, r', \alpha$ are uniform random variables.

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

**Proof of Traceability**

**Theorem 25 (Traceability of TBuRP)** $TBoRP$ is traceable for any PPT adversary $A$ (without possession of trapdoor).

**Proof.** The proof is similar as Theorem 16 and Theorem 21, and is omitted. □

### 5.4 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 is $2^{n_1}$ (can be modified to $2^{n_1} - 1$). To sum up, the total tracing time is $T(n_1) = (2^{n_1} - 1) \cdot \frac{2}{n_1}$, which meets the minimum $T(n_1) = n$ when $n_1 = 1$. When $n_1 = 1$, there will be $n$ trapdoors, together with $2n$ $TK_i$s, which is twice as much as TBuRP, so we choose $n_1 = 2$ as the parameter example ($T = 1.5n$), with $n/2$ trapdoors and $n$ tracing keys. Moreover, the parameter selection is not restricted, $n_1$ can be adjusted (such as 1, 4, 8, ⋅⋅⋅, $n$) to apply for different requirements and regulatory policies.

In the table below we compare TBoRP, TBoRP’ and TBuRP in various aspects, where $T$ refers to tracing time, $n_2$ refers to number of tracing keys ($TK_i$s), extra size refers to the number of elements in $(\mathbb{G}, \mathbb{G}_{\alpha}^*)$ for tracing keys, mirror commitments and their validity proofs, soundness and traceability refer to the corresponding adversary type (for regulators).
6 Conclusion

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, we give the construction of traceable and linkable ring signature (TLRS), which realizes the regulatory function for signers’ 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. In addition, we also introduce modifications to Monero system to realize traceability of long-term addresses. Moreover, we propose the traceable Borromean range proof TBoRP and traceable Bulletproofs range proof TBuRP for the first time to realize the regulatory function for amounts of transactions, with security against malicious regulators by adding mirror commitments (TBoRP’). All the results combined together to become the first construction of fully regulatable privacy-preserving blockchains against malicious regulators.

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

28. Van Saberhagen, N.: Cryptonote v 2.0 (2013)