Preventing Spread of Spam Transactions in Blockchain by Reputation

Jiarui Zhang*, Yukun Cheng†, Xiaotie Deng‡, Bo Wang§, Jan Xie§, Yuanyuan Yang*, Mengqian Zhang¶
*Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY 11794, USA
†School of Business, Suzhou Key Laboratory for Big Data and Information Service,
Suzhou University of Science and Technology, Suzhou, 215009, China
‡Centers on Frontiers of Computing Studies, Department of Computer Science and Technology,
Peking University, Beijing, 100871, China
§Hangzhou Cryptape Technology Co., Ltd., Hangzhou, 310000, China
¶Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, 200240, China

***Abstract*—As one of the fastest-growing applications in the Peer-to-Peer (P2P) network, the development of blockchain technology is accompanied by different attacks. Those include white-washing, free-riding, and distributed denial of service (DDoS) attacks, particularly because of features such as anonymity, distributed, permissionless in the blockchain network. One popular of them is spam transactions. Although the blockchain protocol requires each node to verify all received transactions, many nodes choose to forward transactions without verification to conserve their computational power, as there is no punishment for such a shirking. And it makes the blockchain vulnerable to the spreading of spam transactions over the network and creates extra burdens for all nodes in the network. We propose a reputation mechanism for the blockchain system to tackle this problem: Each node will locally compute reputations of its neighbors, and decide the probability to verify a received transaction based on the reputation value of the transaction sender. In turn, its neighbors will have an incentive to conduct verification to keep its reputation high. Subsequently, spam transactions can be blocked before reaching the miners. We have conducted a series of simulations, which clearly demonstrate the advantage of our reputation mechanism.**

I. Introduction

The blockchain technology has since emerged as an impressive and transformational ledger to safely store the history of transactions and the accumulated credits for participants. During this time period, Bitcoin [1] has become the most successful cryptocurrency, followed by a large number of cryptocurrencies such as Ethereum [2] [3] and EOS [4]. Together they have made the cryptocurrencies one of the most visible successful examples in Peer-to-Peer (P2P) systems.

Accompanied by the rapid development of blockchain technology, several attacks [5] [6] [7] are also observed in the system. In this paper, we focus on the spam transactions attack, which means the attackers attempt to propagate a lot of spam transactions to the system. Although those transactions won't compete for inclusion in blocks, they can delay the transmissions of valid transactions, and waste communication resources of the network. For efficient executions of the blockchain system, it is necessary to prevent the spread of spam transactions. As a solution, Bitcoin requires nodes to verify received data, and only forward valid transactions. However, this protocol has two disadvantages. The first is that nodes can still transfer unverified transactions as free-riders to save the computational resources without any punishment. The second is that repetitive verifications of all transactions waste computational resources and decrease the efficiency of the network. Thus, we need to design a mechanism to prevent spam transactions and punish nodes who transfer them, while making nodes to verify only part of received transactions.

In this paper, we design a reputation mechanism to prevent the spread of spam transactions. We first describe the spam transactions attack. Accordingly, we specify different types of transactions and nodes in the network. Based on the nodes' behaviors, a reputation mechanism is proposed, which mainly brings two following benefits. A node can decide the probability to verify a transaction according to the sender's reputation value. Moreover, nodes can identify spam transaction generators with the help of the mechanism. Extensive simulations further support the effectiveness of our method. The main contributions in this paper are in the following.

- We design a reputation mechanism, by which each node can compute the reputation values of its neighbors. Intuitively, nodes who forward more valid and fewer spam transactions would have a higher reputation value.
- We propose a strategy that allows reducing the spread of spam transactions by verifying part of transactions. Simply speaking, nodes tend to verify a transaction with a higher probability that is sent by a neighbor with a lower reputation.
- We also conduct extensive simulations for different environments to demonstrate the performance of our reputation mechanism. The results show that our proposed reputation mechanism can efficiently reduce the spread of the spam transactions, and most of the malicious nodes who generate the spam transactions can be screened out.

In Section II we discuss some related works on blockchain technology and reputation mechanism applications. In Section III we introduce the problem discussed in this paper and

Corresponding author: Xiaotie Deng, Email: xiaotie@pku.edu.cn

978-1-7281-6887-6/20/$31.00

formulate the overall environment. The reputation mechanism to prevent the spread of spam transactions is proposed in Section IV and the features it owns are also discussed. In Section V, we conduct a series of simulations to demonstrate our mechanism. The last section concludes this paper.

II. Related Work

Bitcoin was invented in 2008 by Satoshi Nakamoto [1] and has been one of the most popular P2P applications. The dependent technology, blockchain, records transactions that are generated by P2P network participants as a distributed ledger. Its properties such as decentralization, openness, and immutability decide that blockchain is the most suitable carrier of cryptocurrencies. Encouraged by the huge success of Bitcoin, other cryptocurrencies like Ethers in Ethereum [2] [3] and EOS [4] are developed to attract users and investments. Those cryptocurrencies come up with some new features. Ethereum introduces smart contract deployment to extract the capability of blockchain, while EOS attempts to provide a decentralized application development environment that offers high transaction throughput.

Like two sides of the same coin, blockchain properties also incur attacks. Denial of service (DoS) and distributed denial of service (DDoS) attackers propagate a large number of spam transactions, which can be mainly divided into two types. One is to propagate massive data to waste resources. Luu et al. [8] study this attack in the blockchain system with smart contracts. It shows that massive invalid transactions either waste computational resources of miners or lead miners to accept blocks with incorrect script results. Pontiveros et al. [9] further develop a strategy that purposes sluggish contracts which have a slow execution time in the virtual machine. As a result, attackers gain an advantage over other competitors. The second kind of DoS or DDoS attack is to propagate transactions with rather high transaction fees. Baqer et al. [5] present an empirical study of this attack on Bitcoin. The study shows that it increases the transaction fee of non-spam transactions and delays transaction processing. This type of attack is also analyzed in the Ethereum network [7], which has similar results and proves the increase of gas price. Saad et al. [6] focus on the impacts on memory pools of cryptocurrency systems and formulate the attack model that causes massive transaction backlog and higher mining fees.

As a possible solution to the above attacks, the reputation mechanism is for nodes to evaluate the trust level of other participants in the network. One of the earliest centralized Internet reputation mechanism is eBay, and Resnick et al. [10] point out the pros and cons of it. However, most P2P networks do not have centralized parts, so the decentralized reputation frameworks are proposed. Aberer et al. [11] propose a reputation-based trust management approach. This approach uses decentralized storage to provide data and runs data mining using statistical data analysis of historical transactions. Buchegger et al. [12] design an approach that includes two layers to describe the behaviors of nodes. As the first layer, nodes build reputation ratings by observing the behaviors of other nodes. Nodes can exchange first-hand information and maintain their trust ratings, which is the second layer. Srivatsa et al. [13] propose TrustGuard which counters vulnerabilities in reputation management. The TrustGuard framework has mainly several components, including strategic oscillation guard, fake transaction detection, and dishonest feedback filter.

The advent of the blockchain brings new ideas and challenges to the design of reputation systems. Multiple works that apply the reputation mechanism to blockchain are studied based on various application scenarios. Liu et al. [14] focus on reputation management in the consumer-retailer channel and propose an anonymous reputation mechanism based on blockchain. Khaqqi et al. [15] propose a novel Emission Trading Scheme (ETS) model customized for Industry 4.0 Integration. They combine blockchain technology and reputation mechanism to improve ETS efficacy and address fraud issues. Chai et al. [16] apply a blockchain-based resource sharing paradigm on the Internet of Vehicles (IoV). The reputation value indicates the reliability of vehicles, and it also reduces resource consumption of consensus mechanism by utilizing Proof-of-Reputation. Limited by the specific requirements and features of different blockchain applications, we cannot directly use their approaches.

III. Problem Formulation

In this section, we give environmental assumptions to formulate our problem. Then we state the problem definition and our design goals.

A. Transaction Verification Cost

We illustrate a way to measure transaction verification costs. A transaction includes essential parts, such as transaction input, output, transaction fees, and so on. All of these parts need to be verified. For instance, nodes shall verify the authority and the format of components. To ensure the validity of the transactions, nodes need to run different verification operations. These operations can be indicated by a series of instructions. Inspired by the fact that different CPU instructions have different predetermined running cycle costs [17], the instructions that may occur during the verification process also have different costs.

By indicating the cycle cost of each instruction in advance, the computational resources consumed during transaction verifications are defined as the total number of **cycles**. Based on these rules, all nodes can compute the total cycles of the verification for each transaction unambiguously. When proposing a new transaction, the node is required to attach the corresponding cycle cost an essential part of the transaction.

B. Types of Transactions

The transactions in the common blockchain environment are simply divided into two types, valid and invalid transactions. In our environment, each transaction records not only the regular transaction components but also a cycle cost to measure the consumption of the verification, therefore the correctness of cycle cost is another factor used for classification. We

define three types of transactions based on the validity and correctness of cycle cost.

- Valid and correct cycle transaction (**VC transaction**): A VC transaction is valid and attaches correct cycle cost. It can be verified by running instructions that cost the attached cycle cost.
- Valid and incorrect cycle transaction (**VI transaction**): A VI transaction is valid and attaches incorrect cycle cost. It can be verified by running instructions, but the attached cycle cost does not match the real cycle cost. Honest nodes will revise the cycle cost after verifying this transaction.
- Invalid transaction: An invalid transaction cannot be verified. It should be intercepted and cannot be recorded in the blockchain.

In our following discussion, we focus on these three types of transactions. And the invalid transaction defined here is the spam transaction we discuss in this paper.

C. Types of Nodes

According to the nodes' possible behaviors, we divide them into three types and give the following definitions.

- Honest node: An honest node verifies transactions by verification rules. It forwards verified VC transactions and verified VI transactions after revising the cycle cost. It also forwards transactions that are decided not to verify. It generates VC transactions.
- Lazy node: A lazy node does not verify transactions. It forwards all transactions without verifications. It generates VC transactions and VI transactions.
- Malicious node: A malicious node forwards all received transactions without any verifications. It generates VC transactions, VI transactions, and invalid transactions.

Honest nodes follow the verification rules and generate VI transactions. Lazy nodes and malicious nodes do not follow the verification rules. Lazy nodes and malicious nodes are also called dishonest nodes. The network system is supposed to protect the rights of honest nodes and punish dishonest nodes, thereby encouraging nodes to follow the rules.

D. Problem Definition and Design Goals

If all nodes in the blockchain network transfer transactions without verifications, invalid transactions would spread over the whole network. The spread of invalid transactions causes a waste of storage and network bandwidth resources. In addition, the block generator needs to verify the transaction set that contains a lot of invalid transactions before generating the block, thereby affecting the speed of generating the block. A straightforward way to prevent the spread of invalid transactions in the network is to have all nodes to verify all transactions. However, this way has the burden of repeatedly verifying valid transactions, and there is no penalty for free-riders who do not verify in the current blockchain system.

Therefore, we need to design a mechanism with the help of reputation to ensure the invalid transactions cannot be widely spread as well as reduce the number of verifications. The main idea of our mechanism is to encourage nodes who behave honestly and punish nodes who forward invalid transactions by adjusting their reputations. Intuitively, a node believes another node with a high reputation and doubts the node with a low reputation. Therefore, a node could make a decision to check a transaction or not, based on the reputation of the transaction sender. Moreover, honest nodes can derive behaviors of their neighbor nodes, accordingly infer the types of these neighbors and optimize the connection qualities.

IV. Reputation Mechanism

We begin this section by presenting our reputation mechanism. The reputation mechanism helps nodes strategically verify transactions. Additionally, we summarized several properties of the reputation mechanism.

A. Design of Reputation Mechanism

1) Overview: Every node in the network maintains a list of reputations, each of which is corresponding to one of its neighbors. We introduce the reputation mechanism from the perspective of an individual node i for brevity. Node i maintains the reputation R_{ij} for each neighbor node j. R_{ij} represents i's view of j's historical behaviors. Intuitively, node i tends to believe the neighbors with higher reputations, which makes it possible that node i does not need to check every received transaction. Thus node i can verify a transaction from neighbor j with a probability, depending on the reputation R_{ij}. In general, the verification probability of one transaction is lower if the sender of this transaction has a higher reputation. And then node i may update the reputation R_{ij} according to the verified results. A verified VC transaction from j may increase reputation R_{ij}, while a verified VI transaction or invalid transaction may decrease R_{ij}. Once the reputation R_{ij} reaches a minimum threshold of node i, which is predetermined, node i will think neighbor j as a dishonest node and cut the connection with j.

2) Verification probability function: Assume that node i receives a transaction T from neighbor node j. Node i will check T with a probability obtained from the verification probability function $f_i(x)$ of node i, which has the range $[q_i, 1]$. The parameter q_i is the minimum verification probability, which will be discussed in the following paragraph. The variable x of f_i is related to the reputation R_{ij}. For simplicity, the probability to verify transaction T is $f_i(R_{ij})$ in this paper.

3) Minimum verification probability: Adversaries may attack the reputation mechanism by accumulating reputation. To be specific, the nodes controlled by adversaries may behave honestly in advance to increase their reputations deliberately. After enough time, these nodes start to attack by conducting malicious behaviors. Since these nodes have a high enough reputation, other nodes will check the transactions from adversaries with a small probability. This will bring about the accelerated spread of invalid transactions over the network. Setting a minimum verification probability can help nodes to resist this kind of attack. Node i sets its minimum verification

probability q_i. No matter how much reputation has been accumulated before, node i would still verify transactions with a probability of at least q_i.

4) Reputation value updating: Assume that node i receives a transaction T from a neighbor j, the attached cycle cost is c_cycle, and the real cycle cost is r_cycle. Let S_T be a node set, each node $s \in S_T$ has forwarded transaction T to node i before. It is possible that $j \in S_T$, meaning j has sent T to i. Although S_T is related to i, we omit the index i in our discussion if there is no confusion.

If node i never received transaction T before, i.e. $S_T = \emptyset$, node i will verify this transaction T with a probability, obtained from probability function $f_i(R_{ij})$. Then node i updates R_{ij} by distinguishing following three cases of T:

- VC transaction, $c\_cycle = r\_cycle$: $R_{ij} \leftarrow R_{ij} + r\_cycle$.
- VI transaction, $c\_cycle \neq r\_cycle$: $R_{ij} \leftarrow R_{ij} - \max\{r\_cycle, c\_cycle\}$.
- Invalid transaction: $R_{ij} \leftarrow min\{R_{ij}/2, R_{ij} - \max\{r\_cycle, c\_cycle\}\}$.

If node i has received transaction T before, i.e., $S_T \neq \emptyset$, then there are two cases. The first is that node i has verified the transaction and $j \notin S_T$, i keeps the verified results, including the real cycle cost and the validity of T. Once node i receives T from node j currently, the reputation R_{ij} can be updated according to the existing verified results. The second is that node i has not verified transaction T, or $j \in S_T$, R_{ij} remains unchanged.

5) Reputation attenuation: Long term behaviors provide historic guarantees of the types of nodes. However, node behaviors may fluctuate for multiple reasons, such as corruption from adversaries, etc. As an accumulated value, the reputation value cannot reveal the behavior fluctuation of nodes through time. Meanwhile, when data from a long time ago conflicts with the latest data, we tend to use the latest behaviors as the main reference part and mitigate the impact of ancient behaviors. Thus, we propose an attenuation process. Each node i has private own variables t_i and p_i that indicate the frequency and the proportion of attenuation, respectively. Each reputation value R_{ij} turns to be $p_i R_{ij}$ per t_i time, where $p_i \in [0, 1]$ and t_i are selected by node i.

B. Properties of Reputation Mechanism

1) Privacy: Each node has reputation values to measure the reliability of neighbors. R_{ij} is private, since only node i computes and maintains the accurate value of R_{ij}. Furthermore, R_{ij} is only impacted by the verification results of node i.

2) Independence: Reputation R_{ij} is only related to historical transferred transactions from node j to node i. In other words, R_{ij} is independent of other R_{ik}, and other transactions from node $k \neq j$ cannot influence R_{ij}. This independence feature prevents R_{ij} from possible strategic attacks of any other nodes except for node i and node j.

3) Accountability: Assume that node j receives a transaction T from other nodes and forwards T to node i. Then R_{ij} is probably increased or decreased based on the validity of the transaction T. Although there may be more nodes forward T to node j through some transmission paths, node i cannot track the transmission paths of T except j. This implies node j takes accountability of this transmission.

V. Performance Evaluation

In this section, we provide the results of our simulations. We conduct multiple sets of trials in various environments.

A. Simulation Process

We show our simulation process in this subsection. For the convenience of simulations, we discretize the time into time slots with a fixed length. Each simulation has 1,000 time slots. In the following figures, if a figure only shows results until time period t instead of 1,000, it means that all observation measurements vary little in terms of performance from time period t to 1,000. A node performs the following operations in each time slot.

1) Generating transaction: In our simulations, we let each node generate a new transaction with a probability of 0.01 in one slot. As stated before, honest and lazy nodes only generate VC transactions and the malicious nodes can generate all kinds of transactions, including VC, VI and invalid transactions. Among all new transactions generated by a malicious node, half of them are VC transactions and others are invalid transactions in our setting.

2) Transaction cycle cost: The cycle cost of each transaction is similar to the gas, which is the pricing for running a transaction in Ethereum [3]. We crawl 388,691 transactions in Ethereum and use the gas number of these transactions as a reference to estimate the cycle cost in our simulations. Among all of these 388,691 transactions, there are 157,967 transactions, each having 21,000[1] gas. About 86.1615% of all crawled transactions have a gas number less than 10^5, and the percentage of transactions whose gas number is larger than 10^6 is less than 0.5%.

To generate a transaction cycle cost in our simulations, we randomly select one from all crawled transactions and use its gas number as a cycle cost. Since most of the gas numbers from the crawled transactions are less than 10^6, we let all cycle costs in our simulations are no more than 10^6 for simplicity.

3) Verifying transaction: All honest nodes use the same following verification probability function for simplicity:

$$f_i(x) = \begin{cases} 1, & x < 0 \\ 1 - x/(2 * 10^6), & 0 \leq x < 3 * 10^6 \\ 0.25, & x \geq 3 * 10^6 \end{cases}$$

The verification probability function is piecewise linear, where the probability of 0.25 is the minimal probability q to avoid the accumulated reputation attack. The breakpoint of $3 * 10^6$ is selected based on the gas numbers from the crawled transactions in Ethereum.

Assume node j sends a transaction T to node i. If i has not received a transaction T before, then i will verify T with

[1] 21,000 is the standard gas limit for regular transactions.

probability derived from $f_i(R_{ij})$. If i decides to verify T, R_{ij} will be updated according to the verification result. Otherwise, R_{ij} remains unchanged.

If node i has received a transaction T before, then i will not verify it anymore. If i has verified transaction T before, then it can use the existing verification result to update R_{ij} accordingly. Otherwise, R_{ij} remains unchanged.

4) Transferring transaction: All nodes can only forward T after receiving it for the first time. Dishonest nodes just transfer the received transactions to all neighbors without verification. When honest node i receives a transaction T from node j, it will conduct different operations as follows:

- If T is a VC transaction, then node i will transfer it to all neighbors.
- If T is a VI transaction, then node i will first revise the cycle cost, and then transfer it to all neighbors.
- If T is invalid, then node i will discard it.
- If T is decided to be unverified, then node i will transfer it to all neighbors.

5) Reputation attenuation: All nodes use the same attenuation frequency and proportion. For each reputation value R_{ij}, node i set it to $R_{ij} = R_{ij} - \left\lfloor \frac{R_{ij}}{10} \right\rfloor$ per 10 time slots.

B. Test Graph

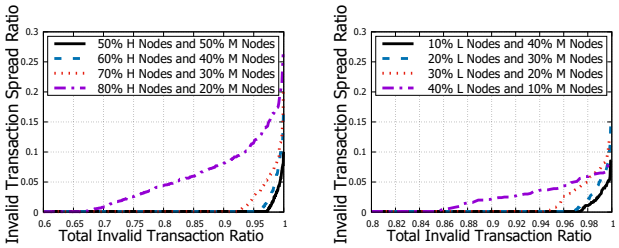
We apply Watts-Strogatz model [18] to generate our test graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, in which $\mathcal{V}$ is the node set and $\mathcal{E}$ is the connection set. In our simulations, $\mathcal{G}$ satisfies $|\mathcal{V}| = 2000$, $|\mathcal{E}| = 20000$, and we rewire each connection with probability of $\beta = 0.5$. This model can generate graphs which reveal properties of social network in the real world.

There are three kinds of nodes in our model. For the sake of simplicity, we distribute these three kinds of nodes evenly in our simulations. Since we conduct several simulations in which the proportions of each kind of nodes are different, the type of the same node in $\mathcal{G}$ may be different in these simulations.

C. Simulation Results

We run simulations in two series of environments. The first series includes 4 sets of simulations without lazy nodes, the proportions of honest and malicious nodes in these 4 sets are 50% and 50%, 60% and 40%, 70% and 30%, 80% and 20%, respectively. The second series includes 4 sets of simulations, in each of which the proportion of honest nodes is fixed as 50%. In addition, the proportions of lazy and malicious nodes in these 4 sets are 10% and 40%, 20% and 30%, 30% and 20%, 40% and 10%, respectively. For each set, we conduct 10 times of simulations and plot the figures by using the average of these 10 simulations. For simplicity in the following figures, we use *H Nodes*, *L Nodes* and *M Nodes* to represent honest, lazy and malicious nodes, respectively.

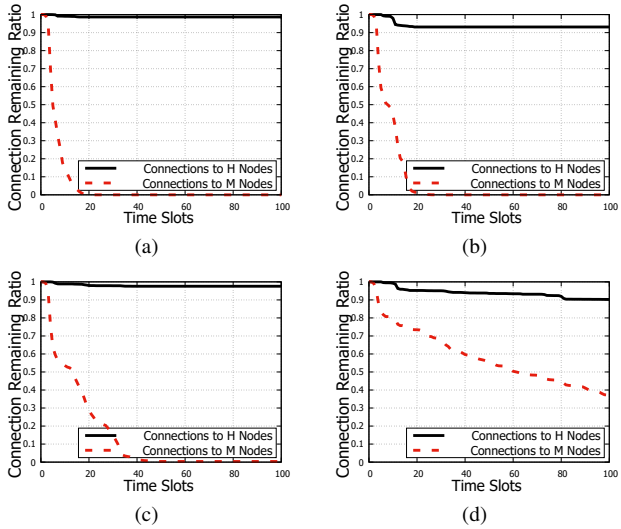
To characterize the spread of the invalid transactions, for each invalid transaction, we compute the ratio that the number of honest nodes who have received it over the total number of honest nodes. Fig. 1 shows the plots in different environments.



(a) (b)

Fig. 1. The spread of invalid transactions in honest nodes. (a) shows the results in environments without lazy nodes. (b) shows the results in environments with 50% honest nodes.

From these plots, we can find the extent of the spread of invalid transactions is higher if the proportion of honest nodes is higher. The reason is that if there are more honest nodes, the ratio of VC transactions in the network is higher too. Then, the malicious nodes could gain high reputations since they could forward more VC transactions. The high reputation will further decrease the verification probability and increase the possibility of forwarding invalid transactions by honest nodes. As the worst case in Fig. 1(a), the results of running simulations with 80% honest nodes and 20% malicious nodes show that no invalid transaction spreads to over 27% honest nodes, over 99% transactions spread to less than 18% honest nodes, over 90% transactions spread to 8% honest nodes. Fig. 1(b) shows that over 95% invalid transactions spread to 5% honest nodes. Compared with the mechanism that may propagate invalid transactions to all nodes, our mechanism can effectively prevent the spread of invalid transactions.



(a) (b) (c) (d)

Fig. 2. The connection remaining ratio between honest nodes and other nodes in no lazy nodes environments. The proportions of honest and malicious nodes are (a) 50% and 50%, (b) 60% and 40%, (c) 70% and 30%, (d) 80% and 20%, respectively.

Then we focus on the connections between the honest nodes and three kinds of nodes over time by computing the ratio of the current remaining number of connections over the initial number of connections. It can reflect the performance of distinguishing different types of neighbors by honest nodes.

Fig. 2 shows the results in environments without lazy nodes. It implies two significant phenomena. First, most of the honest nodes can keep high reputations to maintain connections while disconnecting from malicious nodes with low reputations. This implies our reputation mechanism can help honest nodes detect malicious nodes among their neighbors. Second, as the ratio of malicious nodes increases, the speed of disconnection also increases. This is because when the ratio of malicious nodes increases, fewer VC transactions can be transferred by them, thereby leading to a decrease in the reputation of malicious nodes.

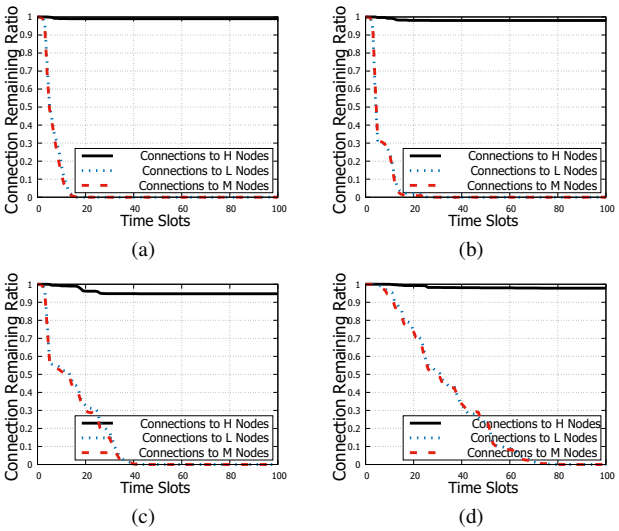


Fig. 3. The connection remaining ratio between honest nodes and other nodes in 50% honest node environments. The proportions of lazy and malicious nodes are (a) 10% and 40%, (b) 20% and 30%, (c) 30% and 20%, (d) 40% and 10%, respectively.

Fig. 3 shows the results in the environments with fixed 50% honest nodes. When the honest node ratio is fixed, the higher the malicious node ratio, the slower the malicious nodes are found. This is similar to the results in Fig. 2. Another phenomenon is that the connection remaining ratio of lazy nodes and malicious nodes are similar. It implies that honest nodes can find lazy nodes with the same efficiency as finding malicious nodes, even though malicious nodes are considered more harmful than lazy nodes. This is because the vast majority of transactions sent by each node are transferred from its neighbors. Invalid transactions generated by a malicious node itself only account for a few portions. Since lazy nodes and malicious nodes do not verify transactions, they transfer similar proportions of invalid transactions from the perspective of honest nodes.

VI. Conclusion and Future Work

In this paper, we have proposed a holistic reputation mechanism that aims for preventing the spread of spam transactions in the blockchain system. Each node locally maintains a list of reputations of its neighbors, which are derived from the verified results of the received transactions. At last, we have conducted a series of simulations to demonstrate the performance of our method. The simulation results show that our reputation mechanism can prevent the spread of the invalid transactions and can help the honest nodes distinguish the lazy and malicious neighbors efficiently.

In terms of future work, we will seek a relationship between the fault-tolerance accuracy ϵ and verification probability complexity. Moreover, regular new nodes in permissionless environments have no historical records nor reputation. We will propose a solution to allow nodes with low reputation values to exchange their computational resources for services.

Acknowledgement

This work was supported in part by National Natural Science Foundation of China (No. 61632017).

References

[1] S. Nakamoto *et al.*, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[2] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, p. 37, 2014.

[3] G. Wood *et al.*, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, no. 2014, pp. 1–32, 2014.

[4] B. Xu, D. Luthra, Z. Cole, and N. Blakely, "Eos: An architectural, performance, and economic analysis," 2018.

[5] K. Baqer, D. Y. Huang, D. McCoy, and N. Weaver, "Stressing out: Bitcoin "stress testing"," in *International Conference on Financial Cryptography and Data Security*. Springer, 2016, pp. 3–18.

[6] M. Saad, M. T. Thai, and A. Mohaisen, "Poster: deterring ddos attacks on blockchain-based cryptocurrencies through mempool optimization," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*. ACM, 2018, pp. 809–811.

[7] S. Paavolainen, T. Elo, and P. Nikander, "Risks from spam attacks on blockchains for intemet-of-things devices," in *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE, 2018, pp. 314–320.

[8] L. Luu, J. Teutsch, R. Kulkarni, and P. Saxena, "Demystifying incentives in the consensus computer," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 706–719.

[9] B. B. F. Pontiveros, C. F. Torres *et al.*, "Sluggish mining: Profiting from the verifier's dilemma," in *International Conference on Financial Cryptography and Data Security*. Springer, 2019, pp. 67–81.

[10] P. Resnick and R. Zeckhauser, "Trust among strangers in internet transactions: Empirical analysis of ebay's reputation system," in *The Economics of the Internet and E-commerce*. Emerald Group Publishing Limited, 2002, pp. 127–157.

[11] K. Aberer and Z. Despotovic, "Managing trust in a peer-2-peer information system," in *Proceedings of the tenth international conference on Information and knowledge management*. ACM, 2001, pp. 310–317.

[12] S. Buchegger and J.-Y. Le Boudec, "A robust reputation system for mobile ad-hoc networks," Tech. Rep., 2003.

[13] M. Srivatsa, L. Xiong, and L. Liu, "Trustguard: countering vulnerabilities in reputation management for decentralized overlay networks," in *Proceedings of the 14th international conference on World Wide Web*. ACM, 2005, pp. 422–431.

[14] D. Liu, A. Alahmadi, J. Ni, X. Lin, and X. Shen, "Anonymous reputation system for iiot-enabled retail marketing atop pos blockchain," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3527–3537, 2019.

[15] K. N. Khaqqi, J. J. Sikorski, K. Hadinoto, and M. Kraft, "Incorporating seller/buyer reputation-based system in blockchain-enabled emission trading application," *Applied Energy*, vol. 209, pp. 8–19, 2018.

[16] H. Chai, S. Leng, K. Zhang, and S. Mao, "Proof-of-reputation based-consortium blockchain for trust resource sharing in internet of vehicles," *IEEE Access*, vol. 7, pp. 175 744–175 757, 2019.

[17] J. L. Hennessy and D. A. Patterson, *Computer architecture: a quantitative approach*. Elsevier, 2011.

[18] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *nature*, vol. 393, no. 6684, p. 440, 1998.