2020

A secure hybrid IoT architecture using blockchain and decentralized VPN: a use case of smart farming.

Hilim, Peleke Madina
Faculty of Infrmation Technology
Strathmore University

A Secure Hybrid IoT Architecture Using Blockchain and Decentralized VPN: A Use Case of Smart Farming

By

HILIM-PELEKE Madina

Student Number: 114852

A Thesis Submitted to the Faculty of Information in partial fulfillment of the requirements for the award of Master of Science in Information Technology

Faculty of Information Technology

Strathmore University

Nairobi, Kenya

July 2020

Declaration and Approval

I declare that this work has not been previously submitted and approved for the award of a degree by this or any other University. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

HILIM-PELEKE Madina, 114852

Signature …….

Date: 13th July 2020

Approval

The thesis of HILIM-PELEKE Madina was reviewed and approved by the following:

Dr. Vitalis Ozianyi (Ph.D.),
Senior Lecturer, Faculty of Information Technology
Strathmore University.

Dr. Joseph Orero (Ph.D.)
Dean, Faculty of Information Technology
Strathmore University.

Dr. Bernard Shibwabo,
Director of Graduate Studies,
Strathmore University.

Abstract

The Internet of Things (IoT) technology has brought a revolution to every sector of man's life by making it interactive, insightful, and smart. IoT refers to a network of objects that make up a network that configures itself. The proliferation of smart farming IoT-based devices transforms the face of farming turn over day after day by not just only boosting it but also making it cost-effective and reducing wastage. Nonetheless, many IoT characteristics can lead to matters of security and privacy, such as connectivity, wireless, embedded usage, variety, and scale. These features make IoT special in its security requirements and pose numerous new information security threats. The objective of this study is to propose a secure IoT based smart farming using Blockchain Technology and decentralized VPN for secure and efficient environment monitoring which could encourage farmers and other stakeholders, despite other challenges, to embrace smart farming and increase their overall yield and quality of products. In this paper, the proposed integrated solution is based on Hyperledger and OpenVPN technology. The architecture is based on a wireless mesh network with robust encryption at the network and application layer. The proposed solution is resilient to the common wireless network attack.

Keywords: Internet of Things (IoT), Raspberry Pi, Arduino nodeMCU, Smart Farming, Blockchain, Docker, VPN, Soil Moisture Sensor, Hyperledger, and OpenVPN, Temperature Sensor, MQTT.

Table of Contents

List of Tables

List of Figures

List of Acronyms

AP	-	Access Point
API	-	Application Programming Interface
BC	-	Blockchain
BFT	-	Byzantine Fault Tolerance
DDoS	-	Distributed Denial of Service
FBI	-	Federal Bureau of Investigation
IoT	-	Internet of Things
RFID	-	Radio-Frequency Identification
SDK	-	Software Development Kits
US	-	United State
USDA	-	United State Department of Agriculture
VPN	-	Virtual Private Network
WAN	-	Wide Area Network
WSN	-	Wireless Sensor Networks

Definitions of Terms

Internet of Things: The Internet of Things (IoT) system is a system of linked computing devices which can be digital or mechanical machines, or objects, or living beings or people that have unique attribute or identifiers and can transfer data across a network without requiring any human intervention or computer-to-computer interaction (Akhil, 2019).

Blockchain Technology: it is a data structure, a combination of blocks that contain data. Each block holds a hash of the information of the previous block. The blocks are digitally signed to ensure data integrity and chained after the consensus process (Selimi et al., 2018).

Decentralized Virtual Private Network: is the tunnel infrastructure between nodes that ensure encryption of the data being transmitted. It is a recent VPN solution where the users are both client and server that enable a distributed communication without a unique central control (Karuna & Indira, 2018).

Smart Farming refers to managing farms using modern Information and Communication Technologies to increase the quantity and quality of products while optimizing the human labor required (Amandeep et al., 2017).

Raspberry Pi: is a low cost, credit-card sized Single Board Computer (Raspberry Pi, 2018).

Arduino NodeMCU: refers to an open-source electronics board and the software used to program it. It is a Wi-Fi microcontroller designed to make electronics more accessible to anyone interested in creating interactive objects or environments (Eetu, 2016)

Wireless Sensor Networks: refers to a group of spatially dispersed and dedicated sensors for monitoring and recording the physical conditions of the environment and organizing the collected data at a central location. WSNs measure environmental conditions like temperature, sound, pollution levels, humidity, wind, and the like (Muthuswamy & Ganapathi, 2016).

Acknowledgements

I am so grateful to God who has been faithful to me along this journey. I would like to express my gratitude to my supervisor Dr. Vitalis Ozianyi for his guidance in this master thesis. I would also like to acknowledge Dr. Bernard Shibwabo for his availability and his very valuable comments on this thesis.

Furthermore, I would like to thank my loved ones, especially the Marianist Community for their love and for providing me with unfailing support and continuous encouragement throughout my years of study. This accomplishment would not have been possible without them.

May God Almighty bless you all.

Chapter 1: Introduction

1.1 Background

Globally, farming is one of the sources of livelihood around the world, especially in developing countries. It plays a crucial role in the economy's growth since it is considered as an important industry in the global market. Growth in the farming sector enables the development of economic conditions and provides ample employment opportunities.

Around the world and mostly in developing countries, farming still resorts to traditional methods and faces a lot of challenges such as climate change, the scarce availability of water throughout the year, and drought (Amandeep, 2017). All this poses a major problem and leads to poor yield and low productivity. The recent adoption and implementation of scientific methods in farming are leading to radical innovation in the production of crops, due to improved farming techniques with high efficiency (Amandeep, 2017). Many farmers are adopting new technologies such as the Internet of Things (IoT) to overcome their challenges. As a matter of fact, of the various benefits that the Internet of Things offers, its capability to innovate and improve the traditional farming methods is revolutionary.

The Internet of Things is gaining popularity in research across the globe especially when it comes to modern wireless communications (Weber, 2010). Internet of Things is a set of uniquely identifiable devices, things, and their respective virtual representations in Internet-like structure which was proposed in the year 1998 (Suo et al., 2012). According to Weber, R.H (2010), the Internet of Things was discovered by Kevin Ashton in 1999 about supply chain management. Nowadays, there is a proliferation of IoT devices for normal users and has laid to the development of various IoT environments such as e-health, smart house, and smart education. Moreover, IoT is being used these days in smart grid, business management, smart logistic, manufacturing, smart transportation, smart cities, and smart farming (Ashton, 2009). Farming is one of the areas where a rapid change and constant innovations are coming up with several new IoT products based on research and innovations to improve activities by making them more efficient for better productivity since the Farming sector is regarded as the more crucial sector globally for ensuring food security (Nayyar & Puri, 2016).

In Kenyan's farming production, activities such as animal monitoring, plant and soil management, pest control, environmental monitoring as temperature, weather, supply chain management, transportation, control systems management are the basic requirement for efficient productivity. IoT based farming system reduces the burden on farmers by creating and adding high value in terms of quality and increased production. With data collected via smart sensors on the field and the integration of Big Data analytics to smart farming equipment, enable efficient use of water and time; production wastages could be reduced considerably while farmers would be able to improve their productivity and profitability (Nayyar & Puri, 2016). Several studies have proposed diverse and integrated IoT based solutions to take care of and resolve all factors affecting the productivity of farming at every stage like cultivation, water management, harvesting, and post-harvesting storage. However, the concern about data security and data privacy related to the IoT system has slowed down the adoption of this revolutionary and innovative technology.

One of the drawbacks and big concerns in the use of IoT systems is security and privacy. Amongst the biggest IoT security issues that need solutions, we can point out some such as wireless security, RFID tag security, privacy protection, and information processing security, network transmission security (Muthuswamy & Ganapathi, 2016). The development of wireless sensor networks (WSNs) has considerably contributed to the popularity of the IoT system. The wireless sensor network provides new ways of sensing information from the physical environment. When connected, these devices provide significant and reliable, precise, and deep information about the phenomenon being observed. Due to its importance and its multiple applications, WSN could be connected to the Internet and managed through a cloud system.

However, the integration of WSN to the Internet brings with it several security challenges (Muthuswamy & Ganapathi, 2016). These security challenges are noticed in all fields where IoT is applied and smart farming is not exempted. According to the US Department of Agriculture (USDA) and the Federal Bureau of Investigation (FBI), the farming sector is getting more exposed to cybersecurity threats since farming is relying more and more on digitized data. While smart farming enables cost reduction and improves productivity, farmers should be aware and have a good understanding of cyber threats and risks associated with their data and ensure that appropriate measures are put in place to secure sensitive data.

With the growing popularity of IoT systems that sense and compute smart farm data, including data related to weather, soil content, temperature, humidity, and past crop yields, it is without a doubt that the farming sector will face an increased cyber targeting. A recent example of US government-authorized big data analytics proves the important value of aggregated farm-level data in tracking and even anticipating crop availability and pricing. Likewise, cyber attackers can steal analyzed farming data or access sensitive data with the evil intention of exploiting resources and market trends. Apart from theft, farm-level data could also be exposed to data destruction through ransomware. Indeed, there is a significant increase in the use of ransomware as a threat to individuals and businesses (Federal Bureau of Investigation, 2016). However, the emergence of new technology such as blockchain (BC) and decentralized VPNs offers new opportunities to improve security and privacy in IoT systems.

Traditional security protection mechanisms are mostly centralized which has challenges in scaling up to meet the security demand of IoT. With the encryption process behind it, blockchain can provide distributed solutions to ensure security protection for IoT. The convergence of blockchain and IoT is an opportunity that can help to build and secure smart farming to gain more effective and efficient use of our resources. The blockchain is a technology that provides security in communication amongst the IoT devices.

Moreover, it offers a decentralized, distributed, and available shared ledger that can store information on the blocks that are processed and verified in an IoT network. The data stored in the shared ledger is based on the Peer-to-peer topology. The blockchain is a technology that enables transactions to be triggered in the form of a block in the Blockchain amongst IoT nodes. The blocks are connected and every device has its previous device address. The blockchain can be integrated into the framework of IoT and Cloud computing. In the future, the blockchain would revolutionize IoT communication (Reyna et al., 2018). The advantages of the integration of IoT and BC could be highlighted as follows (Tanweer, 2019):

- Decentralized framework: This eases the implementation of a decentralized system. In contrary to the centralized system, it increases the performance and reduces the probability of failure of the system by spreading the point of failure to several nodes.

- Security: the BC technology provides secure transactions among nodes. It is perceived as an efficient way of securing communication. It enables communication amongst IoT devices in such a way that data security, integrity, and privacy are fulfilled.
- Identification: The IoT devices are uniquely identified with an ID; the same as in Blockchain, every block is uniquely identified. Therefore, Blockchain is considered as a trusted technology that provides secure authentication and data integrity in a distributed ledger.
- Reliability: In the IoT network system, nodes in Blockchain have the capabilities to authenticate the information passed in the network. The data is reliable since it is verified before storing it in Blockchain. Only verified blocks can enter in the chain.
- Autonomous: with Blockchain providing a decentralized system, IoT devices can freely communicate with each other in the network.
- Scalability: with Blockchain, the IoT nodes can communicate in high-available and share data in real-time across a distributed smart network that connects with the destination.

Besides, Decentralized Virtual Private Networks (VPNs) technology has grown in interest since they have enabled possibilities such as the creation of distributed and collaborative environments, such as homes, enterprises, and universities. It's important to point out that, there are different types of VPNs: centralized, decentralized solutions. In a centralized system, VPN server(s) are provided and managed by a single node in the network; whereas in a decentralized approach, more than one node provides and manages the VPN infrastructure. The VPN creates a virtual tunnel of a local area network through a Wide Area Network (WAN) by providing secure data exchange tunnels amongst nodes. VPNs ensure data integrity, via encryption, to enterprise network resources from remote and insecure locations, connecting distributed resources from multiple sites amongst the IoT system. The combination of the two technologies BC and decentralized VPNs can bring a significant improvement to the security and the privacy issues of today's IoT system. This paper aims to focus on the technical challenges in blockchain and VPNs based security protection of the IoT system.

1.2 Problem Statement

The Internet of Things is becoming significantly important in our society. It opens the door to new technological opportunities such as smart homes, smart cities, smart supply chain, smart

farming, and other variety of smart environments. However, at the same time, the popularity and the use of IoT pose security risks. The vulnerabilities of the IoT system have been demonstrated by the massive DDoS attacks caused by the Mirai Botnet in 2016 and many other variant attacks such as Satori botnet which exploit several vulnerabilities found in smart cameras and home routers.

IoT is on the way to become much more complex and reliability and trustworthiness in providing services could not be done securely in the current centralized architecture. This implies that the number of cyber-attacks will increase rapidly since most of the IoT platforms are not well secured enough (Kennedy, 2017). According to a 2018 Symantec study, there was a 600% increase in the number of IoT attacks between 2016 and 2017.

Most IoT solutions rely on mobile apps and cloud to provide and manage services. Such design widens the attack. In the context of smart farming, the IoT devices used to collect a huge amount of data are vulnerable to numerous cyber-attacks and are exposed to numerous threats undermining the privacy and the integrity of sensitive data.

The emergence of new technology such as blockchain and decentralized VPNs provides solutions for the IoT system. Therefore, there is a need to further investigate the security aspect of IoT especially related to smart farming, and develop new solutions to ensure the resilience against advanced IoT based attacks. The goal of this study is to address the diverse security challenges of the IoT local network by using blockchain and decentralized VPNs technology.

1.3 Aim

The purpose of this study is to build a hybrid IoT architecture for securing smart farming specifically for smart agriculture using blockchain (Hyperledger-Fabric) and decentralized VPN (OpenVPN).

1.4 Specific Objectives

This study aims to design and implement a secure IoT network for smart farming using blockchain and decentralized VPNs. The specific objectives of this paper are as follows:

i) To investigate the current vulnerabilities of the IoT network.

ii) To investigate the current architecture solution for securing IoT network.

iii) To build a secured hybrid IoT smart farming architecture specifically in the context of smart agriculture using blockchain and decentralized VPN.

iv) To test the proposed hybrid architecture against common attacks.

1.5 Research Questions

i) What are the current vulnerabilities of the IoT network?

ii) What are the current architecture solutions for securing IoT networks?

iii) How can the use of integrated blockchain and decentralized VPN be designed to improve IoT security in smart farming?

iv) To what extent the proposed hybrid architecture is resilient against common attacks in the context of smart farming?

1.6 Justification

There is a growing interest in research on IoT for its potentiality of automating and improving processes and the most concern with IoT application are security and privacy. Despite its proliferation, IoT devices remain vulnerable to several security attacks. This study intends to investigate how efficient can be blockchain and decentralized VPNs technology in securing IoT local area networks applied to smart farming. The significance of the study is that through this paper if it is proved that blockchain and decentralized VPNs technology significantly improve security and privacy in a smart farming network that could be useful for anyone who would like to invest in technology to optimize the use of resources and secure the production to gain competitive advantage and increase productivity and profitability.

Moreover, the result of this study could help in the adoption of the effective use of blockchain and decentralized VPNs in other smart environments. Furthermore, the result of this research can be used by policymakers to develop standards and policies about data security and privacy.

Besides, the research could be of value to the body of knowledge by filling the gap existing in the literature on the specific application of blockchain and decentralized VPNs technology in securing smart farming. Therefore, other researchers, practitioners, and consultants might use this study to get an insight on blockchain and decentralized VPNs technology as a security

solution for the IoT system and would find it suitable as a source of knowledge and a basis for further research.

1.7 Scope and Limitation

The study specifically focused on securing IoT local area networks in the context of smart farming using Hyperledger Fabric and OpenVPN. This study excludes the mining and cryptocurrency aspect of BC. The research only exploits cryptographic, distributed ledger aspects of the BC technology. Moreover, this paper does not consider physical security either the energy consumption performance of the system.

Chapter 2: Literature Review

2.1 Introduction

In this section, we present the concept of the Internet of Things and its application in smart farming; then the security challenges related to the use of IoT are exposed. Furthermore, the research reviews the current security solutions for IoT platforms. Besides, a review of the literature on blockchain technology and Hyperledger fabric is performed and this section ends with a review on VPN technology.

2.2 Internet of Things (IoT)

Internet of Things (IoT), also known as Internet of Everything (IoE) refers to the computing devices that are web-enabled and have the capability of sensing, collecting and sending data using sensors, and the communication hardware and processors that are embedded within the device. IoT also refers to devices interconnected via a network infrastructure which can be of different types such as wired network or wireless network. According to Akhil (2019), The Internet of Things (IoT) system is a system of linked computing devices which can be digital or mechanical machines, or objects, or living beings or people that have unique attribute or identifiers and can transfer data across a network without requiring any human intervention or computer-to-computer interaction

Moreover, The Internet of Things (IoT) has been defined as a dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where virtual and physical Things have identities, virtual personalities, physical attributes, and are seamlessly integrated into the information network, use intelligent interfaces and often communicate data related to users and their environments (Smith, 2012). In other words, IoT can also be regarded as a platform or an architecture which enables intercommunication between connected devices and provides a way of managing them.

The development and the technological improvement observed in wireless sensors network devices that were able to interconnect through a network and share and perform various analysis on data has enabled new opportunities and various engineering application in our modern society (Jayavardhana et al., 2013). It appears clearly that IoT is not a single technology but an

integration of several technologies such as information technology, data analytics, communication, and electronic sensor technology that work together to provide an IoT platform.

2.2.1 Internet of Things: Characteristics

The IoT provides the ability to exchange information amongst the connected devices in the heterogeneous network. According to Ramachandran and Krishnamachari (2018), an IoT platform is composed of the following components:

i. End devices: They are also called IoT nodes. Each node is uniquely identified in the network via an Ip address that can be either ipv4 or ipv6. All the IoT nodes can communicate and share data amongst themselves. Example: sensors, actuators.
ii. Network infrastructure: it is the structure that controls the flow of data and establishes the connection and selects the shortest route among the IoT nodes. The infrastructure can be based on a wired network or wireless network using wireless sensors network devices.
iii. Gateways: gateway functions are performed by routers that provide the connectivity to the internet. The gateways are used as devices that protect the network and connect the IoT end devices to the cloud platform through a wide area network or internet.
iv. Cloud platform: It serves as a backend and storage. It provides several functions such as data computation and data analytics. Figure 2.1 shows the layered architecture of IoT application.

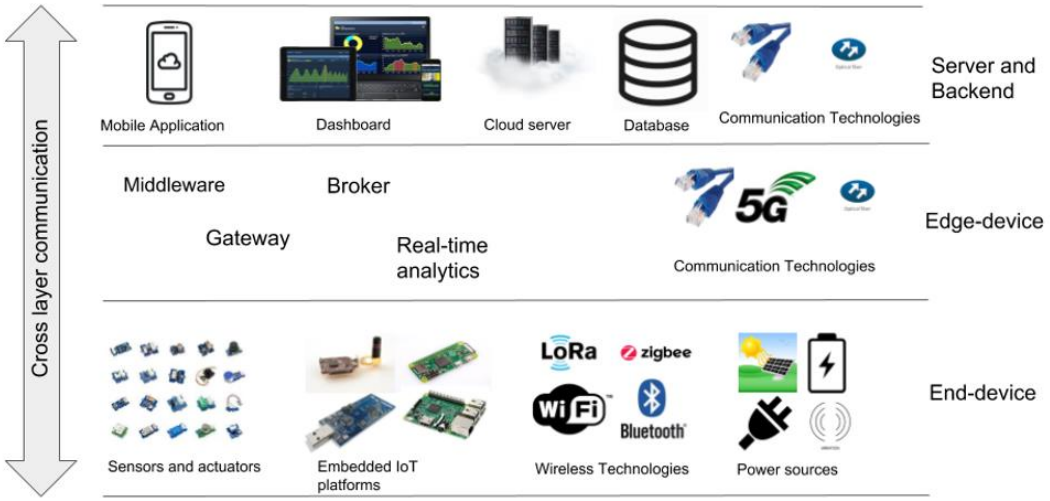


Figure 2.1: Layered Architecture of IoT Applications (Ramachandran & Krishnamachari, 2018)

2.2.2 IoT and Smart Farming

Internet of Things has a strong backbone of various enabling technologies Wireless Sensor Networks, Big Data, Protocols enabling communication, Embedded Systems, Cloud Computing, Internet and Search Engines, Security Protocols, and Architectures, web services, (Nayyar & Puri, 2016). With the increasing implementation of IoT platforms by several industries such as construction and even urbanism, the big benefit or contribution is seen to create and integrate everything in an Intelligent and Smart environment. Even the farming industry is embracing and integrating IoT technology and as a result that has led to the expansion of smart farming. IoT offers several benefits to farming such as:

a. With the use of sensors, IoT eases the data collection and management and with the integration of cloud computing services like cloud-based storage, field maps for Agriculture, etc., data is available and accessible in live from anywhere and can be live monitored.
b. The IoT has been improving performance and efficiency levels in terms of usage of productivity, automation of Water management, Soil assessment, appropriate pesticides, and fertilizers, etc.

c. The integration of the IoT platform in operations reduces remarkably the costs of production which will as a result increase profitability and sustainability.
d. In Smart Farming, with accurate sensors and smart equipment, farmers can increase food production by 70% until the year 2050 as depicted by experts.

The smart farming with an intelligent security system will be a convenient way to get the security to the field as well as to the grain store.

Several scholars have covered the topic of smart farming and IoT. Nayyar (2016) proposed IoT based smart Sensors Agriculture Stick for Live Temperature and Moisture Monitoring using Arduino with over 98% accuracy in data feeds. However, his approach was a cloud-based solution. This solution doesn't focus on the security aspect of the IoT implementation rather it focuses on the performance of monitoring the field data. Moreover, Amandeep et al. (2017) suggested a design of smart farming with the help of IoT that consists of a remote-controlled vehicle that operates on both automatic and manual modes, for various agriculture operations like spraying, cutting, weeding. The controller keeps monitoring the temperature, humidity, soil condition, and accordingly supplies water to the field. His design does not focus on security aspects related to the use of IoT devices in a smart farming environment.

Furthermore, Vineela et al. (2018) proposed a system to monitor crop-field using a raspberry pi, sensors for soil moisture, humidity, and temperature. By monitoring these parameters, the irrigation system can be automated if soil moisture is low. This solution was a cloud base monitoring system and does not address security issues related to IoT-smart farming. Linsner et al. (2019) have researched Vulnerability Assessment in the Smart Farming Infrastructure through Cyber-attacks. The experiment was carried out through a simulation of several attacks on a ZigBee-based wireless sensor network. Their results showed that WSN-based smart farming infrastructures are vulnerable to several attacks such as delay attack, interference attack, and routing attacks.

Blockchain technology has been applied in smart farming. Lin et al. (2018) propose a trusted, self-organized, open, and ecological food traceability system based on Blockchain and Internet of Things (IoT) technologies, which involves all parties of a smart agriculture ecosystem. Even FAO also recommended that ICT (Information and Communications Technology) e-agricultural infrastructure components are a confluence of ICT and Blockchain technology requirements.

They think that when ICT e-agricultural systems with Blockchain infrastructure are immutable and distributed ledger systems for record management, baseline agricultural environmental data integrity is safeguarded for those who participate in transparent data management.

Besides, Lin et al. (2017) reviewed Blockchain-based concepts associated with ICT-based technology. They proposed a model ICT e-agriculture system with a Blockchain infrastructure for use at the local and regional scale. Moreover, Patil et al. (2018) proposed a lightweight Blockchain-based architecture for smart greenhouse farms to provide security and privacy. The IoT devices in greenhouses which act as a Blockchain managed centrally to optimize energy consumption have the benefit of private immutable ledgers. Also, they presented a security framework that blends the Blockchain technology with IoT devices to provide a secure communication platform in Smart Greenhouse.

Shyamala et al. (2019) proposed a new architectural framework IoT Blockchain Based Smart Agriculture for Enlightening Safety and Security which enhances the performance of security and data transparency. The Proof of Concept for the proposed system is implemented with the Ethereum Private Blockchain network under a genesis block. Moreover, the architecture involved different nodes such as temperature control, pressure control, pH control, Moisture water control, pollution control, smoke and fire control, and CO_2 control. A security mechanism of a public/private key automatically generated per device was used to identify uniquely the IoT devices. However, the proposed solution was also a cloud-based solution and the problem with Ethereum blockchain is that it requires large storage (Seyoung, 2017). Figure 2.2 presents an overview of the current data flow in IoT in smart farming.

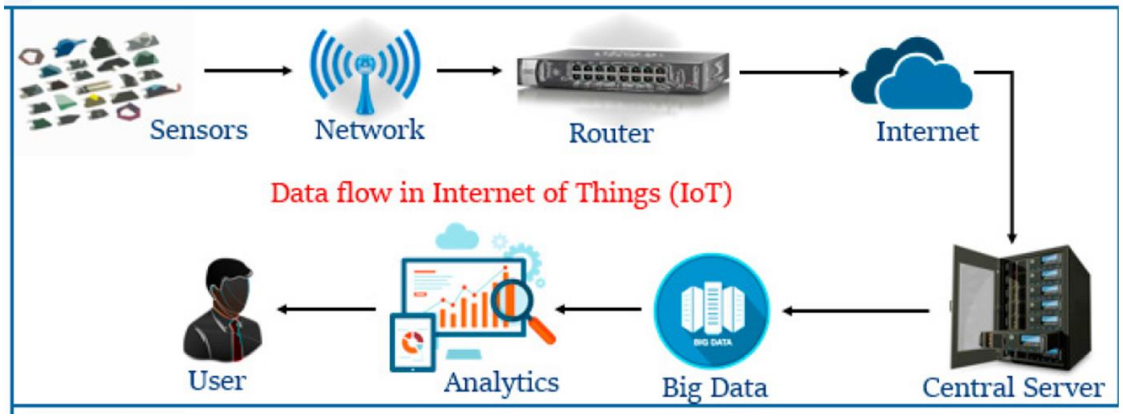


Figure 2.2: Data Flow in the Internet of Things (Nallapaneni et al., 2018).

2.3 Security Challenges of IoT implementation in Smart Farming

Various security risks and threats undermine IoT platforms today. Whether it is a home Wi-Fi network or an organization's network, or smart devices; today's devices and systems pose serious risks. These constant risks are also found in IoT platforms. The fact that IoT is designed on open architecture makes it much more complex and hard to secure, bringing about many more challenges in terms of security (Rizvi et al., 2018). According to Iqbal et al. (2016), there are many properties of IoT when implemented in smart farming that can lead to security and privacy issues such as wireless, embedded use, mobility, diversity, and scale. These characteristics make IoT unique in its security needs and rise several new challenges in the context of smart farming.

According to a report on IoT issues and challenges by the Internet Society (Karen et al., 2015), there are several issues and challenges related to the implementation of IoT in smart farming but most are security and privacy concerns. IoT platforms are challenged by the lack of standards and metrics in identifying security in IoT devices. Moreover, the lack of an optimally controlled role in communication models exposes the IoT environment to hijacking and cyber-attacks. The issues of upgradeability and maintainability weaken most of the IoT devices. Apart from the technical aspect, IoT software deployments are done without security laws and there is a lack of informed decisions over the cost-benefit analysis of IoT investment. The other concern is privacy. It is noticed that there is not only a lack of strict rule against data collection and use but also a lack of multi-party models that enable enforcement and transparency. Furthermore, data collected by IoT devices are not protected efficiently since by their nature they are limited in resources to develop integrated trained privacy principles.

2.3.1. Points of Vulnerabilities of IoT in Smart Farming

The IoT environment is composed of several components and layers that could be vulnerable to attacks. Figure 2.3 presents an overview of points of vulnerabilities in the IoT platform.

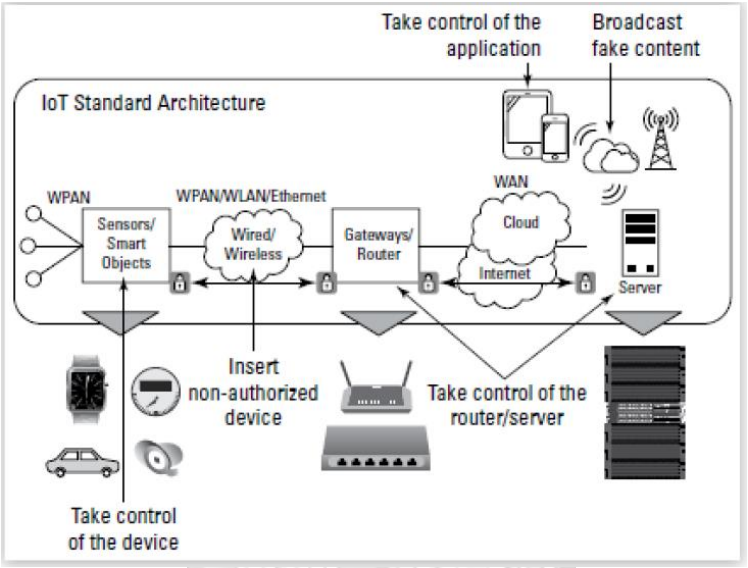


Figure 2.3: IoT Points of Vulnerabilities (Miller, 2016)

As shown above, the IoT vulnerabilities are not only hardware related but also software and application-related (Sulkamo, 2018). The transport layer weaknesses of the IoT environment are caused by the lack of strong encryption enabled for sensitive data, the use of old protocols, and certifications with weak ciphers. Moreover, the problem of shared default secret, very crucial in the case of a cloud-based IoT platform that consists of using the same user password and encryption key for new users presents security risks. All these aspects cause plenty of vulnerabilities risk in IoT platforms (Praetorian, 2017).

2.3.2 Architecture and Layers

The IoT devices are considered to have three different operational layers such as the perception layer, network layer, and application layer. Each layer has unique threats. Since layers are interconnected, the security measure should cover and secure all the layers (Rizvi et al., 2018).

Perception Layer: The collection of data is performed at the perception layer. It is crucial to protect this layer since it plays the role of sensing, collecting, and inputting huge and sensitive amounts of data and the attackers can easily compromise them. Several security measures can be

implemented at this layer such as cryptographic elements and detection of abnormal sensor readings (Yang et al., 2017).

Application Layer: The lack of an accepted standard for the development of this layer makes this layer the most complicated and diverse of the IoT architectural layers. Data access and authentication controls are amongst the most challenging at this layer. It also presents issues such as data protection and recovery. Besides, the lack of good security practices and standards by software developers can expose their applications at risk (Swamy et al., 2017).

Network Layer: This layer controls data transmission; therefore, it also suffers from the common security issues. Amongst the common security challenges in this layer, we can point out unauthorized access such as integrity damage, confidentiality compromise, and eavesdropping, Man-in-the-middle attack, virus invasion, DoS attack, etc. (Zhao, 2013).

2.3.3 Threat Vector

A threat vector refers to a means for an intruder to infiltrate into the IoT platform and attempt unauthorized actions to break devices, or systems (The OWASP Foundation, 2014). Several threat vectors undermine the IoT platform, including physical threat, embedded security, identity management, dynamic binding, and storage management.

Communication Attacks: they are performed over the IoT network. These attacks include, but are not limited to flooding attacks, spoofing MITM, DoS, and Distributed DoS (DDoS), and network injecting (Rizvi et al., 2018).

Physical approach Attacks: These approach attacks are restricted to compromising the network by accessing physically the system either through the wireless medium or a wired or by accessing physically the devices. As physical attacks, we can point out the Radio Interference, Reverse Engineering, Jamming, and Tampering. It consists of accessing and misusing physically the devices for unauthorized actions to compromising the device to either block, record, or transfer of data. These attacks are performed at the layers one and two of the TCP/IP model.

Application approach Attacks: Any device running software is exposed and vulnerable to Application threats. Several tactics such as source code misconfiguration, malicious scripting lead inevitably to security breaches. As far as IoT software deployment is concerned, they are

mostly implemented through APIs and web applications (Dorsemaine et al., 2016). It is very vital to put in place strong security measures to protect this application layer since it is exposed to more opportunities and possibilities for attacks. Furthermore, the information handled in this vector is highly sensitive and needs to be stored and transferred more securely.

Exploitation of a Misconfiguration: for IoT applications to work properly, the appropriate configuration of all the components involved is more than crucial. Therefore, each of these elements of the system needs a correct and secure configuration. Otherwise, they are easily exposed and attackers can without difficulty exploit the vulnerabilities by a compromised actor (Rizvi et al., 2018).

2.3.4 Trust in IoT

In an IoT environment, it is difficult to explain what trust means. Most IoT users don't care about security. As long as the device is effective for the desired purpose and is cheap. Most expensive IoT devices are usually safer; however, users cannot afford them in most cases. This attitude of giving priority to value rather than security poses a challenge in the IoT platform. Trust in the IoT environment can be divided into 3 distinctive security categories such as availability, reliability, and privacy.

Availability: availability is one of the characteristics IoT users seek; IoT devices must be and power-driven to fully finish the intended actions. We can point out 3 distinctive options: Always Off, On, Low Power (Dorsemaine et al., 2015). Also, IoT devices have to be compelled to be updated more. A proper recommendation for this is to make available to all IoT devices, a standard updating platform within a particular domain. The Makrov Model provides solutions to keep systems up with redundant power management and control for IoT platforms (Homeland Security, 2016).

Reliability: ensuring the data that is shared and received is genuine could be a good thing involving the huge number of devices on the IoT platform (Dorsemaine et al., 2015). Availability, confidentiality, and integrity are the utmost crucial characteristic to take into consideration when implementing a reliable IoT architecture. Therefore, reliable and efficient communication is key.

Privacy: for business operation performance, IoT devices gather relevant data that is required for analysis. Usually, these data could be tangible, communications, medical, or net browsing history. Whereas this appears to be easy, quick, convenient, and cheap, it poses security challenges. For instance, firms might sell user's data to other organizations or the corporate might not have implemented enough control measures to secure user sensitive data (Rizvi et al., 2018).

2.4 Blockchain Technology

Blockchain technology is a data structure that provides immutability. It is used to guarantee the trust of financial transactions between different non-trusting parties using peer to peer technology. It leaves out the need for a third financial party. Blockchain is a combination of blocks that contain data. Each block holds a hash of the information of the previous block. The blocks are digitally signed to ensure data integrity and chained after the consensus process performed by all the nodes in the network. Moreover, the blockchain offers a digital and distributed ledger across the network, in other words, the content of the blockchain is the same on all the nodes in the network (Selimi et al., 2018). As shown in Figure 2.4, the concept of blockchain can be compared to a linked list, where each data structure is tied to the next one via the use of a pointer (Restuccia et al., 2018).

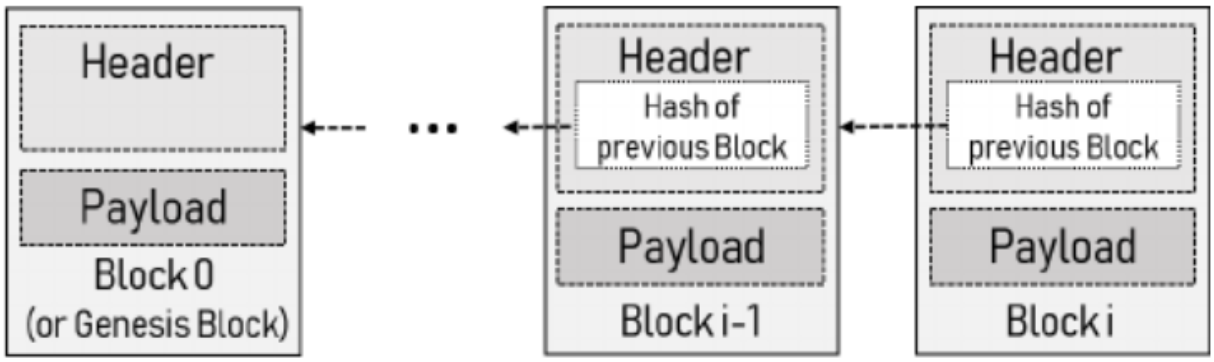


Figure 2.4: The Structure of Blockchain (Restuccia et al., 2018)

The blockchain platform is composed of three main components:

- Network nodes: The network architecture of the blockchain is a set of autonomous nodes that generate and write validated transactions into the digital and distributed ledger without a centralized authority or a trusted third party. The consistency of the digital ledger is achieved through the collaboration of all the nodes in the network. This collaboration is performed via the execution of a mechanism called consensus whereby all the nodes agreed on how to update the blockchain by validating the same set of transactions.
- Ledger of transactions: It is a set of blocks that contains a set of validated transactions, a timestamp, and a link to each other forming then a chain of blocks. This linkage is based on the cryptographic hash which is a mathematical algorithm that maps data with a set of different size strings. For instance, a set of transaction A, B, C and D are hashed H(A), H(B), H(C), H(D); Then aggregated into successive hashes H(H(A)| H(B)), H(H(C)| H(D)) to form a Merkle tree. Finally, the top hash also called Merkle tree root is integrated into the block header (Restuccia et al., 2018). Figure 2.5 shows an overview of the Merkle Tree Connecting Blocks.

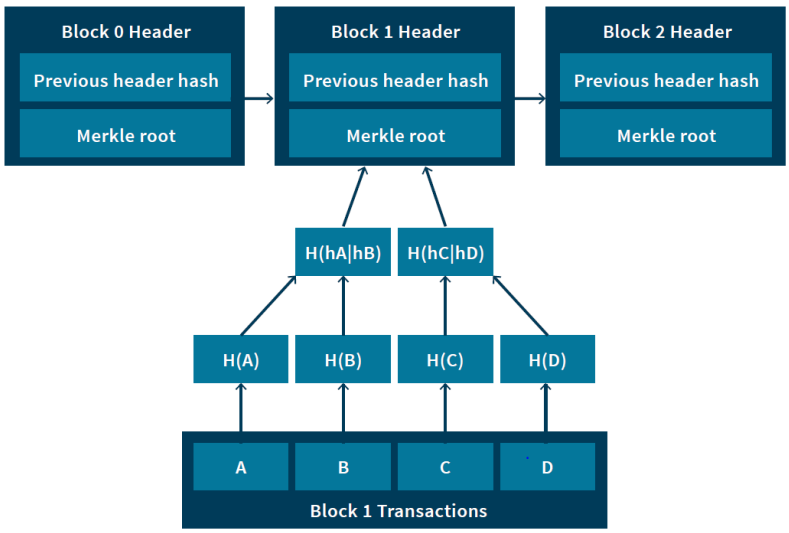


Figure 2.5: Merkle Tree Connecting Block Transactions to Block Header Merkle Root (Blockchain / Distributed Ledger Working Group, 2018).

- Distributed database: As new transactions come in, the ledger is built and updated on every node thus each node in the network holds its copy of the database, therefore, it can

retrieve the history of any data. Depending on the type of the blockchain platform, the required storage capacity will vary.

2.4.1 Consensus Mechanism

The goal of the blockchain network is to provide peer-to-peer data exchange which is verified, stored into blocks, and integrated into a distributed ledger (Nakamoto, 2008). To achieve this purpose, a decentralized consensus algorithm is performed to determine how and when a set of transactions can be stored in the ledger. The consensus mechanism is at the heart of the blockchain. When the consensus among the majority of nodes in the network is reached about a new block, it is broadcast over the network, then append to the ledger. These mechanisms ensure that the majority of nodes possess the same copy of the blockchain through the creation of several consistent copies of the leger (Restuccia et al., 2018).

The purpose of distributed consensus is to maintain the correctness and consistency of the digital ledger on the majority of the nodes. It is a good measure against malicious nodes that can compromise the integrity of the ledger such as modifying retroactively transactions or performing unpermitted transactions.

There are three main implementations of the consensus mechanism in a blockchain platform:

i) Proof-of-Work:

It is the most widely known since it is implemented in Bitcoin and Ethereum platforms. In PoW, the generation of a block is based on the proof of a computationally hard task which is a cryptographic difficult puzzle on hash functions. When a node called Miner generates a block with its PoW, it must submit it to the other nodes for validation. When the block is validated, the miner receives a crypto monetary incentive (Restuccia et al., 2018). This mechanism provides strong network stability and reliability (Nakamoto, 2008). However, it is expensive in terms of computational resources and energy consumption (O'Dwyer & Malone, 2014). Also, the PoW is vulnerable to the 51% attack. If a malicious node gains control of 51% of the hash power in the network, it could easily perform malicious transactions. These attacks are effective against many cryptocurrencies such as Verge, Bitcoin Gold, and Zencash (Alyssa, 2018). To overcome the fifty-one percent threat, systems are integrating improved security approaches. For instance, the

use of real-time validators can improve the security aspect. That entails that access and control over the network can be possible only if an attacker can compromise almost all peers in the network (Vitalik, 2018).

ii) Proof-of-Stake:

It shares similarities with PoW, such as incentive upon generating a block successfully. Nevertheless, not all nodes can participate in this consensus mechanism. Only nodes deterministically chosen based on accumulated wealth can participate (Vasin, 2014). This mechanism is implemented in Peercoin. Similarly, the Proof of Importance (PoI) takes into consideration the stake as a vital metric; however, it considers other metrics that measure the miner's participation in the network, such as the amount of transactions (Restuccia et al., 2018).

iii) Byzantine Fault Tolerant algorithm:

It intends to prevent attacks and software errors that lead to faulty nodes with arbitrary behavior. It ensures that consensus is reached and not compromised despite the participation of a malicious node. This approach's disadvantage is the scalability limit in terms of the number of peers participating in the network. To improve this aspect, Practical Byzantine Fault Tolerance (TBFT) was proposed. This consensus mechanism is implemented in Hyperledger Fabric and Riplle.

2.4.2 Smart Contract

The crucial struggle of the IoT platform is to enable autonomous control and self-organized peer-to-peer transactions. Therefore, it is of preponderant vital to come up with management solutions that enable automated interactions; such that individual control and verification of the trustworthiness of each interaction/communication won't be needed. The above issue is complex, definitely not trivial, and exacerbated by the fact that there is a huge amount of interconnected devices with a heterogeneous design (Restuccia et al., 2018).

It is important to point out that the fore mentioned issue is not specific to the IoT platform; however, it conjointly impact any network systems and architectures where there is a lack of centralized parties that operate centralized management and control and that demand automated and self-organized protocols. One best case is the blockchain platform, an architecture where

distributed parties are needed to reach consensus in an automated way by running complex code locally. In this context, the smart contract has been proven to be effective to overcome the problems mentioned above. In a nutshell, smart contracts are software programs that control and dictate contracts among 2 or several entities.

There are various benefits of the smart contracts, and their contribution to IoT networks is considerable. Primarily, by the fact that contracts are kept within the blockchain, it provides and ensures data integrity among parties since it can not be altered or corrupted when its integration within the blockchain. Secondarily, every contract is assigned a unique address within the blockchain and could be remotely and directly accessed from the internet, therefore creating smart contracts well-adapted to be accessed by remotely IoT systems. Finally, contracts are a set of lines of code understandable and executable by IoT devices.

IoT and Blockchain share several common points and given the fact that smart contracts have been successfully and effectively implemented in the blockchain it is therefore without doubt that smart contracts could provide new opportunities in IoT applications to enable automation and self-regulatory systems transactions.

Though the appliance of smart contracts to the IoT continues to be investigated, outcomes prove that many IoT implementations could take advantage of the smart contracts within the blockchain technologies. For instance, its integration and deployment to access management mechanisms which control the access to the platform have been demonstrated profitable and efficient (Lin, Huang, & Choo, 2018).

These works highlight the blockchain capability to come up with real-time access management mechanisms that additionally control and describe policies in terms of access to system resources. Moreover, it has been proven that blockchain and its smart contracts feature can also be applied in the smart supply chain. The smart contract can be used not only for transaction control and charges control associated with production and shipment but also can be applied to keep track of their position.

It is important to point out that there are three kinds of blockchain: the public blockchain, the private blockchain also called permissioned blockchain and the consortium blockchain with different properties (Tanweer, 2019). A public blockchain is open to anyone. Anyone can be part

of the blockchain network and transact with other peers in the network and participate in the consensus process. Cryptocurrencies such as Bitcoin, Ethereum, are classified as open and public blockchains. In contrast to the public blockchain, a private or permissioned blockchain is used mostly in an organization network for business applications. It is not open to anyone; only the peers within the organization participate and make transactions in the blockchain network. One of the most known permissioned blockchains is Hyperledger, popularized by Linux Foundation. Hyperledger provides not only an immutable, trusted and distributed ledger but also a cryptographic membership service

2.5 Hyperledger Fabric

As an open-source application, the Hyperledger fabric network provides a platform to implement distributed applications mostly written in Java, Go, or python. This platform can support different implementations of its features such as consensus protocols. The architecture of Hyperledger Fabric includes:

- Peers: Peers are nodes in the network that can be either committers or endorsers. A peer performs the role of endorser when it executes the chaincodes, simulates the transactions, and endorses the result. While a peer assumes the role of committer when it commits transactions into its blocks and keeps and maintains a copy of the ledger. However, it is possible to have a peer committer for specific transactions and at the same time, it plays the role of endorser for other transactions.
- Ordering node: this is node has the responsibility to create generate new blocks of transactions and share them with the peers which then integrate these blocks to their local ledger. It is also in charge of ordering chronologically the transaction by timestamping them. This component is crucial since it is where the process of consensus such proof of work is performed. It can implement as a decentralized service.
- Chaincode: also called smart contract, is a program code installed and instantiated in the Hyperledger network and run by the peer's node in a distributed way enabling communication with the network shared ledger.
- Channel: it can be considered as a higher layer on top of the Hyperledger network that provides confidentiality abstraction. Each channel owns member entities and a set of chaincodes

2.5.1 Hyperledger Fabric Protocols

- Tx proposal: the first step in Hyperledger fabric network communication, a client submits a proposal for a transaction to be integrated into one of the blocks. This is performed via an application that uses an SDK's API such as Java, Go or Python).

- Endorsement and Tx simulation: at this step, the proposed transaction is broadcasted to all the endorsing peers in the network. Then the Tx proposal is checked for its correctness in terms of uniqueness, membership, permission status, signature, and structure. After the verification is done, the transaction simulation is performed. The endorsers execute a chaincode according to the specific arguments mentioned in the Tx proposal. This results in output against the current state of the ledger. Then, the output is transferred back to the client in the form of a proposal response via the SDK.

- Verification of the proposal response: the client collects the responses and cross-checks in terms of endorsers' signature and the response's content. In case the response is the same and complies with the endorsement policy, the client can then submit this Tx to the ordering node.

- Tx submission to the ordering node: The Tx proposals are collected by the ordering node through several channels and ordered according to their receiving time. Then the Txs are stored in a block specific to a channel.

- Tx validation and commit: the block is sent to all the peers of the specific channel which then check the validity of the block in term of endorsement policy. In case of validation, the Txs are tagged and the block is appended to the chain maintained by the peers of this specific channel.

- Ledger update notification: after the update of the ledger, a notification is sent to the clients that submitted a Tx proposal, specifying the validity or invalidity of the that was appended in the latest block.

Figure 2.6 portrays the sequence of transaction execution in the Hyperledger platform.

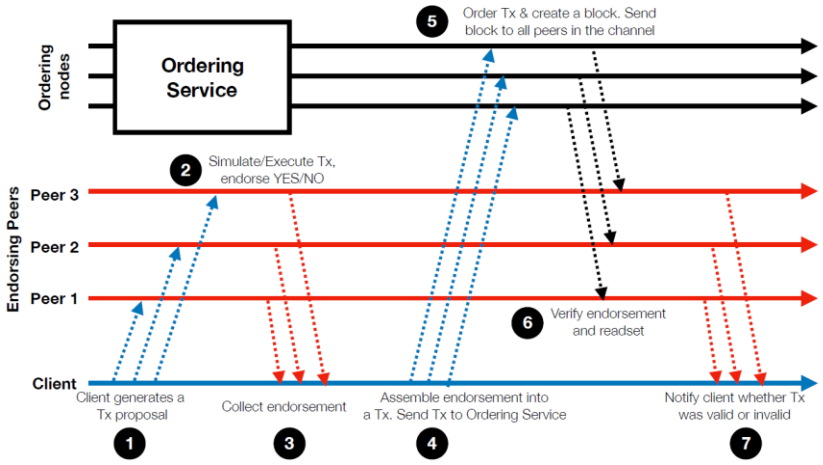


Figure 2.6: Sequence of Transaction Execution in Hyperledger Platform (Selimi et al., 2018)

Sousa et al. (2017) proposed a design, implementation, and evaluation of a BFT ordering service in the Hyperledger fabric network based on the BFT SMART state machine replication/consensus library. The research showed that the ordering service of Hyperledger Fabric can write a transaction in the blockchain in less than half second and perform ten thousand transactions per second even with nodes distributed across different continents. Seyoung (2017) presented a platform for managing IoT devices using Ethereum. His study showed that the Ethereum network has a fast time transaction around 12 seconds but not fast enough for some domains and it required large storage.

Moreover, Sherer (2017) compared the private blockchain and the public blockchain in terms of performance, scalability, and security of the two approaches. His result showed that private blockchain fit better for business application and has more potential in terms of scalability and security.

The Blockchain (Hyperledger Fabric) technology has the potential to overcome the aforementioned IoT challenges as a result of its distributed, secure, and private nature (Nallapaneni et al., 2018). The BC-IoT integration approach provides numerous opportunities such as building Trust between parties, reduce the cost, reduce time, and provide security and

privacy (Tanweer, 2019). While Blockchain technology provides a decentralized and distributed architecture for the IoT system improving the security and privacy; an integration of VPN technology would additionally increase security and privacy features since it provides secure communication between devices through data encapsulation and encryption.

2.6 Current Blockchain-based IoT Security Solutions in Smart Farming

The integration of the Blockchain-IoT approach brings a heap of outstanding opportunities in smart farming. This approach opens new avenues for both together. Figure 2.7 presents the overall opportunities for blockchain in IoT-smart farming. The benefits of this integrated approach can be presented as follows (Tanweer, 2019):

a) Establishing Trust amongst parties: The Blockchain-IoT with its security features will enable trust between the numerous connected devices. Only authorized nodes are allowed to share data within the network, each block of the transaction must first be controlled and validated by the endorsing peers then they can be stored in the digital ledger.
b) Saving Cost: The Blockchain-IoT approach will drop consistently the cost since it is based on peer to peer design that doesn't need a third party to communicate directly. It removes the third-parties' peers between a sender-node and a receiver-node. It offers a peer to peer communication.
c) Saving Time: it enables efficiency by reducing operations time. It saves the transaction time from hours to second.
d) Providing Privacy and Security: this approach offers privacy, confidentiality, therefore, security to the data and devices of the system.

Figure 2.7: Opportunities in Blockchain & IoT (Tanweer, 2019)

2.6.1 Cloud-based IoT Blockchain Network in Smart Farming

Current security solutions to secure IoT platforms in smart farming are cloud-Blockchain based. In cloud blockchain network deployment, mining and transaction are hosted in the cloud. The nodes could be enterprise servers, computers, phones, or tablets. The operational devices act as a client and don't store the distributed ledger they interact with the node located in the cloud-based blockchain through APIs. Figure 2.8 shows an example of a cloud-based solution.

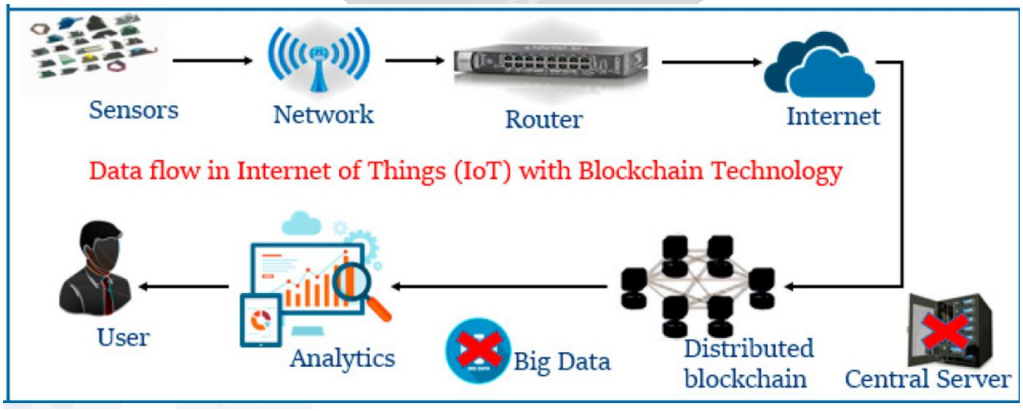


Figure 2.8: Cloud-based IoT Blockchain Network Solution Diagram (Nallapaneni et al., 2018)

In this approach, data is collected by IoT devices and sent to the nodes that participate in the smart-contract transaction in the cloud. In this case, the cryptographic keys are provided to IoT devices to sign their data. However, to get a secure data transfer, a separate agreement of trust

between the transaction nodes and the IoT devices should be put in place (Blockchain / Distributed Ledger Working Group, 2018). In the case of private blockchain service, restrictions could be applied when accessing the mining nodes.

Cloud-based IoT Blockchain services are mostly provided by Internet Service Providers (ISP) that provides API capabilities that simplify the integration of IoT and blockchain service. Moreover, there are solution providers that provide frameworks that run on transaction nodes for quickly building a private blockchain service. Despite that this approach contributes a lot in securing the IoT platform, the fact that it is based in the cloud leaves out the local network insecure and vulnerable to several attacks.

2.6.2 Communication Model in Smart Farming

This solution approach consists of installing the blockchain directly on the IoT devices and /or in the cloud with APIs to the nodes. This approach provides security not only in the cloud but also in the local network. Figure 2.9 shows an example of the communication model.

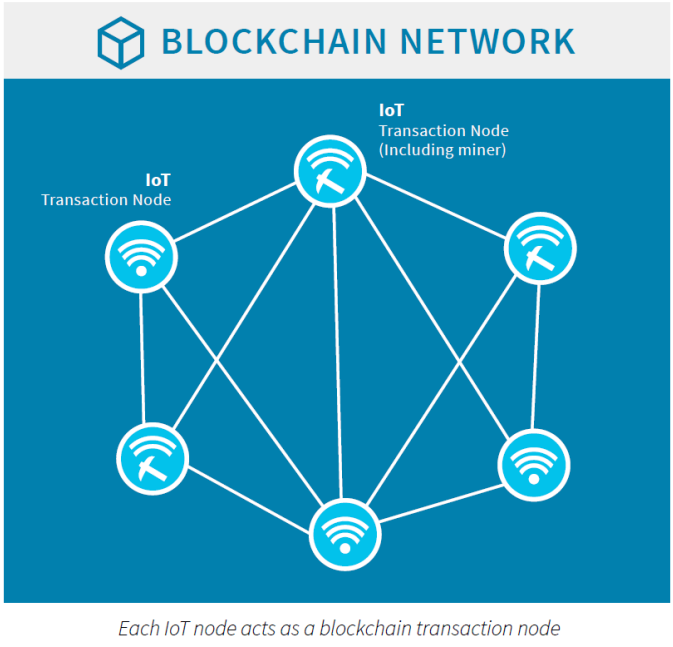


Figure 2.9: Communication Model (Source: Blockchain / Distributed Ledger Working Group, 2018

In this model, each device hosts can participate in the transaction through a self-generated private key. This approach presents challenges related to hardware limitations such as low

processing, small storage, and limited connectivity of current IoT devices. However, the improvement of technology such as Raspberry Pi offers new opportunities to implement such a solution to come up with a better IoT platform security.

According to CSA and Blockchain/Distributed Ledger Working Group (2018), there are five features to consider when securing the IoT platform:

- Scalability of IoT discovery

The adoption of IoT will result in the connection of potentially billions of devices to the internet that must collaborate. Moreover, those devices should be able to discover peers and services to interact. A private blockchain can support the bootstrap and enrollment of IoT devices either in a network. Transaction nodes could authenticate devices before providing a trusted node list. The enrolment credentials can be done either by installing or internally self-generated in the IoT device or provided by the owner or the installing technician of the device. Besides, all communication must be authenticated and encrypted to ensure confidentiality and integrity.

- Trusted Communication

In an IoT platform there is a need for a secure communication channel for data transfer and build and store transactions in the ledger. Therefore, the blockchain service can be used as a public key infrastructure. When a new IoT device is enrolled in the blockchain, a new transaction that contains the IoT properties including its public key is created. While Wallet keys are mostly used in blockchain implementation. Some keys represent identity key that can be used to encrypt traffic (Traffic Encryption Keys) or content (Content Encryption Keys) to secure communication between IoT peers.

- Message authentication

Blockchain offers the capability of autonomous transactions in IoT platform using smart contract functionality. Smart-contract can provide access restrictions based on the identity of each device that enters into a transaction.

- IoT configuration and Updates

As more IoT devices are connected to the cloud, blockchain can provide trust and security in the sense that the ledger can host not only the IoT properties such as the last version of validated firmware and configurations but also it can host the hash value the latest configuration file for each device.

- Secure firmware image distribution and update

The IoT system is regarded as a heterogeneous platform where various devices communicate with each other and users. In this context, therefore, it is vital to ensure that all the applications being run on every device always comply with security policies, regulations, and security requirements in a network such as keeping software up-to-date. Even though the huge number of devices within the system the BC is a well-fitted solution that already offers helpful characteristics that efficiently overcome the issues listed above. By the fact that the blockchain is sustained via consensus mechanisms, it has, therefore, the potential to come up with trusted chains that hold all trusted and up-to-date firm's software; and as a matter of fact, the network nodes could then identify easily and download these updates. IoT devices can update firmware images regularly using blockchain, removing thus the need to have the new image stored on the device. Furthermore, IoT devices can be able to validate each update provided by the vendor just by comparing the image hash against the one for the vendor.

Several kinds of literature on the integration of blockchain and IoT contributed to the development of more secure and privacy mechanisms. Kshetri (2017) has proven that the applications of blockchain offer more security in the IoT platform. Besides, Dorri et al. (2017) researched on the gaps in contemporary security and privacy methods and contributed to LSB, a lightweight and scalable blockchain for IoT security and privacy. LSB's lightweight protocols reduce bandwidth and computation costs.

The benefits and issues of applying blockchain for the IoT are exposed in several kinds of literature. Seyoung et al. (2017) demonstrated how blockchain could help store sensor data using smart contracts. Canoscenti et al. (2016) reviewed the use cases of the blockchain and highlighted the open issue in adaptability, anonymity, and integrity in terms of IoT data storage in a decentralized architecture. Selimi et al. (2018) have implemented the hyperledger network in

a Community Mesh Network. Their study showed hyperledger offers security features to a mesh network.

Furthermore, Fremantle et al. (2017) proposed an architecture to enhance IoT security and privacy using a distributed ledger called namecoin. The research showed that distributed ledger provided by blockchain can guarantee security and privacy for IoT devices. This paper, therefore, intends to investigate the architectural issues and performance implications when deploying blockchain and decentralized VPN for IoT platforms in terms of security such as confidentiality, integrity, and data storage in a local network. The crucial and valuable consensus mechanisms, public-key cryptography, and building blocks distributed ledger of blockchain architecture is an opportunity for the Internet of Things and smart farming management systems.

2.7 Virtual Private Network (VPN)

A virtual private network (VPN) can be defined as a way to provide secure communication amongst members of a group via a telecommunication infrastructure. It is also an extension of an organization's private network to connect remote users over shared or public networks mainly the Internet (Karuna & Indira, 2018). VPNs extend geographic connectivity to telecommuters, mobile users, remote offices, customers, and suppliers who need to connect to the main office. VPN offers secure, dynamic, and low-cost access to private networks while increasing scalability, productivity, and maintain a good level of security. The VPN system uses tunnels. The following figures depict the VPN mechanism:

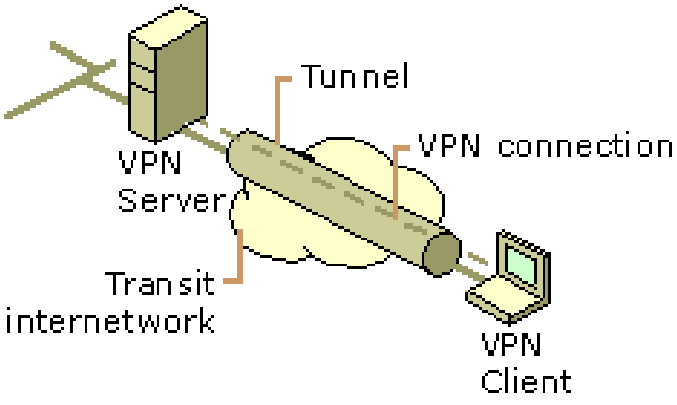


Figure 2.10: Typical VPN Connection (Microsoft Corporation, 2000)

Tunneling is the mechanism of encapsulating a network packet within another. While the encapsulating packet is referred to as the transport packet, the encapsulated packet is referred to as the tunneled packet. The inner packet is encrypted at the data link and network level of the OSI architecture. During the tunneling, an additional IP header is joined to the original header then sensitive data about the tunnel is placed between the two headers. The external header provides the destination and source for the tunneling process, while the internal header gives the original destination and source of the packet (Karuna & Indira, 2018).

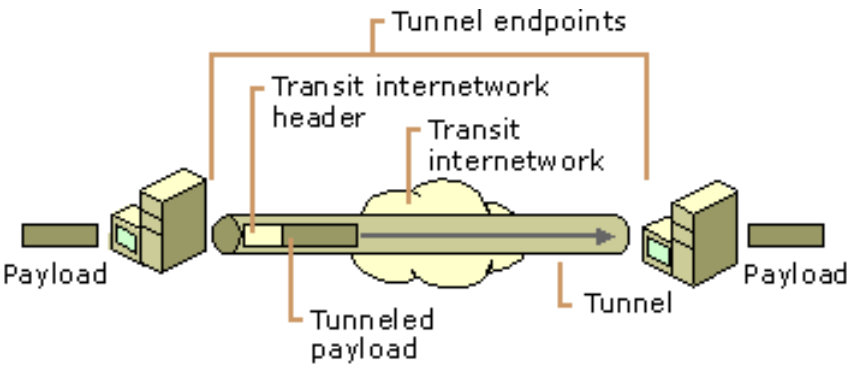


Figure 2.11: Tunneling Mechanism (Microsoft Corporation, 2000)

There are three ways of using VPNs technology:

- VPN used for remote access: this is used by a remote access client. A remote access client is a one-node user who accesses the private network from a remote location. The VPN server is in charge of granting authorized access to the resources of the network to which it is connected. The packets sent across the VPN connection originate at the VPN client. For a successful connection establishment, mutual authentication is needed (Karuna & Indira, 2018).
- VPN used for Site-to-site: A site-to-site VPN enables the connection of two private networks. It provides secure communication between separate offices, or with other organizations, across the Internet (Yurcik & Doss, 2001).
- Extranet VPNs: Extranet VPNs provide a secure connection to the corporate network in a limited way for business-to-business (B2B) communications for the organization's suppliers, partners, and customers.

Furthermore, VPNs are IP-based networks that are most useful to provide security and privacy in communication links thus offers properties like confidentiality, integrity, and authentication.

- Confidentiality: Preserving the privacy of the data exchanged between the two communicating entities using the Encryption process. Secret-key cryptography and public-key cryptography are two main cryptographic schemes that are used in VPN.
- Integrity: Data exchanged over the internet will not be altered during any transit. This can be done either by digital signatures or one-way hash, message authentication codes (MACs).
- Authentication: guarantees that the interacting parties are similar. It is achieved by authenticating a password or by digital certificates.

Moreover, there are various types of VPNs. VPN technology could be classified in several ways based on different standards. However, two main categories emerge. We have secure VPNs and Trusted VPNs:

Secure VPNs have built-in encryption mechanism and other security controls such as integrity and authentication. The communication between the nodes is encrypted based on the notion of a point to point encryption meaning the traffic is encrypted by the sender before being forwarded to the receiver. They assure security. Examples: is L2TP, PPTP, IPsec, SSL, and OpenVPN.

Trusted VPN are VPN provided by Internet Service Providers (ISP). It consists of offering leased paths that connect different sites of an organization. Trusted VPNs assure properties path. Moreover, trusted VPNs can include encryption services thus offering a hybrid solution. An example of a Trusted VPN is Multi-Protocol Label Switching (MPLS)

2.7.1 VPN Protocols

There are several tunneling protocols. They carry PPP datagram over a non-point-to-point network. Amongst the commonly used we can point out:

- *Point to Point Tunneling Protocol (PPTP)*

Developed by Microsoft, the Point-to-Point Tunneling Protocol (PPTP) is the most commonly accepted VPN solution by Windows users. PPTP uses similar authentication forms to PPP (PAP, SPAP, CHAP, and MS-CHAP). PPTP sets up the tunnel but does not have Encryption. To build

a stable VPN, the encryption is achieved using the Microsoft Point-to-Point Encryption (MPPE) protocol. The overhead for PPTP is fairly small, making it the fastest among the different methods used for VPN.

- *Layer 2 Tunneling Protocol (L2TP)*

The Layer 2 Tunneling Protocol (L2TP) has been developed in collaboration between Cisco and Microsoft, merging PPTP, and Layer 2 Forwarding (L2F) protocol functions. One benefit of L2TP over PPTP is it can be used on non-IP networks like ATM, frame relay, and X.25.

- *Internet Protocol Security (IPsec)*

IPsec can be used as a tunneling protocol on its own and is considered the standard VPN solution, especially for a gateway to gateway VPNs connecting two LANs. IPsec works in a packet Network (Packet 3).

2.7.2 Risks and Limitation of VPNs

Within a network, an attacker could exploit misconfiguration and bugs in an end device by using hacking tools to perform an attack such as VPN hijacking and man in the middle attacks. VPN hijacking is performed when an unauthorized device takes over an established connection between two authorized devices and impersonating the client. As far as the man in the middle attacks is concerned, it consists of intercepting, injection, deletion or modification, and redirecting of traffic between two devices. By default, strong user authentication is not implemented in most VPN, therefore they are vulnerable to several attacks. A strong user authentication could provide enough restriction to unauthorized access.

In the case where the end device is compromised by virus or spyware before or during the authentication, the VPN connection will be at risk and there is a high probability that the credentials might be leaked to the attacker. Therefore, it is crucial to have an efficient and up-to-date anti-virus system. Moreover, misconfiguration that grants more access rights than needed can be also a risk to the VPN connection. Also, it is important to keep in mind the interoperability issues between the different implementation of VPN from different vendors. When implementing a VPN, it is important to consider that the type of VPN chosen provides

strong authentication and encryption algorithm. It should support anti-virus, intrusion system, digital certificate, and be able to provide strong default security for administrative ports.

2.7.3 OpenVPN:

OpenVPN is an open-source VPN application. It supports point to point tunnel and encrypted connection. It offers a strong authentication system based on a digital certificate (Iqbal & Riadi, 2019). OpenVPN presents several characteristics and features that make it suitable for security purposes.

i) OpenVPN characteristics:

OpenVPN offers two basic modes, which operates as both a VPN layer 2 or layer 3 tunnel so that OpenVPN can also run on Ethernet Frames, IPX packets and Windows Network Packet Browsing, all of which is a problem in the VPN solution (Purbo, 2013). Besides, users who connect to the VPN Server will make the tunnel and turn the client node network settings, so that network traffic is sent through the tunnel. If the tunnel is established then the firewall from the VPN server will be able to protect the client device connected, even though it is not the local machine. OpenVPN connections can be tunneled through almost every firewall tunnel OpenVPN can work on sites that use HTTPS protocol. Moreover, OpenVPN has proxy support and can be configured to run as a service and TCP or UDP as a server or client. As the OpenVPN server, just wait until the client connection requests, while as a client, it tries to make the connection that corresponds to the configuration (Iqbal & Riadi, 2019).

Apart from that, all regulations, restrictions, and forwarding mechanisms concepts like NAT can be used with the OpenVPN tunnel. It offers many starting points for individual scripts. This script can be used for a variety of purposes from authentication to failover or more.
Furthermore, OpenVPN no longer needs to use static IPs on both sides of the tunnel. The second point is the end of the tunnel can have cheap DSL, access with dynamic IP users will rarely see the IP changes on both sides. Both the OpenVPN server and clients can be in a network that uses private IP addresses only. Any firewall can be used to send traffic to another tunnel. Both installation and usage are very simple especially if you've tried to set up IPsec connections with different implementations. OpenVPN provides a modular design with a high degree of simplicity

both in security and networking. There are no other VPN solutions that can offer different possibilities on the same security level (Iqbal & Riadi, 2019).

ii) OpenVPN Protocols and encryption algorithm:

OpenVPN uses the stable, secure, and lauded SSL/TLS protocols for security and integrates them in its reliability layer. While other VPNs suffer from the complexity of their implementation like IPsec, OpenVPN offers an easier implementation. OpenVPN is based on a modular concept not only for networking but also for the underlying security. It is an OSI layer 2 or 3 full-mesh internetwork tunneling solution. OpenVPN can support several encryption functions such as AES, CAMELLIA, and SEED; however, Blowfish is used by default. Nonetheless, it is recommended to modify OpenVPN's behavior and replace Blowfish by a more recent algorithm (AES-256-CBC). It is worth noting that with OpenVPN, encryption algorithms are, by default, negotiated in the control channel. The default behavior if a client and a server both handle this negotiation would be to use AES-256-GCM or AES-128-GCM, thus a theoretically strong algorithm (Quarkslab, 2017).

Packets integrity is ensured by an HMAC, whose underlying hashing function. The hash functions list available depends on the back end in use. OpenSSL and TLS allow standard functions to be used but also functions known to be vulnerable, such as MD2 or MD4.

iii) Drawbacks of OpenVPN:

- OpenVPN is not compatible with IPsec the standard VPN solution.
- It is not defined by any RFC.
- There is no enterprise-class GUI for administration
- It runs in userspace meaning all network traffic needs to go from the kernel space to user space and back.

Commonly the communication using OpenVPN is based on the client-server model; however, it is possible to have set a VPN communication between VPN clients without a centralized server. Since our proposed architecture is decentralized and each node in the blockchain network should be able to communicate directly with other nodes in a decentralized fashion, a VPN that allows a client to client communication would fit into our architecture.

In definitive, to ensure confidentiality, Virtual Private Network encapsulates and encrypts data before sending the packets over the Internet. The specific protocols employed include IPsec, L2TP, PPTP, SOCKS, etc. Although PPTP, developed by Microsoft and heavily implemented on its legacy operating system, has its limitations and needs the support of an extra protocol, currently IPsec, to be secure. The various protocols operate at different levels of the stack layer model of the OSI protocol and can, therefore, be combined to improve protection in the VPN.

2.8 Conceptual Framework:

Figure 2.12 illustrates the conceptual framework that was used to implement and design the system prototype. As shown, the data in the field is sensed and collected regularly and monitored and controlled; then the data is passed through a process of validation and encryption. Thereafter, the data is encapsulated in a block, forwarded and stored in the distributed ledger. The User can be able to access in real-time the data stored in the ledger through the visualization and analytic tools.

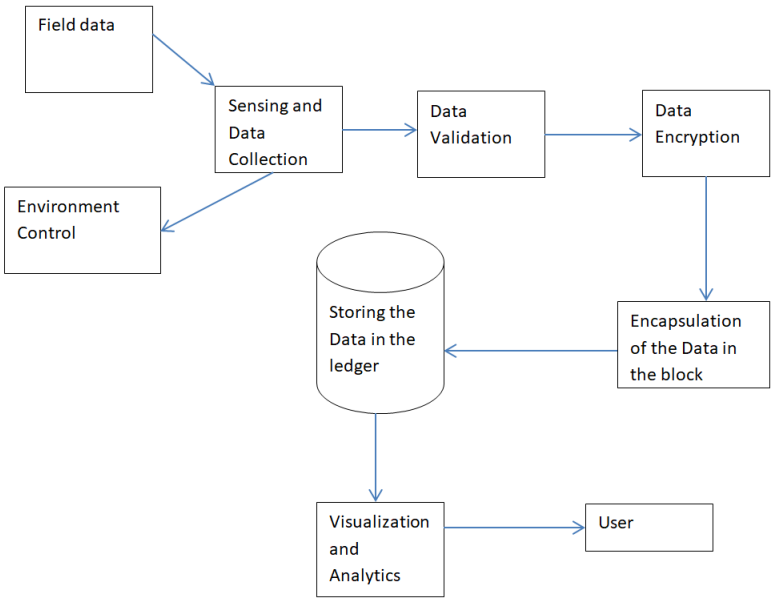


Figure 2.12: Conceptual Framework

Chapter 3: Research Methodology

3.1 Introduction

The research aims at designing and implementing a secure IoT network system using blockchain and decentralized VPN. In this chapter, is discussed the methodology which is used for the research project to achieve the research objectives. The research design, system development, system design and analysis, system implementation and evaluation, and ethical considerations are carried out.

3.2 Research Design

Research design can be defined as a framework of methods and techniques used by a researcher to carry out the research project in a reasonably logical and systematic way so that the research problem is handled effectively and efficiently. It provides insights into how research is conducted using a particular methodology. Kothari (2008) defined the research design as the conceptual structure within which research is conducted. It constitutes the road map for the collection, measurement, and analysis of data. The research design can also be regarded as an arrangement of conditions for collection and analysis of data in a manner that aims to combine relevance with the research purpose (Kombo & Tromp, 2006).

This study has, therefore, adopted the experimental research design since the research covers an emerging area with relatively new terminologies and it might be resource-consuming in terms of finance if implemented in a real farm. The research approach was a practical simulation that involves setting up an experimental smart agriculture environment prototype composed by the sensor that collected the data from the field such as temperature and humidity, and microcontrollers for the automation and control of the environment parameter (humidity, temperature). The Raspberry Pi is used as network nodes and Access points in the architecture while one computer served as node and server for visualization of the data sensed. The simulations focused on the security aspect of the architecture with the deployment of Hyperledger fabric and the virtual private network. Consequently, target population, sample techniques, location of the study as well as data collection and data analysis were not required in this study.

A quantitative approach is used in the sense that the data sensed from the experimental environment prototype is quantified. Moreover, the number of simulations that are performed during the test is also quantified. All the simulations are done in an experimental agriculture environment setup therefore the study is not tied to any location and does not involve any human participant. Therefore, the target population, sample techniques, and location of the study are not required.

Besides, as the research is based on simulations, the dataset is collected from the experimental environment setup and the data collection technique used is experiment since the collected data served as the input for the proposed security system. Then, the collected data is validated, encrypted, and encapsulate in the blocks and stored in the distributed ledger. The stored data can be visualized through the web application.

3.3 System Development Methodology

The system development methodology used in this study is Rapid Prototyping. Prototyping is the process of designing and developing a trial version of the system or its components to clarify the requirements of the system or reveal critical design considerations (Bieman, 2006). Rapid prototyping is used as the development methodology for this research. This method was adopted since it offers two main such us saving cost and time and enabling easy incorporation of changes to the prototype.

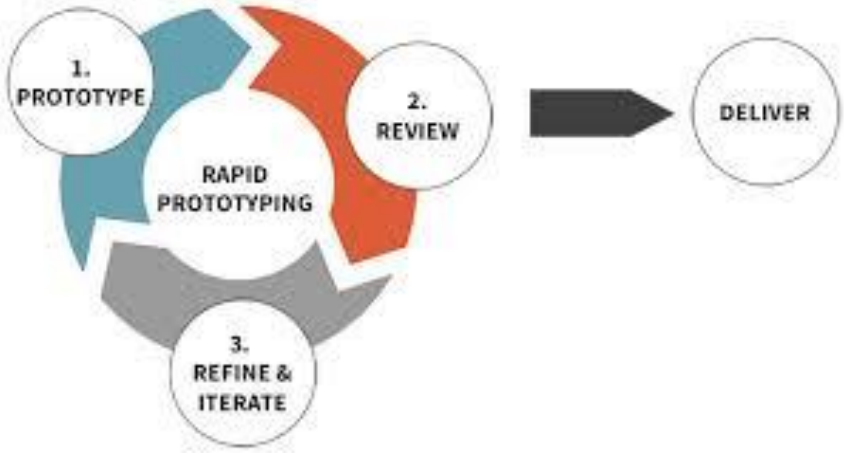


Figure 3.1: Rapid Prototyping Process

3.4 Testing

Upon completion of the implementation of the system, it is important to perform tests to ensure the end product is working logically as expected. In this phase, the prototype is evaluated to ensure it meets the research objectives. In any security design, Confidentiality, Integrity, and Availability (CIA) are 3 key security requirements that need to be addressed (Komninos, Philippou & Pitsillides, 2014). Confidentiality is a security property that assures the data is not accessible to an unauthorized user. Integrity on the other hand makes sure that the sent information reaches its destination without any modification, and availability means that each data, resources, and services are available when it is needed. The testing of the system was not tied to any location since it was performed on the experimental environment setup. Furthermore, the testing consisted of performing and simulating attacks such as network sniffing attacks, password cracking attacks, data tampering attacks and denial of service attack, and reporting the resilience of the proposed architecture against these kinds of attacks.

3.5 Research Quality

According to Kombo (2006), the validity of an instrument is a measure of how well the instrument measures what it is intended to measure. It is the extent to which the data accurately measures what they were intended to measure. The reliability, on the other hand, is a measure of how consistent and stable the results from a test are (Kombo, 2006). To ensure the validity of the research design, the research instrument was presented to the supervisor for guidance, recommendations, and advice. Also, necessary amendments were made based on the input of the supervisor.

3.6 Ethical Considerations

In this study, the researcher adopted a high level of professional ethics in conducting the research. To ensure that this project complies with the ethical requirement, the researcher made sure to get approval from the ethical committee. Forgery of documents, fake data, plagiarism, and other unethical practices was avoided, and intellectual property and copyright upheld.

Chapter 4: System Analysis and Design

4.1 Introduction

The System analysis and design in this research report on the planning process for the development of the prototype through understanding and specifying in detail what the prototype system should do and how the different parts of the system should be implemented and work all together. The system analysis and design refer to all activities, which lead to the transformation of requirement specification into implementation. Requirement specifications specify all non-functional and functional expectations from the system. These requirement specifications come in such a way that the documents are readable and understandable by a human being.

4.2 System Analysis

Requirements specification can be defined as a complete and detailed description of the behavior of the intended system prototype. There are several ways of categorizing system requirements such as non-functional and functional. This section covers the non-functional and functional requirements of the system. The system analysis is done based on the security aspect of the prototype. In the proposed work, the researcher implemented a smart farming system with a security mechanism using Hyperledger fabric and OpenVPN. To demonstrate the proof of concept of work, the researcher first designed a smart farming network with IoT devices with an integration of the security solution.

Therefore, in the first phase, different sensors such as temperature, humidity, moisture, and PIR Sensors are used to collect data from the field and send data to the raspberry pi via an Arduino microcontroller. The microcontroller which is in charge of analyzing the data and making a decision could activate actuators based upon their respective sensor values. Therefore, the sensors and controllers (actuators) should be connected to the Arduino microcontroller and the Arduino microcontroller should be connected to the Raspberry Pi. A wireless mesh network is implemented comprising three raspberry pi and a computer serving as a server for remote management. The network layer is secured using OpenVPN for a more secure data transmission through an encrypted tunnel. On the other hand, the application layer is secured using TLS encryption provided by the private distributed ledger Hyperledger fabric.

4.2.1 Functional Requirements

The proposed prototype should be able to:

- Sense temperature, humidity, water level, motion and apply automatically the appropriate control response in case of any problem according to the design.

- Transmit data sensed by sensors to the Raspberry Pi

- Transfer encrypted and encapsulated data to the ledger

- Resilient enough to provide access control, confidentiality, integrity, and availability of data.

- Visualize the content of the ledger in a web application.

4.2.2 Non-Functional Requirements

The main non-functional requirements of the proposed system are:

- Usability requirements

- Scalability and flexibility.

-Manageability and maintainability.

4.3 System Components Requirements

The hardware devices used in the implementation are listed in table 4.1 with their functionality.

Table 4.1: List of Hardware Devices

Hardware	**Functionality**
Computer	It should act as a server for remote management
Raspberry Pi 3	It should work as an operating system and CPU which helps for decision making
Arduino NodeMCU microcontroller	It should act as a microcontroller for the sensors and actuators
Micro SD card	should serve as Hard drive for the raspberry pi
Soil moisture water sensor	This should sense the water content in the soil and send it to the raspberry pi.

DHT 22 Sensor	This should sense the temperature and humidity of the environment.
PIR Sensor	It should sense the motion based upon the heat of the moving object.
Module Wi-Fi	Should be used for network connections
DEL	This device should enlighten a field when the motion of an object is detected
Buzzer	This device should make a sound when the gas is detected in the field area.
Micro solenoid valve	This should control the watering system
Fan	This actuator should get activated when the humidity value crosses the threshold value.

As far as security is concerned, in the second phase, the researcher used software such as OpenVPN and Hyperledger Fabric to secure the IoT system. The proper configuration is needed to ensure that all the software applications are working as intended. The software applications used in the implementation of security are listed in Table 4.2 with their role.

Table 4.2: List of Software Applications

Software	Role
Blockchain: Hyperledger Fabric	It should provide a distributed ledger where data can be stored with the guaranty of data integrity and security
Blockchain: Hyperledger Explorer	Web application tool to view, deploy, invoke or query blocks, transactions and associated data, network information, chain codes stored in the ledger
Hostapd	Provides WPA2 Access Point
Batman-Adv	It provides the wireless mesh infrastructure and routing protocols

Decentralized VPN: OpenVPN	It ensures security and privacy via secure tunneling and good encryption
Raspbian Buster	Raspberry Pi Operating System

4.4 System Architecture

To achieve the research objectives, the researcher intends to create an IoT system for smart farming based on a local and mesh network design. The system design is done, using UML diagrammic representation such as use cases, data flow, and interaction diagrams to give a good representation of how the system works. The system architecture is shown as follow:

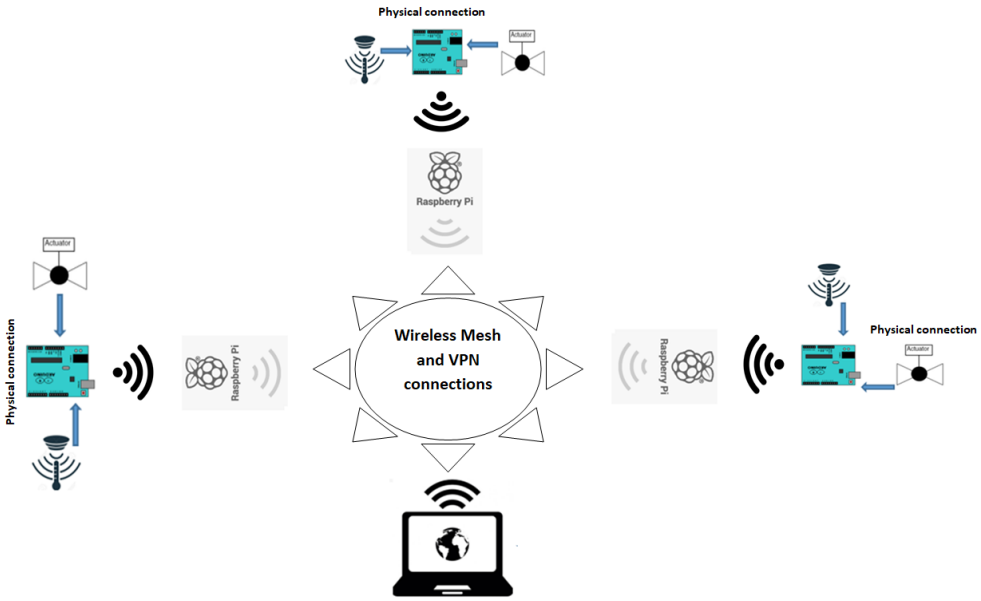


Figure 4.1: System Block Diagram

4.5 Data Transmission:

The data transmission is based on a wireless network sensor throughout the mesh network and the IPv4 protocol is used to uniquely identify each node in the network. Furthermore, the data is stored in the Hyperledger Fabric after authentication and validation.

4.6 Diagrammic Representation of the system

The diagrammic representation of the system is performed using the Unified Modelling Language (UML). This section is presented common diagrammic representation such as the use case diagram, sequence diagram, and the flow chart diagram.

4.6.1 Use Case

In this section, the use case diagram and the system contract are presented to portray the basic functionalities of a smart farming prototype.

a) Use case Diagram

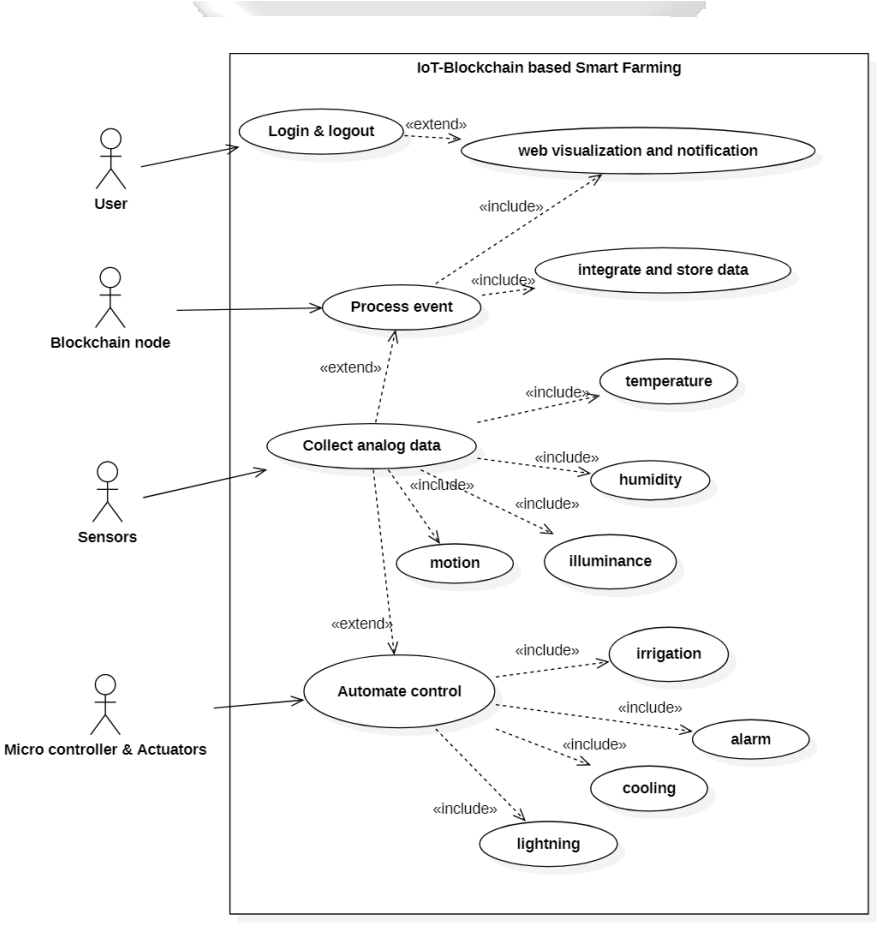


Figure 4.2: Use Case Diagram

b) Use Case Descriptions

Use Case Name	Collect Analog data
Description	Multiple sensors connected to the prototype collect data from the field
Primary Actor	Sensors Nodes
Trigger	Each 30 seconds
Pre-Condition	The microcontrollers are powered and operational
Post-Condition	Accurate field data sensed by sensors

Used Case Name	Automate field control
Description	Multiple actuators connected to the microcontroller control the environment based on a threshold
Primary Actor	Microcontroller/Actuator
Trigger	Physical characteristics value of the environment no within the threshold
Pre-Condition	The microcontrollers are powered and operational Actuators operational
Post-Condition	Environment physical characteristics controlled and maintained within the threshold

Used Case Name	Process Event Service
Description	The Hyperledger Fabric service validates, encrypts, encapsulates and stores data
Primary Actor	Blockchain nodes
Trigger	Validation and consensus from all nodes
Pre-Condition	Proposition for available data to be stored
Post-Condition	Data stored into the ledger

Used Case Name	Visualize data and Notify in Web App
Description	The User can visualize the data and receive Notifications
Primary Actor	User
Trigger	-
Pre-Condition	Available data and notifications to be visualized
Post-Condition	Data is visualized and notifications displayed

4.6.2 System Sequence Diagram

The system sequence diagram in figure 4.3 illustrates the interactions amongst the different components of the system. It depicts how the prototype collects data via the sensors, how the

environment is controlled via the microcontroller and actuator; how the data is sent and stored in the Blockchain, and how the data can be visualized by the user.

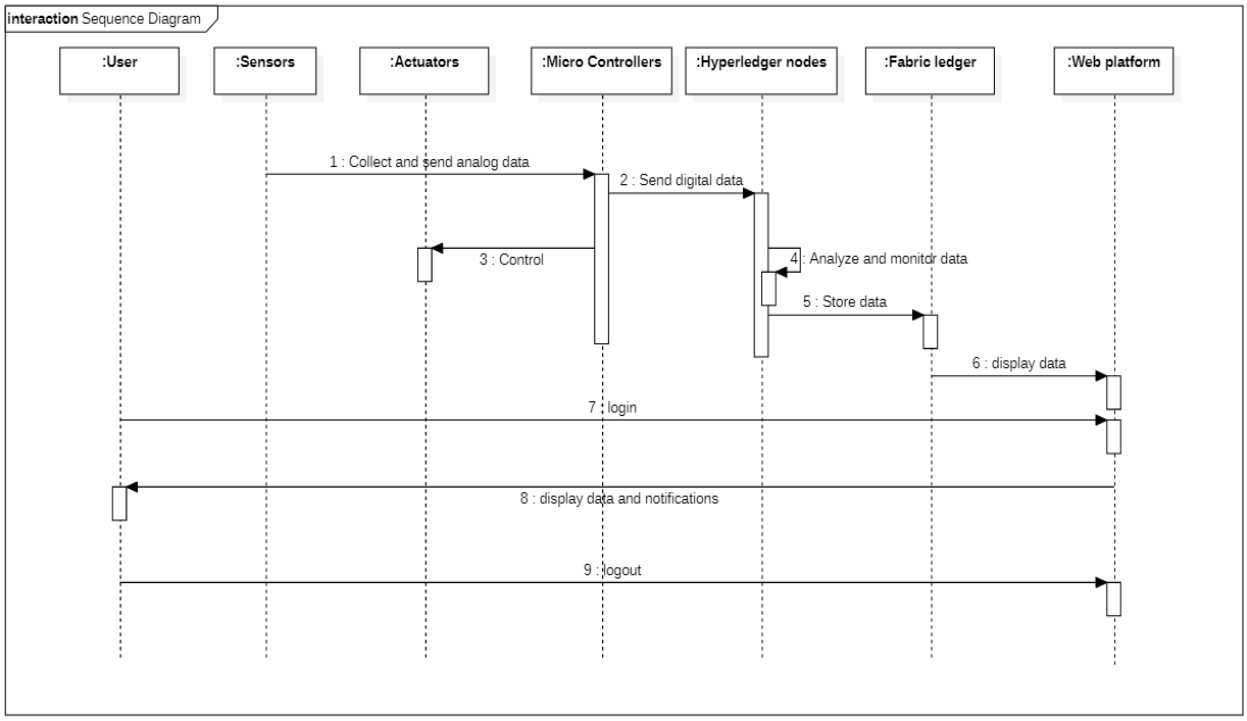


Figure 4.3: Sequence Diagram

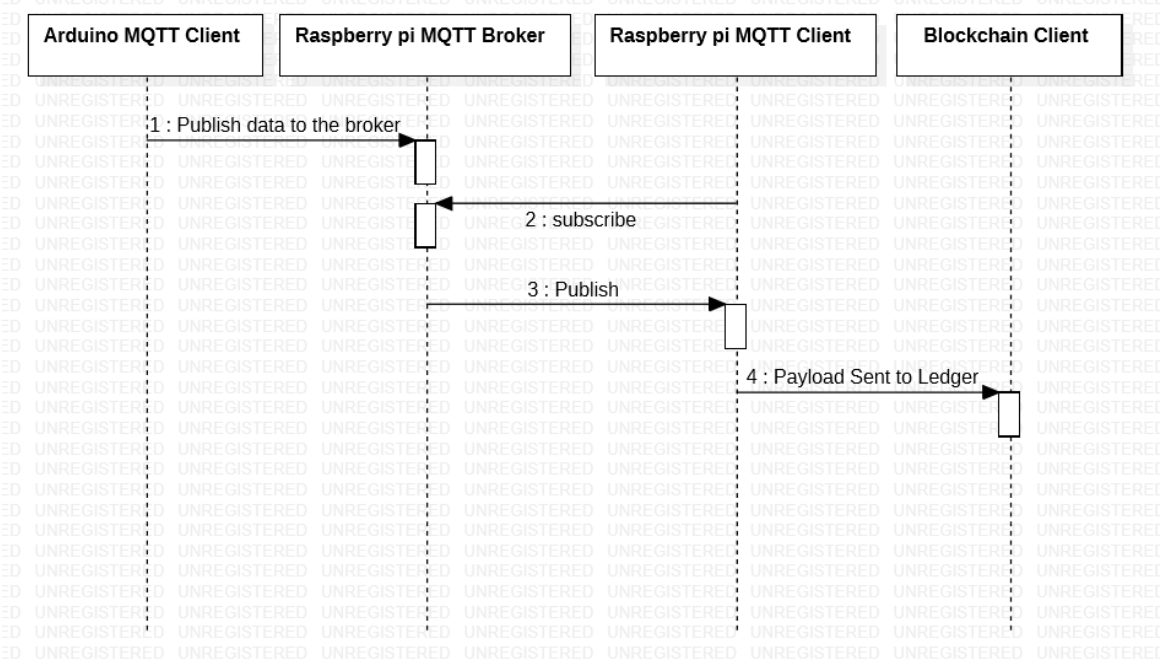


Figure 4.4: Communication Sequence from Sensor to Blockchain

Chapter 5: System Implementation

5.1 IoT data collection and environment control

We are using sensors DHTT22 that collects temperature and humidity of the environment, the PIR which is a motion sensor, the soil moisture sensor that collects the moisture of the soil which is used for automating the irrigation. Some actuators as a buzzer, LED, fan are used for environmental control. Then we use an Arduino sketch to run the IoT environment to collect data. Above is shown the IoT setup:

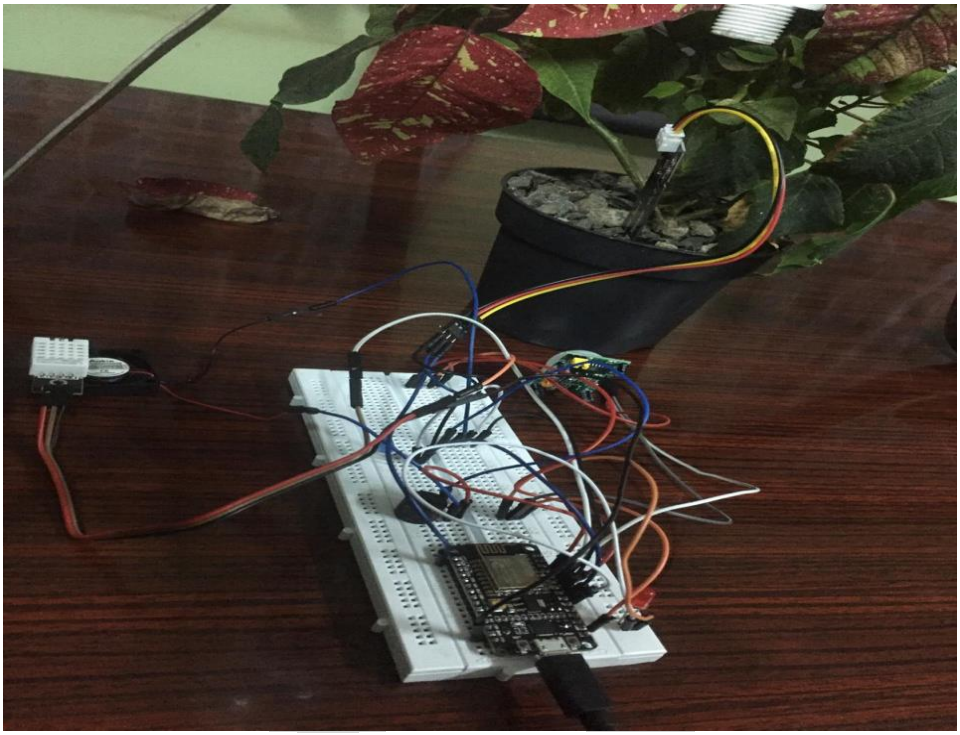


Figure 5.1: NodeMCU-Sensors-Actuators setup

This setup is connected to the NodeMCU that is used to send data via wifi to the raspberry pi which acts as an Access point. This access point is configured using Hostapd with WPA2 configuration and a strong password. Figure 5.2 shows the Hostapd configuration:

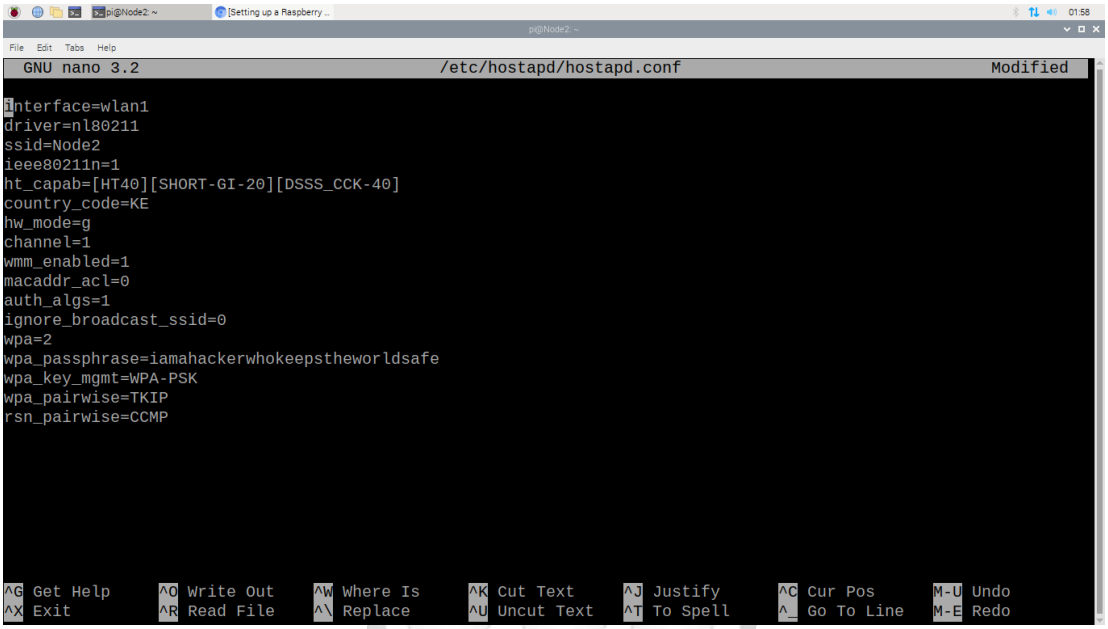


Figure 5.2: Snippet of AP Configuration

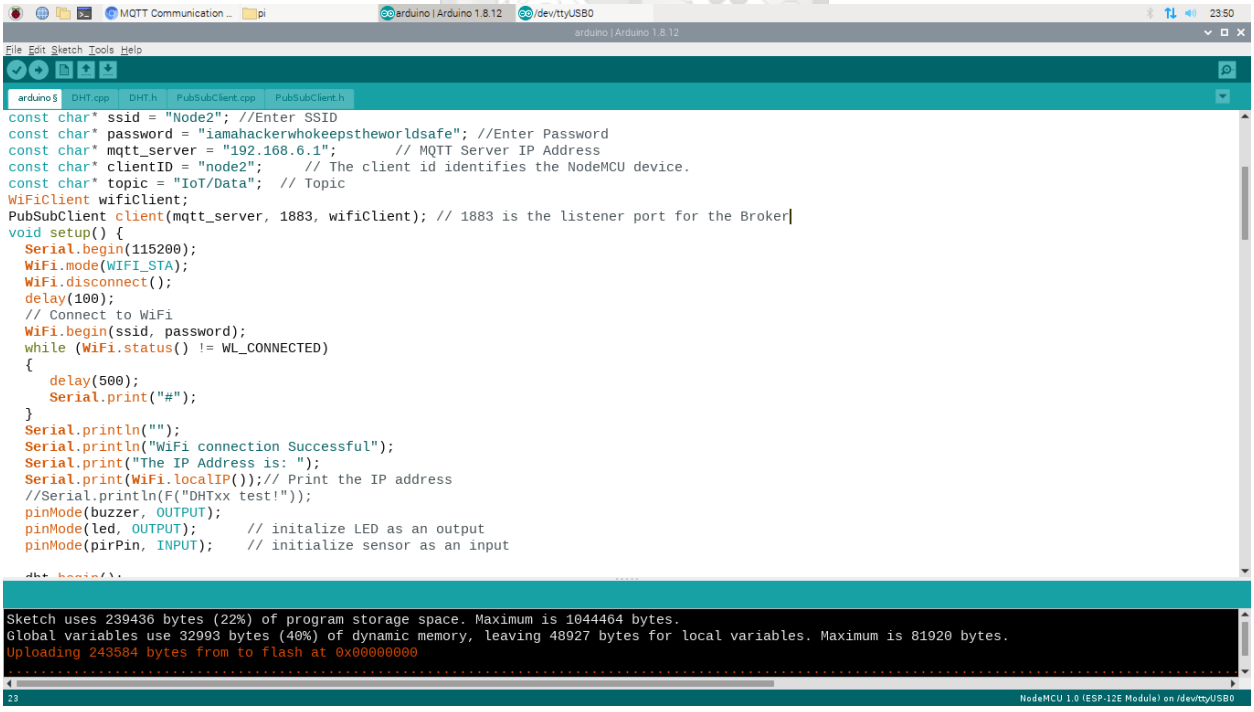


Figure 5.3: Sketch of Arduino Program to connect to AP and MQTT Server

We use MQTT Protocol to send the IoT data from the Arduino-nodeMCU to the raspberry. The data is pushed via wifi to an MQTT broker running on the raspberry pi node to a topic named "IoT/Data". An MQTT broker is installed on each node that receives data from the nodeMCU

and then forwards it to any clients that subscribe to the specific topic "IoT/Data". The snippets of the publishing to the topic is shown below:

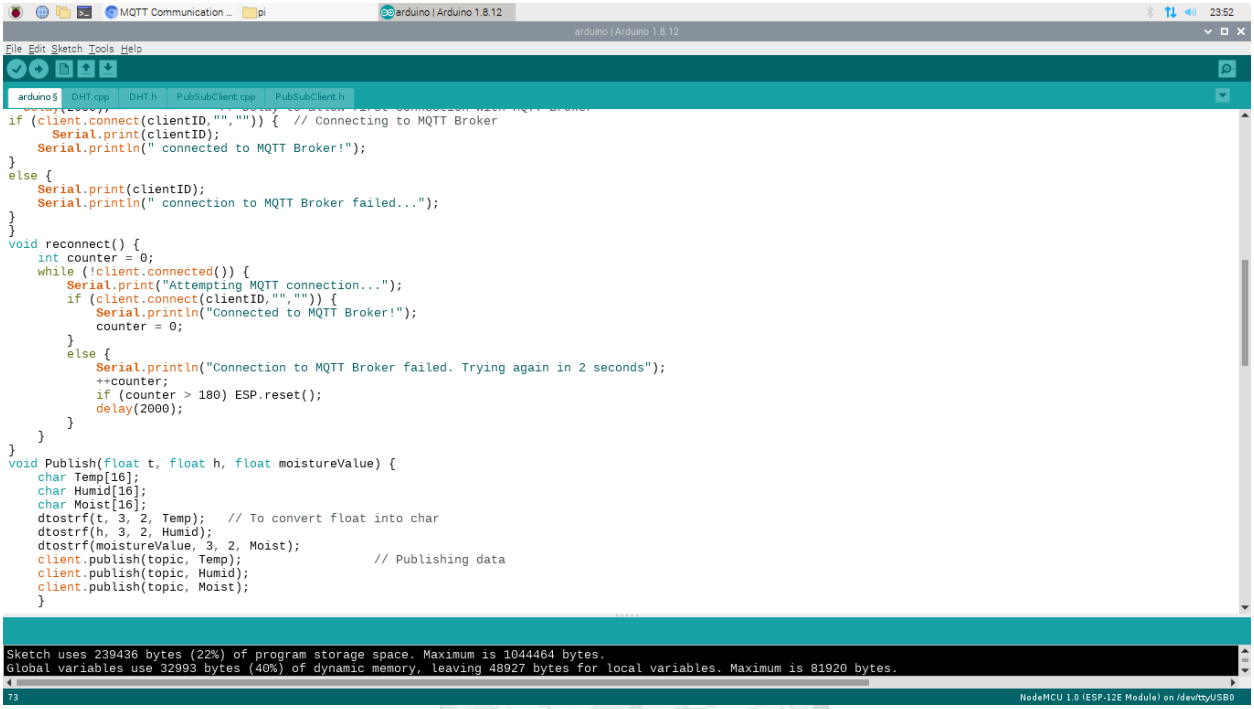


Figure 5.4: Snippet of the Program Publishing to the Topic

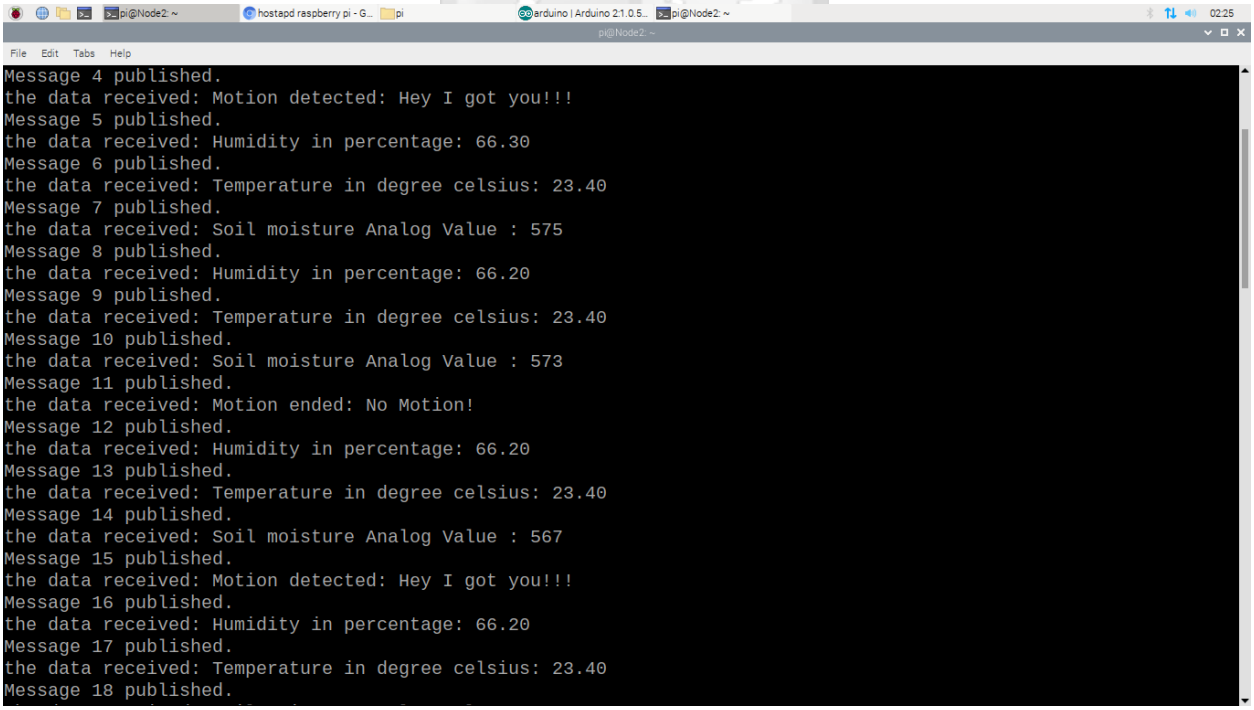


Figure 5.5: Screenshot of Data Being Published

An MQTT client running on the Hyperledger fabric node subscribes to the topic and receives this data. On the same Node, a node.js application receives this data and performs a transaction proposal from the received payload. The transaction is accepted only if it has been signed by peers in the network. Once the transaction is signed and submitted, it is validated for the signatures. If the transaction is valid then the data is stored to the ledger. the screenshot of the data received with the subscription is shown below:

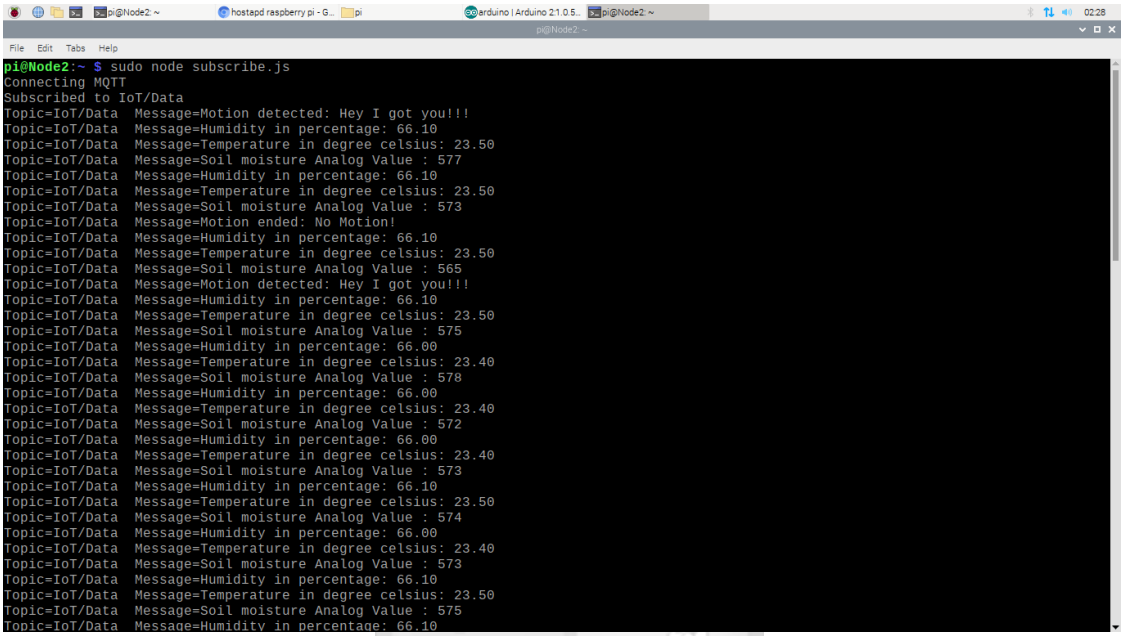


Figure 5.6: Screenshot of Data Received by Subscriber Client

5.2 Mesh Network Design

All the nodes of our prototype participate in a mesh network through the batman-adv protocol. We used an external Wi-Fi module dongle on the raspberry pi's to implement the mesh network. Batman-adv is a proactive routing protocol for WMNs. It uses a distance-vector approach and a routing metric that incorporates the reliability of the radio links for better performance. It is proactive in that each node maintains a routing table containing potential next hops to all other nodes forming the WMN. Batman-adv operates at layer 2 of the OSI model.

We installed batman-adv on each node. It provides a mesh and distributed architecture among the nodes. Figure 5.7 shows the neighboring nodes in the mesh network and shows that each node can communicate with each other directly.

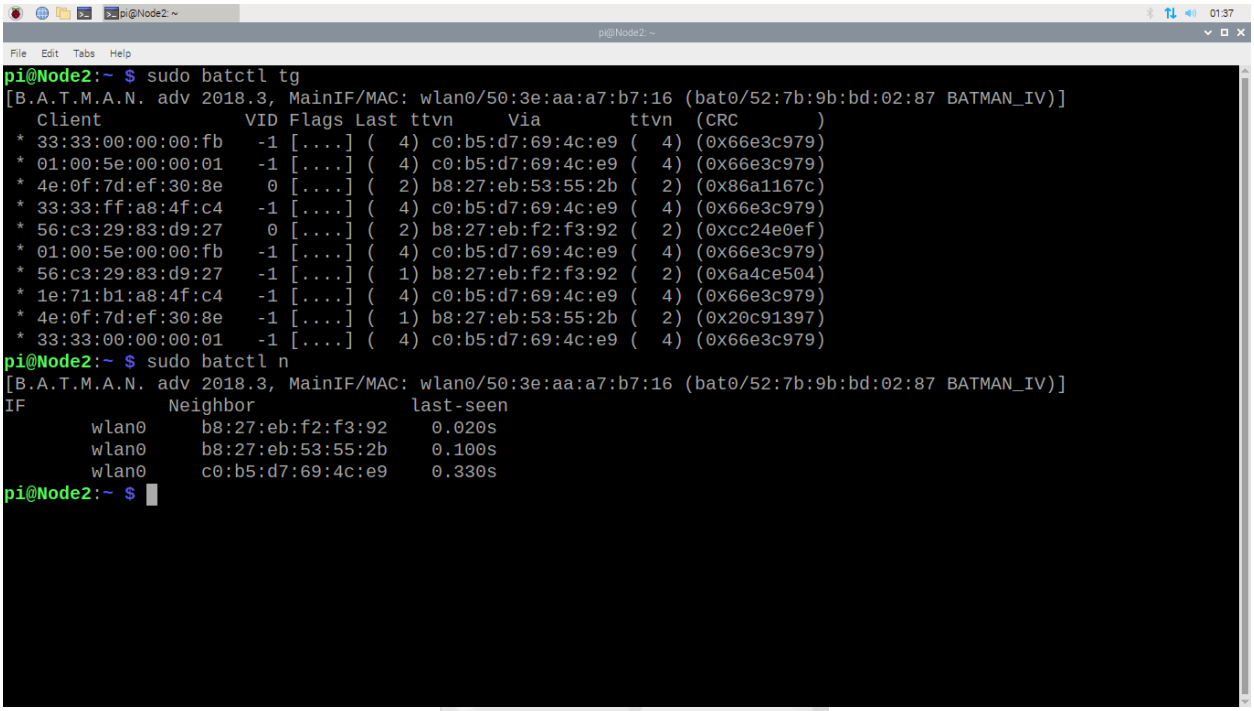


Figure 5.7: screenshots of the Neighboring nodes in the Mesh Network

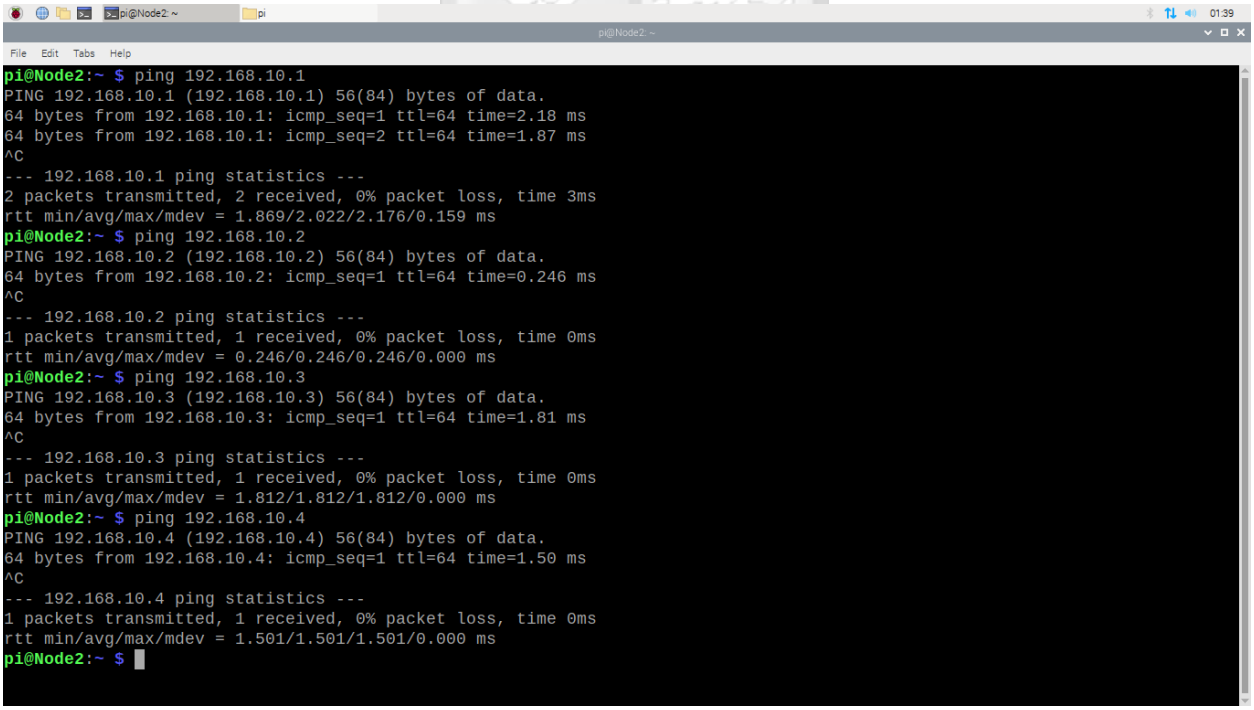


Figure 5.8: Node2 Pinging Other Nodes

As a security measure, each node is required to have a specific Cell number to participate in the mesh network. A node with a different cell number should not be able to join the network. below Figure 5.9 shows a screenshot of the interface.

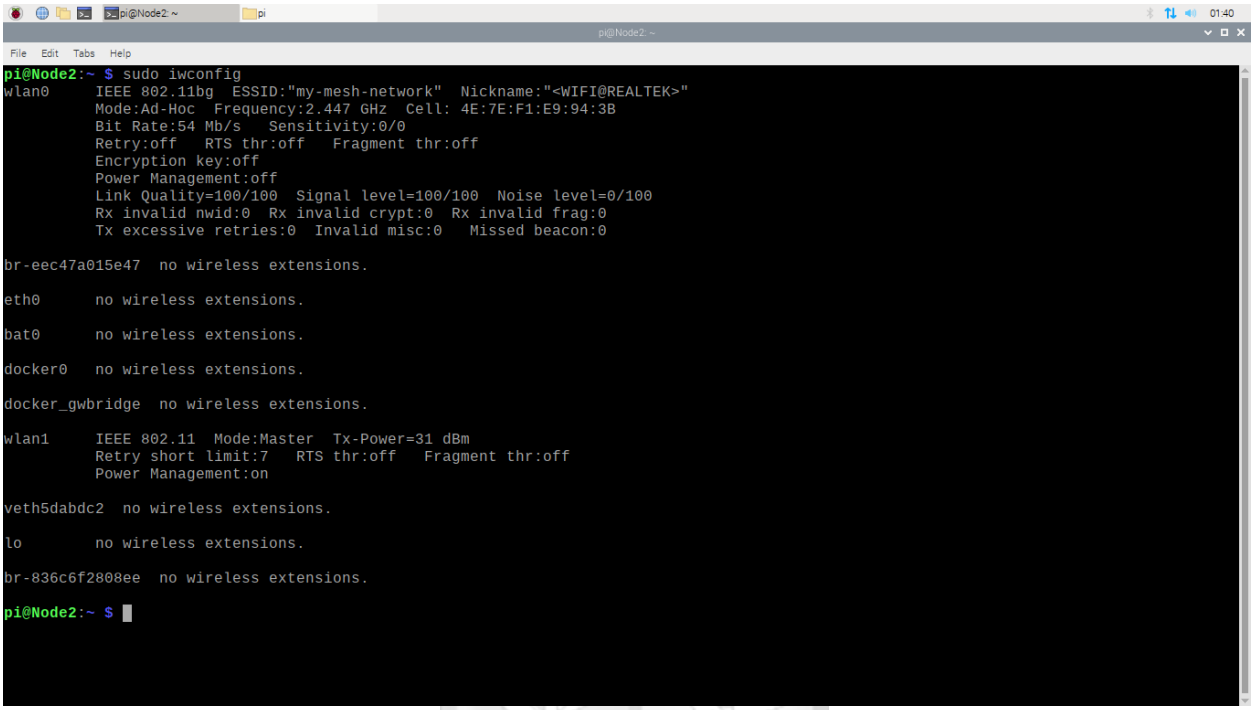


Figure 5.9: Screenshot of Mesh Network Interface

5.3 Configuration for Docker Containers

The network is implemented on Multihost. This is made possible with the Docker swarm and Overlay network implemented on top of the mesh network. The Docker swarm and overlay network are initiated on the Ubuntu laptop and acts as the leader or master and each raspberry joins the network as workers or managers. In our case, the raspberry joined as a worker. The overlay network enables communication amongst Docker containers across multi-machine. Figure 5.10 shows the Docker running the containers and the fabric network on all the nodes:

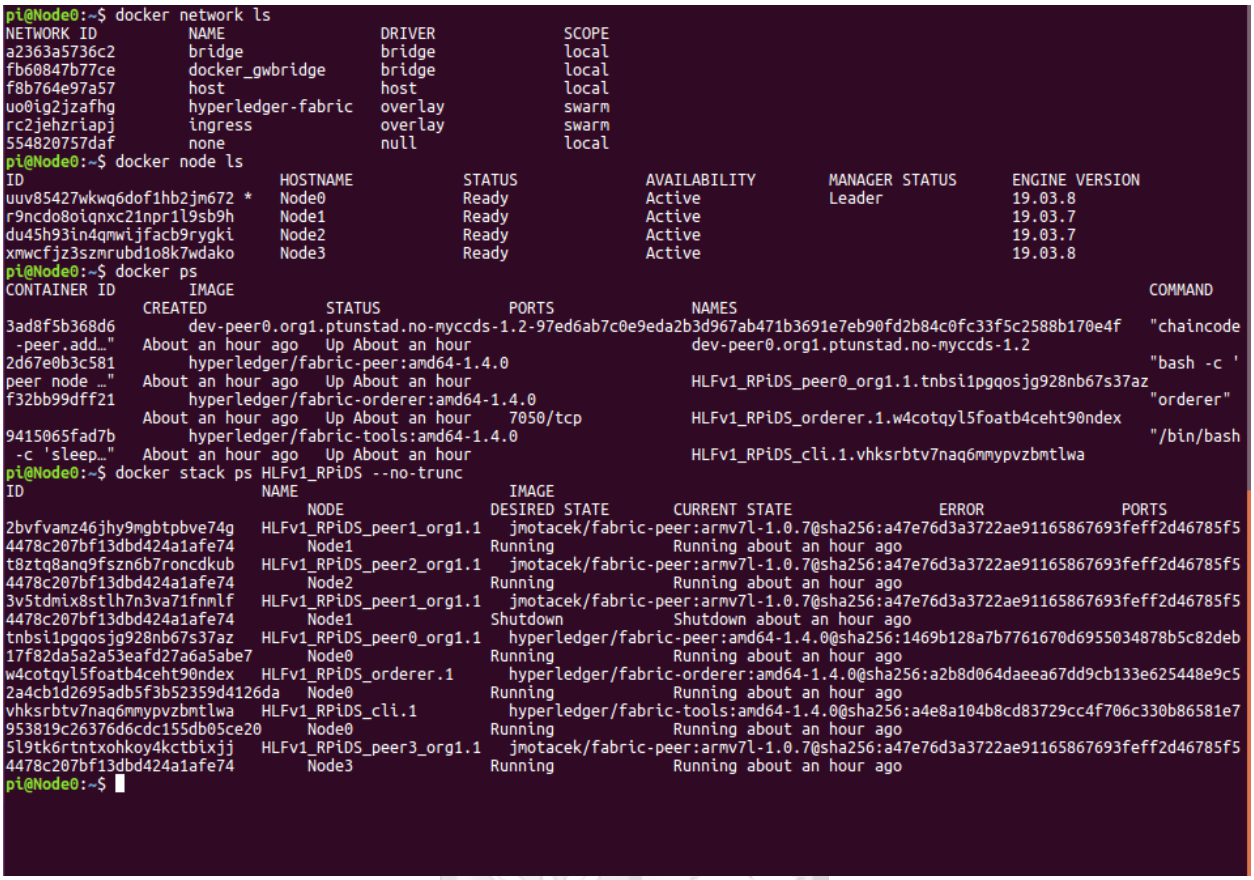


Figure 5.10: Docker Network and Containers

The Docker-compose-cli.yaml contains all the critical information needed to bring up the blockchain network. It describes the structure of the network and specifies the certs and signing key to be used. Figure 5.11 shows a snippet of the Docker-compose-cli.yaml:

```
 GNU nano 2.5.3                          File: docker-compose-cli.yaml


cli:
   image: hyperledger/fabric-tools:amd64-1.4.0
   tty: true
   environment:
     - GOPATH=/opt/gopath
     - CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
     - FABRIC_LOGGING_SPEC=DEBUG
     - CORE_PEER_ID=cli
     - CORE_PEER_ADDRESS=peer0.org1.ptunstad.no:7051
     - CORE_PEER_LOCALMSPID=Org1MSP
     - CORE_PEER_TLS_ENABLED=true
     - CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.ptunstad.no/peers/pee$
     - CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.ptunstad.no/peers/peer$
     - CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.ptunstad.no/peers$
     - CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.ptunstad.no/users/Adm$
     - CORE_VM_DOCKER_HOSTCONFIG_MEMORY=536870912
     - CORE_CHAINCODE_BUILDER=hyperledger/fabric-ccenv:amd64-1.4.0
     - CORE_CHAINCODE_GOLANG=hyperledger/fabric-baseos:amd64-0.4.14
     - CORE_CHAINCODE_CAR=hyperledger/fabric-baseos:amd64-0.4.14
     - CORE_CHAINCODE_JAVA=hyperledger/fabric-javaenv:amd64-1.4.0
   working_dir: /opt/gopath/src/github.com/hyperledger/fabric/peer
   command: /bin/bash -c 'sleep 30; ./scripts/script_ds.sh; while true; do sleep 20170504; done'
   volumes:
     - /var/run/:/host/var/run/
     - /home/pi/hlf_multihost/hyperledger-pi-composer/chaincode:/opt/gopath/src/github.com/hyperledger/fabric/examples/chaincode/go
     - /home/pi/hlf_multihost/hyperledger-pi-composer/crypto-config:/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
     - /home/pi/hlf_multihost/hyperledger-pi-composer/scripts:/opt/gopath/src/github.com/hyperledger/fabric/peer/scripts/
     - /home/pi/hlf_multihost/hyperledger-pi-composer/channel-artifacts:/opt/gopath/src/github.com/hyperledger/fabric/peer/channel-ar$
   depends_on:
     - orderer
     - peer0_org1
     - peer1_org1
     - peer2_org1
     - peer3_org1
   deploy:
     placement:
       constraints:
         - node.hostname == Node0
   networks:
     hyperledger-fabric:
       aliases:
         - cli.ptunstad.no

networks:
  hyperledger-fabric:
    external: true
```

Figure 5.11: Docker-Compose-Cli.yaml

5.4 OpenVPN configuration and implementation

The OpenVPN service is run in a Docker container. The client to client configuration is enabled to fit peer to peer communication offering a decentralized design amongst the Hyperledger-fabric node.

```
tls-server
local 10.0.1.17
#ifconfig  192.168.10.1 192.168.10.2
port 1194
proto tcp4-server
dev tun
tun-mtu 1500
tun-mtu-extra 32
mssfix 1450
ca /etc/openvpn/easy-rsa/pki/ca.crt
cert /etc/openvpn/easy-rsa/pki/issued/vpn-server.crt
key /etc/openvpn/easy-rsa/pki/private/vpn-server.key
dh /etc/openvpn/easy-rsa/pki/dh.pem
tls-auth /etc/openvpn/ta.key 0
server 192.168.10.1 255.255.255.0
duplicate-cn
client-to-client
push "dhcp-options DNS 8.8.8.8"
user nobody
group nogroup
keepalive 10 120
comp-lzo
persist-key
persist-tun
status /var/log/openvpn.log
verb 3
```

Figure 5.12: OpenVPN Configuration

5.5 Setting up the Hyperledger-Fabric Network

Hyperledger-Fabric and Composer are installed on each node. This setup provides the blockchain platform with the distributed ledger. We used the Hyperledger repository provided by Tunstad on github.com with the pre-built HLFV1 images for the ARM architecture by Joe Motacek.

5.5.1 Setting up the Network Artifacts and Configuration of the Network

Our Hyperledger network artifacts are composed of one organization with four peers. Each raspberry pi hosts a peer and the Ubuntu laptop hosts a peer, the orderer, the cli, and the endorser. The network artifact also includes the x509 certificates and signing key for each entity and it is provided by the cryptogen tool. Besides, the crypto-config.yaml file describes the structure of the network and enables the generation of key and certs for each entity:

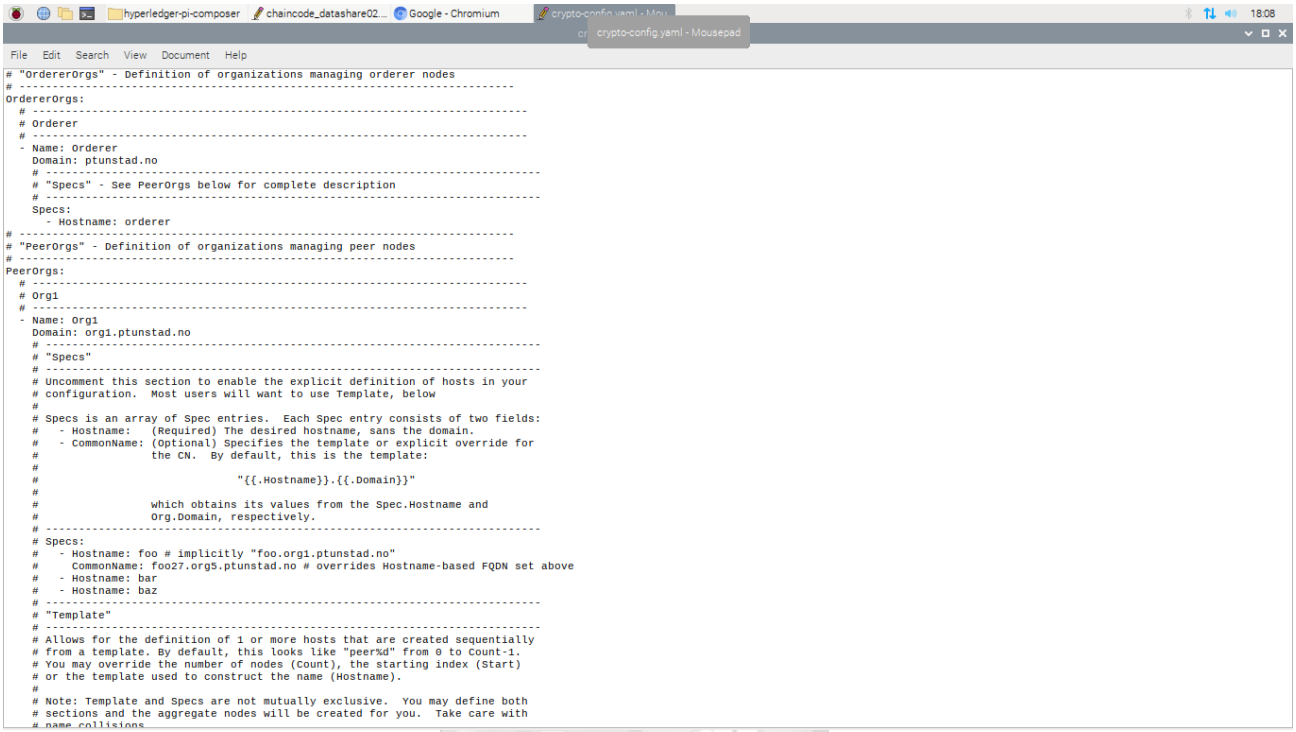


Figure 5.13: Crypto-Config.yaml

The operations in the h\Hyperledger-fabric network are performed in the form of transactions therefore there is a binary file configtxgen provided which is used to generate the configurations transactions. Moreover, the genesis block, the channels for transactions, and the anchor peer update are generated by the configtx file. It describes the policies that define the right of the organization, but it also describes all the configurations, the orderer, peers, and organizations participating in the network:

Figure 5.14: Configtx.yaml

5.5.2 Chaincode Design and Execution Environment

Hyperledger platforms support go, node.js, and java as smart contract languages. The Chaincode is a smart contract written in any of the supported languages which implement the prescribed interface. In another word, it is a program that runs on top of the blockchain to implement the business logic of how applications interact with the ledger. When a transaction is proposed, it triggers chaincode that decides what state change should be applied to the ledger. We are using go language for writing the chaincode.

The chaincode is executed in a Docker container which is separate from the endorsing peer. Mostly, the chaincode encapsulated the business logic that is agreed by the various business members in the network. The ledger state that created by one chaincode is not directly accessible to another chaincode within the same network. However, a chaincode can invoke another chaincode which is in the same network to make its state accessible if it has permission to do so. Figure 5.15 shows a screenshot of the chaincode function.

```
func (r *SmartContract) creatData(APIstub shim.ChaincodeStubInterface) td.Response {

        // Get the args from the transaction proposal
        args := stub.GetStringArgs()
        if len(args) != 3 {
                return shim.Error("Incorrect arguments. Expecting a key and a value")
        }

        // Set up any variables or IoTdata here by calling stub.PutState()

        // We store the data, key and the value on the ledger
                var IoT = IoTData{Temp:args[1], Humid:args[2], Moist:args[3]}
                data, _:jsom.Marshal(IoT)

        err := APIstub.PutState(args[0], data)
        if err != nil {
                return shim.Error(fmt.Sprintf("Failed to store data: %s", args[0]))
        }
        return shim.Success(nil)
}
```

Figure 5.15: Chaincode Function

5.5.3 Installing and Instantiating the Chaincode

The transaction install does the following two things. First, it packages the smart contract or chain code into a standard format called CDS or ChaincodeDeploymentSpec. Second, it also proceeds with the installation of the packaged chaincode on the peer node. The Chaincode is to be installed on each endorsing peer node of the channel. It should not be installed on the peer nodes which are not endorsing peer nodes as it may affect the confidentiality of the chaincode logic. This type of transaction is used for invoking the LSCC or Lifecycle System Chaincode which binds the chaincode to a channel. The creator of an instantiate transaction must comply with the chaincode instantiation policy included in SignedCDS and must also be a writer on the channel configured as part of the channel creation process. This is necessary for channel security to prevent unauthorized entities from deploying chaincode or making members on an unbound channel to execute chaincode. Once a chaincode has been properly endorsed, it can be installed and instantiated on Peers in the network. During the instantiation process, the Peer uses Docker to run a container with the chaincode inside. Figure 5.16 shows the function to store data.

```
//Functionality for storing a Iot data as a key,value entry in the blockchain.
function storeData(arglist, respo, body){
    if(resp){
        var key = arglist[0]
        var args = [key, body]
        ccSet(args, SetCallback, null, respo)

    }else{
        if(arglist.length < 2){
            console.log("Need two arguments to store Iot data. Usage: key, data")
        }
        var key = arglist[0]
        var value = base64fromFile(arglist[1])
        var args = [key, value]
        ccSet(args)
    }
}
```

Figure 5.16: Function to Store Data in the Ledger

5.6 Storing Data into the Blockchain Ledger

The storage of the IoT data in the ledger is performed through a client nodejs application that we built. This application interacts with the blockchain network and generates a transaction proposal that contains the smart contract function to be called in this case createdata, it also contains the chaincode id, the arguments for the function, the channel, and transaction Id. Transaction proposal request for inserting data into the ledger by the client is shown below:

```
// get a transaction id object based on the current user assigned to fabric client
tx_id = fabric_client.newTransactionID();
console.log("Assigning transaction_id: ", tx_id._transaction_id);

// must send the proposal to endorsing peers
var request = {
    //targets: let default to the peer assigned to the client
    chaincodeId: 'myccds',
    fcn: 'createdata',
    args: [ Temp, Humid, Moist],
    chainId: 'mychannel',
    txId: tx_id
};
```

Figure 5.17: Screenshot of Client Nodejs Application

```
        // send the transaction proposal to the peers
        return channel.sendTransactionProposal(request);
.then((results) => {
        var proposalResponses = results[0];
        var proposal = results[1];
        let isProposalGood = false;
        if (proposalResponses && proposalResponses[0].response &&
                proposalResponses[0].response.status === 200) {
                        isProposalGood = true;
                        console.log('Transaction proposal was good');
                } else {
                        console.error('Transaction proposal was bad');
                }
        if (isProposalGood) {
                console.log(util.format(
                'Successfully sent Proposal and received ProposalResponse: Status - %s, message - "%s"',
                        proposalResponses[0].response.status, proposalResponses[0].response.message));

                // build up the request for the orderer to have the transaction committed
                var request = {
                        proposalResponses: proposalResponses,
                        proposal: proposal
                };
```

Figure 5.18: Function of Sending the Transaction proposal

5.7 Hyperledger Explorer Web Application

Hyperledger Explorer is a user-friendly Web application tool used to view, invoke, deploy, or query blocks, transactions and associated data, network information (name, status, list of nodes), chain codes and transaction families, as well as any other relevant information stored in the ledger. In our case, Hyperledger Explorer was installed on the Ubuntu laptop which serves as the leader of the swarm network. This node is hosting the endorser, the orderer, and the anchor peer. The access of the web application is protected by a mechanism of authentication (username and password). Figure 5.19 shows the interface of the web application with the blockchain network information:

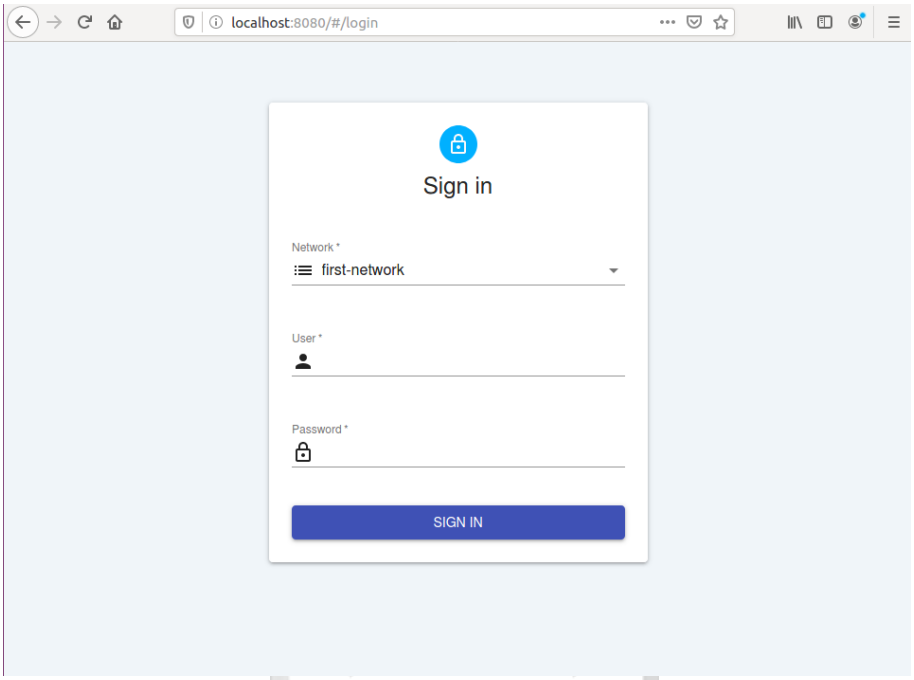


Figure 5.19: Screenshot of the Login Interface

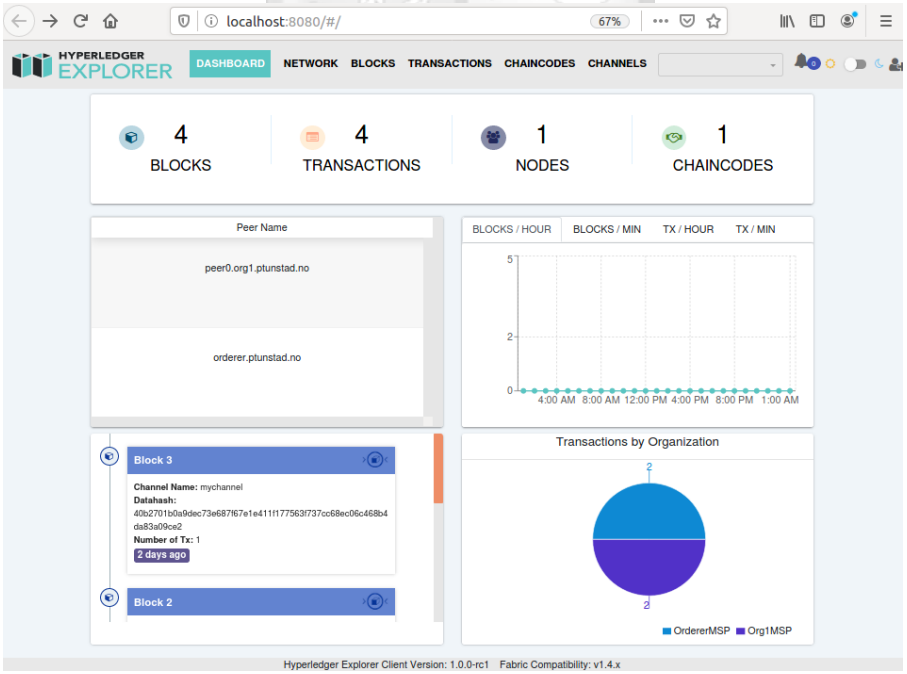


Figure 5.20: Interface of the Web Application

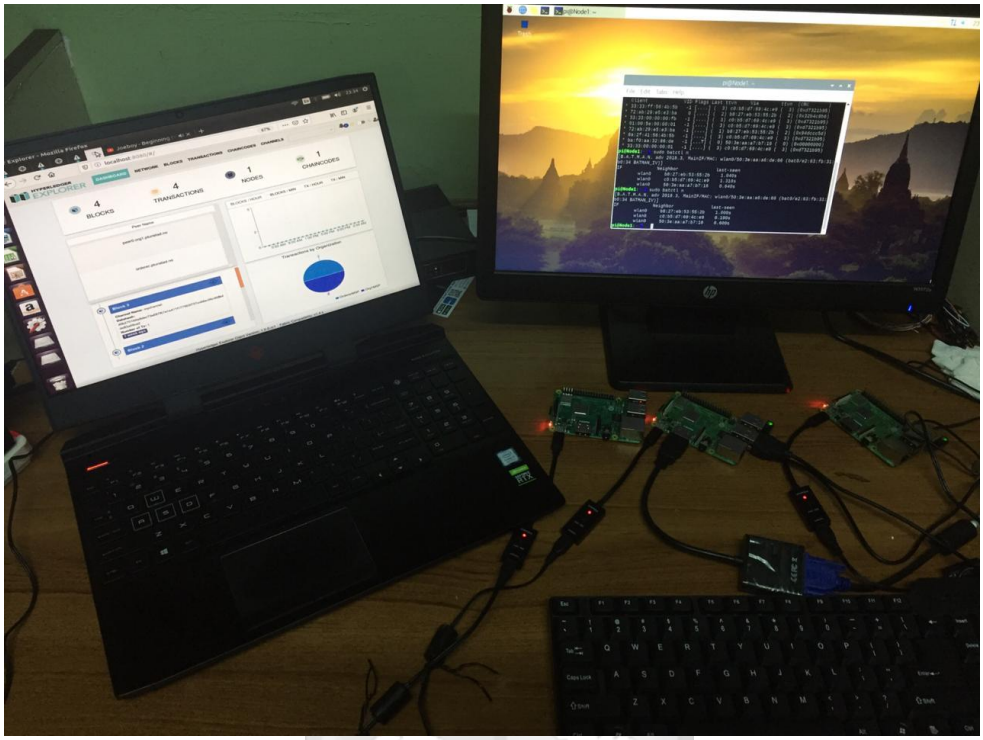


Figure 5.21: Hyperledger-Fabric Nodes Setup

5.8 Security Design

- The web application is secured with a username and password that restrict access

- The wireless Access point network provides by the raspberry in encrypted using WPA2-AES encryption with a strong password of 31 characters. That could withstand dictionary and brute force attacks.

- The mesh network is secured by using a specific CELL number that allows only authorized nodes to join the network.

- On top of the mesh network, the Docker swarm and overlay network provide the OpenVPN service amongst the Hyperledger-fabric node. Ensuring confidentiality and privacy.

- The Hyperledger-fabric network itself is configured to use TLS encryption with x509 certificates and signing key for authentication and privacy.

Chapter 6: System Testing

6.1 Functional Testing

The functional testing was performed to check that the system effectively collects IoT data and stores it in the Hyperledger fabric ledger and can be visualized via the Explorer web application.

Table 6.1: Functional Testing Results

No	Component	description	Expected behavior	Observed behavior	error	conclusion
1	Data collection	Test if data is being collected by sensors	The system should be able to read and collect field data	The system reads and collects field data	None	OK
2	Field control	Test if actuators can control the field environment	Actuators should be able to control temp/humidity, soil moisture, and motion detection	Actuators control temp/humidity, soil moisture, and motion detection	None	OK
2	Connection to Access Point (AP) and MQTT broker	Testing if the nodeMCU connect to the AP and MQTT broker	The nodeMCU should be able to connect to the AP and broker	The nodeMCU connects to the AP and MQTT broker	None	OK
3	Data transfer with the broker	Testing if data is being published and received from the broker	Data should be published to the broker and data should be received by the subscriber	Data is published to the broker and data is received by the subscriber	None	OK
4	Data storage in the ledger	Testing if data is being stored in the ledger	The system should be able to store data into the ledger	The system stores data into the ledger	None	OK
5	Network visualization	Testing if the network can be visualized in the web application	The web application should be able to display the network information	The web application displays the network information	None	OK

6.2 Security Testing

The security testing was focused on the confidentiality attacks, integrity attacks, and availability attacks. We used Kali Linux distribution to perform the testing. The choice of the type of attack in each category is based on the common attacks on the IoT system.

6.2.1 Confidentiality Testing

- Network Sniffing attack: the network can be sniffed and analyzed by using Wireshark or Tcpdump tools. We were not able to sniff the network since the mesh network is protected with a Cell number that allows only authorized node; Moreover, the fact that Hostapd access point is protected with a strong password, protect is against unauthorized access to the network, therefore, prevents the sniffing attack. The confidentiality of data in the network was preserved with the AES and TLS encryption of the network at layer 3 and 4 of the OSI model.
- Man-In-The-Middle Attack: it is an eavesdropping attack whereby the attacker secretly intercepts, relays, and alters the communication between two parties. The attack was not successful due to the OpenVPN encryption and authentication mechanisms.
- Password cracking attack: it consists of cracking the WPA2/PSK pre-shared key after capturing the 3 way-handshake. In our case, the attack was able to capture the 3 way-handshake but was not successful in cracking the password due to the length of the password which is 31 characters.

6.2.2 Integrity Testing

- Data replay attack: network-based security hacks that delay, replay the valid transmission of data between genuine users. This attack was not successful due to the implementation of the VPN authentication
- Data tampering attack: This is when an attacker can tamper with the data stored in one node of the network. This attack was not successful due to the characteristics of the blockchain network that ensure data integrity

6.2.3 Availability Testing

Denial of service attack aims at preventing a legitimate user from accessing the wireless network resources by disrupting wireless connections.

- De-authenticate flood / Disassociation attack: this attack disconnects the wireless client from the AP by sending continuous broadcast de-authenticate commands. This attack was not successful due to the configuration of the batman-adv mesh network that binds all the nodes based on a specific Cell number.

Table 6.2: Security Testing Results

Link	Number of Testing simulations	Attack performed on the system	Result of the attack
Wireless Mesh Network and OpenVPN links	5	Network Sniffing attack	Failed
	5	Man-In-The-Middle Attack	Failed
	5	Password cracking attack	Failed
	5	De-authenticate flood / Disassociation attack	Failed
	5	Data replay attack	Failed
	5	Data Tampering attack	Failed
Wireless NodeMCU-Pi Node links	5	Network Sniffing attack	Succeeded
	5	Man-In-The-Middle Attack	Succeeded
	5	Password cracking attack	Failed
	5	De-authenticate flood / Disassociation attack	Succeeded
	5	Data replay attack	Failed
	5	Data Tampering attack	Failed

Chapter 7: Conclusion and Future Work

7.1 Conclusion

Our IoT Blockchain implementation works successfully which allows us to store IoT data that is trustworthy and verified. Our network works well for a decent number of transactions but having just one orderer limits its performance in terms of the number of transactions it can handle. We conclude that the Hyperledger fabric platform can be used for managing IoT data whereby the in-built endorsements, encryption, certs, and signing keys ensures the security, validity, and integrity of the data. All the transactions in Hyperledger are TLS encrypted which ensures that data cannot be compromised during the internal network communications. Besides, the implementation of a VPN, strong authentication mechanisms the mesh network, and wireless network provide additional security to the proposed architecture.

7.2 Future Work

IoT blockchain network implementation is based on wireless technology and the security aspect is a concern. The security of the IoT blockchain network can be done in different ways from the design of the architecture of the network to the configuration of each component participating in the network. In our case, we focused on the distributed architecture using the batman-adv protocol to avoid a single point of failure and also guarantee consistent availability and scalability. Moreover, we implemented strong authentication and access control to our network through a strong password, certificates, and signing key.

However, this work can be extended by using babel mesh protocol which is a distance-vector routing protocol for IPv6 and IPv4 with fast convergence properties that could offer more scalability and security mechanism for a distributed architecture. Moreover, the implementation of the IPsec connection between the data collection node and the Hyperledger fabric node could also contribute to the security of the architecture. Future work can also focus on the use of WPA2 with AAA authentication such as RADIUS or KERBEROS to improve access to network resources.

References

Alysaa, H. (2018). *Blockchain Feared 51% Attack Now Becoming Regular*. Coindesk. June 8, 2018. Retrieved from https://www.coindesk.com/blockchains-feared-51-attack-now-becoming-regular-

Akhil, D. (2019). *Managing IoT Data on Hyperledger Blockchain*. UNLV Theses, Dissertations, Professional Papers, and Capstones.

Ashton, K. (2009). That 'internet of things' thing. *RFID Journal*, *22*(7), 97-114.

Amandeep, A., Arshia B., Paboni D., Debjit B., Somudit R., Spandan G., Sayan S., Souvik P., Sourav D., & Rana, T.K. (2017). Smart Farming Using IoT. *2017 8th IEEE Annual Information Technology, Electronics, and Mobile Communication Conference (IEMCON).*

Connect, D. (2017). *The Internet of Things*. European Commission.

Conoscenti, M., Vetro, A., & De Martin, J. C. (2016). Blockchain for Internet of Things: A systematic Literature Review. *2016 IEEE/ACS 13th International Conference Computer Systems and Applications (AICCSA)*. Agadir.

Dorri, A., kanhere, S. S., Jurdak, R., & Gauravaram, P. (2017). Blockchain for IoT Security and Privacy: The Case study of smart home. *2017 IEEE international conference on Pervasive computing and communications workshops, Kona*, HI.

Dorsemaine, B., Gaulier, J. P., Wary, J. P., Kheir, N., & Urien, P. (2015). Internet of Things: A Definition and Taxonomy. *In 2015 9th International Conference on Next Generation Mobile Applications, Services and Technologies*. IEEE, 72–77.

Dorsemaine, B., Gaulier, J. P., Wary, J. P., Kheir, N., & Urien, P. (2016). A new approach to investigate IoT threats based on a four-layer model. *2016 13th International Conference on New Technologies for Distributed Systems (NOTERE)*, Paris, 2016, 1-6.

Federal Bureau of Investigation, Cyber Division. (2016). *Smart Farming May Increase Cyber Targeting Against US Food and Agriculture Sector*. FBI Cyber Bulletin as retrieved 17th

April 2016 from https://publicintelligence.net/fbi-smart-farm-hacking/

Fremantle, P., Aziz, B., & Kirkham, T. (2017). *Enhancing IoT Security and Privacy with Distributed Ledgers*. A Position Paper.

Group, I. W. (2016). *Future-Proofing the Connected World: 13 Steps to Developing Secure IoT Products*. Cloud Security Alliance.

Group, M. W. (2015). *Security Guidance for Early Adopters of the Internet of Things (IoT)*. Cloud Security Alliance.

Homeland Security. (2016). *Strategic Principles for Securing the Internet of Things (IoT)*. U.S. Department of Homeland Security.

Insights, A. C. S. (2015). The CEO 's Guide to Securing the Internet of Things. AT&T.

Iqbal, A. M., & Riadi, I. (2019). Analysis of Security Virtual Private Network (VPN). *Using OpenVPN., 8,* 58-65.

Iqbal, A., Saleem, R., & Suryani, M. (2016). Internet of Things (IoT): On going Security Challenges and Risks. *International Journal of Computer Science and Information Security*, *14*, 671.

Krishnamachari, B., & Ramachandran, G. (2018). *Blockchain for the IoT: Opportunities and Challenges*. Retrieved from https://www.researchgate.net/publication/325033863_Blockchain_for_the_IoT_Opportunities_and_Challenges

Karen, R., Scott, E., & Lyman, C. (2015). *The Internet of Things: An overview Understanding the issues and challenges of a More Connected World*. The Internet Society (ISOC). Pp. 1-80. Retrieved from https://www.internetsociety.org/wp-content/uploads/2017/08/ISOC-IoT-Overview-20151221-en.pdf

Karuna J. K., & Indira R.B. (2018). Study on Virtual Private Network (VPN), VPN's Protocols And Security. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*. *3*(5).919-932.

Kennedy, C. (2017). *Internet of things: The Cyber Security risks and how to protect against them*. Accessed 23 March 2018, Retrieved from https://www.itpropotal.com/features/the-internet-of-things-the-cyber-security-risks-and-how-to-protect-them/

Kombo, D. K., & Tromp, A. D. L. (2006). *Proposal and thesis writing, an introduction*. Kenya: Paulines Publications Africa.

Kothari, C. R. (2008). *Research Methodology, Methods and Techniques* (2nd ed.). New Delhi: New Age International Publishers Limited.

Komninos, N., Philippou, E., & Pitsillides, A. (2014). Survey in smart grid and smart home security: Issues, challenges and countermeasures. *IEEE Communications Surveys and Tutorials, 16(4),* 1933–1954. https://doi.org/10.1109/COMST.2014.2320093

Kshetri, N. (2017). Blockchain's roles in strengthening Cybersecurity and Protecting Privacy. *Telecommunications Policy*, 41(10), 1027 –1038. https://doi.org/10.1016/j.telpol.2017.09.003

Lin, C., He, D., Huang, X., & Choo, K. K. R. (2018). Secure and privacy preserving smart contract-based solution for access control in IoT. *Newsletter, 2018*.

Lin, J., Shen, Z., Zhang, A., & Chai, Y. (2018). Blockchain and IoT based Food Traceability System. *International Journal of Information Technology, 24(1),* 2018.

Lin, Y., Petway, J. R., Anthony, J., Mukhtar, H., Liao, S., Chou Orcid, C., & Ho, Y. (2017). Blockchain: The Evolutionary Next Step for ICT E-Agriculture. *Environments 4, 3* (2017), 50.

Miller, L. (2016). *IoT Security For Dummies.* INSIDE Secure Edition. Accessed 10 August 2017. Retrieved from https://www.insidesecure.com/content/download/909/10066/file/IOT%2BSecurity%2Bfor%2BDummies.pdf-

Mugenda O., & Mugenda A. (2003). *Research Methods: Quantitative and Qualitative Approaches.* Nairobi: Acts Press.

Muthuswamy, S., & Ganapathi, P. (2016). IoT Security Challenges and Issues-An Overview. *World Scientific News 41* (2016) 1-315. Retrieved from https://bitcoin.org/bitcoin.pdf

Nallapaneni, M. K., & Preadeep, M. K. (2018). Blockchain Technology for security issues and Challenges in IoT. *Procedia Computer Science 132*, 1815-1823.

Nayyar, A., & Puri, V. (2016). *Smart farming: IoT based smart sensors agriculture stick for live temperature and moisture monitoring using Arduino, cloud computing & solar technology*. 673-680. https://doi.org/10.1201/9781315364094-121

O'Dwyer, K. J., & Malone, D. (2014). *Bitcoin mining and its energy footprint*. 2014. https://doi.org/10.1049/cp.2014.0699

OWASP Foundation. (2014*). OWASP Internet of things top ten projects*. Open Web Application Security Project, 2014. Retrieved from https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project#tab=Main-

Patil, A.S., Tama, B.A., Park, Y., Rhee, K.H. (2018). A Framework for Blockchain Based Secure Smart Green House Farming. In Advances in Computer Science and Ubiquitous Computing, Park J., Loia V., Yi G., Sung Y. (Eds.). *Lecture Notes in Electrical Engineering, 474*. Springer, Singapore, CUTE 2017, CSA 2017. https://doi.org/10.1007/978-981-10-7605-3_185

Praetorian. (2017). *The 8 Biggest IoT Security Mistakes and How to Avoid Them*. Accessed 10 August 2017. Retrieved from https://p16.praetronian.com/downloads/report/The_8_Biggest_IoT_Security_Mistakes_and_How_to_Avoid_Them.pdf-

Purbo, O. W. (2013). *Solution Mesh Network Building a Mutual Cooperation Wireless Network Without Access Points*. Yogyakarta: Andi.

Quarkslab. (2017). *OpenVPN 2.4.0 Security assessment*. Technical Report. Ref: 17-03-284 REP version 1.2

Restuccia, F., d'Oro, S., Kanhere, S., Melodia, T., & Das, S. (2018). Blockchain for the Internet of Things: Present and Future. *IEEE INTERNET OF THINGS JOURNAL, 1(1)*, January

2018.

Reyna, A., Martín, C., Chen, J., Soler, E., & Díaz, M. (2018). (2018). On blockchain and its integration with IoT. Challenges and opportunities. *Future Generation Computer Systems, 88*, 173–190. https://doi.org/10.1016/j.future.2018.05.046

Rizvi, S., Kurtz, A., Pfeffer, J., & Rizvi, M. (2018). *Securing the Internet of Things (IoT): A Security Taxonomy for IoT*. 163-168.

Selimi, M., Rao, A., Ali, A., Navarro, L., & Sathiaseelan, A. (2018). Towards Blockchain-enabled Wireless Mesh Networks. *CryBlock'18*, June 15, Munich, Germany.

Seyoung, H., Sangrae, C., & Soohyung, K. (2017). Managing IoT Devices Using Blockchain Platform. ICACT2017 February 19-22, 2017.

Shyamala, D., Suguna, R., Joshi, A., & Bagate, R. (2019). *Design of IoT Blockchain Based Smart Agriculture for Enlightening Safety and Security*. 10.1007/978-981-13-8300-7_2.

Sulkamo, V. (2018). *IoT from Cyber Security Perspective: Case study JYVSECTEC*. Master's Thesis.

Suo, H., Wan, J., Zou, C., & Liu, J. (2012). *Security in the internet of things: a review.* In Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference, 3, 648-651. IEEE. https://doi.org/10.1109/ICCSEE.2012.373

Tanweer, A. (2019). Blockchain and its Role in the Internet of Things (IoT*). International Journal of Scientific Research in Computer Science, Engineering and Information Technology*. 151-157.

The Blockchain / Distributed Ledger Working Group. (2018). *Using Blockchain Technology to Secure the Internet of Things*. Cloud Security Alliance.

Vasin, P. (2014). *Blackcoin's proof-of-stake protocol v2*. Retrieved from https://blackcoin-pos-protocol-v2-whitepaper.pdf

Vitalik, B. (2018). *Blockchain Resource Pricing*. Retrieved from

https://ethresear.ch/uploads /default/original-

Weber, R. H. (2010). Internet of Things- New security and privacy challenges. *Computer Law & Security Review*, *26*(1), 23-30. https://doi.org/10.1016/j.clsr.2009.11.008

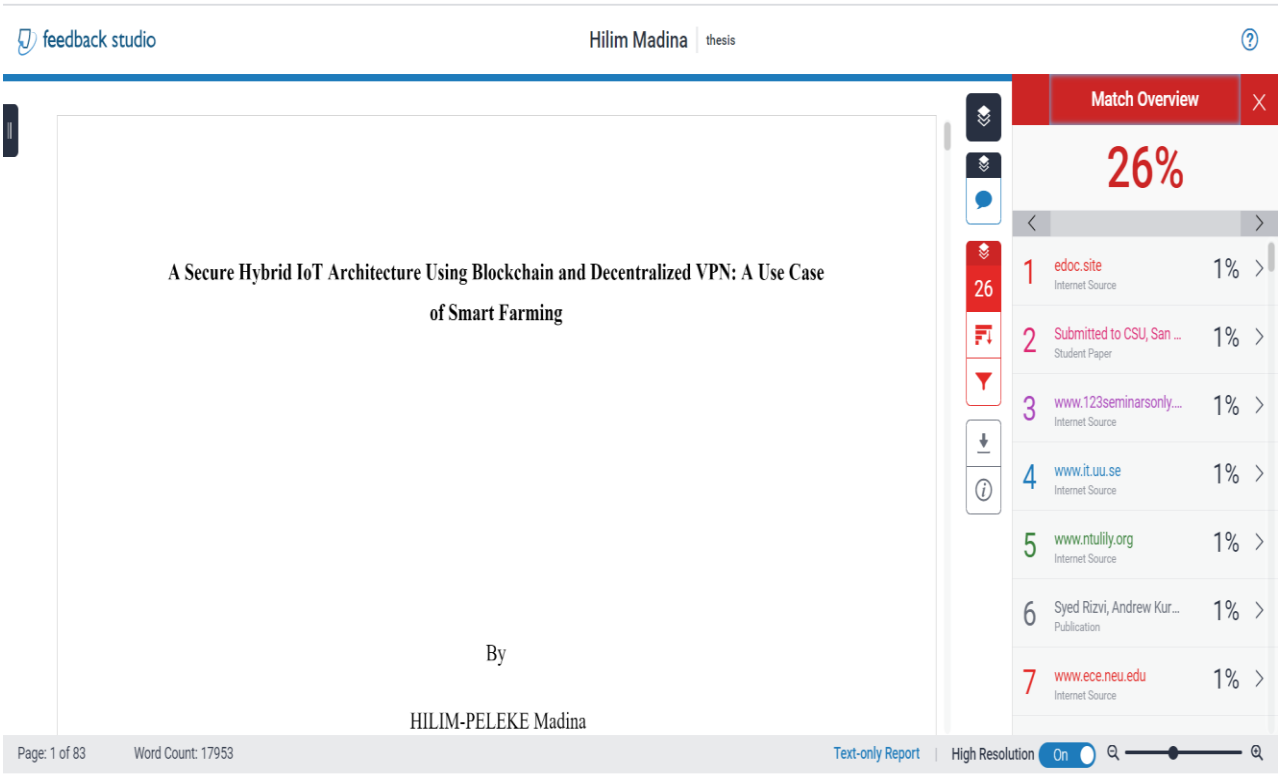
Yang, Y., Wu, L., Yin, G., Lifie, L., & Hongbin, Z. (2017). A Survey on Security and Privacy Issues in Internet-of-Things. *IEEE Internet of Things Journal*, 1250-1258.

Yurcik & Doss, W. (2001). A planning framework for implementing VPNs, 3(3), May-June.

Zhao, K., & Ge, L. (2013). *A survey on the Internet of Things security*. Proc. 9th Int. Conf. Comput. Intell. Secure. (CIS), 663-667. https://doi.org/10.1109/CIS.2013.145

Appendix

Appendix A: Originality Report



Appendix B: Ethical Approval

Strathmore UNIVERSITY

9th January 2020

Mr Hilim-Peleke Madina,
hilim.madina@strathmore.edu

Dear Mr Hilim,

RE: A Secure Hybrid IoT Architecture Using Block chain and Decentralized VPN: A Case Study of Smart Farming

This is to inform you that SU-IERC has reviewed and **approved** your above research proposal. Your application approval number is **SU-IERC0587/19** .The approval period is **9th January, 2020 to 8th January, 2021.**

This approval is subject to compliance with the following requirements:

i. Only approved documents including (informed consents, study instruments, MTA) will be used
ii. All changes including (amendments, deviations, and violations) are submitted for review and approval by SU-IERC.
iii. Death and life threatening problems and serious adverse events or unexpected adverse events whether related or unrelated to the study must be reported to SU-IERC within 72 hours of notification
iv. Any changes, anticipated or otherwise that may increase the risks or affected safety or welfare of study participants and others or affect the integrity of the research must be reported to SU-IERC within 72 hours
v. Clearance for export of biological specimens must be obtained from relevant institutions.
vi. Submission of a request for renewal of approval at least 60 days prior to expiry of the approval period. Attach a comprehensive progress report to support the renewal.
vii. Submission of an executive summary report within 90 days upon completion of the study to SU-IERC.

Prior to commencing your study, you will be expected to obtain a research license from National Commission for Science, Technology and Innovation (NACOSTI) https://oris.nacosti.go.ke and also obtain other clearances needed.

Yours sincerely,

for: Dr Virginia Gichuru,
Secretary; SU-IERC

STRATHMORE UNIVERSITY INSTITUTIONAL ETHICS REVIEW COMMITTEE (SU-IERC)
09 JAN 2020
TEL: +254 (0)703 034 000
P. O. Box 59857 - 00200
NAIROBI - KENYA

Cc: Prof Fred Were,
Chairperson; SU-IERC