A Novel Algorithm to Withstand Attacks on Blockchain Using Transaction History and Block Publishing Time

Anjaneyulu Endurthi, Aparna Pyarapu, Gayathri Jagiri, and SaiSuma Vennam

Abstract Blockchain is an emerging technology with many kinds of cryptocurrencies like bitcoin, Ethereum etc. where one can earn rewards by creating new blocks through the process of mining. But how about only few getting all these rewards and trying to get control over the network? Does it pose a threat to blockchain integrity? Of course, yes which is leading to attacks such as 51% attack, selfish mining attack and double spending. Proof of work (PoW) is a protocol used in blockchain to reduce these problems, but it is not sufficiently secure. So, this paper proposes a technique, using history of miners and the history of their transactions that helps us to reduce the chances of these attacks. We have chosen history of transactions in order to find the genuineness of a miner, thus helping us to reduce double spending. Analysis shows that the risk of these attacks can be decreased using this technique.

Keywords Blockchain · Sybil attack · Selfish mining attack · Double spending · Proof of work · Cryptocurrency

1 Introduction

Today's generation is overflowing with technologies. Still, the craze over new technology is rising day by day. Not only there is growth in technology, but also there are a lot of privacy and security vulnerabilities. Presently, one of the new technologies is Blockchain [1]. Blockchain technology records information in the transparent digital ledger such that the information is immutable and cannot be compromised. Blocks in the blockchain are records of transactions joined by cryptographic hashes. Every block in the blockchain has a recent block hash value, timestamp and transactions data in it (Fig. 1).

Blockchain has a distributed ledger [2], i.e. data is used in a decentralized [3] way and can be managed by multiple participants (miners). A transaction is valid in blockchain only if more than 51% of participants in distributed network of blockchain approves it. Transactions are added to the block after mining and validation.

A. Endurthi (✉) · A. Pyarapu · G. Jagiri · S. Vennam
Department of Computer Science and Engineering, Rajiv Gandhi University of Knowledge Technologies—IIIT, Basar, Telangana, India

S. Smys et al. (eds.), *Inventive Computation and Information Technologies*, Lecture Notes in Networks and Systems 336, https://doi.org/10.1007/978-981-16-6723-7_51

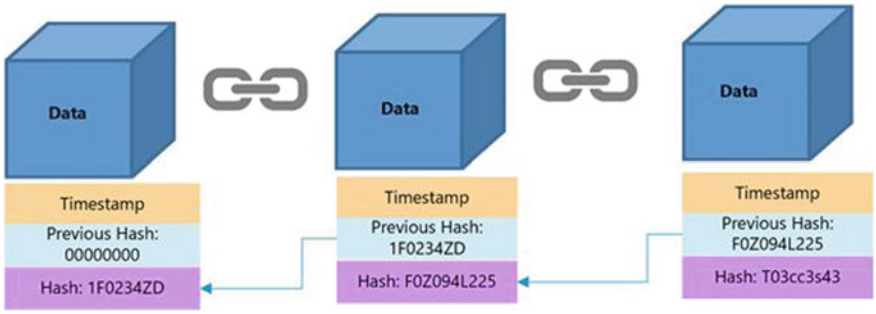

Fig. 1 Blockchain representation

1.1 Attacks on Blockchain

Even though blockchain is immutable, there are many attacks present as of now [4]. Blockchain network is secure and scalable, but if someone collectively in the network controls over 51% of the network, then there are chances of attacks on blockchain. The following sub-sections gives an idea of such attacks.

Double Spending It occurs when the attacker (Bob) would make some transaction between two other people (Lisa and Alice) and claim that he has sent the actual currency to both the people. But actually, he has sent digital currency to one person (Lisa) and a copy to other (Alice) or vice versa (Fig. 2).

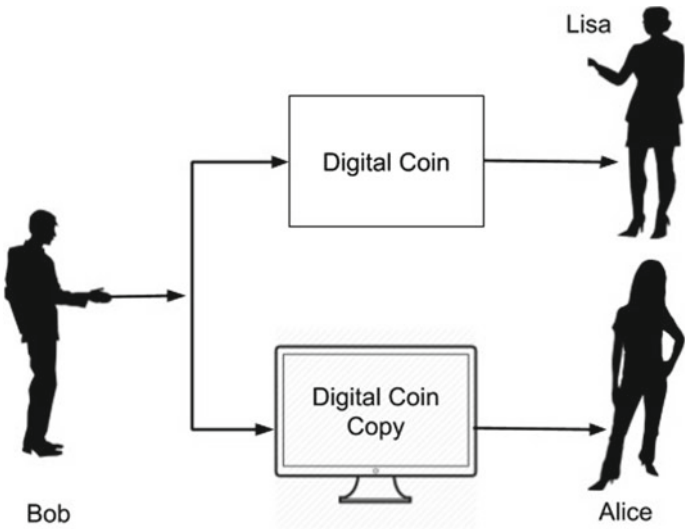

Fig. 2 Diagram displaying double spending

Fig. 3 Diagram displaying how 51% of the network attacking blockchain

51% Attack It is an attack on a blockchain by a miner or a group of miners who control more than 51% of the network's mining power [5]. Thus, misusing the power by modifying the transactions of new blocks or including double spending transactions in the upcoming blocks. In the following figure, the red-coloured nodes are collectively working and has more than 51% hashing power (Fig. 3).

Sybil attack It is a kind of security threat where one person (attacker) tries to take over the network by creating multiple accounts, nodes and pretend as different people. Thus, trying to outvote the honest miners and rejecting the blocks those are created by honest miners (Fig. 4).

Selfish mining attack The selfish mining attack [6] or block withholding attack occurs when a miner decides to keep a valid block, that they have successfully mined, instead of broadcasting it on to the network, he/she mines the next blocks making his branch as a long chain and claim rewards by outrunning the existing valid blockchain [7] (Fig. 5).

In this paper, literature is discussed in the next section, i.e. Sect. 2 and proposed algorithm along with its advantages is discussed in Sect. 3. Section 4 consists of conclusions and future directions. References are provided at the end of the paper.

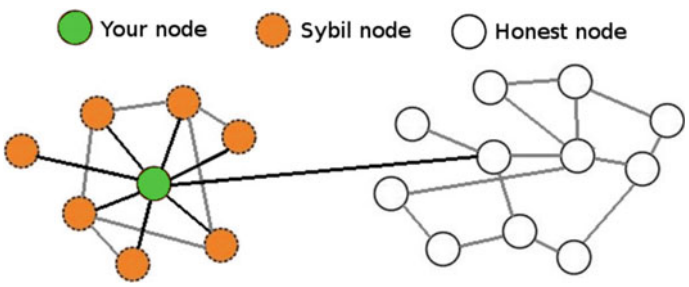

Fig. 4 Diagram showing Sybil attack

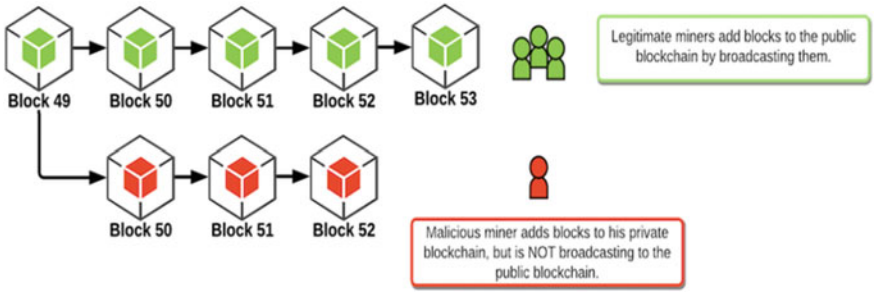

Fig. 5 Diagram of selfish mining attack

2 Literature Review

As there are many attacks that can occur on blockchain, there are many solutions proposed in the literature. Most of the solutions are specific to a particular kind of attack. To reduce block withholding or selfish mining attack, Siamak Solat, Maria Potop-Butucaru proposed ZeroBlock mechanism. In this, they added a timestamp. If miner creates a block, then he/she has to publish within the specified duration given by the network. If miners are unable to publish, then a dummy zero block is created which will be added to the blockchain [8]. This solves the selfish mining attack, but a greater number of unnecessary zero blocks will be added to the blockchain.

In order to overcome the drawback of ZeroBlock mechanism, a paper entitled Selfish Mining [9] was proposed. In this algorithm, a block has to be generated and propagated within maximum acceptance time (MAT). If no block is created within this MAT interval, then new target hash will be generated. All the blocks which were created based on old hash will not be accepted after the generation of new target hash.

To differentiate honest and dishonest branches, there is an algorithm called history weighted difficulty [10]. In this, they have calculated the frequency of each miner. If there is a split in the branches, history weighted difficulty of the branches is calculated based on frequency of miners and difficulty of the branch. Branch with higher history weighted difficulty value is selected.

Penalizing equivocation by loss of Bitcoins is proposed in [11]. In this paper, they proposed a technique to mitigate the behaviour of a double spending attacker. They are penalizing the one who is trying to double spend. This can be done in two ways; one way is to deposit some funds, and the other way is to extract the secret key of the one who has done double spending and use it to force the loss of attacker's funds.

The concept of transaction creation time was given as a solution for one of the attacks in [12]. In this paper, they have proposed fork protocol to differentiate honest and dishonest miners. They have changed the block structure by adding one field to the block that is transaction created time. Using this, we can calculate the average transaction time. The miners with less average transaction time were called as honest miners.

The paper titled Majority is not enough Bitcoin mining is vulnerable [13]. In this, they have proved that the existing bitcoin protocol is not incentive-compatible. It means that the selfish miners are getting more rewards than their computational power. So, they have modified the bitcoin protocol to encourage the honest miners. By this protocol, network will consist of 2/3 honest miners.

A backward compatible defence against selfish mining in bitcoin is discussed in [14]. In this, they have introduced fork-resolving policy weighted FRP and an upper bound 'T' on block propagation time which is known as In-time. A block A1 is considered to be the uncle of another block A2 if A1 is a competing in time block of A2's parent block. Weight of the chain is calculated based on in-time blocks and in-time uncle hashes. So, when there are competing chain of blocks, weighted FRP is calculated; if one chain is less than other chain by 'k' blocks, the chain with highest weight is chosen by the miner.

Another paper named Disincentivize large bitcoin mining pool, which they have proposed two-phase proof of work [15] to control the formation of large mining pools. Process involves two steps: (1) Double hash of the header. (2) Hash over header which is signed with private key of the miner, i.e. SHA256 (SIG header, private key).

Another paper entitled Weird trick to stop selfish mining [16] proposed a technique to stop selfish mining. In this, for every 's' seconds, it publishes a new random value 'R'. 'R' is unpredictable before the publication, and they both form a tuple (R, T) and verified by anyone. Where, 'R' is the randomness and 'T' is the timestamp. If any block whose 'R' is before the timestamp, 'T' is considered as honest block and if it is after the timestamp or if beacon is compromised by an attacker, then it is considered as dishonest block.

Each of these solutions in the literature gives solutions to some specific attacks could not be able to withstand other attacks. This motivated us to propose an algorithm, which can withstand against more than one kind of attack. The proposed algorithm is discussed in the next section.

3 Proposed Algorithm

This section describes about minimizing attacks on blockchain using history of transactions and selfish mining strategy. The main motto of this algorithm is to detect a selfish miner and to differentiate honest and dishonest branches based on miner's frequency in the history and history of their transactions.

Selfish miner can be identified by comparing the time he/she is trying to publish the block and his/her block creation time. This paper concentrates on setting a time limit to publish the newly mined block. If the miner has published his block within this time limit, his/her block will be accepted, else it will be rejected.

Double spending can be reduced using this algorithm as we are disposing a constraint on the transactions of miner's and trying to calculate how malicious a miner is based upon his/her valid and invalid transactions. So, if the miners tend to make invalid transactions, they are treated as malicious and thus their mining branch

would be discarded. As we are discarding the malicious branch, attacker cannot have control over the network and thus withstanding against 51% attack.

3.1 Algorithm

The following algorithm is divided into three steps. The second step is executed only when the first step is cleared and step 3 is executed only when it clears step 2.

Step 1—Check if the miner is selfish

Time limit for newly mined block to publish = "T".

```
If(there is branch split)
        Calculate HWD and HTV for all the nodes present in the
        branches(Where, HWT – History Weighted difficulty and HVT
        - History Transaction Value)

        HW = 0
        HTV=0
        d = 0
        w = Length (W)   (Where W - array of historic blocks window)
         l = Length (B)                (Where B - array of branch blocks)
         Let R[1 ...l] be new arrays
        Let Tr[1...l]=0 -> be transaction history of an individual miner
        in the branch
        for i = 1 to l do=>Calculate miner appearance frequency in
        historic blocks window  and calculate transaction history of
        miner
                R[i]=0
                Tr[i]=Number of invalid transactions of a miner/Total
                number of transactions of a miner
                for j = 1 to w do
                if( Miner(W[j]) == Miner(B[i]) then
                        R[i]+ = 1
                R[i]/ = w
        for k = 1 to Length(B) do=>Sum Historic Weight
                HW = HW + R[k]
        for k = 1 to Length(B) do =>Sum branch difficulty
                d = d + Diff(B[[k])
        for k = 1 to Length(B) do =>sum transaction history
                HTV=HTV+Tr[k]
        HWD = HW ∗ d
```

Step 2—If newly mined block is accepted

If(there is branch split)

Calculate HWD and HTV for all the nodes present in the branches(Where, HWT – History Weighted difficulty and HVT - History Transaction Value)

HW = 0
HTV=0
d = 0
w = Length (W) (Where W - array of historic blocks window)
l = Length (B) (Where B - array of branch blocks)
Let R[1 ...l] be new arrays
Let Tr[1...l]=0 -> be transaction history of an individual miner in the branch
for i = 1 to l **do=>**Calculate miner appearance frequency in historic blocks window and calculate transaction history of miner
R[i]=0
Tr[i]=Number of invalid transactions of a miner/Total number of transactions of a miner
for j = 1 to **w** do
if(Miner(W[j]) == Miner(B[i]) then
R[i]+ = 1
R[i]/ = w
for k = 1 to Length(B) **do=>**Sum Historic Weight
HW = HW + R[k]
for k = 1 to Length(B) **do =>**Sum branch difficulty
d = d + Diff(B[[k])
for k = 1 to Length(B) **do =>**sum transaction history
HTV=HTV+Tr[k]
HWD = HW ∗ d

Step 3—Compare the HWD and HTV values of the branches

```
If(HTV is high and HWD is high)
        Do not accept this branch
else if(HTV is high and HWD is low)
        Do not accept this branch
else if(HTV is low and HWD is high)
        Continue mining from this branch
else if(HTV is low and HWD is high)
        Continue mining from this branch
```

3.2 Flow Chart Representation:

Following is the flow chart representation of the algorithm (Fig. 6).

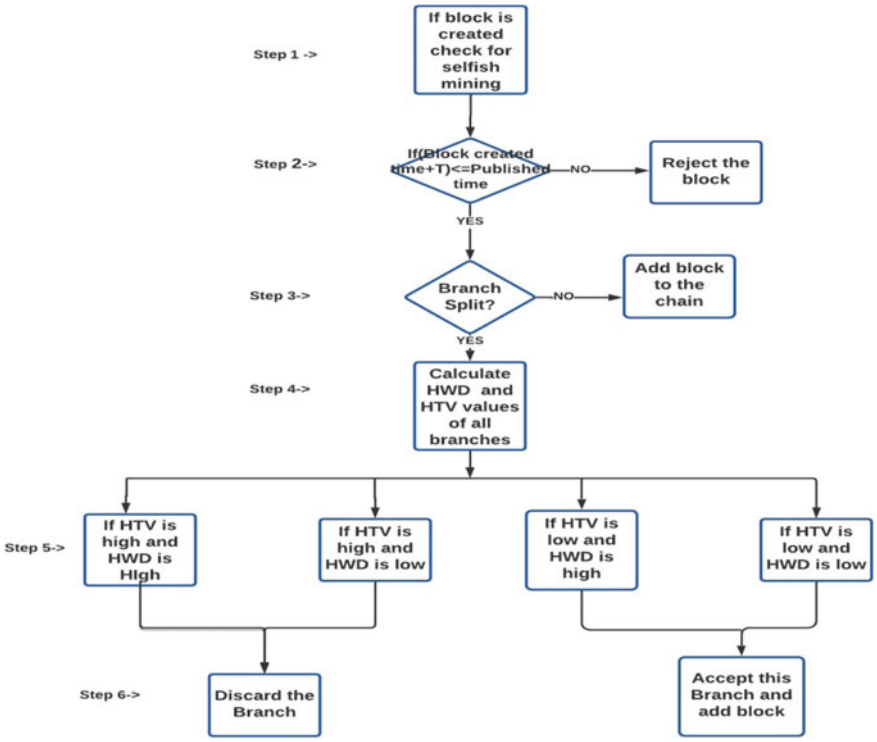

Fig. 6 Flow chart representation of algorithm

If any block is mined recently, it should be checked whether it is selfishly mined or not. And then honest and malicious branches should be differentiated, if there's more than one branch. History weighted difficulty (HWD), transaction history value (HTV) should be calculated for all the branches. Based on those values, branch should be either accepted or discarded, and newly mined block will be added to honest blockchain branch.

3.3 Advantages of Proposed Algorithm

All the existing algorithms of selfish mining have efficacy to some extent and possesses drawbacks. But in the proposed algorithm, we can reduce selfish mining to greater extent as we are making miner to publish block soon after its creation and also restricting the miners based on their transaction history. Moreover, we can successfully differentiate honest and dishonest branches which can withstand against attacks like 51% attack, selfish mining attack and also double spending attack.

4 Conclusion and Future Directions

This paper proposes an approach to reduce the chance of 51% attack and selfish mining attack on proof of work-based blockchain protocols. This approach makes use of frequency of miners and history of their transactions and determines if a branch switch is needed or not. It also detects selfish miner based upon their block publishing time. Thus, making it withstand against dangerous attacks on blockchain. In future, the same solution could be extended to withstand other attacks.

References

1. S. Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System* (2008)
2. R. Wattenhofer, *Distributed Ledger Technology—The Science of Blockchain* (Forest Publishing, 2017)
3. M. Atzori, *Blockchain Technology and Decentralized Governance: Is the State Still Necessary* (2015)
4. M. Saad, J. Spaulding, L. Njilla, C. Kamhoua, S. Shetty, D. Nyang, et al., Exploring the Attack Surface of Blockchain: A Comprehensive Survey. IEEE Commun. Surveys Tuts. **22**(3), 1977–2008 (2020)
5. D. Mories, *51% Attack Are Growing a Threat to Smaller Blockchain; Komodo May Be the Solution* (2019)
6. S. Zhang, K. Zhang, *Bettina Kemme Analysing the Benefit of Selfish Mining with Multiple Players* (2020)
7. V. Chicarino, E.F. Jesus, C. Albuquerque, A. Rocha, A heuristic for the detection of selfish miner and stalker attacks in blockchains networks, in *IEEE Blockchain, Robotics and AI for Networking Security Conference, 2019, Rio de Janeiro.* BRAINS, IEEE, pp 1–6 (2019)

8. S. Solat, M. Potop-Butucaru, *Zeroblock: Preventing Selfish Mining in Bitcoin*. arXiv preprint arXiv:1605.02435 (2016)
9. A. Endurthi, B. Dastagiri, S. Janipasha, D. Santhosh, A. Khare, *Transformed Puzzle for Preventing Selfish Mining: A Non-Viable Way to Defend Zero Block Algorithm* (2020)
10. X. Yang, Y. Chen, X. Chen, *Effective Scheme Against 51% Attack on Proof-of-Work Blockchain with History Weighted Information* (2019)
11. T. Ruffing et al., Liar, liar, coins on fire!: Penalizing equivocation by loss of Bitcoins, in *ACM Conference on Computer and Communications Security* (2015)
12. J. Lee, Y. Kim, *Preventing Bitcoin Selfish Mining Using Transaction Creation Time* (2018)
13. I. Eyal, E.G. Sirer, Majority is not enough: Bitcoin mining is vulnerable, in *International conference on financial cryptography and data security* (Springer, Berlin, Heidelberg, 2014)
14. R. Zhang, B. Preneel, Publish or perish: a backward-compatible defense against selfish mining in bitcoin, in *Cryptographers' Track at the RSA Conference* (Springer, Cham, 2017)
15. I. Eyal, E.G. Sirer, *How to Disincentivize Large Bitcoin Mining Pools* (2014)
16. E. Heilman, One Weird Trick to Stop Selfish Miners: Fresh Bitcoins, a Solution for the Honest Miner, in *International Conference on Financial Cryptography and Data Security* (Springer, Berlin, Heidelberg, 2014)