Peer-to-Peer Networking and Applications (2020) 13:2251–2259
https://doi.org/10.1007/s12083-020-00905-6

Achieving reliable timestamp in the bitcoin platform

Guangkai Ma[1] · Chunpeng Ge[1,2] · Lu Zhou[2]

Received: 25 July 2019 / Accepted: 19 March 2020 / Published online: 13 May 2020

Abstract
Blockchain, the underlying technology of the Bitcoin cryptocurrency, is an innovation of information technology. The blockchain technology has been widely applied in the evidence storage scenarios to prove that an event occurred at a certain time due to its publicity and immutability. However, the timestamp of a block in the blockchain is introduced by the blockchain node and can be manipulated in hours. This will either lead the failure of the evidence storage system built on top of the blockchain platform or increase the risk of double spending of the blockchain platform itself. In this paper, we introduced an optimized blockchain timestamp mechanism. We narrow the range of the timestamp in a block to an average of ten minutes by leveraging an outside trust timestamp service to the blockchain consensus. Finally, we present a security analysis of the proposed scheme.

Keywords Blockchain · Bitcoin · Timestamp service · Time-jacking

1 Introduction

The timestamp service [25] is a most common requirement for today's information systems such as evidence storage system [3], secure logging system [18] and so on. In these systems, a user storages a piece of data in the system and prove to the public later that the data is stored before a certain time. However, the existing timestamp service relies on a trust third party to sign on the data to proof that the timestamp with the signed data is correct.

In order to eliminate trusting on a third party, many decentralized timestamp service [12] have been proposed based on the Bitcoin cryptocurrency [20] due to its publicity, immutability and authenticity. In these decentralized timestamp service [12], a node in the Bitcoin network issues a transaction that contains a specific claim. Later, when this transaction was included into a Bitcoin block by a miner, it will visible by the whole Bitcoin network. In such a manner, the user can present the timestamp of the claim using the timestamp of the block.

However, the timestamp in the Bitcoin platform can be manipulated in hours [4] which will result in the inaccurate timestamp service. Extremely, the timestamp of a block may later than its previous blocks and thus causes the failure of the timestamp service built on top of it. Moreover, the inaccurate timestamp in the Bitcoin may cause the time jacking attack which may increase the probability of a double spending attack.

1.1 Related work

The concept of Bitcoin was first proposed by [20] in 2008. The Bitcoin is a decentralized peer to peer network, where each node stores a transcript of the whole ledge. The nodes, called miners, competed to solve the mining puzzle, and the lucky miner who found the answer to the puzzle got the right to add a new block to the ledge. Once the block is added to the Bitcoin network, the miner will get a reward in the form or Bitcoin cryptocurrency. Since the advent of Bitcoin, the underlying technique, blockchain, has attracted the academia and industry. Recently, the blockchain technology

This article is part of the Topical Collection: *Special Issue on Security and Privacy in Machine Learning Assisted P2P Networks* Guest Editors: Hongwei Li, Rongxing Lu and Mohamed Mahmoud

✉ Lu Zhou
luzhouua@163.com

Guangkai Ma
guangkaiyanlu@gmail.com

Chunpeng Ge
gecp@nuaa.edu.cn

[1] Nanjing University of Aeronautics and Astronautics, Nanjing, 210000, China

[2] University of Aizu, Aizuwakamatsu, 965-8580, Japan

has been widely adopted in many application scenarios including the timestamping service [25], the evidence storage system [3], the supply chain finance [7], the cloud computing [8], the big data technology [26], industrial artificial intelligence (IAI) [13], Internet of Things (IoT) [21], and privacy-preserving technology [27, 28], due to its properties. *Decentralization,* in the blockchain, every node can join the peer to peer network, and all the peers are equal. Every node can store and validate transactions without a central server. *Publicity,* the data in the blockchain is open to all peers. The data's source and trace are transparent in the system. *Immutability,* the data records stored in the blocks forever. It is unable to tamper the record ever stored in the network. *Authenticity,* the data stored in the Bitcoin network can be authenticated by all the nodes in the network.

However, all these systems assume the timestamps of the blocks in the Bitcoin network are accurate. Recently, Apostolaki et al. [2] proposed the timestamp in the Bitcoin network can vary in hours which may make the Bitcoin helpless in the time sensitive applications. Moreover, they pointed out that, the manipulated timestamp may cause the time jacking attack to the underlying Bitcoin itself and thus increase the possibility of double spending the Bitcoin network. To achieve reliable timestamp for the Bitcoin platform, Szalachowski [24] proposed a reliable timestamp service for the Bitcoin network. In their scheme, whenever a new block is added into the Bitcoin network, a verifier issues a transaction that contains a trusted timestamp. Thus, the timestamp of a block will be narrowed by two transaction. However, the approach works in a suffixed way, which means the block with incorrect timestamp has already been added into the network when its timestamp is found incorrect.

1.2 Our contribution

In this paper, for the first time, we present a reliable timestamp service for the Bitcoin network. In our scheme, we leveraged an outside trusted timestamp authority (TSA) in the initial mining phase. We added a trusted timestamp into to input of the mining puzzle. By integrating the chronological order property of the Bitcoin, the accuracy of timestamps are narrowed in about 10 minutes. Moreover, the security analysis demonstrates that our scheme can prevent the time jacking attack.

1.3 Roadmap

The remainder of this paper is organized as follows. In Section 2, we provide some background and preliminaries. Section 3 presents our system architecture and concrete protocol. In Section 4, we analyze the security of the proposed protocol. Finally, in Section 5, we conclude this work and present some future researches.

2 Background and preliminaries

2.1 Bitcoin block structure

Bitcoin is decentralized ledge which consists of blocks and each block is connected in a chain manner. The block is the basic data structure in the bitcoin. A block consists of a block header and a block body. The block header contains the version number, previous hash, Merkle root, timestamp, target hash, and nonce. The block body are a set of transactions organized in the merkel tree manner. Details of the block are presented in Fig. 1 and Table 1.

The block size and the version number is to describe this block's size and the version for better communication and verification. The previous hash is the previous block header's hash value. It can be used to validate this block's father block to approve the link.

The Merkle root is the most important data structure to do quick induction and verification on the data in this block. The Merkle root's binary tree is presented in Fig. 1. Transactions are represented as leaves of a Merkle tree [19] whose root is in the header. With the Merkle root, lightweight or resource-limited nodes can also join in the bitcoin to contribute the power to validate data easily. The timestamp is the necessary data to record the accurate time when this block is found out. It is also the timestamp certification for the inner transactions. Timestamp can be used to provide proof of existence, which helps to build an evidence storage system and other time-sensitive systems.

The target hash is the key to the next block. Its difficulty depends on this block header's hash value and the adjustment by the system. The target hash generally is a string of zero bits, the required zero bits is the target hash difficulty.

The nonce is one solution to the previous block's target hash. Every node competes to find the nonce first to get the right packing transactions to the new block. The nonce's hash value can satisfy the required zero bits.

The new valid block will link to the previous block by the previous hash. All nodes continue competing to find the new nonce to the new block. Repeating this mechanism, again and again, blocks will be a chain that makes the block before harder and harder to change. If an attacker tries to modify a past block, it has to rebuild all the blocks after the target block. Difficulty to change the information in the block makes bitcoin has credit.

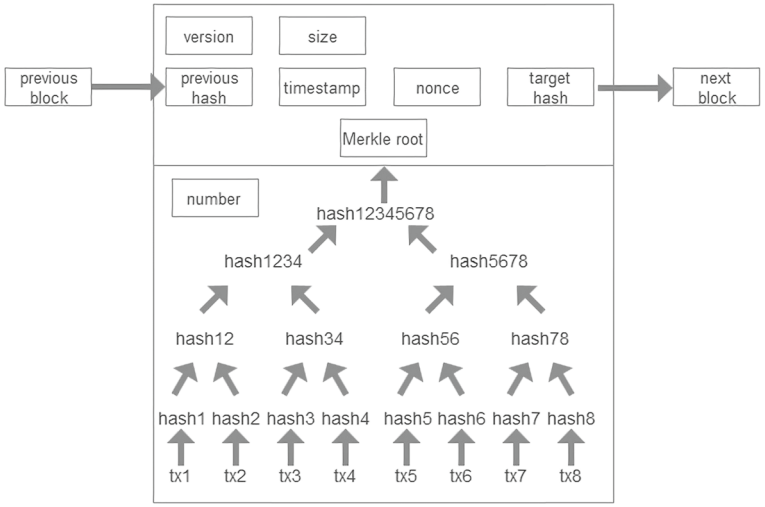

Fig. 1 The data structure of the block

2.2 Proof of work

Proof of Work is a consensus mechanism to confirm a node's work. Each node validates the broadcasted transactions and added them into a block. But there is only one node that can have the right to publish the new block. The consensus to the new block's validity is the basis of Proof of Work. In bitcoin's block, there is a nonce to adjust the workload. The target hash difficulty begins with a string of zero bits, only the proper nonce can make the new block's hash value satisfy the required zero bits. Finding the first proper nonce is proof of work. The other nodes can not change the block without redoing this work. Later blocks will also improve the difficulty to change the history blocks.

Hash function's unidirectional property makes the nonce is easy to validate but is very computationally difficult to find. The only approach way is to try all possible nonces one by one. The target hash begins with a number of zero bits. Assume the zero bits is n, then an average of 2^n attempts are needed to find the solution nonce. The nonce's hash value should be equal to or lower than the target hash. The more required zero bits in the target, the more difficult to find a proper nonce solution.

The target hash is adjusted every 2016 blocks to keep the block generation speed at 10 minutes a block on average, which helps to keep the system stable. The new target hash T is given by

$$T = \frac{T_{prev} * t_{actual}}{2016 * 10min}$$

T_{prev} is the previous target hash, t_{actual} is the actual time to generate these 2016 blocks. With the increasing faster mining machine, the time to generate 2016 blocks is becoming shorter, which makes the target hash is smaller. Smaller target hash needs more zero bits makes more difficult to find solution nonce. Proof of work has the autonomy to stabilize itself.

Proof of Work is to approve that the miner node finds the right and valid nonce. Once the effort has been expended to find this nonce, the block cannot be changed without redoing this work. Blocks after blocks, the difficulty to modify one previous block is harder and harder.

Table 1 The detail information in the block

Field	Subfield	Size	Description
Block Header	Block Size	4 bytes	the Size of the Block
	Version Number	4 bytes	the Protocol Version of this Block
	Previous Hash	32 bytes	Previous Block Header's Hash Value
	Merkle Root	32 bytes	All Transactions' Merkle Tree Root Hash Value
	Timestamp	4 bytes	the Unix Creation Time of this Block
	Target Hash	4 bytes	the Target Difficulty to Find this Block for POW
	Nonce	4 bytes	the Solution to the Target Hash
Block Body	Number	1-9 bytes	the Number of Transactions
	Transactions	Depend on the Number	Transaction Details

During the mining phase, every transaction is broadcasted to the whole network. Each node collects new transactions and validates them. Valid transactions will be packed into a new block. The nonce is one solution to the previous block's target hash. Every node competes to find the nonce first to get the right packing transactions to the new block, which also means the node works on finding the proof of work for the new block. A special empty-input coinbase transaction in the block allocates a fixed amount of new bitcoins to the miner, thus the miners have the incentive to contribute their resources to the network [30]. The new block should broadcast to the network immediately to get validation. And the next block will follow this new block's target hash to find next nonce.

2.3 Timestamp in the bitcoin

The timestamp in the bitcoin has multiple effects. The timestamp is not only the birth time of the block but also input to calculate the target hash. Timestamps' statistical data helps the system to change the target hash which can control the block creating speed. Bitcoin has some consideration to keeping the production rate stable at almost one block ten minutes. The target hash value adjusts every 2016 blocks.

In Bitcoin, the timestamp may have less accuracy to provide freshness property. For example, two new blocks are found out in a very short time. Maybe the first one's timestamp is later than the second timestamp. But the block order proves the first timestamp is the former. If these two blocks have related transactions both sensitive to time, logic confusion to the data will appear, which can result in immeasurable impact.

Each node has two counters to represent two timestamps, one is its local time and the other is the network time. These two counters' difference should be no more than 70 minutes otherwise the network time counter will revert to the local system time counter's time. The network time is the median time of the node's peers timestamp [4]. During the invalidation by all nodes, the timestamp has a conventional legitimate range to judge if the timestamp is valid or not. Assume the block's timestamp is T, then T should be in [T_0 ,T_1+2h] interval. T_0 is the median time of the previous 11 blocks. T_1 is the present network time. h means an hour.

The nodes' time can't be accurate all the time. The accuracy is also limited to hours. Less accuracy will cause several difficult problems. Firstly, it can't provide strict time information for some time-sensitive applications. Secondly, it is possible to bring threatens to the data and the system. Selfish mining and time jacking can take advantage of the vulnerability of timestamp to attack. So we have to design a better mechanism to get a more reliable timestamp to react to the new threatens.

2.4 Trusted timestamp authority (TSA)

The trusted timestamps are guaranteed accuracy and security by authoritative time agencies. The timestamp generated by the third-party time stamp service organization is mainly used to solve the existence and content integrity proof of the data message and is applicable to the legal validity certificate of the data message.

Trusted Timestamp Authority (TSA) should define in the RFC 3161 standard [1]. Commercial and free timestamp service providers can be found on the internet. In this paper, we suggest NTSC UniTrust Time Stamp Authority. As a core infrastructure for resolving reliable electronic signatures and securing the original form of data messages, the trusted timestamps issued by the Trust Time Stamp Service Center have been widely used in judicial, administrative enforcement, intellectual property protection, archives, finance, securities insurance, e-commerce, health care, telecommunications, and other fields.

Trusted timestamping is a process that could securely track the creation and modification times of digital data, in addition, it guarantees the existence and integrity of the data [23]. The client sends a digital file's hash to TSA, where the file is signed with the current time [11].

The communications between nodes and TSA should use public key cryptography system to protect data security. Besides traditional algorithms such as RSA and ECC, some new public key cryptography called proxy re-encryption scheme [10], the searchable encryption [16] and attribute-based encryption (ABE) [17] can also be used in the communications for different requirements. What's more, TSA can construct blockchain-based PKI [14] to provide advanced security.

TSA also can have new forms. SSL/TLS servers, web servers of reputable organizations (e.g. mozilla.org) or high-profile websites (like google.com or live.com) can be used as TSAs [24].

3 System

3.1 System architecture

Facing the inaccurate timestamp problem, the first natural solution is to use Trusted Timestamp Authority (TSA) to give accurate timestamp. If each node can get an accurate timestamp from TSA before digging the next block, and then use this accurate timestamp as an input to find the next block, the timestamp will be more accurate with this uninterrupted time synchronization. When a new block is found, the finder node should communicate with TSA to get accurate timestamps secretly and then broadcast the new

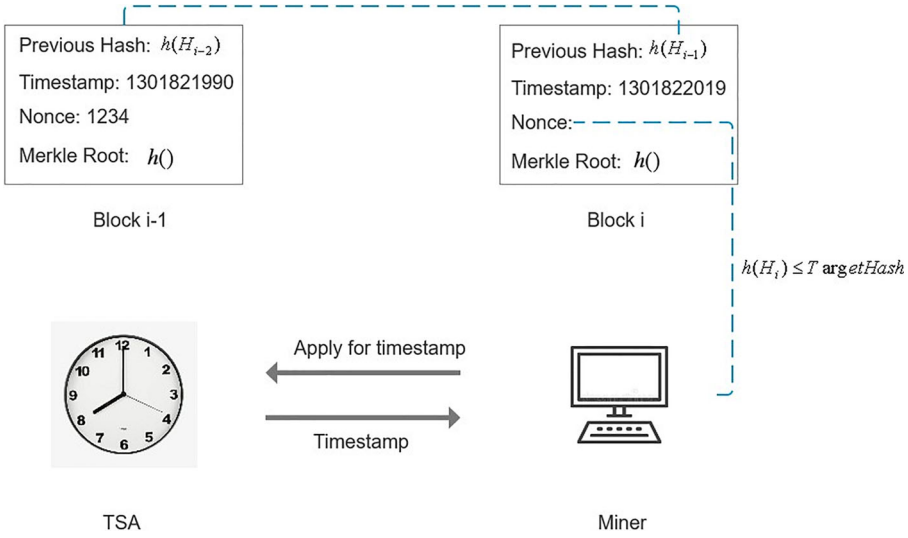

Fig. 2 System architecture

block to other nodes. Repeating this process, again and again, the timestamp of the node and the block will be more accurate than before. Selfish mining and time jacking will be more difficult for its attack window will be smaller. A high-level description is presented in Fig. 2.

3.2 Protocol details

In this subsection, we illustrate the details of our protocol.A high-level description is presented in Fig. 3 The protocol can be divided into the following steps:

1. When a node starts to mine a new block, the miner node frist determines the previous block to follow.
2. Then, the miner requests a timestamp from the TSA to get the accurate timestamp. Denotes the TSA's public key and private key pair as (pk, sk).
   - Node sends $\{height\}$ to apply for a accurate timestamp. Height is the number of the block, which presents the block's order.
   - The TSA returns $Sig(\{height \| T\}, sk)$, where T is the current timestamp and Sig is a signature algorithm.
   - When receiving a response from the TSA, the node verifies whether the signature is correct using TSA's public key pk. If the verification holds, using the timestamp T as the input of the mining puzzle.
3. The miner collects the transactions and calculates the Merkle tree root of all transactions.
4. The miner attempts to find a proper nonce, which satisfying

$$SHA256(Version, PreviousHash, MerkleRoot, T, Nonce) \leq TargetHash,$$

and broadcast the new block if such a nonce is found.
5. When receiving a new block from the network, a node checks wether the following conditions hold:
   - Whether the nonce satisfies the mining puzzle;

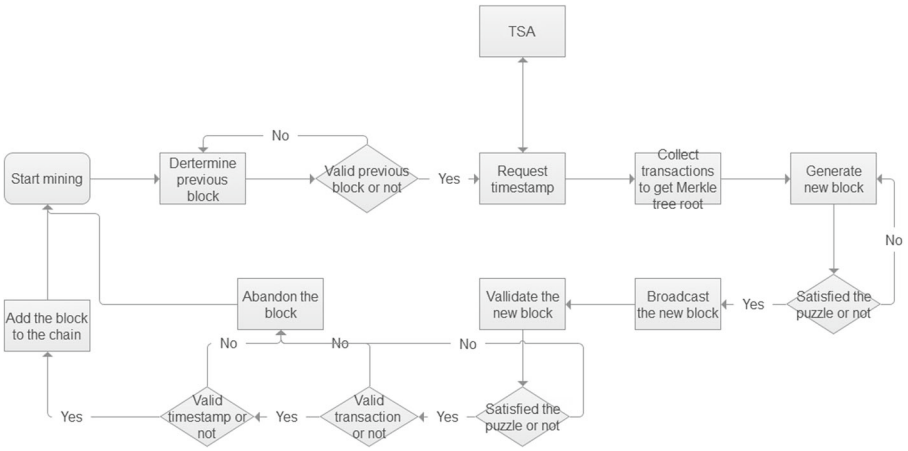

Fig. 3 Flow of protocol details

– Validate the timestamp and check whether the current block timestamp T is larger than the previous block's timestamp.
– Validate the transactions to check whether all the contained transactions are correct.

If one of the above conditions doesn't hold, abandons the block. Otherwise, adds the new block to the blockchain.

4 Security analysis

The time line of the events in this mechanism is presented in Fig. 4.

When starting to find the new block, the miner should determine which block to follow to dig the next block, then contact TSA immediately to get the accurate timestamp. Verifier nodes can easily validate the timestamp from TSA is the right timestamp which links to the new block by connecting to TSA. The mining period will be repeated with the calculation of the block's hash value, to find the proper nonce which satisfying the target hash difficulty. The timestamp from TSA is also an important input in the hash function, which adds trusted standard time to the bitcoin found reliable links.

When the new block is accepted by the whole network, it is very hard to change the timestamp in the block header. For the hash function's unidirectional property ensures the Proof of Work. Tampering the block is impossible without redoing the Proof of Work. Blocks link to the chain, the difficulty to change the historic blocks increases geometrically.The node changes the timestamp to the latter timestamp is impossible, for trusted TSA can not give out the unreached time.

The miner or attacker node can change the timestamp before the accurate timestamp. But the revised range is very small, for the Bitcoin has a generation speed adjustment mechanism that makes block's timestamp differs in about 10 minutes. The fake timestamp's range is the previous block's timestamp to the accurate timestamp. The time before the previous block will make the new block invalid.

The previous legal timestamp interval in the Bitcoin system is several hours. The reason is that the mining nodes in the previous protocol use their own independent system timestamp to do mining work. The legal system and the mining timestamp can be no more than 70 minutes. Every nodes' timestamp can be different, and the difference can be several hours. Although the block generation time can be controlled to about ten minutes by a system algorithm. But these ten minutes are relative ten minutes. The timestamps of the two adjacent blocks may be inaccurate for they are not based on the same timeline. The highest error between the two block's timestamps can reach several hours. In our newly designed protocol, the timestamp of each block is applied from the TSA by the miner node that mined the new block. The accuracy and reliability of the timestamp are objectively guaranteed. The timestamps of all the blocks on the blockchain are accurate and reliable timestamps from the same TSA which following the same timeline. With collaboration of the algorithm of the blockchain system itself, the average block generation time is about 10 minutes. Therefore, the accuracy of the timestamp in the block is improved to about 10 minutes.

According to the TSA's time, we can have an accurate time standard to regulate the mining timeline. Every block's timestamp can have a real-time based on the TSA. The timestamp accuracy is accurate to ten minutes from hours. Assume the new block's timestamp from TSA is 3:00, which means at 3:00, this node begins its work to mine. If some nodes try to modify this timestamp, it is very hard to modify it after the block being accepted by the whole network. The only way is to modify it before the acceptance of this block. If the new timestamp is earlier than 3:00, it also has to be after the previous block's timestamp. For the generation speed is ten minutes on average. The timestamp earlier than 2:50 is not impossible. If the malicious want to change the timestamp after 3:00, the TSA can not give an invalid latter timestamp. So the accuracy can be within the average generation time of the Bitcoin.

4.1 Defense against selfish mining

Selfish mining [9] originally comes from the strategy of the bitcoin how to deal with the fork problem [5]. When there are two blocks come out nearly the same time, because of

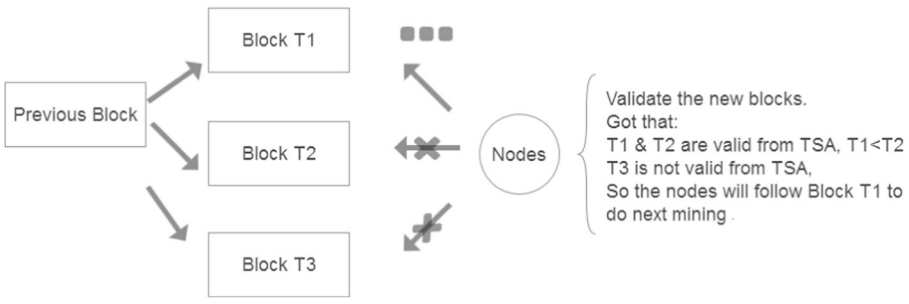

Fig. 4 The validation of different blocks

the difference of geographic location, the broadcast speed, and other reasons, they will divide the whole network into two parts. The two parts will choose one block (generally the first received) to continue finding new blocks. The chain becomes two chains to compete for survival. The longest chain will have priority to attract nodes to keep working. The failed block or shorter chain will be abandoned which causes a huge waste of network computing power and huge damage to the transactions in these blocks. Afterward, Ethereum [6] designs the Ghost protocol [22] to activate the abandoned blocks to protect the security of the main chain. They consider the heaviest chain is the main chain instead of the longest chain.

Selfish mining, as the name suggests, it is a miner or a mining pool mining secretly to accumulate secret blocks. The more blocks they control, the greater the harm they can do. As normal conditions, the new block should broadcast to the whole network but selfish miners keep it secretly to wait and use it to attack other miners. When other miners find a new block, selfish miners broadcast its secret block to keep forking competition conditions, which can greatly waste their competitors' computing power. Selfish mines can also broadcast all secret blocks in one time to destroy competitors' block immediately. Selfish mining strategy can use game theory to analysis.

Selfish mining is not for the purpose of disrupting the normal operation of the blockchain network, but more simply to obtain greater economic benefits. This difference is very important, which is the most important reason for that the Bitcoin is far more likely to encounter selfish mining attacks than other types of cyber attacks, such as the 51% attack often mentioned. Although 51% attacks are very harmful, there are very few real attackers. In addition to the difficulty of the attack, the main reason is that 51% attacks directly lead to the damage of the Bitcoin consensus mechanism, which indirectly leads to a large loss of Bitcoin. Selfish mining attack is different. It does not undermine the consensus mechanism of the Bitcoin network, it just hide secret blocks to gain the biggest profit. Moreover, its attack difficulty is far lower than 51% attacks, and experiments have proved that selfish mining attacks have the greatest threat when the computing power reaches 1/3.

Selfish mining need to keep secret blocks. In our mechanism, miner node is very hard to keep secret block, for it has to contact with TSA to apply for accurate timestamp as soon as possible, otherwise, the new block's timestamp will become invalid. Keeping a secret block will do harm to itself for a secret block is invalid. New block's timestamp is affected by TSA, TSA has a counter to verify the block height. When it comes to a fork problem, our mechanism has accurate timestamp from TSA, the blocks can tell the sequential order between the time nearby blocks. The timestamp former block has the priority to get the next block. TSA can do some audit work to verify the link of the former and the present blocks. The first block to get accurate timestamp is valid, which helps no-hiding secret blocks.

4.2 Defense against time-jacking

Time-jacking is an attack mode against timestamp's vulnerability. The attacker can change the target node's network time and deceive it into accepting an alternate blockchain by announcing inaccurate timestamp [4]. This could create a "poison pill" block which increases the possibility to launch a double-spending attack [15]. The goal of the "poison pill" attack is to destroy the availability of the target node, which will mislead the network to make incorrect actions [29]. If the target node is a miner, it will make the miner being isolated from the whole network, the miner's computing power will be wasted for a long time. The present time-jacking attack is very difficult to notice for its latency period can be very long. Once the attack is launched, it can do great harm. Last April, verge (XVG) was attacked by hackers using a time-jacking attack. This attack lets us see the great harm and make us find a solution to the time-jacking attack.

Time-jacking attack is mainly aimed at the vulnerability of timestamps in the Bitcoin system. There are no standard timestamp sources in Bitcoin for timestamps. The block's timestamps come from independent nodes, which will give the attackers opportunity to modify the timestamp of the target node. An attacker can pretend to be a multiplexed node that synchronizes version information or transaction information to communicate with the target node in a planned manner. During the communication process, malicious nodes will continuously broadcast a large number of unreliable timestamps until the target node considers these unreliable timestamps to be correct. The timestamp used by each node in the Bitcoin system for mining is the median of the timestamps from all surrounding nodes. Enough malicious timestamp information can change the system timestamp of the target node. The node will also have its own local timestamp. When the system timestamp is too large (usually 70 minutes), the node will notice that it is illegal. It will automatically restore the system timestamp to its own local timestamp. The legal time interval in the Bitcoin system is $[T_0, T_1 + h]$. T_0 is the median timestamp of the eleven parent blocks before this block, T_1 is the system time of the miner node which mined this block, and h represents one hour. Based on the above mechanism, a malicious node can modify the timestamp of the target node to a certain extent, usually within a few hours, without causing system confusion. After the target node's timestamp for mining has been modified, the first thing being affected is the reliability and integrity of the information in the block. If the target node is a miner node, the mining efficiency of

the miner node will be greatly reduced. During the process of time-jacking attack, it is almost impossible to mine. The unreliable timestamp will cause the miner node to become isolated, and the correct transaction information will not be obtained, which will cause a huge waste of the node's computing power. A malicious node or mining pool will use time-jacking attack to attack competitors in order to obtain greater benefits in the Bitcoin system.

The time-jacking attacker can change nodes' network time by announcing an inaccurate timestamp when connecting to the nodes. The attacker can slow down the target's time and speed up other nodes' time to isolate the target. The present acceptable time range is still too long so that the system can not notice the attack immediately, the attacker's attack window will be at least 140 minutes [4]. In our mechanism, all nodes' timestamp synchronizes with TSA frequently. Even if the attacker changes the target timestamp, the target will revert to a normal timestamp based on TSA when every new block comes out. Timestamp differences between all nodes become smaller can tighten the acceptable time ranges, which can lessen the attack window. It will be more difficult to launch a time-jacking attack.

4.3 Defense against double-spending

If the target node is attacked by time-jacking, it wouldn't receive any more valid transaction confirmations during the attack period. The attackers can feed confirmations to the target without the honest miners intervening, for they know the fake confirmations will be corrected later by the whole blockchain. The fake confirmations can cause double-spending of some transactions.

Our protocol tightens the time range between blocks, which helps the nodes connect inseparably. It's harder to change some nodes' timestamp to isolate them from the whole network. Because the block's timestamp is based on trusted TSA timestamp, the node's time can not intervene the block's timestamp. The resulting double-spending attack will be harder too.

5 Conclusions

In this paper, we presented a mechanism to achieve reliable timestamp for Bitcoin. In our design, we leveraged a trusted TSA to offer a reliable timestamp which is added as one of the inputs of the mining hash puzzle. By this approach, the nonce of the hash puzzle will useless which will cause a lost of Bitcoin cryptocurrency loss of the malicious miner that changes the timestamp. Thus, in our proposed protocol, the range of a valid timestamp is narrowed into ten minutes from hours. Benefit from this, the Bitcoin platform can provide timestamp service to time-sensitive applications. Moreover, our protocol prevents the Bitcoin platform from selfish mining and time-jacking attack.

Acknowledgments This work was supported by the National Natural Science Foundation of China (Grant No.61802180, 61702236, 61872181), the Natural Science Foundation of Jiangsu Province (Grant No.BK20180421), the National Cryptography Development Fund (Grant No.MMJJ20180105), the Fundamental Research Funds for the Central Universities (Grant No.NE2018106), the State Key Laboratory Foundation of smart grid protection and operation control, the Science and Technology Funds from National State Grid Ltd. (The Research on Key Technologies of Distributed Parallel Database Storage and Processing based on Big Data).

References

1. Adams C, Cain P, Pinkas D, Zuccherato R (2001) Rfc 3161: internet x. 509 public key infrastructure timestamp protocol (tsp). Internet Engineering Task Force
2. Apostolaki M, Zohar A, Vanbever L (2017) Hijacking bitcoin: routing attacks on cryptocurrencies. In: 2017 IEEE symposium on security and privacy (SP). IEEE, pp 375–392
3. Ateniese G, Goodrich MT, Lekakis V, Papamanthou C, Paraskevas E, Tamassia R (2017) Accountable storage. In: International conference on applied cryptography and network security, Springer, pp 623–644
4. Boverman A (2011) Timejacking & bitcoin
5. Bradbury D (2013) The problem with bitcoin. Computer Fraud & Security 2013(11):5–8
6. Buterin V et al (2013) Ethereum white paper. GitHub repository, pp 22–23
7. Camerinelli E (2009) Supply chain finance. Journal of Payments Strategy & Systems 3(2):114–128
8. Chunpeng G, Liu Z, Xia J, Liming F (2019) Revocable identity-based broadcast proxy re-encryption for data sharing in clouds. IEEE Transactions on Dependable and Secure Computing
9. Eyal I, Sirer EG (2018) Majority is not enough: bitcoin mining is vulnerable. Communications of the ACM 61(7):95–102
10. Fang L, Susilo W, Ren Y, Ge C, Wang J (2010) Chosen public key and ciphertext secure proxy re-encryption schemes
11. Gao Y, Nobuhara H (2017) A decentralized trusted timestamping based on blockchains. IEEJ Journal of Industry Applications 6(4):252–257
12. Gipp B, Meuschke N, Gernandt A (2015) Decentralized trusted timestamping using the crypto currency bitcoin. arXiv:1502.04015
13. Hao M, Li H, Luo X, Xu G, Yang H, Liu S (accepted 2019, to appear) Efficient and privacy-enhanced federated learning for industrial artificial intelligence. IEEE Transactions on Industrial Informatics, pp 1–1. https://doi.org/10.1109/TII.2019.2945367
14. Jiang W, Li H, Xu G, Wen M, Dong G, Lin X (2019) Ptas: privacy-preserving thin-client authentication scheme in blockchain-based pki. Future Generation Computer Systems 96:185–195
15. Karame G, Androulaki E, Capkun S (2012) Two bitcoins at the price of one? double-spending attacks on fast payments in bitcoin. IACR Cryptology ePrint Archive 2012(248)
16. Li H, Liu D, Dai Y, Luan TH, Yu S (2018) Personalized search over encrypted data with efficient and secure updates in mobile clouds. IEEE Transactions on Emerging Topics in Computing 6(1):97–109
17. Li H, Yang Y, Dai Y, Yu S, Xiang Y (accepted 2017, to appear) Achieving secure and efficient dynamic searchable symmetric encryption over medical cloud data IEEE Transactions on Cloud Computing, pp 1–1. https://doi.org/10.1109/TCC.2017.2769645

18. Ma D, Tsudik G (2009) A new approach to secure logging. ACM Transactions on Storage (TOS) 5(1):2
19. Merkle RC (1989) A certified digital signature. In: Conference on the theory and application of cryptology. Springer, pp 218–238
20. Nakamoto S et al (2008) Bitcoin: a peer-to-peer electronic cash system
21. Ren H, Li H, Dai Y, Yang K, Lin X (2018) Querying in internet of things with privacy preserving: challenges, solutions and opportunities. IEEE Network 32(6):144–151
22. Sompolinsky Y, Zohar A (2015) Secure high-rate transaction processing in bitcoin. In: International conference on financial cryptography and data security. Springer, pp 507–527
23. Syta E, Tamas I, Visher D, Wolinsky DI, Jovanovic P, Gasser L, Gailly N, Khoffi I, Ford B (2016) Keeping authorities honest or bust with decentralized witness cosigning. In: 2016 IEEE symposium on security and privacy (SP). IEEE, pp 526–545
24. Szalachowski P (2018) Towards more reliable bitcoin timestamps. arXiv:1803.09028
25. Veizades J, Guttman E, Perkins C, Kaplan S (1997) Service location protocol. Tech rep
26. Xu G, Li H, Dai Y, Yang K, Lin X (2019) Enabling efficient and geometric range query with access control over encrypted spatial data. IEEE Trans Inform Forensics Secur 14(4):870–885
27. Xu G, Li H, Liu S, Wen M, Lu R (2019) Efficient and privacy-preserving truth discovery in mobile crowd sensing systems. IEEE Trans Vehicular Technol 68(4):3854–3865
28. Xu G, Li H, Liu S, Yang K, Lin X (2020) Verifynet: secure and verifiable federated learning. IEEE Trans Inform Forensics Secur 15(1):911–926
29. Xu G, Li H, Ren H, Yang K, Deng RH (2019) Data security issues in deep learning: attacks, countermeasures and opportunities. IEEE Communications Magazine 57(11):116–122. https://doi.org/10.1109/MCOM.001.1900091
30. Zhang R, Preneel B (2017) Publish or perish: a backward-compatible defense against selfish mining in bitcoin. In: Cryptographers' track at the RSA conference. Springer, pp 277–292