SmartChain: A Blockchain-based user authentication system for smart homes

by

Mushaidul Islam
18201030
Mohammed Sayeem Sadat Hossain
18201156
Shijon Das
18241009

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
January 2022

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Mushaidul Islam

Mushaidul Islam
18201030

Sayeem Sadat

Mohammed Sayeem Sadat Hossain
18201156

Shijon Das

Shijon Das
18241009

Approval

The thesis/project titled "SmartChain: A Blockchain-based user authentication system for smart homes" submitted by

1. Mushaidul Islam (18201030)

2. Mohammed Sayeem Sadat Hossain (18201156)

3. Shijon Das (18241009)

Of Fall, 2021 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 16, 2022.

Examining Committee:

Supervisor:
(Member)

Zavid Parvez
Digitally signed by Zavid Parvez
DN: cn=Zavid Parvez, o=Brac University, ou=CSE, email=zavid.parvez@bracu.ac.bd, c=US
Date: 2022.01.19 09:51:08 +11'00'

Dr. Mohammad Zavid Parvez
Assistant Professor
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam
Associate Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Dr. Sadia Hamid Kazi
Associate Professor and Chairperson
Department of Computer Science and Engineering
BRAC University

Abstract

Smart houses are more about being environmentally conscious and offering security. Smart home technologies can make homes more energy-efficient and allow users to save money. This concept of making the life of the common people easier has increased the demand for smart homes. However, the rising demand also calls for security concerns. Many approaches have been taken for enhancing the security of IoT installation, but most of the issues concerned with its protection are still left to deal with. On the other hand, the rise of the popularity of Blockchain, because of the security of data it ensures, has made it a potential component to pair up with IoT. Along with the many benefits of Blockchain integration, it also ensures the decentralization of data. In this paper, we aim to discuss the need for Blockchain-based IoT and its limitations. Further, along with the paper, we consider the past implementations. To support our work, we proposed a model that enhances security in smart homes using blockchain by verifying the incoming requests and giving access to only authorized users.

Keywords: Smart home; IoT; Blockchain; Consensus protocol

Dedication

All praise to the Almighty Allah for keeping us safe during the global pandemic. We dedicate our thesis to the late victims of COVID19. May their souls rest in peace.

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.
Secondly, to our Supervisor Dr. Mohammad Zavid Parvez Sir for his kind support and advice in our work. He helped us whenever we needed help.
And finally to our parents without their throughout support it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

ABI	Application Binary Interface
ACL	Access Control List
API	Application Programming Interface
dApps	Decentralized Apps
$DDoS$	Distributed Denial of Service
DoS	Denial of Service
EVM	Ethereum Virtual Machine
$FOTA$	Firmware Over-the-Air
GPS	Global Positioning System
IDE	Integrated Development Environment
IoT	Internet of Things
IPC	Inter Process Communication
IT	Information Technology
$M2M$	Machine to Machine
$NTUA$	National Technical University of Athens
PoA	Proof of Authority
PoS	Proof of Stake
PoW	Proof of Work
SeN	Sensory Nodes
$SHA3$	Secured Hash Algorithm 3
SN	Super Node

Chapter 1

Introduction

The Internet of Things (IoT) is a generic term for an increasing number of electronic devices that connect to the Internet to send data, receive instructions, or both, rather than traditional computing devices [22]. The IoT plays an essential role in the implementation of smart home systems. A smart home, also known as a connected home or eHome, is a highly developed automated system [7]. The overall idea of a smart home is to make everyday life more accessible, and the IoT will help you realize that vision by connecting every device in your home to the Internet. However, as the existence of IoT objects and visibility from the Internet increases, security, i.e., access to authorized user resources, is a significant issue [12]. Here, the use of blockchain comes to play. A blockchain is essentially a digital ledger that is duplicated and distributed over the blockchain's network of computer systems. Each square in the chain contains countless exchanges. Each time another exchange happens in the blockchain, a record of that exchange is added to every member's catalog [17]. When implemented decentralized, storing information in the blockchain makes the data an invariant timeline.

1.1 Motivation

Nowadays, the process by which the data requester collects user data from IoT devices lacks transparency. It creates space for suspicion as the legitimate user has no part in the access control of how the data will be shared with the requester [18]. Most IoT implementations include a cloud-based system where all the sensitive data are stored in one centralized location, making it a more accessible target for a potential hacker. To make matters worse, IoT appliances have finite computational and memory capabilities, making them more vulnerable to cyber-attacks [21].

The matter of security cannot be mitigated in the case of smart homes, as tampering with the home devices by a third party can lead to life-threatening scenarios. Blockchain is a potential solution for the issues faced with the usual implementations because of its intrinsic security [12].

1.2 Problem Statement

The IoT-Internet of Things has taken a toll on our lives quite steadily and invisibly as smart devices and high-speed networks saw rapid growth. In 2017 worldwide, almost 10 billion connected things joined this network and are expected to reach nearly 20 billion by 2020 [24]. Henceforth, it is necessary to consider the massive number of IoT devices connected and ensure their orderly sharing of information and expected functionalities. Nevertheless, users incur an economic loss due to security and privacy issues and hamper the advancement in the IoT sector. As a result, safe and secure entrance control is regarded as one of the most critical technologies for ensuring IoT privacy and security [24]. Due to the IoT, the network environment of smart homes is considered an essential factor. Communications now are shifting from wired to wireless as the network structure of smart homes, consisting mainly of embedded computers, is connected to various IoT devices on the internet [23].

The gateway is responsible for controlling and monitoring the communication of multiple devices, which the smart home collectively consists of. This network configuration is harmful as it can leak data of the house, breach privacy, malfunction devices, and harm users. When a user logs in to a smart home network that explores our model's demonstration and testing in a household, there is always a probability of data collected by devices being leaked [23]. Difficulties arise when we bring in different heterogeneous devices, but smart home and appliance security standards are missing, preventing the smooth handing over of various smart devices to users. Therefore, it can be easily inferred that security is vital for gateways. The security requirements for gateways in smart homes have been explained in the following [23].

Confidentiality: A smart home network is responsible for collecting and storing a wide range of data, including sensitive data from users. This data should be accessed by authorized users only and thus is an essential part of ensuring security in smart homes. The characteristics of smart homes are kept secret through the use of blockchains with cryptographic algorithms along with keys to configure them [23].

Solidarity: While sending and receiving data between individual configurations, no data should be corrupted to make the transmission successful. Hash functions reduce the chance of this data being corrupted and allow accurate tracking and validation of data records [23].

Authentication: The authentication function is used to prevent malicious attacks from an outsider who is not part of our smart home network. The network is verified using blockchain. The verification ability at specific times can enable the correct network configuration of smart homes [23].

1.3 Research Objective.

A blockchain allows the data in a database to be distributed across several network nodes in different places. This adds redundancy to the database and ensures that the data is accurate. If one node of the database is updated, the other nodes are not affected, preventing a bad actor from doing so. If one user tampers with Bitcoin's transaction record, all other nodes will cross-reference, making it easy to find the node with erroneous data. This system aids in the establishment of a precise and visible sequence of occurrences. In this manner, no one node in the network may change the data it contains.

Most of the current implementation involves a centralized IoT system, where a network of IoT is built around a central server to manage all the nodes in the system. This server handles requests from nodes and assigns roles to the nodes. However, this approach brings with it the risk of single-point failure. On the other hand, implementing a Blockchain system for the network of IoT will create a decentralized system that avoids such risks to arise. Thus, in this paper, we proposed a model that enhances security in smart homes using blockchain by verifying the incoming requests and giving access to only authorized users.

1.4 Thesis outline

Exploring a Blockchain-based IoT model which provides a decentralized network and handles authentication is the primary focus of our report.

The report starts with Chapter-1 - "Introduction" element, where we have discussed our motivation to choose this topic of research, the problem statement, and our research objective.

Next comes our Chapter-2, "Background" section, where we have vastly explained the IoT system, defined Blockchain, and provided applicable, related definitions of terms that we have used throughout the paper. Furthermore, we have discussed research papers written on issues similar to our topic.

In Chapter-3, "Proposed Model," we have talked about our proposed model and provided relevant diagrams to understand the model.

Later in Chapter-4, "Experimental Progress," we provided details of the technical arrangement we used in this research and explored our model's demonstration and testing.

We end the report with Chapter-5, "Result Analysis," where we analyzed our model in terms of performance, storage handling, and cost-benefit.

Chapter 2

Background

2.1 IoT

The term Internet of Things (IoT) refers to all the appliances in a network that are Internet-enabled. A real-life application to illustrate the term is whenever we enter our Credit/Debit Card info into the card machine while paying for any of our buyings, that machine is IoT-enabled. We are getting all the payments done through the Internet. Further examples can be thought of as our PCs or laptops or even mobile phones, but indeed IoTs are far broader than these examples. Any such device that can connect to the Internet or even send/receive data through the Internet, including those considered ordinary. With the advent of IoT, automation has become possible in almost every corner of our lives. Now, a device without Internet-enabled in it is considered one rarity.

According to [4], up until 2019, almost 9 billion devices were considered to be active IoT-enabled appliances. Still, the original numbers after 2019, according to the same paper, have been found out to be almost 10 billion. Even as the numbers speak for themselves, to further enhance it, in 2020, according to [4], it was found out that nearly 50 billion devices and appliances around the world are IoT-enabled. It is further estimated that the numbers will keep growing, and at one point, there will not be any such device without the capability of the Internet in it [4]. All these numbers prove how much the world is dependent on such devices. As a result, it becomes even more critical to ensure a proper security scheme with much data continuously accessed and retrieved in sent.

A high-end security and safety mechanism in an IoT Device is hence considered one of the most important design principles. Unfortunately, during primitive times, this security was not considered a mandatory objective while designing those devices. As a result, the data breach was a common problem in these appliances. Furthermore, IoT-enabled devices provide only the required basic functionalities that the users ask for. They are designed so that selling is prioritized more than their security. These devices are there only to meet basic needs such as temperature regulation, light regulation, etc. However, numerous incidents have occurred recently where security has failed to be strong enough to stop the data breaches. This forced the researchers and many personnel to bring changes to the built-in architecture of this divider to provide a more secure data handling mechanism, such as by providing different encryption formats. One example can be applying Lightweight Cryptography at the application layer, shown in Figure 2.1. This method works so that the data from the IoT-enabled devices are encrypted first before going to the Internet and then decrypted before going to the good receivers. Several other encryption algorithms have been developed to deal with such security issues in the IoT sector.

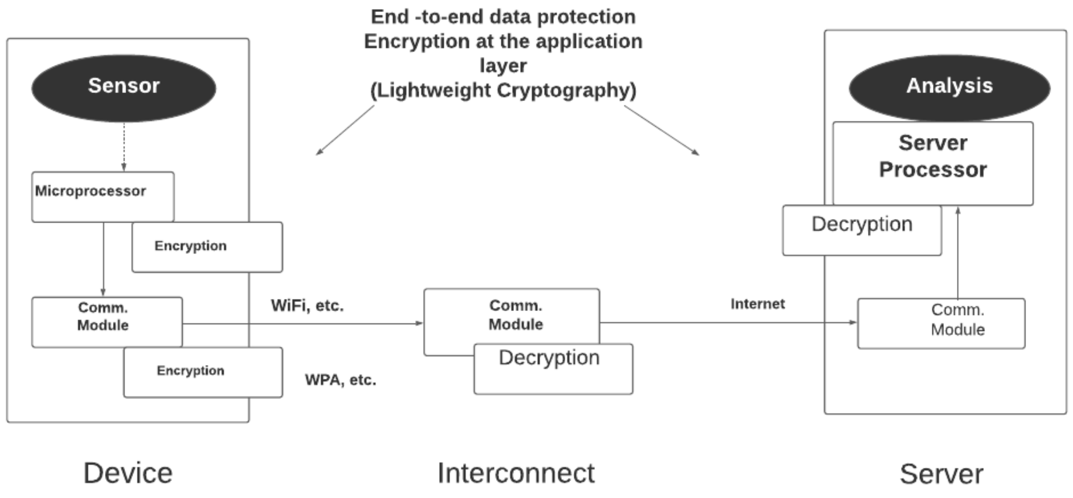


Figure 2.1: Lightweight Cryptography summarized

While developing a well-defended and secure mechanism in those devices, some things need to be kept reminded. The IoT-enabled devices are built with low resource space, low processing capabilities, and low computational power. Hence, an algorithm rich in resources cannot be implemented typically inside an IoT appliance. The security mechanism has to be lightweight. For this reason, ensuring a proper and secure IoT-enabled device is still considered as one of the challenges the researchers face during recent times.

The matter of security cannot be mitigated in the case of smart homes, as tampering with the home devices by a third party can lead to life-threatening scenarios. Blockchain is a potential solution for the issues faced with the usual implementations because of its intrinsic security [12].

2.2 Blockchain

A blockchain can be considered as a database that is distributed and append-only. This database is shared between nodes existing in a network. Information is stored in the database by blockchain in digital format. Blockchain eliminates the requirements of a trusted authority as non - trusting members can easily communicate in the distributed network. Blockchains are recognized for their crucial role in maintaining, securing, mining bitcoins and providing decentralized servers to keep track of transactions. Blockchain is innovative because it ensures the security and fidelity of data without a trusted third party.

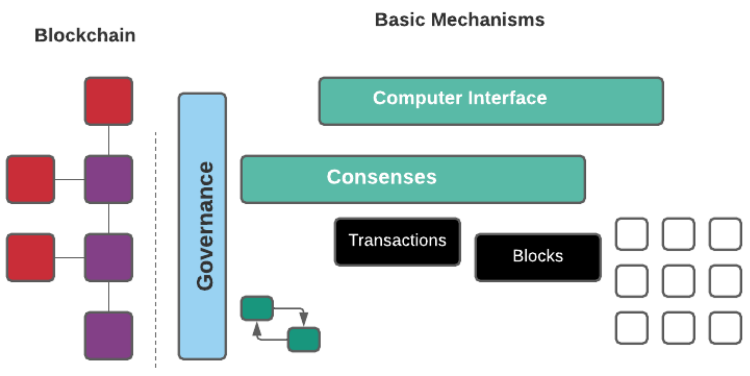


Figure 2.2: An Overview of Blockchain Architecture

The data structure of blockchain differentiates itself from a typical database. In the blockchain, the blocks contain data collected in groups [8]. These blocks come in a specific size. When the size is filled, the blocks do not take in information anymore and are closed. This closed block is added to the blocks that got filled up before, resulting in a blockchain chain. Any information will be added to another block that will also get added to the chain of blocks once it runs out of capacity.

In a typical database, we observe that data is structured into tables, unlike in blockchain, where the information is stored into blocks joined together using a chain. An immutable timeline of data is formed upon implementation in a decentralized server. Each block contains an exact timestamp of when it got added to the system once filled.

We have two forms of blockchain available currently. They are public and private blockchains.

As the name suggests, a public blockchain is public; it is open to everyone as it is permissionless. Anyone over the internet can participate in the network. No single user controls this network as the system is decentralized. Even though it is open to everyone, it is quite secure as the blocks are immutable once validated. An example of a public blockchain is Bitcoin.

On the other hand, a private blockchain is limited as it requires permission. It restricts the participation of users through permission [8]. The network is under the control of one or two users and hence requires the support of third parties to complete the transaction. Since it restricts users, only the nodes participating will have access to the data in the blocks. An example of a private blockchain would be the Hyperledger Fabric of Linux.

2.3 Related Definitions

- *Application Binary Interface (ABI):* When a third-party application or smart contract wishes to communicate with the blockchain, it must understand the smart contract's interface, such as identifying a method and its parameters. The Ethereum Application Binary Interface makes this possible (ABI). ABI is a program interface that connects two software modules, one of which is primarily machine code. The interface is the default approach for encoding/decoding data into or out of machine code.

- *Bitcoin:* This virtual currency runs independently of any central authority. Every bitcoin transaction is handled in a Blockchain network where the nodes achieve consensus cryptographically on who owns which coins.

- *Consensus:* A consensus algorithm is a method through which all peers in a Blockchain network achieve a consensus on the current state of the distributed ledger. Consensus algorithms perform blockchain network resilience and create trust amongst unknown peers in a distributed computing environment in this way. In essence, the consensus protocol ensures that every new block added to the Blockchain is the only version of the truth that all nodes in the Blockchain agree on.

- *dApps (decentralized apps):* The distinction between them and regular applications is that they reside and function on a blockchain or peer-to-peer network of computers rather than a single computer.

- *Embedded Software:* Is software embedded in hardware. GPS devices, smartwatches are just some of the examples of it. This software usually is used to compensate for the lack of computational ability of the devices that implement them.

- *Encryption:* Encryption refers to the technological process of converting plaintext to ciphertext and back, which secures data and systems and makes it difficult for unauthorized parties to access encrypted data.

- *Ethereum:* Ethereum is a Bitcoin-like blockchain computer application. It may be used to build automated contracts or distribute Ether, digital money. Ethereum is a multi-purpose platform that is more than just a cryptocurrency. Other coins and smart contracts are hosted on Ethereum.

- *Ethereum Account:* An Ethereum account combines an Ethereum address and its private key. Furthermore, an Ethereum account may store a balance (Ether) and send transactions. Accounts can be managed by users or used as smart contracts.

- *Ethererum Address:* The Ethereum address is a 64-character hex string produced according to a set of criteria. It is connected with a private key and represents a unique account on the Ethereum network. This private key is necessary to confirm address ownership and must be kept secure.

- *EVM:* EVM stands for Ethereum Virtual Machine. EVM is responsible for stating the guidelines for creating a new valid state from block to block, as Ethereum has only one acceptable form at any particular instance.

- *Firmware Over-the-Air (FOTA):* This technique allows operators of Internet-connected devices to update their firmware versions remotely and effortlessly without physically accessing the device. It's critical to update the operating system of linked assets to keep them safe, introduce new features, and correct errors.

- *Genesis block:* The initial block of any blockchain system is a Genesis Block. The Genesis Block, also known as Block 0, is the first block in a blockchain upon which subsequent blocks are placed. Because every block refers to the one before it, it is the ancestor to whom all subsequent blocks may trace their genealogy.

- *Inter-process communication:* This communication and synchronization mechanism allows processes to communicate. Communication between these processes might be viewed as a way to work together.

- *Machine to Machine:* Also referred to as M2M, is a general term that refers to any technology that allows networked devices to communicate data and conduct operations without human intervention.

- *Miners:* Mining is a vital component of the maintenance and growth of the blockchain ledger since it is how a blockchain network confirms new transactions. "Mining" is done with high-tech apparatus that solves a very difficult computational arithmetic problem. Miners are the entities that carry out this procedure.

- *Nodes:* A blockchain node is a cross-platform, open-source runtime that allows developers to build various applications. The P2P protocol lets nodes connect and share information about transactions and new blocks. The correctness and dependability of storing the entered data in the distributed ledger is the responsibility of network nodes. A complete copy of the distributed ledger may be stored on each node. Thanks to the blockchain nodes, any user may access the data and examine all transactions done or saved on the network.

- *Remix:* Remix IDE is an open-source program that allows you to build Solidity contracts directly from your browser.

- *Smart Contracts:* Smart contracts, maintained on a blockchain network, are automated executable programs triggered when a particular agreement has been made. Their involvement provides confidence to all the parties involved in a transaction and avoids wastage of time.

- *Solidity:* Solidity is a high-level object-oriented language for creating smart contracts.

- *Token:* A token is a unit of value that blockchain-based businesses or programs established on top of current blockchain networks.

2.4 Literature Review

In [1], the three primary layers of IoT are described: Application layer, Network layer, and Physical layer. This paper sheds light on the security risks and internet attacks that could take place in an IoT ecosystem. Due to the presence of the low power devices, the protocols of the TCP/IP Stack can not be applied to an IoT system despite having homogenous architecture. Physical layer attacks include but are not limited to Physical damage, Node jamming, Malicious node injection, Node Tampering, RF interface, Sleep deprivation, Social engineering, etc. Likewise, in the network layer, we have DOS, Sinkhole attack, RFID spoofing, Sybil, Man in the middle attack, RFID cloning, DDOS, Sybil attacks, etc. The application layer consists of Spywares, Malware, Adware, DOS, viruses and worms, Trojan horse, etc. They provided a few layer-specific protections to secure the layers from the threats mentioned here. Physical layer is kept secure with the help of fast booting of IoT devices using cryptographic hash algorithms, authenticating devices with the support of less power consumption, securing data before transmission where data of each device is encrypted, and anonymity where sensitive data like identity and location of nodes is kept anonymous. The network layer uses authentication mechanisms and point-to-point encryption to prevent access to the sensor nodes for counter-attacking incoming threats. In this way, the privacy of data is maintained. Furthermore, multiple paths ensure the information is routed securely, enhancing the system's error detection ability and performing even when a failure is detected. Finally, the application layer uses Firewalls, protective software like antivirus or anti-adware to counter threats and ACLs. However, to ensure that an IoT system functions smoothly, it is mandatory to continue applying patches and updates, upgrading systems, providing improvements, monitoring devices, finding threats, using IDS (Intrusion Detection System), Trust management, Securing IoT physical premises, etc.

This research paper [14] surveys threats that could potentially damage sensors of IoT devices and applications. The survey paper is divided into four sections where; the first section provides a detailed discussion about the sensor management systems used in the operating systems of the numerous IoT devices and comprehends the drawbacks of the current systems. In contrast, the second section gives a detailed study of sensor-based threats. IoT devices' existing security solutions and their weaknesses for sensor-based threats are analyzed in the third section. Finally, the fourth section discusses further research scopes that enhance IoT devices and applications' security. The sensing layer is responsible for obtaining data from the real world and picking patterns from the devices' surroundings and consists of several sensors amongst which motion sensors gauge the change in the two types of motions - linear and angular activity - and also account for any difference in devices' orientation. In contrast, environmental sensors consisting of Light sensors, Pressure sensors, etc., are inserted in IoT devices to sense any changes in the ecological parameters of the devices' surroundings. The devices' position sensors handle the physical location and position. IoT devices' sensors contain threats mainly classified into 4 major categories depending on their nature and purpose. The threats are (1) False Sensor Data Injection, (2) Transmitting Malicious Sensor Patterns or Commands, (3) Denial-of-Service, (4) Information Leakage. The most recurring sensor-based threat is Information Leakage, as sensors are prone to revealing sensitive information like passwords, a cryptographic system's secret keys, credit card PINs, etc., violating a user's privacy. It consists of a few sub-parts: Audio Sensors, Eavesdropping, Video Sensors, Motion Sensors, Keystroke Inference using Light Sensors, Power Analysis, Task Inference using, Location Inference and Magnetic Sensors. It is possible to pass on harmful sensor patterns or trigger commands for activating malware present in a victim's device using sensors. This transmission of malicious commands can occur via Light Sensors, Magnetic Sensors, Audio Sensors. In the False Sensor Data Injection, the sensor data is deliberately changed for performing malicious activities, which can be done through physical access of the device or various communication devices. Lastly, in a denial of service attack, the operation of a device is denied through the use of malware. This paper delved into a few current security procedures for preventing threats based upon sensors such as Protecting Sensed Data and Enhancing Existing Sensor Management Systems along with excellent opportunities for the further scope of their research, including Control Sharing of Data among Sensors, Protect Integrity of Sensor Operations, Study of Expected Functionality to Identify Threats, Protect Sensor Data when at Rest, Adoption of Intrusion Mechanisms to Detect Attacks and Prevent Leakage of Secret Data.

This paper [15] guided us on how blockchain technology can make a secure and trustworthy IoT model as blockchain is revolutionizing IoT security. The four pillars of blockchain in enhancing IoT security. First is the consensus responsible for providing the proof of work (PoW) and confirming the activities in the networks. Second comes the ledger, which contains the details of transactions within networks. At the same time, the third pillar, Cryptography, ensures all information in the ledger and networks are encrypted and provides access to decrypt the information only to the authorized user. Lastly, the smart contract verifies and confirms the participants of the network. Several patterns are being used for implementing blockchain-based IoT. This paper defined three such ways. First is the communication model, where three primary functions of the blockchain network are applied: peer-to-peer messaging, distributed data sharing, and autonomous coordination with the device. However, this model is challenging to implement due to its drawbacks, such as slow processing, since blockchain networks need high configuration CPU and memory to function correctly as the size of the ledger increases with time. The second model discussed here connects multiple blockchain networks, and the third is Discovering legitimate IoT at a large scale.

In another paper [20], we came across techniques for securing our online transactions, vitality exchanging, digital money/cryptocurrency, industrial IoT, etc. While working over the internet, it is crucial to ensure network layer security as data security is necessary to safeguard a system from hackers. To achieve this, at first, the data is secured from unauthorized access and protected from hackers. This is done by storing all data in blocks linked together, maintaining a sequence to make a continuous chain. Here, blockchain is applied to safeguard transactional data. The existing blockchain nodes validate the blocks containing transactions.

However, these procedures contain specific challenges and limitations too. The energy and costs associated with it are very high, and there is a high chance of people not being familiar with blockchain as it is an emerging technology. Blockchain-based applications in IoT focus on the security and privacy issues the most since we know that the biggest concern for IoT is the centralization of the Server continuously interacting with the devices connected to it, for which multiple attacks by intermediary entities are possible [6]. Hence, the numerous proposed solutions to those problems, upsides, and challenges are discussed below. Most models use Ethereum based blockchain and smart contracts to develop a viable solution.

According to [9], one proposed solution to issues regarding IoT consists of implementing an Intrusion Detection System (IDS). Typically, an anomaly-based IDS senses all the threats possible by an unknown intruder in the devices. All the solutions that have been already proposed regarding IDS implementations as per [3], [5], [2], either ignore the straightforward fact that the training phases may have adversarial attacks (where a potential attacker can completely disguise themselves and continue with the attack after the execution) or some activities of the detection models can result into false positives (occurs when the anomaly detection is done under a small number of devices). As a result, Tomer et al. proposed a CIoTA (Collaborative IoT Anomaly Detection) system, which accounts for these attacks and false positives. It makes it a more robust design and lightweight [10]. In this model, this framework uses blockchain with an extension to using the Extensible Markov Model (EMM) to gradually update a trusted anomaly-detection model such as that of [3], [5], or [2] through self-attestation and the consensus protocols of the IoT Devices. However, the downfall of the model was that it needed a significant overhead since the code was not optimized enough, for which a greater utilization of the CPU (6.5%) and greater memory consumption was required (60 KB) according to [10].

In this research paper [16], the authors have seemingly considered the possible failures and limitations of a realistic IoT Architecture that they have developed using the existing technologies. They have reasonably used the Ethereum model of blockchain technology, suggesting that Ethereum can simultaneously create multiple copies of smart contracts. They build a vast scale event-based Internet Of Things (IoT) control system using that Ethereum model. Their architecture consists of a blockchain-based environment where a smart contract helps generate the events required by the Internet Of Things Devices or the nodes in the network, which also act as RPC Servers for the architecture. They have further included devices and passing gateways to control the Ethereum-blockchain-wallet [16]. They propose that a Client in their model does not have to talk to the IoT Devices directly; instead, their communications will be done through the blockchain. To summarize the phase of their proposed model, their architecture allows the clients to act as both the Nodes themselves in the Ethereum based blockchain system or can be networked to the RPC server via a full node separately. Every device's calls are considered and predefined in their approach as they are portrayed onto specific procedures. When a user asks for that particular operation, the function in that smart contract initiates the blockchain-based events to deal with that call [16]. Then, only the interested IoT gateways receive the possibilities and help provide them to the appropriate IoT Device to execute the operation. They have also used an access control mechanism to control which nodes can act as the client. Only authorized clients can ask for that specific operation and communicate with the IoT Devices intended [16]. This mechanism creates custom tokens to either use the functions balance or transfer defined in the ERC-20 token standard of Ethereum [16]. They have successfully implemented their proposed architecture using a private Ethereum based network, Rinkebyy and Ropsten Ethereum testnet, and some considerable APIs, but some drawbacks. The main ones were that Ethereum based networks are costly, and their transaction is somewhat delayed compared to other blockchain technologies.

As the world of the Internet Of Things grows, the need for IoT in several fields is also on a new rise. IoT has recently played an important role in one such intimidating area is providing emergency services for smart home systems (SHS). In the proposed model in the paper [19], Thitinan et al. developed a model where the system will send automated calls to the nearby emergency public services such as hospitals, fire departments, police offices, etc. As to the primary research domain, this paper also uses Ethereum based blockchain technology and its corresponding intelligent contract system, which helps manage a decentralized access control system among unauthorized public services. In the paper, the public services have been termed Home Service Providers (HSPs). In the proposed design, the authors have included Ethereum based miners for HSPs, Smart Contracts by Ethereum, and web-based applications and APIs for the homeowners (HO) and HSP staff [19]. Every SHS in their architecture consists of Raspberry Pi version 3 installed in the primary sensory manager helping in detecting the emergency calls by studying the surrounding environment and sending the calls to the respective HSP predefined by IPFS. The authors use Solidity smart contract and deploy the code to an Ethereum based private blockchain via Remix IDE [19]. Their evaluation and testing phase installed two Ethereum Miner nodes (EM1 and EM2) in Ubuntu OS with significant CPU and RAM differences. Their JSON ABI consists of the ETI functions setEmergencyType() and getEmergencyType(), which create awaiting transactions in their system and serve as the basis for the call and response emergency services. Their test successfully provided the intended results mentioned by the authors [19]. However, their large-scale architecture has failed to develop an efficient and cost-productive blockchain-based IoT environment. The system also lacked efficiency in access control mechanisms as DDoS attacks seemed viable. However, the authors are considering introducing some significant changes to their model, as they are indeed adding some features and further derivability [19]. They consider adding a safe and secure access control mechanism and introducing an Ethereum wallet for every charge, focusing on the proposed model's control and costs.

Peer To Peer (P2P) architecture is preferable, fast, robust, and advantageous to a Client-Server model in any network topology. The main reason behind this is the decentralization of the Servers in a P2P model, which makes it fast to upload/-download anything to/from the Servers saving the connected Devices from a single point of failure. However, the limited number of IPv4 addresses has madimplementingment a topology using the P2P model challenging. In this case, "NAT-Network Address Translation" helps manage the currently used network addresses, including the topologies having the P2P Architecture. Two significant methods which have been provided by the "IETF-Internet Engineering Task Force" to fix the passing of Datagrams through NAT-Translations include "STUN-Simple Traversal of UDP through NATs" and "TURN=Traversal Using Relays around NATs." Both methods are bound to some extent but are useful enough to help a topology carry out the P2P architecture. Elie et al. proposed one model in their paper [11], which uses the latter method, and TURN serves as relays to the Internet of Things devices connected via the blockchain-based platform. This model focuses on both IoT Device Types, i.e., Constrained and Non-Constrained devices. Constrained Devices have finite storage management, memory, and executional abilities staying on the back of a Border Router for connection with the Internet. At the same time, the latter are those IoT Devices that can be networked straight to the internet [11]. Their proposed architecture consists of a Wallet Management Function (WM), some TURN Servers, which are the central Relays of the Network as they are registered to the Blockchain, IoT Client-Module, and Smart Contract [11]. The purpose of the Wallet Management function is to transfer the Ether (an Ethereum based currency) on the connected device accepting/denying the ledger requests, while the job of a built-in IoT-Client module is to build an Etherum wallet and map a connected device successfully to a valid server through TURN, storing its corresponding generated address encrypted with the public key [11]. The Smart Contract in their system consists of the functions provoked to successfully connect the Clients to the TURN Servers for their relayed communications such as register server(IP), approved server(IP), etc. To implement their proposed model, they have used existing technologies such as Solidity to build and deploy smart contracts, NodeJS, and Client Module, as mentioned earlier [11]. To finally simulate and see the results of the Blockchain Network, they have used the Ropsten Testnet. After evaluating their results, it was concluded that their implementation could solve the NAT Traversal issue as proposed and provide an End-To-End (e2e) secured setup [11]. However, a detailed cost analysis was not provided, which may be a drawback for small-scale IoT Networks.

As the technologies advance, the need for a decentralized system using blockchain in the IoT sector keeps rising. Even though the connected peers are untrusted towards each other, they work in such a way that their transactions will be discarded without any proper authorization. One such paper that focuses on a Decentralized Application (DApp) is Georgios et al. [13]. Like the previous models, this paper uses Ethereum based blockchain to utilize their smart contracts during execution. The model specifically focuses on IoT weather sensors data and provides viable management regarding the buying and selling of their data, hence providing a hub of IoT sensor data [13]. The main idea behind their model is that it creates an Ethereum based blockchain DApp where the registered users can easily buy and sell any sensor data to use in their specific fields, whatever they need. While trading data, the currency mode can be any custom predefined token. In this case, a token named NTUA has been used for experimentation and this architecture for IoT weather sensors. Still, the authors have implemented the model to use any other sensor data [13]. The components of the DApp consist of two smart contracts, named NtuaToken and Broker, with discrete functionalities which are compiled, processed, and released in the blockchain mentioned above technology [13]. The job of the NtuaToken smart contract is to create an NTUA (National Technical University of Athens) type-token which is self-made and used to exchange the main sensor data information, while the main task of the latter smart contract is to take note of every transaction and registered users in the system concerning IoT sensor information

[13]. The model's next component is a web application, which acts as a boundary between the blockchain and the corresponding users, aiding them with their interaction with the smart contracts to occupy the data they have bought [13]. It also serves other functionalities that help develop user interactions with the proposed system [13]. There are three types of users in their system: 1) the main man, smart contract owner who is basically in charge of all the values placed by a single NTUA Token and also possesses the private key of the Externally Owned Account (EOA) from where the smart contracts were initially released, 2) IoT sensor operators or owners who basically provide the main IoT sensors and information in the marketplace of the system and also own the private key of their EOAs, and finally, 3) The buyers who are the customers of the marketplace, and who but the sensor data provided by the operators. The smart contract NtuaToken consists of the functions payable(), change price(uint price), retrieveEthereum(uint256 amount), etc., which focuses on the value of the custom tokens used by the system, and also for transferring the Ethers throughout the application [13]. The smart Broker contract creates a sensor() that focuses solely on the data to be shared in the application. They have also used the Ropsten Testnet to deploy the smart contracts for evaluating their results. Even though the proposed application did provide expected results, the authors failed to reduce the cost by each gas usage of the functions [13], as the approximation was not well enough. Another drawback of the model was that the range of transaction delay was from seconds to minutes [13], suggesting that no consideration regarding the delays was done.

Chapter 3

Work Plan

3.1 Architecture of the Proposed Model

This section contains the description of the architecture to be implemented by focusing on the shortcomings we experienced on the existing architectures based on blockchain.

3.2 Existing Works

Implementation of the current architectures based on blockchain is complex since the network on which public blockchain works is open to all and lacks scalability. Taking this into account, new private and consortium blockchain implementations have emerged. However, these architectures are prone to attack as they are known to make maximum use of cloud storage; this results in compromising the user's privacy, potentially increasing the cost of implementing the solution. Moreover, smart contracts that use Ethereum-based models are not cost-effective for smart home systems.

Recent literature rarely contains the real-life implementation of the models they propose. Additionally, the TX verification requires an extra node (user node). Therefore, whenever a node wishes to get connected to the home network, the TX created by the associated node needs to be validated by the other nodes; this is troublesome for a single homeowner.

Considering all these issues, we developed a solution for smart homes that work on a private local blockchain. Here, in place of the user nodes, the IoT devices play the role of a node in the blockchain process and actively participate during the transaction verification. A separate process known as RESTful API is used to authorize the users. The security and privacy of the architecture are enhanced by implementing the core blockchain process through Proof of Work together with additional checking for verification and security that increases the system's integrity and safety. The solution provides a design that can be implemented simply. It is also cost-effective, safe, and consumes less amount of time. The following subsection contains a detailed description of the architecture.

3.3 Building Blocks

1) SUPER NODE (SN)
Super Node (SN) is described as a storage for all the ledgers in the blockchain, which is done by a Peer-to-Peer server. By communication with the sensor nodes (SeNs), this node is utterly responsible for the transaction verification process of the users connected to the smart home environment. Moreover, the supernodes help a user verify themselves in the smart home network, and in the process, the registered user's data is stored for future use. It connects to the users through RESTful (Representational State Transfer). A safe and secure API that applies the principles of REST architecture for services provided by the web, and hence it ensures us with a secured communication process over the internet.

2) SENSOR NODES (SeN)
Sensor Nodes are equally crucial for user authorization and transaction verification by communicating with the SNs. The Sensor Nodes receive the broadcasted transactions made by the SNs; these are being called miners. When a transaction is created when a new user enters the intelligent home network, a block is made by the miners. This block needs to verify a PoW by other miners before finally adding the block into the blockchain storage.

Blockchain, Sensor Nodes(SeN), Users, and Super Node(SN) are used to create the building blocks in the proposed model. All SeNs and SN use the mesh network topology to communicate inside the smart home network. A renewed version of blockchain called consortium blockchain has been used to design the solution[23]. Here, the pre-chosen nodes generate the blocks and participate in consensus; hence, all nodes do not need to participate in consensus. The reason for choosing consortium blockchain was its ability to reduce the load on the network and communication overhead, which are considered ideal for smart home environments. This proposed model has eliminated the concept of using a user's performance as a node. Instead, mining occurs through the participation of all the smart devices present inside the smart home, where they play the role of a node. Nonetheless, if the number of devices increases, the user will have the liberty to choose two devices for mining where N=2 (N represents the number of nodes).

A mixture of predefined $SeN \in N$ and $SN \in N$ is used to represent the total number of the miner, which Q. denotes. Mathematically we represent it using the equation below.

$$Q = \sum_{i=1}^{N} (SeN\ i) + SN \tag{3.1}$$

where N=5, as the proposed model has only SeNs and no SN. Here we are excluding SN because it takes the role of the top layer and ensures that each SeN takes part in multiple processes and starts communicating to the users. Furthermore, the communication between sensor nodes is shown using Pij. Here P is used to indicate the packet containing the data being sent from node i to node j where $i, j \in N$. The network model with SeNs and SN is analogous to a strategic game which has a set of N players(network sensor nodes) such that N = 1,2,3,4,5,6. . . , N. Participation rules for transaction verification are dictated by SN for $SeNs \in N$. The role of each SeN is to ensure that its participation is maximum in the verification process. The greater the number of nodes, the greater the time to complete a transaction. Let T denote the total time it takes for a transaction to complete, which includes the time Q miners take given by Eq 3.1, and mathematically can be expressed as

$$T_M = \sum_{i=1}^{M} (ti) \tag{3.2}$$

Next, we observe the crucial building blocks for the proposed model.

3.4 Process for Verifying Transactions

If a block, bi, wants to get recognized by participants in the network and be a part of the blockchain, miner(sensor nodes), Q, needs to complete PoW. The creation of blocks is slowed down using PoW since it gets challenging to tinker as tinkering one block means the offender needs all blocks' PoW calculation, which is nearly impossible. The process requires significant time since PoW covers the entire data needed for mining in block bi. A complex mathematical problem is resolved for initiating PoW, a unique transaction for each block. As the mathematical problems to each block are unique, a mathematical problem that is unique for the block they made is attempted by each miner, with each of these problems having equal amounts of difficulty when it comes to fixing. Sufficient computer power is required for solving this mathematical problem. However, for cases including turning on/off lights, locking, and unlocking doors, it is not regarded as optimal as they need to be performed quickly. Despite this, it is a significantly proven effective technique known to provide the highest level of safety in systems involving blockchain. Hence, considering the security measures of a smart home, PoW is given a difficulty target of 1 for creating a reasonable delay. SN looks at the blockchain ledger in its database when the authorized user sends a command to perform an activity. After SN generates a block transaction, the old ledger is updated if the old ledger is found. Through the P2P server, a new block is broadcasted to all SeNs. The SeNs(miners) are detected automatically by the SN based on which SeN is readily available and has robust network connectivity.

The new block is being validated against the last five blocks they previously had in the blockchain by the SeNs. After this validation process, mining will be performed by the SeNs by generating a hash output to verify the data in its block with the difficulty target 1. All SeNs check the intended referenced device upon completion in the incoming request. The activity then gets accepted by the targeted SeN and performs the requested action while waiting for other SeNs for acknowledgment. Multiple security checks are carried out on the user's device's request before block mining, and the creation of the transaction is performed. Figure 3.1 explains how the entire process of the proposed architecture takes place.

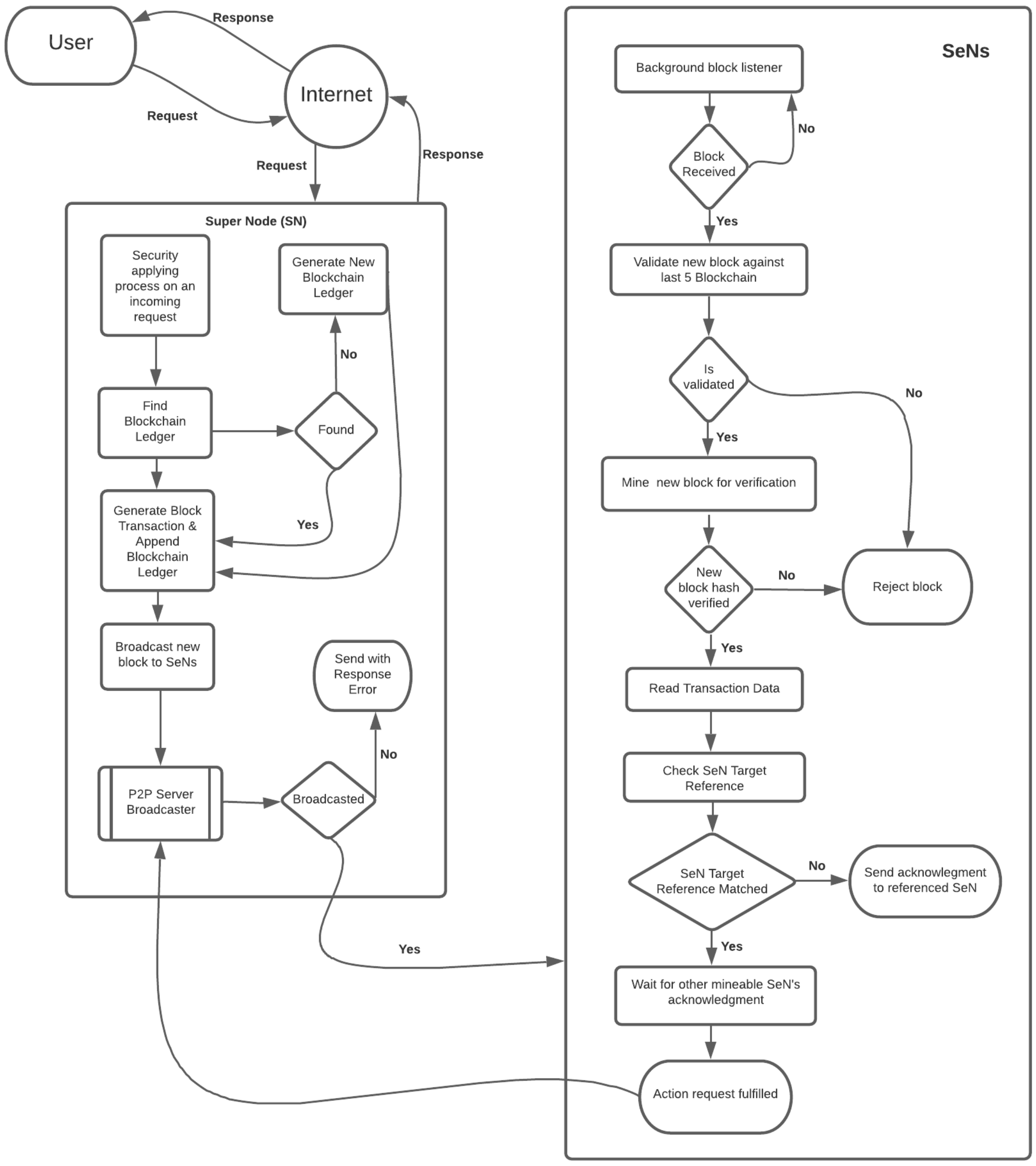

User
Response
Request
Internet
Request
Response
Super Node (SN)
Security applying process on an incoming request
Generate New Blockchain Ledger
No
Find Blockchain Ledger
Found
Yes
Generate Block Transaction & Append Blockchain Ledger
Broadcast new block to SeNs
Send with Response Error
No
P2P Server Broadcaster
Broadcasted
Yes
SeNs
Background block listener
No
Block Received
Yes
Validate new block against last 5 Blockchain
Is validated
No
Yes
Mine new block for verification
New block hash verified
No
Reject block
Yes
Read Transaction Data
Check SeN Target Reference
SeN Target Reference Matched
No
Send acknowlegment to referenced SeN
Yes
Wait for other mineable SeN's acknowledgment
Action request fulfilled

Figure 3.1: Process Flow of the proposed architecture

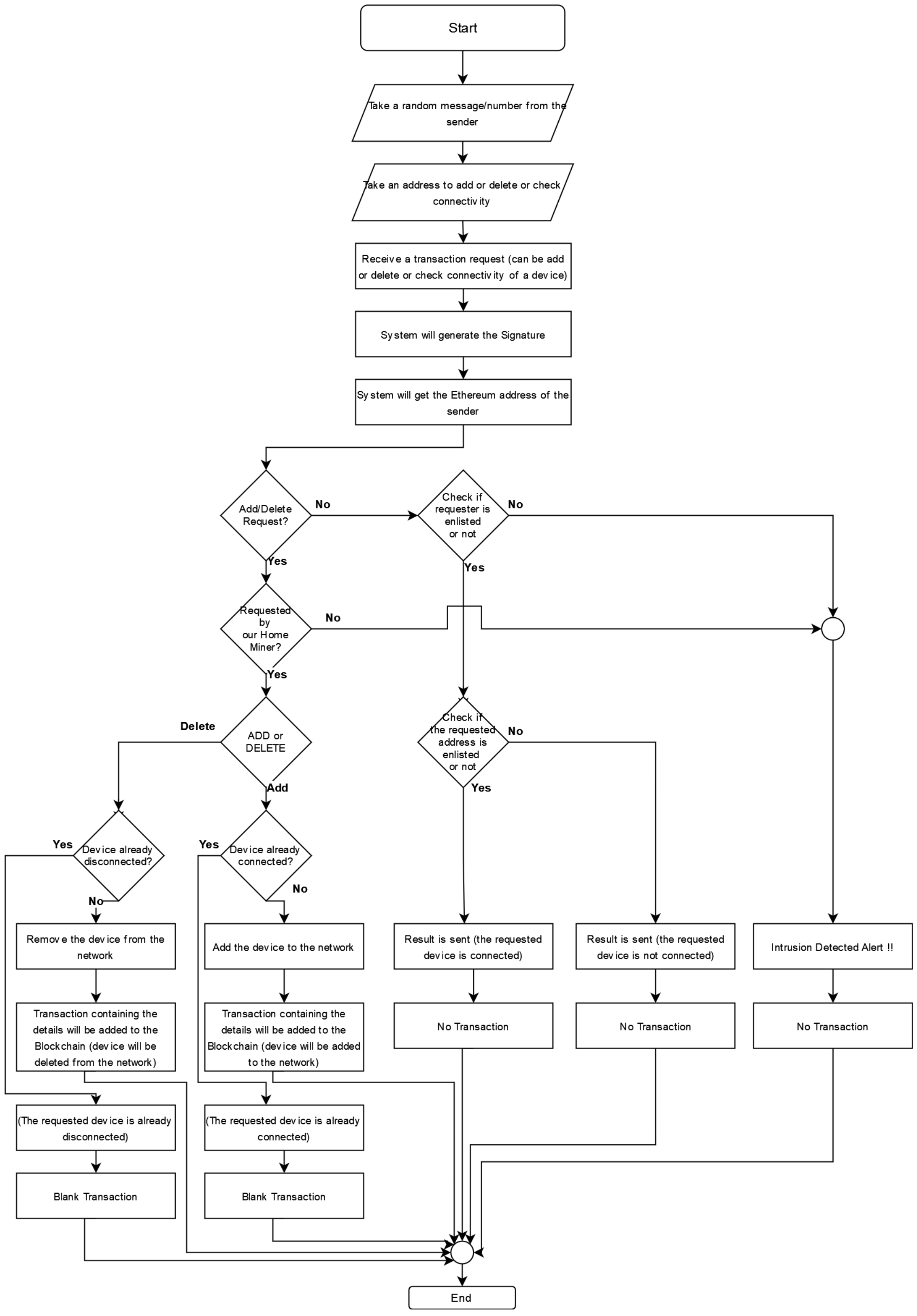
3.5 Ensuring Security on Incoming Requests

In almost all of the IoT networks, the users are aware of the interconnecting devices, i.e., and a thermostat comes with an IoT enables air conditioner where the thermostat is responsible for controlling the temperature of the surrounding after measuring the temperature; similarly, an IoT enabled motor is connected with a depth sensor which delivers signals to turn the motor on or off.

The architecture will consist of a decentralized server where every device will possess an instance of a consortium and local blockchain, which by definition is granted only to a group of approved individuals and is regarded as the most firm system when it comes to immutability. The unique Ethereum address of each node (here, devices are called nodes) will be contained in a smart contract. This unique address will be required by nodes or devices to connect with the blockchain instance to carry out their functions. As such, the devices on our network cannot establish communication with nodes whose addresses are missing from our smart contract. This, in turn, ensures that our IoT devices are secured from malicious attacks.

Here, the owner of the Ethereum account can only make changes to the blockchain network, that is - add or delete a device. The owner here is called the 'Home Miner' who only has the authority to decide which node can establish communication. The secured model will be run using a local private blockchain, and hence every IoT device connected to the safe architecture will need to install an instance of this blockchain. The system can process only three types of requests for a device from the home miner: add, delete, and the last is where it checks connectivity. The first two are valid only upon requests of the home miner. A device needs to send an address of an Ethereum account that is valid coupled with a random message regardless of the nature of the request. Both, in turn, form an encrypted signature allowing the system to get the signature associated with the message that is hashed upon the message being sent over by the requester. The signature can be decrypted through this process by our smart contract and obtain the sender's Ethereum address.

If the request for add or delete has been sent by any other node other than the home miner, then our system will not go through with the request and report possible intrusion. This identification of intrusion was possible because our system is already aware of the Ethereum address of the home miner. On the contrary, the address shall be verified if the home miner made the request. The verification is carried out to add or delete from the network. The blockchain will be updated with a transaction where it will have the address and information regarding the device.



Start
Take a random message/number from the sender
Take an address to add or delete or check connectivity
Receive a transaction request (can be add or delete or check connectivity of a device)
System will generate the Signature
System will get the Ethereum address of the sender
Add/Delete Request?
No
Check if requester is enlisted or not
No
Yes
Yes
Requested by our Home Miner?
No
Yes
Delete
ADD or DELETE
Add
Check if the requested address is enlisted or not
No
Yes
Yes
Device already disconnected?
No
Yes
Device already connected?
No
Remove the device from the network
Add the device to the network
Result is sent (the requested device is connected)
Result is sent (the requested device is not connected)
Intrusion Detected Alert !!
Transaction containing the details will be added to the Blockchain (device will be deleted from the network)
Transaction containing the details will be added to the Blockchain (device will be added to the network)
No Transaction
No Transaction
No Transaction
(The requested device is already disconnected)
(The requested device is already connected)
Blank Transaction
Blank Transaction
End

Figure 3.2: Flowchart for Blockchain-based IoT Model

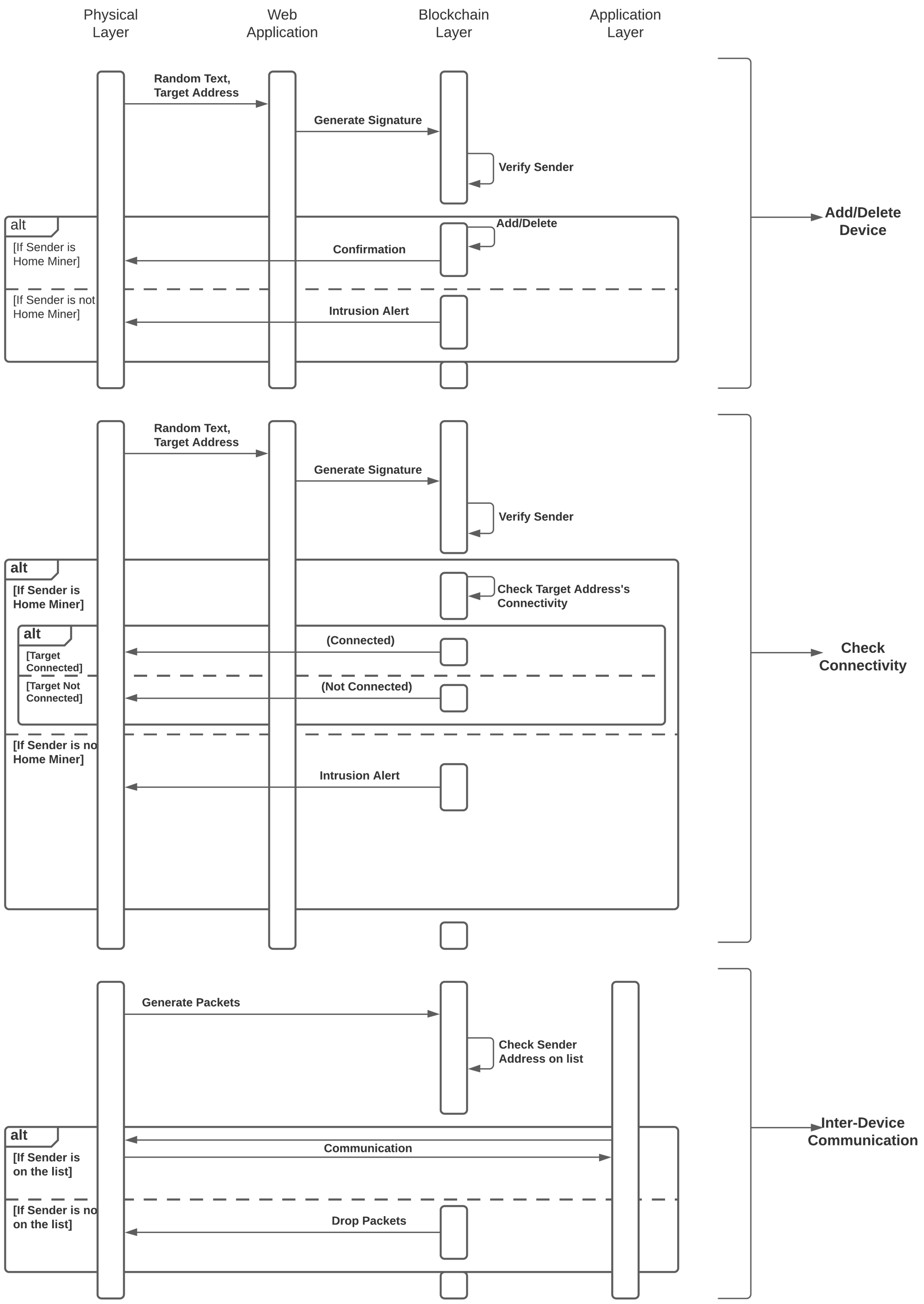


Figure 3.3: Sequence Diagram for Blockchain-based IoT Model

3.6 Checking Connectivity

A request for check connectivity is carried out to verify if any device from outside is part of our network or not. To put it simply, this request is made when an IoT device wants to establish communication with another IoT device carrying the provided Ethereum address. The smart contract first checks the requester's account address to execute this request. When it matches with the home miner, it indicates a valid request. The home miner then seeks to determine whether a communication needs to be established with the sent address. Following this, a security check is carried out by the smart contract to verify whether the address that was requested exists on our list. A true or false is sent as a result after confirming. If the sender is not our home miner, the system will realize that someone from outside of our system is looking for access and hence will count this as intrusion before taking the appropriate steps.

Elliptic-curve cryptography is used to verify the home miner, a well-known algorithm for verifying digital signatures. It is Public-key cryptography where the structure of elliptic curves is used over finite fields. Here, a public key is used to verify the signature of a sender, where the only possible way of generating the signature is by using the sender's private key.

3.7 User Authorization

This subsection describes how a user is verified after entry has been authorized. SN takes the help of RESTful API(Representational State Transfer) for ensuring users are allowed. A RESTful API is responsible for transferring data securely from one system to another over the internet. It designs web services through which the system can access its resources by applying a set of predefined rules. The architectural constraints make the RESTful API suitable for transferring data safely in the proposed model. The smart home system will have two users - Admin and General Users. At first, the SN registers the admin user because the admin user is pre-authorized. General users can be a part of the smart home network if the admin user adds them by providing the admin user with the IMEI of their device. The admin user initially installs the smart home application as an authorized user, and a request is sent through the application for adding general users. The application will generate the particular user's unique key. RESTful API then sends the unique key to SN. SN then verifies the key; SN registers the user as a new client through its unique identifier. For further communication, the general user is provided with a username and password by the admin user. The unique key, stored in SN's database, gets identified through its unique key whenever the user requests the SN. Figure. 3.4 represents the proposed architecture's user authorization process.

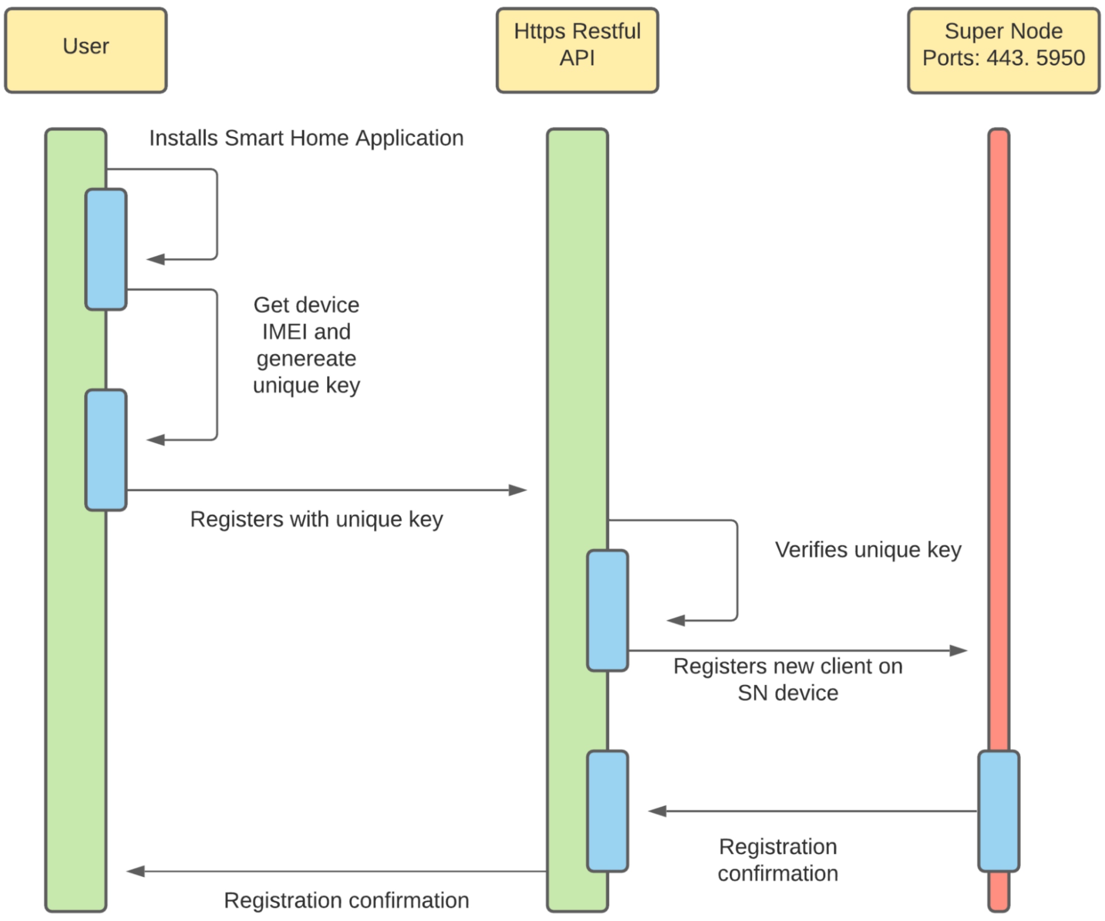

User
Https Restful API
Super Node Ports: 443. 5950
Installs Smart Home Application
Get device IMEI and genereate unique key
Registers with unique key
Verifies unique key
Registers new client on SN device
Registration confirmation
Registration confirmation

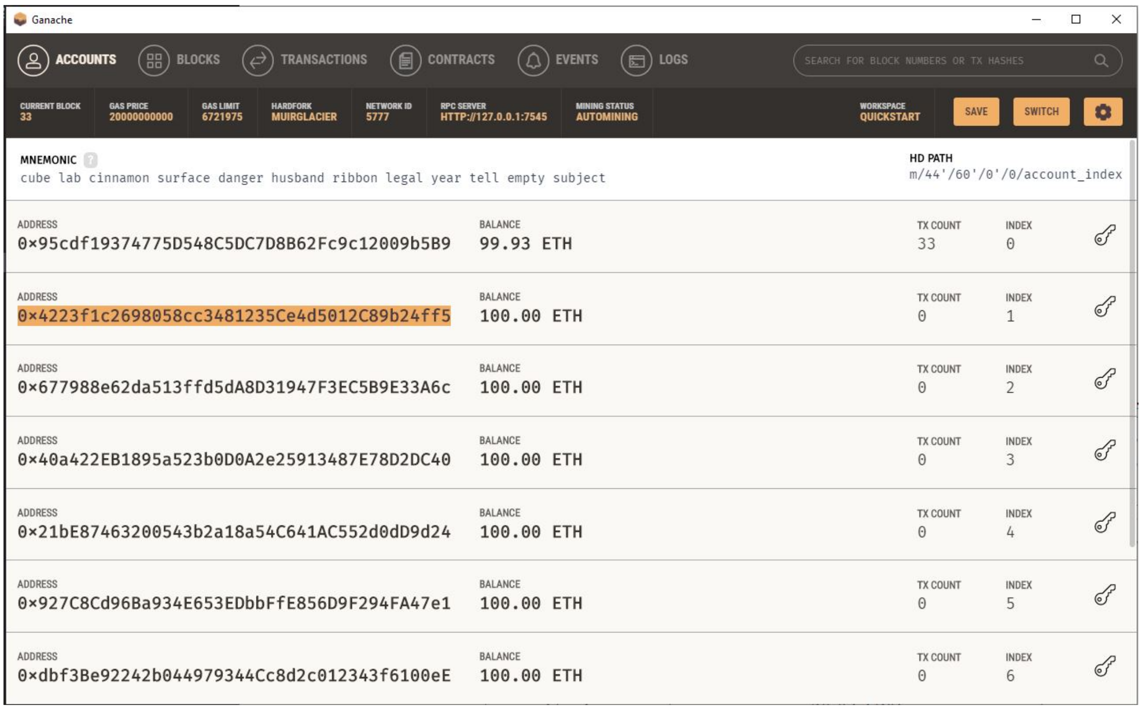
Figure 3.4: User Authentication process

Chapter 4

Progress

4.1 Experimental Configuration

We have used Intel(R) Core(TM) i5-8250U Processor to administer the model. This processor has four cores with a maximum clock speed of 1.60GHz. Our operating system was Microsoft Windows 10 Home Single Language of 10.0.19042 version.



CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
33	20000000000	6721975	MUIRGLACIER	5777	HTTP://127.0.0.1:7545	AUTOMINING	QUICKSTART

MNEMONIC
cube lab cinnamon surface danger husband ribbon legal year tell empty subject

HD PATH
m/44'/60'/0'/0/account_index

ADDRESS	BALANCE	TX COUNT	INDEX
0×95cdf19374775D548C5DC7D8B62Fc9c12009b5B9	99.93 ETH	33	0
0×4223f1c2698058cc3481235Ce4d5012C89b24ff5	100.00 ETH	0	1
0×677988e62da513ffd5dA8D31947F3EC5B9E33A6c	100.00 ETH	0	2
0×40a422EB1895a523b0D0A2e25913487E78D2DC40	100.00 ETH	0	3
0×21bE87463200543b2a18a54C641AC552d0dD9d24	100.00 ETH	0	4
0×927C8Cd96Ba934E653EDbbFfE856D9F294FA47e1	100.00 ETH	0	5
0×dbf3Be92242b044979344Cc8d2c012343f6100eE	100.00 ETH	0	6

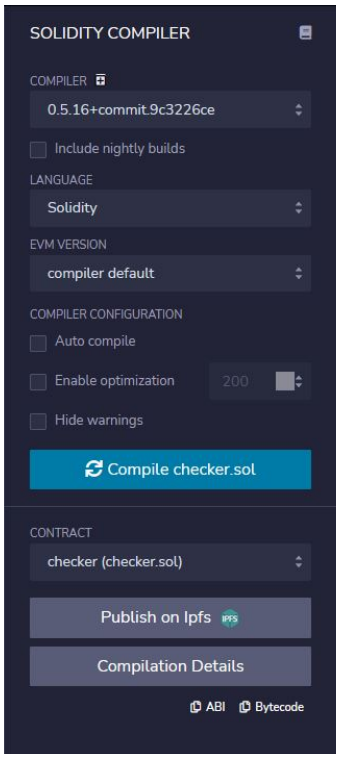
Figure 4.1: Ganache UI

4.2 Experimental Implementation of the Model

A genesis block can be created to deploy a private blockchain on our computer through which we will be able to run an instance of private blockchain and carry out the rest of the procedure after deploying our smart contract. But instead, Ganache is used, a truffle framework to make the process easier. Truffle developed Ganache which is a personal blockchain for deploying or developing dApps. Figure

4.1 is used to exhibit the UI of Ganache upon starting. Ten ethereum accounts are given after starting, where each of them contains 100 ethers. The entire process is run on a private blockchain.

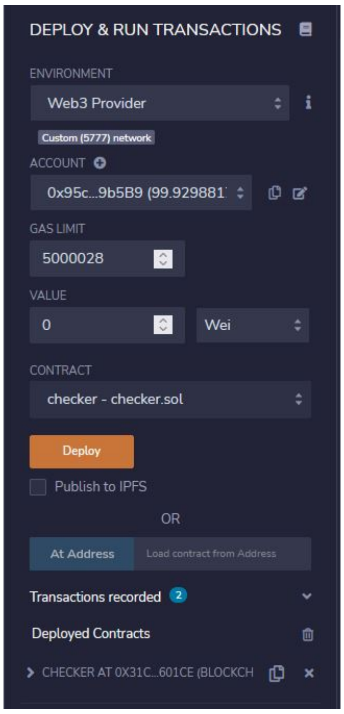
Our blockchain needs an intelligent contract to be deployed while running on Ganache. This is done using the Remix IDE. ABI is collected after writing and compiling the solidity code on the Remix IDE. This collection of Application Binary Interface is shown in figure 4.2. After this, the smart contract was deployed on our ganache blockchain, which gave us a deployment address. The smart contract uses this deployment address for transferring virtual ethers to complete the required transaction. The collection of the deployment address is shown in figure 4.3.


SOLIDITY COMPILER

COMPILER
0.5.16+commit.9c3226ce
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization 200
Hide warnings
Compile checker.sol

CONTRACT
checker (checker.sol)
Publish on Ipfs
Compilation Details
ABI Bytecode

Figure 4.2: Smart Contract ABI Extraction


DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: Web3 Provider

Custom (5777) network

ACCOUNT: 0x95c...9b5B9 (99.929881

GAS LIMIT: 5000028

VALUE: 0 Wei

CONTRACT: checker - checker.sol

CHECKER AT 0X31C...601CE (BLOCKCH

Figure 4.3: Extraction of Smart Contract Address

Our web application allows our blockchain to communicate with the help of smart contracts. Figure 4.4 shows how the system looks when started. It represents the model we proposed and can be considered its UI. Our system will ignore requests from every account containing our device's unique Ethereum address if it tries to tamper with information. Since our blockchain has no speech or device saved, we will get an output like figure 4.5 if a request is made to check connectivity after starting. Figure 4.6 shows the interface that will appear if we add a device. Here, it can be seen that the user needs to input a random message and the target device's Ethereum address. Figure 4.7 is the interface we get after requesting a connectivity check with the same inputs we gave before, indicating that devices have been successfully added. Figure 4.8 shows the user interface that will appear if a device is removed from the network. Figure 4.9 shows how to check connectivity to confirm that our trusted device list does not contain the specified device.

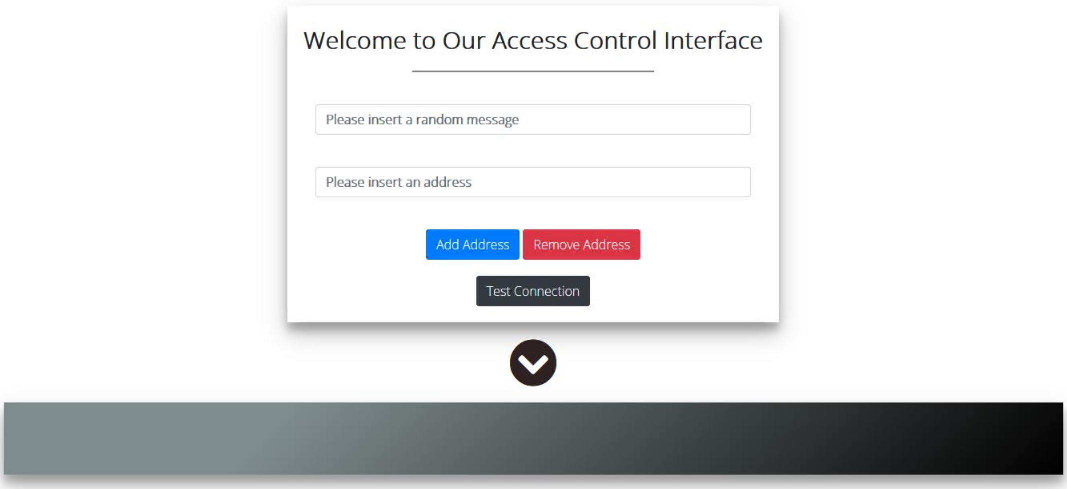
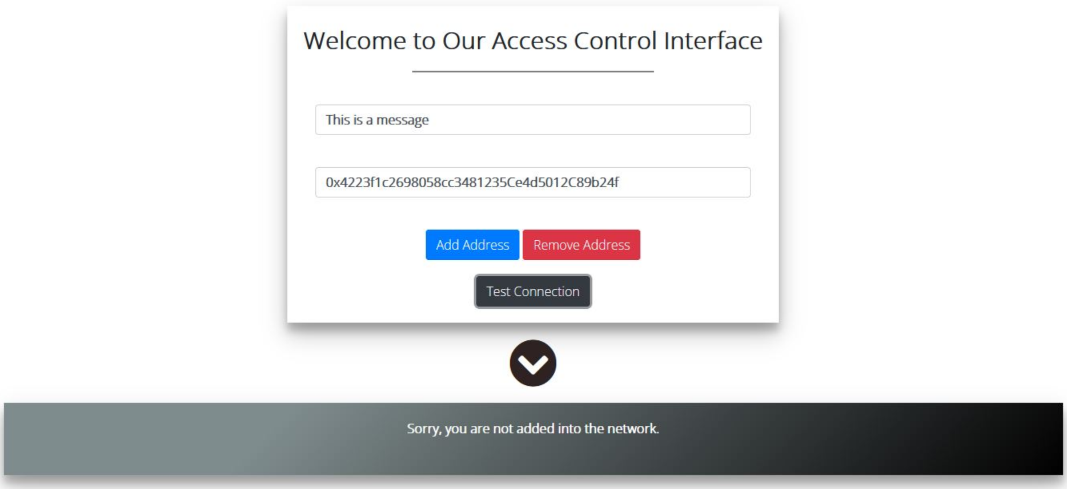
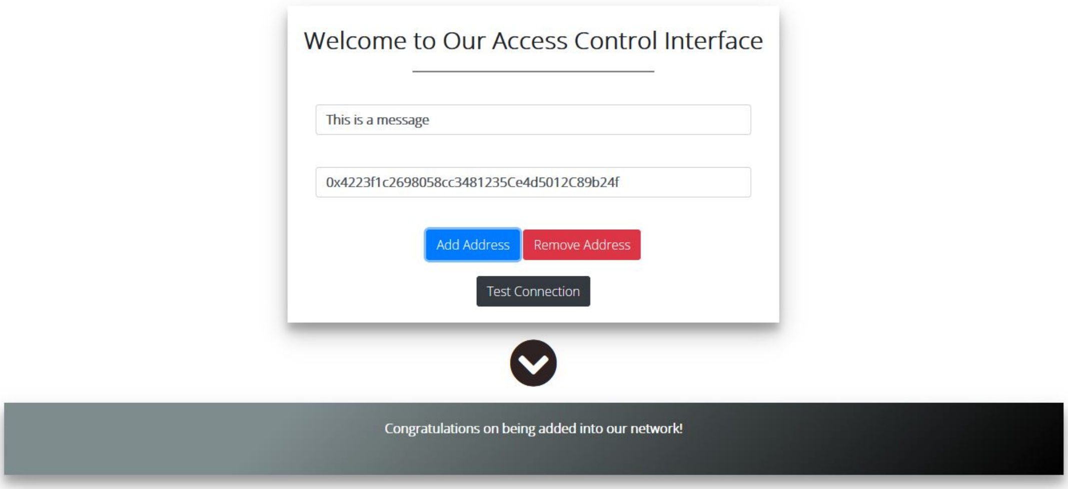

Welcome to Our Access Control Interface
Please insert a random message
Please insert an address
Add Address
Remove Address
Test Connection

Figure 4.4: First state of the system


Welcome to Our Access Control Interface
This is a message
0x4223f1c2698058cc3481235Ce4d5012C89b24f
Add Address
Remove Address
Test Connection

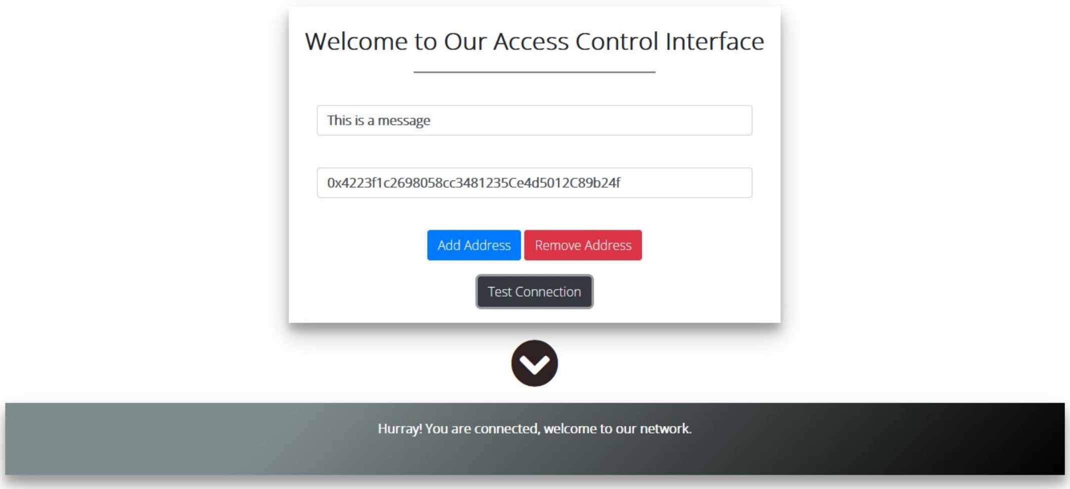
Sorry, you are not added into the network.

Figure 4.5: Checking Connectivity


Welcome to Our Access Control Interface
This is a message
0x4223f1c2698058cc3481235Ce4d5012C89b24f
Add Address
Remove Address
Test Connection

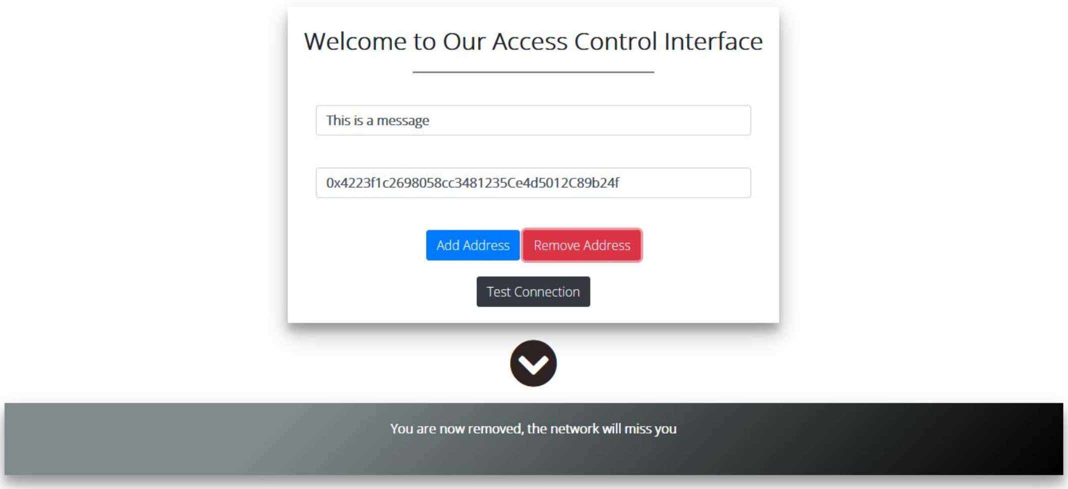
Congratulations on being added into our network!

Figure 4.6: Adding Address of a Device


Welcome to Our Access Control Interface
This is a message
0x4223f1c2698058cc3481235Ce4d5012C89b24f
Add Address
Remove Address
Test Connection

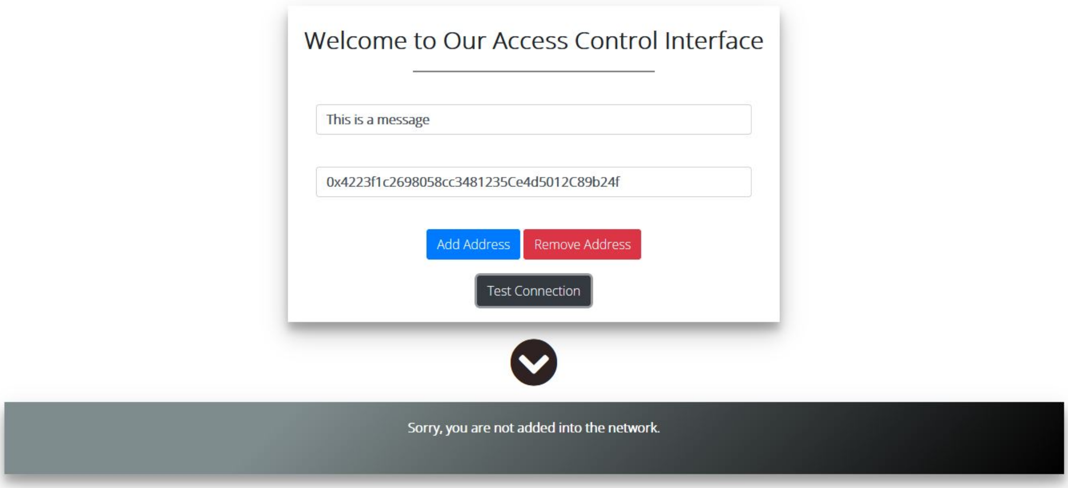
Hurray! You are connected, welcome to our network.

Figure 4.7: Checking Connectivity Again


Welcome to Our Access Control Interface
This is a message
0x4223f1c2698058cc3481235Ce4d5012C89b24f
Add Address
Remove Address
Test Connection

You are now removed, the network will miss you

Figure 4.8: Removing Address of a Device


Welcome to Our Access Control Interface

This is a message

0x4223f1c2698058cc3481235Ce4d5012C89b24f

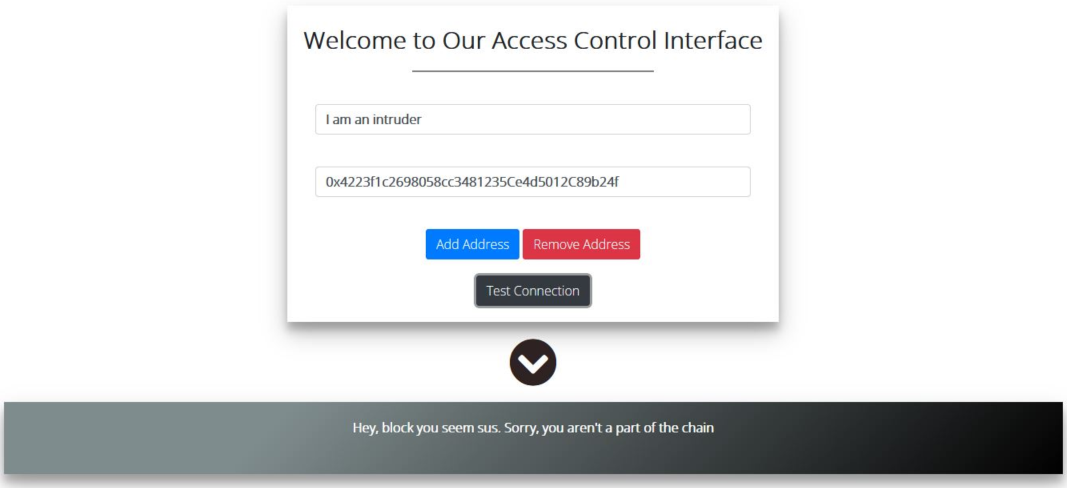
Sorry, you are not added into the network.

Figure 4.9: Check After Removal of Address

4.3 Experimental Testing of the Model

An outsider can't enter our network as we run a local private Ethereum network. We deliberately altered a digit of the smart contract's address to test our application. Figure 4.10 shows the output during possible intrusion after a check connectivity request was made. It is a clear indication it is possible to detect whether there have been any changes made to the digits during setup, and thus we can take the needed actions.

Figure 4.11 shows how we attempted to change blockchain data as an outsider. It is exhibited that the information is changed from the outsider's point of view. Figure 4.12 shows that the information was not altered after we established a connection, and the device was there in the network despite our attempt to remove it as an outsider.

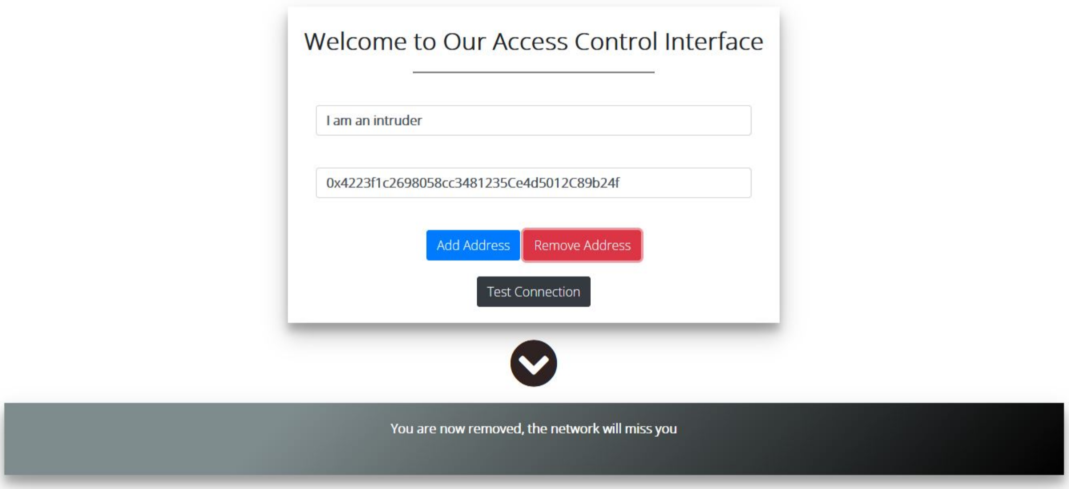

Welcome to Our Access Control Interface

I am an intruder

0x4223f1c2698058cc3481235Ce4d5012C89b24f

Hey, block you seem sus. Sorry, you aren't a part of the chain

Figure 4.10: Intrusion Detection

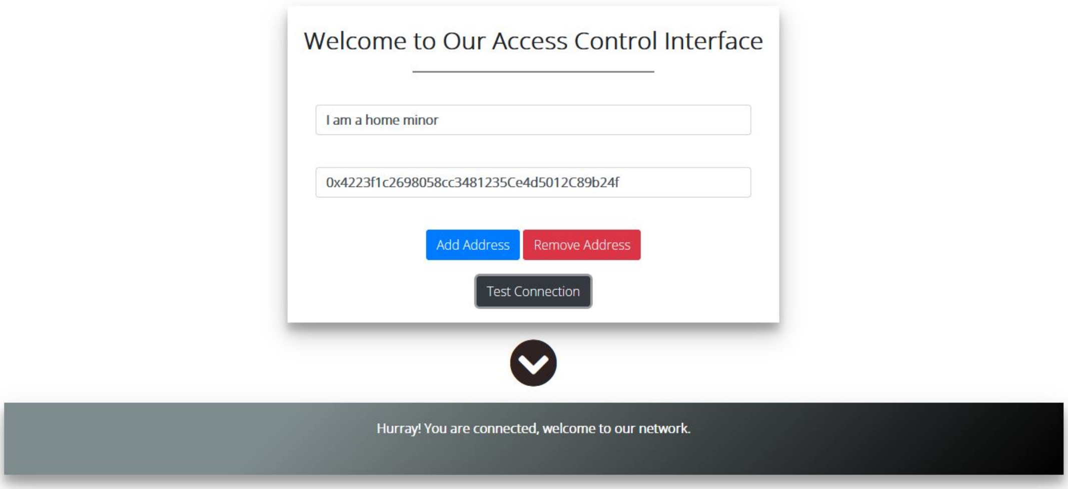


I am an intruder

0x4223f1c2698058cc3481235Ce4d5012C89b24f

You are now removed, the network will miss you

Figure 4.11: Intruders view to a request



I am a home minor

0x4223f1c2698058cc3481235Ce4d5012C89b24f

Hurray! You are connected, welcome to our network.

Figure 4.12: Home Miner's view of a request

Chapter 5

Analysis and Results

In this chapter, we will analyze the performance, capacity and implementing cost of the proposed model.

5.1 Performance

It takes 10 minutes for the bitcoin blockchain to generate transactions, while Ethereum can affect up to 20 transactions per second. Twenty inputs per second should be ideal for the model we proposed. We need only take speed when admitting devices greater than 20 per second. Getting 20 Ethereum address inputs per second is quite impractical for humans. Hence, this won't be a drawback for our proposed model.

When a request for a valid transaction appears during mining of a block by a node, then mempool stores the incoming request. Mempool is the place where the transaction for a node waits. The transaction is processed upon completion of mining. To put it simply, the probability that a mempool will hold the transactions is directly proportional to the time a block takes for mining. The time to mine a block is based upon a blockchain's difficulty level. Since we are running an instance of a private blockchain, we can adjust the difficulty level to low, ensuring that the time taken to process a transaction is minimal. This ensures that it can run smoothly on high-configuration devices and low-configuration devices.

The performance of this model is further enhanced through the use of Ganache for the web version. Normally, when we implement transactions using Ethereum, there is a high probability of transactions getting stuck. This arises because the transactions are not getting picked up by a miner. This problem is solved when we implement Ethereum using Ganache.

Most of the models that we saw used public blockchains. As a result performance was compromised in those models since public blockchains require more energy than private blockchain. This happens due to public blockchains requiring large amounts of electrical resources for functioning.

Public blockchains are inherently slower as they need to process multiple transactions at a time since they can be accessed by everyone on the internet. On the contrary, private blockchain can only be accessed by a selected few. Hence, the speed remains the same.

5.2 Storage Handling

A primary concern regarding blockchain is storage. Approximately 1.498 GB is the capacity of the Ethereum blockchain currently. The power is increasing every day, with additional blocks getting added. Here, a string containing a list of Ethereum addresses is saved only. It can be deduced that there won't be a large number of blocks due to the architecture of the model we proposed, and hence it is safe to say that storage won't act as a hindrance to the proposed architecture.

5.3 Cost-benefit Analysis

It is practically possible to run this model at zero cost as a private blockchain is free. Thus IoT devices can have private blockchain instances running for free on a 24/7 basis after deploying smart contracts. Virtual ethers are used on the local blockchain and will not cost a penny to the device owner.

The prices for models that run on public blockchains rise drastically as it contains a greater number of nodes and receives numerous requests every second which increases the transaction cost significantly. Our model is cheaper than the models we have come across so far as most of the models used public blockchain, which is quite expensive and is mainly used for large-scale applications, i.e., intelligent power grid, smart city, etc.

A summary of the analysis is shown in Table 5.1.

Category	Private Blockchain	Public Blockchain
Performance	Faster due to lesser nodes	Slower due to greater number of nodes
Storage	Optimal	Provides access to anyone on the Internet
Transaction Cost	Cheap	Expensive

Table 5.1: The pros of private blockchain over public blockchain

Chapter 6

Conclusion

The cost of this model makes it an ideal choice for a smart clinic where data that is not so important can be kept on a cloud server. The blockchain is attached to the cloud server so that the blockchain does not have to hold onto unnecessary data.

In our report, we researched previous work that has been done on smart home security using blockchain. Taking the drawbacks into account, we took a more simple step where we focused on user verification and emphasized security in incoming requests. Most of the architectures we came across used public blockchain, which is costly and possesses a lot of security threats since it is open to everyone. Hence, we pivoted our model on a private blockchain which only allows authorized users. It ensures a smooth run on both low and high-end devices and comes at a total free of cost. In the future, we hope to carry out further research on it and test the performance using Raspberry Pi.

References

[1] I. Andrea, C. Chrysostomou, and G. Hadjichristofi, “Internet of things: Security vulnerabilities and challenges,” in *2015 IEEE Symposium on Computers and Communication (ISCC)*, Larnaca: IEEE, Jul. 2015.

[2] D. Ott, C. Vishik, D. Grawrock, and A. Rajan, “Trust evidence for IoT: Trust establishment from servers to sensors,” in *ISSE 2015*, Wiesbaden: Springer Fachmedien Wiesbaden, 2015, pp. 121–131.

[3] T. Abera, N. Asokan, L. Davi, *et al.*, “C-flat: Control-flow attestation for embedded systems software,” in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 743–754.

[4] A. Nordrum, *Popular internet of things forecast of 50 billion devices by 2020 is outdated*, Aug. 2016. [Online]. Available: https://spectrum.ieee.org/popular-internet-of-things-forecast-of-50-billion-devices-by-2020-is-outdated.

[5] L. J¨ager, R. Petri, and A. Fuchs, “Rolling dice: Lightweight remote attestation for cots iot hardware,” in *Proceedings of the 12th International Conference on Availability, Reliability, and Security*, ACM, 2017.

[6] N. Kshetri, “Can blockchain strengthen the internet of things?” *IT Prof.*, vol. 19, no. 4, pp. 68–72, 2017.

[7] T. Malche and P. Maheshwary, “Internet of things (IoT) for building smart home system,” in *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam, Tamilnadu, India: IEEE, Feb. 2017.

[8] *2018 10th International Conference on Advanced Infocomm Technology (ICAIT)*, Stockholm, Sweden: IEEE, Aug. 2018.

[9] M. S. Ali, M. Vecchio, M. Pincheira, K. Dolui, F. Antonelli, and bibinitperiod M. H. Rehmani, “‘applications of blockchains in the internet of things: A comprehensive survey,” *Commun. Surveys Tuts*, vol. 21, no. 2, pp. 1676–1717, 2018.

[10] T. Golomb, Y. Mirsky, and Y. Elovici, “Ciota: Collaborative iot anomaly detection via blockchain,” *arXiv preprint arXiv:1803.03807*, 2018.

[11] E. Kfoury and D. Khoury, “Securing NATted IoT devices using ethereum blockchain and distributed TURN servers,” in *2018 10th International Conference on Advanced Infocomm Technology (ICAIT)*, Stockholm, Sweden: IEEE, Aug. 2018.

[12] A. Panarello, N. Tapas, G. Merlino, F. Longo, and A. Puliafito, “Blockchain and iot integration: A systematic survey,” *Sensors*, vol. 18, no. 8, p. 2575, 2018.

[13] G. Papadodimas, G. Palaiokrasas, A. Litke, and T. Varvarigou, "Implementation of smart contracts for blockchain-based iot applications," in *2018 9th International Conference on the Network of the Future (NOF)*, IEEE, 2018, pp. 60–67.

[14] A. K. Sikder, G. Petracca, H. Aksu, T. Jaeger, and A. S. Uluagac, "A survey on sensor-based threats to internet-of-things (iot) devices and applications," *arXiv preprint arXiv:1802.02041*, 2018.

[15] M. Singh, A. Singh, and S. Kim, "Blockchain: A game-changer for securing iot data," in *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, 2018, pp. 51–55.

[16] N. Fotiou, I. Pittaras, V. A. Siris, S. Voulgaris, and G. C. Polyzos, "Secure iot access at scale using blockchains and smart contracts," in *2019 IEEE 20th International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*, IEEE, 2019, pp. 1–6.

[17] M. A. Islam and S. Madria, "A permissioned blockchain based access control system for iot," in *2019 IEEE International Conference on Blockchain (Blockchain)*, IEEE, 2019, pp. 469–476.

[18] M. A. Islam and S. Madria, "A permissioned blockchain based access control system for IOT," in *2019 IEEE International Conference on Blockchain (Blockchain)*, Atlanta, GA, USA: IEEE, Jul. 2019.

[19] T. Tantidham and Y. N. Aung, "Emergency service for smart home system using ethereum blockchain: System and architecture," in *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, Kyoto, Japan: IEEE, Mar. 2019.

[20] S. Arif, M. Khan, S. Rehman, M. Kabir, and M. Imran, "Investigating smart home security: Is blockchain the answer?" en," *IEEE Access*, vol. 8, pp. 117 802–117 816, 2020.

[21] S. Arif, M. A. Khan, S. U. Rehman, M. A. Kabir, and M. Imran, "Investigating smart home security: Is blockchain the answer?" *IEEE Access*, vol. 8, pp. 117 802–117 816, 2020.

[22] J. Fruhlinger, *What is IoT? the internet of things explained*, Accessed: 2022-1-13, May 2020.

[23] Y. Lee, S. Rathore, J. H. Park, and J. H. Park, "A blockchain-based smart home gateway architecture for preventing data forgery," *Human-centric Computing and Information Sciences*, vol. 10, no. 1, pp. 1–14, 2020.

[24] S. Sun, R. Du, S. Chen, and W. Li, "Blockchain-based IoT access control system: Towards security, lightweight, and cross-domain," *IEEE Access*, vol. 9, pp. 36 868–36 878, 2021.

Appendix

Smart Contract Code:

```
pragma solidity ^0.5.16;
contract checker {
 address [] connected_devices;
 bytes32 miner = 0
   x8ea5947f4a057cd88cb5d0523bdb47efd3de8cd4ae45704b0c7d277e95dd17b7
   ;    //this will vary upon network to network
 function check(bytes32 hash, bytes memory signature) public view
   returns (bool) {
     if(keccak256(abi.encodePacked(msg.sender)) == miner &&
   keccak256(abi.encodePacked(recover(hash,signature))) == miner){
      return true;
     }
     else{return false;}
 }

function addit(bytes32 hashadd, bytes memory signatureadd, address
   targetadd) public payable returns (int) {
     if(check(hashadd,signatureadd) == true){
         for (uint i=0; i<connected_devices.length; i++) {
         if(keccak256(abi.encodePacked(targetadd)) == keccak256(abi
   .encodePacked(connected_devices[i]))) {
             return 1;
         }
     }
         connected_devices.push(targetadd);
         connected_devices.length++;
         return 0;
     }
     else{return 2;}
 }

function dltit(bytes32 hashdlt, bytes memory signaturedlt, address
   targetdlt) public payable returns (int) {
     if(check(hashdlt,signaturedlt) == true){
         for (uint i=0; i<connected_devices.length; i++) {
         if(keccak256(abi.encodePacked(targetdlt)) == keccak256(abi
   .encodePacked(connected_devices[i]))) {
             connected_devices[i]=connected_devices[
   connected_devices.length-1];
             delete connected_devices[connected_devices.length-1];
             connected_devices.length--;
             return 0;
         }
     }
     }
     else{return 1;}
 }

  function recover(bytes32 hash, bytes memory signature)
    public
    pure
    returns (address)
  {
    bytes32 r;
    bytes32 s;
    uint8 v;
    bytes memory prefix = "\x19Ethereum Signed Message:\n32";
    bytes32 prefixedHash = keccak256(abi.encodePacked(prefix, hash)
   );
    // Check the signature length
    if (signature.length != 65) {
      return (address(0));
    }

    // Divide the signature in r, s and v var*iables
    // ecrecover takes the signature parameters, and the only way
   to get them
    // currently is to use assembly.
    // solium-disable-next-line security/no-inline-assembly
    assembly {
      r := mload(add(signature, 0x20))
      s := mload(add(signature, 0x40))
      v := byte(0, mload(add(signature, 0x60)))
    }

    // Version of signature should be 27 or 28, but 0 and 1 are
   also possible versions
    if (v < 27) {
      v += 27;
    }

    // If the version is correct return the signer address
    if (v != 27 && v != 28) {
      return (address(0));
    } else {
      // solium-disable-next-line arg-overflow
      return ecrecover(prefixedHash, v, r, s);    //another functio
   ( heart of address recovery *)
    }
  }

function connectivity_check(bytes32 hashcnc, bytes memory
   signaturecnc, address targetcnc) public view returns (bool ,
   bool) {
     bool requester_coneected= false;
     bool target_connected=  false;
     for (uint i=0; i<connected_devices.length; i++) {
         if(keccak256(abi.encodePacked(msg.sender)) == keccak256(
   abi.encodePacked(connected_devices[i])) || keccak256(abi.
   encodePacked(msg.sender)) == miner) {
             if(keccak256(abi.encodePacked(msg.sender)) ==
   keccak256(abi.encodePacked(recover(hashcnc,signaturecnc)))){
```

```
                  requester_coneected=true;
              }
          }
      for (uint j=0; j<connected_devices.length; j++) {
      if(keccak256(abi.encodePacked(targetcnc)) == keccak256(abi.
    encodePacked(connected_devices[j]))) {
      target_connected=true;
          }
          }
      }
      return (requester_coneected, target_connected);
 }
}
```