2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering

FPPB: A Fast and Privacy-preserving Method Based on the Permissioned Blockchain for Fair Transactions in Sharing Economy

Bin Li, Yijie Wang, Peichang Shi, Huan Chen, Li Cheng
Science and Technology on Parallel and Distributed Processing Laboratory,
College of Computer, National University of Defense Technology,
Changsha, Hunan, P.R.China, 410073
{libin16a, wangyijie, pcshi}@nudt.edu.cn, chenhuan245@gmail.com, chengli09@nudt.edu.cn

***Abstract*—Blockchain is a distributed system with efficient transaction recording and has been widely adopted in sharing economy. Although many existing privacy-preserving methods on the blockchain have been proposed, finding a trade-off between keeping speed and preserving privacy of transactions remain challenging. To address this limitation, we propose a novel Fast and Privacy-preserving method based on the Permissioned Blockchain (FPPB) for fair transactions in sharing economy. Without breaking the verifying protocol and bringing additional off-blockchain interactive communication, FPPB protects the privacy and fairness of transactions. Additionally, experiments are implemented in EthereumJ (a Java implementation of the Ethereum protocol) to measure the performance of FPPB. Compared with normal transactions without cryptographic primitives, FPPB only slows down transactions slightly.**

***Index Terms*—blockchain, privacy-preserving, sharing economy, transaction fairness, cloud service, zero-knowledge proof, stealth address**

I. Introduction

Blockchain is a continuously increasing list of records, which are linked and secured using cryptography. It offers a novel way for bookkeeping in a fully distributed manner, which makes blockchain potential to innovate other areas where there is a lack of trust between involved parties[1]. One such area is the sharing economy[2]. Blockchain has been successfully applied in the sharing economy to address centralized problem, such as single point failure and malicious behaviors. For example, Joint Cloud Computing (JCC) is an application of sharing economy. It borrows the ideas from airline alliances and aims at empowering the cooperation among various cloud vendors to provide cross-cloud services via software definition[3]. In JCC, some cloud service providers (CSP) form a consortium together to provide cloud service for normal customers or each other. A customer selects a cloud service he wants in this consortium and sends his request for a cloud service to the CSP. Then, the CSP confirms this request and provides the cloud service that the customer wants off blockchain. All transactions are verified and recorded by verifiers on the blockchain.

Despite the advantages and power of the blockchain in distributed environment, the present form of these technologies lacks transaction privacy. Due to the public verification of blockchain, even though parties can create new pseudonymous public keys to increase their anonymity, the values of all transactions and balances for each (pseudonymous) public key are publicly visible. The public trading contents and relationships can disclose earnings and expenditures as well as marketing plans of a service provider. Each provider regards these information as core secret and do not want these secrets to be disclosed by anyone else.

Existing privacy-preserving methods for transactions on the blockchain cannot behave well in protecting privacy and ensuring speed without breaking the verification protocol. First, a number of technologies cannot protect privacy as predicted. The laundry service confuses trading relationships between inputs and outputs without slowing transactions down significantly[4]. However, it does not protect transaction content in sharing economy from privacy attackers. Because, if the service is just owned by a few providers, these privacy attackers can narrow the scope of trading partners to disclose the actual identity of provider. Second, other technologies for preserving privacy on the blockchain obviously reduce transaction speed and increase the storage overhead, such as Zerocash[5] and Confidential Transactions[6]. Last, the traditional asymmetric encryption achieves high efficiency and privacy protection. However, it will break the verification protocol of blockchain and verifiers cannot check the validity of transactions.

In this work, we propose a fast and privacy-preserving method based on the permissioned blockchain (FPPB) for fair transactions in sharing economy to address these problems. The main novelty of FPPB lies in adopting zero-knowledge proof and stealth address on the blockchain to ensure the uniformity of transaction contents without disclosing privacy. Meanwhile, it does not bring additional off-blockchain interaction between trading partners. We achieve following aims: **(1)Privacy:** Due to these cryptographic primitives, FPPB achieves protecting privacy of transaction content and trading relationships. **(2)Fairness:** Verifiers on the blockchain can confirm what the provider promises to provide is the same with the customer wants without knowing the trading relationship of true identity and transaction contents. Moreover, these requests and promises are recorded on the blockchain as proofs to solve potential disputes between providers and customers. **(3)Efficiency:** We design and implement FPPB on EthereumJ[7], which is an another version of Ethereum and implemented in Java[8]. Experiments show, compared with normal transactions without cryptographic primitives, FPPB only slows down transaction slightly.

II. Related Work

Existing researches of preserving privacy of transactions on the blockchain can be categorized into data distortion and data encryption.

A. Data Distortion

In data distortion, it is difficult for attackers to obtain the accurate information since part of a transaction is distorted. One major challenge of data distortion is to prevent disclosing trading relations without affecting transaction results. The most common way of distortion is mixing technology[4]. It provides anonymity by transferring payments from an input set of addresses to an output set of addresses so that it is hard to link the payer's address and the payee's address. The mixing technology includes centralized and decentralized mixing. A variety of websites provide centralized mixing service on Bitcoin by using the third party, such as Bitlaunder[9], Bitcoin Fog[10]. However, centralized problems in these methods may violate users' privacy and steal users' coins. To address these problems, many researchers propose the decentralized mixing service. For example, CoinJoin combines transactions without disclosing the relations between inputs and outputs[11], which is widely used in JoinMarket[12]. These methods also have limitations. First, the process of finding mixing partners will bring centralized problem, that is, a node will be a proxy to help users to find other mixing partners. Second, the mixing process will disclose transaction information of each other, such as public keys and addresses. Third, when a user deliberately aborts transaction, it will lead to DDoS attack and the failure of all transactions.

B. Data Encryption

In data encryption, many cryptographic tools are used to protect the privacy of transaction contents. It is a challenge to ensure the verification on the blockchain without being affected. Homomorphic encryption is a form of encryption that allows computation on cipher texts to generate an encrypted result[13]. When the result is decrypted, it will be the same with operations directly performed on the plain text. However, this technology is not mature and will bring much computation cost in resources and time. Confidential Transaction is proposed to protect the privacy without disclosing the value and convince these verifiers validity of transactions[14]. However, a complete confidential transaction takes up much more space storage than the transaction without preserving privacy. Zk-SNARK in Zerocash[5] and extended Zerocash[15] exhibits a high level privacy protection on users' identity, transaction contents, and transaction relationships. However, its complex cryptography method heavily slows down transactions. According to [5], at the 128-bit security level, Zerocash averagely needs 1-2 minutes to generate a proof before every transaction, which is much slower than the traditional on-line transaction. Even if the process of encryption before transaction does not affect the performance on the blockchain, this period of time should still be included in processing time of a transaction and affects transaction speed.

III. Preliminaries

A. Stealth Address

CryptoNote[16] proposes the stealth address method to make transactions unlinkable by generating one-time address and bring convenient and non-interactive trading. The destination of each output is a public key, derived from recipient's address and sender's random data. The main advantage against Bitcoin is that every destination key is unique by default (unless the sender utilizes the same data for each of his transactions to the same recipient) and the receiver does not need to convey his anonymous address to the sender. Hence, there is no such issue as "address reuse" by design and no observer can determine if any transactions are sent to a specific address or link two addresses together. The detailed schemes and proofs of stealth address can be seen in [16].

B. Zero-knowledge Proof

In cryptography, the zero-knowledge proof or zero-knowledge protocol[17] is a method based on the discrete logarithm problem by which the prover can prove to the verifier that the given statement is true without conveying any information details to the verifier. The zero-knowledge proof based on the ECC discrete logarithm includes four interactive steps.

(1) GenParam Phase: A trusted party initializes an elliptic curve to get module q and the generator g of finite field GF_q where q is a large prime. x is randomly selected from [0, q-1] and compute $h = xg$ (the elliptic curve point g added to itself x times). Then, the trusted party conveys these public parameters q, g and h to the prover and the verifier.

(2) Commitment Phase: If the prover wants to convince the verifier that he knows the knowledge, then he uses his knowledge m (selected from [0, q-1]) and public parameters from the trusted party to compute the Pedersen commitment[18], such as $M = mg + rh$ and $d = yg + sh$. r, y, s are randomly selected from [0, q-1].

(3) Challenge Phase: After the prover sends commitment values to the verifier, he is returned a random challenge e , which is also randomly selected from [0, q-1].

(4) Response Phase: The prover computes new values $u = y + em \ modq$, $v = s + er \ modq$ with the challenge and sends them to the verifier back. The verifier will verify whether the prover knows the knowledge m by checking the equation $d + eM = ug + vh$.

If succeeds, the verifier believes the prover knows knowledge m.

IV. A Fast and Privacy-preserving Method Based on the Permissioned Blockchain for Fair Transactions in Sharing Economy

In this section, we provide a fast and privacy-preserving method based on the permissioned blockchain (FPPB) for fair transactions in sharing economy and give detailed description of it. FPPB can be deployed for various sharing economy applications. For convenience, we use the case of Joint Cloud Computing (JCC) to illustrate our proposed approach. Moreover, FPPB is based on a reasonable assumption that the number of malicious participants in sharing economy is relatively small than the number of honest participants.

A. Settings

FPPB involves following roles to make transactions:

CSPs and Customers: Cloud Service Providers (CSP) register their service details on the blockchain so that customers could search for what they want and buy the service. They are not fully trusted. They may try to maximize their own benefits with malicious behaviors. We denote them as P and C.

Verifiers: Some CSPs as the initiators of JCC will be verifiers on the permissioned blockchain to maintain operations of FPPB, they check the validity of transactions and record them on the blockchain. They can also make transactions with customers.

Supervisor: In sharing economy, the supervisor plays the role as a trusted party to ensure trading fairly, like government and bank. They are responsible for releasing parameters to the public and solving disputes between participants off blockchain.

B. Basic Idea

In FPPB, we adopt some cryptographic primitives on transactions to protect the privacy of transaction contents and relationships as Figure 1 shows. First, the supervisor releases some necessary public parameters to the network. Second, every CSP will register their services and service descriptions on the blockchain in real name. Third, the customer looks through the blockchain to find the service he wants. Then, he sends his request for this service to the blockchain. To protect his identity privacy, the customer could send this transaction in a one-time anonymous address. Last, every CSP searches for new transactions about himself. And then, he also submits his response, which promises to provide the service that the customer wants with a one-time anonymous address. During this process, verifiers ensure that the trading partners have reached an agreement on the service without knowing what the service exactly is and the true identity of partners. If the customer and provider have disagreements on their off-blockchain providing service, the party who loses benefits will resort to the supervisor with these transactions on the blockchain. The supervisor can find the malicious party according to these parameters and give him punishments.

The goal of FPPB is to protect the privacy of transaction information and ensure the uniformity of transaction contents for fairness, without breaking the verifying protocol on the blockchain and bringing additional off-blockchain interactive communication. The uniformity of transaction contents means what the CSP has promised to provide is the same with the customer requests.

C. Transaction Procedure

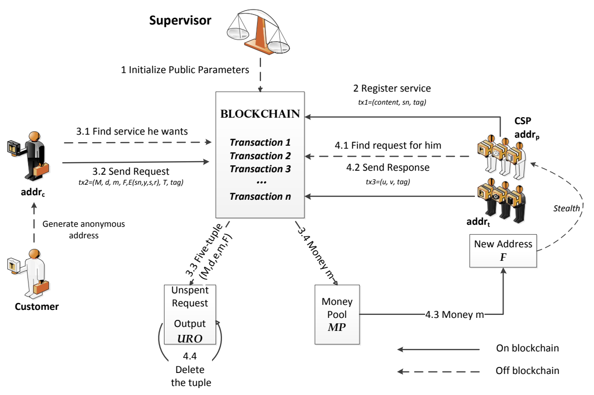

Figure 1. Overview of FPPB

FPPB has three types of transaction: *Register Transaction*, *Request Transaction*, and *Response Transaction*. Each transaction should be charged transaction fees to prevent DDoS attack[19]. We first introduce the new mechanism that we bring in FPPB. MP is a money pool that stores transactions money from customers. URO (unspent request output) is a distributed hashtable storing all unspent five-tuple (M, d, e, m, F) in *Request Transaction*. These five parameters will be described in the end of this section. MP and URO are maintained by all verifiers. Each verifier stores a copy of states of MP and URO in local.

1) Initialize Public Parameters: In this step, the supervisor initializes an elliptic curve to get parameters q, g and h. Specially, q is a large prime. g is a generator of finite field GF_q. Then, he randomly chooses x from [0, q-1] and computes $h = xg$. x is the secret value and must be hidden from other participants in JCC. No one can obtain x when g and h are given due to the discrete logarithm. If x is disclosed to others, they can use x to cheat verifiers. We will explain the reason why x is the secret in the end of this section. The supervisor informs every participant of parameters q, g, h for their later privacy protection.

2) Register Service: In this step, P registers his services on the permissioned blockchain with a permanent address $addr_p$ and broadcasts *Register Transaction* as $tx1 = (content, sn, tag)$. $addr_p$ denotes the identity of P. This transaction is recorded on the blockchain and shows everyone $content$, including service types, usage time, price and other necessary information, which is convenient for customers' selections. sn is a unique serial number selected in [0, q-1] to represent a service. tag in $tx1$ denotes the type of this transaction for all participants. This transaction is recorded on the blockchain so that customers can access it to compare different cloud services.

3) Send Request: In this step, C sends *Request Transaction* in an anonymous temporary address $addr_c$ for service he wants. To protect his identity, he should change this address in every service purchase.

First, when C decides on a cloud service, C unpacks the address $addr_p$ of P, obtains this CSP's public key (A, B), and generates a nonce t in [0, q-1] to compute a one-time public key $F = H_s(tA)G + B$ where H_s is a cryptographic hash function $\{0,1\}^* \rightarrow GF_q$ and G is a base point of the elliptic curve to generate public keys. Then, C uses F as a destination key for the output and packs value $T = tG$ into the transaction. Second, C randomly generates commitment parameters y, s, r from [0, q-1] to compute Pedersen commitment values M and d with the corresponding service sn, satisfying $M = sn \cdot g + r \cdot h$ and $d = y \cdot g + s \cdot h$. Last, C encrypts sn, y, s, r with the public key (A, B) to obtain the encrypted $E(sn, y, s, r)$. He sends m to the money pool MP by submitting $tx2 = (M, d, m, F, E(sn, y, s, r), T, tag)$. Only the CSP owning corresponding private key (a, b) could decrypt $E(sn, y, s, r)$ where $A = aG$ and $B = bG$ $(a \neq b)$.

After $tx2$ is broadcast, the verifier checks whether the $addr_c$ has sufficient money. If yes, then this verifier will approve this transaction and label e, randomly selected from [0, q-1], on $tx2$. This verifier will update his local state copy of MP and URO by transferring the corresponding money from $addr_c$ to MP and adding the five-tuple (M, d, e, m, F) into URO. When the block containing this $tx2$ has been broadcast and approved by other verifiers, it will be linked to the main chain and other verifiers will update their local state copies of MP and URO. Once $tx2$ is stored on the blockchain, no one can deny the fact that the customer C actually has requested a service and has paid the money to the money pool MP even though nobody knows what the service exactly is.

4) Send Response: In this step, P sends *Response Transaction* as $tx3 = (u, v, tag)$ with his temporary address $addr_t$. To protect the identity privacy, $addr_t$ should be a one-time address. Since it does not exist any money flow to disclose the relationship of $addr_p$ and $addr_t$, other participant in FPPB cannot link these two addresses.

First, P checks every new *Request Transaction*s recorded on the blockchain with the public key T of each transaction and own private key (a, b) to compute the new address F' as $F' = H_s(aT)G + bG$. If P is the recipient of a transaction from an anonymous address $addr_c$, F' satisfies $F' = F$ due to $aT = atG = tA$. P recovers the corresponding one-time private key $f = H_s(aT) + b$ and spends his output at any time by designing a transaction with the private key f.

Second, P uses his private key (a, b) to decrypt $E(sn, y, s, r)$ in this *Request Transaction* to compute u and v, satisfying $u = y + e \cdot sn \ mod \ q$, $v = s + e \cdot r \ mod \ q$. Last, P responds the customer's request by submitting $tx3$. When $tx3$ is broadcasted to the network, the verifier uses u and v to check the presence of a corresponding five-tuple (M, d, e, m, F) in URO satisfying $ug + vh = d + eM$. If the verification succeeds, then this verifier updates his local state copy of URO and MP by deleting the five-tuple (M, d, e, m, F) from URO and transferring the corresponding money m from MP to the address F. When the block containing this $tx3$ has been broadcast and approved by other verifiers, it will be linked to the main chain and other verifiers will update their local state copies as we mentioned above. After that, $tx3$ can be recorded on the blockchain, which shows that P has promised to provide the cloud service that his corresponding customer C wants. C can confirm that his request is responded with the unique u and v appearing on the blockchain.

We demonstrate that if a *Request Transaction* is not responded by any *Response Transaction* over a period of time, then V will update their local state copies by transferring money m from MP to C and deleting the corresponding tuple from URO. Based on the permissioned blockchain, FPPB reduces the number of participants accessing MP and URO, and improves the efficiency of verification and state updates.

In the end of this section, we explain the reason why x is a secret value owned by the supervisor. For example, if a malicious CSP obtains x, and a customer makes a Pedersen commitment M of service number $sn = 2$ and d satisfying $M(2, r) = 2g + rh$ and $d = yg + sh$. Since

$$\begin{aligned} M(2,r) &= 2g + rh = 2g + rxg \\ &= (2 + rx)g = (3 + (rx - 1))g \\ &= 3g + (rx - 1)g = 3g + (\frac{rx - 1}{x})h \\ &= M(3, r') \end{aligned}$$

the malicious CSP only computes $r' = \frac{rx-1}{x}$ to replace r and uses $sn' = 3$ to replace $sn = 2$ to get the same commitment $M(3, r')$ (If he does not know x, he cannot generate the same $M(3, r')$ and $M(2, r)$ with known polynomial-time algorithm). Then he computes $u' = y + e \cdot sn' \ mod \ q$ and $v' = s + er' \ mod \ q$ to make a response that he will provide the service $sn' = 3$ to the customer in order to cheat the verifier. Because $u' \cdot g + v' \cdot h = d + e \cdot M(3, r') = d + e \cdot M(2, r)$, the verifier will approve the transaction. When the customer does not get service he wants, he will ask for the supervisor to solve this dispute. The supervisor cannot regard this CSP as a malicious CSP since parameters in *Response Transaction* satisfies $u' \cdot g + v' \cdot h = d + e \cdot M(3, r')$ and the CSP actually provides service $sn = 3$. Consequently, this malicious behavior will lead to customers' losses.

D. Fairness Analysis

In FPPB, we demonstrate that the blockchain in FPPB is only an account to provide the proof rather than a method to solve off-blockchain economic disputes between trading parties. Moreover, FPPB protects on-blockchain privacy and fairness. In some off-blockchain scenarios where the CSP does not provide cloud service as he promises, customers need to resort to supervisors for help with transactions recorded on the blockchain.

Although FPPB cannot prevent P providing the incorrect service or nothing since the process of providing service is off blockchain, FPPB can provide these promises of P to the supervisor for judgements. The supervisor will mediate the dispute to check the request from C' address $addr_c$ and the response from P's address $addr_t$ by disclosing necessary parameters. For example, C convinces the supervisor that a *Request Transaction* belongs to him by initiatively disclosing his sn, y, s, r, satisfying corresponding commitments M and d. Other participants cannot forge these commitments if they do not own these parameters[18]. After that, the supervisor checks whether u and v appear in a *Response Transaction* by computing $u = y + e \cdot sn \ mod \ q$, $v = s + e \cdot r \ mod \ q$. If it is true, it means that the corresponding CSP actually promises to provide the service sn for C. If C does not receive the service sn as detailed on the blockchain, the supervisor can judge that P is actually a malicious provider and give punishment on him.

E. Privacy Analysis

1) Trading Relationship: For these customers, they use the one-time anonymous address in every service purchase, which prevents attackers linking their every transactions to analyze transaction information.

For these providers, first, FPPB adopts the stealth address to prevent linking $addr_p$ with the payment address F. Since the random data t is generated by the customer in every transaction, the payment address is unique. Participants on the blockchain cannot obtain their relationships with any data analysis as long as the provider does not use this one-time address any more. Additionally, the anonymous one-time temporary address $addr_t$ removes its relationship with $addr_p$. Consequently, if the trading partners do not disclose the relationship of these addresses initiatively, the transaction of P will not be disclosed.

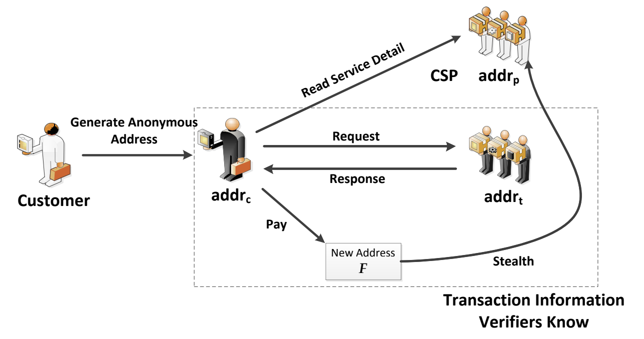

Figure 2. Transaction Information Verifiers Know

Moreover, these verifiers know the true trading relationship between three one-time anonymous addresses $addr_t$, F, and $addr_c$ as the dashed box in Figure 2. However, these addresses cannot be linked to their true owners as we mentioned above. Since transaction parameters are one-time, verifiers can get three completely different addressees in other transactions about the same trading partners.

2) Transaction Content: Transaction contents include the unique serial number of service. we hide the serial number in the Pedersen commitment and allows verifiers to check the uniformity of contents in *Request Transaction* and *Response Transaction*.

V. Experimental Evaluation

A. Description

We design and implement the prototype of FPPB on EthereumJ and conduct experiments to compare the performance of three types of transactions in FPPB with *Normal Transaction* (the basic money transaction without any cryptographic primitives). Experimental results show that compared with normal transactions without cryptographic primitives in the same blockchain platform, FPPB only slows down transaction slightly.

We conduct experiments under different transaction intervals, block sizes and block propagation time on EthereumJ, which are basic parameters of blockchain. The block size shows how many transactions a block can contain. The block propagation time shows the interval between two blocks created. The frequency of new transactions and new blocks are modeled as a Poisson process with a mean of 20ms and 15s, respectively. And the block is set to contain 224 transactions averagely by default. The block propagation time and the number of transactions a block can contain are set to be the same with those in Ethereum averagely now[20]. We send 20000 transactions with adopting the average processing time of transaction to estimate the performance of FPPB. q is set 256 bits to achieve the 128-bit security level.

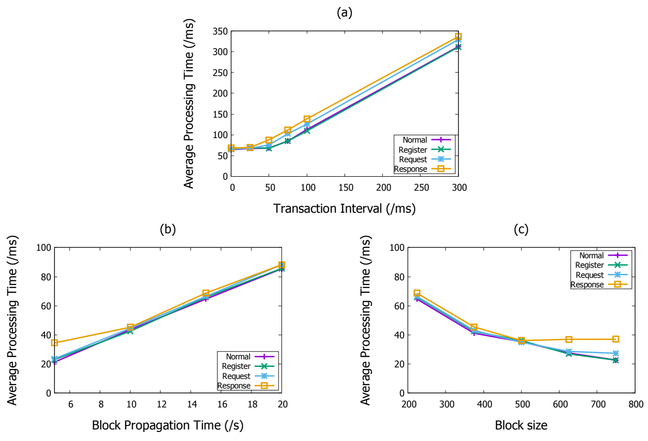

Figure 3. Effects of Different Parameters on Processing Time of Transaction

B. Effects of Transaction Interval

To evaluate effects of transaction intervals on FPPB and send transactions stochastically, we fix its block size and block propagation time with running a Poisson process on sending transactions. The transaction intervals are set from 0ms to 300ms.

Fig.3 (a) reports a linear relationship between transaction intervals and processing time roughly. The processing time increases with the growth of the interval. It is because the longer the transaction interval is, the more time a new transaction needs to be recorded on the blockchain. Since the cryptographical tools have been brought in *Request Transaction* and *Response Transaction*, they result in longer verification time. Thus, the processing time of transactions will be longer than other two types transactions.

C. Effects of Block Propagation Time

In this experiment, to evaluate the effects of block propagation time on transactions, we fix the transaction interval and the block size to test the performance of FPPB with different block propagation time. The block propagation time is set between 5s, 10s, 15s, 20s. The experimental results are plotted in Figure 3 (b).

In Figure 3 (b), the relationship between block propagation time and processing time is linear roughly. This is because the growth of block propagation time leads to more transactions pending in the pool to be taken away by a new block and less transactions recorded on the blockchain during the same period of time. Thus, the processing time of transactions will be longer as the block propagation time increases.

D. Effects of Block Size

In this experiment, to validate the effects of block size, we fix the transaction interval and the block propagation time to test the performance with different block sizes. In this experiment, a block can contain different number of transactions from 224 to 750.

In Figure 3 (c), the processing time of transaction gradually decreases with the growth of block size. The decreasing trend slows down and then keeps steady. It is because the more transactions a block contains, the faster each transaction can be recorded on the blockchain and the shorter its processing time is. With the growth of block size, even if a block is able to take more transactions, it can only take away all transactions in the pending pool at most since later transactions have not been sent into the pending pool yet. Consequently, when the block size is greater than a threshold, the processing time is not affected by the growth of block size any more and keeps steady as Figure 3 (c) shows. Additionally, with the growth of block size, the *Response Transaction* first reaches the lowest value since it needs the longest verification time. In Figure 3 (c), the lowest processing time of *Response Transaction* is 36.98 ms approximately when a block can contain 500 at most. The lowest value of other three transactions will come later.

VI. Conclusion

In this paper, we propose a fast and privacy-preserving method based on the permissioned blockchain (FPPB) for fair transactions in sharing economy. Based on cryptographic primitives, such as stealth address and zero-knowledge proof, FPPB realizes protecting the privacy of transaction information and ensuring the uniformity of transaction contents for trading fairness. Meanwhile, it does not break the verifying protocol and does not bring additional off-blockchain interactive communication. We use the case of Joint Cloud Computing to illustrate FPPB. Experiments show, compared with normal transactions without cryptographic primitives in the same blockchain platform, FPPB only slows down transaction slightly with the changes of different parameters.

Acknowledgment

This work was supported by the National Key Research and Development Program (Grant No.2016YFB1000101), the National Natural Science Foundation of China (Grant No.61379052, 61772030), the Natural Science Foundation for Distinguished Young Scholars of Hunan Province (Grant No.14JJ1026), Specialized Research Fund for the Doctoral Program of Higher Education (Grant No.20124307110015). The corresponding author is Yijie Wang and her email address is wangyijie@nudt.edu.cn.

References

[1] M. Swan, *Blockchain: Blueprint for a New Economy*. O'Reilly Media, Inc., 2015.

[2] J. Hamari, M. Sjöklint, and A. Ukkonen, "The sharing economy: Why people participate in collaborative consumption," *Journal of the Association for Information Science and Technology*, vol. 67, no. 9, pp. 2047–2059, 2016.

[3] H. Wang, P. Shi, and Y. Zhang, "Jointcloud: A cross-cloud cooperation architecture for integrated internet service customization," in *IEEE International Conference on Distributed Computing Systems*, 2017.

[4] J. Bonneau, A. Narayanan, A. Miller, J. Clark, J. A. Kroll, and E. W. Felten, "Mixcoin: Anonymity for bitcoin with accountable mixes," in *International Conference on Financial Cryptography and Data Security*. Springer, 2014, pp. 486–504.

[5] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," in *Security and Privacy (SP), 2014 IEEE Symposium on*. IEEE, 2014, pp. 459–474.

[6] A. Gibson, "An investigation into confidential transactions." *https: //github.com/ AdamISZ/ ConfidentialTransactionsDoc/ blob/master/essayonCT.pdf*, 2016.

[7] "Ethereumj," https://github.com/ethereum/ethereumjv.

[8] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, 2014.

[9] "Bitlaunder," https://bitlaunder.com.

[10] "Bitcoin fog," https://en.wikipedia.org/wiki/Bitcoin_Fog.

[11] G. Maxwell, "Coinjoin: Bitcoin privacy for the real world," in *Post on Bitcoin Forum*, 2013.

[12] "Joinmarket," https://bitcoinadvice.com/joinmarket-enhanced-bitcoin-transaction-privacy/.

[13] K. Gai and M. Qiu, "Blend arithmetic operations on tensor-based fully homomorphic encryption over real numbers," *IEEE Transactions on Industrial Informatics*, 2017.

[14] S. Noether, A. Mackenzie *et al.*, "Ring confidential transactions," *Ledger*, vol. 1, pp. 1–18, 2016.

[15] C. Garman, M. Green, and I. Miers, "Accountable privacy for decentralized anonymous payments," in *International Conference on Financial Cryptography and Data Security*. Springer, 2016, pp. 81–98.

[16] N. Van Saberhagen, "Cryptonote v 2. 0," 2013.

[17] O. Goldreich and Y. Oren, "Definitions and properties of zero-knowledge proof systems," *Journal of Cryptology*, vol. 7, no. 1, pp. 1–32, 1994.

[18] T. P. Pedersen *et al.*, "Non-interactive and information-theoretic secure verifiable secret sharing." in *Crypto*, vol. 91, no. 7. Springer, 1991, pp. 129–140.

[19] J. Mirkovic and P. Reiher, "A taxonomy of ddos attack and ddos defense mechanisms," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 39–53, 2004.

[20] "Etherscan," https://etherscan.io/.