Multi-party Fair Exchange Protocol with Smart Contract on Bitcoin

Lijuan Guo[1], Xuelian Li[1], and Juntao Gao[2]
(Corresponding author: Lijuan Guo)
School of Mathematics and Statistics, Xidian University[1]
No. 2 South Taibai Road, Xi'an, Shaanxi 710126, China
(Email:lijuanguo16@163.com)
State Key Laboratory of Integrated Services Networks, Xidian University[2]
Xi'an, Shaanxi 710071, China
(Received Sept. 17, 2017; revised and accepted Dec. 8, 2017)

Abstract

Traditional multi-party exchange protocols need a third party to ensure fairness. It can bring some communication costs and cryptanalytic attacks. In recent years, researchers have focused on blockchain to design a fair exchange protocol without a central authority. So far, there are a few works on fair exchange protocols for any topology based on bitcoin. This paper puts forward a decentralized protocol for star topology based on the bitcoin, that is, our protocol does not contain a third party. Communication costs, information disclosure, and cryptanalytic attacks are not considered for the third party. These features greatly reduce the burden and increase the efficiency of the protocol. To guarantee the fairness, a commitment scheme is provided. And the proposed protocol constructs an ideal function as a smart contract. The bitcoin is automatically transferred in the limited time instead of manual operations. Security analysis shows that our construction can guarantee fairness, resist double spending and sybil attack. Meanwhile, the proposed protocol enjoys high efficiency. Moreover, with a slight modification, our protocol can be extended to apply to any topology.

Keywords: Bitcoin; Compensation; Fair Exchange; Smart Contract; Topological Construction

1 Introduction

Secure multi-party computation protocols originated from the work of Yao [27] and have evolved and expanded by Goldreich *et al.* [14]. Following the protocol, a group of participants can work together to achieve their goals by their private inputs. Note that the participants do not trust each other, therefore, a basic requirement is that any participant cannot obtain more messages about other participants' inputs than they would learn when executing the protocol. For an honest participant, it is not fair if dishonest participants disappear or terminate the protocol after receiving their desired results instead of following the protocol to send messages to the expected participants. In this case, it is impossible to force the dishonest participants to send their private information, that is, the fairness is lost. Eslami *et al.* [11] propose a secure group key exchange protocol in the presence of dishonest participants. A third party is usually introduced to deal with this dilemma. Pagnia *et al.* pointed out that it is out of the question to ensure strong fairness of the protocol without the third party because the dishonest participants maybe vanish after receiving the final part of honest participants' messages without transmitting their own final part [22]. Of course, there are also some researches on other aspects. Lal and Das [21] propose a security analysis of protocols using action language. Chang *et al.* [28] propose a privacy preserving protocol in the multi-party model. On the other hand, researchers have been focusing on designing the protocol which applies to various topology [10, 20]. There are four common topologies, namely, ring topology, sequential topology, star topology and mesh topology. Without exception, they all need the third party to guarantee fairness.

But the third party increases communication costs and is vulnerable to the cryptanalytic attack. For example, the Sybil attack is possible if most participants are dishonest. In addition, this process will produce high transaction costs for the micro payment. More importantly, the multi-party protocol also does not guarantee strong fairness in the case of most participants being dishonest.

As early as 1981, some researchers have tried to solve the problems in the electronic coin field, such as privacy, security, fairness and inclusiveness and so on. But, due to the existence of the third party, it is difficult to design a protocol as a solution to these problems simultaneously. Excitedly, a distributed digital currency of bit-

coin is proposed in 2008. Bitcoin has attracted a lot of attention because it has no an authority center to control transactions. The underlying technology of bitcoin is the blockchain technology. One of the features of the bitcoin is the anonymity. Participants are identified by the hash value of the public key. Therefore, it is difficult to link the transaction with the participant spending the money. Another major feature is that transactions are open and transparent so that anyone has access to it on the blockchain. Fairness is usually guaranteed by paying some compensation to the honest participants when the dishonest participants fail to execute the protocol. Meanwhile, honest participants will not lose any bitcoins. Kılınç *et al.* put forward that we do not need to learn how much value the output will be before assessment, and they think that the method for obtaining fairness with bitcoin is inappropriate [20]. However, The Kılınç *et al.*'s problem can be settled as long as the compensation payment is agreed by the participants. Meanwhile, convenient conditions are provided for designing protocols that apply to any topology. In some topologies, participants may need to execute the protocol in the default order. The proposed protocol significantly improve efficiency compared with the traditional multi-party protocols [10, 20] which suitable for any topology. So far researchers put forward a lot of protocols but most protocols only apply to the mesh topology. However, there are a few works based on bitcoin to apply to any topology.

In recent years, security is studied frequently on bitcoin [4,12,17]. Maxwell proposes the zero knowledge contingent payment. It solves the problem of buying a settlement to an NP-problem with bitcoin. Juels *et al.* [18] buy private keys to the specified public keys with Turing-complete scripting language in the protocol. The platform is relatively complex. Kiayias *et al.* [19] propose a formal model with compensation to achieve security by using a shared global transaction ledger. Ruffing *et al.* [25] put forward a completely decentralized non-equivocation contracts by the penalty mechanism. These works explore how to design a general fair exchange protocol with a penalty. We prefer a specific application scenario, but the general theory is not completely applicable in different scenarios. Researchers also have focused on complex financial transactions on blockchain. Bentov *et al.* [7] propose a lottery protocol based on the definition of the ideal primitive. One drawback is that the winner is randomly selected among all participants. Andrychowicz *et al.* [1] design a winner function in which the winner is decided by the input of all participants. The lottery protocol which describes the input and output of the transaction through scripting language is given based on the commitment scheme. Bartoletti *et al.* [5] also propose a lottery protocol but they make a contribution to the constant deposit. However, many tournaments need to be held to get the winner if the number of participants is large.In brief, researchers have paid more attention to lottery protocols [1, 5, 7]. At present, there are few works about e-commerce of using bitcoin [16, 22]. Goldfeder *et al.* [13] designed an escrow protocol which has a mediator to deal with disputes. This approach is similar to the protocol with an optimistic third party. Additional costs and attacks may be added to the protocol. Zheng presents a reputation system that deals with small amount of payment [29]. Strictly speaking, the above protocols only apply to the mesh topology.

Time-lock is also introduced in the bitcoin transactions. The functions change with different protocols. Back *et al.* [3] propose a lottery protocol that is fair and secure. Time-lock is used to refund the deposit if the protocol is terminated maliciously. Andrychowicz *et al.* [1] advocate compensation for the honest participants beyond the limited time. Our ideal function F_{Ledger}combines these two functions. The function F_{Ledger} first uses time-lock to make compensation for honest participants whenever dishonest participants appear and last time-lock is used to refund the deposit and transfer the consumption funds.

The three largest bitcoin trading platforms in China, Huobi, Bitcoin China and OKcoin, provides bitcoin market, bitcoin price, litecoin market and other digital currency trading. Users can store bitcoin safely on the platform. There is an optimal balance between high security and the convenience of users. However, currently, bitcoin has not been able to be used on a large scale in actual transactions. On the one hand, the law is not established about the trading platform of virtual currency in China yet. The so-called "virtual currency" such as bitcoin has increasingly become a tool for money laundering, drug traffic, smuggling and illegal fund-raising activities. On the other hand, the blockchain technology is still undeveloped in the security aspect. If users are increasing sharply, the security of the system will be threatened.

Over these considerations, we propose a multi-party fair exchange protocol with smart contract on bitcoin. Our fair exchange protocol is an online B2C based on bitcoin. Our protocol is also inspired by the commitment scheme [1] and function [2]. Blockchain is characterized by its nature of decentralization, anonymous, information sharing. Based on these features, the advantages of our proposed protocol are manifold.

1) The proposed protocol strongly relies on blockchain which is no central controller. If there is a third party, we must consider communication costs, information leakage and cryptanalytic attacks for it. Our protocol avoids these problems. Meanwhile, we do not employ some general methods, such as zero knowledge compilers and oblivious transfers, therefore, our protocol has a high efficiency.

2) We construct an ideal functionF_{Ledger} with a counter as a smart contract. Meanwhile, a commitment schemeF_{cs} is proposed based on the scheme in [1]. Our protocol is a $(F_{Ledger,F_{cs}})$–hybrid model. Some protocols [1, 5, 13] have also taken advantage of the ideal of deposit and time-lock, but none of them have given a detailed description of the smart contract.

Our protocol not only gives a specific process but also improves security and efficiency, that is, not only can fairness be achieved but it can be executed automatically within the specified time.

3) The proposed protocol is a star topology. Some consumers quit will not affect other consumers. Finally, the protocol can be extended to any topology. Until now, no other constant-round protocol can offer no center,($F_{Ledger,F_{cs}}$)–hybrid model and any topology with a slight modification simultaneously. And there is no compromise on security, that is, protocol provides fairness and can resist forgery attack, double spending and sybil attack.

The rest of this paper is given as follows. Section 2 introduces bitcoin transactions and some symbols. Section 3 shows a commitment scheme. Section 4 proposes fair exchange protocol with compensation and presents an ideal functionF_{Ledger} which can be shared by all participants in an open and transparent manner. Section 5 gives the security analysis. Section 6 describes how to extend the protocol to any topology. Section 7 performs a protocol comparison. Section 8 provides a brief conclusion.

2 Preliminaries

Bitcoin system is a decentralized system that allows participants to exchange virtual currency anonymously. All bitcoin transactions are recorded on the blockchain (also called ledger). These data are open and transparent and everyone has access to it. Recently researchers are devoted to expanding the application from the simple transfer currency to complex financial transactions on the blockchain. The script language is relatively comprehensive. But it is not Turing-complete, one of reasons is to avoid denial of service attacks.

Bitcoin structure contains many nodes called miners. The transactions are collected by miners who participate in the calculation of proof of work to produce blocks. Miners attempt to produce a block, containing the previous block's hash value, by calculating the hash function value satisfying the current transactions' data. If one or more new blocks are formed on top of the longest chain simultaneously, they appear parallel branches. If it happens, miners must choose a branch to continue mining process. This contradiction is resolved when one branch becomes longer than the other branches and miners continue mining in the longer branch. Therefore, it is very difficult if adversaries want to mine a new alternate branch. The probability of success decreases exponentially with the number of new blocks on top of the longest chain. Transactions are confirmed if six blocks have been added to the block (It is about 60 minutes).

A transaction is the basic component of the ledger. The transaction may have one input and one output, multiple inputs and one output, one input and multiple outputs or multiple inputs and multiple outputs. We assume that an address is a hash of public key. Each participant can execute a bitcoin transaction, sending bitcoin from one address to another address. In order to illustrate the principle we give two transactions in Table 1.

Table 1: Simple form of transaction

T_a
in:
in-script:sig(.)
$out-script(depict,\sigma):vek(depict,\sigma)$
$value:v_a$
lock-time:t

T_b
in:T_a in-script:sig(.)
out-script(...):...
$value:v_b$
lock-time:t

The in-script of the transaction T_a is a signature, and the out-script is a validation algorithm. The transaction T_a transfers a value v_a. Moreover, there is a lock-time t that tells us when the transaction is over. Transactions like T_a are called standard transaction. Anybody can spend an amount of v_a bitcoin as long as she/he can satisfy the specific rules in T_a's out-script. The transaction T_b contains a list which is the cryptographic hash of the whole T_a,and in-script contains values to evaluate to true on the out-script of T_a. Then the v_a bitcoin is transformed from the transaction T_a to a new transaction T_b,and T_a cannot be redeemed again. The transaction T_b can be redeemed by meeting its out-script. Now parametersf_x, w_x are given,where f_x is a description function and output is a Boolean function,w_x is the number of bitcoins that is are transformed from one address to another. We give a more specific description, that is, a transaction is in the form $T_b=(T_a,f_b,w_b,\sigma_b)$, where $[T_b]=[T_a,f_b,w_b]$ is defined as *depict*. σ_b is considered to be a witness that is used to evaluate the correctness of f_b on T_b. The witness can be simplified as a signature. Transaction T_b is valid when f_b's evaluation of the input T_a is correct.To describe simplicity,σ represents the witness and *depict* stands for $[T_x]$ of the current transaction in the subsequent scripting language.

There are other styles of bitcoin transaction. A transaction may have multiple inputs and outputs. It is given in the Table 2. The transaction has multiple outputs but only one out-script and one value, and outputs can be independent redeemed. We ignore the fact that there are multiple out-scripts because it will not be used in our paper. Therefore, we should specify which output is redeemed. A suitable in-script must be provided for each of

them if a transaction is redeemed using multiple outputs of the above transaction as inputs. In order to ensure success of the transaction, the sum of all outputs values should be equal to or less than the sum of all inputs.

Table 2: General form of transaction

T
in[0]:T_0
....
in[n]:T_0
in-script:W_0
....
in-script:W_n
out-script(...):...
value:v
lock-time:t

2.1 Symbols

In this section we describe some notations in the paper.

- M: Merchant;
- P_i: Customer, where $i \in \{1, 2, \cdots, n\}$;
- $H(.)$: Collision resistant one-way hash function;
- (pk_j, sk_j): Public and private key of the j-th participant, where $j \in \{1, 2, \cdots, n\}$;
- (pk'_j, sk'_j): Updated public and private key of the j-th participant, where $j \in \{1, 2, \cdots, n\}$;
- $sig_j(.), vek_j(.)$: The RSA signature on message with private key sk_j and verification of message with public key pk_j,where $j \in \{1, 2, \cdots, n\}$;
- $M^1, \cdots, M^n$: They are unredeemed transactions which only can be redeemed by the merchant;
- D^i, C^i: They are unredeemed transactions which only can be redeemed by the customer P_i, where $i \in \{1, 2, \cdots, n\}$;
- s, s_i: The unique secret of merchant and the i-th customer respectively where $i \in \{1, 2, \cdots, n\}$;
- m, m_i: Blinded secret of merchant and the i-th customer respectively where $i \in \{1, 2, \cdots, n\}$;
- $commit^i, deposit^i, open^i$: Merchant creates transactions for the i-th consumer where $i \in \{1, 2, \cdots, n\}$;
- $commit_i^M, deposit_i^M, open_i^M$: Consumer P_i creates transactions for merchant M,where $i \in \{1, 2, \cdots, n\}$;
- T: The maximum delay time in which transactions appear on the ledger;
- BTC: Bitcoin.

3 Models

We will consider some scenarios in which a merchant has some information/goods and many consumers intend to buy it,or the first class agent wants to expand multiple second class agents simultaneously. The proposed protocol is applicable to the scenario that information is sent to multiple participants at the same time, and participants do not know each other but they know how many people are involved in the protocol. Obviously, the efficiency of transmitting to multiple participants simultaneously is higher than the efficiency of transmitting to a user. For convenience, we take the merchant and consumer as an example. A merchant is trade with multiple consumers simultaneously. It is a star topology. The proposed protocol provides the following security properties. Sun *et al.* [26] also proposed a multi-receiver protocol but it is based on chaotic maps with privacy protection.

Fairness. Once the protocol ends, either all participants have the desired information, or none of them can receive it. There are three main characteristics.

1) A malicious merchant cannot gain bitcoins from an honest consumer unless he creates a proper open transaction.

2) A malicious consumer cannot gain desired information from the merchant if he refuses to pay bitcoin.

3) They not only cannot obtain the desired information but also lose the deposit if malicious participants conspire to try to cheat honest participants information or BTCs.

Resistingdoublespendingattacks. The same transaction cannot be redeemed more than once.

Resistingsybilattacks. It does not work even if an adversary creates lots of fake identities.

We assume that the merchant and consumer are connected through insecure channels. Accordingly, a transaction may be intercepted or tampered with. Participants (including the merchant and consumer) and the ledger are connected with secure channels. This problem of transaction malleability [1] must be considered in designing protocol. In Section 4, we propose a protocol that is secure even though an adversary gets all the transaction information before posting on the ledger.

3.1 Commitment Scheme F_{cs}

In [1], the commitment scheme solves the problem of standard commitment schemes which are not able to force a committer to open his real secret if he/she terminates before *open* transaction. The protocol [1] requires each committer to pay some BTCs as deposit. The deposit will be sent to other participants if the committer refuses to open the promise within the specified time. There are three phases: pre-condition phase, commitment phase

and open phase. Each participant has the same commitment, that is, the number of deposits is same. Our commitment scheme is inspired by the scheme in [1]. The proposed scheme has only two phase and commitments are different between merchants and consumers from that in [1]. Our commitment program has a distinctive feature, that is, a consumer has agreed to take part in the protocol, but he may have not enough money or lose interest in the deal in commitment phase. If it happens, he can quit the protocol. Other consumers will not be affected because their transactions are independent.

We now define the commitment scheme. First of all, the ledger has n unredeemed transactions $M^1, \cdots, M^n$ which only can be redeemed by the merchant and has one output-script. However, multiple output transactions are required. In fact, one output-script can contain multiple output transactions in the real world in order to avoid a complex description of the script. The ledger also has n unredeemed transactions D^i which only can be redeemed by the consumer $P_i, i \in 1, .., n$ independently. And the commitment phase has time limit. The specific description of the commitment scheme is shown.

Pre-condition:

1) The merchant M has a key pair (pk_M, sk_M) and the consumer has a key pair $(pk_i, sk_i), i \in \{1, .., n\}$.
2) The ledger has n unredeemed transactions $M^1, \cdots, M^n$ which only can be redeemed by the merchant and the sum of value $v = dn$ BTCs. The ledger also contains n unredeemed transactions $D^1, \cdots, D^n$ which only can be redeemed by the consumer $P_1, \cdots, P_n$ and the value is d BTC, respectively.

Commitment phase:

1) The merchant M computes $h = H(m)$. Then he posts the transactions $commit^1, .., commit^n$ on the ledger. The transactions $M^1, \cdots, M^n$ are used as input. Consumer P_i computes $h_i = H_{m_i}$. Then he posts the transaction $commit_i^M$ on the ledger and the transaction D^i is used as input, *where* $i \in \{1, .., n\}$. The hash value is a part of the commitment.
2) If some transactions $commit^i$ from M are not posted on the ledger at the end of time T, or some of them are wrong. Then the protocol is cancelled. If a transaction(or more)$commit_i^M$ from P_i is not posted on the ledger at the end of time T. For simplicity,we assume that there is a consumer P_1 who does not post the transaction $commit_1^M$. This indicates that P_1 gives up the deal.
3) The merchant M creates the transactions $deposit^1, \cdots, deposit^n$,signs them and sends the transaction $deposit^i$ to P_i, where $i \in \{1, .., n\}$. The transaction $deposit^1$ will not be created if P_1 does not post transaction $commit_1^M$ in step4. P_i stops the deal if P_i has not received $deposit^i$ by the end of the time $2T$. Consumer P_i creates the transaction $deposit_i^M$, signs it and sends $deposit_i^M$ to the M,respectively. M has not received $deposit_j^M$ by the end of the time $2T$. It marks that P_j stopped protocol, where $j \in \{1, .., n\}$.

4 Fair Exchange Protocol with Compensation

Loosely speaking, the proposed fair exchange protocol has the following features.

1) Participants can take part in the protocol only if he has enough BTCs.
2) No honest participant needs to pay a penalty. Honest participants will obtain the desired information or be compensated as long as the protocol is executed correctly.
3) If an adversary and/or dishonest participant replace(s) the secret but honest participants reveal secret in the right way, then the honest participants are compensated accordingly.
4) Transactions will not be affected between consumers and merchants even if there are dishonest participants. Meanwhile, a consumer's quit does not affect other participants because consumers are independent.

We construct a fair exchange protocol with compensation in a mixed model(F_{Ledger}, F_{cs}). The commitment scheme F_{cs} makes sure that each participant has enough BTCs to make a promise. In other words, each participant must have a number of BTCs that are required to participate in the protocol. The ideal function (F_{Ledger} (It will be presented in Section 4.2.)ensures that we provide fairness. In the following, we assume that all the participants are rational, that is to say, they do not want to lose their own interests. Therefore, participants will not deliberately delay time to post transitions on the ledger. Moreover, consumers also need to earn others BTCs in order to purchase information/goods from the merchant. These BTCs come from unredeemed transactions C^i which only can be redeemed by the consumer $P_i, i \in 1, ..n$ and the value is x BTCs, respectively. In the bitcoin system, key pair is updated in each new transaction. In the commitment scheme F_{cs}, every consumer P_i has a blinded secret $m_i, i \in 1, ..n$ and the merchant M has a blinded value m. For the sake of simplicity, blinded secret is denoted as $z \in m, m_i$ and the committer sends blinded secret to the corresponding receiver in the execution phase. Honest participants keep z secret until the transaction *open* in the execution phase. Each participant plays a role of the committer. If the committer is honest, an adversary

would not able to get any valuable information about the secret before opening the transaction. Every recipient can ensure that commitment can only be opened in one way and the secret cannot change with the committer. If committer is trying to cheat or an adversary tampered with the information, every recipient can terminate the protocol. If the committer refuses to execute open transaction, his deposit is transferred to the appropriate recipient as compensation. Therefore, the rational participants will open their commitment in the specified time T in order not to lose their BTCs. The process of protocol is described as follows.

Pre-condition phase:

1) Every participant P_j has a key pair$(pk_j, sk_j), j \in \{1, \cdots, n, M\}$, respectively.
2) The ledger contains unredeemed transactions C^i which only can be redeemed by the consumer $P_i, i \in \{1, \cdots, n\}$ and the value is x BTC(s).
3) Each consumer P_i generates a new key pair $(pk'_i, sk'_i), i \in \{1, \cdots, n\}$ and the merchant M generates a new key pair (pk'_M, sk'_M). They send public key to all other participants.

Commitment phase:

1) All participants must perform the commitment scheme. Assume that the current time is t. This phase ends at the time $t + 2T$.
2) If $h_i = h_j$ for $i \neq j$, the participants ofP_i/P_j abort protocol.

Execution phase:

1) Consumer P_i posts transaction $consume_i^M$ on the ledger using transaction C^i as input. If some transactions are not posed on the ledger at the end of the time $t + 3T$. The merchant M signs appropriate transaction $deposit_i^M$ and sends it to ideal function F_{Ledger}.
2) The merchant M posts the transactions $open^1, .., open^n$ on the ledger and the consumer P_i posts the transaction $open_i^M$ on the ledger, respectively. Meanwhile, they reveal secrets.
3) If a transaction $open^i$ does not posted on the ledger within the time $4T$, P_i signs $deposit^i$ and sends it to ideal function F_{Ledger}. This process is the same for the merchant.

Our protocol is composed of three parts. Pre-condition phase prepares with all pre-protocol information, enough money and public messages. Commitment phase performs commitment scheme. Step 2 is to resist a copy attack in the execution phase. For example,P_i makes a commitment to his hash h_i then P_j promises with the same hash $h_i = h_j$. P_j does nothing until P_i reveals his secret m_i. Then P_j reveals the same secret $m_i = m_j$. During execution phase, consumers post transactions $consume_i^M$ on the ledger. Ideal function F_{Ledger} is used if some transactions go wrong. Finally, all participants perform transaction *open* to reveal secret.

Suppose $s, s_i \in \{0,1\}^*$. We define $z = (r_1||(s/s_i)||r_2)$ where r_1 and r_2 are randomly selected in $\{0,1\}^{k/2}$. The receiver verifies whether $H(z)$ is equal to h or not. If it is right, then restore s/s_i by isolating left-hand $k/2$ and right-hand $k/2$ bits from z. The receiver rejects the transaction *open* if $H(z) \neq h$. H is a collision resistance one way hash function in order to prevent malicious committers or adversaries from opening their promises in different ways.

4.1 Ideal Function F_{Ledger}

The function F_{Ledger} is a public ledger. It can be accessed by participants and even the others entities. Participants generate valid transactions. Miners gather these transactions in a regular sequence which is treated as the state of the ledger. In bitcoin system, a new block of transactions will be embedded in the ledger around every 10 minutes, and the state of the ledger will update accordingly. Transactions are not posted on the ledger directly. Miners first add a transaction to a $buffer$ if the transaction is valid. After a certain time, all transactions in the $buffer$ will be posted on the ledger in sequence. The bitcoins of transactions *commit* rom the merchant and customers are transferred to a default account. Participants have an agreement that conditionally transfers some bitcoins to other party who can provide some special data in a transaction. We employ F_{Ledger} as a smart contract. Smart contract can keep data in a local memory and change its local storage whenever a transaction is received. This bitcoin will not be transferred until a certain time. In the end, the bitcoins in the account may be back to the party who initiated the transaction or send to other participant. A detailed description of the process is as follows.

All participants have access to function F_{Ledger}. Set the parameter values for the function. There are constant $T, buffer$ and $counter$. The default setting is $buffer := \xi, counter = 0$ at the start of communication. The $counter$ adds1 every T minutes.

Step 1: Upon receiving $commit^i$ from the merchant and/or $commit_i^M$ from the consumer,where $i \in \{1, .., n\}$. If $Validate(commit) = 1, commit \in \{commit^i, commit_i^M\}$, then set $buffer := buffer||commit$. At $counter = 1$, received *commit* are listed as a table that is defined as $List1$, such as $(commit^1, P_1), \cdots, (commit^n, P_n), (commit_i^M, P_i), 1 \in \{1, .., n\}$.

Step 2: Upon receiving $deposit^i$ from the merchant and/or $deposit_i^M$, $i \in \{1, \cdots, n\}$ from the consumer. If $Validate(deposit) = 1, deposit \in \{deposit^i, deposit_i^M\}$, then set $buffer :=$

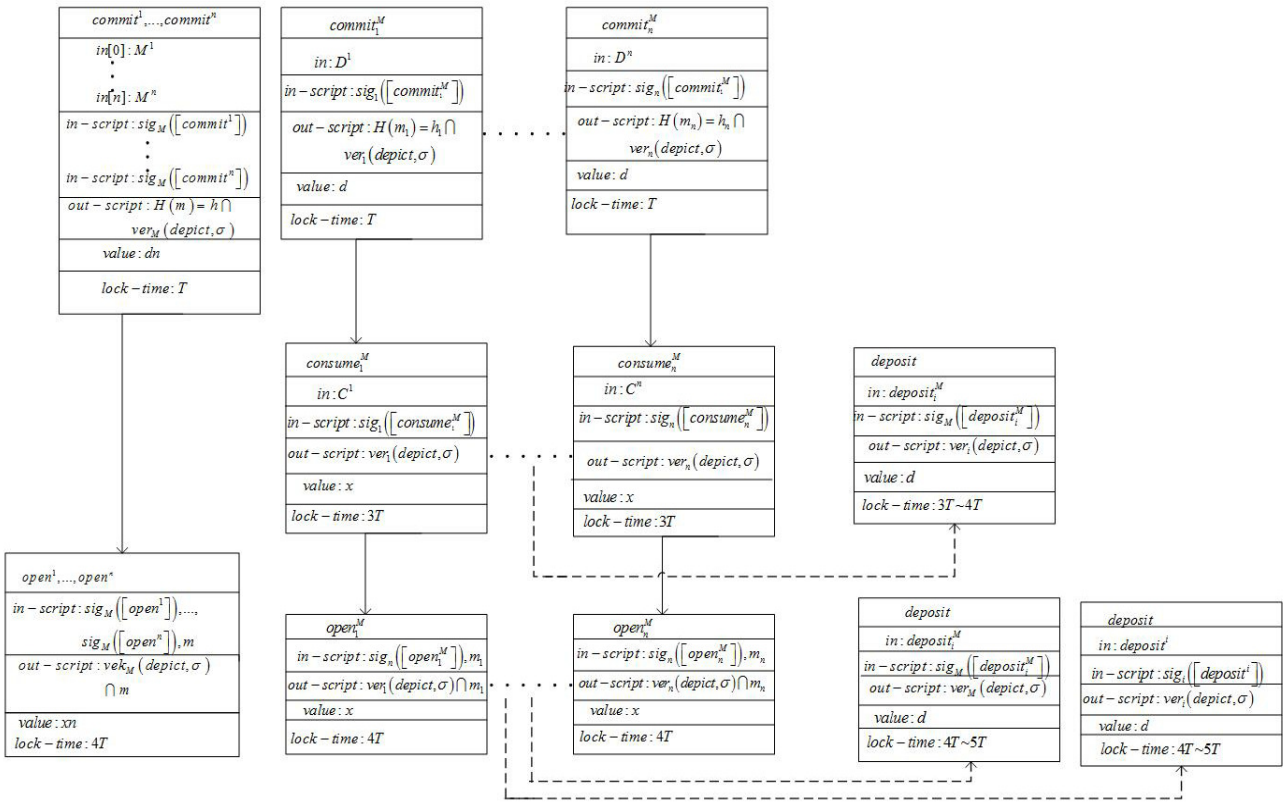

Figure 1: The process of protocol's transactions

$buffer||deposit$. At $counter = 2$, received $deposit$ are listed as a table that is defined as $List2$, such as $(deposit^1, P_1), \cdots, (deposit^n, P_n), (deposit_i^M, P_i), i \in \{1, .., n\}$.

Step 3: Upon receiving $consume_i^M$ from consumer,where $i \in \{1, .., n\}$. If $Validate(consume_i^M) = 1, i \in 1, .., n$, then set $buffer := buffer||consume_i^M$. At $counter = 3$, received $consume_i^M$ are listed as a table that is defined as $List3$, such as $(consume_1^M, P_1), \cdots, (consume_n^M, P_n), i \in \{1, .., n\}$.

Step 4: During counter 3 to 4, a signed transaction $sig_M(deposit_i^M), i \in \{1, \cdots, n\}$ from the merchant is received. If $vek_M[sig_M(Validate(deposit_i^M))] = 1$ and $P_i \in List1 \cap P_i \in List2 \cap P_i \notin List3$, then sends x BTC(s) to the merchant and remove P_i from the $List1$ and $List2$.

Step 5: Upon receiving $open_i^M$ from the consumer and/or $open^i$ from the merchant, where $i \in \{1, \cdots, n\}$. If $Validate(open) = 1, open \in \{open_i^M, open^i\}$, then set $buffer := buffer||open$. At $counter = 4$, received $open$ are listed as a table that is defined as $List4$, such as $(open^1, P_1), \cdots, (open^n, P_n), (open_i^M, P_i), i \in \{1, \cdots, n\}$.

Step 6: During counter 4 to 5, a signed transaction $sig_M(deposit_i^M), i \in \{1, \cdots, n\}$ from the merchant is received. If $vek_M[sig_M(Validate(deposit_i^M))] = 1$ and $P_i \in List1 \cap P_i \in List2 \cap P_i \in List3 \cap P_i \notin List4$, then sends x BTC(s) to the merchant and remove P_i from the $List1, List2$, and $List3$. Otherwise the information is ignored. A consumer also performs a similar process if he/she does not get an effective $open^i$.

Step 7: At $counter = 6$, BTCs which are belong to the rest of $P_i \in List3$ are transferred to the merchant and the coins of $commit$ are returned to the party that initiated the transaction.

The contract storage can be used for function F_{Ledger} to preserve account balances for each address. The balance of accounts has one feature that a certain amount of BTCs can be shelved. Once the function F_{Ledger} begins, block timestamp and counter will be checked. The participant requests the function F_{Ledger} if he does not receive or receive the wrong transaction information. The function F_{Ledger} solves the problem automatically.

5 Models

In bitcoin system, all transactions can be traced and every BTC can be traced back to the first block in which BTC is created from the transaction. However, this does not mean anonymity is lost. Public and private key pairs

are generated randomly in every new transaction, and it is difficult to known before they are produced. And an address is a hash of public key. Therefore, it is impossible to recognize the true identity of the participant only from the public transaction. To strengthen anonymity, the address is updated for each new transaction in the protocol. Meanwhile, every transaction contains the signature of a merchant or/and a consumer. The probability of forging a signature is negligible [16, 24]. It is difficulty to tamper with and/or forge a transaction.

One of the main problems of electronic currency is double spending because electronic sequence number can be copied easily. There is no center node to monitor transactions to prevent the double spending. For overcoming this disadvantage, all transactions are broadcast. Then they can be verified by the nodes in the network. Through the P2P network, all nodes can keep a transaction chain and record the flow of transfer funds of transactions. The essence of bitcoin transaction records is currency transfer records. We can learn the source and destination of BTCs from the records. We set the participants can receive the BTCs after verification of six nodes (that is to say, about six blocks are formed.). Of course, the more blocks you produce, the more secure the transactions can be. However, we cannot prevent duplicate payments of dishonest consumers. Some participants may wait until the transactions are fully accepted by the network nodes before completion of the payment, but some careless participants may be deceived. Once the payment is successful, there are not relevant mechanisms which are used to recover the illegal transfer in the BTC system. Therefore, there is a possibility that a consumer will also pay the same currency to different parties to form a double payment. Now let us calculate the probability.

Definition 1. *An adversary can catch up with honest miners with probability* $1 - \sum_{k=0}^{c} \lambda^k e^{-\lambda}/k! \times (1 - (q_a/q_h)^{c-k})$ *when there are c blocks behind the block in which contains the real transaction.*

Proof. Supposing that q_h indicates the probability of finding the next block of honest nodes, q_a indicates the probability of finding the next block of adversaries and q_c indicates the probability that an adversary can catch up with honest nodes after c blocks. Obviously,

$$q_c = \begin{cases} 1 & if \quad q_h \leqslant q_a \\ (q_a/q_h)^c & if \quad q_h > q_a \end{cases} \tag{1}$$

q_c decreases exponentially with the increasing number of new blocks if q_h is greater than q_a. The probability of success is smaller with the increasing of the block if an adversary does not succeed at the beginning. An honest participant is waiting the transaction which is added to a block and might even be c blocks behind it. However, he does not know how many blocks an adversary had generated. It is assumed that the speed at which honest nodes generate block is the same as that of the blockchain. Then the progress of an adversary is consistent with Poisson distribution that is $\lambda = c(q_a/q_h)$. There are c blocks behind the block which contains the real transaction, but the adversary is still able to catch up with honest nodes.In this case, we calculate the probability as follows.

$$\sum_{k=0}^{\infty} \lambda^k e^{-\lambda}/k! \times \begin{cases} (q_a/q_h)^{c-k} & if \quad k \leq c \\ 1 & if \quad k > c \end{cases} \tag{2}$$

Simplify the above Equation (2) leads to $1 - \sum_{k=0}^{c} \lambda^k e^{-\lambda}/k! \times (1 - (q_a/q_h)^{c-k})$. The probability of double spending can be ignored if the adversary fails to cheat at the beginning. Our protocol will confirm success of the transaction in the sixth block. Therefore, the probability of double spending can be ignored in the proposed protocol. □

Then we discuss another property. An adversary creates a large number of false identities under his control in the sybil attack. In order to attack our agreement, the adversary may create l consumers who perform the protocol. Firstly, the commitment scheme should be implemented and the deposit is necessary if he wants to get the desired information. This is contrary to the original intention for sybil attack. The adversary wants to get information from the merchant or BTCs from the honest participant but does not want to pay any BTC. Therefore, the sybil attack does not work. On the other hand, the aim of the sybil attack is to break the fairness. In the BTC system, miner nodes employ themselves in proof of work computations to produce the block. The adversary wants to break the system. He needs to control at least a half of the computing power of total computing power which is the linked computing power of all the other honest participants of the protocol. To summarize, it is not helpful to the adversary by producing many false identities.

Finally, fairness will be proved.We construct a encapsulate function $E(G)$ and set three models $G^{init}, G^{div}, G^{abt}$. Specifically, G^{init} guarantees that parties can participate in the protocol only if he has enough BTCs. G^{div} guarantees that the honest participants do not lose BTCs when performing the protocol. G^{abt} guarantees that the honest participants will obtain BTCs as compensation if they do not receive the information or receive the wrong information. More specifically, the encapsulate function $E(G)$ ensures that G^{init} is true by checking global setup after receiving information from participants. If validation passes, the next step is executed.Otherwise, protocol is terminated. At the same time, $E(G)$ is useless if there are not enough BTCs to execute the protocol. G^{div} s satisfied when the dishonest participants have a negative financial balance. G^{abt} is met when honest participants have a positive financial balance. Anyhow, any input will be ignored if the requirements are not met in the model. The input of function UC should be given with a high priority. UC is a local function (for more info please refer [7,19]). A participant can obtain resource setup which contains updated public and private keys. Resource setup is associated with global ledger by generating algorithm

Gen $\{(0,1)^*\}^2 \leftarrow 1^*$. Then public key is broadcast and the simulator receives public and private keys. The specific description of the encapsulate function is given.

The function $E(G)$ interacts with the merchant M, consumers $P_i, i \in \{1,..,n\}$, the adversary S, the local function UC and the environment Z.There are three models $G^{init}, G^{div}, G^{abt}$, and there is a generate algorithm $\{(0,1)^*\}^2 \leftarrow 1^*$ for generating resource setup. It is a one-way process that transactions are posted on the ledger. The ledger has two output transfer models, that is, fair transfer model and delay transfer model. The function $E(G)$ also has an indicator bit c which is set to 0 at the beginning. c is used to indicate whether information sent by the adversary S to UC is blocked.

- Setup. The algorithm $G^{init}, G^{div}, G^{abt}$ generates resource setup which is needed in every new transaction.
- Once receiving information N from UC to its simulator, if $c = 0$ sends N to S.
- Once receiving information N from S if $c = 0$ sends N to UC as information from the simulator.
- Once receiving a transaction *commit* from M/P_i, posts it to global ledger. If G^{init} is not satisfied(*e.g.* BTCs are not enough, transaction has redeemed, address is inconsistent, *etc.*) then set $c = 1$.
- Delay output. Once receiving information from UC marked $(delay, sid, N, P_i)$ send N to M/P_i by delay output.
- Fair output. Once receiving information from UC marked $(fair, sid, obj, (m, P_1), \cdots, (m, P_i), (m_1, P_1), \cdots, (m_i, P_i)), (m_s, S))$, it sends $(sid, obj, P_1, \cdots, P_i, m_s)$ to S.
- Fair delivery. Messages $(delivery, sid, obj)$ are received from S then the information $(obj, ...)$ will be sent to S by doing the following.Each pair $(m, P/M)$ is associated with the obj, and sets $H_d = \{(m, P/M)|P/M \ is \ honest\}$. It forwards $\{(m, P/M)|P/M \ is \ corrupted\}$ to S. If the P/M in the H_d is corrupted on the way, then sends the corrupted $(m, P/M)$ to S. Next, perform the following operations.

Remark 1. *Once input information $(m, P/M)$ from S and the obj has the pair $(m, P/M) \in H_d$. Then the information is posted on the ledger.The information is ignored if G^{div} is wrong. Otherwise, remove $(m, P/M)$ from H_d.*

Remark 2. *Once abort information $(m, P/M)$ from S and the obj has the pair $(m, P/M) \in H_d$. Then the information is posted on the ledger.The information is ignored if G^{abt} is wrong. Otherwise, removes $(m, P/M)$ from H_d.*

Definition 2. *Let π be a probabilistic non-uniform polynomial time (PPT) protocol.We say that π achieves fairness with global ledger defined as G if the status of following statement is correct.Let Π be a non-uniform PPT protocol in $(G, E(G))$ hybrid model. For every non-uniform PPT real word adversary A attacking π there exists a non-uniform PPT ideal word simulator S so that for every non-uniform PPT environments Z it holds.*

$$IDES^{G,E(G)}_{\Pi,S,Z} \approx REAL^{G}_{\pi,A,Z} \tag{3}$$

Proof. By the conditions described above, π achieves fairness with G. We hold $\forall A', \exists S'$ that leads to $\forall Z$.

$$IDES^{G,E(G)}_{S',Z'} \approx REAL^{G}_{\pi,A',Z'} \tag{4}$$

Next, let us prove Equation (4). The proof process is similar to [19]. First, let us start with an introduction to $REAL^{G}_{\pi,A,Z}$. Suppose that L is a polynomial upper bound value of many specific examples of π and let $\pi(l)$ represents $l-th$ reproduce of protocol π. Suppose adversaries $A = (A^{\Pi}, A^{\pi(1)}, A^{\pi(2)}, \cdots, A^{\pi(L)})$, let $A^{\pi(l)})$ represent an interact with the $l-th$ reproduce of protocol π. And the environment supplies input to the protocol Π and obtains the corresponding output result from protocol Π. Meanwhile, the input and output of subroutines $\pi(l)$ are provided by protocol Π. Then let us show that $(G, E(G))$ hybrid model performs $IDES^{G,E(G)}_{\Pi,B,Z}$. Suppose $E(G)[l]$ represents $l-th$ reproduce of the function $E(G)$. Also, we define the $B = (A^{\Pi}, S^{\Pi(1)}, S^{\Pi(2)}, \cdots, S^{\Pi(L)})$, where every $S^{\Pi(l)}$ is an interact with the $l-th$ examples of function $E(G)$. Similarly, the environment supplies input to the protocol Π, and the input of subroutines of Π is supplied by Π. It is no doubt that all examples of protocol are allowed to contact global ledger. □

Through the above, we say that ideal world and real world are indistinguishable in the mixed argument. The mixed is defined as Mix^l. In order to describe the convenience, Mix^l are given as following.

Suppose Π^l expresses an example of the protocol Π.

- $l-1$ examples of the protocol Π,defined $\pi(1), \cdots, \pi(l-1)$.
- $L-l+1$ examples of the function $E(G)$, defined $E(G)[l], \cdots, E(G)[L]$.

Suppose A^l expresses the reproduction of the following adversary.

- A^{Π};
- $l-1$ examples of the protocolA^{π},defined $A^{\pi(1)}, \cdots, A^{\pi(l-1)}$.
- $L-l+1$ examples of the simulator S^{π}, defined $S^{\pi(l)}, S^{\pi(L)}$.

Suppose $\bar{B}$ contains l reproductions of adversaries and l reproductions of protocol(/function) examples. Define B' contains $l-th$ reproduction of adversary A and $l-th$

reproduction of protocol (/function) example. $\bar{B}$ equals Mix^l if $A = S^{\pi(l)}$ and $\pi = E(G)[l]$. $\bar{B}$ equals Mix^{l+1} if $A = A^{\pi(l)}$ and $\pi = \pi(l)$. Next, we assume that Mix^l and Mix^{l+1} are indistinguishable.

Lemma 1. *The output between Mix^l and Mix^{l+1} (adjacent mixtures)is indistinguishable for non-uniform PPT environments Z, where $l \in \{1, .., L\}$.*

Proof. We assume that a non-uniform PPT environment Z can show the difference between Mix^l and Mix^{l+1}. In other words, $IDEAL^G_{\Pi^l, A^l, Z} \not\approx IDEAL^G_{\Pi^{l+1}, A^{l+1}, Z}$. In this case, it is assumed that Z can simulate all interactive behaviors except for the $l-th$ subroutine.

We can learn that $IDEAL^G_{\Pi^{l+1}, A^{l+1}, Z}$ can be represented by $IDEAL^G_{S^{\pi(l)}, Z^l}$. By the same reason, $IDEAL^G_{\Pi^{l+1}, A^{l+1}, Z}$ can be represented by $REAL^G_{\pi, A^{\pi(l+1)}, Z^l}$. On the basis of discussion above, we see that$IDEAL^G_{\Pi^l, A^l, Z} \not\approx IDEAL^G_{\Pi^{l+1}, A^{l+1}, Z}$. Then $IDEAL^{G, E(G)}_{S^{\pi(l)}, Z^l} \not\approx REAL^G_{\pi, A^{\pi(l)}, Z^l}$. However, based on the Eq.(4) $IDEAL^{G, E(G)}_{S', Z'} \approx REAl^G_{\pi, A', Z'}$. They are contradicted. It shows that our hypothesis $IDEAL^G_{\Pi^l, A^l, Z} \not\approx IDEAL^G_{\Pi^{l+1}, A^{l+1}, Z}$is wrong.

To summarize, the Mix^l and Mix^{l+1} are indistinguishable for non-uniform PPT environments Z, where $l \in \{1, ...L\}$. The lemma is proved. □

Obviously, theMix^lequals $IDEAL^{G, E(G)}_{\Pi, A^l, Z}$, and Mix^{l+1} equals to $REAL^G_{\pi, A, Z}$. Through $Mix^l \approx Mix^{l+1}$, where $l \in \{1, \cdots, L\}$.We can obtain $Mix^1 \approx Mix^2 \approx \cdots \approx Mix^l \approx Mix^{l+1}$, that is, $IDES^{G, E(G)}_{\Pi, A^l, Z} \approx REAL^G_{\pi, A, Z}$. The Definition 2 is proved. That is, our protocol provides fairness.

6 Other Topology

Following the same way, we extend the proposed protocol to any topology. Participants send and receive messages sequentially in ring topology and sequential topology. However, participants have no order in star topology and mesh topology. First, we discuss the circumstances in which participants need to execute protocol in sequence. All transactions are publicly visible on the ledger and participants also have some offline communications, such as negotiating the deposit and maximum delay time *et al.*Now the order of participants is also agreed offline. That is, the transaction of the latter participant appears on the ledger unless the transaction of the former participant appears on the ledger. Order of participants, deposit and the delay time of the transaction *et al.* may vary in different protocols. Meanwhile, the number of copies of the deposit is determined by the participants involved (A deposit is required between participants who exchange information directly) in the transactions. Obviously, these issues are easy to solve. If the above problems have been resolved, our proposed protocol can be applied to ring topology and sequential topology. There is no doubt that our protocol is easier to apply to mesh topology. The process of applying our protocol to hybrid topologies is similar.

7 Protocol Comparison

Every multi-party exchange protocol is dependent on different technologies. We define the MFE to be the traditional multi-party fair exchange and MPCS to be the traditional multi-party contract signing. Recently, researchers have proposed some multi-party fair exchange protocols based on bitcoin. The literatures [10,20] are traditional multi-party protocols, and the literatures [1, 5] are multi-party protocols based on bitcoin. In Table 3, the efficiency and some features are compared between the proposed protocol and some related protocols. For a fair comparison, the data should be calculated under the same security conditions. Therefore, these data are collected under mesh topology. However, each protocol solves the dispute in different ways. It is difficult to measure with the same standard. Accordingly, these data in Table 3 are collected in optimistic situation, that is, all participants are honest and no network problems.

n is the number of participants. In [5], $n = 2^L$, $x \in \{1, ...L-1\}$ and the mix topology means mesh and sequential topologies. Message shows the number of signatures on information which is produced by each party$P_i, i \in \{1, ...n\}$. In the form of A/B, A represents the transmission and message of the general participant and B stands for transmission and message of winner.

Draper-Gil *et al.* [10] presents a MPCS protocol for different topologies. In their paper, n rounds are required to obtain the signature because the participants generate a partial signature per round. H. Kılınç, *et al.* [20] propose a MFE protocol that requires constant round. The transmission of mesh topology is $o(n^2)$ which is less than $o(n^3)$ of [10]. Compared efficiency with protocols [10,20], the number of transmissions and messages is less than our proposed protocol. Meanwhile, our proposed protocol only requires constant round. What's more, the proposed protocol does not have a center (TTP).

Andrychowicz *et al.* [1] and M. Bartoletti, *et al.* [5] propose multi-party exchange protocols based on bitcoin. The protocol in [5] is a mix topology of mesh and sequential, and the protocol in [1] is a mesh topology. However, the protocol in [5] obtains the winner by $n-1$ two-party matches so that rounds match are needed. By comparing efficiency with protocols in [1,5], the message in our protocol and the protocol in [1] is basically the same but it is less than the protocol in [5]. The transmission of our protocol is less than the protocol in [1] but it is more than the protocol in [5]. Participants decrease exponentially as the round number increases in the protocol in [5] but the number of participants is constant in each round in our protocol. Nevertheless, the proposed scheme can be applied to any topology and the protocols in [1, 5] do not

Table 3: Comparison of the proposed protocol with previous protocol

Protocol	Technique	Topology	TTP	Number of rounds	Transmission(mesh)	Message(mesh)
[10]	MFE	Any	Yes	n	$n^2(n-1)$	$(n-1)^2+1$
[20]	Bitcoin	Mesh	No	Constant	$4n^2+3n+3$ *or* $4n^2+2n+2$	$2n/2n+1$
[1]	MPCS	Any	Yes	Constant	$5n(n-1)$	$5n(n-1)$
[5]	Bitcoin	Mix	No	L	$2n+7\times 2^{L-1}-8$	$2n+1+6x$ *or* 2n+6L-1
our	Bitcoin	Any	No	Constant	n^2+4n-2	$2n$

apply to.

In summary, our protocol provides more properties, that is, applicable to any topology,constant round and no center are simultaneously satisfied. So far there has been no protocol to meet these properties at the same time. Efficiency has also been improved except for the transmission of the protocol in [5].

8 Conclusions

Fairness is one of the most fundamental properties that need to be addressed by all exchange protocols. However, we use the deposit instead of a third party to ensure fairness. It avoids some problems which are caused by the third party. The proposed protocol is a hybrid model without center. It can not only achieve fairness and security but also can automatically execute the protocol within a limited time. As far as we know, no other fair exchange protocol offers the specific process of the smart contract in the hybrid model. Moreover, we have shown a fair exchange protocol which is suitable for various topologies with small modifications. So far, there are few researches on the topological structure based on bitcoin. These problems have been studied in our paper at the same time. However, the merchant gets the BTCs of the consumers in about an hour ($6T$). And the deposit is returned in about an hour. The cost is a slightly longer time to transactions. This is a problem for the proposed protocol. How to shorten transactions time is our future work.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China (No.2016YFB0800601), the Natural Science Foundation of China (No.61303217, 61502372), the Natural Science Foundation of Shaanxi province (No.2013JQ8002, 2014JQ8313).

References

[1] M. Andrychowicz, S. Dziembowski, D. Malinowski, and L. Mazurek, "Secure multiparty computations on bitcoin," in *IEEE Symposium on Security and Privacy*, pp. 443-458, May. 2014.

[2] M. Andrychowicz, S. Dziembowski, D. Malinowski, and Ł, Mazurek, "Modeling bitcoin contracts by timed automata," in *Formal Modeling and Analysis of Timed System*, pp. 7-22, Sept. 2014.

[3] A. Back, and I. Bentov, *Note on Fair Coin Toss via Bitcoin*, 2013. (`https://arxiv.org/abs/1402.3698`)

[4] C. Badertscher, U. Maurer, D. Tschudi, and V. Zikas, *Bitcoin as a Transaction Ledger: A Composable Treatment*, Aug. 2017. (`https://eprint.iacr.org/2017/149.pdf`)

[5] M. Bartoletti, and R. Zunino,"Constant-deposit multiparty lotteries on bitcoin," in *Bitcoin and Block Chain Bibliography*, pp. 231-247, 2017.

[6] K. Beekman, "A denial of service attack against fair computations using bitcoin deposits," *Information Processing Letters*, vol. 116, no. 2, pp. 144-146, 2016.

[7] I. Bentov, and R. Kumaresan, "How to use bitcoin to design fair protocols," *Lecture Notes in Computer Science*, vol. 8617, pp. 421-439, Aug. 2014.

[8] J. Bonneau, A. Miller, J. Clark, and A. Narayanan, "SoK: Research perspectives and challenges for bitcoin and cryptocurrencies," in *IEEE Symposium on Security and Privacy*, pp. 104-121, May 2015.

[9] T. Y. Chang, M. S. Hwang, and W. P. Yang,"An improved multi-stage secret sharing scheme based on the factorization problem," *Information Technology and Control*, vol. 40, no. 3, 2011.

[10] G. Draper-Gil, J. L. Ferrer-Gomila, M. Francisca Hinarejos, and J. Y. Zhou, "On the efficiency of multiparty contract signing protocols," in *18th International Conference (ISC'15)*, pp. 227-243, Sep. 2015.

[11] Z. Eslami, M. Noroozi, and S. K. Rad, "Provably secure group key exchange protocol in the presence of dishonest insiders," *International Journal of Network Security*, vol. 18, no.1, pp. 33-42, 2016.

[12] J. Garay, A. Kiayias, and N. Leonardos, "The bitcoin backbone protocol: Analysis and applications," *Lecture Notes in Computer Science*, vol. 9057, pp. 281-310, Apr. 2015.

[13] S. Goldfeder, J. Bonneau, R. Gennaro, and N. Arvind, "Escrow protocols for cryptocurrencies: How to buy physical goods using bitcoin," in *Bitcoin and Block Chain Bibliography*, pp. 321-339, 2017.

[14] O. Goldreich, S. Micali, A. Wigderson, "How to play any mental game or a completeness theorem for protocols with honest majority," in *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pp. 218-229, Jan. 1987.

[15] E. Heilman, F. Baldimtsi, and S. Goldberg, "Blindly signed contracts: anonymous on-blockchain and off-blockchain bitcoin transactions," in *International Financial Cryptography Association*, pp.43-60, Feb. 2016.

[16] R. J. Hwang, and C. H. Lai, "Provable fair document exchange protocol with transaction privacy for e-commerce," *Symmetry*, vol. 7, no. 2, pp. 191-196, 2015.

[17] M. H. Ibrahim, "SecureCoin: A robust secure and efficient protocol for anonymous bitcoin ecosystem," *International Journal of Network Security*, vol. 19, no. 2, pp. 295-312, 2017.

[18] A. Juels, A. Kosba, and E. Shi, "The ring of gyges: Using smart contracts for crime," in *Memento Computer and Communication Science*, pp. 40-54, 2015.

[19] A. Kiayias, H. S. Zhou, and V. Zikas, "Fair and robust multi-party computation using a global transaction ledger," *Lecture Notes in Computer Science*, vol. 9666, pp. 705-734, May 2016.

[20] H. Kılınç, and A. Küpçü, "Optimally efficient multi-party fair exchange and fair secure multi-party computation," *Lecture Notes in Computer Science*, vol. 9048, pp. 330-349, Apr. 2015.

[21] S. Lal and M. L. Das, "On the security analysis of protocols using action language," *International Journal of Electronics and Information Engineering*, vol. 2, no. 1, pp. 1-9, 2015.

[22] H. Pagnia, H. Vogt, and F. C. Gärtner,"Fair exchange," *Computer Journal*, vol. 46, no. 1, pp. 55-75, 2003.

[23] M. Patrick, M. Malter, F. Siamak, Shahandasti, and F. Hao, "Towards bitcoin payment networks," *Lecture Notes in Computer Science*, vol. 9722, pp. 57-76, July 2016.

[24] R. Rivest, and B. Kaliski,"RSA problem," in *Encyclopedia of Cryptography and Security*, pp. 532-536, 2005.

[25] T. Ruffing, and A. Kate, "Liar, liar, coins on fire!: Penalizing equivocation by loss of bitcoins," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 219-230, Oct. 2015.

[26] Y. Sun, H. F. Zhu, and X. S. Feng, "A novel and concise multi-receiver protocol based on chaotic maps with Privacy Protection," *International Journal of Network Security*, vol. 19, no. 3, pp. 371-382, 2017.

[27] A. C. Yao,"Protocols for secure computations," in *Foundations of Computer Science*, pp. 160-164, Nov. 1982.

[28] Y. F. Yao and F. H. Yu,"Privacy-preserving similarity sorting in multi-party model," *International Journal of Network Security*, vol. 19, no. 5, pp. 851-857, 2017.

[29] S. W. Zheng, *Credit Model Based on P2P Electronic Cash System Bitcoin*, School of Information Security Engineering Shanghai Jiao Tong University, 2011.

Biography

Lijuan Guo received the BS degree in Mathematics and Applied Mathematics from Changzhi University in 2015. She is currently working toward the M.S. degree in School of Mathematics and Statistics from Xidian University. Her research interests include information security, fair exchange protocol, blockchain.

Xuelian Li received her PhD degree of Cryptography in December 2010 at Xidian University. Currently, she is currently an associate professor at School of Mathematics and Statistics, Xidian University. Her research interests include information security, fair exchange protocol, cryptographic functions and stream ciphers.

Juntao Gao received his PhD degree of Cryptography in December 2006 at Xidian University. He is currently an associate professor at School of Telecommunications, Xidian University. He is also a member of Chinese Association for Cryptologic Research. His research interests include information security, stream cipher and cryptographic functions, pseudorandom sequences and blockchain.