IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 7, 2018, accepted April 11, 2018, date of publication April 16, 2018, date of current version May 16, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2827163

Unlinkable Coin Mixing Scheme for Transaction Privacy Enhancement of Bitcoin

YI LIU, XINGTONG LIU, CHAOJING TANG, JIAN WANG, AND LEI ZHANG
College of Electronic Science, National University of Defense Technology, Changsha 410073, China
Corresponding author: Xingtong Liu (liuxingtong@nudt.edu.cn)

This work was supported by the National Natural Science Foundation of China under Project 61601476 and Project 61672527.

ABSTRACT Bitcoin combines a peer-to-peer network and cryptographic algorithm to implement a distributed digital currency system, which keeps all transaction history on a public blockchain. Since all transactions recorded on the blockchain are public to everyone, Bitcoin users face a threat of leaking financial privacy. Many analysis and deanonymization approaches have been proposed to link transaction records to real identities. To eliminate this threat, we present an unlinkable coin mixing scheme that allows users to mix their bitcoins without trusting a third party. This mixing scheme employs a primitive known as ring signature with elliptic curve digital signature algorithm (ECDSA) to conceal the transfer of coins between addresses. The mixing server is only able to check whether the output addresses belong to its customers, but it cannot tell which address owned by which customer. Customers do not have to rely on the reputation of a third party to ensure his money will be returned, and his privacy will not be leaked. This scheme needs no modifications on current Bitcoin system and is convenient to deploy by any communities. We implemented a prototype of our scheme and tested it under the Bitcoin core's regtest mode. Security and privacy of our mixing scheme are ensured through the standard ring signature and ECDSA unforgeability.

INDEX TERMS Anonymity, Bitcoin, coin mixing, ECDSA, ring signature.

I. INTRODUCTION

The creation of genesis block in 2009 indicates the birth of Bitcoin, which is regarded as the first digital currency based on blockchain technology [1]. Since then, decentralized digital currencies have drawn much attention from the public due to low transaction fees, global availability and no governmental party. Bitcoin has already surpassed a market capitalization of 150 billion US dollars with a potential of continuous growth. A public distributed ledger called blockchain is used to preserve all transaction history for preventing double-spending and avoiding central domination.

We specify Bitcoin as the whole distributed system and bitcoins as digital currency transferred among customers in the following paper. Every transaction with valid signatures is recorded on the blockchain and can never be tampered with. Users only believe in data on the blockchain instead of trusting powerful third parties like banks or governments. Each user utilizes Bitcoin addresses to receive, preserve and spend bitcoins. All transaction records can be kept on a local storage or on the Internet. A Bitcoin address is the hash value of a public key generated from ECDSA which works as an account in a bank. Unless addresses and transactions can be linked to actual identities in society, Bitcoin is considered as an anonymous financial system. It is especially anonymity that helps Bitcoin spread over the world at an astonishing speed.

However, all data kept on the blockchain is accessible to everyone so as to check the validity of transactions and forbid double-spending. Users have to face a threat that their real identities and financial privacy may be leaked through transaction aggregation and analysis. Private information may be exploited by adversaries to cause serious financial catastrophe. Once an adversary links address information to real identities, he can obtain some financial secrets through analysis of behavior and tracking transfer of funds. Some approaches have been presented to link transactions to a particular user by exploiting suitable heuristics [2]–[5]. Users are suggested to use a new address in each transaction for enhancing privacy, but their real identities still can be discovered through analysis of transaction graph [6]–[8]. Literature [9], [10] present methods to link IP addresses with specific nodes and reveal their owners.

Since Bitcoin has reached a large amount of users and will keep this status for a long future, solutions to eliminate

2169-3536 © 2018 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

privacy threat are urgently in need. Nowadays, there are many privacy enhancing technologies to strengthen users' confidence in Bitcoin. Some commercial coin mixing services [11]–[13] and anonymity enhancing approaches [14] emerged as effective methods to tackle this issue. Nevertheless, there are two main drawbacks lying in these approaches. Some approaches require users trusting another third party to mix bitcoins without stealing money or keeping transcripts. Others utilize another cryptographic framework which is not compatible with current Bitcoin system. In the first situation, there is a possibility that service providers may not return mixed coins or reveal their customers' behaviour for benefits. For example, even though systems proposed in [15] and [16] make promises to not steal funds, customers' anonymity still can't be protected. In the second situation, extra cryptographic algorithms are introduced in consensus protocol such as [17]–[19], which create another new digital currency. The created new currency can't operate with current Bitcoin.

Motivated by the privacy requirement of Bitcoin, we utilize features of ring signature [20] to protect address information of Bitcoin users. Based on ring signature, we design a centralized coin mixing scheme to conceal the map relation between inputs and outputs, which needs no trust of the service provider. We combine ECDSA adopted in current Bitcoin system with ring signature to implement this coin mixing scheme, which does not deviate from fundamental design principles of current Bitcoin system. A prototype of our scheme is implemented and tested with Bitcoin core's regtest mode which is used for Bitcoin application development. The result shows it is compatible with existing Bitcoin system and easy to scale with more users. Our scheme inherits the main advantage of a centralized approach, which is easy to deploy and scales well to a large anonymity sets. We also utilize the same elliptic curve secp256kl as used in Bitcoin to avoid adding other more cryptographic components.

The content of this paper is presented as follows: A brief introduction of major components and function modules in Bitcoin is presented in section 2. Related work of coin mixing service is presented in Section 3. The basic cryptographic primitives and preliminary for our approach are introduced in section 4. Details of our mixing scheme are laid out in section 5. We provide our experiment result and a comprehensive analysis of proposed mixing scheme in section 6. Finally, section 7 concludes this paper.

II. BACKGROUND

In Bitcoin [1], electronic payments are implemented through transactions appended to the blockchain, which indicate that bitcoins transferred from one user to another. Input transactions denote where bitcoins come from and outputs denote who will receive bitcoins. Bitcoin wallets help users manage their addresses to receive and spend bitcoins. A valid signature generated by the corresponding private key of an address is required in a transaction as authority. Addresses hashed and encoded from public keys provide anonymity for users and are suggested to be used only once.

Each transaction can be made up of multiple input transactions and multiple output addresses. Every input transaction referenced to an output of a previous transaction needs to satisfy its redemption condition such as a corresponding signature. An output address has to be assigned with a specified amount of bitcoins. After verified by other nodes in the network, a valid transaction is bundled in a block and then appended to the blockchain. Balance of a user is the sum of bitcoins transferred to his addresses which haven't appeared in other transactions as inputs. All the bitcoins transferred to an output have to be spent in a single transaction referenced to it. Hence, if Alice doesn't want to spend all her money from a previous output, she has to add another output address owned by herself to get change back.

Network nodes who devote their computation resources to verify transactions and bundle transactions to generate blocks are called miners. Every block comprises a bundle of transactions, a timestamp and the hash of its predecessor block. The blockchain preserves the whole history of all transactions, which are organized as a Merkle Tree [21]. It is a special data structure which is only allowed to append and can't be updated or deleted. With the continuous block generation, the length of blockchain keeps growing at a steady speed. A miner who generates a new valid block will be rewarded with some bitcoins. Others will turn to work on this new block discarding unfinished block before. In order to obtain rewards, all miners try their best to solve a computationally difficult mathematic puzzle. Only one miner who is the first to solve the puzzle wins the game and appends his block to the blockchain. This mining process is called proof-of-work(PoW). Bitcoin requires miners searching for a nonce which will generate a hash value with a certain number of zeros at the beginning. The hash function adopted by Bitcoin is SHA-256. The only way to find the appropriate nonce is trying different nonce one by one because it's unpredictable which nonce generates the required hash value. Upon finding a specific nonce whose hash value lower than a designated target, the miner appends this block to the blockchain stored on his local storage and broadcasts it into the Bitcoin network. Miners who receive a newly generated block will check if the block is valid. After passing the verification process, the valid block is appended to these miners' local blockchain. Otherwise, this newly generated block will be discarded.

There is a possibility that two or more valid blocks are generated approximately at the same time because of distributed network and competition mechanism. Due to the latency in a peer-to-peer network, there may exist two or more valid blockchain forks with the same length. These forks incur different states of blockchain which must be pruned to left only one global state. Miners can choose one of these forks arbitrarily and try to generate a new block to extend it. According to the consensus protocol, every miner devotes its resource to extend the longest valid chain based on its local storage. Thanks to the random property of PoW, different forks will grow in different speeds. A longest valid fork with most PoW will emerge and spread to the entire network. After

receiving blocks on the longest chain, miners will update their local storage and turn to work on this longest valid fork. Finally, there is only one valid blockchain left for miners to agree with and keep in local storage.

The security of Bitcoin is ensured by PoW and consensus protocol. Anyone who wants to tamper with a transaction already recorded on the blockchain has to complete all PoW from the block including this transaction to now again. For instance, the change of a transaction will impact its hash value in the Merkle tree. Then the hash value of the entire block is altered. The adversary needs to solve the same computationally difficult mathematic puzzle once more and do the same with all blocks after the tampered transaction. It is an impossible task as long as honest participants take charge of the majority of computation resources. All nodes have to prove that they are real entities through completing some calculating tasks. It protects the whole network from Sybil attacks [22].

Transaction fees is another mechanism to incentivize miners contributing their computation resources and electricity to generate blocks. Funds of all output addresses must be less than that of all input transactions. The difference is going to be collected by miners as transaction fees. Although the amount of transaction fees are decided as users like, at least 0.0001 BTC per 1 kB of transaction size is needed nowadays. All transaction fees in a block will be collected by the miner who first finds it and appends it to the blockchain [23].

The speed of block generation relies on the computation difficulty and the whole computation resources. The computation difficulty is adjusted according to the generation time of previous 2016 blocks, in order to keep a steady speed of block generation. This adjustment keeps block generation at a speed of nearly 6 blocks per hour. The equation to calculate the new target value of next 2016 blocks in Bitcoin is presented as follows [23].

$$T_{new} = T_{prev} * \frac{P_{actual}}{2016 * 10min} \tag{1}$$

T_{prev} represents the target value of previous 2016 blocks, and P_{actual} represents the actual generation time of previous 2,016 blocks. T_{new} is the new target value which will be used to generate the next 2016 blocks.

A Bitcoin address is generated by computing the hash of a public key through SHA-256 and RIPEMD-160 hash functions. Then, it is encoded with a special Base58 after concatenated with a version number and a checksum. Even though it provides a method to receive money without real identities, there is a possibility to link addresses to real identities relying on information on the Internet and blockchain data. A multi-input heuristic method is presented to cluster addresses, in order to recognize different participants [6], [8]. Since a user preserves his funds in multiple unspent transactions, a larger payment will require funds from multiple Bitcoin addresses. These transactions can be combined with real common ownership. Change outputs provide another heuristic way to identify ownership [3], [7]. Until the end of June 2016, 79 percent of all transactions include two outputs, one of which is probably a change output of input transactions [7]. Thereby, it is necessary to take measures for protecting private information in Bitcoin. Related countermeasures to enhance the privacy of Bitcoin transaction are presented in the next section.

III. RELATED WORK

General Bitcoin mixing services [11]–[13] adopt a completely centralized architecture which cause two disadvantages. They may keep records of mixing information mapping input transactions to output addresses or refuse to return mixed coins. [24].

Mixcoin [16] provides accountability to prove a mixing server has initiated a mixing task. Customers have to negotiate with the mixing server about addresses to retrieve funds and other operation parameters. This scheme is compatible with Bitcoin and works as a third party service. It keeps the mixing server accountable if funds of customers have been stolen, but it doesn't solve the problem of exposing address relation to the mixing server.

Blindcoin [25] introduces blind signatures to hide map relationships between inputs and outputs. However, a bootstrapping problem may happen because customers have to publish their output addresses on a public log. It still can not protect bitcoins from being stolen regardless of detection of theft.

CoinSwap [26] allows two participants sending bitcoins through a third party in an anonymous way. It is a fair exchange mixing service which prevents the mixing server from stealing bitcoins. This scheme lacks the ability to ensure that the intermediary can't link input transactions to output addresses.

Blindly Signed Contract [27] combines blind signatures and smart contracts to provide anonymity and security without trusting the mixing server. But it may generate a soft fork in current Bitcoin system due to the requirement of a protocol change. Furthermore, it completes a mixing round task through four Bitcoin transactions and takes nearly 30 minutes.

TumbleBit [28] provides a mixing service which does not rely on the reputation of a third intermediate. It is easy to scale for a great number of customers and has implemented a mixing task for 800 customers. It has a disadvantage that at least two normal Bitcoin transactions are needed in per round.

There are two problems in centralized services. First, users have to trust the mixing server will give their funds back after mixing. Second, users have to trust the mixing server does not store the mapping from inputs to outputs which may be leaked after mixing. Distributed mixing services have emerged to remedy these shortages in centralized systems.

Xim [29] provides a mixing service based on a Peer-to-Peer(P2P) network, which utilizes announcements on the blockchain for aggregating users who would like to take part in a mixing service. The mixing server can't deny what it has done since all data have been stored on the blockchain.

Xim only supports two-party mixing and may take several hours to finish a mixing round, since users must wait for others to response their requests.

CoinJoin [15] generates a joint transaction including all users in a distributed protocol. It provides anonymity for users and ensures funds will be transferred to their addresses, because each user will check the mixing transaction before signing on it. If someone refuses to sign on the mixing transaction, it will cause a problem like DoS attack.

CoinParty [30], [31] introduces multiple threshold transactions to eliminate the disadvantage of group transaction pattern, which is easy to distinguish from standard Bitcoin transactions [15], [32], [33]. It is under the assumption that one third of participants are honest.

CoinShuffle [33] uses a distributed oblivious shuffling protocol to improve the performance of Coinjoin [15]. It works on mixnets [34] and provides anonymity for users under the condition that two or more participants are not compromised. Even an attacker compromises some participants, others will still remain anonymous. CoinShuffle++ [35] improves over CoinShuffle. But the coordination between users may cause communication throughput to grow quadratically [24], [29]. ValueShuffle [36] is an extension of CoinShuffle++, which utilizes DiceMix to implement CoinJoin. It conceals funds amount through credential transaction and protects address information through stealth addresses.

CoinSwap and fair exchange protocols eliminate the risk of theft and keep users anonymous after exchanging coins [12], [29]. Nevertheless, it is unknown whether the way to collect mixing fees is an anonymous method.

Distributed mixing services have disadvantages of deployment and upgrade. Many other digital currencies provide a new way to conceal transaction information of Bitcoin. Customers have to exchange their bitcoins for another currency and exchange back to bitcoins after a certain period. We take ZeroCoin, ZeroCash and CryptoNote as examples in the following.

ZeroCoin [17] and its successor ZeroCash [18] create completely different digital currencies. ZeroCoin introduces a new pattern of transaction to extend Bitcoin. Users have to supply their bitcoins to ZeroCoin system and mint zerocoin before use it. Zero-Knowledge Proofs(ZKPs) adopted by ZeroCoin provides correctness proof without revealing which zerocoin was spent. It ensures unlinkability of transactions.

The design of CryptoNote [19] is the most similar to our research in terms of privacy enhancement. It provides anonymity for users in transactions through ring signatures. Monero [37] is an implementation of this idea and utilizes credential transaction to hide funds amount. Ring signatures consume much precious storage space of the blockchain and a lot of computation resources to verify their validity.

IV. PRELIMINARY

A. RING SIGNATURE

Rivest, Shamir and Tauman proposed a novel digital signature algorithm to reveal secrets in an anonymous way, called ring signature in 2001 [20]. Ring signature conceals the actual identity of a signer in a group of users who form a ring. It provides an anonymous way to sign documents without leaking the identity information. A special group signature which excludes the group manager and process of group construction is a form of ring signature. A ring signature is generated by one of ring members and it can be verified by anyone who owns all ring members' public keys. Nobody has the ability to identify who actually sign it. Unless the signer exposes himself, there is no mechanism for others to find out which one in the ring is the actual signer. All information a verifier can confirm is that someone included in the ring has generated the signature. For any verifiers, the identity of the signer is absolutely anonymous. It is very useful in some particular situations where identity information can not be revealed while secrets must be authenticated with signatures.

When generating a ring signature, the signer doesn't have to prearrange a group of users. After generating a ring signature, the signer doesn't have to delete the ring. There is only one assumption that each ring member has been allocated with a public key as identification which is generated from standard signature algorithms like RSA or ECDSA. A signer aggregates an arbitrary set of public keys and signs on a message with these public keys and his own private key. Then he has to announce all public keys of ring members for verifiers to verify this ring signature. In particular, other ring members may not be aware of a ring signature has been generated utilizing their public keys. They may have never seen or known what the signed message is. Maybe they even don't want to sign it at all.

1) DEFININION OF RING SIGNATURE

Suppose a signer is going to sign a document, he aggregates a group of public keys $\{pk_1, pk_2, \cdots, pk_n\}$ from other participants to generate a ring. All participants corresponding to these public keys are regarded as ring members. Everyone in the ring owns a private key sk_i associated with his public key pk_i. The signer generates a signature using his private key sk_s and other ring members' public keys pk_i, $i = 1, 2, \cdots, n$ and $i \neq s$. To generate a ring signature, general cryptographic algorithms like hash function, symmetric encryption and asymmetric encryption can make up function modules. We take a simple construction as an example which utilizes an asymmetric encryption algorithm RSA to sign on a message and verify its signature. The process of a general ring signature scheme is presented as below.

Key Generation(Gen): Users utilize a polynomial cryptographic algorithm to generate their public keys and corresponding private keys. The input of generation algorithm should be a secret seed k with good randomness. Different public-key cryptography such as RSA and ECDSA can be used to generate appropriate public and private keys for different requirements.

Signature Generation(Sign): Suppose every ring member A_i owns a private key sk_i, and pk_i stands for its corresponding public key. $f_i(x, pk_i)$ denotes a trapdoor

one-way permutation with a message x and a public key pk_i as input. Only A_i knows the private key sk_i and can compute the inverse permutation $f_i^{-1}(x, sk_i)$. A symmetric encryption algorithm $E_k(m)$ is used to encrypt message m with a secret key k. Its inverse permutation $E_k^{-1}(m)$ stands for decrypting cyphertext m with secret key k. $H(m)$ presents the hash value of the message m. Choose an initialization value v, a secret key k and a set of $\{y_1, y_2, \cdots, y_n\}$ as inputs of ring funcion $R(k, v, y_1, y_2, \cdots, y_n)$. A_s stands for the actual signer among ring members $\{A_1, A_2, \cdots, A_n\}$.Then, we choose a set $\{x_1, x_2, \cdots, x_n\}$ except for x_s to calculate $y_i = f_i(x_i, pk_i)$,where $1 \leq i \leq n, i \neq s$. Choose the hash value $H(m)$ of signed message as the secret key k. Pick a random value as v and solve the ring equation $R(k, v, y_1, y_2, \cdots, y_n) = v$ for y_s. Finally, calculate x_s according to the inverse permutation $x_s = f_s^{-1}(y_s, sk_s)$ and private key sk_s. The ring signature σ of message m is denoted by the $(2n + 1)$-tuple as $\sigma = (pk_1, pk_2, \cdots, pk_n, v, x_1, x_2, \cdots, x_n)$

Signature Verification(Verify): Upon receiving a signature σ and its corresponding message m, the verifier calculate a secret key $k = H(m)$ and $y_i = f_i(x_i, pk_i)$, $i = 1, 2, \cdots, n$. Then he checks whether the ring equation $R(k, v, y_1, y_2, \cdots, y_n) = v$ is satisfied. The signature σ is accepted under the condition that the ring equation is satisfied. Otherwise, it is regarded as an invalid signature and will be discarded. For more details, please refer to literature [20]. The ring signature sketch is shown in figure 1.

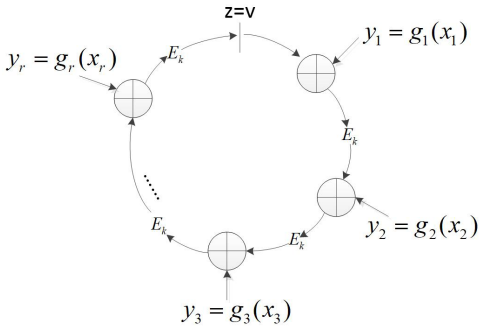


FIGURE 1. Ring signatures.

2) PROPERTIES OF RING SIGNATURE

A ring signature has to satisfy security properties described as below.

Unforgeability: It is impossible for an adversary to generate a valid ring signature under the condition that he can't get any private keys of ring members. Even though he has access to any signatures of a message m through a random oracle, he still can't generate a new one.

Correctness: The verification result of a valid signature must be true if the signature is generated through signing process and is not tampered with during the propagation.

Unconditional Anonymity: The probability for an adversary to recognize the actual signer from all ring members is no more than $\frac{1}{n}$, where n stands for the number of ring members.

There are also some other properties in ring signatures which are not focused in this paper. The anonymity range can be defined arbitrarily by a signer. A ring signature can also work as a group signature without a group administrator in some particular situations.

B. ECDSA

Thanks to the high security and low computation load of elliptic curve digital signature algorithm [38], [39], Bitcoin adopts ECDSA as cryptographic foundation to generate and verify signatures. Elliptic curve cryptographic algorithm has the ability to provide security of high level with shorter key length and lower computation overhead than RSA and DSA. It is described that how ECDSA generate and verify signatures in the following paragraphs. The elliptic curve equation used for the signature algorithm is depicted as below.

$$y^2 = x^3 + ax + b \tag{2}$$

Select an elliptic curve that satisfies the equation above with a specific base point G to generate digital signatures.

1) KEY PAIR GENERATION

A finite field Z_q whose characteristic number is q specifies an elliptic curve E. $G \in Z_q$ stands for a particular base point on elliptic curve E and its order is represented by P. (q, Z_q, G, P) constitutes the domain parameter of ECDSA [40].

Both the private key and its corresponding public key are large integers. Choose a random integer e between 1 and $P-1$ to generate a private key dk. Multiply it with the base point G to generate its public key pk.

2) SIGNATURE GENERATION

Suppose a signer A is going to sign on a message m with his private key dk under the domain parameters (q, Z_q, G, P). The process of generating a signature σ is as follows:

1. Generate a random number r from $[1, P-1]$ and calculate $rG = (x_g, y_g)$.
2. Calculate $n = x_g(\mod P)$. Return to the first step in case of $n = 0$.
3. Calculate $r^{-1}(\mod P)$.
4. Calculate $H(m)$ with SHA-1 algorithm and convert its result to an integer.
5. Calculate $t = r^{-1}(H(m) + dkn)(\mod m)$. Return to the first step, if $t = 0$.
6. (n, t) represents the signature σ of the message m signed by A.

3) SIGNATURE VERIFICATION

Upon receiving a signature $\sigma = (n, t)$ from A, the verifier B obtains the domain parameters (q, E_q, G, P) of A and verify σ using A's public key pk as follows:

1. Check if n and t satisfy $1 \leq n \leq P-1$ and $1 \leq t \leq P-1$.
2. Calculate $H(m)$ with SHA-1 algorithm convert its result to an integer b.
3. Calculate $k = t^{-1}(\mod P)$, $c_1 = bk(\mod P)$ and $c_2 = nk(\mod P)$.

4. Calculate $Q = (x_1, y_1) = c_1G + c_2Pk$.
5. The signature is invalid when $Q = 0$. Otherwise, Calculate $s = x_1(\mod P)$.
6. The signature is valid if and only if $s = n$.

V. MIXING SCHEME

We present details of our coin mixing scheme in this section. Our goal is to provide an efficient scheme for mixing bitcoins without trusting the mixing server. Due to the advantages of centralized mixing services, we utilize a third party as the mixing server to construct a mixing transaction. All input transactions and output addresses of a mixing transaction are aggregated from customers who are willing to enhance their financial privacy. Group transaction in our scheme eliminates the possibility of funds stolen by the mixing server, while ring signature keeps the relationship of inputs and outputs from the mixing server. Although a centralized server is operated in our scheme, it doesn't change the distributed blockchain data of Bitcoin. Successful mixing transactions still need to be verified by the entire network and recorded on the blockchain. The mixing scheme works as a supplementary service rather than an improvement of Bitcoin network. Overview of mixing scheme with three participants as an example is shown in figure 2. Customers send their retrieving addresses signed with ring signatures in UDP packets after sending their unspent transaction indexes.

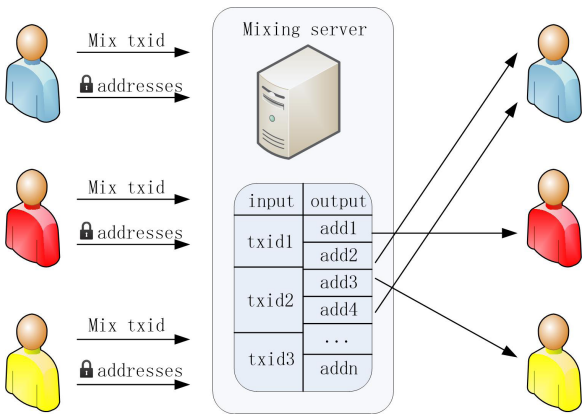


FIGURE 2. Overview of mixing scheme.

A. ECDSA RING SIGNATURE

Elliptic curve digital signature algorithm provides high security and low computation overhead in a shorter key length than other algorithms, while ring signature provides anonymity for the signer [41]. Integrating ECDSA and ring signature, we propose a mixing scheme with high security and efficient process to protect users' privacy. Bitcoin adopts the elliptic curve secp256k1 for address and signature generation. We also utilize elliptic curve secp256k1 in our ring signature scheme in order to use existing functions in current Bitcoin system. This signature process comprises two usual steps as generation of signature and verification of signature.

Let G represents a base point on an elliptic curve E with order P, where P represents a large prime number. H denotes a one-way resisting collision hash function. Parameters P, E, G, and H are all public. Z_q stands for a finite field with P elements. Suppose $\{A_1, A_2, \cdots, A_n\}$ represents a ring member set and they are going to generate a ring signature on a message m under ECDSA. A_s stands for the actual signer among the ring member set, while $1 \leq s \leq n$. The private keys of ring members are $\{dk_1, dk_2, \cdots, dk_n\}$ and $\{pk_1, pk_2, \cdots, pk_n\}$ stands for their corresponding public keys respectively. Each public key satisfies $pk_i = dk_iG$, where $1 \leq i \leq n$. pk_s represents the public key of A_s and dk_s represent his private key.

1) GENERATION OF SIGNATURE

Suppose A_s is going to send a message m with a ring signature σ to a verifier B. A_s generates σ as following steps:
(1) Select a parameter randomly as $e \in_R [1, P-1]$;
(2) Compute the initial state: $(x_s, y_s) = T_s = eG$;
(3) Randomly generate a series $r_i \in_R [1, P-1]$, where $i = s+1, s+2, \cdots, n, 1, \cdots, s-1$.
(4) Compute $c_i = H(m||x_{i-1})$ and $(x_i, y_i) = T_i = r_iG + c_ipk_i$, where $i = s+1, s+2, \cdots, n, 1, \cdots, s-1$. We take $c_1 = H(m||x_n)$ when $i = 1$;
(5) Compute $c_s = H(m||x_{s-1})$ and $r_s = e - dk_sc_s(modP)$;
(6) Finally, construct the signature σ as

$$\sigma = (pk_1, pk_2, \cdots, pk_n, c_1, r_1, r_2, \cdots, r_n).$$

Then send m concatenated with σ to the verifier B.

2) VERIFICATION OF SIGNATURE

When the verifier B receives message m and its ring signature σ, B checks its validity as following steps:
(1) Extract $(Pk_1, Pk_2, \cdots, Pk_n, c_1, r_1, r_2, \cdots, r_n)$ from σ;
(2) Compute $(x_i, y_i) = T_i = r_iG + c_iPk_i$ and $c_{i+1} = H(m||x_i)$, where $i = 1, 2, \cdots, n-1, n$;
(3) σ is a valid ring signature of message m under the condition that $c_{n+1} = c_1$. Otherwise, it's not a valid signature and B will discard it.

3) PROPERTIES OF SIGNATURE

This ring signature scheme based on ECDSA inherits the characteristics of both ECDSA and ring signature, like security in high level, unforgeability, anonymity and undeniability.

Unforgeability: Since it is difficult enough to solve the elliptic curve discrete logarithm problem in existing security protocols, no one has the ability to generate a series parameters $(\{c_1, r_1, r_2, \cdots, r_n\}$ passing through the verification process without knowing one of private keys $\{dk_1, dk_2, \cdots, dk_n\}$.

Anonymity: Anyone can check the validity of a ring signature σ on message m when he receives σ. But nobody is able to recognize the actual signer. Since any one of the ring members has the ability to generate σ, the probability to

identify the actual signer from a ring with n members is no more than $\frac{1}{n}$.

Undeniability: Any other parties can verify a ring signature on message m if there is a conflict. It can be ensured that a message with a ring signature comes from one of the ring members. It is impossible to recognize who is the actual signer among ring members, but who is included in the ring can be confirmed absolutely. Since a valid ring signature has to be generated with at least one private key of ring members, nobody is able to forge a ring signature without any private keys. Even a member never realizes that a ring signature has been generated with his public key, he can't deny the fact that he's one of these ring members.

B. THE MIXING POTOCOL

Our coin mixing scheme utilizes a centralized server to mix bitcoins of customers, avoiding the shortcomings of distributed services such as difficult to deploy and upgrade. Mixing server collects unspent transaction indexes and retrieving addresses from customers to form a group transaction as depicted in figure 2. we will elaborate the communication protocol between one customer and mixing server as an example in this subsection. Figure 6 shows the protocol flow. The whole protocol process comprises three main phases: requesting phase, generation phase and confirmation phase.

(1) In the requesting phase, a customer desiring to mix his bitcoins sends an initial request message m_r to the mixing server. It comprises the number of retrieving addresses $addr_{count}$ which will be included in the output of mixing transaction. After receiving a request from a customer, the mixing server sends back a response message m_b to inform the beginning of mix mission. If the customer receives a message of consenting request, he generates a public key pk_c on the secp256kl elliptic curve and sends it to the mixing server with the transaction $txid$ he wants to mix. After collecting a series of public keys from customers who wants to mix their bitcoins, the mixing server sends a message m_{rp} comprising these public keys $\{pk_1, pk_2, \cdots, pk_n\}$ to every customer. The number of public keys n should be adjusted in consideration of the server performance and customers' demand. The message transmission in requesting phase is presented in figure 3.

(2) In the generation phase, the customer receives a set of public keys $\{pk_1, pk_2, \cdots, pk_n\}$ from the server including his own pk_c and generates Bitcoin addresses $\{addr_1, addr_2, \cdots, addr_m\}$ for getting back his bitcoins. Then he signs a ring signature σ_i on each generated address

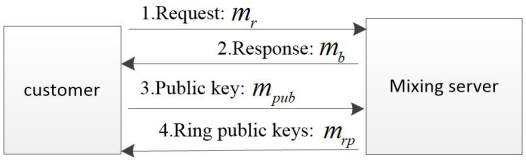


FIGURE 3. Requesting phase.

$addr_i$ respectively through his private key sk_c and public keys received from mixing server. All the addresses concatenated with corresponding ring signatures will be sent to the mixing server one by one in UDP protocol. The source IP address of each UDP packet is replaced by a random IP address to hide their real source IP. Upon receiving an UDP packet including a Bitcoin address, the mixing server check the validity of its ring signature through the public key set $\{pk_1, pk_2, \cdots, pk_n\}$. If it gets through the verification process, the mixing server generates a mixing transaction $Tran_{raw}$ containing all the input transactions and verified addresses with an equal value, and sends it to corresponding customers respectively. Figure 4 shows the message transmission in generation phase.

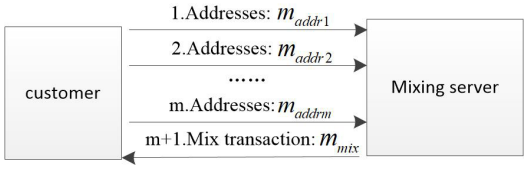


FIGURE 4. Generation phase.

(3) In the final confirmation phase, the customer will check if the mixing transaction $Tran_{raw}$ contains all his input transactions and output addresses with appropriate bitcoins. If all the information included in $Tran_{raw}$ is correct, the customer will sign the mixing transaction using his private key corresponding to input transactions and send signed $Tran_{raw}$ back to the mixing server. After receiving all the valid signatures of input transactions, the mixing server will recombine them to form the final valid mix transaction $Tran_{mix}$ and broadcast it to the Bitcoin network as normal transactions. The message transmission and dissemination in the final confirmation phase is described as figure 5.

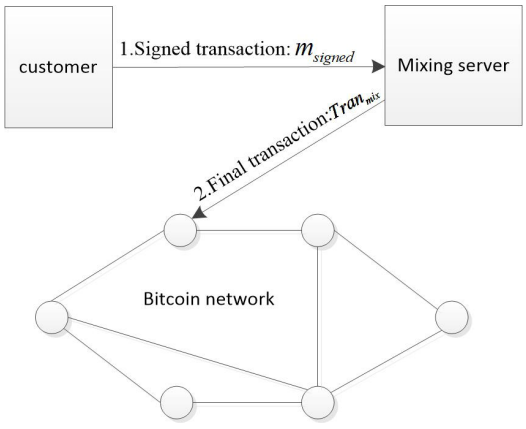


FIGURE 5. Final confirmation phase.

Until the mixing transaction $Tran_{mix}$ is successfully appended to blockchain, the mixing mission is completed and customers have transferred their bitcoins to their new addresses. The mixing server will never know the map relationships between the inputs and outputs of $Tran_{mix}$.

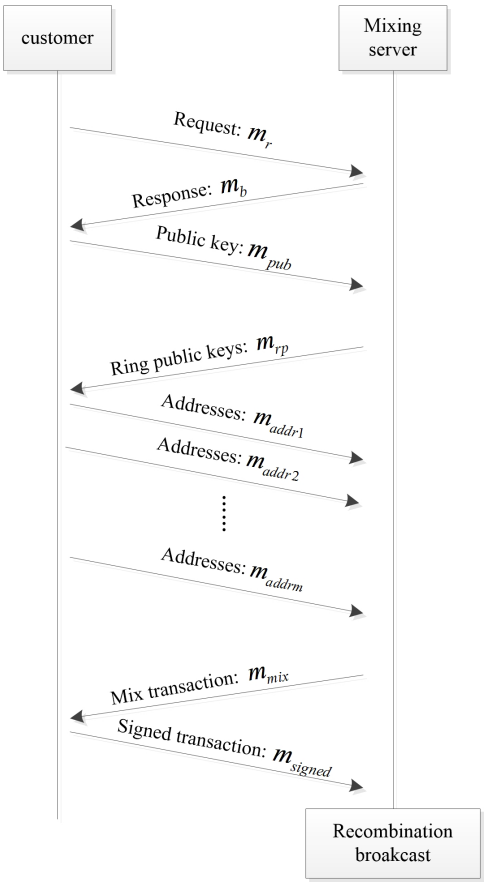


FIGURE 6. Protocol flow of mixing service.

VI. EXPERIMENT AND ANALYSIS

A prototype of proposed mixing scheme is implemented to show its performance and that it's compatible with Bitcoin. Bitcoin developers can use testing tools in Bitcoin Core to test their own applications with reduced risks and limitations [42]. Our mixing scheme is tested under the Bitcoin Core's regression test mode(regtest mode). We utilize API of Bitcoin Core to generate Bitcoin addresses, public keys, private keys and mixing transactions.

In order to make the experimental result easy to observe, we mix bitcoins from three customers as Alice, Bob and Carlo. Each of them chooses an unspent transaction and generates some retrieving addresses to construct a mixing transaction. For ensuring all output addresses retrieve the same amount of bitcoins, let Alice send 20 bitcoins with 2 retrieving addresses, Bob send 30 bitcoins with 3 retrieving addresses and Carlo send 40 bitcoins with 4 retrieving addresses. Then, they send requests for mixing service and do as the protocol described above. In order to confuse the server, they send retrieving addresses through UDP packets with a random source IP address in the generation phase. All the addresses received by the mixing server can not be attached to specific customer due to its ring signature. The mixing server is only able to verify that an address is authorized to be included in the mixing transaction by one of its customers. But it can't tell which address belongs to whom through random fake IP addresses in these UDP packets.

We use a common packet analyzer TcpDump to display UDP packets received by the mixing server. The information of each UDP packet is presented in figure 7. Every UDP packet has a random source IP address as shown in red rectangles. The mixing server only has the ability to check if this Bitcoin address comes from its customer through the ring signature included in the UDP packet. Neither the mixing server nor a third party who intercepts these UDP packets can find their real hosts.

```
ly@ly-virtual-machine:~$ sudo tcpdump -i ens33 -nnnvv udp port 28333
tcpdump: listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
21:43:05.867434 IP (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto UDP (17), length 718)
    159.127.203.24.48952 > 192.168.43.145.28333: [udp sum ok] UDP, length 690
21:43:05.968879 IP (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto UDP (17), length 715)
    70.185.178.79.48952 > 192.168.43.145.28333: [udp sum ok] UDP, length 687
21:43:06.036429 IP (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto UDP (17), length 716)
    147.170.217.194.48952 > 192.168.43.145.28333: [udp sum ok] UDP, length 688
21:43:07.327824 IP (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto UDP (17), length 715)
    116.167.251.55.48952 > 192.168.43.145.28333: [udp sum ok] UDP, length 687
21:43:07.356230 IP (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto UDP (17), length 716)
    153.239.197.152.48952 > 192.168.43.145.28333: [udp sum ok] UDP, length 688
21:43:07.472786 IP (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto UDP (17), length 715)
    120.244.26.109.48952 > 192.168.43.145.28333: [udp sum ok] UDP, length 687
21:43:08.521312 IP (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto UDP (17), length 716)
    56.251.183.41.48952 > 192.168.43.145.28333: [udp sum ok] UDP, length 688
21:43:08.592945 IP (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto UDP (17), length 715)
    57.162.200.42.48952 > 192.168.43.145.28333: [udp sum ok] UDP, length 687
21:43:09.652652 IP (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto UDP (17), length 714)
    78.202.103.149.48952 > 192.168.43.145.28333: [udp sum ok] UDP, length 686
```

FIGURE 7. UDP packets received by mixing server.

The content of the coin mixing transaction is presented in figure 8. Three inputs come from Alice, Bob and Carlo with corresponding amount respectively. There are nine outputs owning the same value of 9.99, two of which belong to Alice. The probability of which address belonging to Alice is the same among these nine output addresses. Only Alice knows which addresses are the retrieving addresses generated by herself. The difference between total input amount and total output values works as transaction fee collected by Bitcoin miners. Thus Alice, Bob and Carlo don't have to worry about their retrieving addresses identified by the mixing server.

1) ANONYMITY

In this experiment, the mixing server has no knowledge about the relation between inputs and outputs. Hence, the probability of an address belonging to Alice is $\frac{2}{9}$, belonging to Bob is $\frac{1}{3}$ and belonging to Carlo is $\frac{4}{9}$. In order to resist the attack of quantitative analysis, all the output addresses retrieve the same value. Since not all customers would like to mix a transaction with the same amount, each input transaction will map to a different number of addresses. If there are n output addresses and m of them belong to a specific customer, the probability of an attacker to guess which address owned by the specific customer is no more than $\frac{m}{n}$. To minimize the probability, a mixing transaction should consist of as many output addresses and input transactions as possible leading to a condition with big n and small m. Ideally, the number of input transactions is equal to the number of output addresses and both of them are big enough.

2) RESISTANT TO DoS ATTACKS

It is easy to launch DoS attacks in a distributed mixing scheme by sending large amounts of request messages and discarding

Coin mixing transaction	
inputs	**outputs**
Txid: 34283be322c6bbb610f59e4acde8aeb3a5b5978382f20115f953b1e7cacd15c3 vout: 1 Amount: 20	address: n28RvuGV8F9ehRmzeiTJN4QYeRQobP6si6 value: 9.99
	address: mm5uQMVSB6Jt8fAxofnX7gYobpZG5EztAa value: 9.99
	address: mizjFMrsmanfLveFkkuWxEBRwPscQgFJiK value: 9.99
Txid: b7515cc3ed9793caad0d2de1030e54314d7e5adfaca7a201e1b95c6b49b384e3 vout: 0 Amount: 30	address: mnqNzm5jRGD4qNPR4c3Yu5fHmdYjBUUaX1 value: 9.99
	address: mmi2ASVGwfiD78ZheBhqiPibJS6PyE95Gi value: 9.99
	address: mu2AnmNhLSw8a5RRmkdf6hZvkDtEntiZxz value: 9.99
Txid: e218975f61d6a90f0149f06cc36675e0b594d317aa41cc92d6ae41d074da177e vout: 1 Amount: 40	address: mvxUHe64TpD18FVnJgCfFNzACBJntUKLq5 value: 9.99
	address: mjMdEeJE218khnQfEd2ZNzGiHmodRiPaRe value: 9.99
	address: mfxsfVs1WzHwDoR5qR8Da5YJdLiQ94cdPP value: 9.99

FIGURE 8. Coin mixing transaction.

response messages. It would cause failure of mixing protocol in a distributed network. In order to resist DoS attacks, a misbehaving customer will be detected and banned because each customer has to build a connection with mixing server. Preventing mixing transactions generated by the mixing server from being broadcasted to Bitcoin network or included in blockchain is another form of DoS attack. But this kind of DoS attack needs to take over at least half of computation resources in Bitcoin network due to the consensus protocol of Bitcoin.

3) SCALABILITY

A Bitcoin group transaction with multiple input transactions and output addresses is generated in one mixing round. The number of inputs and outputs is not restricted by Bitcoin protocol specification. Only the size of a transaction is restricted in Bitcoin protocol. In case that a great number of customers apply for coin mixing at the same time, they can be divided into multiple mixing transactions. Multiple mixing transactions can also realize the function of returning different amount to output addresses like different denominations in physical cash. Improving configuration of a mixing server can alleviate its performance bottleneck.

4) ADVANTAGES

Group transactions prevent funds theft since customers have ability to check whether their addresses are included in outputs. Ring signatures prevent the mixing server from mapping input transactions to output addresses and tracking transactions. Centralized mixing service lowers the expenditure of deployment and upgrade.

VII. CONCLUSION

It is the security of Bitcoin that builds up confidence of users. Transparent transaction records on blockchain cause users to worry about their financial privacy. In order to provide an approach to help Bitcoin users protect their account and transaction information, we present a coin mixing scheme based on ring signature. The proposed scheme implement a ring signature using ECDSA on the secp256kl elliptic curve to be compatible with Bitcoin. Ring signature ensures the mixing server has no ability to keep track of transactions from customers. Our mixing scheme utilizes a centralized mixing server to aggregate input transactions and retrieving addresses from customers to generate a group transaction for mixing. It inherits the advantages of centralized construction such as easy to deploy and upgrade. This scheme just provides a supplementary service which has no impacts on current Bitcoin system so as to be compatible with Bitcoin. Test results of our mixing scheme prototype show it can conceal map relationships between inputs and outputs from the mixing server. At the same time, users need no worry about funds theft and financial privacy leakage.

REFERENCES

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Tech. Rep., 2008.

[2] S. Barber, X. Boyen, E. Shi, and E. Uzun, "Bitter to better how to make bitcoin a better currency," in *Proc. 16th Int. Conf. Financial Cryptogr. Data Secur.*, Kralendijk, Bonaire, 2012, pp. 399–414.

[3] E. Androulaki, G. O. Karame, M. Roeschlin, T. Scherer, and S. Capkun, "Evaluating user privacy in bitcoin," in *Proc. 17th Int. Conf. Financial Cryptogr. Data Secur.*, 2013, pp. 34–51.

[4] M. Spagnuolo, F. Maggi, and S. Zanero, "Bitiodine: Extracting intelligence from the bitcoin network," in *Proc. 18th Int. Conf. Financial Cryptogr. Data Secur.*, Christ Church, Barbados, Mar. 2014, pp. 457–468.

[5] S. Meiklejohn and C. Orlandi, "Privacy-enhancing overlays in bitcoin," in *Proc. Int. Workshops Financial Cryptogr. Data Secur. FC*, Jan. 2015, pp. 127–141.

[6] F. Reid and M. Harrigan, "An analysis of anonymity in the bitcoin system," in *Proc. Secur. Privacy Social Netw.*, 2011, pp. 197–223.

[7] S. Meiklejohn *et al.*, "A fistful of bitcoins: Characterizing payments among men with no names," in *Proc. Conf. Internet Meas. Conf.*, Barcelona, Spain, 2013, pp. 127–140.

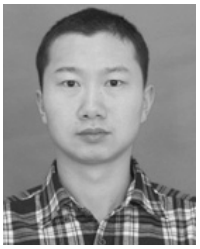
[8] D. Ron and A. Shamir, "Quantitative analysis of the full bitcoin transaction graph," in *Proc. 17th Int. Conf. Financial Cryptogr. Data Secur.*, 2013, pp. 6–24.

[9] P. Koshy, D. Koshy, and P. McDaniel, "An analysis of anonymity in bitcoin using P2P network traffic," in *Proc. 18th Int. Conf., FCFinancial Cryptogr. Data Secur.*, Christ Church, Barbados, Mar. 2014, pp. 469–485.

[10] A. Biryukov and I. Pustogarov, "Bitcoin over tor isn't a good idea," in *Proc. IEEE Symp. Secur. Privacy*, May 2015, pp. 122–134.

[11] (2014). *Bitcoin Fog*. [Online]. Available: http://www.bitcoinfog.com/

[12] (2014). *Bitcoin Wiki:Mixing Services*. [Online]. Available: http://en.bitcoin.it/wiki/Category:Mixing_Services
[13] (2014). *Bitlaundry*. [Online]. Available: http://bitlaundry.appspot.com/
[14] Q. Wang, B. Qin, J. Hu, and F. Xiao, "Preserving transaction privacy in bitcoin," *Future Generat. Comput. Syst.*, to be published.
[15] G. Maxwell, "Coinjoin: Bitcoin privacy for the real world," Tech. Rep., 2013.
[16] J. Bonneau, A. Narayanan, A. Miller, J. Clark, J. A. Kroll, and E. W. Felten, "Mixcoin: Anonymity for bitcoin with accountable mixes," in *Proc. 18th Int. Conf. Financial Cryptograph. Data Secur.*, Christ Church, Barbados, 2014.
[17] I. Miers, C. Garman, M. Green, and A. D. Rubin, "Zerocoin: Anonymous distributed e-cash from bitcoin," in *Proc. IEEE Symp. Secur. Privacy*, May 2013, pp. 397–411.
[18] E. B. Sasson *et al.*, "Zerocash: Decentralized anonymous payments from bitcoin," in *Proc. IEEE Symp. Secur. Privacy*, May 2014, pp. 459–474.
[19] V. Saberhagen. (2013). *Cryptonote*. [Online]. Available: https://cryptonote.org/whitepaper.pdf
[20] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *Proc. 7th Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, Gold Coast, Australia, Dec. 2001, pp. 552–565.
[21] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Proc. Conf. Theory Appl. Cryptogr. Techn. Adv. Cryptol.*, 1988, pp. 369–378.
[22] J. R. Douceur, "The Sybil attack," in *Proc. 1st Int. Workshop Peer-Peer Syst. (IPTPS)*, 2002, pp. 251–260.
[23] K. Kaskaloglu, "Near zero bitcoin transaction fees cannot last forever," in *Proc. DigitalSec*, 2014, pp. 91–99.
[24] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "SoK: Research perspectives and challenges for bitcoin and cryptocurrencies," in *Proc. IEEE Symp. Secur. Privacy*, May 2015, pp. 104–121.
[25] L. Valenta and B. Rowan, "Blindcoin: Blinded, accountable mixes for bitcoin," in *Proc. Int. Workshops Financial Cryptogr. Data Secur. FC*, 2015, pp. 112–126.
[26] G. Maxwll. (2013). *Coinswap*. [Online]. Available: https://bitcointalk.org/index.php?topic=321228
[27] E. Heilman, F. Baldimtsi, and S. Goldberg, "Blindly signed contracts: Anonymous on-blockchain and off-blockchain bitcoin transactions," in *Proc. Int. Workshops Financial Cryptogr. Data Secur. FC BITCOIN, VOTING, WAHC*, Christ Church, Barbados, Feb. 2016, pp. 43–60.
[28] E. Heilman, L. Alshenibr, F. Baldimtsi, A. Scafuro, and S. Goldberg, "TumbleBit: An untrusted bitcoin-compatible anonymous payment hub," in *Proc. 24th Annu. Netw. Distrib. Syst. Secur. Symp. (NDSS)*, San Diego, CA, USA, 2017.
[29] G. Bissias, A. P. Ozisik, B. N. Levine, and M. Liberatore, "Sybil-resistant mixing for bitcoin," in *Proc. 13th Workshop Privacy Electron. Soc.*, Scottsdale, AZ, USA, 2014, pp. 149–158.
[30] J. H. Ziegeldorf, F. Grossmann, M. Henze, N. Inden, and K. Wehrle, "Coinparty: Secure multi-party mixing of bitcoins," in *Proc. 5th ACM Conf. Data Appl. Secur. Privacy*, San Antonio, TX, USA, 2015, pp. 75–86.
[31] J. H. Ziegeldorf, R. Matzutt, M. Henze, F. Grossmann, and K. Wehrle, "Secure and anonymous decentralized bitcoin mixing," *Future Generat. Comput. Syst.*, vol. 80, pp. 448–466, Mar. 2018.
[32] E. Z. Yang. (2012). *Secure Multiparty Bitcoin Anonymization*. [Online]. Available: http://blog.ezyang.com/2012/07/secure-multiparty-bitcoin-anonymization
[33] T. Ruffing, P. Moreno-Sanchez, and A. Kate, "CoinShuffle: Practical decentralized coin mixing for bitcoin," in *Proc. 19th Eur. Symp. Res. Comput. Secur. (ESORICS)*, Wroclaw, Poland, 2014, pp. 345–364.
[34] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–90, 1981.
[35] T. Ruffing, P. Moreno-Sanchez, and A. Kate, "P2P mixing and unlinkable bitcoin transactions," in *Proc. IACR Cryptol. ePrint Arch.*, 2017, p. 824.
[36] T. Ruffing and P. Moreno-Sanchez, "ValueShuffle: Mixing confidential transactions for comprehensive transaction privacy in bitcoin," in *Proc. Int. Workshops, WAHC, BITCOIN, VOTING, WTSC, TA, Financial Cryptogr. Data Secur., FC*, Sliema, Malta, Apr. 2017, pp. 133–154.
[37] S. Noether, "Review of cryptonote white paper," Tech. Rep., 2014.
[38] V. S. Miller, "Use of elliptic curves in cryptography," in *Advances in Cryptology—CRYPTO* (Lecture Notes in Computer Sciences). Berlin, Germany: Springer, 1986, pp. 417–426.
[39] N. Koblitz, "Elliptic curve cryptosystems," *Math. Comput.*, vol. 48, no. 177, pp. 203–209, 1987.
[40] H. Liao and Y. Shen, "On the elliptic curve digital signature algorithm," *Tunghai Sci.*, vol. 8, pp. 109–126, 2006.
[41] Y. F. Chunsg, Z. Y. Wu, and T. S. Chen, "Ring signature scheme for ecc-based anonymous signcryption," *Comput. Standards Interfaces*, vol. 31, no. 4, pp. 669–674, 2009.
[42] (2018). *Bitcoin Core*. [Online]. Available: http://www.bitcoin.org/s



YI LIU received the bachelor's and master's degrees from the National University of Defense Technology, China, in 2011 and 2014, respectively, where he is currently pursuing the Ph.D. degree. His main research interests include information security, electronic payment protocol analysis, and blockchain technology.

XINGTONG LIU received the Ph.D. degree from the National University of Defense Technology, China, in 2014. He is currently a Lecturer with the National University of Defense Technology. His main research interests include information network security, quantum communication, and protocol analysis.

CHAOJING TANG received the Ph.D. degree from the National University of Defense Technology, China, in 2003. He is currently a Professor with the National University of Defense Technology. His main research interests include information security, electromagnetic countermeasure, and software vulnerabilities.

JIAN WANG received the Ph.D. degree from the National University of Defense Technology, China, in 2008. He is currently a Professor with the National University of Defense Technology. His main research interests include wireless network security, quantum communication, and computer network.

LEI ZHANG received the Ph.D. degree in information and communication engineering from the National University of Defense technology (NUDT) in 2010. He is currently an Associate Professor of communication engineering with NUDT. His main research interests include interplanetary networks and space communication security, cyberspace security, industrial control systems security, and Internet of Things.