Chapter

The Impact of Denial-of-Service Attack for Bitcoin Miners, Lisk Forgers, and a Mitigation Strategy for Lisk Forgers

Davi Alves

Abstract

Bandwidth depletion Denial-of-Service (DoS) attack can impact the propagation of a mined block in the Bitcoin blockchain network. On Bitcoin Proof-of-Work (PoW) consensus several machines try to resolve an expensive cryptographic puzzle faster than anyone else and succeed to mine a valid block. Despite a DoS attack impedes one machine to propagate its mined block allowing it to become valid for most peers, there will be several other peers to resolve the puzzle in time, hence the blockchain will continue to grow. However, from the perspective of the owner of the attacked machine, this can be critical because it will not receive a mining reward. This chapter covers such an attack in the Lisk blockchain that utilizes the Delegated Proof of Stake (DPoS) consensus mechanism. A mitigation strategy was created based on two tools that I have created allowing a delegate account to be configured in more than one node, allowing to forge a block even when one of its nodes is under DoS attack. Also, the transaction flood DoS attack is explored, and a mitigation strategy was created for a specific sidechain in the Lisk ecosystem. The mitigation strategy identifies spam transactions and rejects them to be included on the Lisk nodes transaction pool, hence they will not be propagated into the blockchain. Towards the end, I evaluated scenarios and mitigation strategies created for each attack demonstrating solutions for several scenarios.

Keywords: Lisk, DoS, PoW, DPoS, bandwidth depletion attack, transaction flood attack, mitigation strategy, dynamic programming

1. Introduction

The necessity of a solution to allow transactions between peers without a third-party authority was the main reason for the Bitcoin [1] blockchain creation. The Bitcoin whitepaper demonstrates this necessity as follows: “Commerce on the Internet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust based model. Completely non-reversible transactions are not really possible, since financial institutions cannot avoid mediating disputes. The cost of mediation increases transaction costs, limiting the minimum practical transaction size and

cutting off the possibility for small casual transactions, and there is a broader cost in the loss of ability to make non-reversible payments for nonreversible services. With the possibility of reversal, the need for trust spreads. Merchants must be wary of their customers, hassling them for more information than they would otherwise need. A certain percentage of fraud is accepted as unavoidable. These costs and payment uncertainties can be avoided in person by using physical currency, but no mechanism exists to make payments over a communications channel without a trusted party. What is needed is an electronic payment system based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third party. Transactions that are computationally impractical to reverse would protect sellers from fraud, and routine escrow mechanisms could easily be implemented to protect buyers [1]".

Blockchain technology promises to redefine trust in distributed systems by serving as a tamper-proof and transparent public ledger that is easily verifiable and difficult to corrupt [2]. Hence, it is utilized a consensus protocol that allows participants, for example, all copies of the blockchain, to agree on a unique version of the true state of the network without a third authority [3]. Despite blockchain are publicly verifiable and tamper-proof, each node on a public blockchain is a computational node that is exposed on the Internet by default, therefore, exposed to denial-of-service attacks or distributed denial of service attacks (DoS/DDoS), it will be used DoS for DoS or DDoS attacks from here. On Delegated Proof of Stake (DPoS) consensus, a consensus composed by a numeric quantity of nodes configured with delegate accounts, unique accounts responsible to forge blocks and change the state of the network, each node configured with a delegate account has a determined slot of time to forge a block in the network [4].

This chapter exposes two types of DoS attacks that can have a great impact on the Lisk Blockchain network. One attack is called a bandwidth depletion attack that sends several UDP packets at high speed against a computational node. Hence, at the moment a delegate account configured on a single node is successfully attacked by a DoS during its time slot for forging a block, then it cannot forge a block during that specific amount of time exclusively allocated to it. The second type of attack is called transaction flood attack and the main purpose is to send spam transactions that are valid in format, but they have small value and fee costs. The goal of such an attack is to fulfill a queue that resides on each node of the Lisk blockchain called transaction pool. When the transaction pool of a node is full then the node cannot accept any other transaction from anyone in the blockchain, this way all new transactions created and sent to any node are rejected.

A solution for the bandwidth depletion problem was already discussed in [4] and this chapter brings an update on the solution. However, the necessity of a solution for the transaction flood attack is discussed also in [5] and the results demonstrate the resilience of the mitigation strategy that was implemented against such attacks. It was shown two different types of a flood attack, one more expensive, it sends a transaction with proper relay fee and mining fee that would make them first selected for a block not allowing other transactions to be selected first, and another one attack cheaper and powerful in Bitcoin ecosystem. The latter was based in send several transactions with a proper relay fee but a small mining fee, this way such transactions were not selected for a block and always went to mem pool, a queue in Bitcoin.

The necessity for a solution to mitigate transaction flood attacks in the sidechain context and bandwidth depletion attack were the main reason for writing strategies to such problems.

This chapter is organized as follow: Section 2 abords related works, Section 3 presents Lisk version 2.3.8 and brings the first look at Lisk version 5, Section 4

details the impact of DoS attacks in Lisk blockchain, Section 5 demonstrates the strategies to mitigate bandwidth depletion and transaction flood DoS attacks, Section 6 demonstrates results, and Section 7 concludes the chapter.

2. Related works

In the context of Blockchain and transaction flood mem pool attacks, there are some papers, and it was identified [5] as relevant for this chapter. It analyzes two specific ways to perform flood attacks in the Bitcoin network. One is based on sending several transactions with appropriate relay fees and relevant mining fee to allow spam transactions to be included in a block not allowing real transactions from real users be included in a block, hence the transaction from real users go to mem pool. In this form of attack, the mem pool starts to grow as a valid transaction is not been chosen by miners since the spam transaction has more relevant fees to be included in a block. However, this is an expensive attack, and a transaction fee already difficult in this scenario. The second mem pool attack [5] states that spam transactions are created by Sybil accounts with a relay fee above the minimum relay fee required for a transaction, however, the transactions mining fee is below the minimum fee to be included directly in a block, this way transactions go to mem pool and stay there. Also, mem pool size grows, and for a real user to create a transaction and have it included in a block it is needed to include a relevant price for the transaction fee. After the same procedures, it was discovered that it increases Bitcoin transaction fee price by time and keeps other transactions away from been included in a block. Finally, [5] proposes a solution for such a scenario that can detect transactions spams and reject properly such transactions.

Vasek et al. [6] presents an empirical study of DoS attacks on the Bitcoin ecosystem, it was identified that most of the attacks occur on currency exchanges, then mining pools. Also, it was analyzed how was constructed the dataset of DoS registers and currency exchanges that already suffered attacks and took countermeasures of anti-DoS protection. Among the conclusion was presented that big mining pools are bigger targets for DoS attacks than smaller mining pools especially because of the mining race on Bitcoin. A mining pool can launch a DoS attack against other mining pools if it realizes that the concurrent mining pool can have mined a block and tries to spread it into the network.

3. Lisk Blockchain

The Lisk Blockchain utilizes DPoS as its consensus mechanism to forge blocks and updates the state of the network [2]. DPoS is a consensus that elects the top 101 active delegate accounts based on the higher-weight voted delegates. Only delegate accounts can vote for delegate accounts. The voting mechanism in Lisk 2.x can be represented by the formula below and in **Table 1**:

$$\sum_{i=1}^{n} Xi$$

Lisk nodes communicate between them using a P2P network based on transaction propagation and block propagation. Each node on Lisk utilizes Javascript Object Notation (JSON) objects with blocks and transactions compressed to

Labels	Definition
X	Amount of LSK voter holds
i	voter
n	Amount of voters

Table 1.
Vote mechanism formula in Lisk version 2.X.

communicate [4]. The P2P networks enable scalability on the network, avoid a single point of failure, and prevent a small group of participants controls the network [3]. Also, a consensus mechanism allows establishing a new state in the network. On every 10 seconds, a block is forged on the network including a maximum of 25 transactions in it and only delegate accounts can forge a block in Lisk.

The 5 components of a blockchain that are important to achieve a consensus protocol are: Block proposal that generates blocks and attaches essential generation proofs, Information propagation that disseminates blocks and transactions across the network, Block validation that checks blocks for generation proofs and the transactions within, Block finalization that reaches consensus on certain blocks, and Incentive mechanisms that encourage honest participants and drive the system to move forward [7].

3.1 Delegate accounts

Delegate accounts are the unique accounts that can forge a block in the Lisk network. To forge a block each delegate account among the 101 top active delegate accounts has a specific time slot of 10 seconds [2]. There is no competition between delegate accounts during a time slot, each time slot belongs to a single delegate account and only that specific delegate account can forge a block at that moment.

3.2 Transaction pool

Transaction pool is a queue that resides in a Lisk blockchain node [2]. By default, this queue has a capacity of 1000 transactions plus 1000 multi-signature transactions. Any transaction created and broadcasted to a Lisk node is stored in its transaction pool. When the moment to forge a block arrives for a delegate account, it gets transactions from its transaction pool and includes them in a block. In Lisk version 2.x the transaction pool queue sorts transaction by their timestamp.

3.3 Lisk sidechain SDK 2.3.8

"Lisk sidechain is an exclusive blockchain that accepts custom transactions developed with the Lisk transaction library. Despite that a sidechain accepts only transactions supported by the Lisk blockchain network and custom transactions registered in the sidechain, this is a remarkably interesting characteristic especially because the sidechain does not allow transactions from different sidechains. Furthermore, the transaction space on a sidechain block is reserved only for the custom transactions supported by itself and the transactions supported by the Lisk blockchain. Each node in the sidechain network executes and accepts the same type of transaction. However, until the version used in this paper, 2.3.8, there is no communication yet between Lisk blockchain and the Restaurant sidechain [8]".

3.4 A first look in Lisk SDK 5.x

Lisk introduces, on version 3.0, Lisk-BFT, a customizable fault-tolerant framework of consensus algorithms from the famous Paxos protocol [9] to improve efficiency and resistance against Byzantine faults [4]. The Lisk-BFT protocol is a forkful protocol where there is no requirement for a block proposer to achieve consensus before adding more blocks to the block tree, for more information see [10]. Many improvements were added in Lisk SDK 5.x, like an increase of block size to allow a capacity of 128 transactions instead of the only 25 transactions of version 2.x. Also, the increase of the size of the transaction pool, the inclusion of dynamic fees on transfer transactions, and much more can be verified in [11].

4. Impact of DoS in Lisk Blockchain

This section starts by bringing two types of DoS attacks that can directly impact the performance of Lisk Blockchain. These DoS attacks are bandwidth depletion and transaction flood attacks in the transaction pool.

4.1 Denial-of-service bandwidth depletion attack

Bandwidth depletion attacks can have a great impact on blockchain application owners in the Lisk environment and on delegate accounts. This affirmation is important especially because of DPoS consensus mechanisms in Lisk. In DPoS, delegates are the unique accounts that could forge a block in Lisk, hence if a node configured with an active delegate account is attacked during its time slot and prevents the forge of a new block then the blockchain will lose at least 10 seconds. A delegate time slot is attached to one specific delegate, and only it can forge a block. Another problem caused by a DoS attack against a delegate account is the transactions accumulated in the transaction pool, in this period, no block with transactions can be created during the time slot of the attacked delegate. Any new transactions created by users will arrive in the transaction pool increasing its occupancy rate.

Figure 1 demonstrates a use case of a Lisk application that allows users to request food online, and each food request represents a single transaction in the blockchain. **Figure 2** demonstrates a bandwidth depletion attack.

Table 2 shows a scenario of transaction requests in the Lisk application and how much it can be affected by a DoS attack during its most important period of usage.

Figure 1.
Users requesting food online on a Lisk application. Each food request generates a transaction to be submitted into the blockchain.

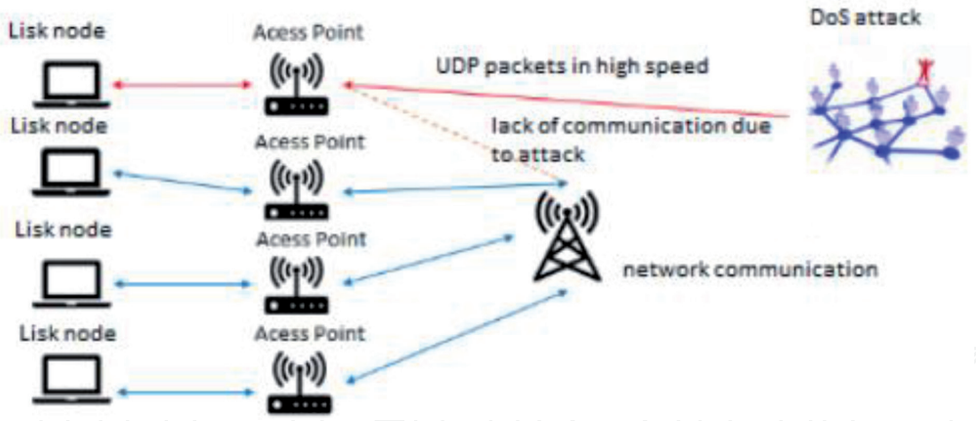

Figure 2.
Bandwidth depletion DoS attack against a delegate account configured on a Lisk node [4].

Labels	Restaurant 1	Restaurant 2
Number of tx	50	90
maximum Request time without dos	20 seconds	40 seconds
Dos duration against 1 delegate account	15 minutes	15 minutes
total waiting time of a user	30 seconds	50 seconds

Table 2.
DoS attack on a single delegate account affects waiting time. Since a delegate account forges only a single block in a round of 16 minutes, the waiting time during a DoS increases at least by 10 seconds. The most impacted in this scenario is the attacked delegate account that cannot receive a reward as it did not forge a block.

4.2 Transaction flood attack

Transaction flood attack consists of sending an immense quantity of transactions that necessarily do not are related to the business itself. For example, using the same use case of Restaurants and food requests, a transaction flood attack would consist of sending several transactions of small value that are not food requests, they are just transactions with small amount value, but in large volume to surpass the capacity of a block and increase the number of transactions into transaction pool until it gets full. After achieving the capacity of the transaction pool, no other transaction created by any user will be accepted in the transaction pool, they will just be refused and discarded not getting a chance to be included in a block. **Figure 3** demonstrates a scenario that the transaction pool is full, and valid transactions created by a user online on the restaurant website are discarded.

Table 3 shows a scenario of transaction requests in the Lisk application and how much it can be affected by a transaction flood DoS attack during its most important period of usage.

The next section will investigate mitigation strategies for bandwidth depletion DoS attacks and flood transaction attacks.

5. Strategies to mitigate DoS attacks on Lisk Blockchain

In this Section, it will be presented the mitigation strategies for transaction flood attacks and bandwidth depletion attacks.

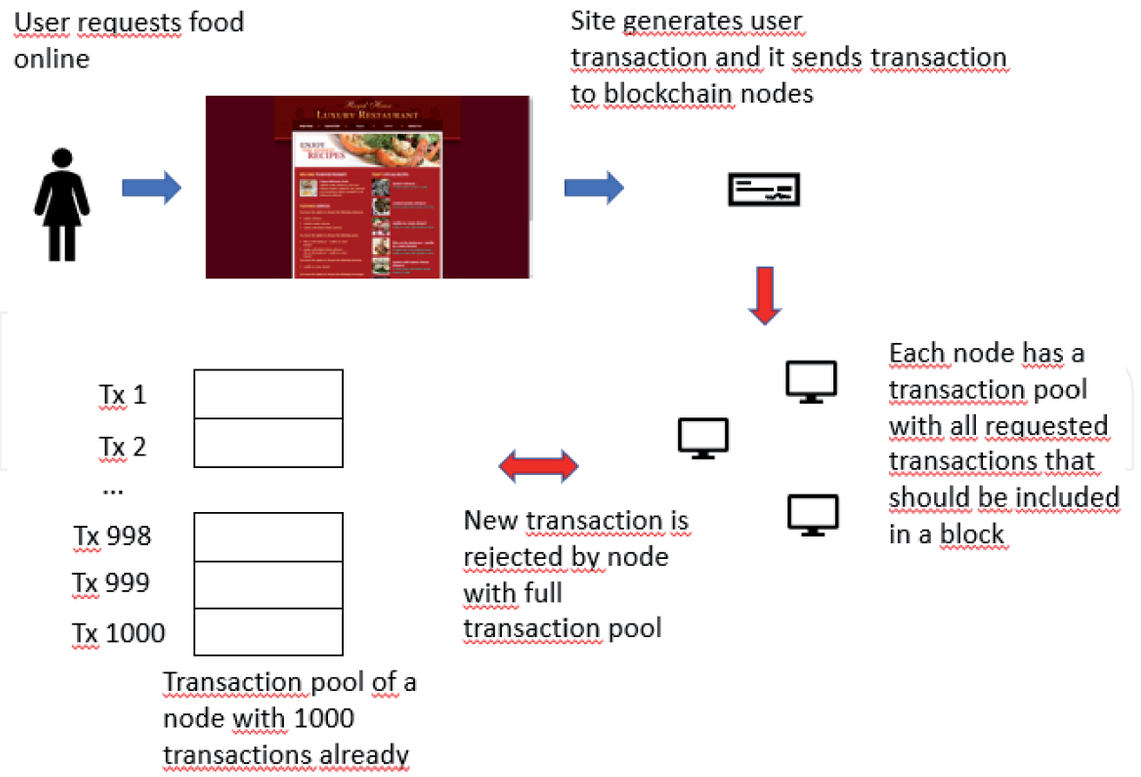

Figure 3.
The transaction pool capacity of 1000 transactions is already full. Any new transaction that arrives on a node is rejected until the transaction pool has some space after including transactions on new blocks.

Labels	Restaurant 1	Restaurant 2
Number of tx	50	90
maximum Request time without dos	20 seconds	40 seconds
Transaction flood Dos duration with full capacity loaded	15 minutes	15 minutes
total waiting time of a user	15 minutes +20 seconds	15 minutes +40 seconds

Table 3.
Transaction flood DoS attack on blockchain. When the transaction pool capacity is full no other transaction can be included in it. Sporadically, a valid transaction from a user can be added into the transaction pool by chance because of the normal flow of block creations by forgers and the inclusion of transactions from the transaction pool into a block. The most impacted in this DoS scenario are users, applications, exchanges that cannot include transactions because any transaction is rejected.

5.1 Transaction flood DoS mitigation solution

The proposed solution for transaction flood DoS attack is based on the use of dynamic programming to verify if a transaction was already sent before and to verify if a transaction has specific characteristics of a valid transaction in the context of the restaurant business. It was used a use case of a restaurant sidechain to explain the flood transaction mitigation strategy.

5.1.1 Restaurant use case mitigation solution

In a restaurant sidechain [8], it is possible to request food online using a specific type of transaction, the food transaction type. This specific type of transaction has a unique identifier for any transaction and a specific transaction type code. Despite

that a sidechain can accept regular transfer transaction types, from a restaurant business perspective, only the food transaction type should be accepted to buy food in the sidechain. Hence, any spam transaction that is not a food transaction should be rejected. Also, if spam transactions are specified as food transaction type, the food type is verified based on another transaction type called menu transaction (MT). The MT type lists all foods accepted by a restaurant with detailed information as price, name, description of all foods. The transaction id is verified to not allow the same transaction id to be used several times into the same block in the blockchain. Finally, spam transactions are also validated in amount and fees. Sender balance is verified before a transaction is spread to other nodes in the blockchain, this reduces the verification impact in case of spam attacks. Any transaction that does not fit in the specified criteria is considered a spam transaction and therefore rejected by the solution. Despite this solution was utilized in a sidechain, it can be adapted to be used directly in a blockchain.

5.1.2 Dynamic programming approach in the solution

To allow the success of the proposed solution it was specified a threshold number of transactions to be verified for transaction flood attack. This is important because the transaction pool of any node has the size of 1000 transactions capacity by default. To allow a response in time is important to specify a threshold, this way the problem becomes an NP-Complete problem to solve. If no threshold is specified, then as much spam transaction arrives harder the problem becomes to solve, and a solution to mitigate such a scenario becomes more expensive to verify.

5.2 Bandwidth depletion DoS mitigation solution

The proposed solution of bandwidth depletion DoS attack mitigates it on an active delegate account configured on monitored nodes of Lisk blockchain as published in [4]. The solution uses tools to monitor the synchronization level of correct blocks in blockchain between connected nodes and allows a delegate account to forge a block and spread it into blockchain nodes even when a specific monitored node is under bandwidth depletion attack. This specific synchronization level is called Broadhash Consensus and is an aggregate rolling hash of the last 5 blocks on de node data storage. When a specific tool called Forge Verifier (FV) detects that a monitored node is under attack, it verifies between all monitored nodes which one has the best synchronization level of blocks in the blockchain, and then it selects the best node for forging at that moment setting the forging status of the best-synchronized node to true and all the other monitored nodes forging status to false. Hence, when a forging slot of a configured delegate account arrives then it can forge a block even when another monitored node is under bandwidth depletion attack. Furthermore, the solution allows the continuity of block generation by the configured delegate account without wasting the slot time allocated to it on each forging round.

5.2.1 Architectures elements

For the mitigation strategy proposed in [4], there are 3 main architectural elements. The blockchain node API allows to perform HTTP requests to know the synchronization level of a monitored node and therefore allows the FV tool to specify the forging node at that moment. The FV tool continuously monitors nodes configured in a file, called monitor.json, based on synchronization level, then it sends a request to a tool called Forger Lisk (FL). The new possibility in the FV tool

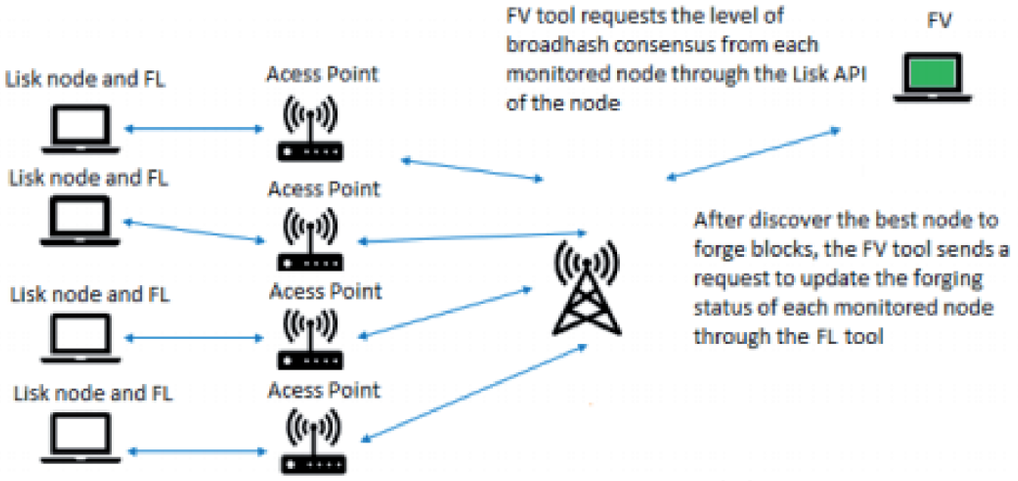

Figure 4.
Architecture elements of bandwidth depletion mitigation solution [4].

is to configure, in the monitor.json file, the maximum waiting time that FV should wait to retrieve the synchronization level from the monitored nodes specified there. For example, if FV is monitoring 3 nodes, and the maximum waiting time is specified to 10 seconds, then FV requests each node for its synchronization level, which goes from 0 to 100, and all monitored nodes should respond in 10 seconds otherwise they will be considered not accessible nodes by FV. After wait 10 seconds for an answer, FV will choose a node from the nodes that responded to the synchronization level request to be a forger in that round. After FV performs this step, it sends a request for an update to the FL tool, the goat is to update the forging status of monitored nodes through nodes API where the FL tool resides. FL tool exists because it is only possible to change status on any Lisk node through local requests. Hence, FV can reside on any computer that has access to the internet, however, FL needs to reside in the same computational node of a monitored Lisk node (**Figure 4**).

6. Performance evaluation

The contribution of this chapter was performed evaluating both DoS scenarios and mitigation solutions in a sidechain network for the transaction flood DoS attack and in the Testnet network for the bandwidth depletion attack. The tools utilized, links, and GitHub source code can be found in Appendices Section.

6.1 Transaction flood sidechain performance evaluation

The sidechain results were performed using Lisk SDK 2.3.8 version, 101 delegate accounts, a backend representing one restaurant connected to sidechain nodes, a web application connected to the restaurant backend, a script to generate spam transactions and send them to the restaurant backend. In this solution, the restaurant backend provides the mitigation strategy on its API method (**Table 4**).

6.1.1 Testing environment

The Lisk SDK 2.3.8 was utilized to create the restaurant sidechain and it was executed in all blockchain nodes. The backend was developed in nodeJS and connected with sidechain nodes. Web application representing restaurant website was developed using React technology.

Scenarios	DoS flood transactions different from food transaction	DoS flood transactions as food transaction but with different food offered by the restaurant	DOS flood with same transaction ID	Dos flood transactions wallet without enough balance
with proposed solution	Mitigated	Mitigated	Mitigated	Mitigated
without solution	Denial-of-service	Denial-of-service	Broadcasted to sidechain nodes that refuse spam transaction (overhead)	Broadcasted to sidechain nodes

Table 4.
*In **Table 4** above, it is possible to understand use case test scenarios performed with DoS attack and mitigation solution.*

6.1.2 Evaluation and use cases

A transaction flood attack was performed using a valid balance transfer transaction that was sent every 100 ms against the restaurant backend. During the attack, it was accessed the Lisk restaurant website and performed a valid food request online successfully. The video was recorded and included in Appendices Section.

6.2 Bandwidth depletion Testnet performance evaluation

The Testnet results were provided already by [4] and are commented on here. Testnet delegate accounts can be found at Lisk Testnet explorer. The Lisk version utilized was 2.3.x, also it was utilized a new relic tool to record monitored nodes API requests and it was calculated response time for API nodes.

6.2.1 Testing environment

"The Lisk Core version 2.1.3-RC.0 was used on all monitored nodes during the tests on Testnet network and it implements Lisk protocol. It was used Visual Studio Code to implement the tools FL and FV and they were executed on NodeJS version 10+. The chosen network for the tests was Testnet. It was assumed as a premise that bandwidth depletion DoS attacks using UDP protocol against one of the monitored nodes by the FV tool. There are some tools and sites that can perform such types of attacks. Also, videos were recorded, calculated average of requests per minute, and response time was monitored with Newrelic. Furthermore, it was captured Pcap files allowing network data to be analyzed with the Wireshark tool or reproduced with Tcpreplay for better comprehension of the tools developed, the solution strategy, and their behavior on Lisk [4]".

6.2.2 Performance analysis

In the following scenario, it was tested bandwidth depletion attack against one of 4 monitored nodes. Even with the attack against a delegate account configured in one of all 4 monitored nodes, it was possible to forge blocks with a single forger node while the other ones were updated to be non-forgers. Also, a video demonstration of a similar attack was performed and recorded, it is available in Appendices Section. **Table 5** shows the result [4]:

Definition	node 1	node 2	node 3	node 4
number of requests per minute on node api	1–2	1–2	1–2	1–2
response time on node API	Good	Good	Good	Insufficient
dos number of requests	0	0	0	+5000000

Table 5.
Test results of DoS bandwidth depletion against node 4 [4].

7. Conclusions

This paper presented a strategy to mitigate transaction flood denial-of-service attacks reducing overhead in the blockchain, and allowing the spread of valid transactions between nodes. Also, it was presented a strategy to mitigate bandwidth depletion denial of service attack on Lisk allowing, in most situations, the continuity of blocks been forged by a delegate account reducing the probability of creating forks and loss of forging block time slots. As a result of the analysis, it was observed that a restaurant online solution continued to provide service even during the spam transaction attack. Also, it was observed that during the bandwidth depletion denial-of-service attack monitored nodes attended the requests from FV while attacked nodes struggle to respond in time. For future works, I expect to adapt proposed solutions to the new Lisk SDK version 5 with Lisk-BFT consensus.

Acknowledgements

Our thanks to Sidechain Solutions for an idea of DoS flood attacks that helped me to create DoS flood scenarios.

Conflict of interest

The authors declare no conflict of interest.

Notes/thanks/other declarations

I would like to thank Manu from the Lisk team for reviewing Lisk information: "It's possible to configure transaction pool to increase the pool size to accept more transactions, this doesn't solve the DoS attack, but allows a restaurant-like application to process valid transactions in sidechain if required."

Appendices and nomenclature

Lisk SDK. https://github.com/LiskHQ/lisk-sdk
Visual studio code. https://code.visualstudio.com/updates/v1_50
Nodejs. https://nodejs.org/en/
ReactJs. https://reactjs.org/
New Relic https://newrelic.com/
Bandwidth depletion videos: https://github.com/davilinfo/ACM_conference/tree/master/videos/testnet

Source code mitigation solution of DoS flood attack Lisk restaurant Backend.: https://github.com/davilinfo/intechopen_2020

Lisk Restaurant http://liskrestaurant.com:5000/

Bandwidth depletion and flood transaction videos: https://github.com/davilinfo/intechopen_2020/videos

Author details

Davi Alves
UFBA, Salvador, Brazil

*Address all correspondence to: davi.alves@ufba.br

References

[1] Nakamoto, S. 2008. Bitcoin: A peer-to-peer electronic cash system. Available from: http://bitcoin.org/bitcoin.pdf. [Accessed: 2020-12-20]

[2] Kordek, M., and Beddows, O. 2016. White paper: Lisk. Technical report. Available from: https://whitepaperdatabase.com/lisk-lsk-whitepaper/. [Accessed: 2020-12-29]

[3] Pahl, C.; EL Ioini, N. and Helmer, S. A Decision Framework for Blockchain Platforms for IoT and Edge Computing. In Proceedings of the 3rd International Conference on Internet of Things, Big Data and Security - Volume 1: IoTBDS, 105-113, 2018; Funchal, Madeira, Portugal; DOI: https://doi.org/10.5220/0006688601050113

[4] Davi Alves. A Strategy for Mitigating Denial of Service Attacks on Nodes with Delegate Account of Lisk Blockchain. In Proceedings of The 2nd International Conference on Blockchain Technology (ICBCT'20); 2020; Association for Computing Machinery, New York, NY, USA, 7-12. DOI: https://doi.org/10.1145/3390566.3391684

[5] Saad, Muhammad & Kim, Joongheon & Nyang, Daehun & Mohaisen, David. Contra-*: Mechanisms for Countering Spam Attacks on Blockchain Memory Pools. Available from: https://arxiv.org/abs/2005.04842

[6] Vasek, M., Thornton, M., and Moore, T. 2014. Empirical Analysis of Denial-of-Service Attacks in the Bitcoin Ecosystem. In Proceedings of the International Conference on Financial Cryptography and Data Security (Christ Church, Barbados, March 07, 2014). FC'2014, 55-71. DOI: https://doi.org/10.1007/978-3-662-44774-1_5

[7] Xiao, Y., Zhang, N., Low, W., and How, Y. T. 2019. A survey of distributed consensus protocols for blockchain networks. ArXiv, https://arxiv.org/abs/1904.04098v3.

[8] Davi Alves. Proof-of-Concept (POC) of Restaurants food requests in the Lisk Blockchain/Sidechain. ISAIC Conference (2020); Journal of Physics Conference Series, ISSN: 1742-6596, 2021. DOI: https://doi.org/10.1088/1742-6596/1828/1/012110

[9] Lamport, L. 2001. Paxos made simple. ACM SIGACT News 32(4), 51-58. [Online]. Available from: http://lamport.azurewebsites.net/pubs/paxos-simple.pdf. [Accessed: 31/12/2020]

[10] Hackfeld, J., Lightcurve 2019. A lightweight BFT consensus protocol for blockchains. ArXiv, https://arxiv.org/abs/1903.11434.

[11] Lisk 2020. Launch of Betanet v5. Available from: https://lisk.io/blog/development/launch-betanet-v5. [Accessed: 2020-12-30]