A Strategy for Mitigating Denial of Service Attacks on Nodes with Delegate Account of Lisk Blockchain

Davi Alves
UFBA
Ondina, Salvador, Bahia, Brazil, PGCOMP
+55 71 997035287
davi.alves@ufba.br

ABSTRACT
In this paper, I evaluate a type of denial of service attack, bandwidth depletion, that difficult the block propagation in blockchain networks. Towards the end, I study the attack on Lisk blockchain and explore its effects in the Delegated Proof of Stake consensus. I also propose a methodology joint with two tools I've created as countermeasures against such type of attack. The methodology is composed of the configuration of the same delegate account in more than one node joint with the use of created tools capable to detect the percentage of consensus on each monitored node and activate block forging status in a single node dynamically. Therefore, allowing a block to be forged even when the delegate account is under attack on another node and reducing the chance of forks creation on the blockchain with the same delegate account configured and activated on two or more nodes in the same forging time slot.

CCS Concepts
• Computing methodologies → Distributed computing methodologies • Security and privacy → system security

Keywords
Lisk; Blockchain; DPoS; Cyber-attacks

1. INTRODUCTION
A blockchain is a distributed data structure that is replicated and shared among the members of a network [1]. It is a computational system that behaves like replicated and distributed state machines composed of several elements, mainly peer-to-peer(P2P) networks, consensus protocol, cryptographic keys, and sidechains or smart contracts. The blockchain technology was developed to allow transactions between a pair of nodes that do not mutually trust each other and without a central authority to intermediate the transactions. 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 [2]. 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.

ICBCT'20, March 12–14, 2020, Hilo, HI, USA

ACM ISBN 978-1-4503-7767-6/20/03…$15.00
https://doi.org/10.1145/3390566.3391684

Lisk is a blockchain [3] that utilizes DPoS consensus to achieve a new state in the network, the slice time to forge a block is the default for each active delegate account during each round. Each round is composed of 101 active delegates until the version 2.1.3, it runs on a round-robin sort and each active delegate has 10 seconds to forge a block. However, even without the necessity to perform proof of work [4] to generate a block, a node configured as active delegate account is the principal target for attackers in the network. Furthermore, 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 the DoS term for DoS and DDoS.

On [5] is shown that mining pools have been sporadically targeted by DoS attacks since 2011. Following [6], mining pools are the second-most frequently targeted of DoS attacks after only currency exchanges. On Bitcoin [4], first blockchain, a denial of service attack against a node that just mined a block can difficult a block propagation on the blockchain network impeding that this recently mined block becomes valid for most nodes in the network. Analyzing the temporal question of Bitcoin blockchain was observed that a block mined but with difficulty of propagation to a majority of nodes in the network would not be valid to the network when another node succeeds to mine a block and propagate faster its block on the network in the same round. Hence, the liveness property of blockchain would be held and a block would be mined and propagated eventually [7].

The problem discussed in this paper is the difficulty for forging a block by a delegate node on its time slot during a bandwidth depletion denial of service attack that overloads traffic on the network against this node configured with the related delegate account. Analyzing the temporal question for forging a block in a blockchain with DPoS consensus, a denial of service attack against a node that has a determined slice of time to forge a block impedes the propagation of block in the network. Differently from other types of consensus protocols, in DPoS, any other node configured with a different delegate account can't forge a block in the time slot of the attacked delegate account time slot. Only after the time slot of the attacked delegate account that another delegate account configured in a different node can forge a block. Hence, occurs loss of time to the blockchain that did not generate a block on proper time and loss of reward on the delegate account that was unable to forge a block on its time slot.

The necessity of a solution more resilient against bandwidth depletion DoS attacks for delegate nodes on Lisk blockchain is the

main reason of this paper that presents a strategy to mitigate such types of attacks on Lisk blockchain. The results demonstrate the resilience of the solution on mitigating bandwidth depletion DoS attack, also the strategy allows to monitor the percentage of consensus level on monitored nodes and facilitates countermeasures against DoS attacks.

This paper is organized as follows: Section 2 abords related works, Section 3 presents Lisk version 2.1.3, Section 4 defines the strategy to mitigate denial of service attack, Section 5 defines the application of the created tools on Lisk, Section 6 describes the test environment, metrics, and scenarios, Section 7 the conclusion and future works.

2. RELATED WORKS

In the context of blockchain there are some scientific papers and among the most relevant is [5] that analyses the intense competition on mining pools. The competition is manifested in two ways: increase of computing resources to try win the next mining race and the other way is a mining pool may trigger a costly distributed denial-of-service attack to lower the expected success outlooked by a competing mining pool. [5] states that the tasks performed by a mining pool could be affected and become slower after a DoS attack and be decisive on a mining race for crypto coins. Also, individual miners could become discouraged and leave mining pools that are not reliable after DoS attacks. Among the conclusion is observed that big mining pools are more susceptible to be attacked than smaller mining pools and bigger mining pools have more incentive to attack than smaller mining pools.

[6] presents an empirical study of DoS attacks on the Bitcoin ecosystem and identifies that most of the attacks are more likely primarily on currency exchanges, then mining pools, gambling operators, eWallets, and financial services. 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 through services like Amazon Cloud, Cloudflare, Incapsula. Furthermore, [6] mentions motivators for DoS attacks and the frequency of such attacks. The study case Mt. Gox was investigated for DoS attacks on currency exchanges and was discovered that were reported a disproportional number of DoS attacks during a high pike of volume transactions in the spring of 2013. Among the conclusion was presented that big mining pools are bigger targets for DoS attacks than smaller mining pools.

3. LISK BLOCKCHAIN

The Lisk blockchain [3] allows registering transactions in blocks identified by a cryptographic hash. Each block refers to the hash of the preceding block establishing a sorted link between the blocks. Furthermore, any node with access to the blockchain can read and discover the global state on the network [1]. To accomplish the blockchain goal is necessary 5 components on a consensus protocol, they 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 encourages honest participants and drives the system to move forward [7].

Lisk utilizes P2P communication with other nodes, it is organized in consensus network, blocks propagation, and transactions propagation. The P2P networks enable scalability on the network, avoid a single point of failure and prevent that a small group of participants control the network [2].

Each node on Lisk utilizes Javascript Object Notation (JSON) objects with blocks and transactions compressed to communicate [3]. The Lisk logic on each node performs remote procedure calls (RPC) and events to communicate transaction objects and blocks objects to other nodes and its backend utilizes NodeJS, a javascript technology as a server.

The Lisk protocol on DPoS consensus utilizes, until the version 2.1.3 that was tested and discussed in this paper, 101 delegate nodes to achieve a new state on the network. The number 101 was chosen because of the experience of other DPoS implementations. EOS [8] utilizes 21 producer's nodes, which would be equivalents to delegate nodes, and it has high performance.

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. [10] states that ideally, the desired would be block proposers to propose one block after another always referencing the previously proposed block via a directed edge and hence forming a tree with only one growing branch, i.e., a blockchain. However, due to the latency of communication between the block proposers or deliberate attacks on the network, there can be multiple child blocks of the same parent block and separate growing branches. Hence, it was needed a consensus protocol for block proposers to agree on one block at every height of the block tree. 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].

3.1 Broadhash Consensus on Lisk

Broadhash consensus on Lisk has a vital function as it prevents forks, due to node mistakes or connection errors [11]. The broadhash consensus of a node is defined as an aggregate rolling hash of the last 5 blocks on the node data storage. Hence, all nodes with the same blocks will produce the same broadhash consensus and propagate that information in the system header message on P2P communication.

Broadhash is established if 51% of randomly selected nodes (100 or a lower number of randomly selected nodes) have the same broadhash. Delegate nodes use the broadhash consensus as a strategy to forge a block. Once the broadhash consensus is established, the delegate node propagates the new forged block to 25 random connected nodes on itself, and the nodes that received the new block continue propagating the block [12].

4. STRATEGY TO MITIGATE BANDWIDTH DEPLETION DENIAL OF SERVICE

The proposed solution in this paper mitigates DoS attack on an active delegate account on a configured node of Lisk blockchain.

The solution allows a delegate account to forge blocks on the network even when it is under bandwidth depletion attack on a node. The solution dynamically activates the forging ability on another node configured with the same delegate account in the network.

The mitigation strategy proposed acts monitoring the level of broadhash consensus on each monitored node configured on a file

of the verification tool. On the moment of a DoS on a monitored node, the tools created allow the update of the forging status of the monitored delegate account on another node also monitored by the verification tool but that is not under attack and is synchronized with the blockchain. Furthermore, the solution allows the continuity of block generation by the delegate account without wasting the slot time allocated to itself on each forging round. The next sections detail the architectural elements and the interaction between elements.

4.1 Architecture Elements

The solution to mitigate the DoS has 3 elements as shown in Figure 1: The API of the Lisk blockchain (BC), the verification tool of broadhash consensus level called forger_verifier (FV), and the tool that updates a monitored node forging status called forger_lisk (FL).

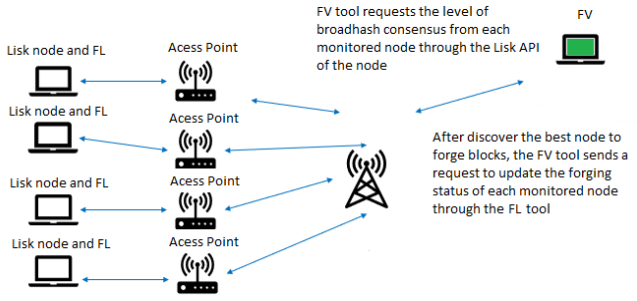

Figure 1. Architecture elements

4.1.1 Blockchain API

The blockchain API allows to gather important information from a node of blockchain as broadhash consensus level, number of peers connected to the node, block height on the blockchain network, block height of the node and other information.

4.1.2 Verification Tool (FV)

The verification tool is the element responsible to monitor the level of broadhash consensus on each monitored node included on the configuration file. The monitoring is possible through the configuration of a list of nodes on a JSON file called monitor.json and it can be represented as: {“time”: 60000, “hosts”: [{ "host" : "10.0.1.2", "port" : "7000", "consensus":0,"publicKey": "62bbb3c41e43df73de2c3f87e6577d095b84cf6deb1b2d6e87612a9156b980f8", "forging": false, "online": false, "gatewayport": 10000 }]}.

The FV tool requests on each monitored node its level of broadhash consensus periodically. The period request is configurable on the tool changing the property time of monitor.json file. Each request is performed through HTTP protocol and utilizes the related port of the node API on the network.

4.1.3 Forger Lisk (FL)

The Forger Lisk. is a tool responsible to update the forging status of a monitored delegate account, it is configured on a Lisk node and it runs on a different port than Lisk ports. Furthermore, it is possible to update the forging status of a node on Lisk only through local requests. The FL tool was developed with javascript technology and NodeJS.

4.2 Elements Interaction

The FV tool determines which monitored node among all nodes specified in monitor.json file has the best broadhash consensus between them on the blockchain, then it requests to update their forging status through FL. The communication between the FV and FL tools is possible through an HTTP endpoint created on FL. The FV tool performs requests to FL tool that executes on all monitored nodes periodically then FL requests locally to the Lisk API of the node to update its forging status. After FV receives the response request from FL, it restarts the monitoring cycle.

5. APPLICATION OF TOOLS ON LISK

The FV and FL tool’s source code are available on github[1]. The FV and FL tools were applied on Lisk blockchain with the goal of allowing the propagation of blocks and forge blocks on a delegate account configured and monitored in more than one node even when one of the monitored nodes have the trouble of synchronization and block propagation in the network.

5.1 FV Specification and Monitoring

FV starts accessing the file monitor.json to discover what nodes to monitor, then FV stores the related nodes on objects called servers. From this moment the FV flow is to discover the broadhash consensus percentage of all monitored nodes performing a request to the nodes, update the online property of each node, determine the best node to forge a block and store it on betterConsensusServer variable, then FV calls the method updateServerProperties that is responsible to update forging state of the betterConsensusServer and invokes a method to update the status forging of each node using the gateway port on FL. Finally, FV reinitializes on each monitored node the properties online, consensus, forging, and starts a new cycle of monitoring.

All the following properties belong to monitor.json file. The host property specifies the IP address of a host to monitor; port property specifies the communication port with the Lisk API of the monitored node; consensus property specifies the initial level of broadhash consensus, and the range is from 0 to 100 where 0 is not synchronized. Also, it is necessary at least 51 to forge a block; publickey property specifies the delegate account to be monitored, each node should be monitored with the same publickey; forging property specifies the initial forging status of the node; online property specifies the access status to the Lisk API of the node; gatewayport property specifies the communication port to FL tool.

5.2 FL Specification and Forging Status Update

FL should execute on each monitored node that was specified on FV tool; its goal is to update the forging status on the node where it executes after receiving a request from FV tool. The Lisk API allows updating a node status only from local request to a node. Hence, FL tool requests to the API of the node to update the forging status of it. For control and security, the FL tool has a configuration file called account.json where is necessary to specify the publickey of the delegate account running on the node. After receiving the request from FV tool, FL tool compares the publickey from the request with the publickey stored on account.json and it requests the update of forging status to the node API only if both publickey information are equal. Once FL receives the response from the node it responds to FV request.

[1] https://github.com/davilinfo/ACM_conference.git.

6. PERFORMANCE EVALUATION

The contribution of this paper and proposed by the mitigation solution against bandwidth depletes DoS was performed by tests on Lisk testnet network, that is a test network to all released version of Lisk and can be used for testing purposes. Any delegate account, forged blocks, transactions and other specific information related to testnet network can be found on Lisk testnet explorer[2]. Also, it was performed dumps from nodes on testnet network, verified using wireshark, analyzed monitored node network data, calculated response time of the API of node during attack[3] and recorded videos. Due to a known issue[4] on Lisk, until version 2.1.3, that affects node connectivity to other peers in the testnet network it was necessary to restart, and change IP address of all monitored nodes once per day in an attempt of a better peer connection to other peers. The FV tool helped to handle the known issue as an alternative during the tests.

6.1 Testing Environment

The Lisk Core[5] 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[6] to implement the tools FL and FV and they were executed on NodeJS[7] 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[8] and sites[9] that can perform such type of attack. Also, was recorded videos, calculated standard deviation, confidence interval, average of requests per minute and response time were monitored with newrelic[10]. Furthermore, it was captured pcap files allowing network data to be analyzed with wireshark or reproduced with tcpreplay for better comprehension of the tools developed, the solution strategy and their behavior on Lisk. Finally, the log of FV tool is available for analysis.

6.2 Evaluation Scenario

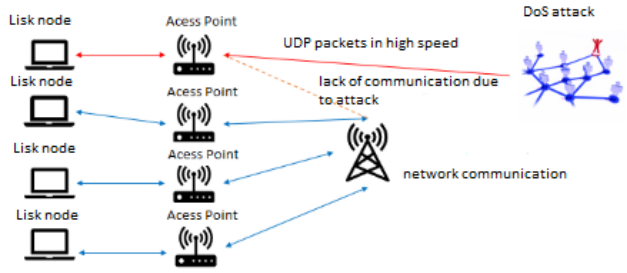

Figure 2. DoS attack

To evaluate the first scenario was utilized 3 Amazon Cloud virtual machines (VMs) each one with Lisk node and FL tool connected to Testnet, and 4th machine was utilized for FV tool. It was performed 3 days of experiments. On the second scenario was utilized 3 Amazon Cloud virtual machines (VMs), 1 VM outside of the cloud, all with Lisk node and FL tool connected to Testnet, a 5th machine was utilized for FV tool. The Figure 2 represents an attack scenario of bandwidth depletion DoS on Lisk.

6.3 Performance Metrics and DoS Parameters

To analyze the node performance before attack and after attack was established some metrics on the network and API of monitored nodes, they are the number of requests on node API, and response time of node API during the tests.

To generate the attack using DoS tool it was established the following parameters: host parameter that represents IP address of attacked node; method that is UDP protocol; threads that are number of concurrent users; port: communication port to attack.

FV tool was configured to perform verification and requests to the node API and FL, generally, 1 time per minute plus the time to retrieve broadhash consensus information.

6.4 Performance Analysis

On Scenario 1 using Amazon Cloud VMs it was not possible to take down a node using bandwidth depletion type of attack with the selected tool and site[11]. However, it was monitored the network in/out of the nodes and API requests to monitored nodes, data are available on github. Table 1 demonstrates the average results of requests and response time on API of monitored nodes and network data. A response time below 1 second is considered good as the current timeslot to forge blocks on Lisk takes 10 seconds, and detailed metrics are available on github. A number of requests between 1-2 is expected, since the FV tool was configured for a cycle of 1 minute and it requests on each monitored node API 1 request to discover the broadhash consensus and it performs another request to FL for updating the forging status of the node.

Figure 3 shows the confidence interval on Scenario 1 that was calculated with a formula and average, standard deviation, critical value and standard error metrics[12].

Figure 4 shows network usage during 3 days of test on Scenario 1.

Table 1. Scenario 1 test results

Definition	Node 1	Node 2	Node 3
Number of requests per minute on node API	1-2	1-2	1-2
Response time on node API	Good	Good	Good
Number of requests per minute on node API with DoS	NA	1-2	1-2
Response time on node API with DoS	NA	Good	Good

[2] http://testnet-explorer.lisk.io/delegateMonitor.

[3] Available on https://github.com/davilinfo/ACM_conference.

[4] https://github.com/LiskHQ/lisk-sdk/issues/4279. Verify 2.1.4.

[5] https://lisk.io/documentation/lisk-core/

[6] https://code.visualstudio.com/

[7] https://nodejs.org/en/

[8] Low Orbit Ion Canon

[9] https://www.stressthem.to/booter

[10] https://www.newrelic.com

[11] It was utilized LOIC and the site stressthem (basic option and UDP protocol) against node on Amazon cloud but without success.

[12] A worksheet of Scenario 1 is available on github.

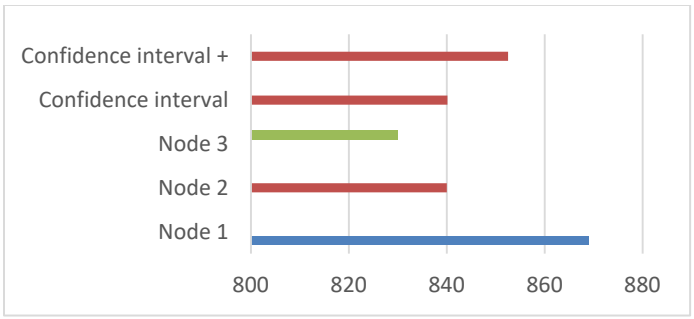

Figure 3. Scenario 1

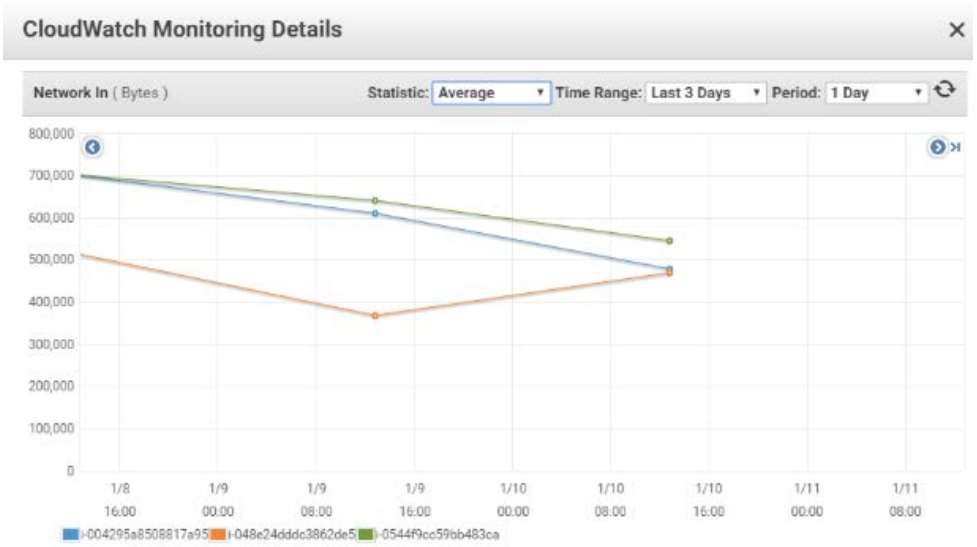

Figure 4. Network usage

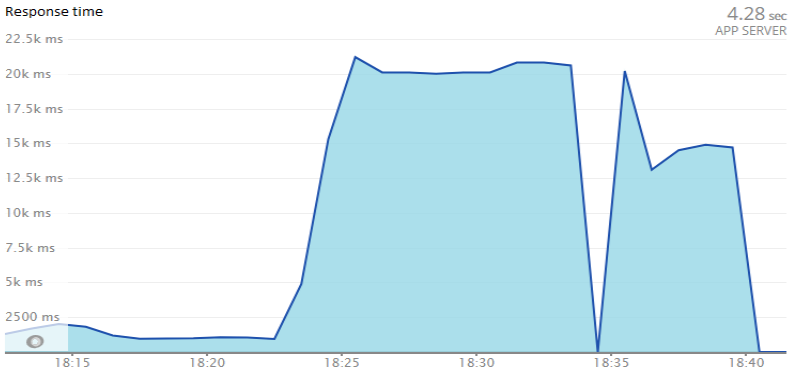

Figure 5. DoS attack

On Scenario 2 It was necessary only a few minutes to perform a bandwidth depletion attack and desynchronize one monitored node outside of the Amazon Cloud as shown in Table 2. However, the FV tool was smart enough to detect the low consensus or offline status of the node and chose another node with better broadhash consensus to be a forger of a block on testnet network in the majority number of tests.

Figure 5 shows the moment that was started a DoS attack on port 7000 against node 4, Table 2, that was running Lisk on port 7500. Also, it shows the moment that the node stops to respond to the FV tool. Furthermore, it shows the increase of response time of the node API during the attack.

Table 2. Scenario 2 test results with DoS on Node 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	+50000000
Threads attack	0	0	0	10

7. CONCLUSION

This paper 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 result analysis, it was observed the increase of response time by a node under such type of attack or even its unavailability that was detected by the FV tool. Also, the FV tool's log can be used as an indicator to verify nodes stability on the network and allowing a user to take countermeasures on a situation of DoS attack. As future works, I would like to improve the developed tools implementing high availability, fault tolerance, automation, reliability as well as continue researching more resilient strategies on blockchains.

8. ACKNOWLEDGMENTS

Our thanks to ACM SIGCHI for allowing us to modify templates they had developed.

9. REFERENCES

[1] Christidis, K., and Devetsikiotis, M. 2016. Blockchain and smart contracts for the internet of things. *IEEE Access*, 4:2292-2303. DOI= http://doi.org/10.1109/ACCESS. 2016.2566339.

[2] Pahl, C., EL Ioini, N., and Helmer, S. 2018. 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. DOI= 10.5220/0006688601050113.

[3] Kordek, M., and Beddows, O. 2016. *White paper: Lisk*. Technical report.

[4] Nakamoto, S. 2008. Bitcoin: A peer-to-peer electronic cash system. [Online]. Available: http://bitcoin.org/bitcoin.pdf.

[5] Benjamin, J., Laska, A., Grossklags, J., Vasek M., and Moore T. 2014. Game-Theoretic Analysis of DDoS Attacks Against Bitcoin Mining Pools. In *Proceedings of the International Conference on Financial Cryptography and Data Security* (Christ Church, Barbados, March 07, 2014). FC'2014, 72-86. DOI= https://doi.org/10.1007/978-3-662-44774-1_6.

[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. [Online]. Available: https://arxiv.org /abs/1904.04098v3.

[8] Larimer, D. 2017. *Whitepaper: EOS.IO*. Technical report.

[9] Lamport, L. 2001. Paxos made simple. ACM SIGACT News 32(4), 51-58. [Online]. Available: https://www.microsoft.com/en-us/research/publication/paxos-made-simple/

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

[11] Wang, W., Hoang, D. T., Hu, P., Xiong, Z., Niyato, D., Wang, P., Wen, Y., and Kim, D. I. 2019. A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks. IEEE Access, 22328-22370. DOI= https://doi.org/10.1109/ACCESS.2019.2896108.

[12] Lisk 2019. Visited in January 2019. [Online]. Available on web archive: http://lisk.io/documentation.