Analyzing the Effects of Network Latency on Blockchain Performance and Security Using the Whiteblock Testing Platform

Qi Trey Zhong
zhongq@usc.edu

Zak Cole
zak@whiteblock.io

***Abstract*— Blockchain technology has received significant attention recently, due in part to the rise of cryptocurrencies such as Bitcoin and their potential to act as a catalyst for economic and financial change. Although tokenomics have helped propel blockchain into the mainstream, the capabilities of this technology extend far beyond cryptocurrency. Also referred to as distributed ledger technology, it is speculated that blockchain will act as a catalyst for disruption on a global scale and blockchain-based solutions are currently being developed and applied within several industries, such as the supply chain, medical, and legal sectors. Due to its immutability, transparency and ability to eliminate the need for trust between transacting parties, blockchain allows for a more secure and decentralized method for the transaction of assets. Ethereum, often referred to as Blockchain 2.0, introduced the smart contract, which allows for the development of decentralized applications running on top of a blockchain. To ensure the development of fully functional and highly secured blockchain systems, protocols, or distributed applications, a reliable framework for testing and evaluating blockchain performance is necessary within the space. As performance and scalability becomes a more obvious bottleneck for the mainstream adoption of blockchain technology, developers and engineers should be encouraged to consider taking a bottom-up approach when architecting these systems and distributed applications by evaluating the effects of network impairments as one of the first steps within the design process. In this paper, we propose a new approach for evaluating the performance and security of blockchain systems under a variety of network conditions and attack scenarios.**

I. INTRODUCTION

Blockchain technology is anticipated to pave the way for the peer-to-peer economy of the future. By combining peer-to-peer networks, cryptographic security, distributed databases, and decentralized consensus mechanisms, it provides a way for people to agree on a particular state of affairs and record that agreement in a secure and verifiable manner. Prior to the advent of blockchain, it was simply impossible to coordinate the transfer of data and other assets without the need to assume a certain degree of trust between parties. A group of unrelated individuals could not confirm that an event had occurred without relying on a central authority to verify that this particular transaction was not fraudulent, invalid, or tampered with in some way. In fact, many computer scientists did not believe that a group of actors could reach consensus without a common clearinghouse due to one of the biggest challenges in distributed systems referred to as the Byzantine General's Problem.[1].

The concept of blockchain technology proposes a probabilistic solution to this classic problem by replacing the need for trust with a process of cryptographic proof, presented as mathematical puzzles that require significant computational power to solve. This makes it both extremely difficult and costly for a bad actor to corrupt a shared database or provide false information, unless they control a majority of the network's computational power. [2] Blockchain protocols thus ensures that transactions on a blockchain network are valid and trustworthy, enabling parties to coordinate individual transactions in a decentralized manner without the need to rely on a central authority to validate transactions.

While the promise of blockchain technology is met with great enthusiasm, there is no standardized testing platform available that allows for the deterministic evaluation of performance and security within these distributed systems, their protocols and the applications running on top of them. Furthermore, there seems to be a general lack of understanding for exactly how network effects, such as congestion, even impact these systems and a conclusive understanding often isn't acquired until it's too late. For example, CryptoKitties [3], a virtual game built on Ethereum that allows players to buy and breed virtual "crypto pets," resulted in the severe throttling of the Ethereum network. By December 2017, CryptoKitties' sales hit $12 million and accounted for almost 15 % of Ethereum's total network transactions which severely congested the entire Ethereum network and reduced transaction throughput. This brought the scalability issue[4] front and center and also presented a need for the development of more effective consensus mechanisms which allow faster transactions at a lower cost. Research and development is currently being conducted to identify new solutions and protocols, such as Casper, however, until now, a reliable framework for evaluating blockchain performance and security has been unavailable.

II. BACKGROUND

A. Popular Blockchain Protocols

Bitcoin is known as the first implementation of blockchain technology. It is a digital currency based on a protocol that allows users in the network to conduct transactions in a fast, and secure way. [5]

Ethereum[6], an early fork of the Bitcoin codebase, can be seen as a operating system for blockchain, which allows for the implementation of smart contracts that can apply business logic to transactions, a concept that had been proposed as early as 20 years ago[7]. Distributed applications extend the functionality of the network, allowing it to move from achieving consensus based on data streams to achieving consensus based on computational proof of work. Because of the supposed turing-completeness provided within the

programming language and the fact that computation is executed on every network node, one of the biggest issues is infinite loop, resulting in a contract that never terminates, causing network outage[8]. To protect against this type of malicious act, programmable computation in Ethereum is funded by 'gas' and a transaction is considered invalid if the 'gas' fee sent along with the transaction is insufficient to perform the associated computational work. Currently, Ethereum implements a Proof-of-Work (PoS) consensus algorithm, the Casper development project aims to implement the use of sharding and Proof-of-Stake (PoS) to address scalability issues.

EOS, which is still under development, proposes a similar function to Ethereum in that it utilizes blockchain, smart contracts and a native token to create a decentralized ecosystem for application development[9]. By providing database, account permission, scheduling, authentication, and inter-application communication functionalities, the goal of this system is to improve the efficiency of smart business development. Through the implementation of parallelization to improve scalability, EOS claims to allow for millions of transactions per second. While the current iteration of Ethereum uses PoW (Proof-of-Work) as its consensus algorithm, EOS uses a Delegated Proof-of-Steak (DPoS) model, which the EOS team claims will be able to process an average of 1,000 transactions per second in comparison to Ethereum's 25 transactions per second.

Hyperledger Fabric (HLF) is another blockchain platform designed for business applications[10]. It is an open-source project within the Hyperledger umbrella project and uses a modular permissioned blockchain platform designed to support pluggable implementations of different components, such as ordering and membership services. HLF allows clients to manage transactions using chaincodes, endorsing peers and an ordering service. Chaincode[11] is HLF's counterpart for smart contracts. It consists of code deployed on the HLF network where it is executed and validated by endorsing peers who maintain the ledger, state of a database (modeled as a versioned key/value store), and abide by endorsement policies. The ordering service is responsible for creating blocks for the distributed ledger, as well as the order by which each block is appended within the ledger. The Hyperledger Fabric team claims that they can achieve higher performance capabilities than Ethereum.

Despite the continual introduction of new blockchain protocols and development teams making lofty claims concerning the performance capabilities of their systems, no proof of these claims has been provided and the tests were more than likely conducted within the labs of each development team on local area networks using homebrewed utilities. It is therefore imperative for the integrity of the blockchain space to have a reliable and standardized platform for blockchain testing to ensure the authenticity of test results. In the following section, we will describe the Whiteblock testing platform that was developed to address these needs.

III. METHODOLOGY

A. Testing metrics

Before discussing the Whiteblock testing platform, it's very important to first define the metrics of blockchain performance. Within the testing series described, we considered 4 primary metrics for measuring the performance of a blockchain: throughput, the frequency of side fork occurrences, the frequency of block sealing occurrences, and fault tolerance.

- Throughput: throughput is measured as the number of successful transactions per second.
- Side fork occurrence frequency: This a very important point of observation in order to determine whether or not a system is stable. The more frequent a discovered block becames a side fork, the less secure the system is, especially when a sequence of blocks become a side fork.
- Block sealing occurrence frequency: This is an intuitive point of observation to see the rate of block generation which is a key factor for transaction rate
- Fault tolerance: fault tolerance is imperative to blockchain security. One intuitive way to measure fault tolerance is to measure the recovery time after a network is attacked. Recovery time is defined as the period of time it takes for the network to re-achieve consensus after a partition attack or a delay attack.

B. The Whiteblock testing platform

The Whiteblock platform was developed in order to address the aforementioned needs within the blockchain space. Using the Whiteblock platform, users can provision multiple nodes within an entirely private testing network. The number of nodes that can be provisioned depends on the system requirements, but can generally be hundreds or thousands. Each node exists within its own LAN with unique IP address. The network conditions of the links between each of these nodes can be configured with latency, packet loss, bandwidth constraints, blackout conditions, BER, and congestion in order to reflect real-world conditions, obtain a deterministic understanding for how the system will perform at scale once live, and provide a scalable testing environment which can be completely controlled.

C. Network topology

Using the Whiteblock testing platform, 10 Geth nodes were provisioned within a private Ethereum testnet. Each individual node existed within its own LAN which was assigned a unique IP address. The links between each node were configured with a variety of common network conditions and impairments, such as latency and packet loss, to reflect the performance of a dynamic and global blockchain network.

During the transaction test, node 1 to node 5 were configured as miners to process the transactions submitted to the transaction pool and node 6 to node 10 were sent transactions between one another according to the following specifications:

- Node 6 continuously sent 0.1 Ether to nodes 7-10. When its wallet balance reaches <90 ETH, the smart contract

deployed to node 6 forces the rotation such that node 7 becomes the sender until the balance of node 7 wallet reaches <90 ether when then the rotation action is triggered again to node 8, and so on.
- Node 7 to node 10 were configured in the same manner as node 6, however, the initial sender wallet address at the corresponding node.
- The reason to design the transaction test in such manner is to ensure that the test never encounters issues related to an insufficient wallet balance.
- The frequency of sending ether to each wallets is configured through an adjustable interval in the contract.

The figure below illustrates the transaction process:

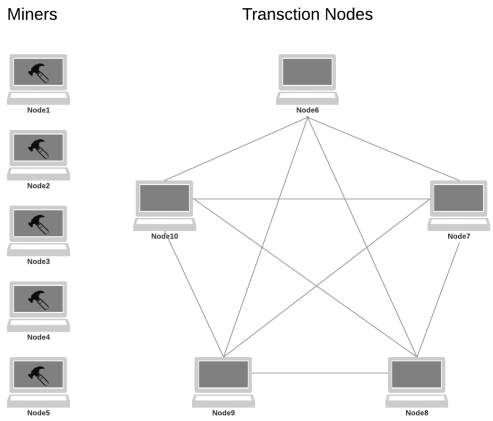

Fig. 1. Transaction test illustration.

D. Test conditions

1) Throughput test: To test the performance of a blockchain network, the first step is to create a control group. In this case, a control group is a set of tests run without implementing any network impairments or conditions. The transaction request rate was tested over a range of intervals from 10s second to 1ms per transaction batch. Each test lasted for 150 seconds. The purpose of this test is to discover the bottleneck of the current network setup. On a larger scale, these tests can be run over a longer period of time.

2) Network latency test: Network latency is applied to the blockchain network using the classification function of the Whiteblock platform. Two different global delays, 50ms and 100ms were applied to the network to emulate the behavior of a real-world network. Node to node delay is 200ms when the delays is 50ms. The node to node delay for 100ms delays is 400ms. The transaction request rate was tested over a range of intervals from 10s second to 1ms per transaction batch. Each test last for 150 seconds. The transaction request rate and the test period are the same as the control group.

3) Packet loss test: Current Ethereum documentation does not specify whether or not the network is able to handle packet loss up to 5%. It is very uncommon for a network to experience more than 5% packet loss. Generally, packet loss is below 5% or the network is completely cut off (which will be next test). 5% packet loss was applied to all inter-node connections during this test.

4) Network outage: Network outage is very common and can impede the performance of the network as well as causing issues with security. For this test, the network was configured such that network is out for 10 seconds every 60 seconds. The transaction request rate is configured at 200ms interval which as discussed in the RESULT Section is where the network throughput starts to plateau.

5) Network partition test: The network partition test is intended to examine the resilience of the network. The partition begins 10 seconds into the test. Two different partition durations were tested, one that lasted until 60 seconds and another that lasted until 90 seconds. During the partition attack, nodes are split into two groups:

- Group 1: Node1, Node2, Node3, Node 10
- Group 2: Node 4, Node 5, Node 6 - 9

6) Selfish mining attack test: This attack test is inspired by an article about the 25% attack (Refer to the article in the reference section). The basic idea is that instead of having control of more than 51% of the network, an attacker group can create an economic attack during which the attacker holds the blocks that are being discovered by the group without broadcasting the block to the rest of the network. The attacker group then engages a sybil attack to influence the propagation of the new block (found by other miners) while broadcasting the previously withheld blocks. The success of the attack will result in a financial loss to other miners which allows the attacker to incentivize other miners to join the attacker's branch. The attacking group can then initiate a 2nd stage 51% attack. Although this attack hasn't occurred within the Bitcoin or Ethereum networks, it's a legitimate threat and was therefore included in this benchmarking test series. To set the stage for this attack to occur, two conditions must be present:

- Condition 1: Attackers (selfish-miners) have some computing advantages. This condition is created by adding latency between miners nodes (non-attackers) and transaction nodes so that miners can't get transactions from the pool as fast as the attackers.
- Condition 2: Attackers have the ability to influence the propagation of new blocks from other miners. This is done by adding additional delays to the non-attacking miners (among miners) so it requires a longer period of time for those miners to reach consensus.

7) Spam attack test: Unlike the economic attack, a spam attack in Ethereum network did occur thanks to CryptoKitty incident, which at one point resulted in the congestion of the entire Ethereum network, delayed transactions, and created a bottleneck of unprocessed transactions. Although it was unintentional, the incident did raise awareness of the issue. In this test, one of the transaction nodes was setup to behave as a spam attacker which broadcasted transactions at 200ms intervals while the remaining transaction nodes operated at

10000ms intervals. Also, low amounts of network latencies were added to the non-attacking nodes to reflect real-world conditions and exaggerate the effects.

The numerical outcomes of the the tests will be illustrated in the RESULTS section.

IV. RESULTS

A. Throughput test

In this test, no network impairments were applied to the network. The transaction request rate was tested over a range of intervals starting at 1ms to 10s per transaction batch. Each transaction batch contains 20 transactions in total from all 5 transaction nodes. Each test lasted for 150 seconds.

Fig3 plots transaction throughput versus transaction request over the intervals described in the previous paragraph. As can be observed, the transaction throughput seems to plateau at 50 transactions per second.

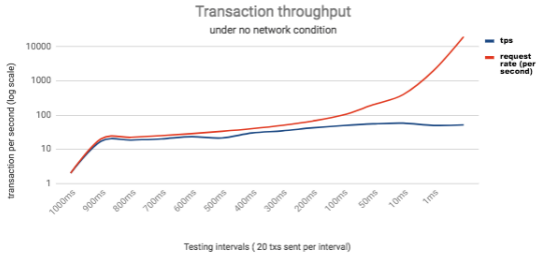

Fig. 2. Network Throughput with Delay

After investigating the log file, it appeared that the plateau is caused by a situation called "replacement transaction underprice". What this means is that when a transaction request rate reaches a certain level, the miners assume the additional transactions are replacements for the previous transactions, however, miners won't be able to process these replacements unless they're provided with 10% higher gas rate, which wasn't the case. Therefore, the additional transactions were rejected by miners.

B. Network Latency test

Two different global VLAN delays, 50ms and 100ms were applied to the network to simulate real-world network delays and geographic separation between nodes. Inter-node delay is 200ms and the VLAN delays are set at 50ms. The inter-node delay for 100ms VLAN delays is 400ms. The comparison between normal network conditions and network with latency is displayed below in Fig4.

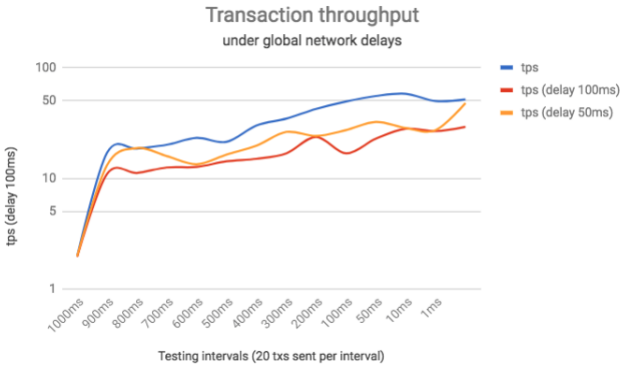

Fig. 3. Network Throughput with Delay

As can be seen in the plot above, 100ms VLAN delay slowed the network down by 58 %. When the VLAN delay is 50ms, the network slowed down by 41%.

C. Packet loss test

In this test, 5% packet loss is applied to all inter-node links during the testing period. The calculated throughput under this test condition turned out to be 48.84, which isn't much different from the normal throughput 49.3 transactions per second.

The table below compares the normal network condition and network with 5% packet loss through observing the fork occurrence and the block sealing occurrence. As can been seen the numbers are very close.

TABLE I
5% PACKET LOSS VS. NORMAL NETWORK

Occurrences	With Packet Loss	Normal Network
Blocks become side Fork	4	5
Blocks Sealed	33	33

The plots below compare the network with 5% packet loss to a network with no packet loss in terms of block number, fork occurrence and block sealing occurrence. The results are very similar which explains why the throughput under two network conditions are similar.

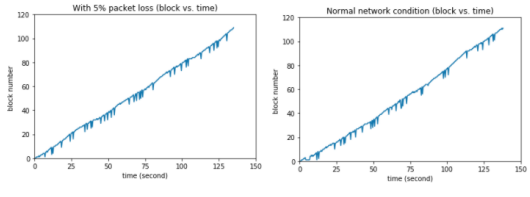

Fig. 4. 5% Packet Loss Test (Block vs. Time)

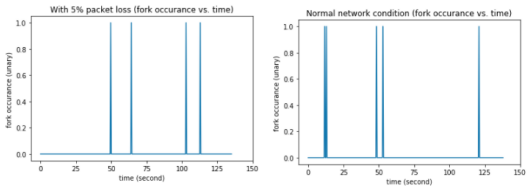

Fig. 5. 5% Packet Loss Test (Fork Occurrence vs. Time)

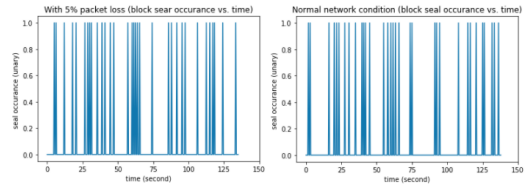

Fig. 6. 5% Packet Loss Test (Seal Occurrence vs. Time)

D. Network outage

In this test, the network was configured such that network outage was set to occur for 10 seconds at 60 second intervals. The transaction request rate was configured at 200ms intervals per batch. The calculated throughput under this test condition was 29.51 tps, 59% lower compared to the normal throughput 49.3 transactions per second.

The table shown below compares normal network conditions and network with periodic outage through observing the fork occurrence and the block sealing occurrence.

TABLE II
5% PERIODIC OUTAGE VS. NORMAL NETWORK

Occurrences	With Packet Loss	Normal Network
Blocks become side Fork	14	6
Blocks Sealed	32	23

The plots below compare the network with periodic outage with the normal network with no impairments in terms of block number, fork occurrence and block sealing occurrence. As can be seen, the periodic outage did slow down the block generation. Fig9 illustrates that network outage can create potential security issues through the generation of more side forks.

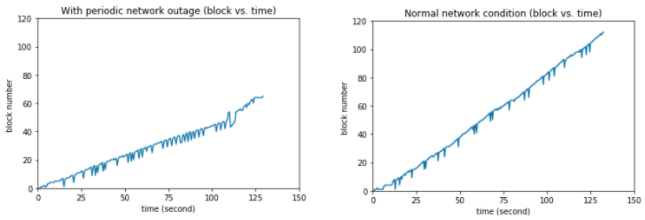

Fig. 7. Network Outage Test (Block vs. Time)

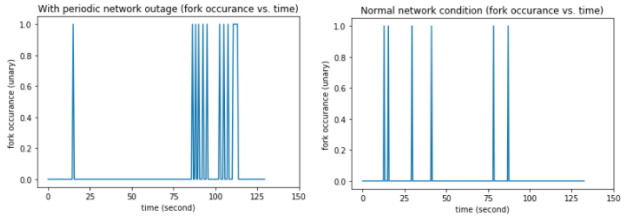

Fig. 8. Network Outage Test (Fork Occurrence vs. Time)

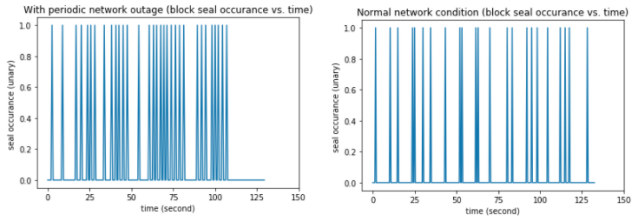

Fig. 9. Network Outage Test (Seal Occurrence vs. Time)

E. Network partition test

Two different partition durations were tested, one lasted 60 seconds and the other lasted 90 seconds. As illustrated in Fig11 and Fig12 below, it took 4 seconds for the network to recover from a 50 second partition and it took 6 seconds to recover from an 80 second partition. It is fair to conclude that the longer the partition, the higher the chance for vulnerabilities to network security since the window of recovery is also the window vulnerable to malicious attacks.

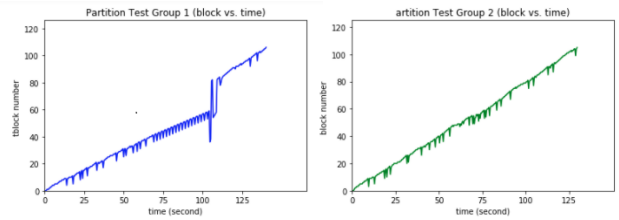

Fig. 10. Network partition Test 1 (90s partition)

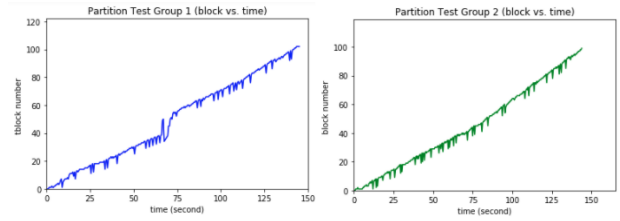

Fig. 11. Network partition Test 2 (50s partition)

F. Selfish mining attack

Below is a demonstration of the selfish mining attack effect as explained in the METHODOLOGY section. One mining group of bad actors withholds discovered blocks until its private chain is longer than the public chain and then they broadcast the blocks in order to cause other miners to migrate

to the private chain they created via selfish mining. As can be seen in the figure below (Fig13), interestingly, one of the normal miners was "lured" to join the side chain created by the attacker while the other two miners remained on canonical chain. It's not hard to foresee that the other two miners will be "lured" to join the side chain as the selfish mining attack repeats a few more cycles.

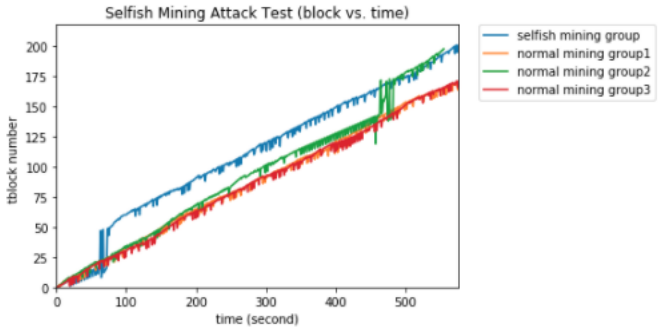

Fig. 12. Selfish mining attack test

G. *Spam attack test*

In this test, one of the transaction nodes was configured to act as a spam attacker which sent out transactions at a higher rate than a normal transaction node. A more detailed description of the test setup can be found in the METHODOLOGY section.

Shown below in Fig14 is the comparison of transaction reception rate between the normal transaction node and the node that initiated spam attack.

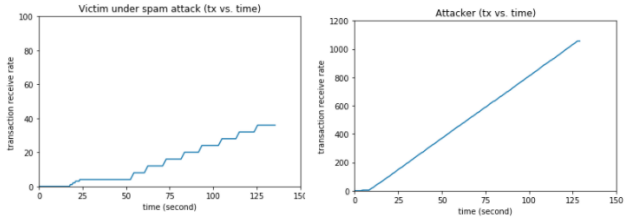

Fig. 13. Spam Attack Test

As can be observed, the spam node largely affected the processing of transactions from normal transaction nodes. The miners were occupied processing the the spam transaction thus causing the normal transaction processing rate to slow down by almost 90% (from 2 tps to 0.33 tps).

V. CONCLUSIONS

In this paper, we introduced the Whiteblock platform and presented a deterministic approach for analyzing blockchain security and performance under the influence of various network conditions within several scenarios. We demonstrated the bottleneck of the blockchain network as found within our testing series. The results show that current blockchain systems are not well suited for large scale data processing workloads and common network impairments. The performance of the blockchain network is largely jeopardized by certain levels of network latency and certain periods of network outage. The current security design of blockchain systems are largely vulnerable to attacks such as network partition, selfish mining and spam attack. Further work and research should be conducted to make blockchain systems more scalable and resilient. Additionally, a standard testing methodology should be developed in order to effectively decrease development overhead and reduce the time to market for systems under development. The Whiteblock platform is now available for use within the development community and the Whiteblock organization

REFERENCES

[1] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3):382–401, 1982.

[2] Christian Decker and Roger Wattenhofer. Information propagation in the bitcoin network. In *Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on*, pages 1–10. IEEE, 2013.

[3] Usman Chohan. The leisures of blockchains: Exploratory analysis. 2017.

[4] Wenting Li, Alessandro Sforzin, Sergey Fedorov, and Ghassan O Karame. Towards scalable and private industrial blockchains. In *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, pages 9–14. ACM, 2017.

[5] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[6] Vitalik Buterin et al. Ethereum white paper. *GitHub repository*, 2013.

[7] Michael Gord. Smart contracts described by nick szabo 20 years ago now becoming reality. *Bitcoin Magazine*, 2016.

[8] Falcon Momot, Sergey Bratus, Sven M Hallberg, and Meredith L Patterson. The seven turrets of babel: A taxonomy of langsec errors and how to expunge them. In *Cybersecurity Development (SecDev), IEEE*, pages 45–52. IEEE, 2016.

[9] T. Cox. Eos.io technical white paper. *GitHub repository*, 2017.

[10] Christian Cachin. Architecture of the hyperledger blockchain fabric. In *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, 2016.

[11] Elli Androulaki, Christian Cachin, Angelo De Caro, Andreas Kind, and Mike Osborne. Cryptography and protocols in hyperledger fabric. In *Real-World Cryptography Conference*, 2017.