Blockchain Security in IoT

Eric Nylander
eriny656@student.liu.se
Linköping University
Linköping, Sweden

Lukas Osipovič
lukos321@student.liu.se
Vilnius University
Vilnius, Lithuania

ABSTRACT

The application of blockchain technology to IoT networks is a subject of ongoing study due to the decentralized nature of the blockchain. IoT devices have unique limitations and requirements placed on them as relates to storage, computation and latency. Blockchain has the desirable property of democratizing processes among concerned nodes on a network. Applying these two fairly novel technologies on one another raises concerns as to the security of such a network, what vulnerabilities exist and what potential remedies can be applied. Any such remedy must be constrained by the aforementioned limitations and requirements on an IoT network.

The following paper analyzes inherent and applicable security measures of blockchain technology when practically applied to a simple IoT network using the Hyperledger Fabric framework and finds that certain aspects of the framework such as its private permissioned nature offer strengths as opposed to a general blockchain network. Further, it explores some limitations in the Fabric framework and documentation in application to a network of devices running mixed architectures.

1 INTRODUCTION

Due to the modern network infrastructure shifting into the domain of Internet of Things (IoT), there is a growing interest in applying novel solutions to communication and administration between these devices. Such devices include wearable devices and smart TVs, sensors such as motion and light sensors or some combination of sensors and computational devices. These devices may have unique requirements as they relate to latency on the network and are often limited in computational power and storage capacity.

Blockchain technology has recently gained traction in application to such IoT networks due to its decentralized nature, democratic administration of resources, and basic security requirements [18]. However, the application of this technology to a network architecture poses challenges to the limitations of devices on an IoT network with minimal computational and storage capacities while maintaining the network security.

Yakoob, et al. outline a series of requirements on an IoT network that an architecture would need to meet as relates to resource control, energy awareness, and security, among others [22]. We expand their definition of energy awareness to a more general resource awareness in order to incorporate the possible limitations of storage and computation power of the IoT devices.

With the expansion of IoT there have been a lot of work put into providing security to the network and its components. For instance, to help establish a secure End-to-End (E2E) connectivity between low-power IoT devices and cloud servers a mobile-based relay solution was presented and documented in detail [15]. There was also a study done on the development of a system model that ensures security on E2E connectivity between IoT devices and Unmanned Aerial Vehicles (UAV) and manages the data processing. [16]

1.1 Motivation

The following paper outlines a literature study done on the prerequisites of an IoT network as well as general information about blockchain technology. The paper then goes on to set up a simple IoT network using two Raspberry Pi 3s running Ubuntu 18.4 as well as a single admin device running Ubuntu 18.4 on x86_64 architecture. This simple network serves as a prototype for creating an IoT network using blockchain technology.

Tests are performed on adding and removing organizations and peers as well as modifying access control on this network. Beyond this, simple applications are deployed on the devices that utilize chaincode to query the ledger while the admin modifies the content of the ledger. Then, we ensure that actions beyond the permissions offered by the access control defined for the network are not possible to perform.

The latest version of Hyperledger Fabric released in January of this year and offers some valuable improvements to areas that relate to decentralization that apply well to an IoT network. [4] However, there exists no official method to deploy a network on distributed devices. Through analysis of the security merits of a blockchain network using the Hyperledger Fabric framework along with the creation a prototypical network using a mixture of different architectures typically present in an IoT network, we demonstrate areas of focus for future development of the security provisions of the framework and provide a basis from which to develop secure distributed networks using the Hyperledger Fabric framework.

2 BACKGROUND AND THEORY

This section will contain an analysis of blockchain technology as applied to IoT from literature studies as well as how Hyperledger Fabric implements various solutions.

2.1 Blockchain Properties

This subsection will analyze the theoretical security benefits and limitations of blockchain technology on an IoT network. Blockchain networks operate and validate transactions of a number of assets by having the relevant parties sign transactions using public-key cryptography. These transactions are easily validated by other members of the network to ensure that nobody on the network performs an unauthorized action.

When enough members have validated a transaction, it is added to a block of committed transactions on a peer who then adds the block to the end of the blockchain when certain conditions are met and propagates that block through the network. Once a block has been added to a blockchain, it cannot be removed by any interaction on the network; this ensures the immutability of the blockchain.[19]

The blockchain does not contain the current state of assets on the network. The blockchain does however store all transactions and other business data manipulations performed on the network, such as smart contracts. Thus, the state of assets on the network can be derived from the blockchain since it contains all state-transitions performed on the network.

2.1.1 *Consensus protocols.*

In order to add blocks to a blockchain, each block needs to be approved by a subset of members of the network. Determining which members need to approve a block before it is added is done using consensus protocols. There are a number of consensus protocols that are available to use on a general blockchain network, but many of them are designed to be used on networks that manage cryptocurrencies, or on devices that are more powerful than those expected on an IoT network.

Since IoT networks are limited in their computational and storage capacity, specific consensus protocols are required in order to preserve security on an IoT blockchain network. Salimitari et. al. suggest that the consensus protocols that are best for such a network are Proof of Elapsed Time (PoET), Practical Byzantine Fault Tolerance (PBFT), and Tangle. [17]

2.1.2 *Smart Contracts.*

Smart contracts allow users to deploy applications on the blockchain utilizing the decentralized aspects of blockchain technology. These applications can be used in IoT networks to manage devices on the network. [12]

2.1.3 *Permissioned Blockchain.*

Permissioned blockchains allow for application of access control to a blockchain network. This restricts who may perform arbitrary actions on the network, and ensures that each member of the network must be given a level of access control. This allows control over integrity and confidentiality of transactions on a network since it is possible to restrict the capabilities of vulnerable devices on an IoT network.

Since each device must be registered on the network to be given access, this also means that breaches on the network can be traced to individual devices. [9] In addition, certain consensus protocols require permissioned networks in order to be effective. [17]

2.1.4 *Ledger Fork.*

The order in which transactions occur on the network may differ as they reach members of the network and thus blocks may be added at different times and in different orders on different members. When two or more members append a block to the chain and propagate that new block through the network, it is called a ledger fork. Different blockchain technologies handle these ledger forks in different ways.

One such example is the Longest Chain Rule which states that when a fork is discovered on the blockchain by a member, the member chooses to switch to the longest chain. This introduces some nondeterminism to the order of transactions on the blockchain at any time and a transaction that has been appended to one fork may not exist on the second fork and thus has some probability of being invalidated later in time. [7]

2.2 Hyperledger Fabric Model

Hyperledger Fabric is a framework that can be used to build a network that uses blockchain to perform validation and ordering of transactions on the network. The project in this paper uses Hyperledger Fabric 2.0.

2.2.1 *Ordering Service.*

The ordering service is a group of members that ensures that transactions are added to blocks in the same order across all members. When a transaction is added to a block, it must first be ordered among all other transactions that have been committed to blocks on the network.

The ordering service by default requires that all orderer members validate the order of transactions before a block is appended to the blockchain. This ensures that the blockchain is deterministic and that a ledger fork is impossible on a Fabric blockchain.

2.2.2 *Ledger.*

Across all Hyperledger frameworks, there exists a ledger that is updated across all members of the network, providing eventual consistency on the blockchain on the network. Hyperledger extends the concept of the ledger and stores the ledger as two objects: the blockchain and the world state database.

The blockchain follows the general definition of a blockchain in that it is an immutable, write-only data entity that contains the information about transactions of assets on the network.

The world state database on the other hand contains information about the current state of assets on the network. As such, when a block is added to the blockchain, it is possible to update the world state database after adding a block to the blockchain. This is possible since the ordering service ensures that it is not possible for different forks of the blockchain to occur.

2.2.3 *Chaincode.*

Chaincode refers to packages of smart contracts in the Hyperledger Fabric framework. In order to deploy chaincode on a channel, the code is packaged, signed and installed on the relevant peers who, in turn, sign the package on success and commit the chaincode deployment to a block on the blockchain in the same way as a transaction is committed.

Determining which peers must sign a chaincode package before it is committed to a block is handled separately from the consensus protocol for the blockchain and must be defined for each chaincode package individually.

2.2.4 *Channel.*

In simple terms, channels are sub-networks on the main Fabric network and each may contain their own ledger and individual configurations. Only members of a channel may write to the blockchain according to their permissions on that channel.

2.2.5 *Organization.*

Organizations are units that may encompass a number of peers and applications. An organization's identity may be used in a channel policy to determine the rights afforded to the peers in that organization.

2.2.6 *Peer.*

Peers are units that may be members of one or more channels and one organization. This is the unit on the network that hosts instances of ledgers and chaincode. Peers are the only units that contain information about the ledger and chaincode on a network, so any unit that needs access to these resources does so through interaction with peers.

Upon joining a channel, a peer is given access control rights according to its identity, as defined in the channel's policy.

2.2.7 Application. Applications interact with the blockchain network to perform tasks on channels. These transactions are done through ordered interaction with peers and orderers who in turn handle the logic for chaincode and administration of a network or channel.

2.3 Hyperledger Fabric vs Hyperledger Iroha

Hyperledger is a collaborative project for supporting open source blockchain frameworks for everyone that is in need of developing applications and services that require blockchain technology. There are plenty of options to chose from depending on the requirements of your network. The distributed ledger frameworks include Besu, Burrow, Fabric, Indy, Iroha and many more. The Hyperledger family tree is shown in the Figure 1.

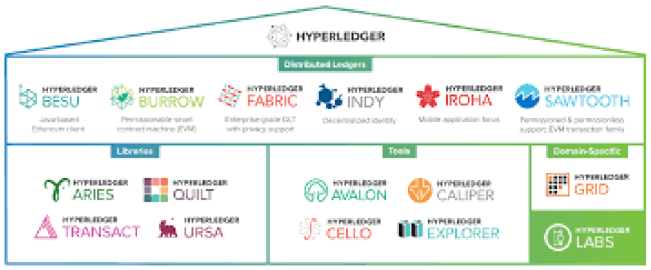

Figure 1: Hyperledger Family Tree. Image is from hyperledger.org.

For our project we needed a framework that features a general-purpose, simple and open-source framework that supports smart contracts and is aimed at developing permissioned applications. Although Fabric was the clear choice for our work, we took the liberty and time in exploring Hyperledger Iroha and its advantages. Iroha has a simple architecture with C++ design and it comes with the YAC [14] consensus algorithm. A more detailed comparison of all the Hyperledger frameworks was done by the team at FPT University [20].

2.3.1 Smart Contracts. In Fabric a smart contract is called a chaincode. An endorsement policy which plays a big part in the validation of transactions is determined by the chaincode. In the situation when a client sends a transaction, a transaction is being executed by specific peers and the output is stored. After that, transaction goes into ordering phase which uses consensus protocol to construct ordered sequence of the transactions that later will be grouped in blocks. Apart from that Iroha also provides a batch of transactions that allows sending its peers several transactions at once while still preserving the order.

2.3.2 Blocks. Blocks in both the Fabric and the Iroha networks are broadcasted to all the peers that are part of the channel. Blocks consist of the Header, Data and MetaData. Blocks are also signed with the cryptographic signatures by the peers for validation.

2.3.3 Consensus. Raft algorithm is responsible for consensus in the Fabric network. The protocol is used to keep all transactions on the blockchain ordered, and requires more than half of orderers to be active.[3] As already mentioned above the YAC consensus algorithm is implemented by the Iroha framework which basically performs ordering and consensus.

2.3.4 Nodes. Both Iroha and Fabric have the same set of nodes in their network. Expect that Fabric network is permissioned so all the nodes need to have an Identity provided by the Member Service Provider (MSP). There can be three kinds of nodes:

- **Clients.** In Fabric peers can submit transactions proposals for execution. On the other hand in Iroha clients have more power. They can query the data while also performing state-changing action or transaction.
- **Peers.** In Fabric peers are responsible for executing transactions proposals and validating them. In the Iroha network a peer has an address and maintains a copy of the current ledger.
- **Ordering Service.** In both frameworks the ordering service establishes the order of the transactions.

After investigating the differences between frameworks that are provided by the Hyperledger open-source projects, the best fitting option was Hyperledger Fabric. It provides permissioned application development with the implementation of smart contracts while maintaining a simple and modular architecture. While being one of the most developed framework, Fabric also has a lot of documentation which helped in setting up the environment and understand the underlying logic of the technology.

3 METHODOLOGY

3.1 Prerequisites

Hyperledger Fabric framework was deployed on the network using Raspberry Pi 3 devices. The Raspberry Pi is a very cheap computational device that operates in Linux environment and provides a great opportunity to explore Internet of Things (IoT).

Our initial setup involved:installing an Ubuntu 18.04 operating system (version for Raspberry Pi ARM architecture), configuring wireless interfaces for establishing SSH connection and establishing a VPN tunnel between our devices. Because this project was done during the spread of COVID-19, Raspberry Pis were setup in different places and were connected using *ZeroTier* VPN [5]. *ZeroTier* using certificates to control access to virtual networks and all the traffic between peers are encrypted using end-to-end keys. The implementation was very simple since it didn't require to manually generate certificates and import them into Raspberry Pis. VPN implementation process consists of installing *ZeroTier* service, creating and joining the VPN network with provided ID.

3.2 Installation

Hyperledger Fabric was built using *Go* progamming language. The version of *Go* that we used with our system was crucial since some older versions had issues and error that could not be dealt with. Furthermore, *Docker*, *Docker Compose* and *Docker Swarm* needed to be

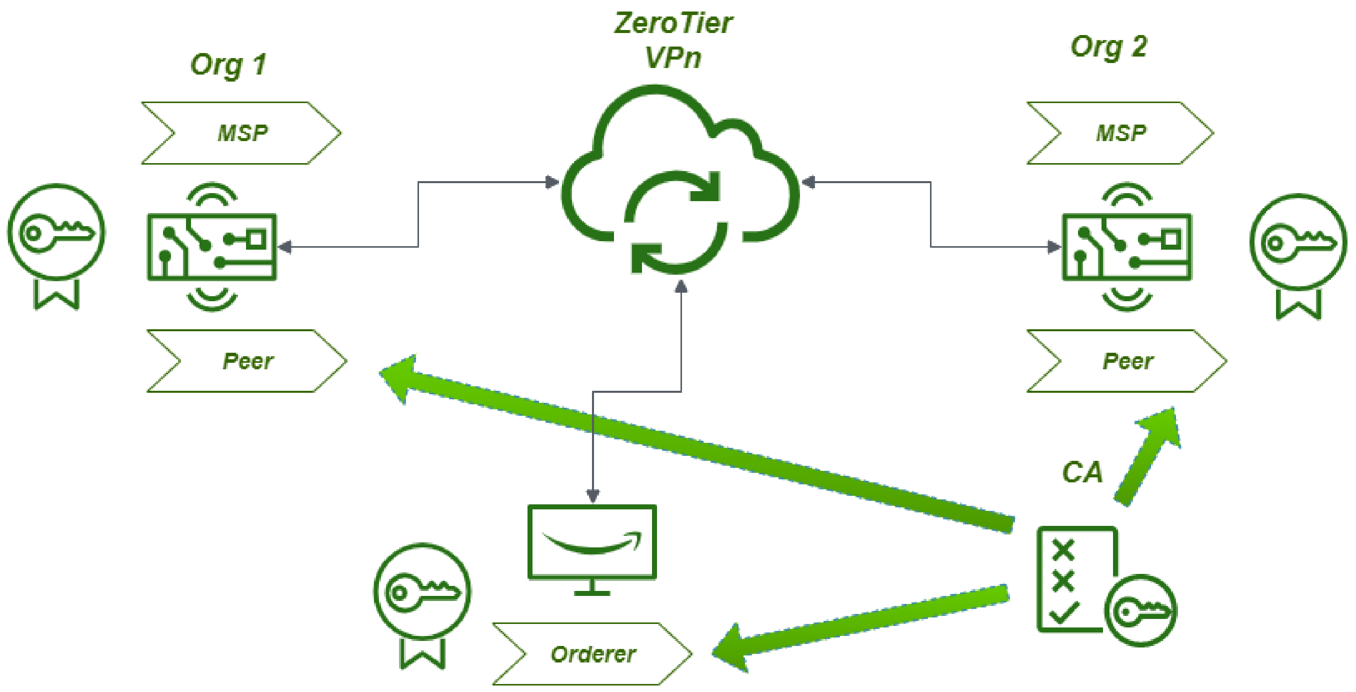

Figure 2: Custom network design with two peers on Raspberry Pi's and one orderer on a computer.

installed. With *Docker Compose* you can run multiple containers at once that work in sync and *Docker Swarm* groups multiple physical devices into one virtual cluster. In order to deploy the blockchain network we needed to download compiled binaries on github [11]. The repository included docker images for AArch64/ARM64 build for use with the Raspberry Pi 2/3/4 ARM which was the crucial part of the installation. It also included configuration files for setting up the network and various test scenarios for understanding the concept Hyperledger Fabric logic.

3.3 Deploying Fabric Network

To deploy the Fabric network we have used scripts that were provided in the repository which was downloaded earlier. It should be noted that scripts generated an environment which was set up for educational purposes to teach developers about smart contracts and blockchain applications. This is why it was a perfect fit for our network as it provided the tools and the documentation to experiment with. Using the *Docker* images we brought up the network.

Our initial Fabric network consisted of two peer nodes that were operating in different organization and one ordering node. See Figure 2 for network design. After the nodes were setup, private layer of communication was needed for the nodes of the network to communicate. We created a Fabric channel for transactions between our organizations. It is important to know that channels can be used only by organizations that are invited to join that particular channel. After the channel is created, the nodes of the network need smart contracts to start interacting with the ledger.

3.4 Deploying Chaincode on the Network

Hyperledger Fabric provides several APIs for the development of Chaincode in Go, Node.js and Java. In order to create a simple test network, we write chaincode that initializes with a set amount of assets and a framework for authorized members to query and transact these resources.

The chaincode created provides each of the raspberry pi:s with the ability to query values on the blockchain.

We also provide an admin peer that exists on one of our computers the rights to edit chaincode on members of the channel. This emulates the ability for a peer to edit configurations of IoT devices. This peer is also the only one that is permitted to perform transactions on the ledger.

3.5 Test applications

Applications may be developed for Hyperledger Fabric 2.0 in Node.js or Java. SDKs for Python and Go exist, but do not support Fabric 1.4+. We write simple applications that perform actions using each of our peers.

On each of the Pi:s, applications query the ledger for values at set intervals and react by printing to the console when the ledger reaches a certain state. This will verify that each Pi has been given appropriate permissions on the network and can access the state of the ledger.

The admin peer has an application that modifies values on the ledger. The admin peer uses chaincode that it has deployed on the blockchain to modify these values on each of the Raspberry Pi:s. Upon successful commits to the ledger, the Raspberry Pi:s should respond by outputting values to the console once their threshold for ledger state has been met.

3.6 Experimentation

3.6.1 Adding an Organization/Peer to the Fabric Network. Adding an organization or peer to the network requires a member to have Admin privileges on the network. As a test, we have an admin create a peer that belongs to its own organization on the main channel of the test network. This peer has minimal rights and in turn attempts to perform tasks for which it is unauthorized.

3.6.2 Adding and removing access control. Access control is defined in a configuration file for each channel that is created on the network. This file determines the rights given to different roles on the network, with default roles such as Reader, Writer and Admin being defined. The channel admin makes edits to access control on the channel and the result is noted for the affected peer.

3.6.3 Deploying and querying chaincode. Using the method described in section 3.5, we deploy, edit and invoke chaincode on the network to ensure that the chaincode is installed only on the relevant peers and that only the chaincode that the admin peer deploys is validated by the Raspberry Pi peers.

4 DISCUSSION

4.1 Literature Review

Much of the literature that has been written on the topic of blockchain have been written very recently, and the topic itself is very broad in scope, even when focused on the security of a blockchain network on IoT.

4.2 Security concerns in IoT

It is very important to note that although blockchain is an innovative solution to our current centralized network, it still has some issues regarding security measures. There is still a wide range of attacks that can be performed on a blockchain-based IoT systems. Researchers at a Swinburne University of Technology discussed and showcased different techniques for avoiding security measures of blockchain [21].

- ***Sybil attacks*** It describes a situation when adversaries produce a large amount of fake IoT user nodes and then tries to impact the blockchain network with the majority of peers. It is worth noting that this attack inflict a huge privacy leakage once the network gets taken over.

- ***Message Spoofing*** In a blockchain environment message-spoofing is an attack when a intruder is broadcasting fake messages in the network to lower security, privacy or efficiency of the network [10].

- ***Linking attacks*** Linking attacks are more associated with the data that are on the blockchain network. Third party operators try to use the external data that is linked with the protected data within the network in order to infer personal information about the node in the blockchain network.

4.3 Storage concerns in IoT

Deploying a blockchain on an IoT network presents difficulties due to the unique limitations of embedded devices.

Storage is a main concern in this regard since the entirety of the blockchain must be stored on each device that communicates with the network for the purpose of validating transactions. According to an IBM white paper on the subject, when expecting a feasible 100 transactions per second, each peer is expected to require over 315TB of storage per year. [8]

Block archiving is a possible solution to this problem, where validated segments of the blockchain are stored on a remote repository while devices utilizing the blockchain store the latest blocks of the blockchain that are still useful for validation. An implementation of this solution in Fabric is underway, and is specifically constructed with IoT in mind.[1] However, this solution is not entirely feasible as of yet in part due to issues in how peers validate blocks - block sizes may differ between peers due to block metadata.[2] The solution may also introduce some latency when querying data history on the blockchain, but has not yet been tested.

The aforementioned solution also limits the desirable property of decentralization since a storage device that contains the archived blocks is required, and while many copies may still be available over the network, this is similar to centralization of data at data centers.

4.4 Performance Evaluation and Comparison with Raspberry Pi Zero

In order to understand the workload for an IoT network implemented with blockchain technology, we first need to acknowledge the ecosystem of the IoT. According to the [6], the ecosystem consists of:

- The hardware components of the interconnected devices with the appropriate gateways.

- Connectivity between the network nodes and with the outside Internet.

- The deployed services on the network and running applications.

It can be noted, that depending on the domain on which the IoT devices specialize in, nodes in the network can a share a huge amounts of data. In these scenarios, the network has to deal with latency issues, congestion and consumption problems.

During the development of our network, we did ran into a few latency problems with Raspberry Pi 3 which was expected. It did not cause any major problems nor malfunctions which would nominate the device for the usage while experimenting blockchain technology in an IoT network.

Although Raspberry Pi 3 could be also described as a real computer due to its computational power, Raspberry Pi Zero is a perfect example of an IoT network component. In our project, we also wanted to research what power and computational limitations there are for blockchain deployment. A study on performance evaluation of Raspberry Pi Zero W [13] showed that in a case you use Raspberry Pi Zero as a IoT Gateway for blockchain service, there are no crucial limits of the hardware and nature of the device. The tests were performed in order to increase the temperature and CPU usage but it was not sufficient enough to damage the device or to disable a running service.

This would indicate that blockchain technology is a plausible security implementation to IoT networks which could improve the security features without consuming the majority of the network traffic.

4.5 Practical Work Review

Due to the COVID-19 pandemic, the scope of the project expanded beyond simply setting up a network for testing. Setting up the test network required the use of a VPN, and some issues that arose on one of the Raspberry Pi:s were not present on the other, likely due to each person involved having experimented with their individual Raspberry Pi.

Using the Raspberry Pi:s necessitated the establishment of a Docker swarm for ease of deployment and consistent network definition across all devices. Once this is set up, however, deploying the network and having members interact with one another is relatively simple since connection between devices is handled automatically once a container is deployed on a remote device.

Once devices are connected and permissions are distributed, it is possible to deploy chaincode, but each device that needs to utilize chaincode simply validates on a successful install. A misconfigured policy for a channel may open up the channel for attack if an attacker can coerce a vulnerable device to run an application that changes values on the ledger. But the nature of blockchain technology makes adding bogus data to the blockchain from a device not authorized on the channel relatively difficult at scale. As such, performing transactions that edit the shape of the channel by adding organizations, peers, or chaincode is relatively difficult.

On the other hand, since Fabric peers store the world state in a database that is unencrypted by default, it is simple to determine the state of the ledger at any time relative to a ledger that is simply stored as a series of transactions on a blockchain.

4.6 Challenges and Error Handling

Many of the issues that held back the development of this project related to the ARM64 architecture of the Raspberry Pi:s. The method used for setting up test network using Fabric relied on Docker images, but there were no images that were compiled for the ARM64 architecture, and we therefore relied on unofficial binaries deployed to the Docker Hub. This meant changing many values in configuration files for the test network before we could progress in writing our own tests. Roughly half the time for this project was devoted to fixing issues such as this through trial-and-error.

We chose to use the Fabric framework due to it being well adapted to the task of setting up an IoT network and relatively modular. However, the decision to use the latest version of Fabric led to its own share of hurdles since there were relatively few guides to setting up a network on Raspberry Pi:s using this version. We "reinvented the wheel" when getting adjusted to managing the network since using examples from these guides was impossible in many cases. Had we chosen to use Fabric 1.4 instead of 2.0, we would have saved a great deal of time in setting up networks and troubleshooting issues that related to misconfigured config files.

Overall, the scope of our initial plan for this project was well beyond what we had time to accomplish in the short time allowed for this project. Since both parties involved in the development of these tests were entirely unfamiliar with development of blockchain networks, reducing the scope of the project to a literature review of blockchain on IoT and instead simulating such a network using Docker containers on a single device running an x64 architecture would have resolved issues so that more time could be spent on testing security on the network such as sybil attacks and methods for enhancing confidentiality of the ledger.

5 CONCLUSIONS

In this paper our aim was to discuss the possibilities of using Hyperledger Fabric framework for securing and keeping the privacy of a blockchain-based Internet of Things network. The main reason why we chose to work with Fabric framework is that it is private and permissioned. Rather than an open permissionless environment that is free to anonymous identities to take part in the network, the nodes of the Fabric network can join the network through a trusted MSP. We mainly focused on understanding the underlying technology of Fabric framework and the principals used for developing blockchain network in IoT. We have also managed to deploy a successful blockchain network in an IoT environment and tested features like smart contracts implementation, channel configuration and privacy validation.

Among the discussed implementations of blockchain technology, we have reviewed some of the possible security threats that surround the innovative technology and its functionalities. However because of the nature of the blockchain technology and its architecture the attacks mentioned in paper would be difficult to execute.

Some issues as relates to storage and computation persist without sufficient solutions when deploying a blockchain on an IoT network. For example, while working with this project, the storage capacity of each Pi was 32GB. Though storage capacity may increase in future for embedded devices, it is infeasible to expect a full blockchain to be stored on each device at scale when the size can grow to the order of hundreds of TB.

6 FUTURE AREAS OF STUDY

This paper aimed to develop a reliable method for deploying a network over distributed devices while focusing on areas of improvement for future development of the Hyperledger Fabric framework. In future, it would likely be beneficial to extend this test network to incorporate more complex network configurations as well as modelling different application areas such as management of industrial devices and smart homes.

As one of the main bottlenecks of IoT devices is the limited computing power and storage capacity, methods for archiving old blockchain data without potential loss of integrity should be analyzed. In addition, practical application of various consensus protocols on the Fabric network could provide valuable insight into how they effect performance on a scaling permissioned network using benchmark testing against the default consensus protocol.

REFERENCES

[1] [n.d.]. https://github.com/hyperledger-labs/fabric-block-archiving

[2] [n.d.]. https://lists.hyperledger.org/g/fabric/topic/58600016

[3] 2020. Raft Consensus Algorithm. https://raft.github.io/.

[4] 2020. What's new in Hyperledger Fabric v2.x.

[5] 2020. ZeroTier VPN. https://www.zerotier.com.

[6] Serge Autexier, J. Campbell, J. Rubio, V. Sorge, M. Suzuki, and Freek Wiedijk. 2008. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface. 5144 (01 2008), V–VI.

[7] Nicolas Courtois. 2014. On The Longest Chain Rule and Programmed Self-Destruction of Crypto Currencies. (05 2014).

[8] Steve Elliott. 2018. Storage requirements for blockchain applications. *IBM* (April 2018). Accessed: 2020-04-03.

[9] Jake Frankenfield. 2019. Permissioned Blockchains. https://www.investopedia.com/terms/p/permissioned-blockchains.asp Accessed: 2020-05-04.

[10] Christoph Günther. 2014. A Survey of Spoofing and Counter-Measures. *Navigation* 61 (09 2014). https://doi.org/10.1002/navi.65

[11] Johan Hedlin. 2020. Hyperledger Fabric binaries for AArch64/ARM64 (Raspberry Pi 2/3/4). https://github.com/busan15/fabric-binaries-pi?fbclid=IwAR1M21pAY43POQ7p0PobMDe89ajnP3Zs_PTSpfjO9yMZVtWhmegEGR2XMDc.

[12] S. Huh, S. Cho, and S. Kim. 2017. Managing IoT devices using blockchain platform.. In *International Conference on Advanced Communication Technology, ICACT*. ETRI, 464–467. https://login.e.bibl.liu.se/login?url=https://search.ebscohost.com/login.aspx?direct=true&AuthType=ip,uid&db=edselc&AN=edselc.2-52.0-85018510486&lang=sv&site=eds-live&scope=site

[13] D. B. C. Lima, R. M. B. da Silva Lima, D. de Farias Medeiros, R. I. S. Pereira, C. P. de Souza, and O. Baiocchi. 2019. A Performance Evaluation of Raspberry Pi Zero W Based Gateway Running MQTT Broker for IoT. In *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. 0076–0081.

[14] Fedor Muratov, Andrei Lebedev, Nikolai Iushkevich, Bulat Nasrulin, and Makoto Takemiya. 2018. YAC: BFT Consensus Algorithm for Blockchain.

[15] P. Porambage, A. Manzoor, M. Liyanage, A. Gurtov, and M. Ylianttila. 2019. Managing Mobile Relays for Secure E2E Connectivity of Low-Power IoT Devices. In *2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC)*. 1–7.

[16] Archana Rajakaruna, Ahsan Manzoor, Pawani Porambage, Madhusanka Liyanage, Mika Ylianttila, and Andrei Gurtov. 2018. Enabling End-to-End Secure Connectivity for Low-Power IoT Devices with UAVs. (2018). https://login.e.bibl.liu.se/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edsarx&AN=edsarx.1811.04283&site=eds-live&scope=site

[17] Mehrdad Salimitari and Mainak Chatterjee. 2018. A Survey on Consensus Protocols in Blockchain for IoT Networks. (2018). https://login.e.bibl.liu.se/login?url=https://search.ebscohost.com/login.aspx?direct=true&AuthType=ip,uid&db=edsarx&AN=edsarx.1809.05613&lang=sv&site=eds-live&scope=site

[18] Mayra Samaniego and Ralph Deters. 2016. Blockchain as a Service for IoT. 433–436. https://doi.org/10.1109/iThings-GreenCom-CPSCom-SmartData.2016.102

[19] Caner Taçoğlu. [n.d.]. Immutability. https://www.binance.vision/glossary/immutability Accessed: 2020-05-04.

[20] Ban Tran Quy, Bui Anh, Ngo Son, and Tran Dinh. 2019. Survey of Hyperledger Blockchain Frameworks: Case Study in FPT University's Cryptocurrency Wallets. *ICSCA '19: Proceedings of the 2019 8th International Conference on Software and Computer Applications*, 472–480. https://doi.org/10.1145/3316615.3316671

[21] Muneeb Ul Hassan, Mubashir Husain Rehmani, and Jinjun Chen. 2019. Privacy preservation in blockchain based IoT systems: Integration issues, prospects, challenges, and future research directions. *Future Generation Computer Systems* 97 (03 2019). https://doi.org/10.1016/j.future.2019.02.060

[22] I. Yaqoob, E. Ahmed, I. A. T. Hashem, A. I. A. Ahmed, A. Gani, M. Imran, and M. Guizani. 2017. Internet of Things Architecture: Recent Advances, Taxonomy, Requirements, and Open Challenges. *IEEE Wireless Communications* 24, 3 (2017), 10–16.