PoPF: A Consensus Algorithm for JCLedger

Fu Xiang, Wang Huaimin, Shi Peichang, Mi Haibo
Science and Technology on Parallel and Distributed Laboratory
College of Computer, National University of Defense Technology
Changsha 410073, China
fuxiang13@nudt.edu.cn

***Abstract*—JointCloud is a new generation of cloud computing model which facilitates developers to customize cloud services. JCLedger is a blockchain based distributed ledger for JointCloud computing which can make cloud resources exchange more reliable and convenient, and it is the combination of JointCloud and BlockChain. One of the most important elements for creating JCLedger is the consensus algorithm. PoW (Proof of Work) is the consensus algorithm for Bitcoin, which is proved to be quite safe but needs much computing power. The original PoW is not suitable for JCLedger because the identities of participants are not equal in computing power, which may lead to accounting monopoly, and the throughput cannot satisfy the requirement of the massive and high-frequency transactions in JointCloud. In this paper, we propose a PoW based consensus algorithm called Proof of Participation and Fees (PoPF), which can save much computing power and handled transactions more efficiently for JCLedger. In our design, only the candidates have the opportunities for mining and the candidates are chosen according to the ranking which is determined by two factors: the times of the participant to be the accountant and the fees the participant has paid. The difficulty for candidates of solving the PoW hash puzzle is different (the higher ranking means easier for mining). The simulation experiment shows that the distribution of accountants is well-balanced, that is to say, the unequal computing power of participants in JointCloud is shielded, and all the users who have enough contribution in JCLedger will have the opportunities to be accountants.**

Keywords—JCLedger; JointCloud; consensus algorithm; PoW

I. INTRODUCTION

The trend of Economic Globalization has provided the trade in global cross-border goods and services with unprecedented opportunities [1]. It also brought new demand for full-time and global service in cloud computing which is explosive, global and diverse. However, the new resource needs cannot be satisfied by a traditional single cloud service provider's service capability. JointCloud is a cross-cloud cooperation architecture for integrated Internet service customization, which is funded by China's Ministry of Science and Technology as a part of the National Key Program for Cloud Computing and Big Data. JointCloud not only focuses on a vertical integration of cloud resources but also a horizontal cooperation among CSPs in the form of service-oriented computing, by which CSPs evolve along with the JointCloud ecosystem to better serve globalized computation at low cost, high availability and assured QoS [2].

To make cloud resources and value exchange more reliable and convenient in JointCloud, we have proposed JointCloud Collaboration Environment (JCCE) architecture in our former work. JCCE is mainly to search for the fair trade and interconnection among different CSPs [3]. The key research on JCCE is how to effectively support services like providing auction, bid, registration of service capability, service query, service binding, auditable and traceable transaction behavior. JCCE contains several services which provide basic services together for enabling the cooperation among independent clouds based on distributed ledger. The distributed ledger is also called JCLedger which was proposed in our former work [4].

In this paper, we propose a PoW based consensus algorithm for JCLedger, which can avoid the massive cost of computing power, shield the unequal computing power of participants to give them equal opportunities for accounting. It is more efficient to handle the massive and high-frequency transactions in JCCE than the original PoW. The remainder of the paper is structured as follows: Section II introduces the related work; The design of PoPF is presented in Section III; Section IV shows the simulation experiments and analyzes the result; The discussion is presented in Section V and Section VI concludes this paper.

II. RELATED WORK

As shown in Fig. 1, JCLedger is a distributed ledger based on BlockChain. BlockChain was first proposed in 2008 and was implemented as Bitcoin in 2009 [5]. The BlockChain is essentially an append-only data structure maintained by a set of nodes which do not fully trust each other. Nodes in the BlockChain agree on an ordered set of blocks which contains multiple transactions [6], [7], [8]. A complicated but secure mechanism based on asymmetric cryptography such as ECC or RSA has been implemented to protect BlockChain from tampering in distributed systems [9]. BlockChain allows transactions to take place in a cloud computing scene without the need of a central intermediary. As a result, BlockChains can significantly save the cost and improve the efficiency of value exchange in JCCE.

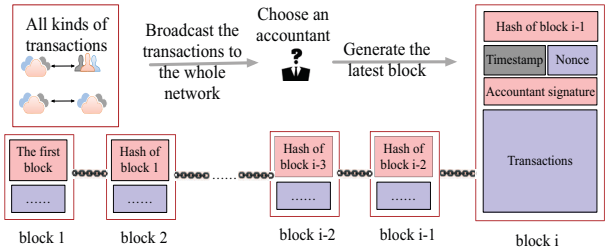

Fig. 1. JCLEdger: An example of blockchain

0-7695-6394-5/18/$31.00 ©2018 IEEE
DOI 10.1109/SOSE.2018.00034

One of the most important technologies for BlockChain is the consensus algorithm. The main function of the consensus algorithm is to select an accountant from all the users/nodes for each block. The ledger/BlockChain is composed of blocks packed by accountants. In JCCE, the participants include cloud services brokers (CSB), cloud services customers (CSC), cloud services providers (CSP), etc. The computing power of these participants is extremely uneven. CSPs always have high-performance servers while CSCs just have ordinary PC. PoW consensus algorithm was introduced by Bitcoin. The original PoW consensus algorithms don't only bring a waste of computing power but also cause the unbalanced distribution of the accounting right. The throughput of PoW is very slow. Considering the massive and high-frequency transactions in JCCE, the scalability of PoW should be developed to make it efficient enough for JCLedger to achieve real-time performance.

The Bitcoin system's approach to consensus is called Proof of Work. In PoW, each node is calculating solving a hash puzzle which is a complicated computational process. All participants have to calculate the hash value continuously by using different nonces until the target is reached. The cost of massive computing power is not the intention of the consensus algorithm but a way to ensure that the accountant selection cannot be predicted and manipulated by a few people. Of course, we hope to find a way not wasting too much computing power, but still meeting the security needs to select the accountant. In Bitcoin, the size of each block is limited to 1MB, the number of transactions processed per second is only about 4 (theoretically up to 7). The low transaction processing speed and at least six confirmation security mechanism lead users to wait at least one hour for a transaction to be confirmed. Increasing the block size may solve the problem temporarily, but it cannot be increased unlimited, which will bring security risks [11]. Ethereum [12] also uses PoW as its consensus algorithm, and it can only handle about 7 transactions per second.

The consensus algorithm Proof of Stake is an energy-saving alternative to PoW. Instead of demanding users to find a nonce in an unlimited space, PoS requires people to prove the ownership of the amount of currency because it is believed that people with more coins would be less likely to attack the network. It is just a trade-off between computing power waste and centralization risk. PoS has already been achieved by Peercoin [13] and NXT [14] with different ideas. Ethereum will switch its consensus algorithm from PoW to PoS. DPoS [15] is similar to POS but requires fewer nodes to validate the block. The significant difference between PoS and DPoS is that PoS is direct democratic while DPoS is representative democratic. Stakeholders elect their delegates to generate and validate a block [16]. DPoS is the backbone of Bitshares [17].

III. Design of PoPF

The name of the approach is called Proof of Participation and Fees. We choose the accountant candidates for each block according to the ranking which is determined by the users' *participation* (the times to be an accountant) and *fees* (the fees a user paid) in the previous transactions. The top *n%* ranking users are the accountant candidates. They compete for the right to packing the next block through solving the hash puzzle, which is the same as the PoW mining in Bitcoin system. However, the difference is that the mining difficulty for each user is not the same, it's easier for the higher-ranking users to win the competition compared to those with lower-ranking. In a word, the system sets different mining difficulty for different users according to their *participation* and *fees*. That's why the algorithm is called PoPF. Fig. 2 shows the usage of PoPF in JCLedger.

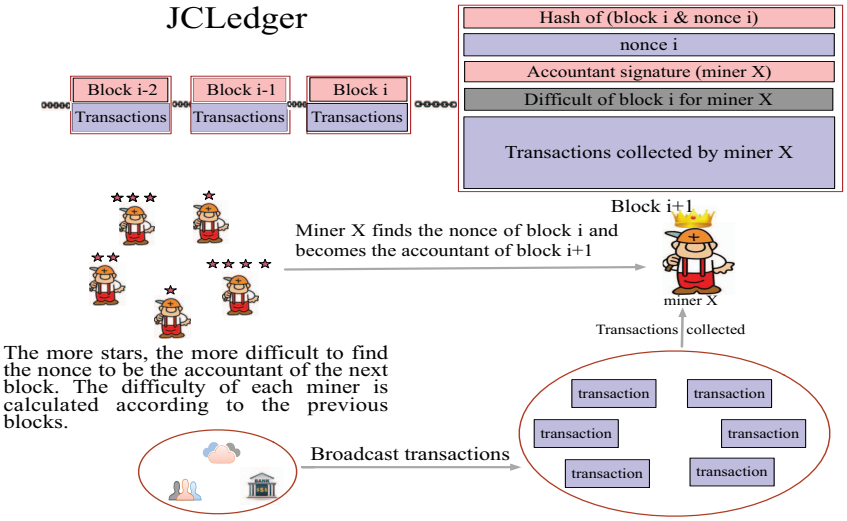

Fig. 2. The usage of PoPF in JCLedger.

A. The Block-Structure

Since our PoPF algorithm needs a certain number of users (e.g., the number of accountant candidates is set to *n*, then PoPF needs at least n users) as a basic running condition, at the beginning of JCLedger, we cannot run PoPF. We use the original Bitcoin PoW as the initial consensus approach until the operating condition for PoPF is satisfied. Each node will check the PoPF operating conditions every time it adds a block. When the number of users in the history blocks is greater than

the set threshold *n*, the nodes switch the consensus approach to PoPF. As shown in Fig. 3, the block-structure is composed of two parts, which are the PoW structure and PoPF structure.

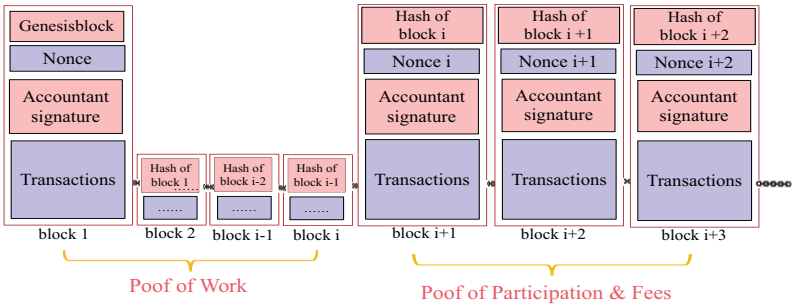

Fig. 3. The Block-Structure for PoPF.

The definition of elements in the structure for PoPF is shown in TABLE I. We should notice that nonce *i* in PoW is provided by accountant *i*, while in PoPF, it is provided by accountant *i+1*. This is because in PoPF, the accountant is selected first, then the accountant collects transaction to pack a block.

TABLE I. THE ELEMENTS' DEFINITION FOR POPF

Element Name	Explanation
Hash of (block i & nonce i)	Block i represents the hash value of block i
Nonce i	Nonce i is mined by the accountant of block i+1
Accountant signature	The signature of accountant of block i+1
Difficulty of block i	The difficulty for the accountant
Transactions	Transactions collected by the accountant

B. *Ranking and the mining difficulty*

Definition of terms:

- *R(x)*: The ranking of user *x*;
- *F(x)*: The fees user *x* paid since the last time he was the accountant;
- *M(x)*: The number of times for user *x* as an accountant;
- *D(x)*: The mining difficulty for user *x*.

The PoPF algorithm is shown in Fig. 4. If a node wants to be an accountant, he must maintain all the historical data, through which *F(x)* and *M(x)* can be easily calculated. Then *R(x)* can be calculated by *F(x)* and *M(x)* using the following formula:

$$R(x)=\frac{F(x)}{M(x)+1}.$$

After getting *R(x)*, we can set rules for calculating *D(x)*. We recommend that candidates in the same ranking area have the same difficulty. The number of candidates in each ranking area can be set according to the scale of users in the system (e.g., the top 10% candidates are in the same ranking area). The *simulation experiment* part in this paper will analyze the difference brought by the scale of the ranking area.

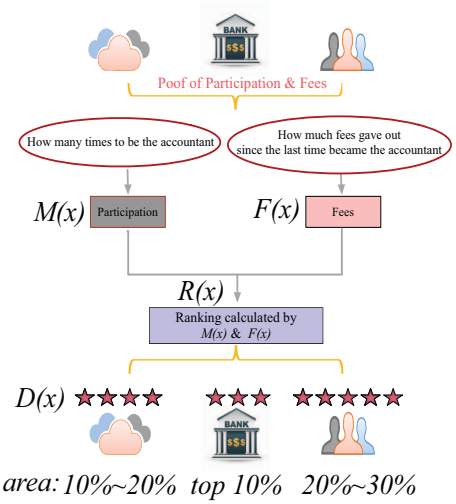

Fig. 4. The PoPF algorithm .

C. *How to compete*

As described above, every user knows whether himself is one of the accountant candidates by calculating his ranking through the historical data of JCLedger. As an accountant candidate, the user calculates his mining difficulty by his ranking firstly and then begins to mine. The user keeps searching for the nonce until a value is found that gives the block's hash the required zero bits. When a miner finds a nonce which makes the block's hash value satisfy the difficulty, he sends the nonce to the other accountant candidates to declare that he will be the next accountant immediately. After the other accountant candidates receive this nonce, they first verify whether the nonce satisfies the difficulty of the sender and then send confirmations back to the miner if the nonce is correct. Algorithm 1 shows the detail of validating a nonce. Only if the miner receives more than half of the accountant candidates' confirmations within a limited time will he be the accountant of the next block. What if a user receives two or more nonces from different accountant candidates at the same time although the probability of this situation is very low? The receiver can only choose one candidate to send the confirmation to just as voting for a unique accountant. If none

of these accountant candidates receives the required number of confirmations within the limited time, the competition of finding the nonce will restart. To encourage the candidates, the accountant will choose the first *k* candidates that send the confirmation to share the transaction fees with. This mechanism not only guarantees each block just has one accountant, but also solves the problem of selfish mining, which means there doesn't exist forks that may lead to double spending in the distributed ledger.

Algorithm 1 Validating a nonce

```
1:  begin
2:    while(true)
3:      mining(miner, block)
4:        if receive a nonce from miner x
5:          result = validate_nonce(nonce, block, miner x)
6:          if(result)
7:            stop mining and send confirmation to x
8:            break
9:          else
10:           Dump the nonce
11:           continue
12:         end if
13:       end if
14:   end while
15:   wait for the next block
16: end
```

D. Packing a block

There are three steps to pack a block for an accountant. Firstly, verify the received transactions. There are two things to verify which are the sender's the account balance and the signature. If a transaction is legal, the receiver will save it in the cache and broadcast it to the whole network. If not, it is just dumped. Secondly, sort the transactions by their receiving time in the cache which have been verified before. Finally, pack the transactions and his signature into a block and broadcast it to the whole network. The first two steps can be done by every user in the network, but the third step is only for the accountant. What's more, the number of transactions in a block is limited, and the accountant must broadcast the block within a limited time. Algorithm 2 shows the details of packing a block.

Algorithm 2 Packing a block

```
1:  begin
2:    while(true)
3:      mining(miner, block)
4:        if find a nonce before receive one
5:          broadcast (miner, block, nonce)
6:          if not receive enough confirmations in time T
7:            break
8:          collect the transactions and generate nextblock
9:          broadcast(miner, nonce, nextblock)
10:         add nextblock to the ledger
11:         break
12:       end if
13:   end while
14: end
```

E. Adding a block to the distribute ledger

When a user receives a block, he should verify whether the block is legal. There are four items to check, which are the signature of the accountant, the transactions in the block, whether the number of transactions and the time is beyond the limitation. If the block is legal, the user adds it to the distributed ledger and set the accountant's fees to zero and the number of being an accountant for the accountant plus one. Then the user will begin to join the next block's competition. If the block is illegal, the user dumps the block, and the accountant will be punished (e.g., set a very high difficulty for him for the next block). Then the user will restart to compete for being the accountant. Algorithm 3 shows the details of validating a block.

Algorithm 3 Validating a block

```
1:  begin
2:    // after receiving a nonce from miner x
3:    while(true)
4:      wait for the next block
5:        if receive the block in time T
6:          result=validate_blaock(block, nonce, miner x)
7:          if(result)
8:            add block to the ledger
9:            break
10:         else
11:           Dump the nonce and block from miner x
12:           break
13:         end if
14:       end if
15:       Dump the nonce from miner x
16:   end while
17:   restart mining
18: end
```

IV. Simulation experiments

We simulate the generation of 100 blocks in JCLedger. In the simulation, there are 1000 nodes which have all the historical data, and they send transactions to each other randomly. The number of accountant candidates is set to 100, which means only the top 100 users can join the competition for accounting right in each block. But with the number of transactions increasing, the ranking is dynamically changing. The hash function we choose is SHA256. In the first simulation experiment, we set the number of candidates in each ranking area to 1, which means every candidate has a different difficulty. In the second simulation experiment, the number of candidates in each ranking area is set to 10, which means the top 10 candidates have the same difficulty for mining just like the situation in Fig. 4.

Fig. 5 and Fig.6 are the results of experiment 1. Fig. 5 shows the distribution of accountant among all the 1000 users. Fig. 6 shows the accountants ranking. We can see that most of the accountants are ranking top 1.

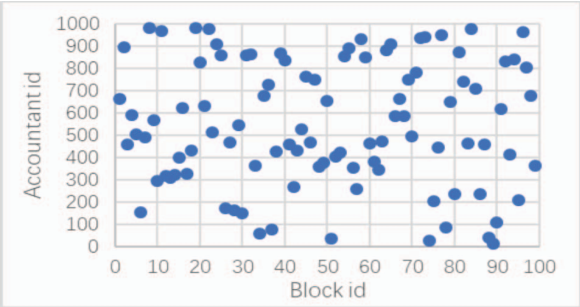

Fig. 5. The accountant id in simulation experiment 1.

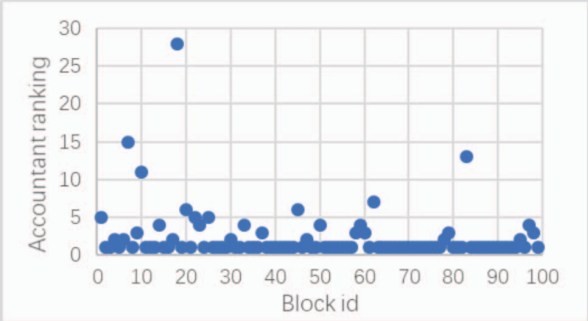

Fig. 6. The accountant ranking in simulation experiment 1.

Fig. 7 and Fig. 8 are the results of experiment 2. Fig. 7 shows the distribution of accountant among all the 1000 users. Fig. 7 shows the accountants ranking. We can see that most of the accountants are ranking top 10.

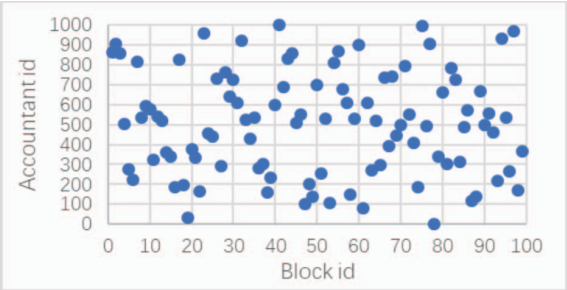

Fig. 7. The accountant id in simulation experiment 2.

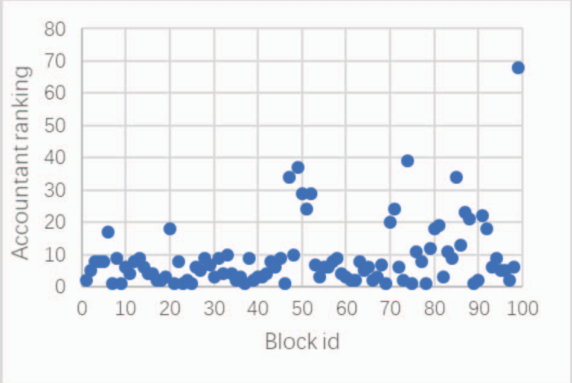

Fig. 8. The accountant ranking in simulation experiment 2.

According to the results, the distribution of accountants is well-balanced, the unequal computing power of participants in JointCloud is shielded. The ranking, which is determined by the participation and fees, is the main factor for winning the competition of accounting right. All the users who have enough contribution in JCLedger will have the opportunities to be accountants.

V. Discussion

Tamper a block. For tampering the block, the attackers must hold more than 50% of the computing power in the network. Although every node with full data is a verification node, only n% of the users are the accountant candidates. What if the attackers control half of the accountant candidates which is much easier than holding 50% of the computing power in the entire network? In this situation, the attacker can make any candidate be the accountant, but the accountant can't pack an illegal block because he would be caught easily. So this kind of attack can't do any harm to JCLedger. What about the attacker make more than half of the candidates vote for two or more accountant candidates at the same time? No doubt there may exist two or more accountants for one block. Because one can only vote for one candidate, the attackers may be found out easily, and we can punish them by removing them from the accountant candidates and restart the competition.

Computing power. In PoPF, only the candidates can join the competition for accounting right, which means not every node using their computing power to solve the hash puzzle. What's more, because of the ranking-difficulty rule, the difficulty of mining for the top-ranked candidates is very low, so they can save much computing power. Compared to the original PoW, PoPF is an energy-saving algorithm.

Throughput. Unlike PoW for Bitcoin, which can only handle 7 transactions every second in theory, we first select the accountant, then the chosen accountant packs a block in PoPF. According to this mechanism, it achieves significantly higher throughput than PoW, and the throughput of PoPF is only limited by the processing speed of the individual nodes.

VI. Conclusion

In the background of Economic Globalization, JointCloud computing is a prospective research area. In this paper, we study consensus algorithm of BlockChain in JointCloud computing. This study has high theoretical value and essential application prospects which can support the development of JCLedger. We propose a PoW (Proof of Work) based consensus algorithm called Proof of Participation and Fees (PoPF) for JCLedger. Compared to PoW, PoPF can save much computing power and handled transactions more efficiently without bringing additional security risks. Our future work will concentrate on the study of smart contract in BlockChain which is an automatic agent that can make the trading in JointCloud more intelligent.

Acknowledgment

This work is supported by The National Key Research and Development Program of China (2016YFB1000100) and the

National Natural Science Foundation of China (61772030, 61502510).

REFERENCES

[1] Gao S. Economic Globalization: Trends, Risks and Risk Prevention[J]. Cdp Background Papers, 2000.

[2] Wang H, Shi P, Zhang Y. JointCloud: A Cross-Cloud Cooperation Architecture for Integrated Internet Service Customization[C]// IEEE, International Conference on Distributed Computing Systems. IEEE, 2017.

[3] Shi P, Wang H, Yue X, et al. Corporation Architecture for Multiple Cloud Service Providers in JointCloud Computing[C]// IEEE, International Conference on Distributed Computing Systems Workshops. IEEE, 2017:294-298.

[4] Fu X, Wang H, Shi P, et al. JCLedger: A Blockchain Based Distributed Ledger for JointCloud Computing[C]// IEEE, International Conference on Distributed Computing Systems Workshops. IEEE, 2017:289-293.

[5] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf, (2008).

[6] Dinh T T A, Liu R, Zhang M, et al. Untangling Blockchain: A Data Processing View of Blockchain Systems[J]. 2017.

[7] Zheng Z, Xie S, Dai H N, et al. Blockchain Challenges and Opportunities: A Survey[J]. 2016.

[8] Yuan Y, Wang F. Blockchain: The State of the Art and Future Trends [J]. Acta Automatica Sinica, 2016, 42(4):481-494.

[9] Lee B, Lee J H. Blockchain-based secure firmware update for embedded devices in an Internet of Things environment[J]. Journal of Supercomputing, 2016, 73(3):1-16.

[10] Gervais A, Karame G O, Glykantzis V, et al. On the Security and Performance of Proof of Work Blockchains[C]// ACM Sigsac Conference on Computer and Communications Security. ACM, 2016:3-16.

[11] Bitcoin block size limit controversy, 2016. Available from:https://en.bitcoin.it/wiki/Block_size_limit_controversy.

[12] G Wood. Ethereum: a secure decentralised generalised transaction ledger. 2014.

[13] Peercoin. Available from: https://peercoin.net/, visited Sep 2016.

[14] NXT. Available from: http://nxtchina.org/, visited Nov 2016.

[15] Larimer D. Delegated proof-of-stake white paper [Online], available: http://www.bts.hk/dpos-baipishu.html, (2014)

[16] Delegated Proof of Stake (DPOS) vs Proof of Work (POW) (2015). http://bytemaster.github.io/bitshares/2015/01/04/Delegated-Proof-of-Stake-vs-Proof-of-Work/

[17] Daniel Larimer, Charles Hoskinson,Stan Larimer. A Peer to Peer Polymorphic Digital Asset Exchange. 2013.

.